

Instructors Manual
Fundamentals of Machine Learning for Predictive Data Analytics

Instructors Manual
Fundamentals of Machine Learning for Predictive Data Analytics

Algorithms, Worked Examples, and Case Studies

Second Edition

John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy

The MIT Press
Cambridge, Massachusetts
London, England

Contents

Notation	vii
I INTRODUCTION TO MACHINE LEARNING AND DATA ANALYTICS	
1 Machine Learning for Predictive Data Analytics (Exercise Solutions)	3
2 Data to Insights to Decisions (Exercise Solutions)	9
3 Data Exploration (Exercise Solutions)	27
II PREDICTIVE DATA ANALYTICS	
4 Information-Based Learning (Exercise Solutions)	71
5 Similarity-Based Learning (Exercise Solutions)	99
6 Probability-Based Learning (Exercise Solutions)	119
7 Error-Based Learning (Exercise Solutions)	141
8 Deep Learning (Exercise Solutions)	175
9 Evaluation (Exercise Solutions)	235

III BEYOND PREDICTION

10	Beyond Prediction: Unsupervised Learning (Exercise Solutions)	273
11	Beyond Prediction: Reinforcement Learning (Exercise Solutions)	291

Notation

In this section we provide a short overview of the technical notation used throughout this book.

Notational Conventions

Throughout this book we discuss the use of machine learning algorithms to train prediction models based on datasets. The following list explains the notation used to refer to different elements in a dataset. Figure 0.1^[vii] illustrates the key notation using a simple sample dataset.

ID	Name	Age	Country	Rating
1	Brian	24	Ireland	B
2	Mary	57	France	AA
3	Sinead	45	Ireland	AA
4	Paul	38	USA	A
5	Donald	62	Canada	B
6	Agnes	35	Sweden	C
7	Tim	32	USA	B

Figure 0.1

How the notation used in the book relates to the elements of a dataset.

Datasets

- \mathcal{D} denotes a dataset.
- A dataset is composed of n instances, (\mathbf{d}_1, t_1) to (\mathbf{d}_n, t_n) , where \mathbf{d} is a set of m descriptive features, and t is a target feature.
- A subset of a dataset is denoted by \mathcal{D} with a subscript to indicate the definition of the subset. For example, $\mathcal{D}_{f=l}$ represents the subset of instances from the dataset \mathcal{D} where the feature f has the value l .

Vectors of Features

- Lowercase boldface letters refer to a vector of features. For example, \mathbf{d} denotes a vector of descriptive features for an instance in a dataset, and \mathbf{q} denotes a vector of descriptive features in a query.

Instances

- Subscripts are used to index into a list of instances.
- \mathbf{x}_i refers to the i^{th} instance in a dataset.
- \mathbf{d}_i refers to the descriptive features of the i^{th} instance in a dataset.

Individual Features

- Lowercase letters represent a single feature (e.g., $f, a, b, c \dots$).
- Square brackets $[]$ are used to index into a vector of features (e.g., $\mathbf{d}[j]$ denotes the value of the j^{th} feature in the vector \mathbf{d}).
- t represents the target feature.

Individual Features in a Particular Instance

- $\mathbf{d}_i[j]$ denotes the value of the j^{th} descriptive feature of the i^{th} instance in a dataset.
- a_i refers to the value for feature a of the i^{th} instance in a dataset.
- t_i refers to the value of the target feature of the i^{th} instance in a dataset

Indexes

- Typically, i is used to index instances in a dataset, and j is used to index features in a vector.

Models

- We use \mathbb{M} to refer to a model.
- $\mathbb{M}_{\mathbf{w}}$ refers to a model \mathbb{M} parameterized by a parameter vector \mathbf{w} .
- $\mathbb{M}_{\mathbf{w}}(\mathbf{d})$ refers to the output of a model \mathbb{M} parameterized by parameters \mathbf{w} for descriptive features \mathbf{d} .

Set Size

- Vertical bars $| \cdot |$ refer to counts of occurrences (e.g., $|a = l|$ represents the number of times that $a = l$ occurs in a dataset).

Feature Names and Feature Values

- We use a specific typography when referring to a feature by name in the text (e.g., POSITION, CREDITRATING, and CLAIM AMOUNT).
- For categorical features, we use a specific typography to indicate the levels in the domain of the feature when referring to a feature by name in the text (e.g., *center*, *aa*, and *soft tissue*).

Notational Conventions for Probabilities

For clarity there are some extra notational conventions used in Chapter 6^[243] on probability.

Generic Events

- Uppercase letters denote generic events where an unspecified feature (or set of features) is assigned a value (or set of values). Typically, we use letters from the end of the alphabet—e.g., X , Y , Z —for this purpose.
- We use subscripts on uppercase letters to iterate over events. So, $\sum_i P(X_i)$ should be interpreted as summing over the set of events that are a complete assignment to the features in X (i.e., all the possible combinations of value assignments to the features in X).

Named Features

- Features explicitly named in the text are denoted by the uppercase initial letters of their names. For example, a feature named MENINGITIS is denoted by M .

Events Involving Binary Features

- Where a named feature is binary, we use the lowercase initial letter of the name of the feature to denote the event where the feature is true and the lowercase initial letter preceded by the \neg symbol to denote the event where it is false. So, m will represent the event MENINGITIS = *true*, and $\neg m$ will denote MENINGITIS = *false*.

Events Involving Non-Binary Features

- We use lowercase letters with subscripts to iterate across values in the domain of a feature.
- So $\sum_i P(m_i) = P(m) + P(\neg m)$.
- In situations where a letter, for example, X , denotes a joint event, then $\sum_i P(X_i)$ should be interpreted as summing over all the possible combinations of value assignments to the features in X .

Probability of an Event

- The probability that the feature f is equal to the value v is written $P(f = v)$.

Probability Distributions

- We use bold notation $\mathbf{P}()$ to distinguish a probability distribution from a probability mass function $P()$.
- We use the convention that the first element in a probability distribution vector is the probability for a true value. For example, the probability distribution for a binary feature, A , with a probability of 0.4 of being true would be written $\mathbf{P}(A) = \langle 0.4, 0.6 \rangle$.

Notational Conventions for Deep Learning

For clarity, some additional notational conventions are used in Chapter 8^[381] on deep learning.

Activations

- The activation (or output) of single neuron i is denoted by a_i
- The vector of activations for a layer of neurons is denoted by $\mathbf{a}^{(k)}$ where k identifies the layer.
- A matrix of activations for a layer of neurons processing a batch of examples is denoted by $\mathbf{A}^{(k)}$ where k identifies the layer.

Activation Functions

- We use the symbol φ to generically represent activation functions. In some cases we use a subscript to indicate the use of a particular activation function. For example, φ_{SM} indicates the use of a softmax function activation function, whereas φ_{ReLU} indicates the use of a rectified linear activation function.

Categorical Targets In the context of handling categorical target features (see Section 8.4.3^[463]) using a softmax function, we use the following symbols:

- We use the \star symbol to indicate the index of the true category in the distribution.
- We use \mathbf{P} to write the true probability distribution over the categories of the target; $\hat{\mathbf{P}}$ to write the distribution over the target categories that the model has predicted; and $\hat{\mathbf{P}}_\star$ to indicate the predicted probability for the true category.
- We write \mathbf{t} to indicate the one-hot encoding vector of a categorical target.
- In some of the literature on neural networks, the term **logit** is used to refer to the result of a weighted sum calculation in a neuron (i.e., the value we normally denote z). In particular, this terminology is often used in the explanation of softmax functions; therefore, in the section on handling categorical features, we switch from our normal z notation, and instead follow this logit nomenclature, using the notation \mathbf{l} to denote a vector of logits for a layer of neurons, and \mathbf{l}_i to indicate the logit for the i^{th} neuron in the layer.

Elementwise Product

- We use \odot to denote an elementwise product. This operation is sometimes called the **Hadamard product**.

Error Gradients (Deltas) δ

- We use the symbol δ to indicate the rate of change of the error of the network with respect to changes in the weighted sum calculated in a neuron. These δ values are the error gradients that are backpropagated during the backward pass of the backpropagation algorithm. We use a subscript to identify the particular neuron that the δ is associated with; for example, δ_i is the δ for neuron i and is equivalent to the term $\frac{\partial \mathcal{E}}{\partial z_i}$. In some cases we wish to refer to the vector of δ s for the neurons in a layer l ; in these cases we write $\delta^{(l)}$.

Network Error

- We use the symbol \mathcal{E} to denote the error of the network at the output layer.

Weights

- We use a lowercase w to indicate a single weight on a connection between two neurons. We use a double subscript to indicate the neurons that are connected, with the convention that the first subscript is the neuron the connection goes to, and the second subscript is the neuron the connection is from. For example, $w_{i,k}$ is the weight on the connection from neuron k to neuron i .
- We use $\Delta w_{i,k}$ to write the sum of error gradients calculated for the weight $w_{i,k}$. We sum errors in this way during batch gradient descent with which we sum over the examples in the batch; see Equation (8.30)^[416] and also in cases in which the weight is shared by a number of neurons, whether in a convolutional neural network or during backpropagation through time.
- We use a bold capital \mathbf{W} to indicate a weight matrix, and we use a superscript in brackets to indicate the layer of the network the matrix is associated with. For example, $\mathbf{W}^{(k)}$ is the weight matrix for the neurons in layer k . In an LSTM network we treat the neurons in the sigmoid and tanh layers within each gate as a separate layer of neurons, and so we write $\mathbf{W}^{(f)}$ for the weight matrix for the neurons in the forget gate, and so on for the weight matrices of the other neuron layers in the other gates. However, in a simple recurrent network we distinguish the weight matrices on the basis of whether the matrix is on the connections between the input and the hidden layer, the hidden layer and the output, or the activation memory buffer and the hidden layer. Consequently, for these matrices it is important to highlight the end of the connections the weights are applied to; we use a double subscript (similar to the subscript for a single weight), writing \mathbf{W}_{hx} for the weight matrix on the connections between the input (\mathbf{x}) and the hidden layer (\mathbf{h}).

Weighted Sums and Logits

- We use a lowercase z to represent the result of the weighted sum of the inputs in a neuron. We indicate the identity of the neuron in which the calculation occurred using a subscript. For example, z_i is the result of the weighted sum calculation carried out in neuron i . Note, however, that in the section on Handling Categorical Target features, we switch to the term *logit* to refer to the output of the weight sum in a neuron and update the notation to reflect this switch; see the previous notation section on Categorical Targets for more details.
- The vector of weighted sums for a layer of neurons is denoted by $\mathbf{z}^{(k)}$ where k identifies the layer.
- A matrix of weighted sums calculations for a layer of neurons processing a batch of examples is denoted by $\mathbf{Z}^{(k)}$ where k identifies the layer.

Notational Conventions for Reinforcement Learning

For clarity there are some extra notational conventions used in Chapter 11^[637] on reinforcement learning (this chapter also heavily uses the notation from the probability chapter).

Agents, States, and Actions

- In reinforcement learning we often describe an agent at time t taking an action, a_t , to move from its current state, s_t , to the next state, s_{t+1} .
- An agent's current state is often modeled as a random variable, S_t . We therefore often describe the probability that an agent is in a specific state, s , at time t as $P(S_t = s)$.
- Often states and actions are explicitly named, in which case we use the following formatting: *STATE* and *action*.

Transition Probabilities

- We use the \rightarrow notation to represent an agent transitioning from one state to another. Therefore, the probability of an agent moving from state s_1 to state s_2 can be written

$$P(s_1 \rightarrow s_2) = P(S_{t+1} = s_2 \mid S_t = s_1)$$

- Often we condition the probability of an agent transitioning from one state, s_1 , to another, s_2 , on the agent taking a specific action, a . We write this

$$P(s_1 \xrightarrow{a} s_2) = P(S_{t+1} = s_2 \mid S_t = s_1, A_t = a)$$

- The dynamics of an environment in which an agent transitions between states, a Markov process, can be captured in a transition matrix

$$\mathcal{P} = \begin{bmatrix} P(s_1 \rightarrow s_1) & P(s_1 \rightarrow s_2) & \dots & P(s_1 \rightarrow s_n) \\ P(s_2 \rightarrow s_1) & P(s_2 \rightarrow s_2) & \dots & P(s_2 \rightarrow s_n) \\ \vdots & \vdots & \ddots & \vdots \\ P(s_n \rightarrow s_1) & P(s_n \rightarrow s_2) & \dots & P(s_n \rightarrow s_n) \end{bmatrix}$$

- When agent decisions are allowed, leading to a Markov decision process (MDP), then the dynamics of an environment can be captured in a set of transition matrices, one for each action. For example

$$\mathcal{P}^a = \begin{bmatrix} P(s_1 \xrightarrow{a} s_1) & P(s_1 \xrightarrow{a} s_2) & \dots & P(s_1 \xrightarrow{a} s_n) \\ P(s_2 \xrightarrow{a} s_1) & P(s_2 \xrightarrow{a} s_2) & \dots & P(s_2 \xrightarrow{a} s_n) \\ \vdots & \vdots & \ddots & \vdots \\ P(s_n \xrightarrow{a} s_1) & P(s_n \xrightarrow{a} s_2) & \dots & P(s_n \xrightarrow{a} s_n) \end{bmatrix}$$

I INTRODUCTION TO MACHINE LEARNING AND DATA ANALYTICS

1 Machine Learning for Predictive Data Analytics (Exercise Solutions)

1. What is **predictive data analytics**?

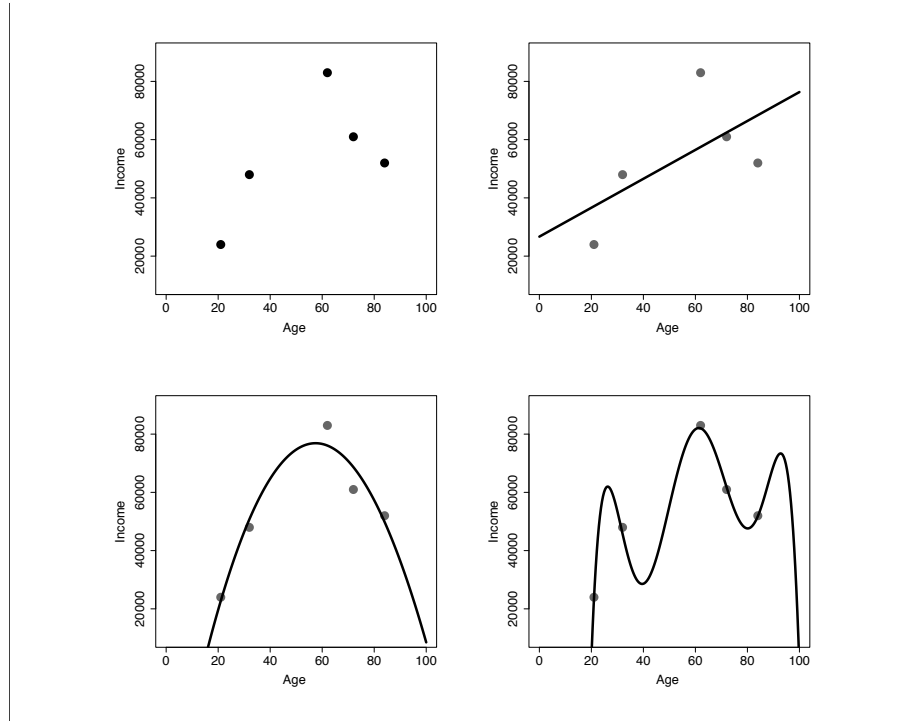
Predictive data analytics is a subfield of data analytics that focuses on building **models** that can make predictions based on insights extracted from historical data. To build these models, we use machine learning algorithms to extract patterns from datasets.

2. What is **supervised machine learning**?

Supervised machine learning techniques automatically learn the relationship between a set of **descriptive features** and a **target feature** from a set of historical **instances**. Supervised machine learning is a subfield of machine learning. Machine learning is defined as an automated process that extracts patterns from data. In predictive data analytics applications, we use **supervised machine learning** to build models that can make predictions based on patterns extracted from historical data.

3. Machine learning is often referred to as an **ill-posed problem**. What does this mean?

Machine learning algorithms essentially search through all the possible patterns that exist between a set of descriptive features and a target feature to find the best model that is consistent with the training data used. It is possible to find multiple models that are consistent with a given training set (i.e., that agree with all training instances). For this reason, inductive machine learning is referred to as an ill-posed problem, as there is typically not enough information in the training data to choose a single best model. Inductive machine learning algorithms must somehow choose one of the available models as the *best*. The images below show an example of this. All the models are somewhat consistent with the training data, but which one is best?



4. The following table lists a dataset from the credit scoring domain that we discussed in the chapter. Underneath the table we list two prediction models consistent with this dataset, **Model 1** and **Model 2**.

ID	OCCUPATION	AGE	LOAN-SALARY	
			RATIO	OUTCOME
1	industrial	39	3.40	default
2	industrial	22	4.02	default
3	professional	30	2.70	repay
4	professional	27	3.32	default
5	professional	40	2.04	repay
6	professional	50	6.95	default
7	industrial	27	3.00	repay
8	industrial	33	2.60	repay
9	industrial	30	4.50	default
10	professional	45	2.78	repay

Model 1

```

if LOAN-SALARY RATIO > 3.00 then
    OUTCOME = default
else
    OUTCOME = repay
end if

```

Model 2

```

if AGE= 50 then
    OUTCOME = default
else if AGE= 39 then
    OUTCOME = default
else if AGE= 30 and OCCUPATION = industrial then
    OUTCOME = default
else if AGE= 27 and OCCUPATION = professional then
    OUTCOME = default
else
    OUTCOME = repay
end if

```

- (a) Which of these two models do you think will generalize better to instances not contained in the dataset?

Model 1 is more likely to generalize beyond the training dataset because it is simpler and appears to be capturing a real pattern in the data.

- (b) Propose an inductive bias that would enable a machine learning algorithm to make the same preference choice that you made in Part (a).

If you are choosing between a number of models that perform equally well then prefer the simpler model over the more complex models.

- (c) Do you think that the model that you rejected in Part (a) of this question is overfitting or underfitting the data?

Model 2 is overfitting the data. All of the decision rules in this model that predict OUTCOME = *default* are specific to single instances in the dataset. Basing predictions on single instances is indicative of a model that is overfitting.

- * 5. What is meant by the term **inductive bias**?

The inductive bias of a learning algorithm

- (a) is a set of assumptions about what the true function we are trying to model looks like;
- (b) defines the set of hypotheses that a learning algorithm considers when it is learning;
- (c) guides the learning algorithm to prefer one hypothesis (i.e., the hypothesis that best fits with the assumptions) over others;

(d) is a necessary prerequisite for learning because inductive learning is an ill-posed problem.

- * 6. How do machine learning algorithms deal with the fact that machine learning is an **ill-posed problem**?

Because machine learning is ill-posed, we have to make some extra assumptions about what we think the real mapping from the descriptive features to the target feature should look like in order to find a unique solution with the data we have. The set of assumptions we make so that learning is possible is called the **inductive bias** of the learning algorithm—this is the main implication of inductive machine learning being ill-posed.

- * 7. What can go wrong when an inappropriate **inductive bias** is used?

There are two possible negative outcomes:

- If the inductive bias of a learning algorithm constrains the search to consider only simple models, we may exclude any models that successfully capture the relationship in the data from this search. In other words, the true model is **unrealizable** by the algorithm. In this case **underfitting** occurs.
- If the inductive bias of the learning algorithm allows the search to consider overly complex models, the algorithm may home in on irrelevant factors in the training set. In other words, the model will **overfit** the training data.

- * 8. It is often said that 80% of the work done on predictive data analytics projects is done in the Business Understanding, Data Understanding, and Data Preparation phases of **CRISP-DM**, and just 20% is spent on the Modeling, Evaluation, and Deployment phases. Why do you think this would be the case?

There is an old adage: “*garbage in, garbage out.*” No matter what machine learning algorithms are used or how they are tuned, it is not possible to build good models from bad data. Because of this, it is crucial that sufficient time be spent gathering, analyzing, and preparing data before any modeling effort starts. This work is primarily done in the Data Understanding and Data Preparation phases of the CRISP-DM process.

The Business Understanding and Data Understanding phases of the CRISP-DM process also involve a huge amount of communication between the business domain experts and the predictive data analytics and machine learning experts in order to fully define the problem that will be solved. This communication is in-

evitably time consuming. It is, however, crucial so that the final models delivered are actually appropriate for the business.

- * 9. The following table lists a dataset of five individuals described via a set of stroke risk factors and their probability of suffering a stroke in the next five years. This dataset has been prepared by an analytics team who are developing a model as a decision support tool for doctors.¹ The goal of the model is to classify individuals into groups on the basis of their risk of suffering a stroke **STROKE RISK**. In this dataset there are three categories of risk: *low*, *medium*, and *high*. All the descriptive features are Boolean, taking two levels: *true* or *false*.

ID	HIGH BLOOD PRESSURE	SMOKER	DIABETES	HEART DISEASE	STROKE RISK
1	true	false	true	true	high
2	true	true	true	true	high
3	true	false	false	true	medium
4	false	false	false	false	low
5	true	true	true	false	high

- (a) How many possible models exist for the scenario described by the features in this dataset?

We begin by calculating the number of combinations of descriptive features. Their four boolean features and so there are $2^4 = 16$ possible combinations of levels. The target feature also takes three levels which gives us a total number of possible mappings from descriptive feature combinations to target feature level of: $3^{16} = 43046721$

- (b) How many of these potential models would be consistent with this sample of data?

There are $16 - 5 = 11$ combinations of descriptive feature values that are not covered by the dataset. And therefore there are $3^{11} = 177147$ potential models that are consistent with this dataset.

- * 10. You are using U.S. census data to build a prediction model. On inspecting the data you notice that the **RACE** feature has a higher proportion of the category *White* than

1. Due to space limitations, this dataset covers only a sample of the risk factors for stroke. There are, for example, a number of non-modifiable risk factors such as age and gender. There is also an array of modifiable risk factors, including alcohol and drug use, unhealthy diet, stress and depression, and lack of physical exercise. Furthermore, this dataset is not based on precise measurements of stroke risk. For more information on stroke and risk factors related to stroke, please see the National Heart, Lung, and Blood Institute on Stroke: <https://www.nhlbi.nih.gov/health-topics/stroke>.

you expected. Why do you think this might be?

The U.S. census is collected via a postal, consequently the census can suffer from non-responders that introduce a **sample bias**. There are a large number of reasons why an individual may not return their census form. A number of these are skewed towards education levels, socio-economic background, ethnicity, and so on. In fact, it is estimated that in the U.S. census in 2000 over 4.5 million people were missing from the census. Importantly, these missing people were much more likely to be from low-income Black and Hispanic backgrounds. The census bureau works hard on removing this sample bias from the census, but it is always present. And, so care should be taken with respect to sample bias, even when using large census data, that minority groups are not underrepresented in the data.

- * 11. Why might a prediction model that has very high accuracy on a dataset not generalize well after it is deployed?

There are a range of factors that can contribute to this happening, but two important ones that we discuss in the chapter is the model overfitting to noise in the data, and sampling bias.

It is possible for a model to be very accurate on one sample of data from a domain and not be accurate on another sample of data taken from the same domain. This is because the model may overfit to the sample of data it is trained on. What this means is that the model has memorised the training data so closely that it has modelled the noise in the sample of data it was trained on, and so it will likely make incorrect predictions on examples that are similar to the noisy examples in the dataset.

Another problem that can lead to poor generalisation is if the data used for training and testing the model suffers from sample bias. In this case, even if the model does not overfit the data, the model will not generalize well because it is trained on data that is not representative of the true distributions in the general population.

2 Data to Insights to Decisions (Exercise Solutions)

1. An online movie streaming company has a business problem of growing **customer churn**—subscription customers canceling their subscriptions to join a competitor. Create a list of ways in which predictive data analytics could be used to help address this business problem. For each proposed approach, describe the predictive model that will be built, how the model will be used by the business, and how using the model will help address the original business problem.

- **[Churn prediction]** A model could be built that predicts the **propensity**, or likelihood, that a customer will cancel their subscription in the next three months. This model could be run every month to identify the customers to whom the business should offer some kind of bonus to entice them to stay. The analytics problem in this case is to build a model that accurately predicts the likelihood of customers to **churn**.
- **[Churn explanation]** By building a model that predicts the propensity of customers to cancel their subscriptions, the analytics practitioner could identify the factors that correlate strongly with customers choosing to leave the service. The business could then use this information to change its offerings so as to retain more customers. The analytics problem in this case would be to identify a small set of features that describe the company's offerings that are important in building an accurate model that predicts the likelihood of individual customers to churn.
- **[Next-best-offer prediction]** The analytics practitioner could build a **next-best-offer** model that predicts the likely effectiveness of different bonuses that could be offered to customers to entice them to stay with the service. The company could then run this model whenever contacting a customer believed likely to leave the service and identify the least expensive bonus that is likely to entice the customer to remain a subscriber to the service. The analytics

problem in this case would be to build the most accurate next-best-offer model possible.

- **[Enjoyment prediction]** Presumably, if the company offered a better service to its customers, fewer customers would churn. The analytics practitioner could build a model that predicted the likelihood that a customer would enjoy a particular movie. The company could then put in place a service that personalized recommendations of new releases for its customers and thus reduce churn by enticing customers to stay with the service by offering them a better product. The analytics problem in this case would be to build a model that predicted, as accurately as possible, how much a customer would enjoy a given movie.

2. A national revenue commission performs audits on public companies to find and fine tax defaulters. To perform an audit, a tax inspector visits a company and spends a number of days scrutinizing the company's accounts. Because it takes so long and relies on experienced, expert tax inspectors, performing an audit is an expensive exercise. The revenue commission currently selects companies for audit at random. When an audit reveals that a company is complying with all tax requirements, there is a sense that the time spent performing the audit was wasted, and more important, that another business who is not tax compliant has been spared an investigation. The revenue commissioner would like to solve this problem by targeting audits at companies who are likely to be in breach of tax regulations, rather than selecting companies for audit at random. In this way the revenue commission hopes to maximize the yield from the audits that it performs.

To help with **situational fluency** for this scenario, here is a brief outline of how companies interact with the revenue commission. When a company is formed, it registers with the company registrations office. Information provided at registration includes the type of industry the company is involved in, details of the directors of the company, and where the company is located. Once a company has been registered, it must provide a tax return at the end of every financial year. This includes all financial details of the company's operations during the year and is the basis of calculating the tax liability of a company. Public companies also must file public documents every year that outline how they have been performing, details of any changes in directorship, and so on.

- (a) Propose two ways in which predictive data analytics could be used to help address this business problem.¹ For each proposed approach, describe the predictive

1. Revenue commissioners around the world use predictive data analytics techniques to keep their processes as efficient as possible. ? is a good example.

model that will be built, how the model will be used by the business, and how using the model will help address the original business problem.

One way in which we could help to address this business problem using predictive data analytics would be to build a model that would predict the likely return from auditing a business—that is, how much unpaid tax an audit would be likely to recoup. The commission could use this model to periodically make a prediction about every company on its register. These predictions could then be ordered from highest to lowest, and the companies with the highest predicted returns could be selected for audit. By targeting audits this way, rather than through random selection, the revenue commissioners should be able to avoid wasting time on audits that lead to no return.

Another, related way in which we could help to address this business problem using predictive data analytics would be to build a model that would predict the likelihood that a company is engaged in some kind of tax fraud. The revenue commission could use this model to periodically make a prediction about every company on its register. These predictions could then be ordered from highest to lowest predicted likelihood, and the companies with the highest predicted propensity could be selected for audit. By targeting audits at companies likely to be engaged in fraud, rather than through random selection, the revenue commissioners should be able to avoid wasting time on audits that lead to no return.

- (b) For each analytics solution you have proposed for the revenue commission, outline the type of data that would be required.

To build a model that predicts the likely yield from performing an audit, the following data resources would be required:

- Basic company details such as industry, age, and location
- Historical details of tax returns filed by each company
- Historical details of public statements issued by each company
- Details of all previous audits carried out, including the outcomes

To build a model that predicts the propensity of a company to commit fraud, the following data resources would be required:

- Basic company details such as industry, age, and location
- Historical details of tax returns filed by each company
- Historical details of public statements issued by each company

- Details of all previous audits carried out
- Details of every company the commission has found to be fraudulent

- (c) For each analytics solution you have proposed, outline the capacity that the revenue commission would need in order to utilize the analytics-based insight that your solution would provide.

Utilizing the predictions of expected audit yield made by a model would be quite easy. The revenue commission already have a process in place through which they randomly select companies for audit. This process would simply be replaced by the new analytics-driven process. Because of this, the commission would require little extra capacity in order to take advantage of this system.

Similarly, utilizing the predictions of fraud likelihood made by a model would be quite easy. The revenue commission already have a process in place through which they randomly select companies for audit. This process would simply be replaced by the new analytics-driven process. Because of this, the commission would require little extra capacity in order to take advantage of this system.

3. The table below shows a sample of a larger dataset containing details of policyholders at an insurance company. The descriptive features included in the table describe each policy holders' ID, occupation, gender, age, the value of their car, the type of insurance policy they hold, and their preferred contact channel.

ID	OCCUPATION	GENDER	AGE	MOTOR VALUE	POLICY TYPE	PREF CHANNEL
1	lab tech	female	43	42,632	planC	sms
2	farmhand	female	57	22,096	planA	phone
3	biophysicist	male	21	27,221	planA	phone
4	sheriff	female	47	21,460	planB	phone
5	painter	male	55	13,976	planC	phone
6	manager	male	19	4,866	planA	email
7	geologist	male	51	12,759	planC	phone
8	messenger	male	49	15,672	planB	phone
9	nurse	female	18	16,399	planC	sms
10	fire inspector	male	47	14,767	planC	email

- (a) State whether each descriptive feature contains numeric, interval, ordinal, categorical, binary, or textual data.

ID	Ordinal	MOTORVALUE	Numeric
OCCUPATION	Textual	POLICYTYPE	Ordinal
GENDER	Categorical	AGE	Numeric
PREFCHANNEL	Categorical		

- (b) How many levels does each categorical, binary, or ordinal feature have?

ID	10 are present in the sample, but there is likely to be 1 per customer
GENDER	2 (<i>male, female</i>)
POLICYTYPE	3 (<i>planA, planB, planC</i>)
PREFCHANNEL	3 (<i>sms, phone, email</i>)

4. Select one of the predictive analytics models that you proposed in your answer to Question 2 about the revenue commission for exploration of the design of its **analytics base table (ABT)**.

For the answers below, the audit yield prediction model is used.

- (a) What is the prediction subject for the model that will be trained using this ABT?

For the audit yield prediction model, the prediction subject is a company. We are assessing the likelihood that an audit performed on a company will yield a return, so it is the company that we are interested in assessing.

- (b) Describe the domain concepts for this ABT.

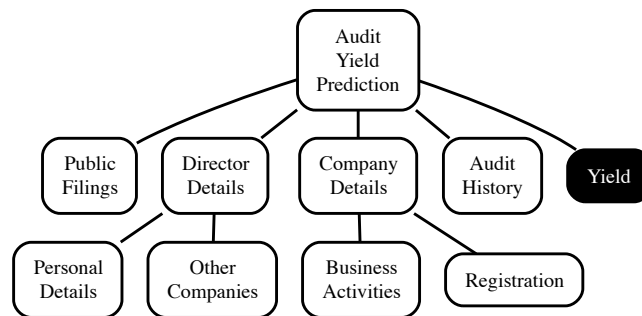
The key domain concepts for this ABT are

- *Company Details*: The details of the company itself. These could be broken down into details about the activities of the company, such as the locations it serves and the type of business activities it performs, *Business Activities*, and information provided at the time of registration, *Registration*.
- *Public Filings*: There is a wealth of information in the public documents that companies must file, and this should be included in the data used to train this model.

- *Director Details*: Company activities are heavily tied to the activities of their directors. This domain concept might be further split into details about the directors themselves, *Personal Details*, and details about other companies that the directors have links to, *Other Companies*.
- *Audit History*: Companies are often audited multiple times, and it is likely that details from previous audits would be useful in predicting the likely outcome of future audits.
- *Yield*: It is important not to forget the target feature. This would come from some measure of the yield of a previous audit.

(c) Draw a domain concept diagram for the ABT.

The following is an example domain concept diagram for the the audit yield prediction model.



(d) Are there likely to be any legal issues associated with the domain concepts you have included?

The legal issues associated with a set of domain concepts depend primarily on the data protection law within the jurisdiction within which we are working. Revenue commissions are usually given special status within data protection law and allowed access to data that other agencies would not be given. For the domain concepts given above, the one most likely to cause trouble is the *Director Details* concept. It is likely that there would be issues associated with using personal details of a company director to make decisions about a company.

- * 5. Although their sales are reasonable, an online fashion retailer is struggling to generate the volume of sales that they had originally hoped for when launching their site. List a number of ways in which predictive data analytics could be used to help address this business problem. For each proposed approach, describe the predictive model that will

be built, how the model will be used by the business, and how using the model will help address the original business problem.

- **[Channel prediction]** Personalised marketing is usually a good way to drive sales. One of the ways in which this can be done is by tailoring the channel through which a business communicates with customers. Typical channels might be email, telephone call, SMS message, Twitter direct message or Facebook message. A predictive model could be built to predict the channel that a customer would most likely respond to out of a set of potential channels. Every time the business needs to communicate with a customer they could use the model to predict which channel the customer would most likely respond to and use that channel to communicate with the customer.
- **[Cross-sell prediction]** One way to drive an increase in sales would be to encourage customers who actively buy products from some departments in the online store to buy products from departments from which they have not bought a product before. This is referred to as cross-selling, and a **cross-sell model** would predict the likelihood that an active customer would buy a product from a particular department. The business could use this model to predict the likelihood that each active customer would make a purchase from a new department and contact those customer who are likely to make a purchase with an appropriate marketing message. By encouraging customers to make purchases from other departments the business should be able to increase sales.
- **[Upsell prediction]** Upselling is the practice of offering a customer the opportunity to buy a more expensive product once they have made the decision to buy something. The danger with making upsell offers is that they can annoy customers and so can jeopardise the original sale. An **upsell model** can predict the likelihood that a customer will respond positively. The business could use the output of the upsell model to only make upsell offers to customers that are likely to respond to them. In this way the business should be able to increase sales through upselling without risking any loss of sales through inappropriate upsell offers.
- **[Reactivation prediction]** Lapsed customers are a good prospect for increasing sales. The retail store could build a model that predicts the propensity of a lapsed customer to reactivate in response to a special offer, thus increasing sales. The business could use this model to periodically make predictions for all lapsed customers and actively pursue those that are most likely to reactivate.

- * 6. An oil exploration company is struggling to cope with the number of exploratory sites that they need to drill in order to find locations for viable oil wells. There are many potential sites that geologists at the company have identified, but undertaking exploratory

drilling at these sites is very expensive. If the company could increase the percentage of sites at which they perform exploratory drilling that actually lead to finding locations for viable wells, they could save a huge amount of money.

Currently geologists at the company identify potential drilling sites by manually examining information from a variety of different sources. These include ordinance survey maps, aerial photographs, characteristics of rock and soil samples taken from potential sites, and measurements from sensitive gravitational and seismic instruments.

- (a) Propose two ways in which predictive data analytics could be used to help address the problem that the oil exploration company is facing. For each proposed approach, describe the predictive model that will be built, how the model will be used by the company, and how using the model will help address the original problem.

The most obvious way to help this company would be to build a predictive model that could analyze the characteristics of a potential drilling site and predict whether that site is likely to be a viable site for an oil well or not. The company could use such a model to assess each potential drilling site identified by its geologists. By doing this, the company could, hopefully, filter out a large number of unviable sites and reduce the number of exploratory drills they perform.

This simple model could be implemented, and subsequently used, in quite different ways depending on the data sources used as the basis of descriptive features. For example, if the features used were based only on ordinance survey maps and aerial photographs, the model could be deployed over vast tracts of land to identify potential drilling sites. On the other hand, if descriptive features based on soil or rock samples were used, then these would need to be manually collected and analyzed before the model could be applied.

As well as predicting the likelihood that a potential site would be viable for an oil well, predictive models could also be built to make other useful predictions. For example, it is likely that geologists studying aerial photographs spend a significant amount of time identifying different types of land in these images that are indicative of different rock and soil types. A predictive model could be built to perform this task—identifying the land type of sections of aerial photographs. A model like this could be used to reduce the amount of work that the geologists at the company need to do and, hopefully, allow them to make better assessments of the likely usefulness of different drilling sites. This would address the original business problem.

Another approach would be to build a model that would predict the likely yield of a potential drilling site. Again, the company could use this to assess each potential site and use the more detailed prediction on yield to make decisions about which sites were worth pursuing. By using the details of yield, the company could add costs and likely profits into their decisions about which sites to drill. Rather than making this a continuous prediction problem (predicting an actual number of barrels of oil that are likely to be extracted from a site), it might be more interesting to categorize sites as small, medium, and large and predict these categories instead. This solution would again allow the company to target its exploratory drilling at those sites that are most likely to yield large returns.

- (b) For each analytics solution you have proposed, outline the type of data that would be required.

For the model that would predict the viability of a drilling site, details of historical potential sites that have been explored would be required. These details would most likely include ordinance survey maps, aerial photographs, characteristics of rock and soil samples taken from the potential sites, and measurements from any instruments used. The outcome of the exploratory drilling that took place would also be key so that a target feature could be generated.

There are a couple of things worth mentioning about these data sources. First, very few raw descriptive features would be available from the types of sources just listed. For example, aerial photos are not usable as descriptive features in their raw form, but rather need significant processing to generate features from them. For a specific area in an aerial photo, we would likely use the average color values, the mix of color values, and other similar features that could be extracted from the raw image data. Map data would require similar processing.

The other key issues worth mentioning are that oil drilling data is likely to cover a significant time horizon—the company might have details of past drilling stretching back ten or twenty years. The characteristics of many of the data sources mentioned above—for example, aerial photos and maps—are likely to have changed significantly over this time. For example, the resolution of aerial photos has changed by orders of magnitude over the past ten years. Approaches would need to be put in place to align data from different periods.

For the model that predicts land types, the key data sources required would be examples of aerial photos and maps with land types marked on them. These would be necessary for training the prediction model.

Finally, for the yield prediction model, information on the final yield from drilling sites would be required as well as the types of data listed above.

- (c) For each analytics solution you have proposed, outline the capacity that would be needed in order to utilize the analytics-based insight that your solution would provide.

For the models that predict the viability or yield of a potential drilling site, very little extra capacity would be required within the company. This type of analysis is already being performed manually in the company, so it would only need to be extended to include the outputs from models.

The proposed model for categorizing land types would run separately from the manual assessment the company geologists are already doing, so some extra capacity to run this model and feed its results into the existing manual process would be required.

- * 7. Select one of the predictive analytics models that you proposed in your answer to the previous question about the oil exploration company for exploration of the design of its **analytics base table**.

In the answers below, we use the model proposed for assessing the viability of a potential drilling site using as much data as possible.

- (a) What is the prediction subject for the model that will be trained using this ABT?

The prediction subject for this example is a potential drilling site.

- (b) Describe the domain concepts for this ABT.

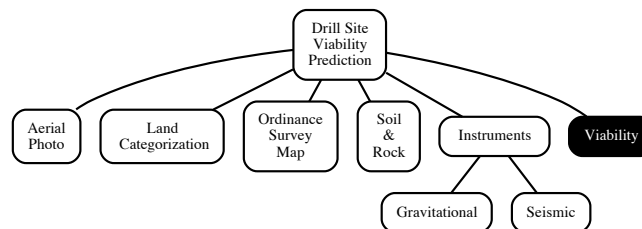
The key domain concepts for this ABT are

- *Aerial Photo*: It should be possible for features to be extracted from the aerial photos of the site being considered.
- *Land Categorization*: If the land-type categorization model proposed in the answer to the previous question were built, then its output could be used as a source of descriptive features for this model.

- *Ordinance Survey Map*: Ordinance survey maps provide a wealth of information, including the elevation of a location and details of any nearby infrastructure.
- *Soil and Rock*: The analysis of soil and rock samples from a potential drilling site should provide potential for a large number of descriptive features to be derived.
- *Instruments*: The output from any analysis of a location using specialized instruments should almost certainly be included in an ABT. It is likely that for this application scenario, the output from specialized instruments could be split into *Gravitational* and *Seismic*.
- *Viability*: It is important not to forget the target feature.

(c) Draw a domain concept diagram for the ABT.

The following is an example domain concept diagram for the the drill site viability prediction model.



(d) Are there likely to be any legal issues associated with the domain concepts you have included?

It is unlikely that any particular legal issues would arise from the data used for this scenario.

- * 8. The following table shows a sample of a larger dataset that has been collected to build a model to predict whether newly released movies will be a hit or not.² The dataset contains details of movies that have already been released, including their title, running time, rating, genre, year of release, and whether or not the actor Kevin Costner had a starring role. An indicator of whether they were a success—a *hit*—or not—a *miss*—based on box office returns compared to budget is also included.

2. This dataset has been artificially created for this book, but machine learning has been used for this task, for example, by ?.

ID	TITLE	LENGTH	RATING	GENRE	COSTNER	YEAR	HIT
1	Jaws	124	PG	action	false	1975	hit
2	Waterworld	135	PG-13	sci-fi	true	1995	miss
3	Hudson Hawk	100	R	adventure	false	1991	miss
4	Downfall	156	R	drama	false	2004	hit
5	The Postman	177	R	action	true	1997	miss
6	Toy Story	81	G	children's	false	1995	hit
7	Field of Dreams	107	G	drama	true	1989	hit
8	Amelie	122	R	comedy	false	2001	hit

- (a) State whether each descriptive feature contains numeric, interval, ordinal, categorical, binary, or textual data.

ID	Ordinal	GENRE	Categorical
TITLE	Textual	COSTNER	Binary
LENGTH	Numeric	YEAR	Interval
RATING	Ordinal	HIT	Binary

- (b) How many levels does each categorical, binary, or ordinal feature have?

ID	8 are present in the sample, but there is likely to be 1 per movie
RATING	4 at least (<i>G, PG, PG-13, R, ...</i>)
GENRE	5 at least (<i>action, adventure, children's, comedy, drama, sci-fi, ...</i>)
COSTNER	2 (<i>true, false</i>)
HIT	2 (<i>hit, miss</i>)

- * 9. The management of a large hospital group are concerned about the readmission rate for patients who are hospitalized with problems relating to **diabetes**. An analysis of historical data suggests that the rate of readmission within 30 days of being discharged for patients who were hospitalized for complications relating to diabetes is approximately 20%, compared to an overall average for all patients of approximately 11%. Sometimes patients are readmitted for a recurrence of the same problem for which they were originally hospitalized, but at other times readmission is for different problems. Hospital management are concerned that the cause of the high readmittance rate for diabetes patients might be that they are discharged too early or that their care plans while in the hospital are not addressing all their needs.

Hospital management would like to explore the use of predictive analytics to address this issue.³ They would like to reduce the readmittance rate of diabetes patients, while at the same time not keeping patients in the hospital longer than they need to be.

- (a) Propose two ways in which predictive data analytics could be used to help address this problem for the hospital group. For each proposed approach, describe the predictive model that will be built, how the model will be used by the business, and how using the model will help address the original problem.

One way in which we could help to address this business problem using predictive data analytics would be to build a predictive model that could be used when a patient is being considered for discharge to predict the likelihood that they would be readmitted to the hospital within 30 days of discharge. Clinicians could take the outputs of this model into account when making decisions about whether or not to discharge patients and not discharge patients with high readmittance risk. By reducing the number of patients discharged too early and the readmittance rate should be reduced. Another way the outputs of this model could be used would be to provide increased follow-on care after discharge to patients predicted to be at high-risk of readmission. This might help treat problems before they became so severe as to require hospitalisation.

Another way that predictive analytics could be used to address this problem would be to perform *co-morbidity* prediction. A co-morbidity is a medical condition that co-occurs with another one. For example, cardiovascular disease, hypertension, and kidney disease commonly co-occur with diabetes. The situation described above noted that when patients are readmitted it is often not for the same problem as before, and co-morbidities could be a key driver of this. Often large hospitals, or hospital groups, have specialised units that treat specific complications but are not as conscious of other complications. When the decision to discharge a patient is being made a predictive models could be used to predict the likelihood that a patient will suffer from the common co-morbidities associated with diabetes. If any of these predictions indicated a high likelihood then patient could be kept in hospital, sent for further screening for the co-morbidity, or directly treated for the co-morbidity. This could greatly reduce the number of patients re-admitted to the hospital.

3. There are many applications of predictive analytics in healthcare, and predicting readmittance rates for diabetes patients, as well as patients suffering from other issues, is well studied, for example, by ? and ?.

- (b) For each analytics solution you have proposed for the hospital group, outline the type of data that would be required.

To build a model that predicts the likelihood of a patient being re-admitted to hospital within 30 days of discharge a large historical dataset of patient details would be required, probably covering a period of multiple years. The following data resources would most likely need to be included in this:

- Basic patient details such as age, sex, address, occupation, ...
- Basic details of the patient's current hospital visit such as the reason for admission, length of stay, the type of room the patient stayed in, ...
- Details of the treatment the patient received during the current visit such as procedures undertaken, drugs administered, ...
- Details of diagnostics performed during the patient's current stay such as blood tests, urine tests, heart rate monitoring, temperature monitoring, ...
- Typically clinicians write a natural language opinion when a patient is being discharged, referred to as a discharge report, describing their opinions of the patient's condition.
- Medical history of the patient covering previous hospital visits, significant procedures, ...
- Historical admission data covering at least multiple years to allow the 30 day readmission target feature to be created.

To build the a model to predict likely co-morbidities for diabetes patients a large historical dataset of patient details would be required, probably covering a period of multiple years. The following data resources would most likely need to be included in this:

- Basic patient details such as age, sex, address, occupation, ...
- Basic details of the patient's current hospital visit such as the reason for admission, length of stay, the type of room the patient stayed in, ...
- Details of diagnostics performed during the patient's current stay such as blood tests, urine tests, heart rate monitoring, temperature monitoring, ...
- Typically clinicians write a natural language opinion when a patient is being discharged, referred to as a discharge report, describing their opinions of the patient's condition.

- Medical history of the patient covering previous hospital visits, significant procedures, . . .
- Historical admission data covering at least multiple years to allow the 30 day readmission target feature to be created.

- (c) For each analytics solution you have proposed, outline the capacity that the hospital would need in order to use the analytics-based insight that your solution would provide.

Utilizing the predictions of the model predicting the likelihood that a patient would be readmitted could be quite simple. In one scenario when clinicians are making the decision to discharge, then the output of the model could be used as an input to this decision. This would be quite straight-forward as, assuming predictions could be generated in a timely manner, the model prediction would be one more piece of information added to an existing decision making process. Alternatively, levels of after-care provided to patients could be decided based on the predicted likelihood of readmission. This would assume different levels of after-care exist and that they can be deployed to patients as required.

The co-morbidity prediction model could be triggered after a discharge decision has been made and, for complications predicted to have a high likelihood extra care actions could be planned. This would require mobilising multiple parts of the hospital group which could be a challenge.

- * 10. Select one of the predictive analytics models that you proposed in your answer to the previous question about the readmission of diabetes patients for exploration of the design of its **analytics base table**.

In the answers below, we use the model proposed for predicting the likelihood of readmission within 30 days of discharge.

- (a) What is the prediction subject for the model that will be trained using this ABT?

The prediction subject for this example is a patient.

- (b) Describe the domain concepts for this ABT.

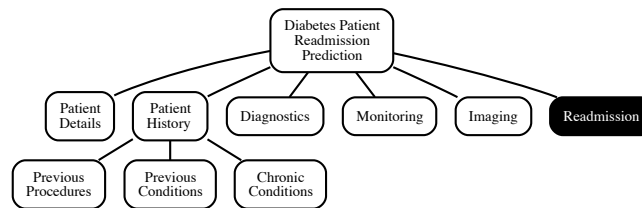
The key domain concepts for this ABT might be

- *Patient Details*: Basic details of a patient including age, sex, . . .

- *Patient History*: Detailed medical history for the patient including chronic conditions, previous procedures, previous conditions, . . .
- *Diagnostics*: Patients in hospital typically undergo multiple diagnostic tests (for example blood tests, urine tests etc).
- *Monitoring*: While in hospital clinicians typically continuously monitor key indicators about patients, for example heart rate, temperature, blood pressure,
- *Imaging*: Many patients undergo extensive imaging while hospitalised which generate useful data, for example x-rays, CT scans, . . .
- *Readmission*: It is important not to forget the target feature.

(c) Draw a domain concept diagram for the ABT.

The following is an example domain concept diagram for the the drill site viability prediction model.



(d) Are there likely to be any legal issues associated with the domain concepts you have included?

Medical applications of predictive analytics often have associated legal issues. In this case the following might need to be considered:

- Almost all medical data is by its nature *personal data*, and even more than that *personal sensitive data*. Therefore it is likely to be securely stored and access is likely to be protected. Depending on the jurisdiction in which this work is taking place explicit consent from the patients may be required in order to use their data for predictive modelling.
- If different care is going to be offered to patients based on the predictions made by the model designed, then *anti-discrimination law* needs to be carefully considered. Is it possible that a model would systematically offer what might be considered *better* treatment to some patients based on characteristics like age, ethnicity, or gender? A machine learning algorithm might select these as especially predictive features within a

prediction model and lead to potential discrimination. This would need to be carefully considered.

- When using data that covers a long time horizon and that has been collected for purposes other than building machine learning models then the *purpose specification* and *use limitation* principles of data protection law need to be taken into account. Subjects need to be informed of the purposes for which their data will be used and the data should only be used for these purposes. Using older data for the purpose of building machine learning models may not be covered under the uses described to subjects at the time the data was collected. This might mean that it cannot be used without obtaining new consent from subjects, which can be an onerous task.

3 Data Exploration (Exercise Solutions)

1. The table below shows the age of each employee at a cardboard box factory.

ID	1	2	3	4	5	6	7	8	9	10
AGE	51	39	34	27	23	43	41	55	24	25

ID	11	12	13	14	15	16	17	18	19	20
AGE	38	17	21	37	35	38	31	24	35	33

Based on this data, calculate the following **summary statistics** for the AGE feature:

- (a) Minimum, maximum, and range

By simply reading through the values we can tell that the minimum value for the AGE feature is: 17.

By simply reading through the values we can tell that the maximum value for the AGE feature is: 55.

The range is simply the difference between the highest and lowest value:

$$\begin{aligned} \text{range}(\text{AGE}) &= (55 - 17) \\ &= 38 \end{aligned}$$

- (b) Mean and median

We can calculate the mean of the AGE feature as follows:

$$\begin{aligned}\overline{\text{AGE}} &= \frac{1}{20} \times (51 + 39 + 34 + 27 + 23 + 43 + 41 + 55 + 24 + 25 \\ &\quad + 38 + 17 + 21 + 37 + 35 + 38 + 31 + 24 + 35 + 33) \\ &= \frac{671}{20} = 33.55\end{aligned}$$

To calculate the median of the AGE feature we first have to arrange the AGE values in ascending order:

17, 21, 23, 24, 24, 25, 27, 31, 33, 34, 35, 35, 37, 38, 38, 39, 41, 43, 51, 55

Because there are an even number of instances in this small dataset we take the mid-point of the middle two values as the median. These are 34 and 35 and so the median is calculated as:

$$\begin{aligned}\text{median}(\text{AGE}) &= (34 + 35)/2 \\ &= 34.5\end{aligned}$$

(c) Variance and standard deviation

To calculate the variance we first sum the squared differences between each value for AGE and the mean of AGE. This table illustrates this:

ID	AGE	$(\text{AGE} - \overline{\text{AGE}})$	$(\text{AGE} - \overline{\text{AGE}})^2$
1	51	17.45	304.50
2	39	5.45	29.70
3	34	0.45	0.20
4	27	-6.55	42.90
5	23	-10.55	111.30
6	43	9.45	89.30
7	41	7.45	55.50
8	55	21.45	460.10
9	24	-9.55	91.20
10	25	-8.55	73.10
11	38	4.45	19.80
12	17	-16.55	273.90
13	21	-12.55	157.50
14	37	3.45	11.90
15	35	1.45	2.10
16	38	4.45	19.80
17	31	-2.55	6.50
18	24	-9.55	91.20
19	35	1.45	2.10
20	33	-0.55	0.30
Sum			1,842.95

Based on the sum of squared differences value of 1,842.95 we can calculate the variance as:

$$\begin{aligned} \text{var}(\text{AGE}) &= \frac{1,842.95}{20 - 1} \\ &= 96.9974 \end{aligned}$$

The standard deviation is calculated as the square root of the variance, so:

$$\begin{aligned} \text{sd}(\text{AGE}) &= \sqrt{\text{var}(\text{AGE})} \\ &= 9.8487 \end{aligned}$$

(d) 1st quartile (25th percentile) and 3rd quartile (75th percentile)

To calculate any percentile of the AGE feature we first have to arrange the AGE values in ascending order:

17, 21, 23, 24, 24, 25, 27, 31, 33, 34, 35, 35, 37, 38, 38, 39, 41, 43, 51, 55

We then calculate the index for the percentile value as:

$$\text{index} = n \times \frac{i}{100}$$

where n is the number of instances in the dataset and i is the percentile we would like to calculate. For the 25th percentile:

$$\begin{aligned} \text{index} &= 20 \times \frac{25}{100} \\ &= 5 \end{aligned}$$

Because this is a whole number we can use this directly and so the 25th percentile is at index 5 in the ordered dataset and is 24.

For the 75th percentile:

$$\begin{aligned} \text{index} &= 20 \times \frac{75}{100} \\ &= 15 \end{aligned}$$

Because this is a whole number we can use this directly and so the 75th percentile is at index 15 in the ordered dataset and is 38.

(e) Inter-quartile range

To calculate the inter-quartile range we subtract the lower quartile value from the upper quartile value:

$$\begin{aligned} IQR(\text{AGE}) &= (38 - 24) \\ &= 14 \end{aligned}$$

(f) 12th percentile

We can use the ordered list of values above once more. For the 12th percentile:

$$\begin{aligned} index &= 20 \times \frac{12}{100} \\ &= 2.4 \end{aligned}$$

Because index is not a whole number we have to calculate the percentile as follows:

$$i^{\text{th}} \text{percentile} = (1 - index_f) \times a_{index_w} + index_f \times a_{index_w+1}$$

Because $index = 2.4$, $index_w = 2$ and $index_f = 0.4$. Using $index_w = 2$ we can look up AGE_2 to be 21 AGE_{2+1} to be 23. Using this we can calculate the 12th percentile as:

$$\begin{aligned} 12^{\text{th}} \text{percentile of AGE} &= (1 - 0.4) \times AGE_2 + 0.4 \times AGE_{2+1} \\ &= 0.6 \times 21 + 0.4 \times 23 \\ &= 21.8 \end{aligned}$$

2. The table below shows the policy type held by customers at a life insurance company.

ID	POLICY	ID	POLICY	ID	POLICY
1	Silver	8	Silver	15	Platinum
2	Platinum	9	Platinum	16	Silver
3	Gold	10	Platinum	17	Platinum
4	Gold	11	Silver	18	Platinum
5	Silver	12	Gold	19	Gold
6	Silver	13	Platinum	20	Silver
7	Bronze	14	Silver		

(a) Based on this data, calculate the following **summary statistics** for the POLICY feature:

i. Mode and 2nd mode

To calculate summary statistics for a categorical feature like this we start by counting the frequencies of each level for the feature. These are shown in this table:

Level	Frequency	Proportion
Bronze	1	0.05
Silver	8	0.40
Gold	4	0.20
Platinum	7	0.35

The proportions are calculated as the frequency of each level divided by the sum of all frequencies.

The mode is the most frequently occurring level and so in this case is *Silver*.

The 2nd mode is the second most frequently occurring level and so in this case is *Platinum*.

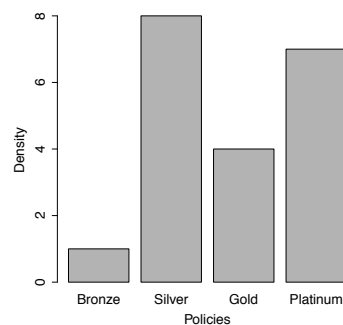
ii. Mode % and 2nd mode %

The mode % is the proportion of occurrence of the mode and in this case the proportion of occurrence of *Silver* is 40%.

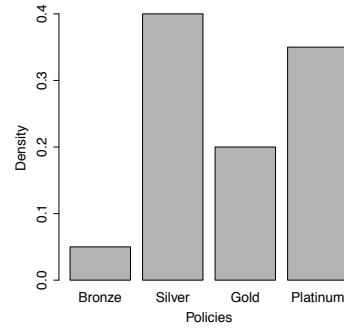
The mode % of the 2nd mode, *Platinum*, is 35%.

(b) Draw a **bar plot** for the POLICY feature.

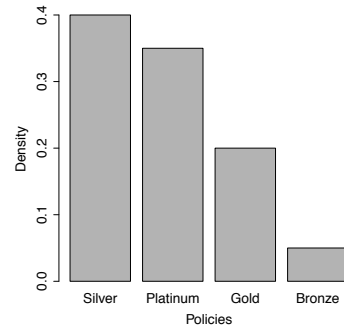
A bar plot can be drawn from the frequency table given above:



We can use proportions rather than frequencies in the plot:



In order to highlight the mode and 2nd mode we could order the bars in the plot by height:



This, however, is not an especially good idea in this case as the data, although categorical, has a natural ordering and changing this in a visualisation could cause confusion.

3. An analytics consultant at an insurance company has built an **ABT** that will be used to train a model to predict the best communications channel to use to contact a potential customer with an offer of a new insurance product.¹ The following table contains an extract from this ABT—the full ABT contains 5,200 instances.

ID	OCC	GENDER	AGE	LOC	MOTOR		HEALTH	HEALTH	HEALTH	HEALTH	PREF
					INS	VALUE			DEPS	DEPS	
							INS	TYPE	ADULTS	KIDS	CHANNEL
1	Student	female	43	urban	yes	42,632	yes	PlanC	1	2	sms
2		female	57	rural	yes	22,096	yes	PlanA	1	2	phone
3	Doctor	male	21	rural	yes	27,221	no				phone
4	Sheriff	female	47	rural	yes	21,460	yes	PlanB	1	3	phone
5	Painter	male	55	rural	yes	13,976	no				phone
		⋮			⋮				⋮		
14		male	19	rural	yes	48,66	no				email
15	Manager	male	51	rural	yes	12,759	no				phone
16	Farmer	male	49	rural	no		no				phone
17		female	18	urban	yes	16,399	no				sms
18	Analyst	male	47	rural	yes	14,767	no				email
		⋮			⋮				⋮		
2747		female	48	rural	yes	35,974	yes	PlanB	1	2	phone
2748	Editor	male	50	urban	yes	40,087	no				phone
2749		female	64	rural	yes	156,126	yes	PlanC	0	0	phone
2750	Reporter	female	48	urban	yes	27,912	yes	PlanB	1	2	email
		⋮			⋮				⋮		
4780	Nurse	male	49	rural	no		yes	PlanB	2	2	email
4781		female	46	rural	yes	18,562	no				phone
4782	Courier	male	63	urban	no		yes	PlanA	2	0	email
4783	Sales	male	21	urban	no		no				sms
4784	Surveyor	female	45	rural	yes	17,840	no				sms
		⋮			⋮				⋮		
5199	Clerk	male	48	rural	yes	19,448	yes	PlanB	1	3	email
5200	Cook	47 female		rural	yes	16,393	yes	PlanB	1	2	sms

The descriptive features in this dataset are defined as follows:

- AGE: The customer's age
- GENDER: The customer's gender (*male* or *female*)
- LOC: The customer's location (*rural* or *urban*)
- OCC: The customer's occupation

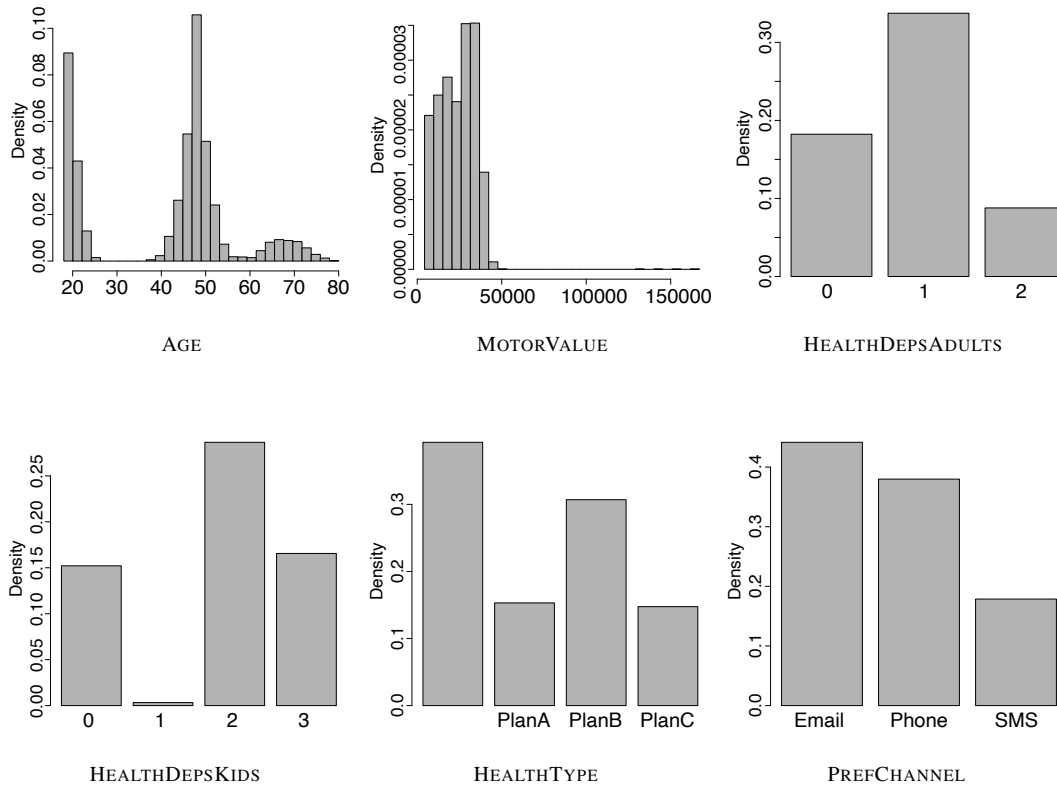
1. The data used in this question have been artificially generated for this book. Channel propensity modeling is used widely in industry; for example, see ?.

- **MOTORINS**: Whether the customer holds a motor insurance policy with the company (*yes* or *no*)
- **MOTORVALUE**: The value of the car on the motor policy
- **HEALTHINS**: Whether the customer holds a health insurance policy with the company (*yes* or *no*)
- **HEALTHTYPE**: The type of the health insurance policy (*PlanA*, *PlanB*, or *PlanC*)
- **HEALTHDEPSADULTS**: How many dependent adults are included on the health insurance policy
- **HEALTHDEPSKIDS**: How many dependent children are included on the health insurance policy
- **PREFCHANNEL**: The customer's preferred contact channel (*email*, *phone*, or *sms*)

The consultant generated the following **data quality report** from the ABT (visualizations of binary features have been omitted for space saving).

Feature	% %			Min.	1 st		Median	3 rd		Std. Dev.
	Count	Miss.	Card.		Qrt.	Mean		Qrt.	Max.	
AGE	5,200	0	51	18	22	41.59	47	50	80	15.66
MOTORVALUE	5,200	17.25	3,934	4,352	15,089.5	23,479	24,853	32,078	166,993	11,121
HEALTHDEPSADULTS	5,200	39.25	4	0	0	0.84	1	1	2	0.65
HEALTHDEPSKIDS	5,200	39.25	5	0	0	1.77	2	3	3	1.11

Feature	Count	% %		Mode	Mode		2 nd		2 nd	
		Miss.	Card.		Freq.	%	Mode	Freq.	Mode	%
GENDER	5,200	0	2	female	2,626	50.5	male	2,574	49.5	
LOC	5,200	0	2	urban	2,948	56.69	rural	2,252	43.30	
OCC	5,200	37.71	1,828	Nurse	11	0.34	Sales	9	0.28	
MOTORINS	5,200	0	2	yes	4,303	82.75	no	897	17.25	
HEALTHINS	5,200	0	2	yes	3,159	60.75	no	2,041	39.25	
HEALTHTYPE	5,200	39.25	4	PlanB	1,596	50.52	PlanA	796	25.20	
PREFCHANNEL	5,200	0	3	email	2,296	44.15	phone	1,975	37.98	



Discuss this data quality report in terms of the following:

(a) Missing values

Looking at the data quality report, we can see continuous and categorical features that have significant numbers of missing: **MOTORVALUE** (17.25%), **HEALTHDEPSADULTS** (39.25%), **HEALTHDEPSKIDS** (39.25%), **OCC** (37.71%), and **HEALTHTYPE** (39.25%).

The missing values in the **OCC** feature look typical of this type of data. A little over a third of the customers in the dataset appear to have simply not provided this piece of information. We will discuss this feature more under cardinality, but given the large percentage of missing values and the high cardinality of this feature, imputation is probably not a good strategy. Rather this feature might be a good candidate to form the basis of a derived flag feature that simply indicates whether an occupation was provided or not.

Inspecting rows 14 to 18 of the data sample given above, we can easily see the reason for the missing values in the `HEALTHDEPSADULTS`, `HEALTHDEPSKIDS`, and `HEALTHTYPE`. These features always have missing values when the `HEALTHINS` feature has a value of *no*. From a business point of view, this makes sense—if a customer does not hold a health insurance policy, then the details of a health insurance policy will not be populated. This also explains why the missing value percentages are the same for each of these features. The explanation for the missing values for the `MOTORVALUE` feature is, in fact, the same. Looking at rows 4780, 4782, and 4783, we can see that whenever the `MOTORINS` feature has a value of *no*, then the `MOTORVALUE` feature has a missing value. Again, this makes sense—if a customer does not have a motor insurance policy, then none of the details of a policy will be present.

(b) Irregular cardinality

In terms of cardinality, a few things stand out. First, the `AGE` feature has a relatively low cardinality (given that there are 5,200 instances in the dataset). This, however, is not especially surprising as ages are given in full years, and there is naturally only a small range possible—in this data, 18–80.

The `HEALTHDEPSADULTS` and `HEALTHDEPSKIDS` features are interesting in terms of cardinality. Both have very low values, 4 and 5 respectively. It is worth noting that a missing value counts in the cardinality calculation. For example, the only values present in the data for `HEALTHDEPSADULTS` are 0, 1, and 2, so it is the presence of missing values that brings cardinality to 4. We might consider changing these features to categorical features given the small number of distinct values. This, however, would lose some of the meaning captured in these features, so it should only be done after careful experimentation.

The `OCC` feature is interesting from a cardinality point of view. For a categorical feature to have 1,830 levels will make it pretty useless for building models. There are so many distinct values that it will be almost impossible to extract any patterns. This is further highlighted by the fact that the mode percentage for this feature is just 0.34%. This is also why no bar plot is provided for the `OCC` feature—there are just too many levels. Because of such high cardinality, we might just decide to remove this feature from the `ABT`. Another option would be to attempt to derive a feature that works at a higher level, for instance, *industry*, from the `OCC` feature. So, for example, occupations of *Occupational health nurse*, *Nurse*, *Osteopathic doctor*, and

Optometry doctor would all be transformed to *Medical*. Creating this new derived feature, however, would be a non-trivial task and would rely on the existence of an ontology or similar data resource that mapped job titles to industries.

(c) Outliers

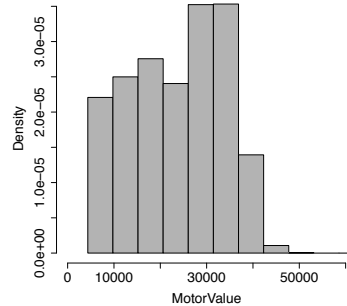
Only the MOTORVALUE feature really has an issue with outliers. We can see this in a couple of ways. First, the difference between the median and the 3rd quartile and the difference between the 3rd quartile and the maximum values are quite different. This suggests the presence of outliers. Second, the histogram of the MOTORVALUE feature shows huge skew to the right hand side. Finally, inspecting the data sample, we can see an instance of a very large value, 156,126 on row 2749. These outliers should be investigated with the business to determine whether they are valid or invalid, and based on this, a strategy should be developed to handle them. If valid, a clamp transformation is probably a good idea.

(d) Feature distributions

To understand the distributions of the different features, the visualizations are the most useful part of the data quality report. We'll look at the continuous features first. The AGE feature has a slightly odd distribution. We might expect age in a large population to follow a normal distribution, but this histogram shows very clear evidence of a multimodal distribution. There are three very distinct groups evident: One group of customers in their early twenties, another large group with a mean age of about 48, and a small group of older customers with a mean age of about 68. For customers of an insurance company, this is not entirely unusual, however. Insurance products tend to be targeted at specific age groups—for example, tailored motor insurance, health insurance, and life insurance policies—so it would not be unusual for a company to have specific cohorts of customers related to those products. Data with this type of distribution can also arise through merger and acquisition processes at companies. Perhaps this insurance company recently acquired another company that specialized in the senior travel insurance market? From a modeling point of view, we could hope that these three groups might be good predictors of the target feature, PREFCHANNEL.

It is hard to see much in the distribution of the MOTORVALUE feature because of the presence of the large outliers, which bunch the majority of the

data in a small portion of the graph. If we limit the histogram to a range that excludes these outliers (up to about 60,000), we can more easily see the distribution of the remaining instances.



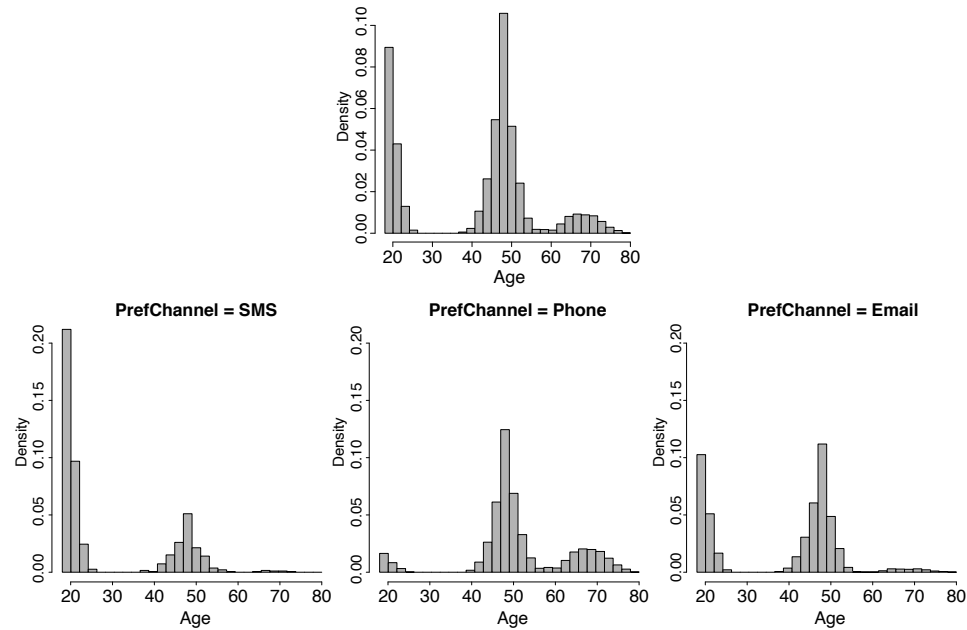
This distribution is not too far away from a unimodal distribution with left skew.

There is nothing especially remarkable about the distributions for HEALTHDEPSADULTS and HEALTHDEPSKIDS. Remembering that these features are populated only for customers with health insurance, it might be interesting for the business as a whole to learn that most customers with dependent children have more than one dependent child.

For the categorical features, the most interesting thing to learn from the distributions is that the target feature is slightly imbalanced. Many more customers prefer email contacts rather than phone or sms. Imbalanced target features can cause difficulty during the modeling process, so this should be marked for later investigation.

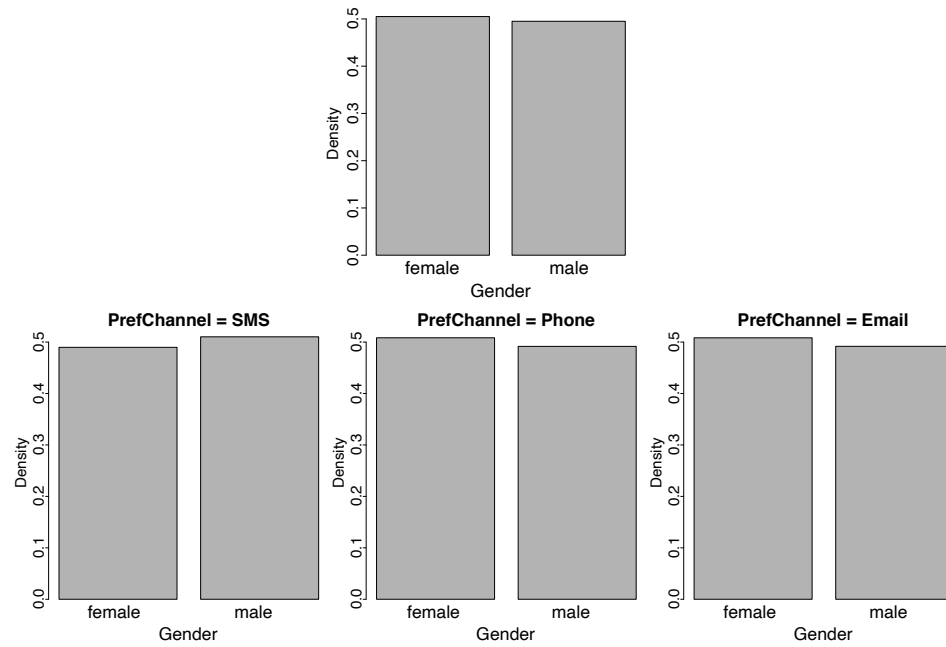
4. The following **data visualizations** are based on the channel prediction dataset given in Question 3. Each visualization illustrates the relationship between a descriptive feature and the target feature, PREFCHANNEL. Each visualization is composed of four plots: one plot of the distribution of the descriptive feature values in the entire dataset, and three plots illustrating the distribution of the descriptive feature values for each level of the target. Discuss the strength of the relationships shown in each visualization.

- (a) The visualization below illustrates the relationship between the continuous feature AGE and the target feature, PREFCHANNEL.

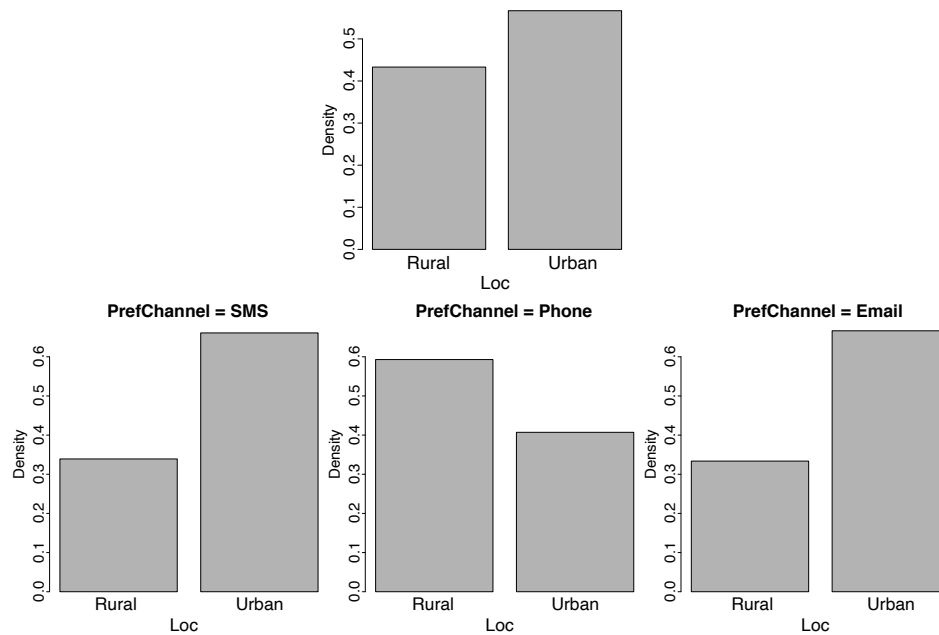


This visualization suggests a strong relationship between the AGE descriptive feature and the target feature, PREFCHANNEL. Overall we can see that the individual histograms for each data partition created by the different target levels are different from the overall histogram. Looking more deeply, we can see that the customers whose preferred channel is *sms* are predominantly younger. This is evident from the high bars in the 18–25 region of the histogram and the fact that there are very few instances in the age range above 60. We can see the opposite pattern for those customers whose preferred channel is *phone*. There are very few customers below 40 in this group. The histogram for the *email* group most closely matches the overall histogram.

- (b) The visualization below illustrates the relationship between the categorical feature GENDER and the target feature PREFCHANNEL.



- (c) The visualization below illustrates the relationship between the categorical feature LOC and the target feature, PREFCHANNEL.



The fact that the individual bar plots for each data partition are different from the overall bar plot suggests a relationship between these two features. In particular, for those customer's whose preferred channel is *phone*, the overall ratio between *rural* and *urban* locations is reversed—quite a few more rural customers prefer this channel. In the other two channel preference groups, *sms* and *email*, there are quite a few more urban dwellers. Together, this set of visualizations suggests that the LOC is reasonably predictive of the PREFCHANNEL feature.

5. The table below shows the scores achieved by a group of students on an exam.

ID	1	2	3	4	5	6	7	8	9	10
SCORE	42	47	59	27	84	49	72	43	73	59

ID	11	12	13	14	15	16	17	18	19	20
SCORE	58	82	50	79	89	75	70	59	67	35

Using this data, perform the following tasks on the SCORE feature:

- (a) A **range normalization** that generates data in the range (0, 1)

To perform a range normalization, we need the minimum and maximum of the dataset and the high and low for the target range. From the data we can see that the minimum is 27 and the maximum is 89. In the question we are told that the low value of the target range is 0 and that the high value is 1. Using these values, we normalize an individual value using the following equation:

$$a'_i = \frac{a_i - \min(a)}{\max(a) - \min(a)} \times (\text{high} - \text{low}) + \text{low}$$

So, the first score in the dataset, 42, would be normalized as follows:

$$\begin{aligned} a'_i &= \frac{42 - 27}{89 - 27} \times (1 - 0) + 0 \\ &= \frac{15}{62} \\ &= 0.2419 \end{aligned}$$

This is repeated for each instance in the dataset to give the full normalized data set as

ID	1	2	3	4	5	6	7	8	9	10
SCORE	0.24	0.32	0.52	0.00	0.92	0.35	0.73	0.26	0.74	0.52

ID	11	12	13	14	15	16	17	18	19	20
SCORE	0.50	0.89	0.37	0.84	1.00	0.77	0.69	0.52	0.65	0.13

- (b) A **range normalization** that generates data in the range (−1, 1)

This normalization differs from the previous range normalization only in that the high and low values are different—in this case, −1 and 1. So the first score in the dataset, 42, would be normalized as follows:

$$\begin{aligned} a'_i &= \frac{42 - 27}{89 - 27} \times (1 - (-1)) + (-1) \\ &= \frac{15}{62} \times 2 - 1 \\ &= -0.5161 \end{aligned}$$

Applying this to each instance in the dataset gives the full normalized dataset as

ID	1	2	3	4	5	6	7	8	9	10
SCORE	-0.52	-0.35	0.03	-1.00	0.84	-0.29	0.45	-0.48	0.48	0.03

ID	11	12	13	14	15	16	17	18	19	20
SCORE	0.00	0.77	-0.26	0.68	1.00	0.55	0.39	0.03	0.29	-0.74

(c) A **standardization** of the data

To perform a standardization, we use the following formula for each instance in the dataset:

$$a'_i = \frac{a_i - \bar{a}}{sd(a)}$$

So we need the mean, \bar{a} , and standard deviation, $sd(a)$, for the feature to be standardized. In this case, the mean is calculated from the original dataset as 60.95, and the standard deviation is 17.2519. So the standardized value for the first instance in the dataset can be calculated as

$$\begin{aligned} a'_i &= \frac{42 - 60.95}{17.2519} \\ &= -1.0984 \end{aligned}$$

Standardizing in the same way for the rest of the dataset gives us the following:

ID	1	2	3	4	5	6	7	8	9	10
SCORE	-1.10	-0.81	-0.11	-1.97	1.34	-0.69	0.64	-1.04	0.70	-0.11

ID	11	12	13	14	15	16	17	18	19	20
SCORE	-0.17	1.22	-0.63	1.05	1.63	0.81	0.52	-0.11	0.35	-1.50

6. The following table shows the IQs for a group of people who applied to take part in a television general-knowledge quiz.

ID	1	2	3	4	5	6	7	8	9	10
IQ	92	107	83	101	107	92	99	119	93	106

ID	11	12	13	14	15	16	17	18	19	20
IQ	105	88	106	90	97	118	120	72	100	104

Using this dataset, generate the following **binned** versions of the IQ feature:

- (a) An **equal-width binning** using 5 bins.

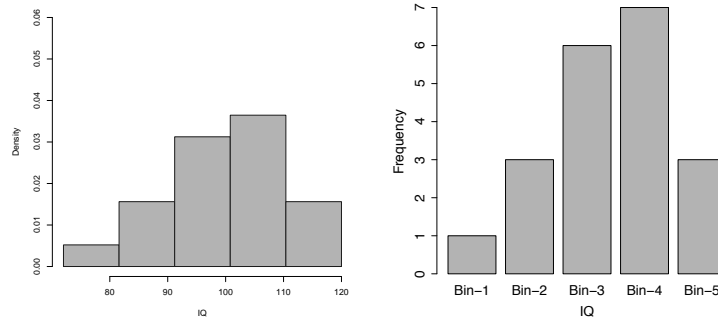
To perform an equal-width binning, we first calculate the bin size as $\frac{\text{range}}{b}$ where b is the number of bins. In this case, this is calculated as $\frac{120-72}{5} = 9.6$, where 72 and 120 are the minimum and maximum values. Using the bin size we can calculate the following bin ranges.

Bin	Low	High
1	72.0	81.6
2	81.6	91.2
3	91.2	100.8
4	100.8	110.4
5	110.4	120.0

Once we have calculated the boundaries, we can use these to determine the bin to which each result belongs.

ID	IQ	IQ (BIN)	ID	IQ	IQ (BIN)
1	92	Bin-3	11	105	Bin-4
2	107	Bin-4	12	88	Bin-2
3	83	Bin-2	13	106	Bin-4
4	101	Bin-4	14	90	Bin-2
5	107	Bin-4	15	97	Bin-3
6	92	Bin-3	16	118	Bin-5
7	99	Bin-3	17	120	Bin-5
8	119	Bin-5	18	72	Bin-1
9	93	Bin-3	19	100	Bin-3
10	106	Bin-4	20	104	Bin-4

It is interesting to graph a histogram of the values in the dataset according to the bin boundaries as well as a bar plot showing each of the bins created.



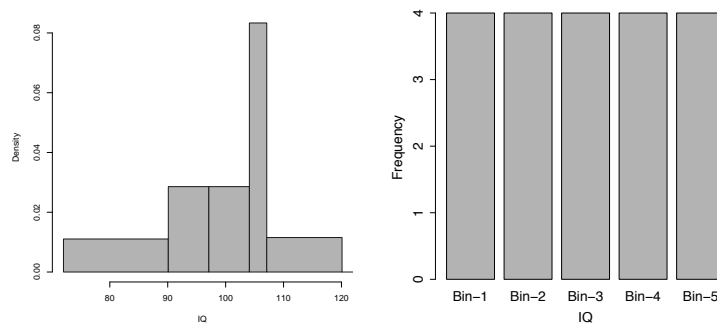
We can see from these graphs that a lot of instances belong to the middle bins and very few in the high and low bins.

(b) An **equal-frequency binning** using 5 bins

To perform an equal-frequency binning, we first determine the number of instances that will belong to each bin. This is simply the number of instances in the dataset divided by the number of bins, in this case $\frac{20}{5} = 4$. Next we sort the data by the binning feature and assign instances in order to the first bin until it is full, before moving to the second and so on. The table below shows how the instances in this dataset, sorted by **RESULT**, would be added to the five bins.

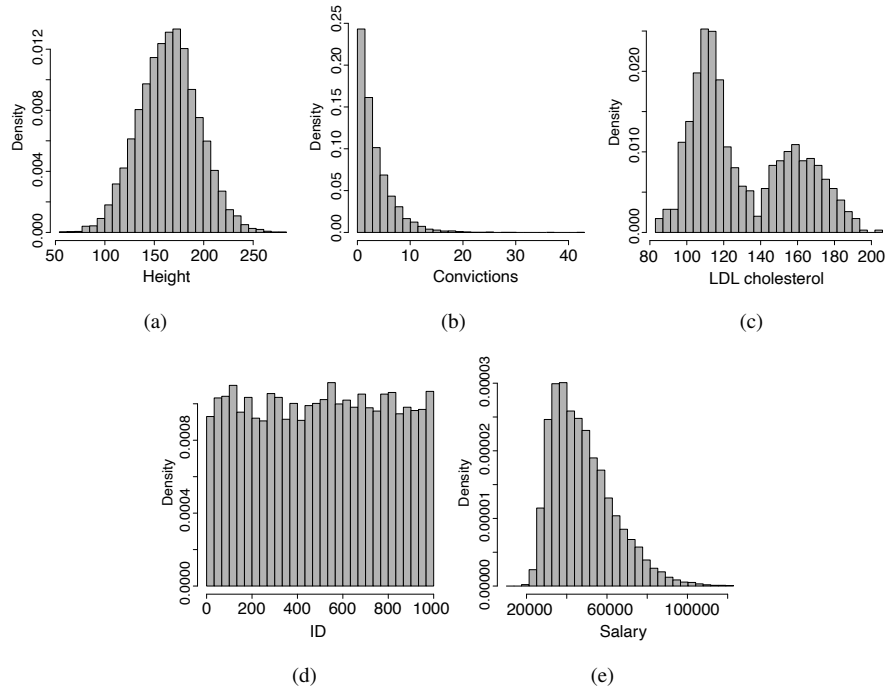
ID	IQ	IQ (BIN)	ID	IQ	IQ (BIN)
18	72	Bin-1	4	101	Bin-3
3	83	Bin-1	20	104	Bin-3
12	88	Bin-1	11	105	Bin-4
14	90	Bin-1	10	106	Bin-4
1	92	Bin-2	13	106	Bin-4
6	92	Bin-2	2	107	Bin-4
9	93	Bin-2	5	107	Bin-5
15	97	Bin-2	16	118	Bin-5
7	99	Bin-3	8	119	Bin-5
19	100	Bin-3	17	120	Bin-5

It is interesting to graph a histogram of the values in the dataset according to the bin boundaries as well as a bar plot showing each of the bins created.



Remember that in the histogram, because the bins are not of equal width, the bars are different heights. The area of a bar represents the density of the instances in the range represented by the bar. The key things to note here are that each bin is equally populated, but that the bins in the middle are much narrower than those at either end of the range.

* 7. Comment on the **distributions** of the features shown in each of the following histograms.



- (a) The height of employees in a trucking company.

This data almost perfectly follows a normal distribution. The mean is about 175. Seeing data that follows a normal distribution is almost always encouraging in data analytics as it is the most well-behaved sort of data.

- (b) The number of prior criminal convictions held by people given prison sentences in a city district over the course of a full year.

This data follows an exponential distribution. There is a strong central tendency around 0 and ever decreasing probability of seeing higher values. Data following an exponential distribution can be hard to manage as there tend to be significant outliers, which can upset modeling algorithms. In this case there is at least one instance of a person having over 40 prior convictions, which is very unusual.

- (c) The LDL cholesterol values for a large group of patients, including smokers and non-smokers.

There is a very evident multimodal, or more specifically bimodal, character to this data. There appear to be two distinct groups, one with a central tendency around 110, and another with a central tendency around 160. The first group is bigger than the second. From the description of the data, these groups could be smokers (with the higher cholesterol values) and non-smokers (with the lower cholesterol values). Without further data, however, there is nothing here to support this conjecture, so more investigation is required—generating separate histograms for smokers and non-smokers would be an easy way to further investigate this issue.

- (d) The employee ID numbers of the academic staff at a university.

This data is almost perfectly uniformly distributed—any ID number is just as likely as any other.

- (e) The salaries of car insurance policyholders.

This data follows a unimodal distribution with a fairly strong right skew. This is typical of monetary features such as salaries, rental prices, and purchase prices. There will be a strong central tendency (in this case around 32,000), but there will be some very high values. A distribution like this should always be seen as an indication that the data contains valid outliers that may need to be dealt with.

* 8. The table below shows socioeconomic data for a selection of countries for the year 2009,² using the following features:

- COUNTRY: The name of the country
- LIFEEXPECTANCY: The average life expectancy (in years)
- INFANTMORTALITY: The infant mortality rate (per 1,000 live births)
- EDUCATION: Spending per primary student as a percentage of GDP
- HEALTH: Health spending as a percentage of GDP
- HEALTHUSD: Health spending per person converted into US dollars

COUNTRY	LIFE EXPECTANCY	INFANT MORTALITY	EDUCATION	HEALTH	HEALTH USD
Argentina	75.592	13.500	16.841	9.525	734.093
Cameroon	53.288	67.700	7.137	4.915	60.412
Chile	78.936	7.800	17.356	8.400	801.915
Colombia	73.213	16.500	15.589	7.600	391.859
Cuba	78.552	4.800	44.173	12.100	672.204
Ghana	60.375	52.500	11.365	5.000	54.471
Guyana	65.560	31.200	8.220	6.200	166.718
Latvia	71.736	8.500	31.364	6.600	756.401
Malaysia	74.306	7.100	14.621	4.600	316.478
Mali	53.358	85.500	14.979	5.500	33.089
Mongolia	66.564	26.400	15.121	5.700	96.537
Morocco	70.012	29.900	16.930	5.200	151.513
Senegal	62.653	48.700	17.703	5.700	59.658
Serbia	73.532	6.900	61.638	10.500	576.494
Thailand	73.627	12.700	24.351	4.200	160.136

(a) Calculate the **correlation** between the LIFEEXPECTANCY and INFANTMORTALITY features.

The first step in calculating correlation is to calculate the variance between the two features. Recall that to calculate covariance, we use

$$\text{cov}(a, b) = \frac{1}{n-1} \sum_{i=1}^n \left((a_i - \bar{a}) \times (b_i - \bar{b}) \right)$$

The table below shows the workings used to make this calculation.

2. The data listed in this table is real and was amalgamated from a number of reports that were retrieved from Gapminder (www.gapminder.org). The EDUCATION data is based on a report from the World Bank (data.worldbank.org/indicator/SE.XPD.PRIM.PC.ZS); the HEALTH and HEALTHUSD data are based on reports from the World Health Organization (www.who.int); all the other features are based on reports created by Gapminder.

ID	LIFE EXPECTANCY	$l - \bar{l}$	INFANT MORTALITY	$i - \bar{i}$	$(l - \bar{l}) \times (i - \bar{i})$
	(l)		(i)		
1	76	6.8	13.5	-14.5	-99.0
2	53	-15.5	67.7	39.7	-614.3
3	79	10.2	7.8	-20.2	-205.5
4	73	4.5	16.5	-11.5	-51.2
5	79	9.8	4.8	-23.2	-227.1
6	60	-8.4	52.5	24.5	-205.4
7	66	-3.2	31.2	3.2	-10.3
8	72	3.0	8.5	-19.5	-58.1
9	74	5.6	7.1	-20.9	-115.9
10	53	-15.4	85.5	57.5	-885.6
11	67	-2.2	26.4	-1.6	3.5
12	70	1.3	29.9	1.9	2.4
13	63	-6.1	48.7	20.7	-126.4
14	74	4.8	6.9	-21.1	-100.7
15	74	4.9	12.7	-15.3	-74.5
Mean	68.75		27.98		
Std Dev	8.25		24.95		
Sum					-2,768.2

Using the sum at the bottom of this table, we can calculate the covariance between the LIFEEXPECTANCY and INFANTMORTALITY features as

$$\frac{-2,768.2}{15 - 1} = -197.7$$

where 15 is the number of instances.

Recall that correlation is calculated as

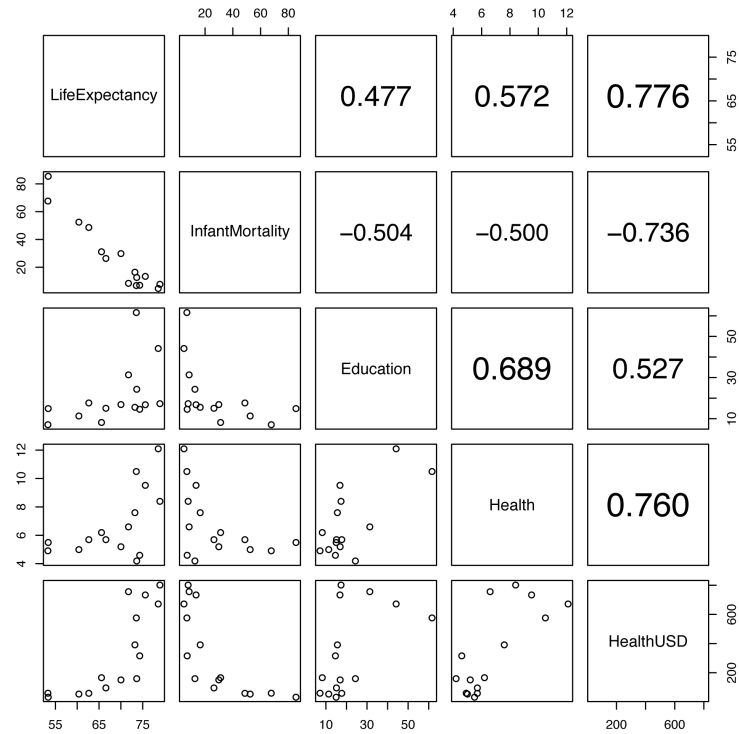
$$\text{corr}(a, b) = \frac{\text{cov}(a, b)}{\text{sd}(a) \times \text{sd}(b)}$$

So we can calculate the correlation between the LIFEEXPECTANCY and INFANTMORTALITY features as

$$\frac{-197.7}{8.25 \times 24.95} = -0.961$$

where 8.25 and 24.95 are the standard deviations of the two features. This implies a very strong negative correlation between these two features.

- (b) The image below shows a **scatter plot matrix** of the continuous features from this dataset (the correlation between LIFEEXPECTANCY and INFANTMORTALITY has been omitted). Discuss the relationships between the features in the dataset that this scatter plot highlights.



The relationships apparent in this plot are as follows:

- **LIFEEXPECTANCY and INFANTMORTALITY:** As calculated previously, the strong negative correlation between these two features is plainly evident.
- **LIFEEXPECTANCY and EDUCATION:** There seems to be a slight positive relationship between these two features.
- **LIFEEXPECTANCY and HEALTH:** There seems to be a slight positive relationship between these two features.
- **LIFEEXPECTANCY and HEALTHUSD:** There seems to be a slightly stronger positive relationship between LIFEEXPECTANCY and this version of health spending. Also this relationship appears to be somewhat non-linear, which would be worth looking into more.

- **INFANTMORTALITY and EDUCATION:** There seems to be a slight negative relationship between these two features—as education spending goes up, the infant mortality rate seems to reduce.
- **INFANTMORTALITY and HEALTH:** It is a little surprising that the relationship here is not stronger, but it is still quite evident—as health spending goes up, infant mortality rates go down.
- **INFANTMORTALITY and HEALTHUSD:** This chart is interesting as it seems to show a pretty strong non-linear relationship between these two features. The correlation coefficient used measures only linear relationships, so this is worthy of further investigation.
- **EDUCATION and HEALTH:** There is quite a strong positive relationship between education and health spending. We can see in the graph, however, that the dataset only sparsely covers the range of possible values, so it would be interesting to look at this trend in a larger dataset.
- **HEALTH and HEALTHUSD:** This relationship shows the limitations of correlation measures. Although these two features measure the same thing—the amount of health spending in a country—they are not the most strongly correlated pair in the dataset. The relationship appears to be non-linear, which makes sense given the fact that it has varying exchange rates embedded in it.

* 9. Tachycardia is a condition that causes the heart to beat faster than normal at rest. The occurrence of tachycardia can have serious implications including increased risk of stroke or sudden cardiac arrest. An analytics consultant has been hired by a major hospital to build a predictive model that predicts the likelihood that a patient at a heart disease clinic will suffer from tachycardia in the month following a visit to the clinic. The hospital will use this model to make predictions for each patient when they visit the clinic and offer increased monitoring for those deemed to be at risk. The analytics consultant has generated an ABT to be used to train this model.³ The descriptive features in this dataset are defined as follows:

- **AGE:** The patient's age
- **GENDER:** The patient's gender (*male* or *female*)
- **WEIGHT:** The patient's weight
- **HEIGHT:** The patient's height

3. The data used in this question have been artificially generated for this book. This type of application of machine learning techniques, however, is common; for example, see ?.

- BMI: The patient's body mass index (BMI) which is calculated as $\frac{\text{weight}}{\text{height}^2}$ where weight is measured in kilograms and height in meters.
- SYS. B.P.: The patient's systolic blood pressure
- DIA. B.P.: The patient's diastolic blood pressure
- HEART RATE: The patient's heart rate
- H.R. DIFF.: The difference between the patient's heart rate at this visit and at their last visit to the clinic
- PREV. TACHY.: Has the patient suffered from tachycardia before?
- TACHYCARDIA: Is the patient at high risk of suffering from tachycardia in the next month?

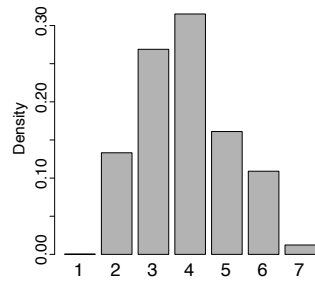
The following table contains an extract from this ABT—the full ABT contains 2,440 instances.

ID	AGE	GENDER	WEIGHT	HEIGHT	BMI	SYS. B.P.	DIA. B.P.	HEART RATE	H.R. DIFF.	PREV. TACHY.	TACHYCARDIA
1	6	male	78	165	28.65	161	97	143			
2	5	m	117	171	40.01	216	143	162	17	true	true
		⋮			⋮				⋮		
143	5	male	108	1.88	305,568.13	139	99	84	21	false	true
144	4	male	107	183	31.95	1,144	90	94	-8	false	true
		⋮			⋮				⋮		
1,158	6	female	92	1.71	314,626.72	111	75	75	-5		false
1,159	3	female	151	1.59	596,495.39	124	91	115	23	true	true
		⋮			⋮				⋮		
1,702	3	male	86	193	23.09	138	81	83		false	false
1,703	6	f	73	166	26.49	134	86	84	-4		false
		⋮			⋮				⋮		

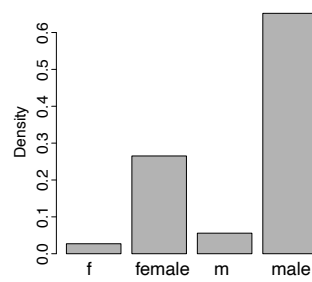
The consultant generated the following **data quality report** from the ABT.

Feature	Count	%		Min.	1 st Qrt.	Mean	Median	3 rd Qrt.	Max.	Std. Dev.
AGE	2,440	0.00	7	1.00	3.00	3.88	4.00	5.00	7.00	1.22
WEIGHT	2,440	0.00	174	0.00	81.00	95.70	95.00	107.00	187.20	20.89
HEIGHT	2,440	0.00	109	1.47	162.00	162.21	171.50	179.00	204.00	41.06
BMI	2,440	0.00	1,385	0.00	27.64	18,523.40	32.02	38.57	596,495.39	77,068.75
SYS. B.P.	2,440	0.00	149	62.00	115.00	127.84	124.00	135.00	1,144.00	29.11
DIA. B.P.	2,440	0.00	109	46.00	77.00	86.34	84.00	92.00	173.60	14.25
HEART RATE	2,440	0.00	119	57.00	91.75	103.28	100.00	110.00	190.40	18.21
H.R. DIFF.	2,440	13.03	78	-50.00	-4.00	3.00	1.00	8.00	47.00	12.38

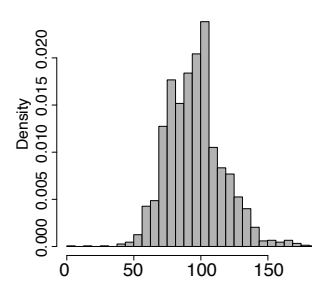
Feature	Count	% Miss.		Card.	Mode	Mode Freq.	Mode %	2 nd Mode		2 nd Mode	
GENDER	2,440	0.00		4	male	1,591.00	65.20	female		647.00	26.52
PREV. TACHY.	2,440	44.02		3	false	714.00	52.27	true		652.00	47.73
TACHYCARDIA	2,440	2.01		3	false	1,205.00	50.40	true		1,186.00	49.60



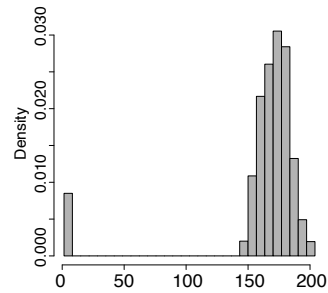
AGE



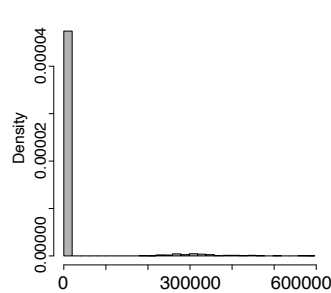
GENDER



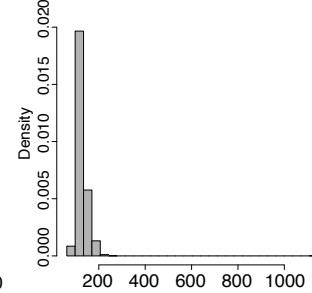
WEIGHT



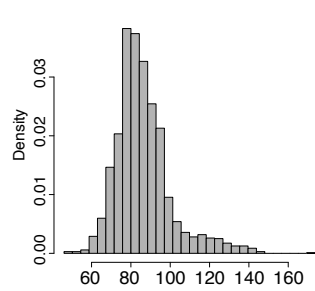
HEIGHT



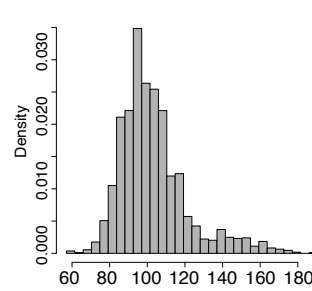
BMI



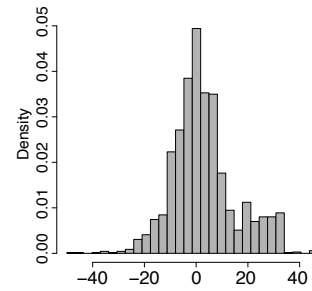
SYS. B.P.



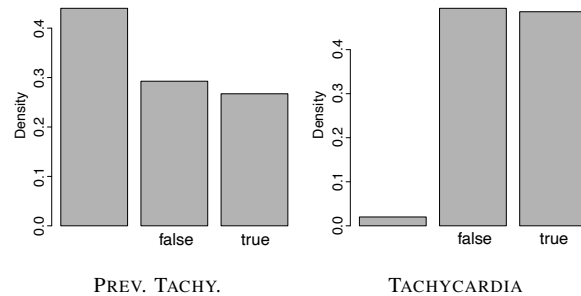
DIA. B.P.



HEART RATE



H.R DIFF.



Discuss this data quality report in terms of the following:

(a) Missing values

There are three features in the dataset that exhibit missing values: H.R. DIFF. (13.03%), PREV. TACHY. (44.02%), and TACHYCARDIA (2.01%).

The number of missing values in the H.R. DIFF. feature is not so high that the feature should be considered for removal from the ABT, but too high to consider complete case removal. This feature should be noted in the data quality plan for later handling if required. Mean imputation would probably work well if these missing values needed to be handled.

The PREV. TACHY. feature is missing values in (44.02%) of the instances in the dataset. This value is so high that this feature should be removed from the dataset—any efforts at imputation would simply alter the dataset too much.

The TACHYCARDIA feature is the target feature in this dataset and so it is slightly alarming that there are missing values for this feature. The instances that are missing values for this feature should be removed from the dataset (that is complete case removal should be used). Imputation should never be attempted for a target feature.

(b) Irregular cardinality

Many of the numeric features in the data set have cardinalities that are much lower than the number of instances in the dataset (2,440). This is not surprising, however, because most of the features are integer values and have defined ranges—for example diastolic blood pressure (DIA. B.P.) values should fall between about 40 and 100. The features that do stand out in terms of cardinality are AGE and GENDER.

The AGE feature has a cardinality of 7 which is very low for a numeric feature. Examining the bar plot for AGE we see that there are only 7 distinct values, each of which is relatively well represented. This would suggest that rather than storing actual age values the AGE feature in this dataset represents is in fact categorical with 7 ordinal categories (this is in fact the case with this dataset—category 1 represents ages between 10 and 19, category 2 represents ages between 20 and 29 and so on).

The GENDER feature has cardinality of 4 which is a little odd. Examining the data extract given above and the bar plot for the GENDER feature we can see that there are actually 4 levels in this dataset—*male*, *female*, *m*, and *f*. This is a common data quality issue and representative of either data from different sources or manually entered data. The *m* and *f* values should be recoded to *male* and *female* respectively.

(c) Outliers

An examination of the data quality report (both tables and visualizations) indicates that three features seem to have problems with outliers: HEIGHT, BMI, and SYS. B.P..

Looking first at HEIGHT we can see that the difference between the 1st quartile and the median values and the difference between the minimum and 1st quartile values are quite different and that the histogram of the HEIGHT feature values shows significant skew to the left hand side. Both of these characteristics are suggestive of outliers. The histogram suggests that the outliers are in a well defined group with values less than 10. Examining the data we see that there are a number of height values that are a couple of orders of magnitude less than the majority of values (for example rows 143, 1,158, and 1,159). It is possible that these are invalid outliers that arose through data entry problems, and this should be investigated further (in this example this is the case as these values were entered in meters rather than centimeters like the rest of the height values). If the root of the error can be found these outliers could be corrected (in this example they can easily be corrected by multiplying them by 100).

The data seems to contain a number of very large values for the BMI feature. This is evident from the unreasonable maximum value shown in the table above (BMI values should range from about 15 to about 60), and the massive skew in the histogram for this feature. In cases like this it is always a good idea to find the instances in the dataset that actually contain the maximum value as this often offers clues to the origin of the outliers. The maximum

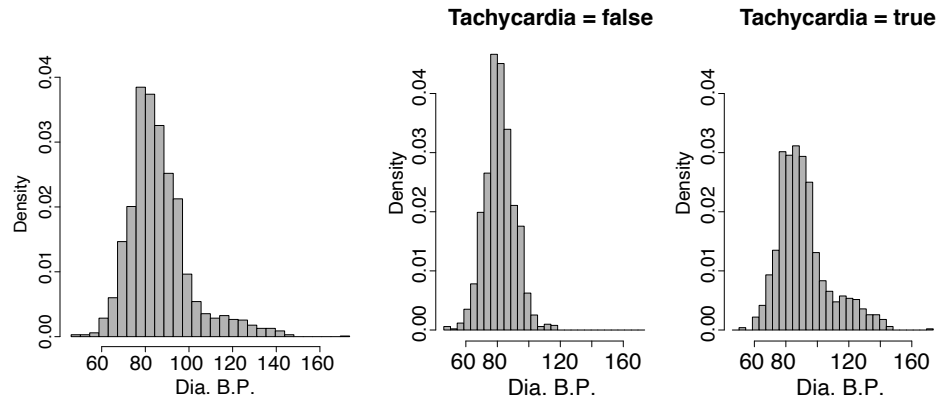
BMI value of 596,495.39 occurs on row 1,159. This is a totally unreasonable value for BMI. It is interesting to note, however, that this value for BMI occurs beside one of the erroneous values for the HEIGHT feature. Looking further through the data sample it is evident that the large values for BMI always occur together with low values for HEIGHT (and this is the case in the full dataset). As BMI is calculated from WEIGHT and HEIGHT it is likely that the incorrect values for HEIGHT are causing the incorrect BMI values. These outliers could be corrected by correcting the incorrect HEIGHT values and recalculating the values for BMI.

The SYS. B.P. feature has a maximum value that is not reasonable for systolic blood pressure and that is much further beyond the 3rd quartile than the 3rd quartile is beyond the median. Again, finding this value in the dataset is a useful first step. It is found on row 144 in the dataset. Further investigation shows that this is the only value in this range and this is a single invalid outlier. This value should be changed to a missing value. An entry should be added to the data quality plan to say that this missing value might be handled using mean imputation if required.

(d) Feature distributions

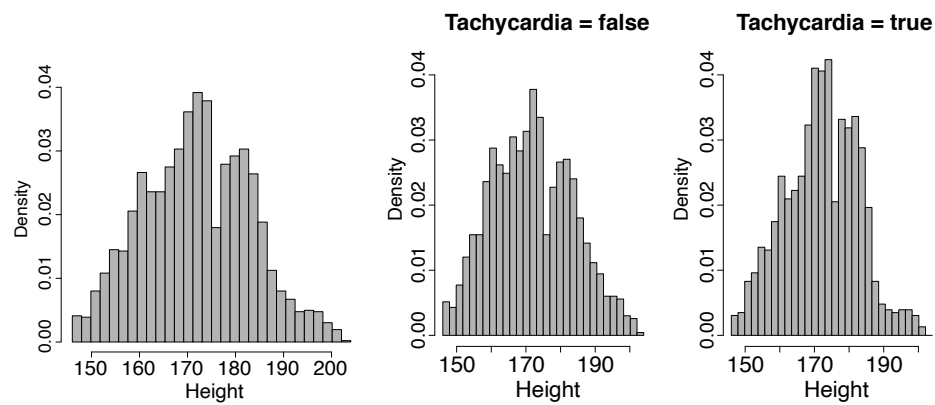
Nothing particularly stands out in the distributions of the features. The target feature levels seem to be evenly distributed which is a positive and should make modelling easier further down the line. Most of the continuous descriptive features are broadly normally distributed. The H.R. DIFF. feature seems to be somewhat bi-modal with a peak on the upper end. It would be interesting to see if there are any relationships between higher values of this feature and the target feature.

- * 10. The following data visualizations are based on the tachycardia prediction dataset from Question 9 (after the instances with missing TACHYCARDIA values have been removed and all outliers have been handled). Each visualization illustrates the relationship between a descriptive feature and the target feature, TACHYCARDIA and is composed of three plots: a plot of the distribution of the descriptive feature values in the full dataset, and plots showing the distribution of the descriptive feature values for each level of the target. Discuss the relationships shown in each visualizations.
- (a) The visualization below illustrates the relationship between the continuous feature DIA. B.P. and the target feature, TACHYCARDIA.



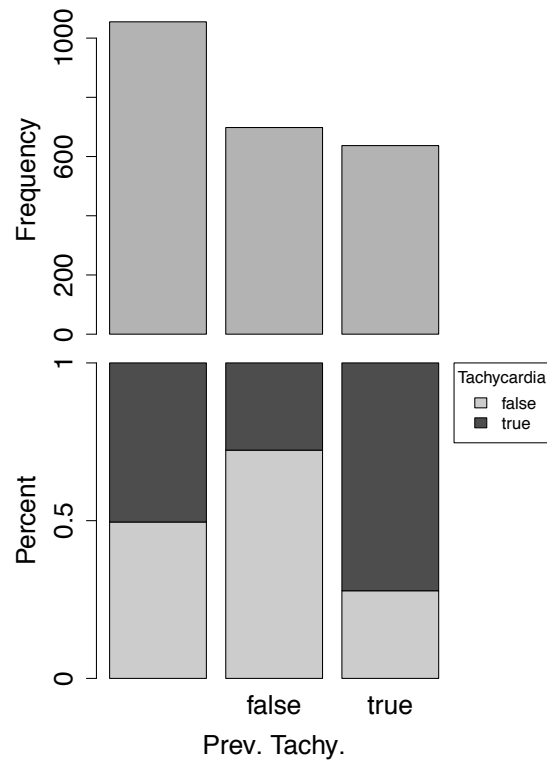
The histogram for the different levels of the TACHYCARDIA feature are slightly different suggesting that there is a relationship between the DIA. B.P. and TACHYCARDIA features. It looks like patients with higher diastolic blood pressure values are more likely to suffer with tachycardia than those with lower blood pressure values.

- (b) The visualization below illustrates the relationship between the continuous HEIGHT feature and the target feature TACHYCARDIA.



The histograms for the the HEIGHT feature split according to the different target levels look very similar. This suggests that there is no strong relationship between the HEIGHT feature and the target feature, TACHYCARDIA.

- (c) The visualization below illustrates the relationship between the categorical feature PREV. TACHY. and the target feature, TACHYCARDIA.



This visualization is interesting as it shows a number of different things. First, it shows that although there are many missing values for the PREV. TACHY. feature, they are equally prevalent for both values of the target feature. This is illustrated by the even split between the target levels in the bar on the left hand side of the bottom chart. The second things that this visualization shows is that when a value for PREV. TACHY. is present it is quite predictive of the target feature—previous occurrences of tachycardia are predictive of feature occurrences. Although we previously suggested that this feature should be omitted from the dataset this visualization would encourage us to return to this decision. While there are just too many missing values to allow this feature to be used at present the analytics consultant should make every effort to source values for this feature for this dataset and any future work in this area.

- * 11. Worldwide **breast cancer** is the most common form of cancer for women, and the second most common form of cancer overall.⁴ Reliable, population-wide screening is one tool that can be used to reduce the impact of breast cancer, and there is an opportunity for machine learning to be used for this. A large hospital group has collected a cancer screening dataset for possible use with machine learning that contains features extracted from tissue samples extracted by biopsy from adults presenting for screening. Features have been extracted from these biopsies by lab technicians who rate samples across a number of categories on a scale of 1 to 10. The samples have then been manually categorized by clinicians as either *benign* or *malignant*.⁵ The descriptive features in this dataset are defined as follows:

- AGE: The age of the person screened.
- SEX: The sex of the person screened, either *male* or *female*.
- SIZEUNIFORMITY: A measure of the variation in size of cells in the tissue samples, higher values indicate more uniform sizes (1 to 10).
- SHAPEUNIFORMITY: A measure of the variation in shape of cells in the tissue samples, higher values indicate more uniform shapes (1 to 10).
- MARGINALADHESION: A measure of how much cells in the biopsy stick together (1 to 10).
- MITOSES: A measure of how fast cells are growing (1 to 10).
- CLUMPTHICKNESS: A measure of the amount of layering in cells (1 to 10).
- BLANDCHROMATIN: A measure of the texture of cell nuclei (1 to 10).
- CLASS: The clinician's assessment of the biopsy sample as either *benign* or *malignant*.

4. Based on data from ?.

5. The data in this question have been artificially created but were inspired by the famous **Wisconsin breast cancer dataset** first described by ?, and available from the UCI Machine Learning Repository (?).

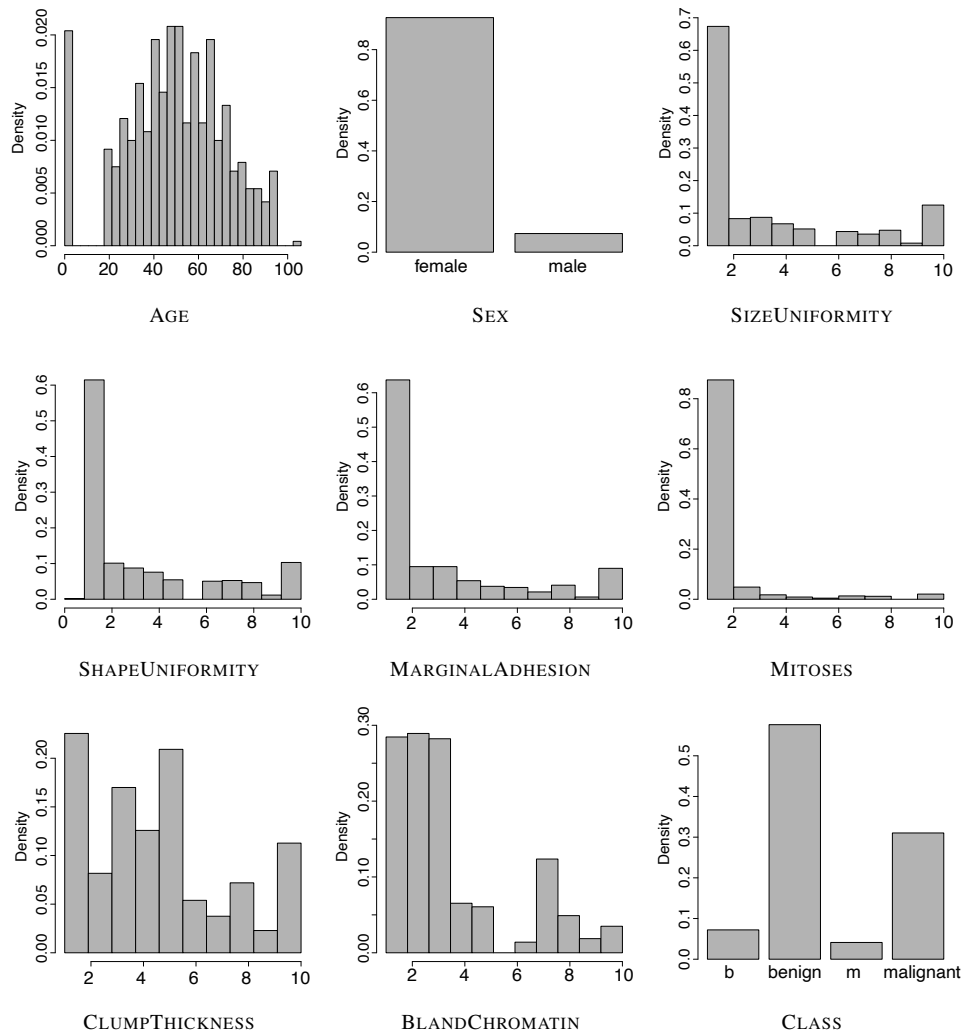
The following table contains an extract from this ABT—the full ABT contains 680 instances.

ID	AGE	SEX	CLUMP	SIZE	SHAPE	MARGINAL	BLAND	MITOSES	CLASS
			THICKNESS	UNIFORMITY	UNIFORMITY	ADHESION	CHROMATIN		
1	56	female	3	4	5	3	4	1	benign
2	77	female	2	1	0	1	1	1	benign
48	34	female	5	2	4	1		1	benign
49	46	female	5	3	1	2	2	1	b
50	106	female	2	1	1	1	1	1	benign
303	95	female	1			1	1	1	benign
304	28	male	5	1	1	1		1	benign
305	0	female	3			1	3	1	benign
679	48	female	10	8	7	4	7	1	m
680	43	female	5	4	6	7		1	malignant

The following **data quality report** has been generated from the ABT.

Feature	Count	% Miss. Card.		Mode Freq.	Mode %	2 nd Mode Freq.		2 nd Mode %	
SEX	680	0	2	female	630	92.65	male	50	7.35
CLASS	680	0	4	benign	392	57.65	malignant	211	31.03

Feature	Count	%		Min.	1 st			3 rd		Std. Dev.
		Miss.	Card.		Qrt.	Mean	Median	Qrt.	Max.	
AGE	680	0.00	80	0	34	49.13	50	65	106	22.95
CLUMPTHICKNESS	680	0.00	10	1	2	4.45	4	6	10	2.82
SIZEUNIFORMITY	680	9.26	10	1	1	3.14	1	5	10	3.08
SHAPEUNIFORMITY	680	9.26	11	0	1	3.20	1	5	10	3.00
MARGINALADHESION	680	0.00	10	1	1	2.84	1	4	10	2.87
BLANDCHROMATIN	680	22.94	10	1	2	3.38	3	4	10	2.44
MITOSES	680	0.00	9	1	1	1.61	1	1	10	1.74



Discuss this data quality report in terms of the following:

(a) Missing values

The features SIZEUNIFORMITY, SHAPEUNIFORMITY, and BLANDCHROMATIN all have significant numbers of missing values—9.26%, 9.26%, and 22.94% respectively. These values are all below the upper limit for imputation.

tation described in Section 3.4.1^[69] and so imputation should be considered for these features for modelling algorithms that cannot handle missing values. Given that these values are integers in the range 1 to 10 and that they largely have exponential distributions, mode imputation probably makes more sense than mean or median imputation.

The fact that the percentage of missing values for both SIZEUNIFORMITY and SHAPEUNIFORMITY are exactly the same, 9.26%, should raise our curiosity. This suggests that there is a systematic pattern of missing values across both of these instances. An examination of the dataset supports this suggestion, for example see rows 303 and 305 in the data sample above.

(b) Irregular cardinality

The most obvious thing to notice is that almost all of the descriptive features (CLUMPTHICKNESS, SIZEUNIFORMITY, SHAPEUNIFORMITY, MARGINAL-ADHESION, BLANDCHROMATIN, and MITOSES), which at first glance appear numeric, each have cardinality of just 10. These descriptive features are based on manual assessment of the biopsy samples in which lab technicians rate on scales from 1 to 10. so, this is really **ordinal** data rather than numeric data. As the scale is from 1 to 10 it is still probably appropriate to treat this data as **numeric**, but some algorithms can perform specific operations on ordinal data and this opportunity should be explored for those algorithms.

Out of these features MITOSES and SIZEUNIFORMITY further stand out. MITOSES has a cardinality of only 9 rather than 10. We can just about see from the histogram for MITOSES that the value 9 never appears in the dataset which accounts for this. SIZEUNIFORMITY has a cardinality of 11 which is accounted for by the presence of an outlier, discussed as part of the answer to the next question.

The target features, CLASS, also has unusual cardinality—a value of 4 rather than the expected value of 2 for a binary feature. Examining the CLASS bar plot and the data sample it is obvious that occasionally outcomes have been record as *b* and *m* rather than *benign* and *malignant*. This is also clear from the fact that the 1st and 2nd mode percentages sum to 88.68% rather than 100%. This mislabeling should be corrected.

Purely numeric features should have cardinality close to the number of rows in the dataset. The AGE feature has cardinality of 80 which is much less than this. This is appropriate in this case, however, as AGE is an integer feature and valid ages for adults probably range from about 18 to 100.

(c) Outliers

The AGE feature has the most obvious issue with outliers. Its minimum value is 0, which is not an appropriate value for a person's age and so is a clear example of an **invalid outlier**. The histogram for age shows that there are multiple rows with this value (we can see one on row 305 above), which suggests that this most likely coding for missing values. A deeper examination of the dataset (this exact information is not directly available from the visualisations and tables shown in this question, but the conclusion is) shows that 49 rows, or approximately 7% of the data, contains 0 values for age. These zero values should be replaced with a missing identifier to distinguish them from genuine values. This means that AGE is another candidate for imputation if required.

AGE also has one unusually high value, 106, we can see this in row 50 above. It is hard to tell from this value alone whether or not this is an instance of a **valid outlier** or an **invalid outlier**. An age of 106 is unusual, but possible. It probably makes sense in this case to be conservative, but this outlier should be recorded for possible handling later.

The SHAPEUNIFORMITY feature also exhibits an outlier, evident from the fact that the minimum value is 0. The values of this feature should only range between 1 and 10 so this is an example of an **invalid outlier** and should probably be replaced with a 1, regenerated, or perhaps the row should be completely removed.

(d) Feature distributions

The AGE feature follows a largely normal distribution, with a little left skew, which is reasonable for age in a population like this. The other numeric features largely follow something close to an exponential distribution skewed heavily towards lower values. In the cases of SIZEUNIFORMITY, SHAPEUNIFORMITY, MARGINALADHESION, and MITOSES this is quite extreme and may be problematic for some modelling algorithms as values of 1 dominate the distribution. It might be interesting to consider recoding these as binary features with values of 1 or above which could lead to simpler models.

The SEX feature is heavily skewed towards *female*, which is not surprising given that the focus of this dataset is breast cancer screening. In fact at first it might seem unusual to see males in this dataset at all. Although much less common than for females, breast cancer does affect males. Less than 1%

of diagnosed cancers in males are breast cancer, whereas for females breast cancer accounts for approximately 24% of all diagnosed cancers.^a

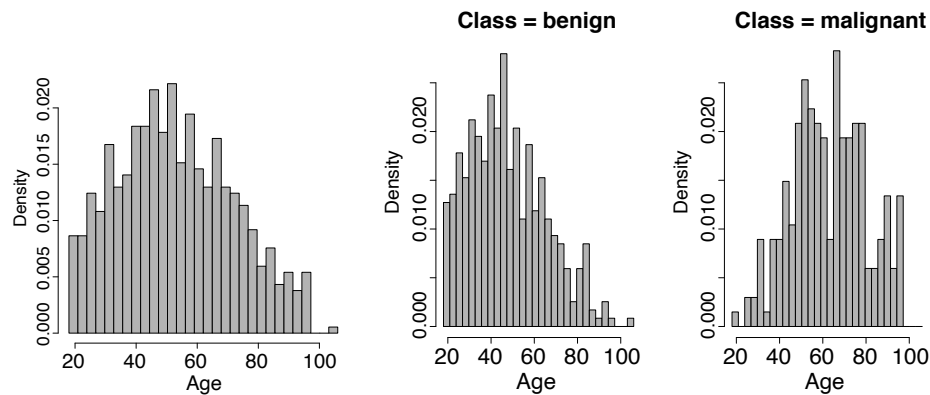
The target feature `CLASS` is also imbalanced with approximately 60% of instances representing *benign* samples and 40% *malignant* samples. This is much higher than the incidence of breast cancer in the population—approximately 13% of women will develop breast cancer in their lifetimes (this is much lower for men).^b This difference in target distribution between a dataset and a population is not unusual, however, as the data has been generated through a screening process which is likely to have pre-selected people with a higher risk.

^a. Based on data from ?.

^b. Based on data from ?.

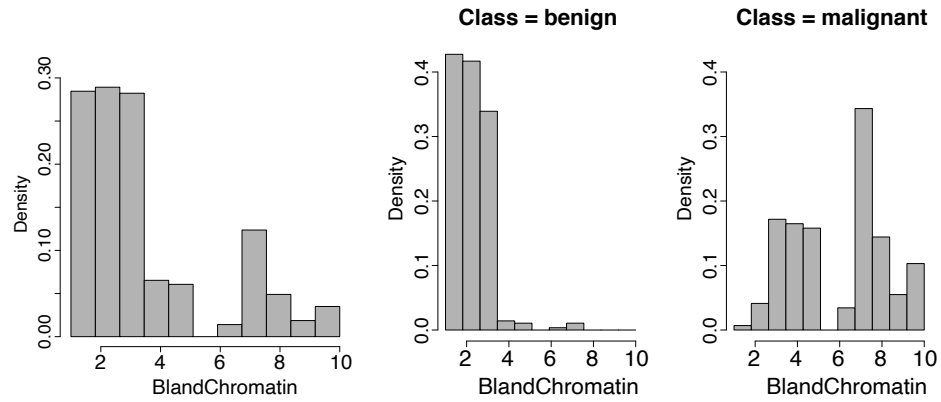
- * 12. The following data visualizations are based on the breast cancer prediction dataset from Question 11 (after some data quality issues present in the dataset have been corrected). Each visualization illustrates the relationship between a descriptive feature and the target feature, CLASS and is composed of three plots: a plot of the distribution of the descriptive feature values in the full dataset, and plots showing the distribution of the descriptive feature values for each level of the target. Discuss the relationships shown in each visualizations.

- (a) The visualization below illustrates the relationship between the continuous feature AGE and the target feature, CLASS.



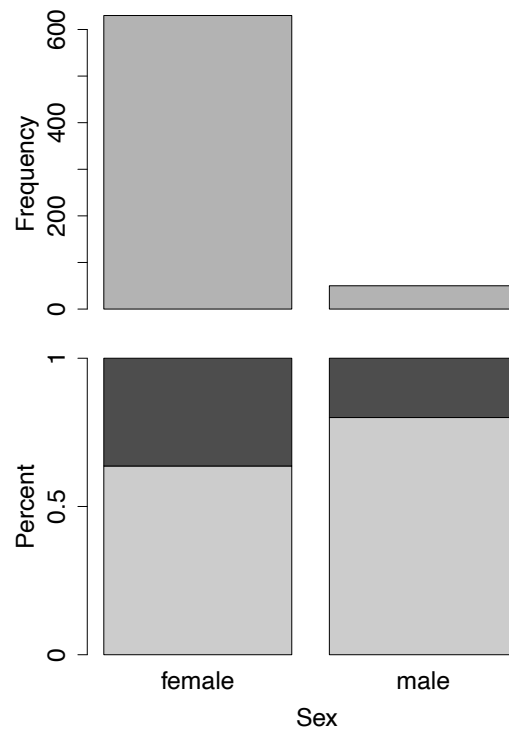
The histogram for the different levels of the CLASS feature are slightly different suggesting that there is a relationship between the AGE and CLASS features. It looks like samples from older people are more likely to be *malignant* than those from younger people. This though is not a perfect predictor as there is a good degree of overlap between the ages of the people to whom *benign* and *malignant* samples belong to.

- (b) The visualization below illustrates the relationship between the continuous BLAND-CHROMATIN feature and the target feature CLASS.



The histograms for the BLANDCHROMATIN feature split according to the different target levels are quite different suggesting a strong relationship between this descriptive feature and the target feature. It looks like *malignant* samples are likely to have higher values of BLANDCHROMATIN. This though is not a perfect predictor as there is a good degree of overlap between the values for *benign* and *malignant* samples.

- (c) The visualization below illustrates the relationship between the categorical feature SEX and the target feature, CLASS.



This visualisation suggests that there is a strong relationship between the SEX and CLASS features. The feature suggests that samples from men are much less likely to be *malignant* than samples from women.

II PREDICTIVE DATA ANALYTICS

4 Information-Based Learning (Exercise Solutions)

1. The image below shows a set of eight Scrabble pieces.



- (a) What is the **entropy** in bits of the letters in this set?

We can calculate the probability of randomly selecting a letter of each type from this set: $P(O) = \frac{3}{8}$, $P(X) = \frac{1}{8}$, $P(Y) = \frac{1}{8}$, $P(M) = \frac{1}{8}$, $P(R) = \frac{1}{8}$, $P(N) = \frac{1}{8}$.

Using these probabilities, we can calculate the entropy of the set:

$$-\left(\frac{3}{8} \times \log_2\left(\frac{3}{8}\right) + \left(\frac{1}{8} \times \log_2\left(\frac{1}{8}\right)\right) \times 5\right) = 2.4056 \text{ bits}$$

Note that the contribution to the entropy for any letter that appears only once is the same and so has been included 5 times—once for each of X , Y , M , R , and N .

- (b) What would be the reduction in entropy (i.e., the **information gain**) in bits if we split these letters into two sets, one containing the vowels and the other containing the consonants?

Information gain is the reduction in entropy that occurs after we split the original set. We know that the entropy of the initial set is 2.4056 *bits*. We calculate the remaining entropy after we split the original set using a weighted

summation of the entropies for the new sets. The two new sets are vowels {O,O,O} and consonants {X,Y,M,R,N}.

The entropy of the vowel set is

$$-\left(\frac{3}{3} \times \log_2 \left(\frac{3}{3}\right)\right) = 0 \text{ bits}$$

The entropy of the consonant set is

$$-\left(\left(\frac{1}{5} \times \log_2 \left(\frac{1}{5}\right)\right) \times 5\right) = 2.3219 \text{ bits}$$

The weightings used in the summation of the set entropies are just the relative size of each set. So, the weighting of the vowel set entropy is $\frac{3}{8}$, and the weighting of the consonant set entropy is $\frac{5}{8}$.

This gives the entropy remaining after we split the set of letters into vowels and consonants as

$$rem = \frac{3}{8} \times 0 + \frac{5}{8} \times 2.3219 = 1.4512 \text{ bits}$$

The information gain is the difference between the initial entropy and the remainder:

$$IG = 2.4056 - 1.4512 = 0.9544 \text{ bits}$$

- (c) What is the maximum possible entropy in bits for a set of eight Scrabble pieces?

The maximum entropy occurs when there are eight different letters in the set. The entropy for a set with this distribution of letters is

$$\left(\frac{1}{8} \times \log_2 \left(\frac{1}{8}\right)\right) \times 8 = 3 \text{ bits}$$

- (d) In general, which is preferable when you are playing Scrabble: a set of letters with high entropy or a set of letters with low entropy?

In general, sets of letters with high entropy are preferable to lower entropy sets because the more diverse the letters in the set, the more words you are likely to be able to make from the set.

2. A convicted criminal who reoffends after release is known as a *recidivist*. The following table lists a dataset that describes prisoners released on parole and whether they reoffended within two years of release.¹

ID	GOOD BEHAVIOR	AGE < 30	DRUG DEPENDENT	RECIDIVIST
1	false	true	false	true
2	false	false	false	false
3	false	true	false	true
4	true	false	false	false
5	true	false	true	true
6	true	false	false	false

This dataset lists six instances in which prisoners were granted parole. Each of these instances is described in terms of three binary descriptive features (GOOD BEHAVIOR, AGE < 30, DRUG DEPENDENT) and a binary target feature (RECIDIVIST). The GOOD BEHAVIOR feature has a value of *true* if the prisoner had not committed any infringements during incarceration, the AGE < 30 has a value of *true* if the prisoner was under 30 years of age when granted parole, and the DRUG DEPENDENT feature is *true* if the prisoner had a drug addiction at the time of parole. The target feature, RECIDIVIST, has a *true* value if the prisoner was arrested within two years of being released; otherwise it has a value of *false*.

- (a) Using this dataset, construct the decision tree that would be generated by the **ID3** algorithm, using entropy-based information gain.

The first step in building the decision tree is to figure out which of the three descriptive features is the best one on which to split the dataset at the root node (i.e., which descriptive feature has the highest information gain). The total entropy for this dataset is computed as follows:

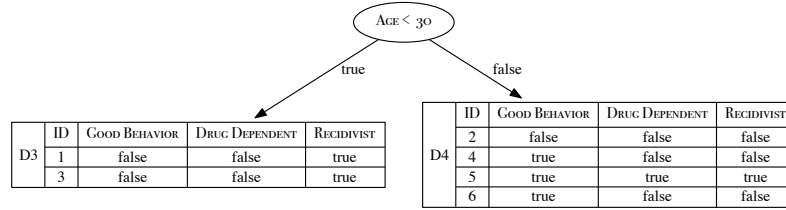
$$\begin{aligned}
 H(\text{RECIDIVIST}, \mathcal{D}) &= - \sum_{l \in \{\text{true}, \text{false}\}} P(\text{RECIDIVIST} = l) \times \log_2(P(\text{RECIDIVIST} = l)) \\
 &= - \left(\left(\frac{3}{6} \times \log_2\left(\frac{3}{6}\right) \right) + \left(\frac{3}{6} \times \log_2\left(\frac{3}{6}\right) \right) \right) = 1.00 \text{ bit}
 \end{aligned}$$

1. This example of predicting recidivism is based on a real application of machine learning: parole boards do rely on machine learning prediction models to help them when they are making their decisions. See ? for a recent comparison of different machine learning models used for this task. Datasets dealing with prisoner recidivism are available online, for example, catalog.data.gov/dataset/prisoner-recidivism/. The dataset presented here is not based on real data.

The table below illustrates the computation of the information gain for each of the descriptive features:

Split by Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
GOOD BEHAVIOR	<i>true</i>	\mathcal{D}_1	$\mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.9183	0.9183	0.0817
	<i>false</i>	\mathcal{D}_2	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$	0.9183		
AGE < 30	<i>true</i>	\mathcal{D}_3	$\mathbf{d}_1, \mathbf{d}_3$	0	0.5409	0.4591
	<i>false</i>	\mathcal{D}_4	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.8113		
DRUG DEPENDENT	<i>true</i>	\mathcal{D}_5	\mathbf{d}_5	0	0.8091	0.1909
	<i>false</i>	\mathcal{D}_6	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6$	0.9709		

AGE < 30 has the largest information gain of the three features. Consequently, this feature will be used at the root node of the tree. The figure below illustrates the state of the tree after we have created the root node and split the data based on AGE < 30.



In this image we have shown how the data moves down the tree based on the split on the AGE < 30 feature. Note that this feature no longer appears in these datasets because we cannot split on it again.

The dataset on the left branch contains only instances where RECIDIVIST is *true* and so does not need to be split any further.

The dataset on the right branch of the tree (\mathcal{D}_4) is not homogenous, so we need to grow this branch of the tree. The entropy for this dataset, \mathcal{D}_4 , is calculated as follows:

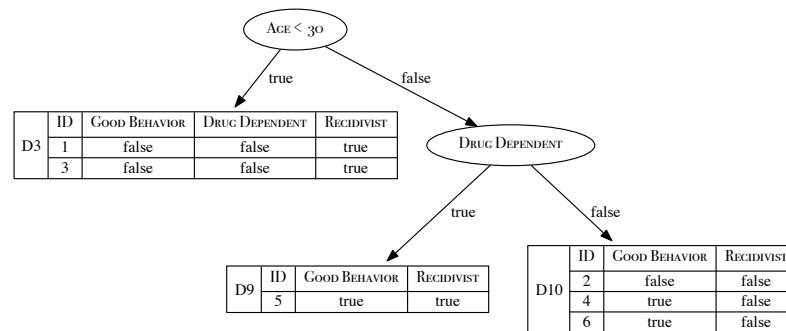
$$\begin{aligned}
 & H(\text{RECIDIVIST}, \mathcal{D}_4) \\
 &= - \sum_{l \in \{\text{true}, \text{false}\}} P(\text{RECIDIVIST} = l) \times \log_2(P(\text{RECIDIVIST} = l)) \\
 &= - \left(\left(\frac{1}{4} \times \log_2\left(\frac{1}{4}\right) \right) + \left(\frac{3}{4} \times \log_2\left(\frac{3}{4}\right) \right) \right) = 0.8113 \text{ bits}
 \end{aligned}$$

The table below shows the computation of the information gain for the GOOD BEHAVIOR and DRUG DEPENDENT features in the context of the \mathcal{D}_4 dataset:

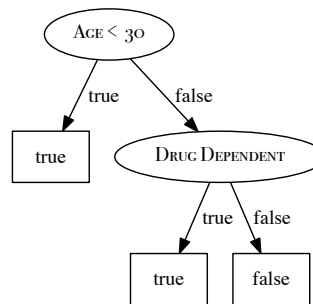
Split by Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
GOOD BEHAVIOR	<i>true</i>	\mathcal{D}_7	$\mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.918295834	0.4591	0.3522
DRUG DEPENDENT	<i>false</i>	\mathcal{D}_8	\mathbf{d}_2	0		
DRUG DEPENDENT	<i>true</i>	\mathcal{D}_9	\mathbf{d}_5	0	0	0.8113
DRUG DEPENDENT	<i>false</i>	\mathcal{D}_{10}	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_6$	0		

These calculations show that the DRUG DEPENDENT feature has a higher information gain than GOOD BEHAVIOR: 0.8113 versus 0.3522 and so should be chosen for the next split.

The image below shows the state of the decision tree after the \mathcal{D}_4 partition has been split based on the feature DRUG DEPENDENT.



All the datasets at the leaf nodes are now pure, so the algorithm will stop growing the tree. The image below shows the tree that will be returned by the ID3 algorithm:



- (b) What prediction will the decision tree generated in Part (a) of this question return for the following query?

GOOD BEHAVIOR = *false*, AGE < 30 = *false*,
DRUG DEPENDENT = *true*

RECIDIVIST = <i>true</i>

- (c) What prediction will the decision tree generated in Part (a) of this question return for the following query?

GOOD BEHAVIOR = *true*, AGE < 30 = *true*,
 DRUG DEPENDENT = *false*

RECIDIVIST = <i>true</i>

3. The following table lists a sample of data from a census.²

ID	AGE	EDUCATION	MARITAL STATUS	OCCUPATION	ANNUAL INCOME
1	39	bachelors	never married	transport	25K–50K
2	50	bachelors	married	professional	25K–50K
3	18	high school	never married	agriculture	<25K
4	28	bachelors	married	professional	25K–50K
5	37	high school	married	agriculture	25K–50K
6	24	high school	never married	armed forces	<25K
7	52	high school	divorced	transport	25K–50K
8	40	doctorate	married	professional	>50K

There are four descriptive features and one target feature in this dataset, as follows:

- AGE, a continuous feature listing the age of the individual;
- EDUCATION, a categorical feature listing the highest education award achieved by the individual (*high school*, *bachelors*, *doctorate*);
- MARITAL STATUS (*never married*, *married*, *divorced*);
- OCCUPATION (*transport* = works in the transportation industry; *professional* = doctor, lawyer, or similar; *agriculture* = works in the agricultural industry; *armed forces* = is a member of the armed forces); and
- ANNUAL INCOME, the target feature with 3 levels (<25K, 25K–50K, >50K).

- (a) Calculate the **entropy** for this dataset.

2. This census dataset is based on the Census Income Dataset (?), which is available from the UCI Machine Learning Repository (?) at archive.ics.uci.edu/ml/datasets/Census+Income/.

$$\begin{aligned}
H(\text{ANNUAL INCOME}, \mathcal{D}) &= - \sum_{l \in \left\{ \begin{smallmatrix} <25K, \\ 25K-50K, \\ >50K \end{smallmatrix} \right\}} P(\text{AN. INC.} = l) \times \log_2(P(\text{AN. INC.} = l)) \\
&= - \left(\left(\frac{2}{8} \times \log_2 \left(\frac{2}{8} \right) \right) + \left(\frac{5}{8} \times \log_2 \left(\frac{5}{8} \right) \right) + \left(\frac{1}{8} \times \log_2 \left(\frac{1}{8} \right) \right) \right) \\
&= 1.2988 \text{ bits}
\end{aligned}$$

- (b) Calculate the **Gini index** for this dataset.

$$\begin{aligned}
\text{Gini}(\text{ANNUAL INCOME}, \mathcal{D}) &= 1 - \sum_{l \in \left\{ \begin{smallmatrix} <25K, \\ 25K-50K, \\ >50K \end{smallmatrix} \right\}} P(\text{AN. INC.} = l)^2 \\
&= 1 - \left(\left(\frac{2}{8} \right)^2 + \left(\frac{5}{8} \right)^2 + \left(\frac{1}{8} \right)^2 \right) = 0.5313
\end{aligned}$$

- (c) In building a decision tree, the easiest way to handle a continuous feature is to define a threshold around which splits will be made. What would be the optimal threshold to split the continuous AGE feature (use information gain based on entropy as the feature selection measure)?

First sort the instances in the dataset according to the AGE feature, as shown in the following table.

ID	AGE	ANNUAL INCOME
3	18	<25K
6	24	<25K
4	28	25K-50K
5	37	25K-50K
1	39	25K-50K
8	40	>50K
2	50	25K-50K
7	52	25K-50K

Based on this ordering, the mid-points in the AGE values of instances that are adjacent in the new ordering but that have different target levels define the possible threshold points. These points are 26, 39.5, and 45.

We calculate the information gain for each of these possible threshold points using the entropy value we calculated in part (a) of this question (1.2988 bits) as follows:

Split by Feature	Partition	Instances	Partition Entropy	Rem.	Info. Gain
>26	\mathcal{D}_1	$\mathbf{d}_3, \mathbf{d}_6$	0	0.4875	0.8113
	\mathcal{D}_2	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_7, \mathbf{d}_8$	0.6500		
>39.5	\mathcal{D}_3	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.9710	0.9456	0.3532
	\mathcal{D}_4	$\mathbf{d}_2, \mathbf{d}_7, \mathbf{d}_8$	0.9033		
>45	\mathcal{D}_5	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6, \mathbf{d}_8$	1.4591	1.0944	0.2044
	\mathcal{D}_6	$\mathbf{d}_2, \mathbf{d}_7$	0		

The threshold AGE > 26 has the highest information gain, and consequently, it is the best threshold to use if we are splitting the dataset using the AGE feature.

- (d) Calculate **information gain** (based on entropy) for the EDUCATION, MARITAL STATUS, and OCCUPATION features.

We have already calculated the entropy for the full dataset in part (a) of this question as 1.2988 bits. The table below lists the rest of the calculations for the information gain for the EDUCATION, MARITAL STATUS, and OCCUPATION features.

Split by Feature	Level	Instances	Partition Gini Index	Rem.	Info. Gain
EDUCATION	<i>high school</i>	$\mathbf{d}_3, \mathbf{d}_5, \mathbf{d}_6, \mathbf{d}_7$	1.0	0.5	0.7988
	<i>bachelors</i>	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$	0		
	<i>doctorate</i>	\mathbf{d}_8	0		
MARITAL STATUS	<i>never married</i>	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_6$	0.9183	0.75	0.5488
	<i>married</i>	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_8$	0.8113		
	<i>divorced</i>	\mathbf{d}_7	0		
OCCUPATION	<i>transport</i>	$\mathbf{d}_1, \mathbf{d}_7$	0	0.5944	0.7044
	<i>professional</i>	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_8$	0.9183		
	<i>agriculture</i>	$\mathbf{d}_3, \mathbf{d}_5$	1.0		
	<i>armed forces</i>	\mathbf{d}_6	0		

- (e) Calculate the **information gain ratio** (based on entropy) for EDUCATION, MARITAL STATUS, and OCCUPATION features.

In order to calculate the information gain ratio of a feature, we divide the information gain of the feature by the entropy of the feature itself. We have

already calculated the information gain of these features in the preceding part of this question:

- $IG(EDUCATION, \mathcal{D}) = 0.7988$
- $IG(MARITAL\ STATUS, \mathcal{D}) = 0.5488$
- $IG(OCCUPATION, \mathcal{D}) = 0.7044$

We calculate the entropy of each feature as follows:

$$\begin{aligned}
 &H(EDUCATION, \mathcal{D}) \\
 &= - \sum_{l \in \left\{ \begin{smallmatrix} high\ school, \\ bachelors, \\ doctorate \end{smallmatrix} \right\}} P(ED. = l) \times \log_2(P(ED. = l)) \\
 &= - \left(\left(\frac{4}{8} \times \log_2 \left(\frac{4}{8} \right) \right) + \left(\frac{3}{8} \times \log_2 \left(\frac{3}{8} \right) \right) + \left(\frac{1}{8} \times \log_2 \left(\frac{1}{8} \right) \right) \right) \\
 &= 1.4056 \text{ bits}
 \end{aligned}$$

$$\begin{aligned}
 &H(MARITAL\ STATUS, \mathcal{D}) \\
 &= - \sum_{l \in \left\{ \begin{smallmatrix} never\ married, \\ married, \\ divorced \end{smallmatrix} \right\}} P(MAR. STAT. = l) \times \log_2(P(MAR. STAT. = l)) \\
 &= - \left(\left(\frac{3}{8} \times \log_2 \left(\frac{3}{8} \right) \right) + \left(\frac{4}{8} \times \log_2 \left(\frac{4}{8} \right) \right) + \left(\frac{1}{8} \times \log_2 \left(\frac{1}{8} \right) \right) \right) \\
 &= 1.4056 \text{ bits}
 \end{aligned}$$

$$\begin{aligned}
 &H(OCCUPATION, \mathcal{D}) \\
 &= - \sum_{l \in \left\{ \begin{smallmatrix} transport, \\ professional, \\ agriculture, \\ armed\ forces \end{smallmatrix} \right\}} P(OCC. = l) \times \log_2(P(OCC. = l)) \\
 &= - \left(\left(\frac{2}{8} \times \log_2 \left(\frac{2}{8} \right) \right) + \left(\frac{3}{8} \times \log_2 \left(\frac{3}{8} \right) \right) \right. \\
 &\quad \left. + \left(\frac{2}{8} \times \log_2 \left(\frac{2}{8} \right) \right) + \left(\frac{1}{8} \times \log_2 \left(\frac{1}{8} \right) \right) \right) \\
 &= 1.9056 \text{ bits}
 \end{aligned}$$

We can now calculate the information gain ratio for each feature as:

- $GR(EDUCATION, \mathcal{D}) = \frac{0.7988}{1.4056} = 0.5683$

$$\begin{aligned} \bullet \text{ GR(MARITAL STATUS, } \mathcal{D}) &= \frac{0.5488}{1.4056} = 0.3904 \\ \bullet \text{ GR(OCCUPATION, } \mathcal{D}) &= \frac{0.7044}{1.9056} = 0.3696 \end{aligned}$$

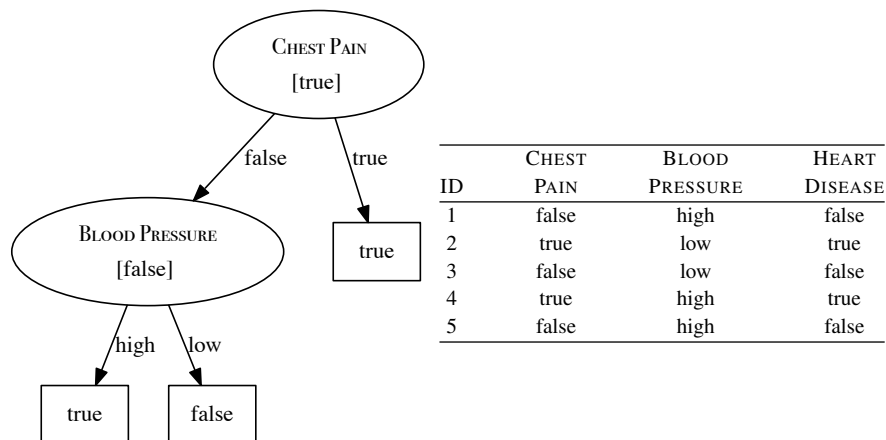
- (f) Calculate **information gain** using the **Gini index** for the EDUCATION, MARITAL STATUS, and OCCUPATION features.

We have already calculated the Gini index for the full dataset in part (b) of this question as 0.5313. The table below lists the rest of the calculations of information gain for the EDUCATION, MARITAL STATUS, and OCCUPATION features.

Split by Feature	Level	Instances	Partition Gini Index	Rem.	Info. Gain
EDUCATION	<i>high school</i>	d₃, d₅, d₆, d₇	0.5	0.25	0.2813
	<i>bachelors</i>	d₁, d₂, d₃	0		
	<i>doctorate</i>	d₈	0		
MARITAL STATUS	<i>never married</i>	d₁, d₃, d₆	0.4444	0.3542	0.1771
	<i>married</i>	d₂, d₄, d₅, d₈	0.375		
	<i>divorced</i>	d₇	0		
OCCUPATION	<i>transport</i>	d₁, d₇	0	0.2917	0.2396
	<i>professional</i>	d₂, d₄, d₈	0.4444		
	<i>agriculture</i>	d₃, d₅	0.5		
	<i>armed forces</i>	d₆	0		

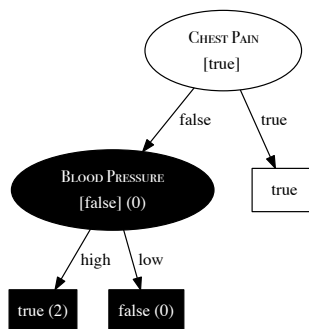
4. The following diagram shows a decision tree for the task of predicting heart disease.³ The descriptive features in this domain describe whether the patient suffers from chest pain (CHEST PAIN) and the blood pressure of the patient (BLOOD PRESSURE). The binary target feature is HEART DISEASE. The table beside the diagram lists a pruning set from this domain.

3. This example is inspired by the research reported in ?.



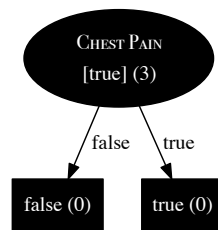
Using the pruning set, apply **reduced error pruning** to the decision tree. Assume that the algorithm is applied in a bottom-up, left-to-right fashion. For each iteration of the algorithm, indicate the subtrees considered as pruning candidates, explain why the algorithm chooses to prune or leave these subtrees in the tree, and illustrate the tree that results from each iteration.

The first subtree that will be considered for pruning by the algorithm is the subtree under the blood pressure node. The nodes colored in black in the figure below illustrate the extent of this subtree. For each node, the value given in square brackets is the majority target level returned at that node, and the number in round brackets is the number of errors in the pruning set made as a result of predictions returned from this node.



The root node of this subtree returns *false*, and this results in 0 errors in the pruning set. The sum of the errors for the two leaf nodes of this subtree is 2. The algorithm will prune this subtree because the number of errors resulting from the leaf nodes is higher than the number of errors resulting from the root node.

The figure below illustrates the structure of the tree after the subtree under the BLOOD PRESSURE node is pruned. This figure also highlights the extent of the subtree that is considered for pruning in the second iteration of the algorithm (the entire tree in this case).



The root node of this tree returns *true* as a prediction, and consequently, it results in 3 errors on the pruning set. By contrast the number of errors made at each of the leaf nodes of this tree is 0. Because the number of errors at the leaf nodes is less than the number of errors at the root node, this tree will not be pruned. At this point all the non-leaf nodes in the tree have been tested, so the pruning algorithm will stop, and this decision tree is the one that is returned by the pruning algorithm.

5. The following table⁴ lists a dataset containing the details of five participants in a heart disease study, and a target feature RISK, which describes their risk of heart disease. Each patient is described in terms of four binary descriptive features

- EXERCISE, how regularly do they exercise
- SMOKER, do they smoke
- OBESE, are they overweight
- FAMILY, did any of their parents or siblings suffer from heart disease

4. The data in this table has been artificially generated for this question, but is inspired by the results from the Framingham Heart Study: www.framinghamheartstudy.org.

ID	EXERCISE	SMOKER	OBESE	FAMILY	RISK
1	daily	false	false	yes	low
2	weekly	true	false	yes	high
3	daily	false	false	no	low
4	rarely	true	true	yes	high
5	rarely	true	true	no	high

- (a) As part of the study, researchers have decided to create a predictive model to screen participants based on their risk of heart disease. You have been asked to implement this screening model using a **random forest**. The three tables below list three bootstrap samples that have been generated from the above dataset. Using these bootstrap samples, create the decision trees that will be in the random forest model (use entropy-based information gain as the feature selection criterion).

ID	EXERCISE	FAMILY	RISK
1	daily	yes	low
2	weekly	yes	high
2	weekly	yes	high
5	rarely	no	high
5	rarely	no	high

Bootstrap Sample A

ID	SMOKER	OBESE	RISK
1	false	false	low
2	true	false	high
2	true	false	high
4	true	true	high
5	true	true	high

Bootstrap Sample B

ID	OBESE	FAMILY	RISK
1	false	yes	low
1	false	yes	low
2	false	yes	high
4	true	yes	high
5	true	no	high

Bootstrap Sample C

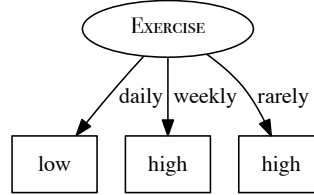
The entropy calculation for Bootstrap Sample A is:

$$\begin{aligned}
 & H(\text{RISK}, \text{BootstrapSampleA}) \\
 &= - \sum_{l \in \{\text{low}, \text{high}\}} P(\text{RISK} = l) \times \log_2(P(\text{RISK} = l)) \\
 &= - \left(\left(\frac{1}{5} \times \log_2 \left(\frac{1}{5} \right) \right) + \left(\frac{4}{5} \times \log_2 \left(\frac{4}{5} \right) \right) \right) \\
 &= 0.7219 \text{ bits}
 \end{aligned}$$

The information gain for each of the features in Bootstrap Sample A is as follows:

Split by Feature	Level	Instances	Partition Entropy	Rem.	Info. Gain
EXERCISE	<i>daily</i>	d₁	0	0	0.7219
	<i>weekly</i>	d₂, d₂	0		
	<i>rarely</i>	d₅, d₅	0		
FAMILY	<i>yes</i>	d₁, d₂, d₂	0.9183	0.5510	0.1709
	<i>no</i>	d₅, d₅	0		

These calculations show that the EXERCISE feature has the highest information gain of the descriptive features in Bootstrap Sample A and should be added as the root node of the decision tree generated from Bootstrap Sample A. What is more, splitting on EXERCISE generates pure sets. So, the decision tree does not need to be expanded beyond this initial test and the final tree generated for Bootstrap Sample A will be as shown below.



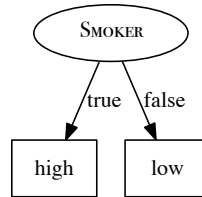
By chance, Bootstrap Sample B has the same distribution of target feature values as Bootstrap Sample A, so the entropy calculation for Bootstrap Sample B is the same as the calculation for Bootstrap Sample A:

$$\begin{aligned}
 & H(\text{RISK}, \text{BootstrapSampleB}) \\
 &= - \sum_{l \in \{\text{low}, \text{high}\}} P(\text{RISK} = l) \times \log_2(P(\text{RISK} = l)) \\
 &= - \left(\left(\frac{1}{5} \times \log_2 \left(\frac{1}{5} \right) \right) + \left(\frac{4}{5} \times \log_2 \left(\frac{4}{5} \right) \right) \right) \\
 &= 0.7219 \text{ bits}
 \end{aligned}$$

The information gain for each of the features in Bootstrap Sample B is as follows:

Split by Feature	Level	Instances	Partition Entropy	Rem.	Info. Gain
SMOKER	<i>true</i>	d₂, d₂, d₄, d₅	0	0	0.7219
	<i>false</i>	d₁	0		
OBESE	<i>true</i>	d₄, d₅	0	0.5510	0.1709
	<i>false</i>	d₁, d₂, d₂	0.9183		

These calculations show that the SMOKER feature has the highest information gain of the descriptive features in Bootstrap Sample B and should be added as the root node of the decision tree generated from Bootstrap Sample B. What is more, splitting on SMOKER generates pure sets, So the decision tree does not need to be expanded beyond this initial test. The final tree generated for Bootstrap Sample B is shown below.



The entropy calculation for Bootstrap Sample C is:

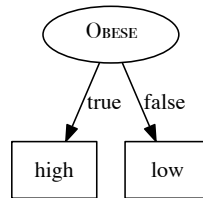
$$\begin{aligned}
 & H(\text{RISK}, \text{BootstrapSampleC}) \\
 &= - \sum_{l \in \{\text{low}, \text{high}\}} P(\text{RISK} = l) \times \log_2(P(\text{RISK} = l)) \\
 &= - \left(\left(\frac{2}{5} \times \log_2 \left(\frac{2}{5} \right) \right) + \left(\frac{3}{5} \times \log_2 \left(\frac{3}{5} \right) \right) \right) \\
 &= 0.9710 \text{ bits}
 \end{aligned}$$

The information gain for each of the features in Bootstrap Sample C is as follows:

Split by Feature	Level	Instances	Partition Entropy	Rem.	Info. Gain
OBESE	<i>true</i>	d₄, d₅	0	0.5510	0.4200
	<i>false</i>	d₁, d₁, d₂	0.9183		
FAMILY	<i>yes</i>	d₁, d₁, d₂, d₄	1.0	0.8	0.1709
	<i>no</i>	d₅	0		

These calculations show that the OBESE feature has the highest information gain of the descriptive features in Bootstrap Sample C and should be added as the root node of the decision tree generated from Bootstrap Sample C. Splitting Bootstrap Sample C creates one pure partition for OBESE=*true* (**d₄, d₅**) where all the instances have RISK=*high*, and an impure partition for OBESE=*false* where two instances (**d₁, d₁**) have RISK=*low* and for one instance (**d₂**) RISK=*high*.

Normally this would mean that we would continue to split the impure partition to create pure sets. However, in this instance there is only one feature that we can still use to split this partition, the FAMILY feature, and all the instances in this partition have the same level for this feature FAMILY=yes. Consequently, instead of splitting this partition further we simply create a leaf node with the majority target level within the partition: RISK=low. So, the final tree generated for Bootstrap Sample C will be as shown below.



- (b) Assuming the random forest model you have created uses majority voting, what prediction will it return for the following query:

EXERCISE=rarely, SMOKER=false, OBESE=true, FAMILY=yes

Each of the trees in the ensemble will vote as follows:

- Tree 1: EXERCISE=rarely → RISK=high
- Tree 2: SMOKER=false → RISK=low
- Tree 3: OBESE=true → RISK=high

So, the majority vote is for RISK=high, and this is the prediction the model will return for this query.

- * 6. The following table lists a dataset containing the details of six patients. Each patient is described in terms of three binary descriptive features (OBESE, SMOKER, and DRINKS ALCOHOL) and a target feature (CANCER RISK).⁵

ID	OBESE	SMOKER	DRINKS ALCOHOL	CANCER RISK
1	true	false	true	low
2	true	true	true	high
3	true	false	true	low
4	false	true	true	high
5	false	true	false	low
6	false	true	true	high

- (a) Which of the descriptive features will the **ID3** decision tree induction algorithm choose as the feature for the root node of the decision tree?

The ID3 decision tree induction algorithm selects the descriptive feature with the highest information gain at the root node of the decision tree. The first step in calculating information gain is to calculate the entropy for the entire dataset \mathcal{D} with respect to the target feature, CANCER RISK:

$$\begin{aligned}
 H(\text{CANCER RISK}, \mathcal{D}) &= - \sum_{l \in \{\text{high}, \text{low}\}} P(\text{CANCER RISK} = l) \times \log_2(P(\text{CANCER RISK} = l)) \\
 &= - \left(\left(\frac{3}{6} \times \log_2 \left(\frac{3}{6} \right) \right) + \left(\frac{3}{6} \times \log_2 \left(\frac{3}{6} \right) \right) \right) \\
 &= 1.00 \text{ bits}
 \end{aligned}$$

The table below shows the calculation of the information gain for each of the descriptive features in the dataset:

5. The data in this table has been artificially generated for this question. The American Cancer Society does, however, provide information on the causes of cancer: www.cancer.org/cancer/cancercauses/.

Split by Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
OBESE	<i>true</i>	DS_1	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$	0.9183	0.9183	0.0817
	<i>false</i>	DS_2	$\mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.9183		
SMOKER	<i>true</i>	DS_3	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.8113	0.5409	0.4591
	<i>false</i>	DS_4	$\mathbf{d}_1, \mathbf{d}_3$	0		
ALCOHOL	<i>true</i>	DS_5	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6$	0.9709	0.8091	0.1909
	<i>false</i>	DS_6	\mathbf{d}_5	0		

From this table, we can see that the feature SMOKER has the highest information gain, and consequently the ID3 algorithm will choose this feature as the one to be tested at the root node of the tree.

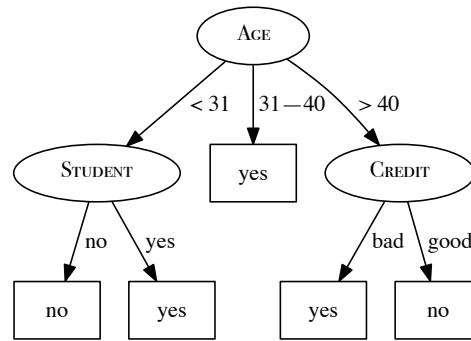
- (b) In designing a dataset, it is generally a bad idea if all the descriptive features are indicators of the target feature taking a particular value. For example, a potential criticism of the design of the dataset in this question is that all the descriptive features are indicators of the CANCER RISK target feature taking the same level, *high*. Can you think of any descriptive features that could be added to this dataset that are indicators of the *low* target level?

Being physically active and having a healthy diet—for example, eating vegetables and fruit—have been linked with reducing the risk of cancer. So the dataset could be extended to include features that capture the activity and diet of patients.

- * 7. The following table lists a dataset collected in an electronics shop showing details of customers and whether they responded to a special offer to buy a new laptop.

ID	AGE	INCOME	STUDENT	CREDIT	BUYS
1	< 31	high	no	bad	no
2	< 31	high	no	good	no
3	31 – 40	high	no	bad	yes
4	> 40	med	no	bad	yes
5	> 40	low	yes	bad	yes
6	> 40	low	yes	good	no
7	31 – 40	low	yes	good	yes
8	< 31	med	no	bad	no
9	< 31	low	yes	good	yes
10	> 40	med	yes	bad	yes
11	< 31	med	yes	good	yes
12	31 – 40	med	no	good	yes
13	31 – 40	high	yes	bad	yes
14	> 40	med	no	good	no

This dataset has been used to build a decision tree to predict which customers will respond to future special offers. The decision tree, created using the **ID3** algorithm, is the following:



- (a) The **information gain** (calculated using entropy) of the feature AGE at the root node of the tree is 0.247. A colleague has suggested that the STUDENT feature would be better at the root node of the tree. Show that this is not the case.

To answer this question we need to calculate the information gain of the STUDENT feature at the root node and show that it is less than the information gain for AGE.

To calculate information gain for the STUDENT feature we first calculate the entropy of the entire dataset \mathcal{D} with respect to the target feature, BUYS

$$\begin{aligned}
 & H(\text{BUYS}, \mathcal{D}) \\
 &= - \sum_{l \in \{\text{yes}, \text{no}\}} P(\text{BUYS} = l) \times \log_2(P(\text{BUYS} = l)) \\
 &= - \left(\left(\frac{9}{14} \times \log_2 \left(\frac{9}{14} \right) \right) + \left(\frac{5}{14} \times \log_2 \left(\frac{5}{14} \right) \right) \right) \\
 &= 0.9403 \text{ bits}
 \end{aligned}$$

Next we calculate the remainder after the dataset is split based on the values of the STUDENT feature.

Split by Feature	Feature Value	Partition	Examples	Partition Entropy	Remainder
STUDENT	yes	DS_1	5,6,7,9,10,11,13	0.5917	0.7885
	no	DS_2	1,2,3,4,8,12,14	0.9852	

Finally, we calculate the information gain as the difference between the original entropy and the remainder:

$$IG = 0.9403 - 0.7885 = 0.1518 \text{ bits}$$

The information gain for STUDENT is 0.1518 which is less than the 0.247 for AGE, so STUDENT is not a better feature to use at the root node.

- (b) Yet another colleague has suggested that the ID feature would be a very effective at the root node of the tree. Would you agree with this suggestion?

There is no need to calculate the information gain for the ID feature as the result will work out to be 0.9403—the total information required to make a prediction (and a value calculated in the answer to previous part of the question). We can know this without performing the calculation, however, because each instance has a unique value for the ID feature.

Because of this we know that ID would not be a good feature at the root node of the tree (or in fact anywhere in the tree) because it actually contains no information, and the resulting decision tree would be massively overfitted to the training data. Information measures such as the gain ratio are designed to address this limitation in information gain.

- * 8. This table lists a dataset of the scores students achieved on an exam described in terms of whether the student studied for the exam (STUDIED) and the energy level of the lecturer when grading the student's exam (ENERGY).

ID	STUDIED	ENERGY	SCORE
1	yes	tired	65
2	no	alert	20
3	yes	alert	90
4	yes	tired	70
5	no	tired	40
6	yes	alert	85
7	no	tired	35

Which of the two descriptive features should we use as the testing criterion at the root node of a decision tree to predict students' scores?

The target feature in this question (SCORE) is continuous. When a decision tree is predicting a continuous target, we choose as the descriptive feature to use at each node in the tree the one that results in the minimum weighted variance after the dataset has been split based on that feature. The table below shows the

calculation of the weighted variance for each of the descriptive features in this domain.

Split by Feature	Level	Part.	Instances	$P(d = l)$	$var(t, \mathcal{D})$	Weighted Variance
STUDIED	<i>yes</i>	\mathcal{D}_1	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6$	$\frac{4}{7}$	$141\frac{2}{3}$	127.3810
	<i>no</i>	\mathcal{D}_2	$\mathbf{d}_2, \mathbf{d}_5, \mathbf{d}_7$	$\frac{3}{7}$	$108\frac{1}{3}$	
ENERGY	<i>alert</i>	\mathcal{D}_5	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6$	$\frac{3}{7}$	1525	829.7619
	<i>tired</i>	\mathcal{D}_6	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_7$	$\frac{4}{7}$	$308\frac{1}{3}$	

From these calculations we can see that splitting the dataset using the STUDIED feature results in the lowest weighted variance. Consequently, we should use the STUDIED feature at the root node of the tree.

- * 9. Calculate the probability of a **model ensemble** that uses simple majority voting making an incorrect prediction in the following scenarios. (Hint: Understanding how to use the **binomial distribution** will be useful in answering this question.)

- (a) The ensemble contains 11 independent models, all of which have an error rate of 0.2.

Because majority voting is being used, the ensemble model will make an incorrect prediction for a query when a majority, in this case 6 or more, of the individual models make an incorrect prediction. So to calculate the overall probability of the model making an incorrect prediction, we need to sum the probabilities of the events where 6, 7, 8, 9, 10, and 11 of the models make the prediction.

Because each model in the ensemble is independent of the others, the predictions returned by the individual models can be viewed as a sequence of independent binary experiments. Consequently, we can calculate the probability of getting k outcomes, each with a probability of p , in a sequence of n experiments using the **binomial distribution** as

$$\binom{n}{k} \times p^k \times (1 - p)^{n-k}$$

For example, we can calculate the probability of the event where exactly 6 of the models makes an incorrect prediction using the binomial distribution as follows:

$$\begin{aligned} \binom{11}{6} \times 0.2^6 \times (1 - 0.2)^{11-6} &= \\ \frac{11!}{6! \times (11 - 6)!} \times 0.000064 \times 0.32768 &= \\ 462 \times 0.000064 \times 0.32768 &= \\ 0.0097 \end{aligned}$$

So, the probability of the model returning an incorrect prediction is

$$\sum_{i=6}^{11} \binom{11}{i} \times 0.2^i \times (1 - 0.2)^{11-i} = 0.0117$$

- (b) The ensemble contains 11 independent models, all of which have an error rate of 0.49.

$$\sum_{i=6}^{11} \binom{11}{i} \times 0.49^i \times (1 - 0.49)^{11-i} = 0.4729$$

- (c) The ensemble contains 21 independent models, all of which have an error rate of 0.49.

$$\sum_{i=11}^{21} \binom{21}{i} \times 0.49^i \times (1 - 0.49)^{21-i} = 0.4630$$

- * 10. The following table shows the target feature, **OUTCOME**, for a set of instances in a small dataset. An ensemble model is being trained using this dataset using **boosting**. The table also shows the instance distribution weights, \mathbf{w}_4 , for this dataset used at the fifth iteration of the boosting process. The last column of the table shows the predictions made by the model trained at the fifth iteration of boosting, \mathbb{M}_4 .

ID	OUTCOME	\mathbf{w}_4	\mathbb{M}_4
1	<i>Bad</i>	0.167	<i>Bad</i>
2	<i>Good</i>	0.047	<i>Good</i>
3	<i>Bad</i>	0.167	<i>Bad</i>
4	<i>Good</i>	0.071	<i>Bad</i>
5	<i>Good</i>	0.047	<i>Good</i>
6	<i>Bad</i>	0.047	<i>Bad</i>
7	<i>Bad</i>	0.047	<i>Bad</i>
8	<i>Good</i>	0.047	<i>Good</i>
9	<i>Bad</i>	0.167	<i>Bad</i>
10	<i>Good</i>	0.071	<i>Bad</i>
11	<i>Bad</i>	0.047	<i>Bad</i>
12	<i>Good</i>	0.071	<i>Bad</i>

- (a) Calculate the error, ϵ , associated with the set of predictions made by the model \mathbb{M}_4 given in the table above.

The error is calculated by summing the instance distribution weights of the instances incorrectly classified by the current model. In this case these are \mathbf{d}_4 , \mathbf{d}_{10} , and \mathbf{d}_{12} . So $\epsilon = 0.071 + 0.071 + 0.071 = 0.213$.

- (b) Calculate the **confidence factor**, α , associated with \mathbb{M}_4 .

The confidence factor, α for a model is calculated following Equation (4.14)^[161] as:

$$\begin{aligned}
 \alpha_4 &= \frac{1}{2} \times \log_e \left(\frac{1 - 0.213}{0.213} \right) \\
 &= \frac{1}{2} \times \log_e (3.695) \\
 &= \frac{1}{2} \times 1.307 \\
 &= 0.653
 \end{aligned}$$

- (c) Calculate the updated instance distribution, $\mathbf{w}^{[5]}$, based on the predictions made by \mathbb{M}_4 .

For each instances \mathbf{d}_i we update its weight in the instance distribution, $\mathbf{w}[i]$, using Equations (4.12)^[161] and (4.13)^[161] depending on whether it has been incorrectly or correctly classified. For example, instance \mathbf{d}_4 is incorrectly classified by \mathbb{M}_4 and so its instance distribution weight is updated as follows:

$$\begin{aligned}
 \mathbf{w}[4] &\leftarrow \mathbf{w}[4] \times \left(\frac{1}{2 \times \epsilon} \right) \\
 &\leftarrow 0.071 \times \left(\frac{1}{2 \times 0.213} \right) \\
 &\leftarrow 0.071 \times 2.347 \\
 &\leftarrow 0.167
 \end{aligned}$$

Similarly, instance \mathbf{d}_2 has been correctly classified by \mathbb{M}_4 and so its instance distribution weight is updated as follows:

$$\begin{aligned}
 \mathbf{w}[2] &\leftarrow \mathbf{w}[2] \times \left(\frac{1}{2 \times (1 - \epsilon)} \right) \\
 &\leftarrow 0.047 \times \left(\frac{1}{2 \times (1 - 0.213)} \right) \\
 &\leftarrow 0.047 \times 0.635 \\
 &\leftarrow 0.030
 \end{aligned}$$

These results match the expectation that the weight for an incorrectly classified instance should be increased slightly, while the weight for a correctly

classified instanced should be decreased slightly. The complete new distribution is shown below.

ID	OUTCOME	w_4	M_4	w_5
1	<i>Bad</i>	0.167	<i>Bad</i>	0.106
2	<i>Good</i>	0.047	<i>Good</i>	0.030
3	<i>Bad</i>	0.167	<i>Bad</i>	0.106
4	<i>Good</i>	0.071	<i>Bad</i>	0.167
5	<i>Good</i>	0.047	<i>Good</i>	0.030
6	<i>Bad</i>	0.047	<i>Bad</i>	0.030
7	<i>Bad</i>	0.047	<i>Bad</i>	0.030
8	<i>Good</i>	0.047	<i>Good</i>	0.030
9	<i>Bad</i>	0.167	<i>Bad</i>	0.106
10	<i>Good</i>	0.071	<i>Bad</i>	0.167
11	<i>Bad</i>	0.047	<i>Bad</i>	0.030
12	<i>Good</i>	0.071	<i>Bad</i>	0.167

- * 11. The following table shows a set of predictions made by six models in an ensemble and the ground truth of the target feature in a small test dataset, PROGNOSIS.

ID	PROGNOSIS	M_0	M_1	M_2	M_3	M_4	M_5
1	<i>Bad</i>	<i>Bad</i>	<i>Bad</i>	<i>Good</i>	<i>Bad</i>	<i>Bad</i>	<i>Good</i>
2	<i>Good</i>	<i>Good</i>	<i>Good</i>	<i>Good</i>	<i>Bad</i>	<i>Good</i>	<i>Bad</i>
3	<i>Good</i>	<i>Bad</i>	<i>Good</i>	<i>Bad</i>	<i>Good</i>	<i>Good</i>	<i>Good</i>
4	<i>Bad</i>	<i>Bad</i>	<i>Bad</i>	<i>Bad</i>	<i>Bad</i>	<i>Bad</i>	<i>Good</i>
5	<i>Bad</i>	<i>Good</i>	<i>Bad</i>	<i>Good</i>	<i>Bad</i>	<i>Good</i>	<i>Good</i>

- (a) Assuming that these models are part of an ensemble training using **bagging**, calculate the overall output of the ensemble for each instance in the test dataset.

Ensembles based on bagging use simple majority voting as their aggregation mechanism. So for each instance we simply count the number of positive and negative votes to determine the output of the overall ensemble. The following table shows the vote counts for each instance in the test dataset and the target feature level that receives the most votes.

ID	PROGNOSIS	<i>Bad</i> Votes	<i>Good</i> Votes	M
1	<i>Bad</i>	4	2	<i>Bad</i>
2	<i>Good</i>	2	4	<i>Good</i>
3	<i>Good</i>	2	4	<i>Good</i>
4	<i>Bad</i>	5	1	<i>Bad</i>
5	<i>Bad</i>	2	4	<i>Good</i>

- (b) Measure the performance of this bagged ensemble using **misclassification rate** (**misclassification rate** is discussed in detail in Section 9.3^[535]; it is simply the percentage of instances in the test dataset that a model has incorrectly classified).

In the case the ensemble model made the correct prediction for 4 out of 5 instances in the test dataset and only made an incorrect prediction for one. Therefore the misclassification rate is $\frac{1}{5} = 20\%$.

- (c) Assuming that these models are part of an ensemble trained using **boosting** and that the confidence factors, α , for the models are as follows:

M_0	M_1	M_2	M_3	M_4	M_5
0.114	0.982	0.653	0.912	0.883	0.233

calculate the overall output of the ensemble for each instance in the test dataset.

In the boosting case the votes are weighted by the confidence factor associated with each model. So we calculate a weighted vote for each target feature level, in this case *Bad* and *Good*, by adding together the confidence factors for the models that predict each target feature level.

For example, for the first instance in the test dataset, \mathbf{d}_1 , the models that predict the *Bad* target level are \mathbb{M}_0 , \mathbb{M}_1 , \mathbb{M}_3 , and \mathbb{M}_4 . So the weighted vote for the *Bad* target level is $0.114 + 0.982 + 0.912 + 0.883 = 2.891$. The models that predict the *Good* target level are \mathbb{M}_2 and \mathbb{M}_5 . So the weighted vote for the *Good* target level is $0.653 + 0.233 = 0.886$. The votes for *Bad* outweigh the votes for *Good* and so that is the overall model prediction in this case.

The votes for other test instances are calculated in the same way. The table below shows these weighted votes.

ID	PROGNOSIS	<i>Bad</i> Votes	<i>Good</i> Votes	\mathbb{M}
1	<i>Bad</i>	2.891	0.886	<i>Bad</i>
2	<i>Good</i>	1.145	2.632	<i>Good</i>
3	<i>Good</i>	0.767	3.01	<i>Good</i>
4	<i>Bad</i>	3.544	0.233	<i>Bad</i>
5	<i>Bad</i>	1.894	1.883	<i>Bad</i>

The last instance, \mathbf{d}_5 is particularly interesting in this case. Although four of the six models in the ensemble have made a prediction of *Good* for this instance, the two models with the highest confidence factors have made a prediction of *Bad* and the strength of their votes outweighs the votes of the mother models.

- (d) Measure the performance of this boosted ensemble using **misclassification rate**.

In the case the ensemble model made the correct prediction for all instances in the test dataset and so the misclassification rate is 0%.

5 Similarity-Based Learning (Exercise Solutions)

- The table below lists a dataset that was used to create a nearest neighbor model that predicts whether it will be a good day to go surfing.

ID	WAVE SIZE (FT)	WAVE PERIOD (SECS)	WIND SPEED (MPH)	GOOD SURF
1	6	15	5	yes
2	1	6	9	no
3	7	10	4	yes
4	7	12	3	yes
5	2	2	10	no
6	10	2	20	no

Assuming that the model uses Euclidean distance to find the nearest neighbor, what prediction will the model return for each of the following query instances?

ID	WAVE SIZE (FT)	WAVE PERIOD (SECS)	WIND SPEED (MPH)	GOOD SURF
Q1	8	15	2	?
Q2	8	2	18	?
Q3	6	11	4	?

The table below lists the training instances along with the distances between each training instance and each query. The distance between each query instance and its nearest training instance is highlighted in bold.

ID	WAVE SIZE (FT)	WAVE PERIOD (SECS)	WIND SPEED (MPH)	GOOD SURF	Euc. Dist. to Q1	Euc. Dist. to Q2	Euc. Dist. to Q3
1	6	15	5	yes	3.61	18.49	4.12
2	1	6	9	no	13.38	12.08	8.66
3	7	10	4	yes	5.48	16.16	1.41
4	7	12	3	yes	3.32	18.06	1.73
5	2	2	10	no	16.40	10.00	11.53
6	10	2	20	no	22.29	2.83	18.79

From this table we can see that

- The nearest neighbor to Q1 is training instance \mathbf{d}_4 which is 3.32 units away from Q1. This training instance has a target level of GOOD SURF=yes. So the model will predict GOOD SURF=yes for Q1.
- The nearest neighbor to Q2 is training instance \mathbf{d}_6 which is 2.83 units away from Q2. This training instance has a target level of GOOD SURF=no. So the model will predict GOOD SURF=no for Q2.
- The nearest neighbor to Q3 is training instance \mathbf{d}_3 which is 1.41 units away from Q3. This training instance has a target level of GOOD SURF=yes. So the model will predict GOOD SURF=yes for Q3.

2. Email spam filtering models often use a **bag-of-words** representation for emails. In a bag-of-words representation, the descriptive features that describe a document (in our case, an email) each represent how many times a particular word occurs in the document. One descriptive feature is included for each word in a predefined dictionary. The dictionary is typically defined as the complete set of words that occur in the training dataset. The table below lists the bag-of-words representation for the following five emails and a target feature, SPAM, whether they are spam emails or genuine emails:

- “money, money, money”
- “free money for free gambling fun”
- “gambling for fun”
- “machine learning for fun, fun, fun”
- “free machine learning”

ID	Bag-of-Words							SPAM
	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	
1	3	0	0	0	0	0	0	true
2	1	2	1	1	1	0	0	true
3	0	0	1	1	1	0	0	true
4	0	0	1	0	3	1	1	false
5	0	1	0	0	0	1	1	false

- (a) What target level would a nearest neighbor model using **Euclidean distance** return for the following email: “machine learning for free”?

The bag-of-words representation for this query is as follows:

ID	Bag-of-Words							SPAM
	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	
Query	0	1	1	0	0	1	1	?

The table below shows the calculation of the Euclidean distance between the query instance and each of the instances in the training dataset:

ID	$(\mathbf{q}[i] - \mathbf{d}_j[i])^2$							Euclidean Distance
	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	
1	9	1	1	0	0	1	1	3.6056
2	1	1	0	1	1	1	1	2.4495
3	0	1	0	1	1	1	1	2.2361
4	0	1	0	0	9	0	0	3.1623
5	0	0	1	0	0	0	0	1

Based on these distance calculations, the nearest neighbor to the query is instance \mathbf{d}_5 , for which $\text{SPAM} = \text{false}$. Consequently, the model will return a prediction of $\text{SPAM} = \text{false}$ for this query.

- (b) What target level would a k -NN model with $k = 3$ and using **Euclidean distance** return for the same query?

Based on the distance calculations in part (a) of this question, the three nearest neighbors to the query are instances \mathbf{d}_5 , \mathbf{d}_3 , and \mathbf{d}_2 . The majority of these three neighbors have a target value of $\text{SPAM} = \text{true}$. Consequently, the 3-NN model will return a prediction of $\text{SPAM} = \text{true}$.

- (c) What target level would a **weighted k -NN** model with $k = 5$ and using a weighting scheme of the reciprocal of the squared Euclidean distance between the neighbor and the query, return for the query?

The weights for each of the instances in the dataset are

ID	Weights	
1	$\frac{1}{3.6056^2}$	$= 0.0769$
2	$\frac{1}{2.4495^2}$	$= 0.1667$
3	$\frac{1}{2.2361^2}$	$= 0.2$
4	$\frac{1}{3.1623^2}$	$= 0.1$
5	$\frac{1}{1^2}$	$= 1$

The total weight for the SPAM = *true* target level is $0.0769 + 0.1667 + 0.2 = 0.4436$. The total weight for the SPAM = *false* target level is $0.1 + 1 = 1.1$. Consequently, the SPAM = *false* has the maximum weight, and this is the prediction returned by the model.

- (d) What target level would a k -NN model with $k = 3$ and using **Manhattan distance** return for the same query?

The table below shows the calculation of the Manhattan distance between the query bag-of-words vector and each instance in the dataset:

	$abs(\mathbf{q}[i] - \mathbf{d}_j[i])$							Manhattan Distance
ID	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	
1	3	1	1	0	0	1	1	7
2	1	1	0	1	1	1	1	6
3	0	1	0	1	1	1	1	5
4	0	1	0	0	3	0	0	4
5	0	0	1	0	0	0	0	1

Based on these Manhattan distance calculations, the three nearest neighbors to the query are instances \mathbf{d}_5 , \mathbf{d}_4 , and \mathbf{d}_3 . The majority of these three neighbors have a target value of SPAM = *false*. Consequently, the 3-NN model using Manhattan distance will return a prediction of SPAM = *false*.

- (e) There are a lot of zero entries in the spam bag-of-words dataset. This is indicative of **sparse data** and is typical for text analytics. **Cosine similarity** is often a good choice when dealing with sparse non-binary data. What target level would a 3-NN model using cosine similarity return for the query?

In order to calculate the cosine similarity between the query and each instance in the dataset, we first need to calculate the vector length of each instance and the query. The table below illustrates the calculation of these vector lengths.

ID	$\mathbf{d}[i]^2$								Vector Length	
									Sum	
1	9	0	0	0	0	0	0	0	9	3
2	1	4	1	1	1	0	0	0	8	2.8284
3	0	0	1	1	1	0	0	0	3	1.7321
4	0	0	1	0	9	1	1	1	12	3.4641
5	0	1	0	0	0	1	1	1	3	1.7321
Query	0	1	1	0	0	1	1	1	4	2

The second component we need to calculate is the dot product between the query and each instance. The table below illustrates the calculation of these dot products.

Pair	$(\mathbf{q}[i] \times \mathbf{d}_j[i])$							Dot Product
$(\mathbf{q}, \mathbf{d}_1)$	0	0	0	0	0	0	0	0
$(\mathbf{q}, \mathbf{d}_2)$	0	2	1	0	0	0	0	3
$(\mathbf{q}, \mathbf{d}_3)$	0	0	1	0	0	0	0	1
$(\mathbf{q}, \mathbf{d}_4)$	0	0	1	0	0	1	1	3
$(\mathbf{q}, \mathbf{d}_5)$	0	1	0	0	0	1	1	3

We can now calculate the cosine similarity for each query-instance pair by dividing the relevant dot product by the product of the respective vector lengths. These calculations are shown below.

Pair	Cosine Similarity	
$(\mathbf{q}, \mathbf{d}_1)$	$\frac{0}{3 \times 2}$	$= 0$
$(\mathbf{q}, \mathbf{d}_2)$	$\frac{3}{2.8285 \times 2}$	$= 0.5303$
$(\mathbf{q}, \mathbf{d}_3)$	$\frac{1}{1.7321 \times 2}$	$= 0.2887$
$(\mathbf{q}, \mathbf{d}_4)$	$\frac{3}{3.4641 \times 2}$	$= 0.4330$
$(\mathbf{q}, \mathbf{d}_5)$	$\frac{3}{1.7321 \times 2}$	$= 0.8660$

When we use a similarity index, such as cosine similarity, the higher the number, the more similar the instances. Given this, the three most similar instances in the dataset to the query are instances \mathbf{d}_5 , \mathbf{d}_2 , and \mathbf{d}_4 . The majority of these three neighbors have a target value of $\text{SPAM} = \text{false}$. Consequently, the 3-NN model will return a prediction of $\text{SPAM} = \text{false}$.

3. The predictive task in this question is to predict the level of corruption in a country based on a range of macroeconomic and social features. The table below lists some countries described by the following descriptive features:

- LIFE EXP., the mean life expectancy at birth
- TOP-10 INCOME, the percentage of the annual income of the country that goes to the top 10% of earners
- INFANT MORT., the number of infant deaths per 1,000 births
- MIL. SPEND, the percentage of GDP spent on the military
- SCHOOL YEARS, the mean number years spent in school by adult females

The target feature is the **Corruption Perception Index (CPI)**. The CPI measures the perceived levels of corruption in the public sector of countries and ranges from 0 (highly corrupt) to 100 (very clean).¹

COUNTRY ID	LIFE EXP.	TOP-10 INCOME	INFANT MORT.	MIL. SPEND	SCHOOL YEARS	CPI
Afghanistan	59.61	23.21	74.30	4.44	0.40	1.5171
Haiti	45.00	47.67	73.10	0.09	3.40	1.7999
Nigeria	51.30	38.23	82.60	1.07	4.10	2.4493
Egypt	70.48	26.58	19.60	1.86	5.30	2.8622
Argentina	75.77	32.30	13.30	0.76	10.10	2.9961
China	74.87	29.98	13.70	1.95	6.40	3.6356
Brazil	73.12	42.93	14.50	1.43	7.20	3.7741
Israel	81.30	28.80	3.60	6.77	12.50	5.8069
USA	78.51	29.85	6.30	4.72	13.70	7.1357
Ireland	80.15	27.23	3.50	0.60	11.50	7.5360
UK	80.09	28.49	4.40	2.59	13.00	7.7751
Germany	80.24	22.07	3.50	1.31	12.00	8.0461
Canada	80.99	24.79	4.90	1.42	14.20	8.6725
Australia	82.09	25.40	4.20	1.86	11.50	8.8442
Sweden	81.43	22.18	2.40	1.27	12.80	9.2985
New Zealand	80.67	27.81	4.90	1.13	12.30	9.4627

We will use Russia as our query country for this question. The table below lists the descriptive features for Russia.

COUNTRY ID	LIFE EXP.	TOP-10 INCOME	INFANT MORT.	MIL. SPEND	SCHOOL YEARS	CPI
Russia	67.62	31.68	10.00	3.87	12.90	?

- (a) What value would a 3-nearest neighbor prediction model using Euclidean distance return for the CPI of Russia?

The table below lists the countries in the dataset and their CPI values by increasing Euclidean distance from Russia (column 2).

1. The data listed in this table is real and is for 2010/11 (or the most recent year prior to 2010/11 when the data was available). The data for the descriptive features in this table was amalgamated from a number of surveys retrieved from **Gapminder** (www.gapminder.org). The Corruption Perception Index is generated annually by **Transparency International** (www.transparency.org).

ID	<i>Euclidean</i> (\mathbf{q}, \mathbf{d}_i)	CPI
Argentina	9.7805	2.9961
China	10.7898	3.6356
U.S.A	12.6033	7.1357
Egypt	13.7217	2.8622
Brazil	14.7394	3.7741
U.K.	15.0621	7.7751
Israel	16.0014	5.8069
Ireland	16.0490	7.5360
New Zealand	16.3806	9.4627
Canada	17.2765	8.6725
Australia	18.1472	8.8442
Germany	18.2352	8.0461
Sweden	19.8056	9.2985
Afghanistan	66.5419	1.5171
Haiti	69.6705	1.7999
Nigeria	75.2712	2.4493

The nearest three neighbors to Russia are Argentina, China, and U.S.A. The CPI value that will be returned by the model is the average CPI score for these three neighbors, which is

$$\frac{2.9961 + 3.6356 + 7.1357}{3} = 4.5891$$

- (b) What value would a **weighted k -NN** prediction model return for the CPI of Russia? Use $k = 16$ (i.e., the full dataset) and a weighting scheme of the reciprocal of the squared Euclidean distance between the neighbor and the query.

The table below shows the calculations required to answer this question.

ID	<i>Euclidean</i> (\mathbf{q}, \mathbf{d}_i)	CPI	Weight	Weight \times CPI
Argentina	9.7805	2.9961	0.0105	0.0313
China	10.7898	3.6356	0.0086	0.0312
U.S.A	12.6033	7.1357	0.0063	0.0449
Egypt	13.7217	2.8622	0.0053	0.0152
Brazil	14.7394	3.7741	0.0046	0.0174
U.K.	15.0621	7.7751	0.0044	0.0343
Israel	16.0014	5.8069	0.0039	0.0227
Ireland	16.0490	7.5360	0.0039	0.0293
New Zealand	16.3806	9.4627	0.0037	0.0353
Canada	17.2765	8.6725	0.0034	0.0291
Australia	18.1472	8.8442	0.0030	0.0269
Germany	18.2352	8.0461	0.0030	0.0242
Sweden	19.8056	9.2985	0.0025	0.0237
Afghanistan	66.5419	1.5171	0.0002	0.0003
Haiti	69.6705	1.7999	0.0002	0.0004
Nigeria	75.2712	2.4493	0.0002	0.0004
Sum Weight:			0.0637	
Sum Weight \times CPI:				0.3665

The value returned by the model is the sum of the instance weights multiplied by the instance target value divided by the sum of the instance weights:

$$\frac{0.3665}{0.0637} = 5.7507$$

- (c) The descriptive features in this dataset are of different types. For example, some are percentages, others are measured in years, and others are measured in counts per 1,000. We should always consider normalizing our data, but it is particularly important to do this when the descriptive features are measured in different units. What value would a 3-nearest neighbor prediction model using Euclidean distance return for the CPI of Russia when the descriptive features have been normalized using range normalization?

The table below lists the range-normalized descriptive features and the un-normalized CPI.

COUNTRY ID	LIFE EXP.	TOP-10 INCOME	INFANT MORT.	MIL. SPEND	SCHOOL YEARS	CPI
Afghanistan	0.3940	0.0445	0.8965	0.6507	0.0000	1.5171
Haiti	0.0000	1.0000	0.8815	0.0000	0.2174	1.7999
Nigeria	0.1698	0.6313	1.0000	0.1384	0.2681	2.4493
Egypt	0.6869	0.1762	0.2145	0.2652	0.3551	2.8622
Argentina	0.8296	0.3996	0.1359	0.0963	0.7029	2.9961
China	0.8053	0.3090	0.1409	0.2786	0.4348	3.6356
Brazil	0.7582	0.8148	0.1509	0.2004	0.4928	3.7741
Israel	0.9785	0.2629	0.0150	1.0000	0.8768	5.8069
U.S.A	0.9034	0.3039	0.0486	0.6922	0.9638	7.1357
Ireland	0.9477	0.2016	0.0137	0.0757	0.8043	7.5360
U.K.	0.9459	0.2508	0.0249	0.3749	0.9130	7.7751
Germany	0.9501	0.0000	0.0137	0.1818	0.8406	8.0461
Canada	0.9702	0.1063	0.0312	0.1996	1.0000	8.6725
Australia	1.0000	0.1301	0.0224	0.2651	0.8043	8.8442
Sweden	0.9821	0.0043	0.0000	0.1760	0.8986	9.2985
New Zealand	0.9617	0.2242	0.0312	0.1547	0.8623	9.4627

We also need to normalize the query in order to generate a prediction, so we show the normalized version of the descriptive features for Russia in the table below.

COUNTRY ID	LIFE EXP.	TOP-10 INCOME	INFANT MORT.	MIL. SPEND	SCHOOL YEARS	CPI
Russia	0.6099	0.3754	0.0948	0.5658	0.9058	?

The table below lists the countries in the dataset by increasing Euclidean distance—calculated using the normalized descriptive features—between Russia and the country (column 2). Notice that this ordering is different from the ordering of the countries when we used the unnormalized descriptive features.

ID	$Euclidean(\mathbf{q}, \mathbf{d}_i)$	CPI
Egypt	0.00004	2.8622
Brazil	0.00048	3.7741
China	0.00146	3.6356
Afghanistan	0.00217	1.5171
Argentina	0.00233	2.9961
United States	0.00742	7.1357
United Kingdom	0.01275	7.7751
Ireland	0.01302	7.5360
Germany	0.01339	8.0461
New Zealand	0.01531	9.4627
Canada	0.01685	8.6725
Israel	0.01847	5.8069
Sweden	0.01918	9.2985
Australia	0.02316	8.8442
Nigeria	0.03753	2.4493
Haiti	0.13837	1.7999

In this instance, the three nearest neighbors to Russia are Egypt, Brazil, and China. The CPI value that will be returned by the model is the average CPI score for these three neighbors:

$$\frac{2.8622 + 3.7741 + 3.6356}{3} = 3.4240$$

- (d) What value would a **weighted k -NN** prediction model—with $k = 16$ (i.e., the full dataset) and using a weighting scheme of the reciprocal of the squared Euclidean distance between the neighbor and the query—return for the CPI of Russia when it is applied to the range-normalized data?

The table below shows the calculations required to answer this question.

ID	<i>Euclidean</i> (\mathbf{q}, \mathbf{d}_i)	CPI	Weight	Weight \times CPI
Egypt	0.00004	2.8622	809,250,011.4	2,316,224,862.0
Brazil	0.00048	3.7741	4,284,287.3	16,169,371.4
China	0.00146	3.6356	471,369.7	1,713,699.1
Afghanistan	0.00217	1.5171	211,391.8	320,701.1
Argentina	0.00233	2.9961	184,029.5	551,366.0
United States	0.00742	7.1357	18,176.9	129,704.9
United Kingdom	0.01275	7.7751	6,154.1	47,849.0
Ireland	0.01302	7.5360	5,899.5	44,459.1
Germany	0.01339	8.0461	5,575.7	44,863.0
New Zealand	0.01531	9.4627	4,263.8	40,347.0
Canada	0.01685	8.6725	3,520.9	30,535.1
Israel	0.01847	5.8069	2,932.5	17,028.7
Sweden	0.01918	9.2985	2,717.1	25,265.1
Australia	0.02316	8.8442	1,864.8	16,492.9
Nigeria	0.03753	2.4493	710.1	1,739.3
Haiti	0.13837	1.7999	52.2	94.0
Sum Weight:			814,452,958	
Sum Weight \times CPI:				2,335,378,378

The value returned by the model is the sum of the instance weights multiplied by the instance target value divided by the sum of the instance weights:

$$\frac{2,335,378,378}{814,452,958} = 2.8674$$

- (e) The actual 2011 CPI for Russia was 2.4488. Which of the predictions made was the most accurate? Why do you think this was?

The most accurate prediction made was the one based on normalized data using the weighted k -NN model, 2.8674. There are two main reasons for this. First, this example illustrates the importance of normalizing data. Because the data ranges in this dataset are so different from each other, normalization is crucial. The second main reason is the small size of the dataset. Using three nearest neighbors probably tends to underfit slightly for such a small dataset. Using weighted distances allows for this.

- * 4. You have been given the job of building a recommender system for a large online shop that has a stock of over 100,000 items. In this domain the behavior of customers is captured in terms of what items they have bought or not bought. For example, the following table lists the behavior of two customers in this domain for a subset of the items that at least one of the customers has bought.

ID	ITEM 107	ITEM 498	ITEM 7256	ITEM 28063	ITEM 75328
1	true	true	true	false	false
2	true	false	false	true	true

- (a) The company has decided to use a similarity-based model to implement the recommender system. Which of the following three similarity indexes do you think the system should be based on?

$$\text{Russell-Rao}(X,Y) = \frac{CP(X,Y)}{P}$$

$$\text{Sokal-Michener}(X,Y) = \frac{CP(X,Y) + CA(X,Y)}{P}$$

$$\text{Jaccard}(X,Y) = \frac{CP(X,Y)}{CP(X,Y) + PA(X,Y) + AP(X,Y)}$$

In a domain where there are hundreds of thousands of items, co-absences aren't that meaningful. For example, you may be in a domain where there are so many items that most people haven't seen, listened to, bought, or visited that the majority of features will be co-absences. The technical term to describe a dataset where most of the features have zero values is **sparse data**. In these situations, you should use a metric that ignores co-absences. For a scenario such as this one, where the features are binary, the **Jaccard similarity index** is ideal as it ignores co-absences.

- (b) What items will the system recommend to the following customer? Assume that the recommender system uses the similarity index you chose in the first part of this question and is trained on the sample dataset listed above. Also assume that the system generates recommendations for query customers by finding the customer most similar to them in the dataset and then recommending the items that this similar customer has bought but that the query customer has not bought.

ID	ITEM 107	ITEM 498	ITEM 7256	ITEM 28063	ITEM 75328
Query	true	false	true	false	false

Using a similarity metric, the higher the value returned by the metric, the more similar the two items are.

Assuming you chose the **Jaccard similarity index**, then the query customer is more similar to customer \mathbf{d}_1 than to customer \mathbf{d}_2 :

$$\bullet \text{ Jaccard}(\mathbf{q}, \mathbf{d}_1) = \frac{2}{2+1} = 0.6667$$

$$\bullet \text{ Jaccard}(\mathbf{q}, \mathbf{d}_2) = \frac{1}{4} = 0.25$$

There is only 1 item that customer \mathbf{d}_1 has bought that the query customer has not bought, item 498. As a result, the system will recommend item 498 to the query customer.

It turns out that in this instance, no matter which of the three similarity metrics we use, customer \mathbf{d}_1 is more similar to the query customer than customer \mathbf{d}_2 . The supporting calculations for Russell-Rao and Sokal-Michener are

$$\bullet \text{ Russell-Rao}(\mathbf{q}, \mathbf{d}_1) = \frac{2}{5} = 0.4$$

$$\bullet \text{ Russell-Rao}(\mathbf{q}, \mathbf{d}_2) = \frac{1}{5} = 0.2$$

$$\bullet \text{ Sokal-Michener}(\mathbf{q}, \mathbf{d}_1) = \frac{4}{5} = 0.8$$

$$\bullet \text{ Sokal-Michener}(\mathbf{q}, \mathbf{d}_2) = \frac{2}{5} = 0.4$$

So, the system will recommend item 498 regardless of which similarity metric is used.

- * 5. You are working as an assistant biologist to Charles Darwin on the *Beagle* voyage. You are at the Galápagos Islands, and you have just discovered a new animal that has not yet been classified. Mr. Darwin has asked you to classify the animal using a nearest neighbor approach, and he has supplied you the following dataset of already classified animals.

ID	BIRTHS LIVE YOUNG	LAYS EGGS	FEEDS OFFSPRING OWN MILK	WARM-BLOODED	COLD-BLOODED	LAND AND WATER BASED	HAS HAIR	HAS FEATHERS	CLASS
1	true	false	true	true	false	false	true	false	mammal
2	false	true	false	false	true	true	false	false	amphibian
3	true	false	true	true	false	false	true	false	mammal
4	false	true	false	true	false	true	false	true	bird

The descriptive features of the mysterious newly discovered animal are as follows:

ID	BIRTHS LIVE YOUNG	LAYS EGGS	FEEDS OFFSPRING OWN MILK	WARM-BLOODED	COLD-BLOODED	LAND AND WATER BASED	HAS HAIR	HAS FEATHERS	CLASS
Query	false	true	false	false	false	true	false	false	?

- (a) A good measure of distance between two instances with categorical features is the **overlap metric** (also known as the **hamming distance**), which simply counts the number of descriptive features that have *different* values. Using this measure of distance, compute the distances between the mystery animal and each of the animals in the animal dataset.

We can calculate the overlap metric between the query instance and each instance in the dataset by counting the number of feature values that are different.

ID	CLASS	Overlap Metric
1	mammal	6
2	amphibian	1
3	mammal	6
4	bird	2

- (b) If you used a 1-NN model, what class would be assigned to the mystery animal?

The nearest neighbor to the mystery animal is \mathbf{d}_2 . So the mystery animal would be classified as an *amphibian*.

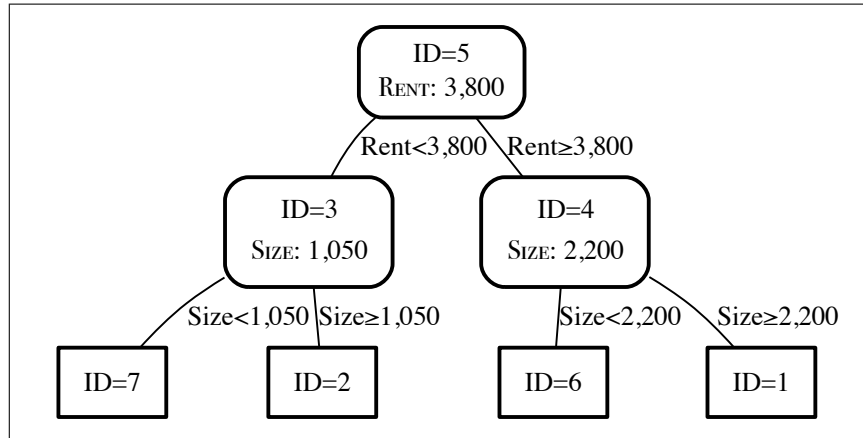
- (c) If you used a 4-NN model, what class would be assigned to the mystery animal? Would this be a good value for k for this dataset?

If you applied a 4-NN model to this dataset, the neighborhood defined around the query would include all the instances in the dataset irrespective of their distance from the query. As a result, any query would simply be assigned the majority class in the dataset, in this case *mammal*. So, for this dataset, a 4-NN model would massively underfit the dataset.

- * 6. You have been asked by a San Francisco property investment company to create a predictive model that will generate house price estimates for properties they are considering purchasing as rental properties. The table below lists a sample of properties that have recently been sold for rental in the city. The descriptive features in this dataset are SIZE (the property size in square feet) and RENT (the estimated monthly rental value of the property in dollars). The target feature, PRICE, lists the prices that these properties were sold for in dollars.

ID	SIZE	RENT	PRICE
1	2,700	9,235	2,000,000
2	1,315	1,800	820,000
3	1,050	1,250	800,000
4	2,200	7,000	1,750,000
5	1,800	3,800	1,450,500
6	1,900	4,000	1,500,500
7	960	800	720,000

- (a) Create a *k-d tree* for this dataset. Assume the following order over the features: RENT then SIZE.



- (b) Using the *k-d tree* that you created in the first part of this question, find the nearest neighbor to the following query: SIZE = 1,000, RENT = 2,200.

The initial step in retrieving the nearest neighbor is to descend the tree to a leaf node. For this query, this descent will terminate at the node that stores instance \mathbf{d}_7 . At this point, the *current best* variable is set to the instance stored at this node \mathbf{d}_7 , and the *current best-distance* variable is set to the Euclidean

distance between the query and \mathbf{d}_7 :

$$\text{current best} = \mathbf{d}_7$$

$$\text{current best-distance} = \text{Euclidean}(\mathbf{q}, \mathbf{d}_7) = 1,400.5713$$

The retrieval algorithm then ascends the tree. The first node the algorithm will encounter is the node that stores instance \mathbf{d}_3 . The Euclidean distance between the query and \mathbf{d}_3 is less than *current best-distance*. Consequently, *current best* and *current best-distance* are updated to reflect this:

$$\text{current best} = \mathbf{d}_3$$

$$\text{current best-distance} = \text{Euclidean}(\mathbf{q}, \mathbf{d}_3) = 951.3149$$

Because the difference between the splitting feature value at this node, SIZE = 1,050, and the query, SIZE = 1,000, is less than the *current best-distance*, the algorithm descends the other branch of the tree from this node. This descent will terminate at the node \mathbf{d}_2 . The Euclidean distance between the query and \mathbf{d}_2 is less than the *current best-distance*. Consequently, *current best* and *current best-distance* are updated to reflect this:

$$\text{current best} = \mathbf{d}_2$$

$$\text{current best-distance} = \text{Euclidean}(\mathbf{q}, \mathbf{d}_2) = 509.1414$$

The algorithm will then ascend the tree; because it has already visited all the nodes on the path back to the root, it does not need to check for nodes closer than the *current best* until it gets back to the root. In this instance, the Euclidean distance between the query and the instance stored at the root node, \mathbf{d}_3 , is greater than *current best-distance*, so neither *current best* nor *current best-distance* are updated when we reach the root node. Furthermore, because the difference between the splitting feature at the root, RENT = 3,800, and the query feature value, RENT = 2,200 is larger than the *current best-distance*, the algorithm can prune the other branch from the *k-d* tree and return \mathbf{d}_2 as the nearest neighbor, which would indicate that the property is worth approximately \$820,000.

* 7. A data analyst building a *k*-nearest neighbor model for a continuous prediction problem is considering appropriate values to use for *k*.

- (a) Initially the analyst uses a simple average of the target variables for the *k* nearest neighbors in order to make a new prediction. After experimenting with values for *k* in the range 0–10, it occurs to the analyst that they might get very good results

if they set k to the total number of instances in the training set. Do you think that the analyst is likely to get good results using this value for k ?

If the analyst set k to the number of training examples all predictions would essentially be the average target value across the whole dataset. In other words, the model would return the average value for the target feature in the training data no matter what query was input into the model. This means that the model would be massively underfitting the data.

- (b) If the analyst was using a distance weighted averaging function rather than a simple average for his or her predictions, would this have made the analyst's idea any more useful?

Yes, if distance weighted voting is used (particularly if a $\frac{1}{d^2}$ type distance weight is used) then examples that are far away from the query will have very little impact on the result and so the model will adjust the predictions it returns to the features in the query. It is worth highlighting that when distance weighted voting is used the value of k in k -NN classifiers is much less important.

- * 8. The following table describes a set of individuals in terms of their WEIGHT in kilograms, HEIGHT in meters, and whether or not they have DIABETES:

ID	WEIGHT	HEIGHT	DIABETES
1	68	1.7	true
2	55	1.6	false
3	65	1.6	true
4	100	1.9	true
5	65	1.5	false

- (a) A doctor has carried out a regular checkup on a patient and measured the patient's WEIGHT to be 65 kilograms and their HEIGHT to be 1.7 meters. The doctor inputs these details into a k -NN classifier to check whether the patient is at risk of DIABETES. Assuming that $k = 1$, and that the model uses Euclidean distance as its similarity metric, will the model return *true* or *false* for this patient?

The Euclidean distance, rounded to 2 places of decimal, between the query patient and each of the individuals in the dataset is as follows:

ID	DISTANCE	DIABETES
1	3.00	true
2	10.00	false
3	0.10	true
4	35.00	true
5	0.20	false

The patient is most similar to individual ID=3 for whom DIABETES is *true* and so the model will return *true* for this patient.

- (b) Clinicians often use BMI as a combined measure of an individual's WEIGHT and HEIGHT. BMI is defined as an individual's weight in kilograms divided by their height in meters-squared. Assuming that the profiles of the five individuals in the system were updated so that the features WEIGHT and HEIGHT were replaced by a single feature BMI and also that the doctor entered the patient's BMI into the system, what prediction would the system return for this patient?

Updating the profiles of the individuals in the dataset to have a feature BMI results in the following dataset (note we have rounded the BMI scores to two places of decimal):

ID	BMI	DIABETES
1	23.53	true
2	21.48	false
3	25.39	true
4	27.70	true
5	28.89	false

The patient's BMI is: 22.49 and the Euclidean distance between the patient and each of the individuals in the dataset in BMI space (rounded to 2 places of decimal) is:

ID	DISTANCE	DIABETES
1	1.04	true
2	1.01	false
3	2.90	true
4	5.21	true
5	6.40	false

Using this BMI representation the most similar individual in the dataset to the patient is individual ID=2 for whom DIABETES is *false* and so the model will return *false* for this patient.

- * 9. A lecturer is about to leave for the airport to go on vacation when they find a script for a student they forgot to mark. They don't have time to manually grade the script before the flight, so they decide to use a *k*-nearest neighbor model to grade it instead.

The model is designed to award a grade to a student on the basis of how similar they are to other students in the module in terms of their grades on other modules. The following table describes a set of students in terms of their grades out of 100 on two other modules (MODULE 1 and MODULE 2) and the GRADE they got in the lecturer's module: *first-class honors*, *second-class honors*, *pass*, or *fail*.

ID	MODULE 1	MODULE 2	GRADE
1	55	85	first
2	45	30	fail
3	40	20	fail
4	35	35	fail
5	55	75	pass
6	50	95	second

- (a) Looking up the results on the other modules of the student whose script hasn't been corrected, the lecturer finds that the student got the following marks: MODULE 1=60, and MODULE 2=85. Assuming that the k -nearest neighbor model uses $k=1$ and Euclidean distance as its similarity metric, what GRADE would the model assign the student?

The Euclidean distance, rounded to 2 places of decimal, between the student and the other students in the sample are as follows:

ID	DISTANCE	GRADE
1	5.00	first
2	57.00	fail
3	68.00	fail
4	55.90	fail
5	11.18	pass
6	14.14	second

The student is most similar to student ID=1 and so the model will return a GRADE of *first*.

- (b) Reviewing the spread of marks for the other two modules, the lecturer notices that there is a larger variance across students in the marks for *Module 2* than there is for *Module 1*. So, the lecturer decides to update the k -nearest neighbor model to use the **Mahalanobis** distance instead of Euclidean distance as its similarity measure. Assuming that the **inverse covariance matrix** for the *Module 1* and *Module 2* results is

$$\Sigma^{-1} = \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix}$$

what GRADE would the k -nearest neighbor model assign the student?

The calculations of the Mahalanobis distance between the student and each of the other students is as follows:

$$\text{Mahalanobis}(\mathbf{a}, \mathbf{b}) = \sqrt{[\mathbf{a}[1] - \mathbf{b}[1], \dots, \mathbf{a}[m] - \mathbf{b}[m]] \times \Sigma^{-1} \times \begin{bmatrix} \mathbf{a}[1] - \mathbf{b}[1] \\ \vdots \\ \mathbf{a}[m] - \mathbf{b}[m] \end{bmatrix}}$$

$$\begin{aligned} \text{Mahalanobis}(\mathbf{q}, \mathbf{d}_1) &= \sqrt{[5, 0] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 5 \\ 1 \end{bmatrix}} \\ &= 1.072380529 \end{aligned}$$

$$\begin{aligned} \text{Mahalanobis}(\mathbf{q}, \mathbf{d}_2) &= \sqrt{[15, 55] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 15 \\ 55 \end{bmatrix}} \\ &= 2.138924964 \end{aligned}$$

$$\begin{aligned} \text{Mahalanobis}(\mathbf{q}, \mathbf{d}_3) &= \sqrt{[20, 65] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 20 \\ 65 \end{bmatrix}} \\ &= 2.770379035 \end{aligned}$$

$$\text{Mahalanobis}(\mathbf{q}, \mathbf{d}_4) =$$

$$\sqrt{[25, 50] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 25 \\ 50 \end{bmatrix}} \\ = 3.708099244$$

$$\text{Mahalanobis}(\mathbf{q}, \mathbf{d}_5) =$$

$$\sqrt{[5, 10] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 5 \\ 10 \end{bmatrix}} \\ = 0.374165739$$

$$\text{Mahalanobis}(\mathbf{q}, \mathbf{d}_6) =$$

$$\sqrt{[10, -10] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 10 \\ -10 \end{bmatrix}} \\ = 2.588435821$$

When the model is updated to use Mahalanobis distance the student is most similar to \mathbf{d}_5 and so the model returns a GRADE of *pass*.

6 Probability-Based Learning (Exercise Solutions)

1. (a) Three people flip a fair coin. What is the probability that exactly two of them will get heads?

There are 8 possible outcomes:

Person1	Person2	Person3
Heads	Heads	Heads
Heads	Heads	Tails
Heads	Tails	Heads
Heads	Tails	Tails
Tails	Heads	Heads
Tails	Heads	Tails
Tails	Tails	Heads
Tails	Tails	Tails

In 3 of these outcomes there are 2 Heads. So the probability of exactly two people getting heads is

$$\frac{3}{8} = 0.375$$

- (b) Twenty people flip a fair coin. What is the probability that exactly eight of them will get heads?

We could use the same approach as we used in part (a) to answer this question: list all possible outcomes and count the number of outcomes that match our criteria. However, this approach doesn't scale up to problems where there are a lot of possible outcomes. For example, in part (a) with 3 people flipping the coin there were $2^3 = 8$ possible outcomes. However, now with 20 people

flipping the coin there are $2^{20} = 1,048,576$ possible outcomes—clearly, too many outcomes for us to list out. So what should we do?

Because each coin flip is independent of the others, the coin flips can be viewed as a sequence of independent binary experiments. Consequently, we can calculate the probability of getting k outcomes, each with a probability of p , in a sequence of n experiments using the **binomial distribution** as

$$\binom{n}{k} \times p^k \times (1 - p)^{n-k}$$

where n is the number of binary experiments, k is the number of particular results we are looking for (e.g., the number of heads we are looking for), and p is the probability of getting the result we are looking for (e.g., the probability of getting a head).

So, we can calculate the probability of the event where exactly 8 of the coin flips comes up heads using the binomial distribution as follows

$$\begin{aligned} \binom{20}{8} \times 0.5^8 \times (1 - 0.5)^{20-8} &= \frac{20!}{8! \times (20 - 8)!} \times 0.5^8 \times 0.5^{12} \\ &= 125970 \times 0.00390625 \times 0.000244141 \\ &= 0.120134354 \end{aligned}$$

Note: the ! symbol represents the factorial operation, for example

$$6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720$$

- (c) Twenty people flip a fair coin. What is the probability that at least four of them will get heads?

The probability that at least 4 people will get heads is equal to 1 minus probability that less than 4 people will get heads.

The probability that less than 4 people will get heads is simply the sum of the following probabilities:

- the probability that exactly 3 people will get heads
- the probability that exactly 2 people will get heads
- the probability that exactly 1 person will get heads

We can calculate each of these probabilities using the binomial distribution as follows:

The probability of exactly 3 people getting heads:

$$\begin{aligned}\binom{20}{3} \times 0.5^3 \times (1 - 0.5)^{20-3} &= \frac{20!}{3! \times (20-3)!} \times 0.5^3 \times 0.5^{17} \\ &= 1140 \times 0.125 \times (7.62939 \times 10^{-6}) \\ &= 0.001087189\end{aligned}$$

The probability of exactly 2 people getting heads:

$$\begin{aligned}\binom{20}{2} \times 0.5^2 \times (1 - 0.5)^{20-2} &= \frac{20!}{2! \times (20-2)!} \times 0.5^2 \times 0.5^{18} \\ &= 190 \times 0.25 \times (3.8147 \times 10^{-6}) \\ &= 0.000181198\end{aligned}$$

The probability of exactly 1 person getting heads:

$$\begin{aligned}\binom{20}{1} \times 0.5^1 \times (1 - 0.5)^{20-1} &= \frac{20!}{1! \times (20-1)!} \times 0.5^1 \times 0.5^{19} \\ &= 20 \times 0.5 \times (1.90735 \times 10^{-6}) \\ &= 0.0000190735\end{aligned}$$

Probability of 3 people or less getting heads is:

$$0.001087189 + 0.000181198 + 0.0000190735 = 0.0012874605$$

So the probability of at least 4 people getting heads is:

$$1 - 0.0012874605 = 0.9987125$$

2. The table below gives details of symptoms that patients presented and whether they were suffering from meningitis.

ID	HEADACHE	FEVER	VOMITING	MENINGITIS
1	true	true	false	false
2	false	true	false	false
3	true	false	true	false
4	true	false	true	false
5	false	true	false	true
6	true	false	true	false
7	true	false	true	false
8	true	false	true	true
9	false	true	false	false
10	true	false	true	true

Using this dataset, calculate the following probabilities:

- (a)
- $P(\text{VOMITING} = \text{true})$

This can be calculated easily by counting:

$$P(\text{VOMITING} = \text{true}) = \frac{6}{10} = 0.6$$

- (b)
- $P(\text{HEADACHE} = \text{false})$

This can be calculated easily by counting:

$$P(\text{HEADACHE} = \text{false}) = \frac{3}{10} = 0.3$$

- (c)
- $P(\text{HEADACHE} = \text{true}, \text{VOMITING} = \text{false})$

This can be calculated easily by counting:

$$P(\text{HEADACHE} = \text{true}, \text{VOMITING} = \text{false}) = \frac{1}{10} = 0.1$$

Or using the product rule:

$$P(\text{HEADACHE} = \text{true}, \text{VOMITING} = \text{false}) = P(\text{HEADACHE} = \text{true} \mid \text{VOMITING} = \text{false}) \times P(\text{VOMITING} = \text{false}) = \frac{1}{4} \times \frac{4}{10} = 0.1$$

- (d)
- $P(\text{VOMITING} = \text{false} \mid \text{HEADACHE} = \text{true})$

This can be calculated easily by counting:

$$P(\text{VOMITING} = \text{false} \mid \text{HEADACHE} = \text{true}) = \frac{1}{7} = 0.1429$$

- (e)
- $P(\text{MENINGITIS} \mid \text{FEVER} = \text{true}, \text{VOMITING} = \text{false})$

This can be calculated easily by counting. First,

$$P(\text{MENINGITIS} = \text{true} \mid \text{FEVER} = \text{true}, \text{VOMITING} = \text{false}) = \frac{1}{4} = 0.25.$$

Then,

$$P(\text{MENINGITIS} = \text{false} \mid \text{FEVER} = \text{true}, \text{VOMITING} = \text{false}) = \frac{3}{4} = 0.75$$

So,

$$\mathbf{P}(\text{MENINGITIS} \mid \text{FEVER} = \text{true}, \text{VOMITING} = \text{false}) = \langle 0.25, 0.75 \rangle$$

3. Predictive data analytics models are often used as tools for process quality control and fault detection. The task in this question is to create a naive Bayes model to monitor a wastewater treatment plant.¹ The table below lists a dataset containing details of activities at a wastewater treatment plant for 14 days. Each day is described in terms

1. The dataset in this question is inspired by the Waste Water Treatment Dataset that is available from the UCI Machine Learning repository (?) at archive.ics.uci.edu/ml/machine-learning-databases/water-treatment. The creators of this dataset reported their work in ?.

of six descriptive features that are generated from different sensors at the plant. SS-IN measures the solids coming into the plant per day; SED-IN measures the sediment coming into the plant per day; COND-IN measures the electrical conductivity of the water coming into the plant.² The features SS-OUT, SED-OUT, and COND-OUT are the corresponding measurements for the water flowing out of the plant. The target feature, STATUS, reports the current situation at the plant: *ok*, everything is working correctly; *settler*, there is a problem with the plant settler equipment; or *solids*, there is a problem with the amount of solids going through the plant.

ID	SS -IN	SED -IN	COND -IN	SS -OUT	SED -OUT	COND -OUT	STATUS
1	168	3	1,814	15	0.001	1,879	ok
2	156	3	1,358	14	0.01	1,425	ok
3	176	3.5	2,200	16	0.005	2,140	ok
4	256	3	2,070	27	0.2	2,700	ok
5	230	5	1,410	131	3.5	1,575	settler
6	116	3	1,238	104	0.06	1,221	settler
7	242	7	1,315	104	0.01	1,434	settler
8	242	4.5	1,183	78	0.02	1,374	settler
9	174	2.5	1,110	73	1.5	1,256	settler
10	1,004	35	1,218	81	1,172	33.3	solids
11	1,228	46	1,889	82.4	1,932	43.1	solids
12	964	17	2,120	20	1,030	1,966	solids
13	2,008	32	1,257	13	1,038	1,289	solids

- (a) Create a naive Bayes model that uses probability density functions to model the descriptive features in this dataset (assume that all the descriptive features are normally distributed).

The prior probabilities of each of the target feature levels are

$$P(\text{STATUS} = \text{ok}) = \frac{4}{13} = 0.3077$$

$$P(\text{STATUS} = \text{settler}) = \frac{5}{13} = 0.3846$$

$$P(\text{STATUS} = \text{solids}) = \frac{4}{13} = 0.3077$$

2. The conductivity of water is affected by inorganic dissolved solids and organic compounds, such as oil. Consequently, water conductivity is a useful measure of water purity.

To create the probability density functions required by the model, we simply need to fit a normal distribution to each feature for each level of the target. To do this, we calculate the mean and standard deviation for each feature for the set of instances where the target takes a given value. The table below lists the normal probability distributions fitted to each descriptive feature and target level.

$P(\text{SS-IN} \mid \text{ok})$	=	$N(x, \mu = 189, \sigma = 45.42)$
$P(\text{SED-IN} \mid \text{ok})$	=	$N(x, \mu = 3.125, \sigma = 0.25)$
$P(\text{COND-IN} \mid \text{ok})$	=	$N(x, \mu = 1,860.5, \sigma = 371.4)$
$P(\text{SS-OUT} \mid \text{ok})$	=	$N(x, \mu = 18, \sigma = 6.06)$
$P(\text{SED-OUT} \mid \text{ok})$	=	$N(x, \mu = 0.054, \sigma = 0.10)$
$P(\text{COND-OUT} \mid \text{ok})$	=	$N(x, \mu = 2,036, \sigma = 532.19)$
$P(\text{SS-IN} \mid \text{settler})$	=	$N(x, \mu = 200.8, \sigma = 55.13)$
$P(\text{SED-IN} \mid \text{settler})$	=	$N(x, \mu = 4.4, \sigma = 1.78)$
$P(\text{COND-IN} \mid \text{settler})$	=	$N(x, \mu = 1,251.2, \sigma = 116.24)$
$P(\text{SS-OUT} \mid \text{settler})$	=	$N(x, \mu = 98, \sigma = 23.38)$
$P(\text{SED-OUT} \mid \text{settler})$	=	$N(x, \mu = 1.018, \sigma = 1.53)$
$P(\text{COND-OUT} \mid \text{settler})$	=	$N(x, \mu = 1,372, \sigma = 142.58)$
$P(\text{SS-IN} \mid \text{solids})$	=	$N(x, \mu = 1,301, \sigma = 485.44)$
$P(\text{SED-IN} \mid \text{solids})$	=	$N(x, \mu = 32.5, \sigma = 11.96)$
$P(\text{COND-IN} \mid \text{solids})$	=	$N(x, \mu = 1,621, \sigma = 453.04)$
$P(\text{SS-OUT} \mid \text{solids})$	=	$N(x, \mu = 49.1, \sigma = 37.76)$
$P(\text{SED-OUT} \mid \text{solids})$	=	$N(x, \mu = 1,293, \sigma = 430.95)$
$P(\text{COND-OUT} \mid \text{solids})$	=	$N(x, \mu = 832.85, \sigma = 958.31)$

- (b) What prediction will the naive Bayes model return for the following query?

SS-IN = 222, SED-IN = 4.5, COND-IN = 1,518, SS-OUT = 74 SED-OUT = 0.25,
COND-OUT = 1,642

The calculation for STATUS = ok:

$P(ok)$	=	0.3077	
$P(SS-IN ok)$	=	$N(222, \mu = 189, \sigma = 45.42)$	= 0.0068
$P(SED-IN ok)$	=	$N(4.5, \mu = 3.125, \sigma = 0.25)$	= 4.3079×10^{-7}
$P(COND-IN ok)$	=	$N(1,518, \mu = 1,860.5, \sigma = 371.4)$	= 0.0007
$P(SS-OUT ok)$	=	$N(74, \mu = 18, \sigma = 6.06)$	= 1.7650×10^{-20}
$P(SED-OUT ok)$	=	$N(0.25, \mu = 0.054, \sigma = 0.10)$	= 0.5408
$P(COND-OUT ok)$	=	$N(1,642, \mu = 2,036, \sigma = 532.19)$	= 0.0006

$$\left(\prod_{k=1}^m P(\mathbf{q}[k] | ok) \right) \times P(ok) = 3.41577 \times 10^{-36}$$

The calculation for STATUS = *settler*:

$P(settler)$	=	0.3846	
$P(SS-IN settler)$	=	$N(222, \mu = 200.8, \sigma = 55.13)$	= 0.0067
$P(SED-IN settler)$	=	$N(4.5, \mu = 4.4, \sigma = 1.78)$	= 0.2235
$P(COND-IN settler)$	=	$N(1,518, \mu = 1,251.2, \sigma = 116.24)$	= 0.0002
$P(SS-OUT settler)$	=	$N(74, \mu = 98, \sigma = 23.38)$	= 0.0101
$P(SED-OUT settler)$	=	$N(0.25, \mu = 1.018, \sigma = 1.53)$	= 0.2303
$P(COND-OUT settler)$	=	$N(1,642, \mu = 1,372, \sigma = 142.58)$	= 0.0005

$$\left(\prod_{k=1}^m P(\mathbf{q}[k] | settler) \right) \times P(settler) = 1.53837 \times 10^{-13}$$

The calculation for STATUS = *solids*:

$P(solids)$	=	0.3077	
$P(SS-IN solids)$	=	$N(x, \mu = 1,301, \sigma = 485.44)$	= 6.9496×10^{-5}
$P(SED-IN solids)$	=	$N(x, \mu = 32.5, \sigma = 11.96)$	= 0.0022
$P(COND-IN solids)$	=	$N(x, \mu = 1,621, \sigma = 453.04)$	= 0.0009
$P(SS-OUT solids)$	=	$N(x, \mu = 49.1, \sigma = 37.76)$	= 0.0085
$P(SED-OUT solids)$	=	$N(x, \mu = 1,293, \sigma = 430.95)$	= 1.0291×10^{-5}
$P(COND-OUT solids)$	=	$N(x, \mu = 832.85, \sigma = 958.31)$	= 0.0003

$$\left(\prod_{k=1}^m P(\mathbf{q}[k] | solids) \right) \times P(solids) = 1.00668 \times 10^{-21}$$

Recall that because we are using the heights of the PDFs rather than calculating the actual probabilities for each feature taking a value, the score of each target level is a relative ranking and should not be interpreted as a probability.

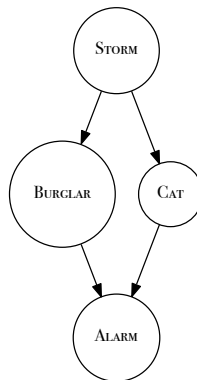
That said, the target level with the highest ranking is $\text{STATUS} = \text{settler}$. This indicates that there was a problem with the plant's settler equipment on the day of the query.

4. The following is a description of the causal relationship between storms, the behavior of burglars and cats, and house alarms:

Stormy nights are rare. Burglary is also rare, and if it is a stormy night, burglars are likely to stay at home (burglars don't like going out in storms). Cats don't like storms either, and if there is a storm, they like to go inside. The alarm on your house is designed to be triggered if a burglar breaks into your house, but sometimes it can be set off by your cat coming into the house, and sometimes it might not be triggered even if a burglar breaks in (it could be faulty or the burglar might be very good).

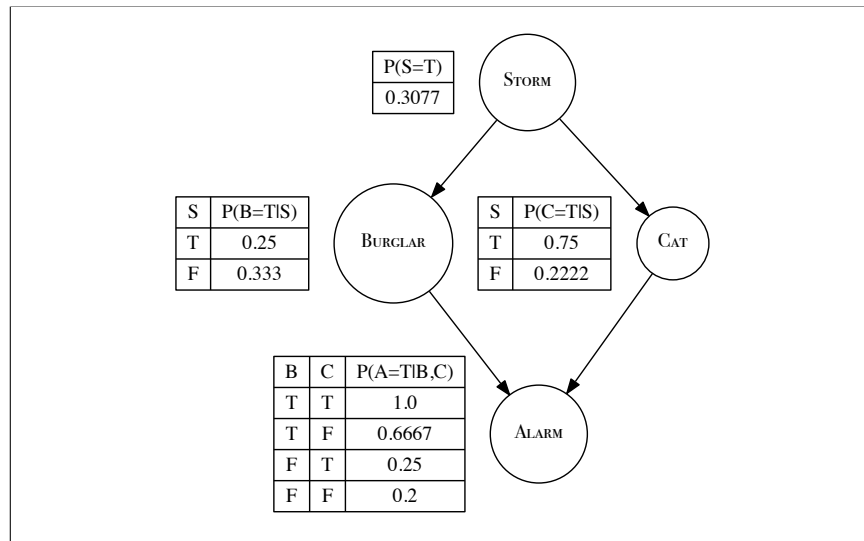
- (a) Define the topology of a Bayesian network that encodes these causal relationships.

The figure below illustrates a Bayesian network that encodes the described causal relationships. Storms directly affect the behavior of burglars and cats, and this is reflected by links from the storm node to the burglar and cat nodes. The behavior of burglars and cats both affect whether the alarm goes off, and hence there are links from each of these nodes to the alarm node.



- (b) The table below lists a set of instances from the house alarm domain. Using the data in this table, create the conditional probability tables (CPTs) for the network you created in Part (a) of this question.

ID	STORM	BURGLAR	CAT	ALARM
1	false	false	false	false
2	false	false	false	false
3	false	false	false	false
4	false	false	false	false
5	false	false	false	true
6	false	false	true	false
7	false	true	false	false
8	false	true	false	true
9	false	true	true	true
10	true	false	true	true
11	true	false	true	false
12	true	false	true	false
13	true	true	false	true



- (c) What value will the Bayesian network predict for ALARM, given that there is both a burglar and a cat in the house but there is no storm?

Because both the parent nodes for ALARM are known, the probability distribution over ALARM is independent of the feature STORM. Consequently, we can read the relevant probability distribution over ALARM directly from the conditional probability table for the ALARM node. Examining the conditional probability table, we can see that when BURGLAR = *true*, and CAT =

true, then $\text{ALARM} = \text{true}$ is the MAP prediction. In other words, the network would predict that the alarm would sound in this situation.

- (d) What value will the Bayesian network predict for ALARM , given that there is a storm but we don't know if a burglar has broken in or where the cat is?

In this case, the values of the parents of the target feature are unknown. Consequently, we need to sum out both the parents for each value of the target. The network would calculate the probability of the event $\text{ALARM} = \text{true}$ as follows:

$$\begin{aligned}
 P(a \mid s) &= \frac{P(a, s)}{P(s)} = \frac{\sum_{i,j} P(a, B_i, C_j, s)}{P(s)} \\
 \sum_{i,j} P(a, B_i, C_j, s) &= \sum_{i,j} P(a \mid B_i, C_j) \times P(B_i \mid s) \times P(C_j \mid s) \times P(s) \\
 &= (P(a \mid b, c) \times P(b \mid s) \times P(c \mid s) \times P(s)) \\
 &\quad + (P(a \mid b, \neg c) \times P(b \mid s) \times P(\neg c \mid s) \times P(s)) \\
 &\quad + (P(a \mid \neg b, c) \times P(\neg b \mid s) \times P(c \mid s) \times P(s)) \\
 &\quad + (P(a \mid \neg b, \neg c) \times P(\neg b \mid s) \times P(\neg c \mid s) \times P(s)) \\
 &= (1.0 \times 0.25 \times 0.75 \times 0.3077) + (0.6667 \times 0.25 \times 0.25 \times 0.3077) \\
 &\quad + (0.25 \times 0.75 \times 0.75 \times 0.3077) + (0.2 \times 0.75 \times 0.25 \times 0.3077) \\
 &= 0.125324 \\
 P(a \mid s) &= \frac{P(a, s)}{P(s)} = \frac{0.125324}{0.3077} = 0.4073
 \end{aligned}$$

This implies that $P(\text{ALARM} = \text{false}) = 0.5927$, so in this instance, $\text{ALARM} = \text{false}$ is the MAP level for the target, and this is the prediction the model will return.

- * 5. The table below lists a dataset containing details of policyholders at an insurance company. The descriptive features included in the table describe each policy holders' ID, occupation, gender, age, type of insurance policy, and preferred contact channel. The preferred contact channel is the target feature in this domain.

ID	OCCUPATION	GENDER	AGE	POLICY TYPE	PREF CHANNEL
1	lab tech	female	43	planC	email
2	farmhand	female	57	planA	phone
3	biophysicist	male	21	planA	email
4	sheriff	female	47	planB	phone
5	painter	male	55	planC	phone
6	manager	male	19	planA	email
7	geologist	male	49	planC	phone
8	messenger	male	51	planB	email
9	nurse	female	18	planC	phone

- (a) Using **equal-frequency binning**, transform the AGE feature into a categorical feature with three levels: *young*, *middle-aged*, *mature*.

There are 3 bins and 9 instances. So using equal-frequency binning we know that there will be 3 instances in each bin. In the table below we have ordered the instances in ascending order by AGE

ID	OCCUPATION	GENDER	AGE	POLICY TYPE	PREF CHANNEL
9	nurse	female	18	planC	phone
6	manager	male	19	planA	email
3	biophysicist	male	21	planA	email
1	lab tech	female	43	planC	email
4	sheriff	female	47	planB	phone
7	geologist	male	49	planC	phone
8	messenger	male	51	planB	email
5	painter	male	55	planC	phone
2	farmhand	female	57	planA	phone

If we put the first 3 instances into the *young* bin, the next 3 instances into the *middle-aged* bin, etc. we end up with the dataset in the table below.

ID	OCCUPATION	GENDER	AGE	POLICY TYPE	PREF CHANNEL
9	nurse	female	young	planC	phone
6	manager	male	young	planA	email
3	biophysicist	male	young	planA	email
1	lab tech	female	middle-aged	planC	email
4	sheriff	female	middle-aged	planB	phone
7	geologist	male	middle-aged	planC	phone
8	messenger	male	mature	planB	email
5	painter	male	mature	planC	phone
2	farmhand	female	mature	planA	phone

The thresholds for the different bins are calculated as the mid-point between the AGE values of the two instances on either side of the boundary. So, the threshold between the *young* and *middle-aged* bins would be the mid point between $\mathbf{d}_3 \text{withAGE} = 21$ and $\mathbf{d}_1 \text{withAGE} = 43$:

$$\frac{21 + 43}{2} = 32$$

Likewise, the threshold between the *middleaged* and *mature* bins would be the mid point between $\mathbf{d}_7 \text{withAGE} = 49$ and $\mathbf{d}_8 \text{withAGE} = 51$:

$$\frac{49 + 51}{2} = 50$$

- (b) Examine the descriptive features in the dataset and list the features that you would exclude before you would use the dataset to build a predictive model. For each feature you decide to exclude, explain why you have made this decision.

As is always the case the ID feature should not be used as a descriptive feature during training. However, in this example there is another feature that should be removed from the dataset prior to training. The OCCUPATION feature has different and unique levels for each instance in the dataset. In other words, the OCCUPATION feature is equivalent to an id for each instance. Consequently, it should also be removed from the dataset prior to training a model.

- (c) Calculate the probabilities required by a **naive Bayes model** to represent this domain.

To train a naive Bayes model using this data, we need to compute the prior probabilities of the target feature taking each level in its domain, and the conditional probability of each feature taking each level in its domain condi-

tioned for each level that the target feature can take. The table below lists the probabilities required by a naive Bayes model to represent this domain.

$P(phone)$	$= 0.56$	$P(email)$	$= 0.44$
$P(GENDER = female phone)$	$= 0.6$	$P(GENDER = female email)$	$= 0.25$
$P(GENDER = male phone)$	$= 0.4$	$P(GENDER = male email)$	$= 0.75$
$P(AGE = young phone)$	$= 0.2$	$P(AGE = young email)$	$= 0.5$
$P(AGE = middle-aged phone)$	$= 0.4$	$P(AGE = middle-aged email)$	$= 0.25$
$P(AGE = mature phone)$	$= 0.4$	$P(AGE = mature email)$	$= 0.25$
$P(POLICY = planA phone)$	$= 0.2$	$P(POLICY = planA email)$	$= 0.5$
$P(POLICY = planB phone)$	$= 0.2$	$P(POLICY = planB email)$	$= 0.25$
$P(POLICY = planC phone)$	$= 0.6$	$P(POLICY = planC email)$	$= 0.25$

(d) What target level will a **naive Bayes model** predict for the following query:

GENDER = *female*, AGE = 30, POLICY = *planA*

The first step in calculating this answer is to bin the AGE feature. We know from part (a) of this question that the threshold between the *young* and *middle-aged* bins is 32. The value for the AGE feature in the query is less than 32 so it is mapped to the *young* bin. This results in the query being defined as

GENDER = *female*, AGE = *young*, POLICY = *planA*

The calculation for $P(\text{CHANNEL} = \textit{phone} | \mathbf{q})$ is

$P(phone)$	$= 0.56$
$P(GENDER = female phone)$	$= 0.6$
$P(AGE = young phone)$	$= 0.2$
$P(POLICY = planA phone)$	$= 0.2$

$$\left(\prod_{k=1}^m P(\mathbf{q}[k] | \textit{phone}) \right) \times P(\textit{phone}) = 0.01344$$

The calculation for $P(\text{CHANNEL} = \textit{email} | \mathbf{q})$ is

$P(email)$	$= 0.44$
$P(GENDER = female email)$	$= 0.25$
$P(AGE = young email)$	$= 0.5$
$P(POLICY = planA email)$	$= 0.5$

$$\left(\prod_{k=1}^m P(\mathbf{q}[k] | \textit{email}) \right) \times P(\textit{email}) = 0.0275$$

The target level with the highest ranking is CHANNEL = *email*, and this is the prediction returned by the model.

- * 6. Imagine that you have been given a dataset of 1,000 documents that have been classified as being about *entertainment* or *education*. There are 700 *entertainment* documents in the dataset and 300 *education* documents in the dataset. The tables below give the number of documents from each topic that a selection of words occurred in.

Word-document counts for the <i>entertainment</i> dataset					
fun	is	machine	christmas	family	learning
415	695	35	0	400	70
Word-document counts for the <i>education</i> dataset					
fun	is	machine	christmas	family	learning
200	295	120	0	10	105

- (a) What target level will a **naive Bayes model** predict for the following query document: “*machine learning is fun*”?

A naive Bayes model will label the query with the target level that has the highest probability under the assumption of conditional independence between the evidence features. So to answer this question, we need to calculate the probability of each target level given the evidence and assuming conditional independence across the evidence.

To carry out these calculations, we need to convert the raw document counts into conditional probabilities by dividing each count by the total number of documents occurring in each topic:

w_k	Count	$P(w_k \mid \text{entertainment})$
fun	415	$\frac{415}{700} = .593$
is	695	$\frac{695}{700} = .99$
learning	35	$\frac{35}{700} = .05$
machine	70	$\frac{70}{700} = .10$

w_k	Count	$P(w_k \mid \text{education})$
fun	200	$\frac{200}{300} = .667$
is	295	$\frac{295}{300} = .983$
learning	120	$\frac{120}{300} = .40$
machine	105	$\frac{105}{300} = .35$

We can now compute the probabilities of each target level:

$$\begin{aligned}
 P(\text{entertainment} \mid \mathbf{q}) &= P(\text{entertainment}) \times P(\text{machine} \mid \text{entertainment}) \\
 &\quad \times P(\text{learning} \mid \text{entertainment}) \\
 &\quad \times P(\text{is} \mid \text{entertainment}) \\
 &\quad \times p(\text{fun} \mid \text{entertainment}) \\
 &= 0.7 \times 0.593 \times 0.99 \times 0.5 \times 0.1 \\
 &= 0.00205
 \end{aligned}$$

$$\begin{aligned}
 P(\text{education} \mid \mathbf{q}) &= P(\text{education}) \times P(\text{machine} \mid \text{education}) \\
 &\quad \times P(\text{learning} \mid \text{education}) \\
 &\quad \times P(\text{is} \mid \text{education}) \\
 &\quad \times p(\text{fun} \mid \text{education}) \\
 &= 0.3 \times 0.667 \times 0.983 \times 0.4 \times 0.35 \\
 &= 0.00275
 \end{aligned}$$

As $P(\text{education} \mid \mathbf{q}) > P(\text{entertainment} \mid \mathbf{q})$, the naive Bayes model will predict the target level of *education*.

- (b) What target level will a **naive Bayes model** predict for the following query document: “*christmas family fun*”?

Because the word *christmas* does not appear in any document of either topic, both conditional probabilities for this word will be equal to 0:

$$P(\text{christmas} \mid \text{entertainment}) = 0 \text{ and } P(\text{christmas} \mid \text{education}) = 0.$$

Consequently, the probability for both target levels will be 0, and the model will not be able to return a prediction.

- (c) What target level will a **naïve Bayes model** predict for the query document in Part (b) of this question, if **Laplace smoothing** with $k = 10$ and a vocabulary size of 6 is used?

The table below illustrates the smoothing of the posterior probabilities for $P(\text{word} \mid \text{entertainment})$.

Raw	$P(\text{christmas} \mid \text{entertainment})$	=	0
Probabilities	$P(\text{family} \mid \text{entertainment})$	=	0.5714
	$P(\text{fun} \mid \text{entertainment})$	=	0.5929
Smoothing	k	=	10
Parameters	$\text{count}(\text{entertainment})$	=	700
	$\text{count}(\text{christmas} \mid \text{entertainment})$	=	0
	$\text{count}(\text{family} \mid \text{entertainment})$	=	400
	$\text{count}(\text{fun} \mid \text{entertainment})$	=	415
	$ \text{Domain}(\text{vocabulary}) $	=	6
Smoothed	$P(\text{christmas} \mid \text{entertainment}) = \frac{0+10}{700+(10 \times 6)}$	=	0.0132
Probabilities	$P(\text{family} \mid \text{entertainment}) = \frac{400+10}{700+(10 \times 6)}$	=	0.5395
	$P(\text{fun} \mid \text{entertainment}) = \frac{415+10}{700+(10 \times 6)}$	=	0.5592

The smoothing of the posterior probabilities for $P(\text{word} \mid \text{education})$ is carried out in the same way:

Raw	$P(christmas \mid education)$	=	0
Probabilities	$P(family \mid education)$	=	0.5714
	$P(fun \mid education)$	=	0.5929
Smoothing	k	=	10
Parameters	$count(education)$	=	300
	$count(christmas \mid education)$	=	0
	$count(family \mid education)$	=	10
	$count(fun \mid education)$	=	200
	$ Domain(vocabulary) $	=	6
Smoothed	$P(christmas \mid entertainment) = \frac{0+10}{300+(10 \times 6)}$	=	0.0278
Probabilities	$P(family \mid entertainment) = \frac{10+10}{300+(10 \times 6)}$	=	0.0556
	$P(fun \mid entertainment) = \frac{200+10}{300+(10 \times 6)}$	=	0.5833

We can now compute the probabilities of each target level:

$$\begin{aligned}
 P(entertainment \mid \mathbf{q}) &= P(entertainment) \\
 &\quad \times P(christmas \mid entertainment) \\
 &\quad \times P(family \mid entertainment) \\
 &\quad \times P(fun \mid entertainment) \\
 &= 0.7 \times 0.0132 \times 0.5395 \times 0.5592 \\
 &= 0.0028
 \end{aligned}$$

$$\begin{aligned}
 P(education \mid \mathbf{q}) &= P(education) \times P(christmas \mid education) \\
 &\quad \times P(family \mid education) \times P(fun \mid education) \\
 &= 0.3 \times 0.0278 \times 0.0556 \times 0.5833 \\
 &= 0.0003
 \end{aligned}$$

As $P(entertainment \mid \mathbf{q}) > P(education \mid \mathbf{q})$, the model will predict a label of *entertainment* for this query.

- * 7. A **naive Bayes model** is being used to predict whether patients have a high risk of stroke in the next five years ($STROKE=true$) or a low risk of stroke in the next five years ($STROKE=false$). This model uses two continuous descriptive features AGE and WEIGHT (in kilograms). Both of these descriptive features are represented by

probability density functions, specifically normal distributions. The table below shows the representation of the domain used by this model.

$P(Stroke = true) = 0.25$	$P(Stroke = false) = 0.75$
$P(AGE = x \mid Stroke = true)$	$P(AGE = x \mid Stroke = false)$
$\approx N\left(\begin{matrix} x, \\ \mu = 65, \\ \sigma = 15 \end{matrix}\right)$	$\approx N\left(\begin{matrix} x, \\ \mu = 20, \\ \sigma = 15 \end{matrix}\right)$
$P(WEIGHT = x \mid Stroke = true)$	$P(WEIGHT = x \mid Stroke = false)$
$\approx N\left(\begin{matrix} x, \\ \mu = 88, \\ \sigma = 8 \end{matrix}\right)$	$\approx N\left(\begin{matrix} x, \\ \mu = 76, \\ \sigma = 6 \end{matrix}\right)$

(a) What target level will the **naive Bayes model** predict for the following query:

AGE = 45, WEIGHT = 80

$P(Stroke = true) = 0.25$	$P(Stroke = false) = 0.75$
$P(AGE = 45 \mid Stroke = true)$	$P(AGE = 45 \mid Stroke = false)$
$\approx N\left(\begin{matrix} x = 45, \\ \mu = 65, \\ \sigma = 15 \end{matrix}\right)$	$\approx N\left(\begin{matrix} x = 45, \\ \mu = 20, \\ \sigma = 15 \end{matrix}\right)$
$= 0.0109$	$= 0.0066$
$P(WEIGHT = 80 \mid Stroke = true)$	$P(WEIGHT = 80 \mid Stroke = false)$
$\approx N\left(\begin{matrix} x = 80, \\ \mu = 88, \\ \sigma = 8 \end{matrix}\right)$	$\approx N\left(\begin{matrix} x = 80, \\ \mu = 76, \\ \sigma = 6 \end{matrix}\right)$
$= 0.0302$	$= 0.0532$
$P(Stroke = true \mid AGE = 45, WEIGHT = 80) = 0.25 \times 0.0109 \times 0.0302$	
$= 0.000082295$	
$P(Stroke = false \mid AGE = 45, WEIDGHT = 80) = 0.75 \times 0.0066 \times 0.0532$	
$= 0.000263340$	
Based on these calculations the model will predict $Stroke = false$ for this patient.	

- * 8. The table below lists a dataset of books and whether or not they were purchased by an individual (i.e., the feature PURCHASED is the target feature in this domain).

ID	SECONDHAND	GENRE	COST	PURCHASED
1	false	romance	expensive	true
2	false	science	cheap	false
3	true	romance	cheap	true
4	false	science	cheap	true
5	false	science	expensive	false
6	true	romance	reasonable	false
7	true	literature	cheap	false
8	false	romance	reasonable	false
9	true	science	cheap	false
10	true	literature	reasonable	true

- (a) Calculate the probabilities (to four places of decimal) that a **naive Bayes** classifier would use to represent this domain.

A naive Bayes classifier would require the prior probability for each level of the target feature and the conditional probability for each level of each descriptive feature given each level of the target feature:

$$\begin{aligned}
 P(\text{Purchased} = \text{true}) &= 0.4 \\
 P(\text{2ndHand} = \text{true} | \text{Purchased} = \text{true}) &= 0.5 \\
 P(\text{2ndHand} = \text{false} | \text{Purchased} = \text{true}) &= 0.5 \\
 P(\text{Genre} = \text{literature} | \text{Purchased} = \text{true}) &= 0.25 \\
 P(\text{Genre} = \text{romance} | \text{Purchased} = \text{true}) &= 0.5 \\
 P(\text{Genre} = \text{science} | \text{Purchased} = \text{true}) &= 0.25 \\
 P(\text{Price} = \text{cheap} | \text{Purchased} = \text{true}) &= 0.5 \\
 P(\text{Price} = \text{reasonable} | \text{Purchased} = \text{true}) &= 0.25 \\
 P(\text{Price} = \text{expensive} | \text{Purchased} = \text{true}) &= 0.25 \\
 P(\text{Purchased} = \text{false}) &= 0.6 \\
 P(\text{2ndHand} = \text{true} | \text{Purchased} = \text{false}) &= 0.5 \\
 P(\text{2ndHand} = \text{false} | \text{Purchased} = \text{false}) &= 0.5 \\
 P(\text{Genre} = \text{literature} | \text{Purchased} = \text{false}) &= 0.1667 \\
 P(\text{Genre} = \text{romance} | \text{Purchased} = \text{false}) &= 0.3333 \\
 P(\text{Genre} = \text{science} | \text{Purchased} = \text{false}) &= 0.5 \\
 P(\text{Price} = \text{cheap} | \text{Purchased} = \text{false}) &= 0.5 \\
 P(\text{Price} = \text{reasonable} | \text{Purchased} = \text{false}) &= 0.3333 \\
 P(\text{Price} = \text{expensive} | \text{Purchased} = \text{false}) &= 0.1667
 \end{aligned}$$

- (b) Assuming conditional independence between features given the target feature value, calculate the **probability** (rounded to four places of decimal) of each outcome (PURCHASED=true, and PURCHASED=false) for the following book:

SECONDHAND=false, GENRE=literature, COST=expensive

The initial score for each outcome is calculated as follows:

$$(Purchased = true) = 0.5 \times 0.25 \times 0.25 \times 0.4 = 0.0125$$

$$(Purchased = false) = 0.5 \times 0.1667 \times 0.1667 \times 0.6 = 0.0083$$

However, these scores are not probabilities. To get real probabilities we must normalise these scores. The normalisation constant is calculated as follows:

$$\alpha = 0.0125 + 0.0083 = 0.0208$$

The actual probabilities of each outcome is then calculated as:

$$P(Purchased = true) = \frac{0.0125}{0.0208} = (0.600961...) = 0.6010$$

$$P(Purchased = false) = \frac{0.0083}{0.0208} = (0.399038...) = 0.3990$$

- (c) What prediction would a **naive Bayes** classifier return for the above book?

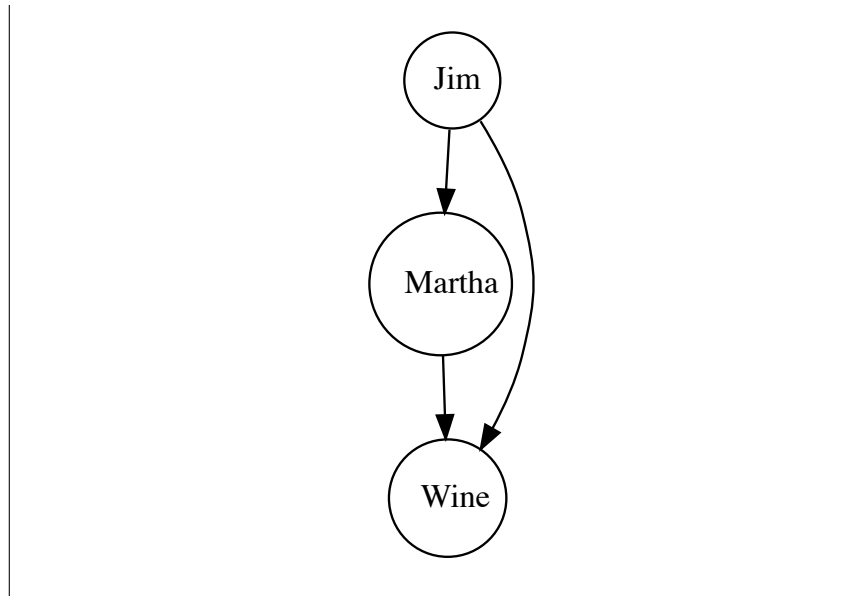
A naive Bayes classifier returns outcome with the maximum a posteriori probability as its prediction. In this instance the outcome `PURCHASED=true` is the MAP prediction and will be the outcome returned by a naive Bayes model.

- * 9. The following is a description of the causal relationship between storms, the behavior of burglars and cats, and house alarms:

Jim and Martha always go shopping separately. If Jim does the shopping he buys wine, but not always. If Martha does the shopping, she buys wine, but not always. If Jim tells Martha that he has done the shopping, then Martha doesn't go shopping, but sometimes Jim forgets to tell Martha, and so sometimes both Jim and Martha go shopping.

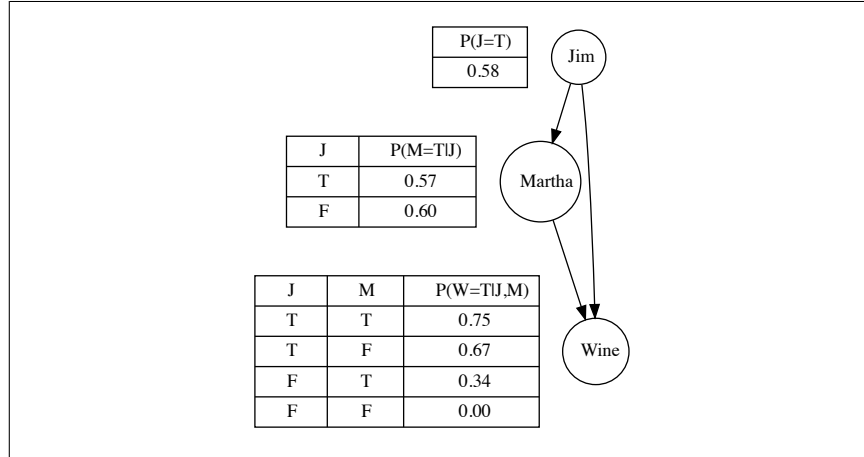
- (a) Define the topology of a Bayesian network that encodes these causal relationships between the following Boolean variables: `JIM` (Jim has done the shopping, *true* or *false*), `MARTHA` (Martha has done the shopping, *true* or *false*), `WINE` (wine has been purchased, *true* or *false*).

The figure below illustrates a Bayesian network that encodes the described causal relationships. `JIM` directly affects the behavior of `MARTHA` and `WINE`, and this is reflected by links from the `JIM` node to the `MARTHA` and `WINE` nodes. The `MARTHA` also affects whether `WINE` was purchased, and hence there is a link from `MARTHA` to `WINE`.



- (b) The table below lists a set of instances from the house alarm domain. Using the data in this table, create the conditional probability tables (CPTs) for the network you created in the first part of this question, and round the probabilities to two places of decimal.

ID	JIM	MARTHA	WINE
1	false	false	false
2	false	false	false
3	true	false	true
4	true	false	true
5	true	false	false
6	false	true	true
7	false	true	false
8	false	true	false
9	true	true	true
10	true	true	true
11	true	true	true
12	true	true	false



- (c) What value will the Bayesian network predict for WINE if:

$JIM = true$ and $MARTHA = false$

Because both the parent nodes for WINE are known, we can read the relevant probability distribution over WINE directly from the conditional probability table for the WINE node. Examining the conditional probability table, we can see that when $JIM = true$, and $MARTHA = false$, then $WINE = true$ is the MAP prediction (0.75 versus 0.25). In other words, the network would predict that wine would be purchased in this scenario.

- (d) What is the probability that JIM went shopping given that $WINE = true$?

$$\begin{aligned}
 P(JIM = true \mid WINE = true) &= \frac{P(JIM = true, WINE = true)}{P(WINE = true)} \\
 &= \frac{\sum_{M \in \{T, F\}} P(MARTHA, JIM = true, WINE = true)}{\sum_{M, J \in \{T, F\}} P(MARTHA, JIM, WINE = true)} \\
 &= \frac{0.75 + 0.67}{0.75 + 0.67 + 0.34 + 0.00} \\
 &= \frac{1.42}{1.76} \\
 &= 0.81
 \end{aligned}$$

7 Error-Based Learning (Exercise Solutions)

1. A multivariate linear regression model has been built to predict the **heating load** in a residential building on the basis of a set of descriptive features describing the characteristics of the building. Heating load is the amount of heat energy required to keep a building at a specified temperature, usually 65° Fahrenheit during the winter regardless of outside temperature. The descriptive features used are the overall surface area of the building, the height of the building, the area of the building's roof, and the percentage of wall area in the building that is glazed. This kind of model would be useful to architects or engineers when designing a new building.¹ The trained model is

$$\begin{aligned}\text{HEATING LOAD} = & -26.030 + 0.0497 \times \text{SURFACE AREA} \\ & + 4.942 \times \text{HEIGHT} - 0.090 \times \text{ROOF AREA} \\ & + 20.523 \times \text{GLAZING AREA}\end{aligned}$$

Use this model to make predictions for each of the query instances shown in the following table.

ID	SURFACE AREA	HEIGHT	ROOF AREA	GLAZING AREA
1	784.0	3.5	220.5	0.25
2	710.5	3.0	210.5	0.10
3	563.5	7.0	122.5	0.40
4	637.0	6.0	147.0	0.60

1. This question is inspired by ?, and although the data used is artificially generated, it is based on the Energy Efficiency Dataset available from the UCI Machine Learning Repository (?) at archive.ics.uci.edu/ml/datasets/Energy+efficiency/.

Calculating the predictions made by the model simply involves inserting the descriptive features from each query instance into the prediction model.

$$\mathbf{1:} -26.030 + 0.0497 \times 784.0 + 4.942 \times 3.5 - 0.090 \times 220.5 + 20.523 \times 0.25 = 15.5$$

$$\mathbf{2:} -26.030 + 0.0497 \times 710.5 + 4.942 \times 3.0 - 0.09 \times 210.5 + 20.523 \times 0.10 = 7.2$$

$$\mathbf{3:} -26.03 + 0.0497 \times 563.5 + 4.942 \times 7.0 - 0.09 \times 122.5 + 20.523 \times 0.40 = 33.8$$

$$\mathbf{4:} -26.03 + 0.0497 \times 637.0 + 4.942 \times 6.0 - 0.09 \times 147.0 + 20.523 \times 0.60 = 34.4$$

2. You have been hired by the European Space Agency to build a model that predicts the amount of oxygen that an astronaut consumes when performing five minutes of intense physical work. The descriptive features for the model will be the age of the astronaut and their average heart rate throughout the work. The regression model is

$$\text{OXYCON} = \mathbf{w}[0] + \mathbf{w}[1] \times \text{AGE} + \mathbf{w}[2] \times \text{HEARTRATE}$$

The table that follows shows a historical dataset that has been collected for this task.

ID	OXYCON	AGE	HEART	ID	OXYCON	AGE	HEART
			RATE				RATE
1	37.99	41	138	7	44.72	43	158
2	47.34	42	153	8	36.42	46	143
3	44.38	37	151	9	31.21	37	138
4	28.17	46	133	10	54.85	38	158
5	27.07	48	126	11	39.84	43	143
6	37.85	44	145	12	30.83	43	138

- (a) Assuming that the current weights in a multivariate linear regression model are $\mathbf{w}[0] = -59.50$, $\mathbf{w}[1] = -0.15$, and $\mathbf{w}[2] = 0.60$, make a prediction for each training instance using this model.

The following table shows the predictions made using the given model weights.

ID	OXYCON	AGE	HEART RATE	Prediction
1	37.99	41	138	17.15
2	47.34	42	153	26.00
3	44.38	37	151	25.55
4	28.17	46	133	13.40
5	27.07	48	126	8.90
6	37.85	44	145	20.90
7	44.72	43	158	28.85
8	36.42	46	143	19.40
9	31.21	37	138	17.75
10	54.85	38	158	29.60
11	39.84	43	143	19.85
12	30.83	43	138	16.85

- (b) Calculate the sum of squared errors for the set of predictions generated in Part (a).

The following table shows the predictions made by the model and sum of squared error calculation based on these predictions.

Initial Weights								
$\mathbf{w}[0]:$		-59.50	$\mathbf{w}[1]:$		-0.15	$\mathbf{w}[2]:$ <td>0.60</td>		0.60
Iteration 1								
ID	OXYCON	Prediction	Error	Squared Error	$errorDelta$ ($\mathcal{D}, \mathbf{w}[0]$)	$errorDelta$ ($\mathcal{D}, \mathbf{w}[1]$)	$errorDelta$ ($\mathcal{D}, \mathbf{w}[2]$)	
1	37.99	17.15	20.84	434.41	20.84	854.54	2,876.26	
2	47.34	26.00	21.34	455.41	21.34	896.29	3,265.05	
3	44.38	25.55	18.83	354.60	18.83	696.74	2,843.45	
4	28.17	13.40	14.77	218.27	14.77	679.60	1,964.93	
5	27.07	8.90	18.17	330.09	18.17	872.08	2,289.20	
6	37.85	20.90	16.95	287.35	16.95	745.86	2,457.94	
7	44.72	28.85	15.87	251.91	15.87	682.48	2,507.71	
8	36.42	19.40	17.02	289.72	17.02	782.98	2,434.04	
9	31.21	17.75	13.46	181.26	13.46	498.14	1,857.92	
10	54.85	29.60	25.25	637.57	25.25	959.50	3,989.52	
11	39.84	19.85	19.99	399.47	19.99	859.44	2,858.12	
12	30.83	16.85	13.98	195.52	13.98	601.25	1,929.61	
Sum				4,035.56	216.48	9,128.90	3,1273.77	
Sum of squared errors ($Sum/2$)				2,017.78				

- (c) Assuming a learning rate of 0.000002, calculate the weights at the next iteration of the gradient descent algorithm.

To calculate the updated weight values we apply the weight update rule for multivariate linear regression with gradient descent for each weight as fol-

lows (using *errorDelta* values given in the answer to the previous part):

$$\begin{aligned}
 \mathbf{w}[0] &\leftarrow \mathbf{w}[0] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[0]) \\
 &\leftarrow -59.50 + 0.000002 \times 216.48 \\
 &\leftarrow -59.4996 \\
 \mathbf{w}[1] &\leftarrow \mathbf{w}[1] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[1]) \\
 &\leftarrow -0.15 + 0.000002 \times 9128.9 \\
 &\leftarrow -0.1317 \\
 \mathbf{w}[2] &\leftarrow \mathbf{w}[2] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[2]) \\
 &\leftarrow 0.60 + 0.000002 \times 31273.77 \\
 &\leftarrow 0.6625
 \end{aligned}$$

- (d) Calculate the sum of squared errors for a set of predictions generated using the new set of weights calculated in Part (c).

The new sum of squared errors calculated using these new weights is given by the following table.

ID	OXYCON	Prediction	Error	Squared Error
1	37.99	26.53	11.46	131.38
2	47.34	36.34	11.00	121.07
3	44.38	35.67	8.71	75.87
4	28.17	22.56	5.61	31.53
5	27.07	17.66	9.41	88.56
6	37.85	30.77	7.08	50.10
7	44.72	39.52	5.20	27.08
8	36.42	29.18	7.24	52.37
9	31.21	27.06	4.16	17.27
10	54.85	40.18	14.67	215.31
11	39.84	29.58	10.26	105.21
12	30.83	26.27	4.57	20.84
			Sum	936.57
			Sum of squared errors (<i>Sum</i> /2)	468.29

3. A multivariate logistic regression model has been built to predict the propensity of shoppers to perform a repeat purchase of a free gift that they are given. The descriptive features used by the model are the age of the customer, the socioeconomic band to which the customer belongs (*a*, *b*, or *c*), the average amount of money the customer spends on each visit to the shop, and the average number of visits the customer makes

to the shop per week. This model is being used by the marketing department to determine who should be given the free gift. The weights in the trained model are shown in the following table.

Feature	Weight
Intercept ($\mathbf{w}[0]$)	-3.82398
AGE	-0.02990
SOCIOECONOMIC BAND B	-0.09089
SOCIOECONOMIC BAND C	-0.19558
SHOP VALUE	0.02999
SHOP FREQUENCY	0.74572

Use this model to make predictions for each of the following query instances.

ID	AGE	SOCIOECONOMIC BAND	SHOP FREQUENCY	SHOP VALUE
1	56	b	1.60	109.32
2	21	c	4.92	11.28
3	48	b	1.21	161.19
4	37	c	0.72	170.65
5	32	a	1.08	165.39

Calculating the predictions made by the model simply involves inserting the descriptive features from each query instance into the prediction model. The only extra thing that must be considered in this case is the categorical descriptive feature SOCIOECONOMIC BAND. We can note from the regression equation that this one feature has been expanded into two: SOCIOECONOMIC BAND B and SOCIOECONOMIC BAND C. These are binary features, indicating that the original feature was set to the level *b* or *c*. It is assumed that when both of these features are set to 0, then the original feature was set to *a* (the choice of which level to leave out is arbitrary). The other pieces of information required are that the *yes* level is the positive level, and the classification threshold is 0.5.

With this information, the predictions can be made as follows:

- 1: $\text{Logistic}(-3.82398 + -0.0299 \times 56 + -0.09089 \times 1 + -0.19558 \times 0 + 0.74572 \times 1.6 + 0.02999 \times 109.32)$
 $= \text{Logistic}(-1.12) = \frac{1}{1+e^{1.12}}$
 $= 0.25 \Rightarrow \text{no}$
- 2: $\text{Logistic}(-3.82398 + -0.0299 \times 21 + -0.09089 \times 0 + -0.19558 \times 1 + 0.74572 \times 4.92 + 0.02999 \times 11.28)$

$$= \text{Logistic}(-0.64) = \frac{1}{1+e^{0.64}}$$

$$= 0.35 \Rightarrow \text{no}$$

$$\mathbf{3:} \text{Logistic}(-3.82398 + -0.0299 \times 48 + -0.09089 \times 1 + -0.19558 \times 0 + 0.74572 \times 1.21 + 0.02999 \times 161.19)$$

$$= \text{Logistic}(0.39) = \frac{1}{1+e^{-0.39}}$$

$$= 0.60 \Rightarrow \text{yes}$$

$$\mathbf{4:} \text{Logistic}(-3.82398 + -0.0299 \times 37 + -0.09089 \times 0 + -0.19558 \times 1 + 0.74572 \times 0.72 + 0.02999 \times 170.65)$$

$$= \text{Logistic}(0.53) = \frac{1}{1+e^{-0.53}}$$

$$= 0.63 \Rightarrow \text{yes}$$

$$\mathbf{5:} \text{Logistic}(-3.82398 + -0.0299 \times 32 + -0.09089 \times 0 + -0.19558 \times 0 + 0.74572 \times 1.08 + 0.02999 \times 165.39)$$

$$= \text{Logistic}(0.98) = \frac{1}{1+e^{-0.98}}$$

$$= 0.73 \Rightarrow \text{yes}$$

4. The use of the **kernel trick** is key in writing efficient implementations of the **support vector machine** approach to predictive modelling. The kernel trick is based on the fact that the result of a **kernel function** applied to a support vector and a query instance is equivalent to the result of calculating the dot product between the support vector and the query instance after a specific set of basis functions have been applied to both—in other words, $\text{kernel}(\mathbf{d}, \mathbf{q}) = \boldsymbol{\phi}(\mathbf{d}) \cdot \boldsymbol{\phi}(\mathbf{q})$.
- (a) Using the support vector $\langle \mathbf{d}[1], \mathbf{d}[2] \rangle$ and the query instance $\langle \mathbf{q}[1], \mathbf{q}[2] \rangle$ as examples, show that applying a polynomial kernel with $p = 2$, $\text{kernel}(\mathbf{d}, \mathbf{q}) = (\mathbf{d} \cdot \mathbf{q} + 1)^2$, is equivalent to calculating the dot product of the support vector and query instance after applying the following set of basis functions:

$$\begin{aligned} \phi_0(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= \mathbf{d}[1]^2 & \phi_1(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= \mathbf{d}[2]^2 \\ \phi_2(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= \sqrt{2} \times \mathbf{d}[1] \times \mathbf{d}[2] & \phi_3(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= \sqrt{2} \times \mathbf{d}[1] \\ \phi_4(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= \sqrt{2} \times \mathbf{d}[2] & \phi_5(\langle \mathbf{d}[1], \mathbf{d}[2] \rangle) &= 1 \end{aligned}$$

To answer this question we should first calculate the result of applying the polynomial kernel function to the support vector and query instance:

$$\begin{aligned}
 \text{kernel}(\mathbf{d}, \mathbf{q}) &= (\mathbf{d} \cdot \mathbf{q} + 1)^2 \\
 &= (\langle \mathbf{d}[1], \mathbf{d}[2] \rangle \cdot \langle \mathbf{q}[1], \mathbf{q}[2] \rangle + 1)^2 \\
 &= (\mathbf{d}[1] \times \mathbf{q}[1] + \mathbf{d}[2] \times \mathbf{q}[2] + 1)^2 \\
 &= (\mathbf{d}[1] \times \mathbf{q}[1] + \mathbf{d}[2] \times \mathbf{q}[2] + 1) \times (\mathbf{d}[1] \times \mathbf{q}[1] + \mathbf{d}[2] \times \mathbf{q}[2] + 1) \\
 &= (\mathbf{d}[1] \times \mathbf{q}[1])^2 + (\mathbf{d}[1] \times \mathbf{q}[1] \times \mathbf{d}[2] \times \mathbf{q}[2]) + (\mathbf{d}[1] \times \mathbf{q}[1]) \\
 &\quad + (\mathbf{d}[2] \times \mathbf{q}[2] \times \mathbf{d}[1] \times \mathbf{q}[1]) + (\mathbf{d}[2] \times \mathbf{q}[2])^2 + (\mathbf{d}[2] \times \mathbf{q}[2]) \\
 &\quad + (\mathbf{d}[1] \times \mathbf{q}[1]) + (\mathbf{d}[2] \times \mathbf{q}[2]) + 1 \\
 &= (\mathbf{d}[1] \times \mathbf{q}[1])^2 + (\mathbf{d}[2] \times \mathbf{q}[2])^2 + 2 \times (\mathbf{d}[1] \times \mathbf{q}[1] \times \mathbf{d}[2] \times \mathbf{q}[2]) \\
 &\quad + 2 \times (\mathbf{d}[1] \times \mathbf{q}[1]) + 2 \times (\mathbf{d}[2] \times \mathbf{q}[2]) + 1
 \end{aligned}$$

We then apply the set of basis functions to the support vector

$$\begin{aligned}
 \phi(\mathbf{d}) &= \langle \phi_0(\mathbf{d}), \phi_1(\mathbf{d}), \phi_2(\mathbf{d}), \phi_3(\mathbf{d}), \phi_4(\mathbf{d}), \phi_5(\mathbf{d}) \rangle \\
 &= \langle \mathbf{d}[1]^2, \mathbf{d}[2]^2, \sqrt{2} \times \mathbf{d}[1] \times \mathbf{d}[2], \sqrt{2} \times \mathbf{d}[1], \sqrt{2} \times \mathbf{d}[2], 1 \rangle
 \end{aligned}$$

and to the query instance:

$$\begin{aligned}
 \phi(\mathbf{q}) &= \langle \phi_0(\mathbf{q}), \phi_1(\mathbf{q}), \phi_2(\mathbf{q}), \phi_3(\mathbf{q}), \phi_4(\mathbf{q}), \phi_5(\mathbf{q}) \rangle \\
 &= \langle \mathbf{q}[1]^2, \mathbf{q}[2]^2, \sqrt{2} \times \mathbf{q}[1] \times \mathbf{q}[2], \sqrt{2} \times \mathbf{q}[1], \sqrt{2} \times \mathbf{q}[2], 1 \rangle
 \end{aligned}$$

we then calculate the dot product between the transformed support vector and query instance as:

$$\begin{aligned}
 \phi(\mathbf{d}) \cdot \phi(\mathbf{q}) &= \langle \mathbf{d}[1]^2, \mathbf{d}[2]^2, \sqrt{2} \times \mathbf{d}[1] \times \mathbf{d}[2], \sqrt{2} \times \mathbf{d}[1], \sqrt{2} \times \mathbf{d}[2], 1 \rangle \\
 &\quad \cdot \langle \mathbf{q}[1]^2, \mathbf{q}[2]^2, \sqrt{2} \times \mathbf{q}[1] \times \mathbf{q}[2], \sqrt{2} \times \mathbf{q}[1], \sqrt{2} \times \mathbf{q}[2], 1 \rangle \\
 &= \mathbf{d}[1]^2 \times \mathbf{q}[1]^2 + \mathbf{d}[2]^2 \times \mathbf{q}[2]^2 + \sqrt{2} \times \mathbf{d}[1] \times \mathbf{d}[2] \times \sqrt{2} \times \mathbf{q}[1] \times \mathbf{q}[2] \\
 &\quad + \sqrt{2} \times \mathbf{d}[1] \times \sqrt{2} \times \mathbf{q}[1] + \sqrt{2} \times \mathbf{d}[2] \times \sqrt{2} \times \mathbf{q}[2] + 1 \times 1 \\
 &= (\mathbf{d}[1] \times \mathbf{q}[1])^2 + (\mathbf{d}[2] \times \mathbf{q}[2])^2 + 2 \times (\mathbf{d}[1] \times \mathbf{q}[1] \times \mathbf{d}[2] \times \mathbf{q}[2]) \\
 &\quad + 2 \times (\mathbf{d}[1] \times \mathbf{q}[1]) + 2 \times (\mathbf{d}[2] \times \mathbf{q}[2]) + 1
 \end{aligned}$$

This is equivalent to the the result of applying the polynomial kernel function calculated above which demonstrates that these two calculations are equivalent—in other words $\text{kernel}(\mathbf{d}, \mathbf{q}) = \phi(\mathbf{d}) \cdot \phi(\mathbf{q})$.

- (b) A support vector machine model has been trained to distinguish between dosages of two drugs that cause a dangerous interaction and those that interact safely. This model uses just two continuous features, DOSE1 and DOSE2, and two target levels, *dangerous* (the positive level, +1) and *safe* (the negative level, -1). The support vectors in the trained model are shown in the following table.

DOSE1	DOSE2	CLASS
0.2351	0.4016	+1
-0.1764	-0.1916	+1
0.3057	-0.9394	-1
0.5590	0.6353	-1
-0.6600	-0.1175	-1

In the trained model the value of w_0 is 0.3074, and the values of the α parameters are $\langle 7.1655, 6.9060, 2.0033, 6.1144, 5.9538 \rangle$.

- i. Using the version of the support vector machine prediction model that uses basis functions (see Equation 7.46) with the basis functions given in Part (a), calculate the output of the model for a query instance with DOSE1 = 0.90 and DOSE2 = -0.90.

The first step in this calculation is to transform the support vectors using the set of basis functions

$$\begin{aligned}\phi(\langle 0.2351, 0.4016 \rangle) &= \langle 0.0553, 0.1613, 0.1335, 0.3325, 0.5679, 1.0 \rangle \\ \phi(\langle -0.1764, -0.1916 \rangle) &= \langle 0.0311, 0.0367, 0.0478, -0.2495, -0.2710, 1.0 \rangle \\ \phi(\langle 0.3057, -0.9394 \rangle) &= \langle 0.0935, 0.8825, -0.4061, 0.4323, -1.3285, 1.0 \rangle \\ \phi(\langle 0.5590, 0.6353 \rangle) &= \langle 0.3125, 0.4036, 0.5022, 0.7905, 0.8984, 1.0 \rangle \\ \phi(\langle -0.6600, -0.1175 \rangle) &= \langle 0.4356, 0.0138, 0.1097, -0.9334, -0.1662, 1.0 \rangle\end{aligned}$$

The query instance then also needs to be transformed

$$\phi(\langle 0.91, -0.93 \rangle) = \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle$$

The output of the support vector machine can then be calculated as:

$$\begin{aligned}
 & \mathbb{M}_{\alpha, \phi, 0.3074}(\langle 0.91, -0.93 \rangle) \\
 &= (-1 \times 7.1655 \times (\langle 0.0553, 0.1613, 0.1335, 0.3325, 0.5679, 1.0 \rangle \\
 &\quad \cdot \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 6.9060 \times (\langle 0.0311, 0.0367, 0.0478, -0.2495, -0.2710, 1.0 \rangle \\
 &\quad \cdot \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 2.0033 \times (\langle 0.0935, 0.8825, -0.4061, 0.4323, -1.3285, 1.0 \rangle \\
 &\quad \cdot \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 6.1144 \times (\langle 0.3125, 0.4036, 0.5022, 0.7905, 0.8984, 1.0 \rangle \\
 &\quad \cdot \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 5.9538 \times (\langle 0.4356, 0.0138, 0.1097, -0.9334, -0.1662, 1.0 \rangle \\
 &\quad \cdot \langle 0.8281, 0.8649, -1.1968, 1.2869, -1.3152, 1.0 \rangle) + 0.3074) \\
 &= 5.3689 + 7.4596 - 8.9686 - 4.8438 - 1.2331 \\
 &= -2.2170
 \end{aligned}$$

Because the output of the model is negative the model makes a prediction of the negative level—*safe*.

- ii. Using the version of the support vector machine prediction model that uses a kernel function (see Equation 7.47) with the polynomial kernel function, calculate the output of the model for a query instance with DOSE1 = 0.22 and DOSE2 = 0.16.

The output of the model can be calculated as

$$\begin{aligned}
 & \mathbb{M}_{\alpha, \phi, 0.3074}(\langle 0.22, 0.16 \rangle) \\
 &= \left(-1 \times 7.1655 \times (\langle 0.2351, 0.4016 \rangle \cdot \langle 0.22, 0.16 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 6.9060 \times (\langle -0.1764, -0.1916 \rangle \cdot \langle 0.22, 0.16 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 2.0033 \times (\langle 0.3057, -0.9394 \rangle \cdot \langle 0.22, 0.16 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 6.1144 \times (\langle 0.559, 0.6353 \rangle \cdot \langle 0.22, 0.16 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 5.9538 \times (\langle -0.66, -0.1175 \rangle \cdot \langle 0.22, 0.16 \rangle + 1)^2 + 0.3074 \right) \\
 &= 9.2314 + 6.2873 - 1.3769 - 8.8624 - 3.8536 \\
 &= 1.4257
 \end{aligned}$$

Because the output of the model is positive, the model predicts the positive level—*dangerous*.

- iii. Verify that the answers calculated in Parts (i) and (ii) of this question would have been the same if the alternative approach (basis functions or the polynomial kernel function) had been used in each case.

First we will calculate the output of the model for the query instance $\langle 0.91, -0.93 \rangle$ using the polynomial kernel

$$\begin{aligned}
 M_{\mathbf{a}, \phi, 0.3074}(\langle 0.91, -0.93 \rangle) &= \left(-1 \times 7.1655 \times (\langle 0.2351, 0.4016 \rangle \cdot \langle 0.91, -0.93 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 6.9060 \times (\langle -0.1764, -0.1916 \rangle \cdot \langle 0.91, -0.93 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 2.0033 \times (\langle 0.3057, -0.9394 \rangle \cdot \langle 0.91, -0.93 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 6.1144 \times (\langle 0.559, 0.6353 \rangle \cdot \langle 0.91, -0.93 \rangle + 1)^2 + 0.3074 \right) \\
 &\quad + \left(1 \times 5.9538 \times (\langle -0.66, -0.1175 \rangle \cdot \langle 0.91, -0.93 \rangle + 1)^2 + 0.3074 \right) \\
 &= 5.3689 + 7.4596 - 8.9686 - 4.8438 - 1.2331 \\
 &= -2.2170
 \end{aligned}$$

This is the same result calculated previously, and would lead to the same prediction.

Next we will calculate the output of the model for the query instance $\langle 0.22, 0.16 \rangle$ using the set of basis functions. To this, first the query instance needs to be transformed using the basis functions

$$\phi(\langle 0.22, 0.16 \rangle) = \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle$$

The output of the support vector machine can then be calculated as:

$$\begin{aligned}
 & \mathbb{M}_{\alpha, \phi, 0.3074}(\langle 0.22, 0.16 \rangle) \\
 &= (-1 \times 7.1655 \times (\langle 0.0553, 0.1613, 0.1335, 0.3325, 0.5679, 1.0 \rangle \\
 &\quad \cdot \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 6.9060 \times (\langle 0.0311, 0.0367, 0.0478, -0.2495, -0.2710, 1.0 \rangle \\
 &\quad \cdot \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 2.0033 \times (\langle 0.0935, 0.8825, -0.4061, 0.4323, -1.3285, 1.0 \rangle \\
 &\quad \cdot \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 6.1144 \times (\langle 0.3125, 0.4036, 0.5022, 0.7905, 0.8984, 1.0 \rangle \\
 &\quad \cdot \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle) + 0.3074) \\
 &\quad + (1 \times 5.9538 \times (\langle 0.4356, 0.0138, 0.1097, -0.9334, -0.1662, 1.0 \rangle \\
 &\quad \cdot \langle 0.0484, 0.0256, 0.0498, 0.3111, 0.2263, 1.0 \rangle) + 0.3074) \\
 &= 9.2314 + 6.2873 - 1.3769 - 8.8624 - 3.8536 \\
 &= 1.4257
 \end{aligned}$$

Again, this output is the same as that calculated previously using the polynomial kernel function.

- iv. Compare the amount of computation required to calculate the output of the support vector machine using the polynomial kernel function with the amount required to calculate the output of the support vector machine using the basis functions.

It is clear, even in this small example, that the calculation using the polynomial kernel function is much more efficient than the calculation using the basis function transformation. Making the transformation using the basis functions and calculating the dot product in this higher dimensional space takes much more computational effort than calculating the polynomial kernel function. This is the advantage of the kernel trick.

- * 5. In building multivariate logistic regression models, it is recommended that all continuous descriptive features be normalized to the range $[-1, 1]$. The following table shows a data quality report for the dataset used to train the model described in Question 3.

Feature	Count	%		Min.	1 st			3 rd		Std. Dev.
		Miss.	Card.		Qrt.	Mean	Median	Qrt.	Max.	
AGE	5,200	6	40	18	22	32.7	32	32	63	12.2
SHOP FREQUENCY	5,200	0	316	0.2	1.0	2.2	1.3	4.3	5.4	1.6
SHOP VALUE	5,200	0	3,730	5	11.8	101.9	100.14	174.6	230.7	72.1

Feature	Count	% Miss.		Card.	Mode	Mode Count	Mode %	2 nd Mode	2 nd	2 nd
									Mode Count	Mode %
SOCIOECONOMIC BAND	5,200	8	3		a	2,664	51.2	b	1,315	25.3
REPEAT PURCHASE	5,200	0	2		no	2,791	53.7	yes	2,409	46.3

On the basis of the information in this report, all continuous features were normalized using **range normalization**, and any missing values were replaced using **mean imputation** for continuous features and **mode imputation** for categorical features. After applying these data preparation operations, a multivariate logistic regression model was trained to give the weights shown in the following table.

Feature	Weight
Intercept ($w[0]$)	0.6679
AGE	-0.5795
SOCIOECONOMIC BAND B	-0.1981
SOCIOECONOMIC BAND C	-0.2318
SHOP VALUE	3.4091
SHOP FREQUENCY	2.0499

Use this model to make predictions for each of the query instances shown in the following table (question marks refer to missing values).

ID	AGE	SOCIOECONOMIC	SHOP	SHOP
		BAND	FREQUENCY	VALUE
1	38	a	1.90	165.39
2	56	b	1.60	109.32
3	18	c	6.00	10.09
4	?	b	1.33	204.62
5	62	?	0.85	110.50

Before inserting the descriptive feature values into the regression model, the same data preparation operations performed on the training data (range normalization and imputation) must be performed on each query instance.

The first step is to handle the missing values. Both the AGE and SOCIOECONOMIC BAND features have small numbers of missing values. In this case we have decided to use mean imputation for continuous features and mode imputation for categorical features. The mean for the AGE feature is 32.7, and the mode for the SOCIOECONOMIC BAND feature is *a*. So the query instances are changed to reflect this as shown in the following table.

ID	SOCIOECONOMIC		SHOP	SHOP
	AGE	BAND	FREQUENCY	VALUE
1	38	a	1.90	165.39
2	56	b	1.60	109.32
3	18	c	6.00	10.09
4	32.7	b	1.33	204.62
5	62	a	0.85	110.50

Each continuous descriptive feature value must then be normalized using the same normalization parameters that were used with the training data. Range normalization was used in this case, so the required minimum and maximum parameters can be obtained from the data quality report. As a reminder, the normalization of the AGE feature for the first instance is calculated as

$$\begin{aligned}
 AGE'_1 &= \frac{AGE_1 - \min(AGE)}{\max(AGE) - \min(AGE)} \times (high - low) + low \\
 &= \frac{38 - 18}{63 - 18} \times (1 - (-1)) + (-1) \\
 &= -0.111
 \end{aligned}$$

These normalized parameters are shown in the following table.

ID	SOCIOECONOMIC		SHOP	SHOP
	AGE	BAND	FREQUENCY	VALUE
1	-0.111	a	-0.346	0.421
2	0.689	b	-0.462	-0.076
3	-1.000	c	1.230	-0.955
4	-0.347	b	-0.565	0.769
5	0.956	a	-0.750	-0.065

One issue that must be considered in this case is what to do with values that are outside the range of the training data. In this case the SHOW FREQUENCY value for query instance 3 is 6, which is higher than anything that was seen in the training data. When range normalization is applied to this value, it is calculated as 1.23. This can be handled by the regression equation, but many modelers will choose to clamp this at a maximum value of 1.0 (similarly, a value below the minimum seen would be clamped at -1.0). We choose to clamp at 1.0 in this case.

With this information, we can calculate the predictions by plugging these values into the regression equation as follows:

$$\begin{aligned}
 \mathbf{1:} \quad & Logistic(0.6679 - 0.5795 \times -0.111 - 0.1981 \times 0 - 0.2318 \times 0 + 2.0499 \times \\
 & -0.346 + 3.4091 \times 0.421)
 \end{aligned}$$

$$= \text{Logistic}(1.46) = \frac{1}{1+e^{-1.46}}$$

$$= 0.811 \Rightarrow \text{yes}$$

$$\mathbf{2: Logistic}(0.6679 - 0.5795 \times 0.6889 - 0.1981 \times 1 - 0.2318 \times 0 + 2.0499 \times -0.4615 + 3.4091 \times -0.0756)$$

$$= \text{Logistic}(-1.1332) = \frac{1}{1+e^{1.1332}}$$

$$= 0.2436 \Rightarrow \text{no}$$

$$\mathbf{3: Logistic}(0.6679 - 0.5795 \times -1 - 0.1981 \times 0 - 0.2318 \times 1 + 2.0499 \times 1.2308 + 3.4091 \times -0.9549)$$

$$= \text{Logistic}(0.2832) = \frac{1}{1+e^{-0.2832}}$$

$$= 0.5703 \Rightarrow \text{yes}$$

$$\mathbf{4: Logistic}(0.6679 - 0.5795 \times -0.3467 - 0.1981 \times 1 - 0.2318 \times 0 + 2.0499 \times -0.5654 + 3.4091 \times 0.7689)$$

$$= \text{Logistic}(2.1330) = \frac{1}{1+e^{-2.1330}}$$

$$= 0.8940 \Rightarrow \text{yes}$$

$$\mathbf{5: Logistic}(0.6679 - 0.5795 \times 0.9556 - 0.1981 \times 0 - 0.2318 \times 0 + 2.0499 \times -0.7500 + 3.4091 \times -0.0651)$$

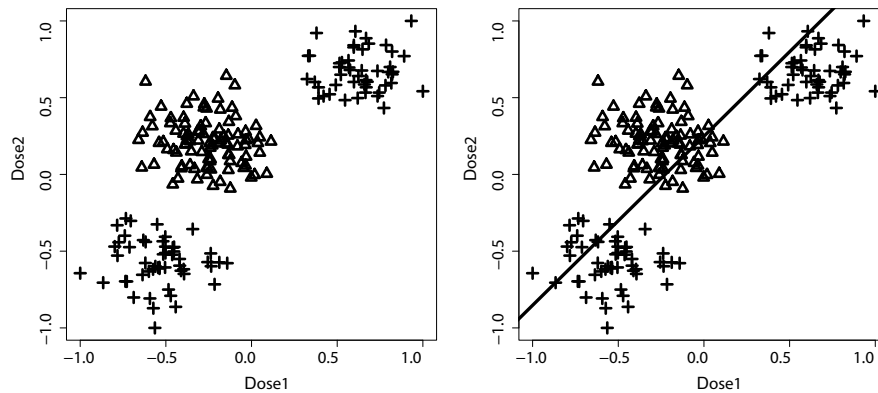
$$= \text{Logistic}(-1.6453) = \frac{1}{1+e^{1.6453}}$$

$$= 0.1617 \Rightarrow \text{no}$$

One interesting thing to note about these calculations is that the second query instance in this question is the same as the first query instance in the previous question. Note that the resulting answers are almost identical. This is reassuring, as simple data preparation operations like those applied in this example should not have a significant impact on the resulting model.

- * 6. The effects that can occur when different drugs are taken together can be difficult for doctors to predict. Machine learning models can be built to help predict optimal dosages of drugs so as to achieve a medical practitioner's goals.² In the following figure, the image on the left shows a scatter plot of a dataset used to train a model to distinguish between dosages of two drugs that cause a dangerous interaction and those that cause a safe interaction. There are just two continuous features in this dataset, DOSE1 and DOSE2 (both normalized to the range $(-1, 1)$ using range normalization), and two target levels, *dangerous* and *safe*. In the scatter plot, DOSE1 is shown on the horizontal axis, DOSE2 is shown on the vertical axis, and the shapes of the points represent the target level—crosses represent *dangerous* interactions and triangles represent *safe* interactions.

2. The data used in this question has been artificially generated for this book. ? is, however, a good example of prediction models used to help doctors select correct drug dosages.



In the preceding figure, the image on the right shows a simple linear logistic regression model trained to perform this task. This model is

$$P(\text{TYPE} = \text{dangerous}) = \text{Logistic}(0.6168 + 2.7320 \times \text{DOSE1} - 2.4809 \times \text{DOSE2})$$

Plainly, this model is not performing well.

- (a) Would the similarity-based, information-based, or probability-based predictive modeling approaches already covered in this book be likely to do a better job of learning this model than the simple linear regression model?

The problem in this case is that the decision boundary is non-linear, so a linear logistic regression model is not capable of capturing it accurately. A similarity-based model would be an obvious candidate for this problem. Because similarity-based models are local learners, they have no problem learning the type of decision boundary required to separate these two groups. A decision tree could also do this, although continuous features such as those in this problem can lead to very complex decision trees. Similarly, although a probability-based approach could work, it may be awkward to handle the continuous descriptive features appropriately.

- (b) A simple approach to adapting a logistic regression model to learn this type of decision boundary is to introduce a set of basis functions that will allow a non-linear decision boundary to be learned. In this case, a set of basis functions that generate a cubic decision boundary will work well. An appropriate set of basis functions is as follows:

$$\begin{aligned}
\phi_0(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= 1 & \phi_1(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE1} \\
\phi_2(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE2} & \phi_3(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE1}^2 \\
\phi_4(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE2}^2 & \phi_5(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE1}^3 \\
\phi_6(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE2}^3 & \phi_7(\langle \text{DOSE1}, \text{DOSE2} \rangle) &= \text{DOSE1} \times \text{DOSE2}
\end{aligned}$$

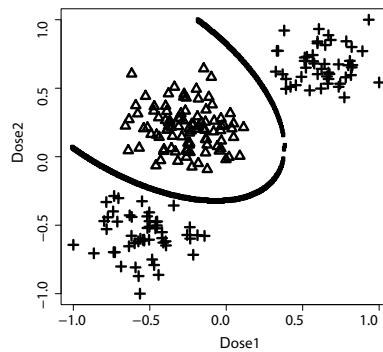
Training a logistic regression model using this set of basis functions leads to the following model:

$$\begin{aligned}
P(\text{TYPE} = \text{dangerous}) = \\
\text{Logistic} \big(& -0.848 \times \phi_0(\langle \text{DOSE1}, \text{DOSE2} \rangle) + 1.545 \times \phi_1(\langle \text{DOSE1}, \text{DOSE2} \rangle) \\
& - 1.942 \times \phi_2(\langle \text{DOSE1}, \text{DOSE2} \rangle) + 1.973 \times \phi_3(\langle \text{DOSE1}, \text{DOSE2} \rangle) \\
& + 2.495 \times \phi_4(\langle \text{DOSE1}, \text{DOSE2} \rangle) + 0.104 \times \phi_5(\langle \text{DOSE1}, \text{DOSE2} \rangle) \\
& + 0.095 \times \phi_6(\langle \text{DOSE1}, \text{DOSE2} \rangle) + 3.009 \times \phi_7(\langle \text{DOSE1}, \text{DOSE2} \rangle) \big)
\end{aligned}$$

Use this model to make predictions for the following query instances:

ID	DOSE1	DOSE2
1	0.50	0.75
2	0.10	0.75
3	-0.47	-0.39
4	-0.47	0.18

The image below shows the decision boundary that is learned.



The first step in making a prediction is to generate the outputs of the basis functions. This is done for the first query as follows:

$$\begin{aligned}
\phi_0(\langle 0.50, 0.75 \rangle) &= 1 & \phi_4(\langle 0.50, 0.75 \rangle) &= 0.5625 \\
\phi_1(\langle 0.50, 0.75 \rangle) &= 0.50 & \phi_5(\langle 0.50, 0.75 \rangle) &= 0.1250 \\
\phi_2(\langle 0.50, 0.75 \rangle) &= 0.75 & \phi_6(\langle 0.50, 0.75 \rangle) &= 0.4219 \\
\phi_3(\langle 0.50, 0.75 \rangle) &= 0.25 & \phi_7(\langle 0.50, 0.75 \rangle) &= 0.3750
\end{aligned}$$

We can now use the regression model to make a prediction:

$$\begin{aligned}
P(\text{TYPE} = \text{dangerous}) &= \text{Logistic}(-0.848 \times 1 + 1.545 \times 0.50 - 1.942 \times 0.75 + 1.973 \times 0.25 \\
&\quad + 2.495 \times 0.5625 + 0.104 \times 0.1250 + 0.095 \times 0.4219 + 3.009 \times 0.3750) \\
&= \text{Logistic}(1.5457) \\
&= 0.8243
\end{aligned}$$

This means that the probability of the query dosages causing a *dangerous* interaction is 0.8243, so we would say that the result for this query is *dangerous*.

We just repeat this for the next query $\langle 0.10, 0.75 \rangle$:

$$\begin{aligned}
\phi_0(\langle 0.10, 0.75 \rangle) &= 1 & \phi_4(\langle 0.10, 0.75 \rangle) &= 0.5625 \\
\phi_1(\langle 0.10, 0.75 \rangle) &= 0.10 & \phi_5(\langle 0.10, 0.75 \rangle) &= 0.0010 \\
\phi_2(\langle 0.10, 0.75 \rangle) &= 0.75 & \phi_6(\langle 0.10, 0.75 \rangle) &= 0.4219 \\
\phi_3(\langle 0.10, 0.75 \rangle) &= 0.01 & \phi_7(\langle 0.10, 0.75 \rangle) &= 0.0750
\end{aligned}$$

We can now use the regression model to make a prediction:

$$\begin{aligned}
P(\text{TYPE} = \text{dangerous}) &= \text{Logistic}(-0.848 \times 1 + 1.545 \times 0.10 - 1.942 \times 0.75 + 1.973 \times 0.01 \\
&\quad + 2.495 \times 0.5625 + 0.104 \times 0.0010 + 0.095 \times 0.4219 + 3.009 \times 0.0750) \\
&= \text{Logistic}(-0.4613) \\
&= 0.3867
\end{aligned}$$

This means that the probability of the query dosages causing *dangerous* interaction is 0.3867, so we would say that these dosages are *safe* together.

And for the next query $\langle -0.47, -0.50 \rangle$:

$$\begin{aligned}
\phi_0(\langle -0.47, -0.50 \rangle) &= 1 & \phi_4(\langle -0.47, -0.50 \rangle) &= 0.2500 \\
\phi_1(\langle -0.47, -0.50 \rangle) &= -0.47 & \phi_5(\langle -0.47, -0.50 \rangle) &= -0.1038 \\
\phi_2(\langle -0.47, -0.50 \rangle) &= -0.50 & \phi_6(\langle -0.47, -0.50 \rangle) &= -0.1250 \\
\phi_3(\langle -0.47, -0.50 \rangle) &= 0.2209 & \phi_7(\langle -0.47, -0.50 \rangle) &= 0.2350
\end{aligned}$$

We can now use the regression model to make a prediction:

$$\begin{aligned}
 P(\text{TYPE} = \text{dangerous}) &= \text{Logistic}(-0.848 \times 1 + 1.545 \times -0.47 - 1.942 \times -0.50 + 1.973 \times 0.2209 \\
 &\quad + 2.495 \times 0.25 + 0.104 \times -0.1038 + 0.095 \times -0.1250 + 3.009 \times 0.2350) \\
 &= \text{Logistic}(1.1404) \\
 &= 0.7577
 \end{aligned}$$

This means that the probability of the query document causing a *dangerous* interaction is 0.7577, so we would return a *dangerous* prediction.

And for the last query $\langle -0.47, 0.18 \rangle$:

$$\begin{aligned}
 \phi_0(\langle -0.47, 0.18 \rangle) &= 1 & \phi_4(\langle -0.47, 0.18 \rangle) &= 0.0324 \\
 \phi_1(\langle -0.47, 0.18 \rangle) &= -0.47 & \phi_5(\langle -0.47, 0.18 \rangle) &= -0.1038 \\
 \phi_2(\langle -0.47, 0.18 \rangle) &= 0.18 & \phi_6(\langle -0.47, 0.18 \rangle) &= 0.0058 \\
 \phi_3(\langle -0.47, 0.18 \rangle) &= 0.2209 & \phi_7(\langle -0.47, 0.18 \rangle) &= -0.0846
 \end{aligned}$$

We can now use the regression model to make a prediction:

$$\begin{aligned}
 P(\text{TYPE} = \text{dangerous}) &= \text{Logistic}(-0.848 \times 1 + 1.545 \times -0.47 - 1.942 \times 0.18 + 1.973 \times 0.2209 \\
 &\quad + 2.495 \times 0.0324 + 0.104 \times -0.1038 + 0.095 \times 0.0058 + 3.009 \times -0.0846) \\
 &= \text{Logistic}(-1.672106798) \\
 &= 0.1581
 \end{aligned}$$

This means that the probability of the query dosages causing a *dangerous* interaction is 0.1581, so we would say that, instead, this is a *safe* dosage pair.

- * 7. The following **multinomial logistic regression** model predicts the TYPE of a retail customer (*single*, *family*, or *business*) on the basis of the average amount that they spend per visit, SPEND, and the average frequency of their visits, FREQ:

$$\begin{aligned}
 \mathbb{M}_{\mathbf{w}_{\text{single}}}(\mathbf{q}) &= \text{logistic}(0.7993 - 15.9030 \times \text{SPEND} + 9.5974 \times \text{FREQ}) \\
 \mathbb{M}_{\mathbf{w}_{\text{family}}}(\mathbf{q}) &= \text{logistic}(3.6526 + -0.5809 \times \text{SPEND} - 17.5886 \times \text{FREQ}) \\
 \mathbb{M}_{\mathbf{w}_{\text{business}}}(\mathbf{q}) &= \text{logistic}(4.6419 + 14.9401 \times \text{SPEND} - 6.9457 \times \text{FREQ})
 \end{aligned}$$

Use this model to make predictions for the following query instances:

ID	SPEND	FREQ
1	-0.62	0.10
2	-0.43	-0.71
3	0.00	0.00

The first thing that we need to do is make a prediction for a query instance for each of the three models. For the query instance $\langle 0.10, -0.62 \rangle$, these are

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_1} P(\text{TYPE} = \textit{single}) &= \text{Logistic}(-1.6114 - 3.4857 \times -0.62 + 2.0356 \times 0.10) \\ &= \text{Logistic}(0.7533) \\ &= 0.6799\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_2} P(\text{TYPE} = \textit{business}) &= \text{Logistic}(-0.6692 + 3.6468 \times -0.62 + 1.7487 \times 0.10) \\ &= \text{Logistic}(-2.7553) \\ &= 0.0598\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_3} P(\text{TYPE} = \textit{family}) &= \text{Logistic}(-0.9070 - 0.1134 \times -0.62 - 4.2036 \times 0.10) \\ &= \text{Logistic}(-1.2571) \\ &= 0.2215\end{aligned}$$

We then combine these results to get a probability for each target level as follows:

$$P(t = \textit{level}_i \mid \mathbf{d}) = \frac{\mathbb{M}_{\mathbf{w}_i}(\mathbf{d})}{\sum_{l \in \textit{levels}(t)} \mathbb{M}_{\mathbf{w}_l}(\mathbf{d})}$$

So,

$$\begin{aligned}P(\text{TYPE} = \textit{single}) &= \frac{0.6799}{(0.6799 + 0.0598 + 0.2215)} \\ &= 0.7074\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{business}) &= \frac{0.0598}{(0.6799 + 0.0598 + 0.2215)} \\ &= 0.0622\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{family}) &= \frac{0.2215}{(0.6799 + 0.0598 + 0.2215)} \\ &= 0.2304\end{aligned}$$

This implies that the most likely target level for this query is *single*, so we would predict that this shopper is from a single household.

We repeat for the next query, $\langle -0.43, -0.71 \rangle$:

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_1} P(\text{TYPE} = \textit{single}) &= \text{Logistic}(-1.6114 - 3.4857 \times -0.43 + 2.0356 \times -0.71) \\ &= \text{Logistic}(-1.5578) \\ &= 0.1740\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_2} P(\text{TYPE} = \textit{business}) &= \text{Logistic}(-0.6692 + 3.6468 \times -0.43 + 1.7487 \times -0.71) \\ &= \text{Logistic}(-3.4789) \\ &= 0.0299\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{\mathbf{w}_3} P(\text{TYPE} = \textit{family}) &= \text{Logistic}(-0.9070 - 0.1134 \times -0.43 - 4.2036 \times -0.71) \\ &= \text{Logistic}(2.126295434) \\ &= 0.8934\end{aligned}$$

We then combine these results to get a probability for each target level as follows:

$$\begin{aligned}P(\text{TYPE} = \textit{single}) &= \frac{0.1740}{(0.1740 + 0.0299 + 0.8934)} \\ &= 0.1585\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{business}) &= \frac{0.0299}{(0.1740 + 0.0299 + 0.8934)} \\ &= 0.0273\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{family}) &= \frac{0.8934}{(0.1740 + 0.0299 + 0.8934)} \\ &= 0.8142\end{aligned}$$

This implies that the most likely target level for this query is *business*, so we would predict that this shopper is from a business.

We repeat for the next query, $\langle 0.0, 0.0 \rangle$:

$$\begin{aligned}\mathbb{M}_{w_1} P(\text{TYPE} = \textit{single}) &= \textit{Logistic}(-1.6114 - 3.4857 \times 0.0 + 2.0356 \times 0.0) \\ &= \textit{Logistic}(-1.6114) \\ &= 0.1663\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{w_2} P(\text{TYPE} = \textit{business}) &= \textit{Logistic}(-0.6692 + 3.6468 \times 0.0 + 1.7487 \times 0.0) \\ &= \textit{Logistic}(-0.6692) \\ &= 0.3387\end{aligned}$$

$$\begin{aligned}\mathbb{M}_{w_3} P(\text{TYPE} = \textit{family}) &= \textit{Logistic}(-0.9070 - 0.1134 \times 0.0 - 4.2036 \times 0.0) \\ &= \textit{Logistic}(-0.9070) \\ &= 0.2876\end{aligned}$$

We then combine these results to get a probability for each target level as follows:

$$\begin{aligned}P(\text{TYPE} = \textit{single}) &= \frac{0.1663}{(0.1663 + 0.3387 + 0.2876)} \\ &= 0.2099\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{business}) &= \frac{0.3387}{(0.1663 + 0.3387 + 0.2876)} \\ &= 0.4273\end{aligned}$$

$$\begin{aligned}P(\text{TYPE} = \textit{family}) &= \frac{0.2876}{(0.1663 + 0.3387 + 0.2876)} \\ &= 0.3628\end{aligned}$$

This query is interesting as it is right in the middle of the feature space in a region in which no target level dominates. For this reason, there is not a strong probability associated with any of the levels, just a slight preference for *business*.

- * 8. A support vector machine has been built to predict whether a patient is at risk of cardiovascular disease. In the dataset used to train the model, there are two target levels—*high risk* (the positive level, +1) or *low risk* (the negative level, −1)—and three descriptive features—AGE, BMI, and BLOOD PRESSURE. The support vectors in the trained model are shown in the table below (all descriptive feature values have been standardized).

AGE	BMI	BLOOD	
		PRESSURE	RISK
-0.4549	0.0095	0.2203	low risk
-0.2843	-0.5253	0.3668	low risk
0.3729	0.0904	-1.0836	high risk
0.558	0.2217	0.2115	high risk

In the model the value of w_0 is -0.0216 , and the values of the α parameters are $\langle 1.6811, 0.2384, 0.2055, 1.7139 \rangle$. What predictions would this model make for the following query instances?

ID	AGE	BMI	BLOOD	
			PRESSURE	
1	-0.8945	-0.3459	0.5520	
2	0.4571	0.4932	-0.4768	
3	-0.3825	-0.6653	0.2855	
4	0.7458	0.1253	-0.7986	

For the first query instance, $\langle -0.8945, -0.3459, 0.552 \rangle$, the output of the support vector machine model is

$$\begin{aligned}
& \mathbb{M}_{\alpha, -0.0216}(\langle -0.8945, -0.3459, 0.552 \rangle) \\
&= (-1 \times 1.6811 \times (\langle -0.4549, 0.0095, 0.2203 \rangle \cdot \langle -0.8945, -0.3459, 0.552 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2384 \times (\langle -0.2843, -0.5253, 0.3668 \rangle \cdot \langle -0.8945, -0.3459, 0.552 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2055 \times (\langle 0.3729, 0.0904, -1.0836 \rangle \cdot \langle -0.8945, -0.3459, 0.552 \rangle) - 0.0216) \\
&\quad + (-1 \times 1.7139 \times (\langle 0.558, 0.2217, 0.2115 \rangle \cdot \langle -0.8945, -0.3459, 0.552 \rangle) - 0.0216) \\
&= -0.9045 - 0.1737 - 0.2195 - 0.8083 \\
&= -2.1061
\end{aligned}$$

Because this output is less than -1 , the model predicts the negative level—*low risk*.

For the second query instance, $\langle 0.4571, 0.4932, -0.4768 \rangle$, the output of the sup-

port vector machine model is

$$\begin{aligned}
& \mathbb{M}_{\alpha, -0.0216}(\langle 0.4571, 0.4932, -0.4768 \rangle) \\
&= (-1 \times 1.6811 \times (\langle -0.4549, 0.0095, 0.2203 \rangle \cdot \langle 0.4571, 0.4932, -0.4768 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2384 \times (\langle -0.2843, -0.5253, 0.3668 \rangle \cdot \langle 0.4571, 0.4932, -0.4768 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2055 \times (\langle 0.3729, 0.0904, -1.0836 \rangle \cdot \langle 0.4571, 0.4932, -0.4768 \rangle) - 0.0216) \\
&\quad + (-1 \times 1.7139 \times (\langle 0.558, 0.2217, 0.2115 \rangle \cdot \langle 0.4571, 0.4932, -0.4768 \rangle) - 0.0216) \\
&= 0.4967 + 0.1129 + 0.1288 + 0.4302 \\
&= 1.1686
\end{aligned}$$

Because this output is greater than 1, the model predicts the positive level—*high risk*.

For the third query instance, $\langle -0.3825, -0.6653, 0.2855 \rangle$, the output of the support vector machine model is

$$\begin{aligned}
& \mathbb{M}_{\alpha, -0.0216}(\langle -0.3825, -0.6653, 0.2855 \rangle) \\
&= (-1 \times 1.6811 \times (\langle -0.4549, 0.0095, 0.2203 \rangle \cdot \langle -0.3825, -0.6653, 0.2855 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2384 \times (\langle -0.2843, -0.5253, 0.3668 \rangle \cdot \langle -0.3825, -0.6653, 0.2855 \rangle) - 0.0216) \\
&\quad + (-1 \times 0.2055 \times (\langle 0.3729, 0.0904, -1.0836 \rangle \cdot \langle -0.3825, -0.6653, 0.2855 \rangle) - 0.0216) \\
&\quad + (-1 \times 1.7139 \times (\langle 0.558, 0.2217, 0.2115 \rangle \cdot \langle -0.3825, -0.6653, 0.2855 \rangle) - 0.0216) \\
&= -0.4092 + -0.1557 + -0.1268 + -0.5367 \\
&= -1.2284
\end{aligned}$$

Because this output is less than -1 , the model predicts the negative level—*low risk*.

For the final query instance, $\langle 0.7458, 0.1253, -0.7986 \rangle$, the output of the support

vector machine model is

$$\begin{aligned}
 & \mathbb{M}_{\alpha, -0.0216}(\langle 0.7458, 0.1253, -0.7986 \rangle) \\
 &= (-1 \times 1.6811 \times (\langle -0.4549, 0.0095, 0.2203 \rangle \cdot \langle 0.7458, 0.1253, -0.7986 \rangle) - 0.0216) \\
 &\quad + (-1 \times 0.2384 \times (\langle -0.2843, -0.5253, 0.3668 \rangle \cdot \langle 0.7458, 0.1253, -0.7986 \rangle) - 0.0216) \\
 &\quad + (-1 \times 0.2055 \times (\langle 0.3729, 0.0904, -1.0836 \rangle \cdot \langle 0.7458, 0.1253, -0.7986 \rangle) - 0.0216) \\
 &\quad + (-1 \times 1.7139 \times (\langle 0.558, 0.2217, 0.2115 \rangle \cdot \langle 0.7458, 0.1253, -0.7986 \rangle) - 0.0216) \\
 &= 0.8425 + 0.1145 + 0.2158 + 0.4498 \\
 &= 1.6227
 \end{aligned}$$

Because this output is greater than 1, the model predicts the positive level—*high risk*.

- * 9. A multivariate logistic regression model has been built to diagnose breast cancer in patients on the basis of features extracted from tissue samples extracted by biopsy.³ The model uses three descriptive features—MITOSES, a measure of how fast cells are growing; CLUMPTHICKNESS, a measure of the amount of layering in cells; and BLANDCHROMATIN, a measure of the texture of cell nuclei—and predicts the status of a biopsy as either *benign* or *malignant*. The weights in the trained model are shown in the following table.

Feature	Weight
Intercept ($\mathbf{w}[0]$)	−13.92
MITOSES	3.09
CLUMPTHICKNESS	0.63
BLANDCHROMATIN	1.11

3. The data in this question has been artificially created but is inspired by the famous **Wisconsin breast cancer dataset** first described in ? and is available from the UCI Machine Learning Repository (?).

- (a) Use this model to make predictions for each of the following query instances.

ID	MITOSES	CLUMP	BLAND
		THICKNESS	CHROMATIN
1	7	4	3
2	3	5	1
3	3	3	3
4	5	3	1
5	7	4	4
6	10	4	1
7	5	2	1

To calculate the predictions made by the model we insert the descriptive feature values from each query instance into the prediction model. With this information, the predictions can be made as follows:

$$\begin{aligned}
 \mathbf{1:} & \text{Logistic}(-13.92 + 3.09 \times 3 + 0.63 \times 7 + 1.11 \times 4) \\
 &= \text{Logistic}(4.2) = \frac{1}{1 + e^{-4.2}} \\
 &= 0.985 \Rightarrow \text{malignant}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{2:} & \text{Logistic}(-13.92 + 3.09 \times 1 + 0.63 \times 3 + 1.11 \times 5) \\
 &= \text{Logistic}(-3.39) = \frac{1}{1 + e^{3.39}} \\
 &= 0.033 \Rightarrow \text{benign}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{3:} & \text{Logistic}(-13.92 + 3.09 \times 3 + 0.63 \times 3 + 1.11 \times 3) \\
 &= \text{Logistic}(0.57) = \frac{1}{1 + e^{-0.57}} \\
 &= 0.639 \Rightarrow \text{malignant}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{4:} & \text{Logistic}(-13.92 + 3.09 \times 1 + 0.63 \times 5 + 1.11 \times 3) \\
 &= \text{Logistic}(-4.35) = \frac{1}{1 + e^{4.35}} \\
 &= 0.013 \Rightarrow \text{benign}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{5:} & \text{Logistic}(-13.92 + 3.09 \times 4 + 0.63 \times 7 + 1.11 \times 4) \\
 &= \text{Logistic}(7.29) = \frac{1}{1 + e^{-7.29}} \\
 &= 0.999 \Rightarrow \text{malignant}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{6:} & \text{Logistic}(-13.92 + 3.09 \times 1 + 0.63 \times 10 + 1.11 \times 4) \\
 &= \text{Logistic}(-0.09) = \frac{1}{1 + e^{0.09}} \\
 &= 0.478 \Rightarrow \text{benign}
 \end{aligned}$$

$$\begin{aligned}
7: & \text{Logistic}(-13.92 + 3.09 \times 1 + 0.63 \times 5 + 1.11 \times 2) \\
& = \text{Logistic}(-5.46) = \frac{1}{1 + e^{5.46}} \\
& = 0.004 \Rightarrow \text{benign}
\end{aligned}$$

- (b) The following are the ground truth labels for the query instances from Part (a).

\mathbf{d}_1	\mathbf{d}_2	\mathbf{d}_3	\mathbf{d}_4	\mathbf{d}_5	\mathbf{d}_6	\mathbf{d}_7
<i>benign</i>	<i>benign</i>	<i>malignant</i>	<i>benign</i>	<i>malignant</i>	<i>malignant</i>	<i>benign</i>

- i. Using the ground truth labels, calculate the **squared error** loss for each query instance (assume that *benign* = 0 and *malignant* = 1).

Squared error loss is easily calculated as

$$(t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i))^2$$

for each instance. These values are shown below.

ID	MITOSES	CLUMP THICKNESS	BLAND CHROMATIN	CLASS	NUMERIC CLASS	$\mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)$	Squared Error
1	7	4	3	<i>benign</i>	0	0.985	0.970
2	3	5	1	<i>benign</i>	0	0.033	0.001
3	3	3	3	<i>malignant</i>	1	0.639	0.130
4	5	3	1	<i>benign</i>	0	0.013	0.000
5	7	4	4	<i>malignant</i>	1	0.999	0.000
6	10	4	1	<i>malignant</i>	1	0.478	0.272
7	5	2	1	<i>benign</i>	0	0.004	0.000

- ii. **Categorical cross entropy** is another loss function that is commonly used for classification models. Categorical cross entropy is defined as

$$-(t_i \times \ln(\mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)) + (1 - t_i) \times \ln(1 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)))$$

Using the ground truth labels previously given, calculate the categorical cross entropy for the query set. Compare these values to the squared error loss values for each instance.

For each instance we can calculate categorical cross as described above. These values are shown below.

ID	MITOSES	CLUMP		BLAND		NUMERIC		Categorical
		THICKNESS	CHROMATIN	CLASS	CLASS	$\mathbb{M}_w(\mathbf{d}_i)$	Entropy	Cross
1	7	4	3	<i>benign</i>	0	0.985	4.200	
2	3	5	1	<i>benign</i>	0	0.033	0.034	
3	3	3	3	<i>malignant</i>	1	0.639	0.448	
4	5	3	1	<i>benign</i>	0	0.013	0.013	
5	7	4	4	<i>malignant</i>	1	0.999	0.001	
6	10	4	1	<i>malignant</i>	1	0.478	0.738	
7	5	2	1	<i>benign</i>	0	0.004	0.004	

The key thing to note about these values is that the categorical cross entropy loss function emphasises predictions that are incorrect and made with high confidence. In this query set instance \mathbf{d}_1 is incorrectly predicted with high confidence, 0.985, and so a very high cross entropy is results, 4.200. An incorrect prediction is also made for instance \mathbf{d}_6 , but this time with very low confidence, 0.478. The categorical cross entropy is 0.738. Comparing the squared errors for these two instances calculated in the previous part, 0.970 and 0.272. to these categorical cross entropy scores we see the further emphasis on high confidence incorrect predictions.

- * 10. The following images are handwritten instances of the digits 0 and 1.⁴ The images are small, 8 pixels by 8 pixels, and each pixel contains a gray level from the range $[0, 7]$.



Rather than use individual pixel values, which can lead to very high-dimensional feature vectors, a simpler way to represent images for use with regression models is to calculate a histogram for each image and use this as the feature vector instead. In this case the histograms simply count the frequency of occurrence of each possible gray level in each image. The table that follows shows the histograms for a small dataset of 16 images split between examples of digits 0 and 1.

4. These images are based on the dataset from the UCI Machine Learning repository[?] and originally described by[?].

ID	GL-0	GL-1	GL-2	GL-3	GL-4	GL-5	GL-6	GL-7	DIGIT
0	31	3	6	2	7	5	6	4	0
1	37	3	1	4	1	3	2	13	1
2	31	3	4	1	8	7	3	7	0
3	38	2	3	0	1	1	5	14	1
4	31	5	3	2	5	2	5	11	0
5	32	6	3	2	1	1	5	14	1
6	31	3	5	2	3	6	2	12	0
7	31	4	3	4	1	5	5	11	0
8	38	4	2	2	2	4	4	8	1
9	38	3	2	3	4	4	1	9	1

A logistic regression model has been trained to classify digits as either 0 or 1. The weights in this model are as follows:

Intercept	GL-0	GL-1	GL-2	GL-3	GL-4	GL-5	GL-6	GL-7
$w[0]$	$w[1]$	$w[2]$	$w[3]$	$w[4]$	$w[5]$	$w[6]$	$w[7]$	$w[8]$
0.309	0.100	-0.152	-0.163	0.191	-0.631	-0.716	-0.478	-0.171

This model has been used to make predictions for the instances in the training set above. These predictions, and the related calculations required for calculating error and *errorDelta* values are shown in the following table.

ID	$\mathbb{M}_w(\mathbf{d}_i)$	t_i	Squared		<i>errorDelta</i> ($\mathcal{D}, w[j]$)								
			Error	Error	$w[0]$	$w[1]$	$w[2]$	$w[3]$	$w[4]$	$w[5]$	$w[6]$	$w[7]$	$w[8]$
0	0.051	0	-0.051	?	-0.0025	-0.0765	-0.0074	-0.0148	-0.0049	-0.0173	-0.0123	-0.0148	-0.0099
1	?	1	0.003	0.0000	-0.0025	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001
2	0.019	0	-0.019	0.0004	-0.0025	-0.0110	-0.0011	-0.0014	-0.0004	-0.0028	-0.0025	-0.0011	?
3	0.993	1	0.007	0.0000	?	0.0018	0.0001	0.0001	0.0000	0.0000	0.0000	0.0002	0.0007
4	?	0	-0.489	0.2391	-0.0025	-3.7879	-0.6110	-0.3666	-0.2444	-0.6110	-0.2444	-0.6110	-1.3441
5	0.945	1	?	0.0030	-0.0025	0.0915	0.0172	0.0086	0.0057	0.0029	0.0029	0.0143	0.0400
6	?	0	-0.400	0.1600	-0.0025	-2.9760	-0.2880	-0.4800	-0.1920	-0.2880	-0.5760	-0.1920	-1.1520
7	0.703	0	?	0.4942	-0.0025	-4.5502	?	-0.4403	-0.5871	-0.1468	-0.7339	-0.7339	-1.6146
8	0.980	1	0.020	?	-0.0025	0.0149	0.0016	0.0008	0.0008	0.0008	0.0016	0.0016	0.0031
9	0.986	1	0.014	0.0002	-0.0025	0.0073	0.0006	0.0004	0.0006	0.0008	0.0008	0.0002	0.0017

- (a) Some of the model predictions are missing in the preceding table (marked with a ?). Calculate these.

We can calculate the missing predictions as

$$\mathbf{1}: \text{Logistic}(0.309 + 0.100 \times 37 + -0.152 \times 3 + -0.163 \times 1 + 0.191 \times 4 + -0.631 \times 1 + -0.716 \times 3 + -0.478 \times 2 + -0.171 \times 13)$$

$$= \text{Logistic}(5.720) = \frac{1}{1 + e^{-5.720}}$$

$$= 0.997 \Rightarrow 1$$

$$\mathbf{4:} \text{ Logistic}(0.309 + 0.100 \times 31 + -0.152 \times 5 + -0.163 \times 3 +$$

$$0.191 \times 2 + -0.631 \times 5 + -0.716 \times 2 + -0.478 \times 5 +$$

$$-0.171 \times 11)$$

$$= \text{Logistic}(-0.046) = \frac{1}{1 + e^{0.046}}$$

$$= 0.489 \Rightarrow 0$$

$$\mathbf{6:} \text{ Logistic}(0.309 + 0.100 \times 31 + -0.152 \times 3 + -0.163 \times 5 +$$

$$0.191 \times 2 + -0.631 \times 3 + -0.716 \times 6 + -0.478 \times 2 +$$

$$-0.171 \times 12)$$

$$= \text{Logistic}(-0.407) = \frac{1}{1 + e^{0.407}}$$

$$= 0.400 \Rightarrow 0$$

- (b) Some of the Error and Squared Error values are missing in the preceding table (marked with a ?). Calculate these.

Errors as simple calculated as: $t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)$. So we have:

$$\mathbf{5:} \quad 1 - 0.945 = 0.055$$

$$\mathbf{7:} \quad 0 - 0.703 = -0.703$$

Squared errors are simply the square of error, so:

$$\mathbf{0:} \quad -0.051^2 = 0.0026$$

$$\mathbf{8:} \quad 0.020^2 = 0.0004$$

- (c) Some of the *errorDelta* values are missing in the preceding table (marked with a ?). Calculate these.

We calculate the *errorDelta* values using Equation (7.32)^[345].

$$\text{errorDelta}(\mathbf{d}_i, \mathbf{w}[j]) =$$

$$(t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)) \times \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i) \times (1 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)) \times \mathbf{d}_i[j]$$

The missing values are calculated as follows (note that because $\mathbf{w}[0]$ refers to the intercept weight associated with a dummy input value indexing into

features starts at 1):

$$\begin{aligned}
 errorDelta(\mathbf{d}_2, \mathbf{w}[8]) &= (t_2 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_2)) \times \mathbb{M}_{\mathbf{w}}(\mathbf{d}_2) \times (1 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_2)) \times \mathbf{d}_2[8] \\
 &= (0 - 0.019) \times 0.019 \times (1 - 0.019) \times 7 \\
 &= -0.0025
 \end{aligned}$$

(This example is for the intercept so we use a dummy input value of 1.)

$$\begin{aligned}
 errorDelta(\mathbf{d}_3, \mathbf{w}[0]) &= (t_3 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_3)) \times \mathbb{M}_{\mathbf{w}}(\mathbf{d}_3) \times (1 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_3)) \times 1 \\
 &= (1 - 0.993) \times 0.993 \times (1 - 0.993) \times 1 \\
 &= -0.0025
 \end{aligned}$$

$$\begin{aligned}
 errorDelta(\mathbf{d}_7, \mathbf{w}[2]) &= (t_7 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_7)) \times \mathbb{M}_{\mathbf{w}}(\mathbf{d}_7) \times (1 - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_7)) \times \mathbf{d}_7[2] \\
 &= (0 - 0.703) \times 0.703 \times (1 - 0.703) \times 4 \\
 &= -0.5871
 \end{aligned}$$

- (d) Calculate a new set of weights for this model using a **learning rate** of 0.01.

The following table summarises the answers from the previous parts and includes the sums of error and *errorDelta* values.

ID	$\mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)$	t_i	Squared		$errorDelta(\mathcal{D}, \mathbf{w}[j])$									
			Error	Error	$\mathbf{w}[0]$	$\mathbf{w}[1]$	$\mathbf{w}[2]$	$\mathbf{w}[3]$	$\mathbf{w}[4]$	$\mathbf{w}[5]$	$\mathbf{w}[6]$	$\mathbf{w}[7]$	$\mathbf{w}[8]$	
0	0.051	0	-0.051	0.0026	-0.0025	-0.0765	-0.0074	-0.0148	-0.0049	-0.0173	-0.0123	-0.0148	-0.0099	
1	0.997	1	0.003	0.0000	-0.0025	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	
2	0.019	0	-0.019	0.0004	-0.0025	-0.0110	-0.0011	-0.0014	-0.0004	-0.0028	-0.0025	-0.0011	-0.0025	
3	0.993	1	0.007	0.0000	-0.0025	0.0018	0.0001	0.0001	0.0000	0.0000	0.0000	0.0002	0.0007	
4	0.489	0	-0.489	0.2391	-0.0025	-3.7879	-0.6110	-0.3666	-0.2444	-0.6110	-0.2444	-0.6110	-1.3441	
5	0.945	1	0.055	0.0030	-0.0025	0.0915	0.0172	0.0086	0.0057	0.0029	0.0029	0.0143	0.0400	
6	0.400	0	-0.400	0.1600	-0.0025	-2.9760	-0.2880	-0.4800	-0.1920	-0.2880	-0.5760	-0.1920	-1.1520	
7	0.703	0	-0.703	0.4942	-0.0025	-4.5502	-0.5871	-0.4403	-0.5871	-0.1468	-0.7339	-0.7339	-1.6146	
8	0.980	1	0.020	0.0004	-0.0025	0.0149	0.0016	0.0008	0.0008	0.0008	0.0016	0.0016	0.0031	
9	0.986	1	0.014	0.0002	-0.0025	0.0073	0.0006	0.0004	0.0006	0.0008	0.0008	0.0002	0.0017	
Sum			-1.563	0.8999	-0.0250	-11.2858	-1.4751	-1.2932	-1.0217	-1.0614	-1.5638	-1.5365	-4.0775	

To calculate the new weight values we again use Equation (7.32)^[345]:

$$\mathbf{w}[j] \leftarrow \mathbf{w}[j] + \alpha \times errorDelta(\mathcal{D}, \mathbf{w}[j])$$

So the new weight values are:

$$\begin{aligned}\mathbf{w}[0] &\leftarrow \mathbf{w}[0] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[0]) \\ &\leftarrow 0.309 + 0.01 \times -0.025 \\ &\leftarrow 0.100\end{aligned}$$

$$\begin{aligned}\mathbf{w}[1] &\leftarrow \mathbf{w}[1] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[1]) \\ &\leftarrow 0.100 + 0.01 \times -11.2858 \\ &\leftarrow 0.298\end{aligned}$$

$$\begin{aligned}\mathbf{w}[2] &\leftarrow \mathbf{w}[2] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[2]) \\ &\leftarrow -0.152 + 0.01 \times -1.4751 \\ &\leftarrow -0.153\end{aligned}$$

$$\begin{aligned}\mathbf{w}[3] &\leftarrow \mathbf{w}[3] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[3]) \\ &\leftarrow -0.163 + 0.01 \times -1.2932 \\ &\leftarrow -0.164\end{aligned}$$

$$\begin{aligned}\mathbf{w}[4] &\leftarrow \mathbf{w}[4] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[4]) \\ &\leftarrow 0.191 + 0.01 \times -1.0217 \\ &\leftarrow 0.190\end{aligned}$$



$$\begin{aligned}\mathbf{w}[5] &\leftarrow \mathbf{w}[5] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[5]) \\ &\leftarrow -0.631 + 0.01 \times -1.0614 \\ &\leftarrow -0.632\end{aligned}$$

$$\begin{aligned}\mathbf{w}[6] &\leftarrow \mathbf{w}[6] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[6]) \\ &\leftarrow -0.716 + 0.01 \times -1.5638 \\ &\leftarrow -0.718\end{aligned}$$

$$\begin{aligned}\mathbf{w}[7] &\leftarrow \mathbf{w}[7] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[7]) \\ &\leftarrow -0.478 + 0.01 \times -1.5365 \\ &\leftarrow -0.480\end{aligned}$$

$$\begin{aligned}\mathbf{w}[8] &\leftarrow \mathbf{w}[8] + \alpha \times \text{errorDelta}(\mathcal{D}, \mathbf{w}[8]) \\ &\leftarrow -0.171 + 0.01 \times -4.0775 \\ &\leftarrow -0.175\end{aligned}$$

- (e) The following table shows handwritten examples of the digits 7 and 8 and their corresponding histogram values.

	GL-0	GL-1	GL-2	GL-3	GL-4	GL-5	GL-6	GL-7
	35	1	5	4	5	2	4	8
	30	6	2	0	5	4	4	13

- i. Calculate the output of the model (using the updated weights calculated in the previous part) for these two instances.

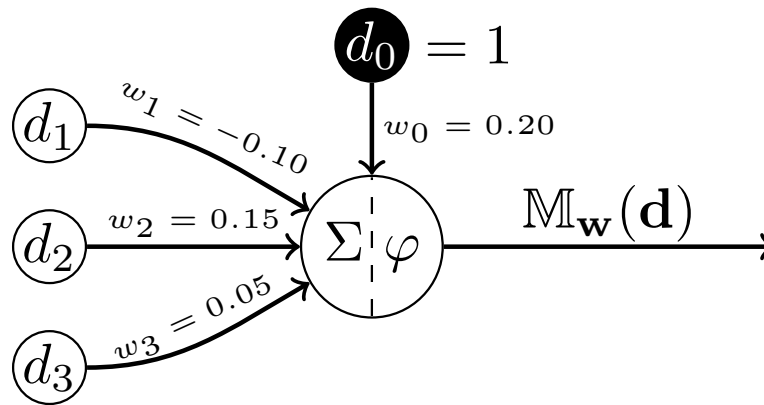
$$\begin{aligned}
 \mathbf{7:} \quad & \text{Logistic}(0.100 + 0.298 \times 35 + -0.153 \times 1 + -0.164 \times 5 + \\
 & \quad 0.190 \times 4 + -0.632 \times 5 + -0.718 \times 2 + -0.480 \times 4 + \\
 & \quad -0.175 \times 8) \\
 & = \text{Logistic}(2.401) = \frac{1}{1 + e^{-2.401}} \\
 & = 0.917 \Rightarrow 1 \\
 \mathbf{8:} \quad & \text{Logistic}(0.100 + 0.298 \times 30 + -0.153 \times 6 + -0.164 \times 2 + \\
 & \quad 0.190 \times 0 + -0.632 \times 5 + -0.718 \times 4 + -0.480 \times 4 + \\
 & \quad -0.175 \times 13) \\
 & = \text{Logistic}(-2.433) = \frac{1}{1 + e^{2.433}} \\
 & = 0.081 \Rightarrow 0
 \end{aligned}$$

- ii. Comment on the appropriateness of these outputs.

The model outputs a 1 for the image of a 7 and a 0 for the image of an 8. While this is not entirely unreasonable—a 7 looks more like a 1 than a 0 and an 8 looks more like a 0 than a 1—it is not ideal behaviour. This illustrates one drawback of classification models. They are essentially discriminators and struggle to recognise that an instance is completely new. In this case we would probably prefer an answer of *none of the above*. Building models that can accurately do this is an open research area and relates to *model calibration* and *anomaly detection*.

8 Deep Learning (Exercise Solutions)

1. The following image shows an artificial neuron that takes three inputs



- (a) Calculate the **weighted sum** for this neuron for the input vector

$$\mathbf{d} = [0.2, 0.5, 0.7]$$

$$\begin{aligned} z &= (1 \times w_0) + (0.2 \times w_1) + (0.5 \times w_2) + (0.7 \times w_3) \\ &= (1 \times 0.2) + (0.2 \times -0.1) + (0.5 \times 0.15) + (0.7 \times 0.05) \\ &= 0.2 + -0.02 + 0.075 + 0.035 \\ &= 0.29 \end{aligned}$$

- (b) What would be the output from this neuron if the activation function φ is a threshold activation with $\theta = 1$?

Using a threshold activation function the weighted sum z is mapped to an output as follows:

$$\mathbb{M}_{\mathbf{w}}(\mathbf{d}) = \begin{cases} 1 & \text{if } z \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

In this instance $z = 0.29$ and $\theta = 1$ and so $\mathbb{M}_{\mathbf{w}}(\mathbf{d}) = 0$

- (c) What would be the output from this neuron if the activation function φ is the logistic function?

The logistic function is defined as:

$$\text{logistic}(z) = \frac{1}{1 + e^{-z}}$$

For $z = 0.29$ the result of this calculation is: 0.571996133

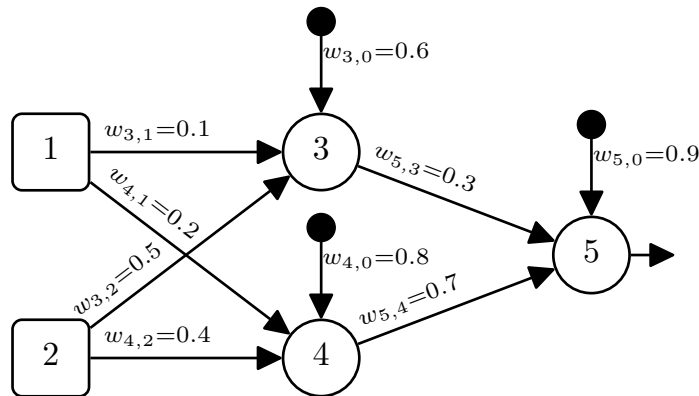
- (d) What would be the output from this neuron if the activation function φ is the rectified linear function?

The rectified linear function is defined as:

$$\text{rectifier}(z) = \max(0, z)$$

For $z = 0.29$ the output of the rectified linear function is 0.29.

2. The following image shows an artificial neural network with two sensing neurons (Neurons 1 and 2) and 3 processing neurons (Neurons 3, 4, and 5)



- (a) Assuming that the processing neurons in this network use a **logistic** activation function, what would be the output of Neuron 5 if the network received the input vector: Neuron 1 = 0.7 and Neuron 2 = 0.3?

The weighted sum calculation for Neuron 3 is:

$$\begin{aligned} z_3 &= (0.6 \times 1) + (0.1 \times 0.7) + (0.5 \times 0.3) \\ &= (0.6) + (0.07) + (0.15) \\ &= 0.82 \end{aligned}$$

The weighted sum calculation for Neuron 4 is:

$$\begin{aligned} z_4 &= (0.8 \times 1) + (0.2 \times 0.7) + (0.4 \times 0.3) \\ &= (0.8) + (0.14) + (0.12) \\ &= 1.06 \end{aligned}$$

Using a logistic activation function the activation for Neuron 3 is:

$$\begin{aligned} a_3 &= \text{logistic}(z_3) \\ &= \frac{1}{1 + e^{-0.82}} \\ &= 0.69423634 \end{aligned}$$

Using a logistic activation function the activation for Neuron 4 is:

$$\begin{aligned} a_4 &= \text{logistic}(z_4) \\ &= \frac{1}{1 + e^{-1.06}} \\ &= 0.742690545 \end{aligned}$$

The weighted sum calculation for Neuron 5 can now be calculated as:

$$\begin{aligned} z_5 &= (0.9 \times 1) + (0.3 \times 0.69423634) + (0.7 \times 0.742690545) \\ &= (0.9) + (0.208270902) + (0.519883382) \\ &= 1.628154284 \end{aligned}$$

And the final output of the network can now be calculated by passing z_5 through the activation function for Neuron 5, which for this question is the

logistic function:

$$\begin{aligned} a_5 &= \text{logistic}(z_5) \\ &= \frac{1}{1 + e^{-1.628154284}} \\ &= 0.835916637 \end{aligned}$$

- (b) Assuming that the processing neurons in this network use a **ReLU** activation function, what would be the output of Neuron 5 if the network received the input vector: Neuron 1 = 0.7 and Neuron 2 = 0.3?

From the solutions to part one of this question the weighted sum calculations for Neurons 3 and 4 are as follows:

$$z_3 = 0.82$$

$$z_4 = 1.06$$

Both of these values are greater than 0 and so using the rectifier function ($\max(0, z)$) as an activation function the activations for each of these neurons is:

$$a_3 = 0.82$$

$$a_4 = 1.06$$

The weighted sum calculation for Neuron 5 can now be calculated as:

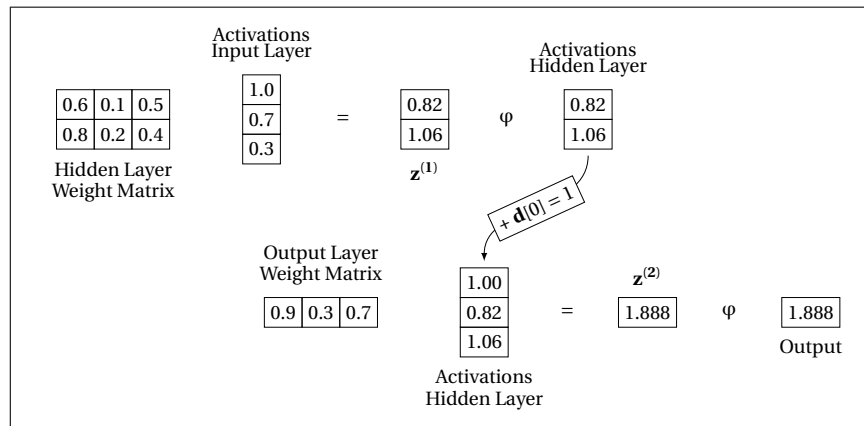
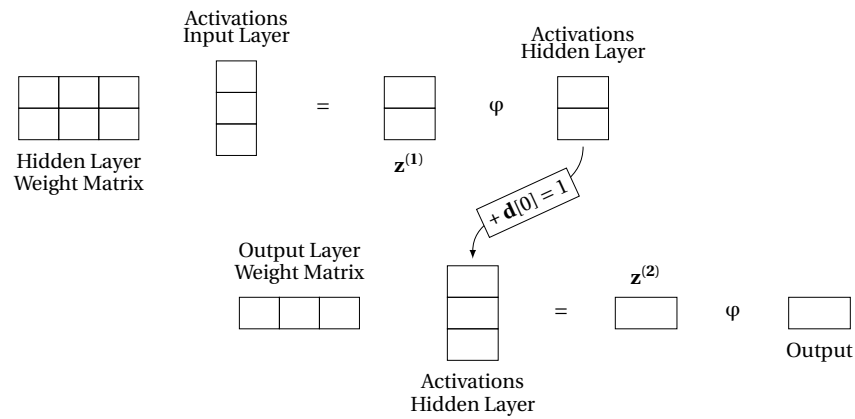
$$\begin{aligned} z_5 &= (0.9 \times 1) + (0.3 \times 0.82) + (0.7 \times 1.06) \\ &= (0.9) + (0.246) + (0.742) \\ &= 1.888 \end{aligned}$$

z_5 is greater than 0 and so passing this value through the rectifier activation function ($\max(0, z)$) does not change it, and so the activation for Neuron 5 (and the output of the network) is:

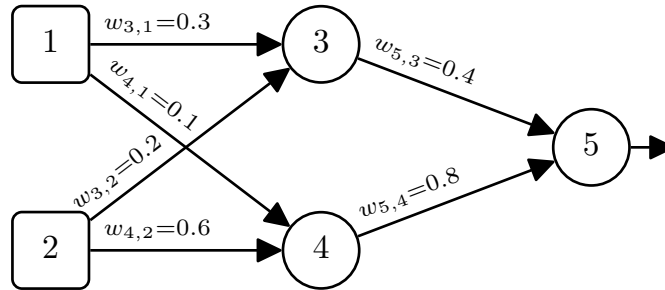
$$a_5 = 1.888$$

- (c) The following image provides a template diagram for the sequence of matrix operations that our neural network would use to process the input vector Neuron 1 =

0.7 and Neuron 2 = 0.3. Assuming that the processing neurons in the network use a ReLU activation function, fill in the diagram with the appropriate weights, bias terms, weighted sum values, and activations.



3. The following image shows an artificial network with two layers of **linear neurons** (i.e., neurons that have no activation function and whose output is simply the result of the weighted sum of their inputs). Furthermore, these neurons have no bias terms.



- (a) Calculate the output for Neuron 5 for the input vector: Neuron 1 = 0.9, Neuron 2 = 0.5.

Weighted sum for Neuron 3:

$$\begin{aligned} a_3 &= ((0.9 \times 0.3) + (0.5 \times 0.2)) \\ &= 0.37 \end{aligned}$$

Weighted sum for Neuron 4:

$$\begin{aligned} a_4 &= ((0.9 \times 0.1) + (0.5 \times 0.6)) \\ &= 0.39 \end{aligned}$$

Weighted sum, and activation, for Neuron 5:

$$\begin{aligned} a_5 &= ((0.37 \times 0.4) + (0.39 \times 0.8)) \\ &= 0.46 \end{aligned}$$

- (b) Calculate the weight matrix for the single layer network that would implement the equivalent mapping that this two-layer network implements.

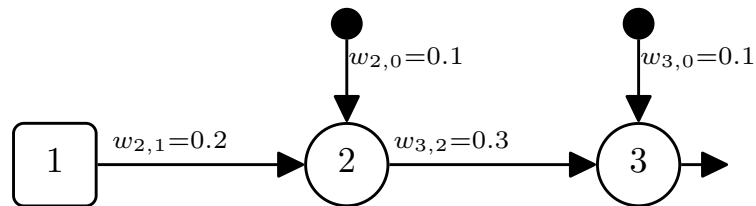
$$\begin{aligned} \mathbf{W}' &= \mathbf{W}_2 \cdot \mathbf{W}_1 \\ &= \begin{bmatrix} 0.4 & 0.8 \end{bmatrix} \cdot \begin{bmatrix} 0.3 & 0.2 \\ 0.1 & 0.6 \end{bmatrix} \\ &= \begin{bmatrix} 0.2 & 0.56 \end{bmatrix} \end{aligned}$$

- (c) Show that the single-layer network using the weight matrix you calculated in Part 2 generates the same output as the network for the input vector: Neuron 1 = 0.9,

Neuron 2 = 0.5.

$$\begin{aligned}
 \text{activation} &= \begin{bmatrix} 0.2 & 0.56 \end{bmatrix} \cdot \begin{bmatrix} 0.9 \\ 0.5 \end{bmatrix} \\
 &= ((0.2 \times 0.9) + (0.56 \times 0.5)) \\
 &= 0.46
 \end{aligned}$$

4. The following image illustrates the topology of a simple feedforward neural network that has a single sensing neuron (Neuron 1), a single hidden processing neuron (Neuron 2), and a single processing output neuron (Neuron 3).



- (a) Assuming that the processing neurons use **logistic** activation functions, that the input to the network is Neuron 1 = 0.2 and that the desired output for this input is 0.7:
- Calculate the output generated by the network in response to this input.

$$\begin{aligned}
 z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.2)) \\
 &= ((0.1 \times 1) + (0.2 \times 0.2)) \\
 &= 0.14
 \end{aligned}$$

$$\begin{aligned}
 a_2 &= \text{logistic}(0.14) \\
 &= 0.534942945
 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.1 \times 1) + (0.3 \times 0.534942945)) \\
 &= 0.260482884
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{logistic}(0.260482884) \\
 &= 0.564754992
 \end{aligned}$$

- ii. Calculate the δ values for each of the neurons in the network (i.e., δ_3 , δ_2).

Calculations for δ_3 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_3} &= -(t_k - a_k) \\ &= -(0.7 - 0.564754992) \\ &= -0.135245008\end{aligned}$$

$$\begin{aligned}\frac{\partial a_3}{\partial z_3} &= \text{logistic}(z_3) \times (1 - \text{logistic}(z_3)) \\ &= 0.564754992 \times (1 - 0.564754992) \\ &= 0.245806791\end{aligned}$$

$$\begin{aligned}\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\ &= -0.135245008 \times 0.245806791 \\ &= -0.033244142\end{aligned}$$

Calculations for δ_2 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_2} &= w_{3,2} \times \delta_3 \\ &= 0.3 \times -0.033244142 \\ &= -0.009973242\end{aligned}$$

$$\begin{aligned}\frac{\partial a_2}{\partial z_2} &= \text{logistic}(z_2) \times (1 - \text{logistic}(z_2)) \\ &= 0.534942945 \times (1 - 0.534942945) \\ &= 0.248778991\end{aligned}$$

$$\begin{aligned}\delta_2 &= \frac{\partial \mathcal{E}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \\ &= -0.009973242 \times 0.248778991 \\ &= -0.002481133\end{aligned}$$

- iii. Using the δ values you calculated above, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial \mathcal{E} / \partial w_{3,2}$,

$$\partial\mathcal{E}/\partial w_{3,0}, \partial\mathcal{E}/\partial w_{2,1}, \partial\mathcal{E}/\partial w_{2,0}).$$

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\ &= -0.033244142 \times 0.534942945 \\ &= -0.017783719\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\ &= -0.033244142 \times 1 \\ &= -0.033244142\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{2,1}} &= \delta_2 \times a_1 \\ &= -0.002481133 \times 0.2 \\ &= -0.000496227\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{2,0}} &= \delta_2 \times a_0 \\ &= -0.002481133 \times 1 \\ &= -0.002481133\end{aligned}$$

- iv. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{3,2}, w_{3,0}, w_{2,1}, w_{2,0}$) after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial\mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned} w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\ &= 0.3 - 0.1 \times -0.017783719 \\ &= 0.301778372 \end{aligned}$$

$$\begin{aligned} w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\ &= 0.1 - 0.1 \times -0.033244142 \\ &= 0.103324414 \end{aligned}$$

$$\begin{aligned} w_{2,1}^{t+1} &= w_{2,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,1}} \\ &= 0.2 - 0.1 \times -0.000496227 \\ &= 0.200049623 \end{aligned}$$

$$\begin{aligned} w_{2,0}^{t+1} &= w_{2,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,0}} \\ &= 0.1 - 0.1 \times -0.002481133 \\ &= 0.100248113 \end{aligned}$$

- v. Calculate the reduction in the error of the network for this example using the new weights, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned} z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.2)) \\ &= ((0.100248113 \times 1) + (0.200049623 \times 0.2)) \\ &= 0.140258038 \end{aligned}$$

$$\begin{aligned} a_2 &= \text{logistic}(0.140258038) \\ &= 0.535007139 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.103324414 \times 1) + (0.301778372 \times 0.535007139)) \\
 &= 0.264777998
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{logistic}(0.264777998) \\
 &= 0.565810465
 \end{aligned}$$

The error with the new weights is:

$$\begin{aligned}
 \text{Error}' &= 0.7 - 0.565810465 \\
 &= 0.134189535
 \end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned}
 \text{Error Reduction} &= 0.135245008 - 0.134189535 \\
 &= 0.001055473
 \end{aligned}$$

- (b) Assuming that the processing neurons are **ReLU**s, that the input to the network is Neuron 1 = 0.2, and that the desired output for this input is 0.7:
- Calculate the output generated by the network in response to this input.

$$\begin{aligned}
 z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.2)) \\
 &= ((0.1 \times 1) + (0.2 \times 0.2)) \\
 &= 0.14
 \end{aligned}$$

$$\begin{aligned}
 a_2 &= \text{rectifier}(0.14) \\
 &= \max(0, 0.14) \\
 &= 0.14
 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.1 \times 1) + (0.3 \times 0.14)) \\
 &= 0.142
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{rectifier}(0.142) \\
 &= \max(0, 0.142) \\
 &= 0.142
 \end{aligned}$$

- ii. Calculate the δ values for each of the neurons in the network (i.e., δ_3, δ_2).

Calculations for δ_3 :

$$\begin{aligned}
 \frac{\partial \mathcal{E}}{\partial a_3} &= -(t_k - a_k) \\
 &= -(0.7 - 0.142) \\
 &= -0.558
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial a_3}{\partial z_3} &= \begin{cases} 1 & \text{if } z_3 > 1 \\ 0 & \text{otherwise} \end{cases} \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\
 &= -0.558 \times 1 \\
 &= -0.558
 \end{aligned}$$

Calculations for δ_2 :

$$\begin{aligned}
 \frac{\partial \mathcal{E}}{\partial a_2} &= w_{3,2} \times \delta_3 \\
 &= 0.3 \times -0.558 \\
 &= -0.1674
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial a_2}{\partial z_2} &= \begin{cases} 1 & \text{if } z_2 > 1 \\ 0 & \text{otherwise} \end{cases} \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \delta_2 &= \frac{\partial \mathcal{E}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \\
 &= -0.1674 \times 1 \\
 &= -0.1674
 \end{aligned}$$

- iii. Using the δ values you have calculated in the preceding, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial\mathcal{E}/\partial w_{3,2}$, $\partial\mathcal{E}/\partial w_{3,0}$, $\partial\mathcal{E}/\partial w_{2,1}$, $\partial\mathcal{E}/\partial w_{2,0}$).

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\ &= -0.558 \times 0.14 \\ &= -0.07812\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\ &= -0.558 \times 1 \\ &= -0.558\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{2,1}} &= \delta_2 \times a_1 \\ &= -0.1674 \times 0.2 \\ &= -0.03348\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{2,0}} &= \delta_2 \times a_0 \\ &= -0.1674 \times 1 \\ &= -0.1674\end{aligned}$$

- iv. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{3,2}$, $w_{3,0}$, $w_{2,1}$, $w_{2,0}$) after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial\mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned} w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\ &= 0.3 - 0.1 \times -0.07812 \\ &= 0.307812 \end{aligned}$$

$$\begin{aligned} w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\ &= 0.1 - 0.1 \times -0.558 \\ &= 0.1558 \end{aligned}$$

$$\begin{aligned} w_{2,1}^{t+1} &= w_{2,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,1}} \\ &= 0.2 - 0.1 \times -0.03348 \\ &= 0.203348 \end{aligned}$$

$$\begin{aligned} w_{2,0}^{t+1} &= w_{2,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,0}} \\ &= 0.1 - 0.1 \times -0.1674 \\ &= 0.11674 \end{aligned}$$

- v. Calculate the reduction in the error for this example using the new weights for the network, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned} z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.2)) \\ &= ((0.11674 \times 1) + (0.203348 \times 0.2)) \\ &= 0.1574096 \end{aligned}$$

$$\begin{aligned} a_2 &= \text{rectifier}(0.1574096) \\ &= \max(0, 0.1574096) \\ &= 0.1574096 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.1558 \times 1) + (0.307812 \times 0.1574096)) \\
 &= 0.204252564
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{rectifier}(0.204252564) \\
 &= \max(0, 0.204252564) \\
 &= 0.204252564
 \end{aligned}$$

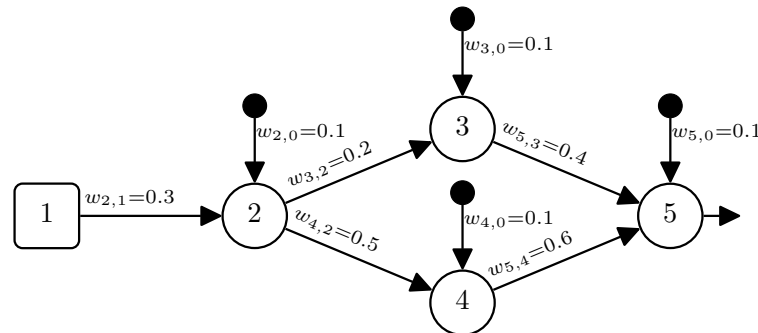
The error with the new weights is:

$$\begin{aligned}
 \text{Error}' &= 0.7 - 0.204252564 \\
 &= 0.495747436
 \end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned}
 \text{Error Reduction} &= 0.558 - 0.495747436 \\
 &= 0.062252564
 \end{aligned}$$

5. The following image illustrates the topology of a simple feedforward neural network that has a single sensing neuron (Neuron 1), three hidden processing neuron (Neurons 2, 3, and 4), and a single processing output neuron (Neuron 5).



- (a) Assuming that the processing neurons use **logistic** activation functions, that the input to the network is Neuron 1 = 0.5 and that the desired output for this input is 0.9:
- Calculate the output generated by the network in response to this input.

$$\begin{aligned}
 z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.5)) \\
 &= ((0.1 \times 1) + (0.3 \times 0.5)) \\
 &= 0.25
 \end{aligned}$$

$$\begin{aligned}
 a_2 &= \text{logistic}(0.25) \\
 &= 0.562176501
 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.1 \times 1) + (0.2 \times 0.562176501)) \\
 &= 0.2124353
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{logistic}(0.2124353) \\
 &= 0.552909994
 \end{aligned}$$

$$\begin{aligned}
 z_4 &= ((w_{4,0} \times 1) + (w_{4,2} \times a_2)) \\
 &= ((0.1 \times 1) + (0.5 \times 0.562176501)) \\
 &= 0.38108825
 \end{aligned}$$

$$\begin{aligned}
 a_4 &= \text{logistic}(0.38108825) \\
 &= 0.594135549
 \end{aligned}$$

$$\begin{aligned}
 z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\
 &= ((0.1 \times 1) + (0.4 \times 0.552909994) + (0.6 \times 0.594135549)) \\
 &= 0.677645327
 \end{aligned}$$

$$\begin{aligned}
 a_5 &= \text{logistic}(0.677645327) \\
 &= 0.663212956
 \end{aligned}$$

- ii. Calculate the δ values for each of the processing neurons in the network (i.e., $\delta_5, \delta_4, \delta_3, \delta_2$).

Calculations for δ_5 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_5} &= -(t_k - a_k) \\ &= -(0.9 - 0.663212956) \\ &= -0.236787044\end{aligned}$$

$$\begin{aligned}\frac{\partial a_5}{\partial z_5} &= \text{logistic}(z_5) \times (1 - \text{logistic}(z_5)) \\ &= 0.677645327 \times (1 - 0.677645327) \\ &= 0.223361531\end{aligned}$$

$$\begin{aligned}\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_5} \times \frac{\partial a_5}{\partial z_5} \\ &= -0.236787044 \times 0.223361531 \\ &= -0.052889117\end{aligned}$$

Calculations for δ_4 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_4} &= w_{5,4} \times \delta_5 \\ &= 0.6 \times -0.052889117 \\ &= -0.03173347\end{aligned}$$

$$\begin{aligned}\frac{\partial a_4}{\partial z_4} &= \text{logistic}(z_4) \times (1 - \text{logistic}(z_4)) \\ &= 0.594135549 \times (1 - 0.594135549) \\ &= 0.241138498\end{aligned}$$

$$\begin{aligned}\delta_4 &= \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} \\ &= -0.03173347 \times 0.241138498 \\ &= -0.007652161\end{aligned}$$

Calculations for δ_3 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_3} &= w_{5,3} \times \delta_5 \\ &= 0.4 \times -0.052889117 \\ &= -0.021155647\end{aligned}$$

$$\begin{aligned}\frac{\partial a_3}{\partial z_3} &= \text{logistic}(z_3) \times (1 - \text{logistic}(z_3)) \\ &= 0.552909994 \times (1 - 0.552909994) \\ &= 0.247200532\end{aligned}$$

$$\begin{aligned}\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\ &= -0.021155647 \times 0.247200532 \\ &= -0.005229687\end{aligned}$$

Calculations for δ_2 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_2} &= (w_{3,2} \times \delta_3) + (w_{4,2} \times \delta_4) \\ &= (0.2 \times -0.005229687) + (0.5 \times -0.007652161) \\ &= -0.004872018\end{aligned}$$

$$\begin{aligned}\frac{\partial a_2}{\partial z_2} &= \text{logistic}(z_2) \times (1 - \text{logistic}(z_2)) \\ &= 0.562176501 \times (1 - 0.562176501) \\ &= 0.246134083\end{aligned}$$

$$\begin{aligned}\delta_2 &= \frac{\partial \mathcal{E}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \\ &= -0.004872018 \times 0.246134083 \\ &= -0.00119917\end{aligned}$$

- iii. Using the δ values you have calculated, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial \mathcal{E} / \partial w_{5,4}$, $\partial \mathcal{E} / \partial w_{5,3}$, $\partial \mathcal{E} / \partial w_{5,0}$, $\partial \mathcal{E} / \partial w_{4,2}$, $\partial \mathcal{E} / \partial w_{4,0}$, $\partial \mathcal{E} / \partial w_{3,2}$, $\partial \mathcal{E} / \partial w_{3,0}$, $\partial \mathcal{E} / \partial w_{2,1}$,

$\partial\mathcal{E}/\partial w_{2,0}$).

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{5,4}} &= \delta_5 \times a_4 \\ &= -0.052889117 \times 0.594135549 \\ &= -0.031423304\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{5,3}} &= \delta_5 \times a_3 \\ &= -0.052889117 \times 0.552909994 \\ &= -0.029242921\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{5,0}} &= \delta_5 \times a_0 \\ &= -0.052889117 \times 1 \\ &= -0.052889117\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{4,2}} &= \delta_4 \times a_2 \\ &= -0.007652161 \times 0.562176501 \\ &= -0.004301865\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{4,0}} &= \delta_4 \times a_0 \\ &= -0.007652161 \times 1 \\ &= -0.007652161\end{aligned}$$

$$\begin{aligned}\frac{\partial\mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\ &= -0.005229687 \times 0.562176501 \\ &= -0.002940007\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\
&= -0.005229687 \times 1 \\
&= -0.005229687
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{2,1}} &= \delta_2 \times a_1 \\
&= -0.00119917 \times 0.5 \\
&= -0.000599585
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{2,0}} &= \delta_2 \times a_0 \\
&= -0.00119917 \times 1 \\
&= -0.00119917
\end{aligned}$$

- iv. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{5,4}$, $w_{5,3}$, $w_{5,0}$, $w_{4,2}$, $w_{4,0}$, $w_{3,2}$, $w_{3,0}$, $w_{2,1}$, $w_{2,0}$) after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned}
w_{5,4}^{t+1} &= w_{5,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,4}} \\
&= 0.6 - (0.1 \times -0.031423304) \\
&= 0.60314233
\end{aligned}$$

$$\begin{aligned}
w_{5,3}^{t+1} &= w_{5,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,3}} \\
&= 0.4 - (0.1 \times -0.029242921) \\
&= 0.402924292
\end{aligned}$$

$$\begin{aligned}
w_{5,0}^{t+1} &= w_{5,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,0}} \\
&= 0.1 - (0.1 \times -0.052889117) \\
&= 0.105288912
\end{aligned}$$

$$\begin{aligned}
w_{4,2}^{t+1} &= w_{4,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,2}} \\
&= 0.5 - (0.1 \times -0.004301865) \\
&= 0.500430187
\end{aligned}$$

$$\begin{aligned}
w_{4,0}^{t+1} &= w_{4,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,0}} \\
&= 0.1 - (0.1 \times -0.007652161) \\
&= 0.100765216
\end{aligned}$$

$$\begin{aligned}
w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\
&= 0.2 - (0.1 \times -0.002940007) \\
&= 0.200294001
\end{aligned}$$

$$\begin{aligned}
w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\
&= 0.1 - (0.1 \times -0.005229687) \\
&= 0.100522969
\end{aligned}$$

$$\begin{aligned}
w_{2,1}^{t+1} &= w_{2,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,1}} \\
&= 0.3 - (0.1 \times -0.000599585) \\
&= 0.300059958
\end{aligned}$$

$$\begin{aligned}
w_{2,0}^{t+1} &= w_{2,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,0}} \\
&= 0.1 - (0.1 \times -0.00119917) \\
&= 0.100119917
\end{aligned}$$

- v. Calculate the reduction in the error of the network for this example using the new weights, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned} z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.5)) \\ &= ((0.100119917 \times 1) + (0.300059958 \times 0.5)) \\ &= 0.250149896 \end{aligned}$$

$$\begin{aligned} a_2 &= \text{logistic}(0.250149896) \\ &= 0.562213395 \end{aligned}$$

$$\begin{aligned} z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\ &= ((0.100522969 \times 1) + (0.200294001 \times 0.562213395)) \\ &= 0.213130939 \end{aligned}$$

$$\begin{aligned} a_3 &= \text{logistic}(0.213130939) \\ &= 0.55308195 \end{aligned}$$

$$\begin{aligned} z_4 &= ((w_{4,0} \times 1) + (w_{4,2} \times a_2)) \\ &= ((0.100765216 \times 1) + (0.500430187 \times 0.562213395)) \\ &= 0.38211377 \end{aligned}$$

$$\begin{aligned} a_4 &= \text{logistic}(0.38211377) \\ &= 0.594382817 \end{aligned}$$

$$\begin{aligned} z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\ &= ((0.105288912 \times 1) + (0.402924292 \times 0.55308195) + (0.60314233 \times 0.594382817)) \\ &= 0.686636503 \end{aligned}$$

$$\begin{aligned} a_5 &= \text{logistic}(0.686636503) \\ &= 0.665218283 \end{aligned}$$

The error with the new weights is:

$$\begin{aligned} Error' &= 0.9 - 0.665218283 \\ &\quad 0.234781717 \end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned} Error\ Reduction &= 0.236787044 - 0.234781717 \\ &= 0.002005326 \end{aligned}$$

- (b) Assuming that the processing neurons are **ReLU**s, that the input to the network is Neuron 1 = 0.5 and that the desired output for this input is 0.9
- i. Calculate the output generated by the network in response to this input.

$$\begin{aligned} z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.5)) \\ &= ((0.1 \times 1) + (0.3 \times 0.5)) \\ &= 0.25 \end{aligned}$$

$$\begin{aligned} a_2 &= \text{rectifier}(0.25) \\ &= \max(0, 0.25) \\ &= 0.25 \end{aligned}$$

$$\begin{aligned} z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\ &= ((0.1 \times 1) + (0.2 \times 0.25)) \\ &= 0.15 \end{aligned}$$

$$\begin{aligned} a_3 &= \text{rectifier}(0.15) \\ &= \max(0, 0.15) \\ &= 0.15 \end{aligned}$$

$$\begin{aligned} z_4 &= ((w_{4,0} \times 1) + (w_{4,2} \times a_2)) \\ &= ((0.1 \times 1) + (0.5 \times 0.25)) \\ &= 0.225 \end{aligned}$$

$$\begin{aligned}
 a_4 &= \text{rectifier}(0.225) \\
 &= \max(0, 0.225) \\
 &= 0.225
 \end{aligned}$$

$$\begin{aligned}
 z_5 &= (w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4) \\
 &= ((0.1 \times 1) + (0.4 \times 0.15) + (0.6 \times 0.225)) \\
 &= 0.295
 \end{aligned}$$

$$\begin{aligned}
 a_5 &= \text{rectifier}(0.295) \\
 &= \max(0, 0.295) \\
 &= 0.295
 \end{aligned}$$

- ii. Calculate the δ values for each of the processing neurons in the network (i.e., $\delta_5, \delta_4, \delta_3, \delta_2$).

Calculations for δ_5 :

$$\begin{aligned}
 \frac{\partial \mathcal{E}}{\partial a_5} &= -(t_k - a_k) \\
 &= -(0.9 - 0.295) \\
 &= -0.605
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial a_5}{\partial z_5} &= \begin{cases} 1 & \text{if } z_5 > 1 \\ 0 & \text{otherwise} \end{cases} \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \delta_5 &= \frac{\partial \mathcal{E}}{\partial a_5} \times \frac{\partial a_5}{\partial z_5} \\
 &= -0.605 \times 1 \\
 &= -0.605
 \end{aligned}$$

Calculations for δ_4 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_4} &= w_{5,4} \times \delta_5 \\ &= 0.6 \times -0.605 \\ &= -0.363\end{aligned}$$

$$\begin{aligned}\frac{\partial a_4}{\partial z_4} &= \begin{cases} 1 & \text{if } z_4 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 1\end{aligned}$$

$$\begin{aligned}\delta_4 &= \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} \\ &= -0.363 \times 1 \\ &= -0.363\end{aligned}$$

Calculations for δ_3 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_3} &= w_{5,3} \times \delta_5 \\ &= 0.4 \times -0.605 \\ &= -0.242\end{aligned}$$

$$\begin{aligned}\frac{\partial a_3}{\partial z_3} &= \begin{cases} 1 & \text{if } z_3 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 1\end{aligned}$$

$$\begin{aligned}\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\ &= -0.242 \times 1 \\ &= -0.242\end{aligned}$$

Calculations for δ_2 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_2} &= (w_{3,2} \times \delta_3) + (w_{4,2} \times \delta_4) \\ &= (0.2 \times -0.242) + (0.5 \times -0.363) \\ &= -0.2299\end{aligned}$$

$$\begin{aligned}\frac{\partial a_2}{\partial z_2} &= \begin{cases} 1 & \text{if } z_2 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 1\end{aligned}$$

$$\begin{aligned}\delta_2 &= \frac{\partial \mathcal{E}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \\ &= -0.2299 \times 1 \\ &= -0.2299\end{aligned}$$

- iii. Using the δ values you have calculated, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial \mathcal{E} / \partial w_{5,4}$, $\partial \mathcal{E} / \partial w_{5,3}$, $\partial \mathcal{E} / \partial w_{5,0}$, $\partial \mathcal{E} / \partial w_{4,2}$, $\partial \mathcal{E} / \partial w_{4,0}$, $\partial \mathcal{E} / \partial w_{3,2}$, $\partial \mathcal{E} / \partial w_{3,0}$, $\partial \mathcal{E} / \partial w_{2,1}$, $\partial \mathcal{E} / \partial w_{2,0}$).

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{5,4}} &= \delta_5 \times a_4 \\ &= -0.605 \times 0.225 \\ &= -0.136125\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{5,3}} &= \delta_5 \times a_3 \\ &= -0.605 \times 0.15 \\ &= -0.09075\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,0}} &= \delta_5 \times a_0 \\
&= -0.605 \times 1 \\
&= -0.605
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{4,2}} &= \delta_4 \times a_2 \\
&= -0.363 \times 0.25 \\
&= -0.09075
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{4,0}} &= \delta_4 \times a_0 \\
&= -0.363 \times 1 \\
&= -0.363
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\
&= -0.242 \times 0.25 \\
&= -0.0605
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\
&= -0.242 \times 1 \\
&= -0.242
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{2,1}} &= \delta_2 \times a_1 \\
&= -0.2299 \times 0.5 \\
&= -0.11495
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{2,0}} &= \delta_2 \times a_0 \\
&= -0.2299 \times 1 \\
&= -0.2299
\end{aligned}$$

- iv. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{5,4}$, $w_{5,3}$, $w_{5,0}$, $w_{4,2}$, $w_{4,0}$, $w_{3,2}$, $w_{3,0}$, $w_{2,1}$, $w_{2,0}$.)

after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned} w_{5,4}^{t+1} &= w_{5,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,4}} \\ &= 0.6 - (0.1 \times -0.136125) \\ &= 0.6136125 \end{aligned}$$

$$\begin{aligned} w_{5,3}^{t+1} &= w_{5,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,3}} \\ &= 0.4 - (0.1 \times -0.09075) \\ &= 0.409075 \end{aligned}$$

$$\begin{aligned} w_{5,0}^{t+1} &= w_{5,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,0}} \\ &= 0.1 - (0.1 \times -0.605) \\ &= 0.1605 \end{aligned}$$

$$\begin{aligned} w_{4,2}^{t+1} &= w_{4,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,2}} \\ &= 0.5 - (0.1 \times -0.09075) \\ &= 0.509075 \end{aligned}$$

$$\begin{aligned} w_{4,0}^{t+1} &= w_{4,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,0}} \\ &= 0.1 - (0.1 \times -0.363) \\ &= 0.1363 \end{aligned}$$

$$\begin{aligned} w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\ &= 0.2 - (0.1 \times -0.0605) \\ &= 0.20605 \end{aligned}$$

$$\begin{aligned}
 w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\
 &= 0.1 - (0.1 \times -0.242) \\
 &= 0.1242
 \end{aligned}$$

$$\begin{aligned}
 w_{2,1}^{t+1} &= w_{2,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,1}} \\
 &= 0.3 - (0.1 \times -0.11495) \\
 &= 0.311495
 \end{aligned}$$

$$\begin{aligned}
 w_{2,0}^{t+1} &= w_{2,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{2,0}} \\
 &= 0.1 - (0.1 \times -0.2299) \\
 &= 0.12299
 \end{aligned}$$

- v. Calculate the reduction in the error of the network for this example using the new weights, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned}
 z_2 &= ((w_{2,0} \times 1) + (w_{2,1} \times 0.5)) \\
 &= ((0.12299 \times 1) + (0.311495 \times 0.5)) \\
 &= 0.2787375
 \end{aligned}$$

$$\begin{aligned}
 a_2 &= \text{rectifier}(0.2787375) \\
 &= \max(0, 0.2787375) \\
 &= 0.2787375
 \end{aligned}$$

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,2} \times a_2)) \\
 &= ((0.1242 \times 1) + (0.20605 \times 0.2787375)) \\
 &= 0.181633862
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{rectifier}(0.181633862) \\
 &= \max(0, 0.181633862) \\
 &= 0.181633862
 \end{aligned}$$

$$\begin{aligned}
 z_4 &= ((w_{4,0} \times 1) + (w_{4,2} \times a_2)) \\
 &= ((0.1363 \times 1) + (0.509075 \times 0.2787375)) \\
 &= 0.278198293
 \end{aligned}$$

$$\begin{aligned}
 a_4 &= \text{rectifier}(0.278198293) \\
 &= \max(0, 0.278198293) \\
 &= 0.278198293
 \end{aligned}$$

$$\begin{aligned}
 z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\
 &= ((0.1605 \times 1) + (0.409075 \times 0.181633862) + (0.6136125 \times 0.278198293)) \\
 &= 0.405507822
 \end{aligned}$$

$$\begin{aligned}
 a_5 &= \text{rectifier}(0.405507822) \\
 &= \max(0, 0.405507822) \\
 &= 0.405507822
 \end{aligned}$$

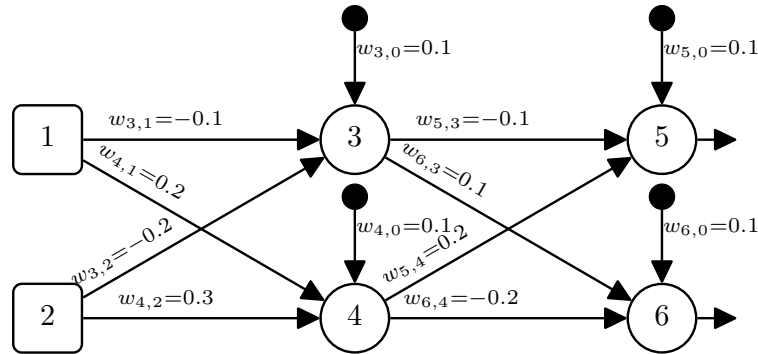
The error with the new weights is:

$$\begin{aligned}
 \text{Error}' &= 0.9 - 0.405507822 \\
 &= 0.494492178
 \end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned}
 \text{Error Reduction} &= 0.605 - 0.494492178 \\
 &= 0.110507822
 \end{aligned}$$

6. The following image illustrates the topology of a feedforward neural network that has two sensing neurons (Neurons 1 and 2), two hidden processing neuron (Neurons 3, and 4), and two processing output neurons (Neurons 5 and 6).



- (a) Assuming that the processing neurons use **logistic** activation functions, that the input to the network is Neuron 1 = 0.3 and Neuron 2 = 0.6, and that the desired output for this input is Neuron 5 = 0.7 and Neuron 6 = 0.4:
- Calculate the output generated by the network in response to this input.

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,1} \times a_1) + (w_{3,2} \times a_2)) \\
 &= ((0.1 \times 1) + (-0.1 \times 0.3) + (-0.2 \times 0.6)) \\
 &= -0.05
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{logistic}(-0.05) \\
 &= 0.487502604
 \end{aligned}$$

$$\begin{aligned}
 z_4 &= ((w_{4,0} \times 1) + (w_{4,1} \times a_1) + (w_{4,2} \times a_2)) \\
 &= ((0.1 \times 1) + (0.2 \times 0.3) + (0.3 \times 0.6)) \\
 &= 0.34
 \end{aligned}$$

$$\begin{aligned}
 a_4 &= \text{logistic}(0.34) \\
 &= 0.584190523
 \end{aligned}$$

$$\begin{aligned}
 z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\
 &= ((0.1 \times 1) + (-0.1 \times 0.487502604) + (0.2 \times 0.584190523)) \\
 &= 0.168087844
 \end{aligned}$$

$$\begin{aligned} a_5 &= \text{logistic}(0.168087844) \\ &= 0.541923301 \end{aligned}$$

$$\begin{aligned} z_6 &= ((w_{6,0} \times 1) + (w_{6,3} \times a_3) + (w_{6,4} \times a_4)) \\ &= ((0.1 \times 1) + (0.1 \times 0.487502604) + (-0.2 \times 0.584190523)) \\ &= 0.031912156 \end{aligned}$$

$$\begin{aligned} a_6 &= \text{logistic}(0.031912156) \\ &= 0.507977362 \end{aligned}$$

- ii. Calculate the sum of squared errors for this network for this example.

$$\begin{aligned} SSE &= (t_5 - a_5)^2 + (t_6 - a_6)^2 \\ &= (0.7 - 0.541923301)^2 + (0.4 - 0.507977362)^2 \\ &= (0.158076699)^2 + (-0.107977362)^2 \\ &= 0.036647354 \end{aligned}$$

- iii. Calculate the δ values for each of the processing neurons in the network (i.e., $\delta_6, \delta_5, \delta_4, \delta_3$).

Calculations for δ_6 :

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial a_6} &= -(t_6 - a_6) \\ &= -(0.4 - 0.507977362) \\ &= -(-0.107977362) \\ &= 0.107977362 \end{aligned}$$

$$\begin{aligned} \frac{\partial a_6}{\partial z_6} &= \text{logistic}(z_6) \times (1 - \text{logistic}(z_6)) \\ &= 0.507977362 \times (1 - 0.507977362) \\ &= 0.249936362 \end{aligned}$$

$$\begin{aligned}
\delta_6 &= \frac{\partial \mathcal{E}}{\partial a_6} \times \frac{\partial a_6}{\partial z_6} \\
&= 0.107977362 \times 0.249936362 \\
&= 0.026987469
\end{aligned}$$

Calculations for δ_5 :

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial a_5} &= -(t_5 - a_5) \\
&= -(0.7 - 0.541923301) \\
&= -0.158076699
\end{aligned}$$

$$\begin{aligned}
\frac{\partial a_5}{\partial z_5} &= \text{logistic}(z_5) \times (1 - \text{logistic}(z_5)) \\
&= 0.541923301 \times (1 - 0.541923301) \\
&= -0.039241345
\end{aligned}$$

$$\begin{aligned}
\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_5} \times \frac{\partial a_5}{\partial z_5} \\
&= -0.158076699 \times 0.248242437 \\
&= -0.039241345
\end{aligned}$$

Calculations for δ_4 :

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial a_4} &= (w_{5,4} \times \delta_5) + (w_{6,4} \times \delta_6) \\
&= (0.2 \times -0.039241345) + (-0.2 \times 0.026987469) \\
&= -0.013245763
\end{aligned}$$

$$\begin{aligned}
\frac{\partial a_4}{\partial z_4} &= \text{logistic}(z_4) \times (1 - \text{logistic}(z_4)) \\
&= 0.584190523 \times (1 - 0.584190523) \\
&= 0.242911956
\end{aligned}$$

$$\begin{aligned}
\delta_4 &= \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} \\
&= -0.013245763 \times 0.242911956 \\
&= -0.003217554
\end{aligned}$$

Calculations for δ_3 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_3} &= (w_{5,3} \times \delta_5) + (w_{6,3} \times \delta_6) \\ &= (-0.1 \times -0.039241345) + (0.1 \times 0.026987469) \\ &= 0.006622881\end{aligned}$$

$$\begin{aligned}\frac{\partial a_3}{\partial z_3} &= \text{logistic}(z_3) \times (1 - \text{logistic}(z_3)) \\ &= 0.487502604 \times (1 - 0.487502604) \\ &= 0.249843815\end{aligned}$$

$$\begin{aligned}\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\ &= 0.006622881 \times 0.249843815 \\ &= 0.001654686\end{aligned}$$

- iv. Using the δ values you calculated above, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial \mathcal{E} / \partial w_{6,4}$, $\partial \mathcal{E} / \partial w_{6,3}$, $\partial \mathcal{E} / \partial w_{6,0}$, $\partial \mathcal{E} / \partial w_{5,4}$, $\partial \mathcal{E} / \partial w_{5,3}$, $\partial \mathcal{E} / \partial w_{5,0}$, $\partial \mathcal{E} / \partial w_{4,2}$, $\partial \mathcal{E} / \partial w_{4,1}$, $\partial \mathcal{E} / \partial w_{4,0}$, $\partial \mathcal{E} / \partial w_{3,2}$, $\partial \mathcal{E} / \partial w_{3,1}$, $\partial \mathcal{E} / \partial w_{3,0}$).

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{6,4}} &= \delta_6 \times a_4 \\ &= 0.026987469 \times 0.584190523 \\ &= 0.015765824\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{6,3}} &= \delta_6 \times a_3 \\ &= 0.026987469 \times 0.487502604 \\ &= 0.013156461\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,0}} &= \delta_6 \times a_0 \\
&= 0.026987469 \times 1 \\
&= 0.026987469
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,4}} &= \delta_5 \times a_4 \\
&= -0.039241345 \times 0.584190523 \\
&= -0.022924422
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,3}} &= \delta_5 \times a_3 \\
&= -0.039241345 \times 0.487502604 \\
&= -0.019130258
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,0}} &= \delta_5 \times a_0 \\
&= -0.039241345 \times 1 \\
&= -0.039241345
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{4,2}} &= \delta_4 \times a_2 \\
&= -0.003217554 \times 0.6 \\
&= -0.001930532
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{4,1}} &= \delta_4 \times a_1 \\
&= -0.003217554 \times 0.3 \\
&= -0.000965266
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{4,0}} &= \delta_4 \times a_0 \\
&= -0.003217554 \times 1 \\
&= -0.003217554
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\
&= 0.001654686 \times 0.6 \\
&= 0.000992812
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,1}} &= \delta_3 \times a_1 \\
&= 0.001654686 \times 0.3 \\
&= 0.000496406
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\
&= 0.001654686 \times 1 \\
&= 0.001654686
\end{aligned}$$

- v. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{6,4}, w_{6,3}, w_{6,0}, w_{5,4}, w_{5,3}, w_{5,0}, w_{4,2}, w_{4,1}, w_{4,0}, w_{3,2}, w_{3,1}, w_{3,0}$) after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned}
w_{6,4}^{t+1} &= w_{6,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,4}} \\
&= -0.2 - (0.1 \times 0.015765824) \\
&= -0.201576582
\end{aligned}$$

$$\begin{aligned}
w_{6,3}^{t+1} &= w_{6,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,3}} \\
&= 0.1 - (0.1 \times 0.013156461) \\
&= 0.098684354
\end{aligned}$$

$$\begin{aligned}
w_{6,0}^{t+1} &= w_{5,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,0}} \\
&= 0.1 - (0.1 \times 0.026987469) \\
&= 0.097301253
\end{aligned}$$

$$\begin{aligned}
w_{5,4}^{t+1} &= w_{5,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,4}} \\
&= 0.2 - (0.1 \times -0.022924422) \\
&= 0.202292442
\end{aligned}$$

$$\begin{aligned}
w_{5,3}^{t+1} &= w_{5,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,3}} \\
&= -0.1 - (0.1 \times -0.019130258) \\
&= -0.098086974
\end{aligned}$$

$$\begin{aligned}
w_{5,0}^{t+1} &= w_{5,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,0}} \\
&= 0.1 - (0.1 \times -0.039241345) \\
&= 0.103924135
\end{aligned}$$

$$\begin{aligned}
w_{4,2}^{t+1} &= w_{4,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,2}} \\
&= 0.3 - (0.1 \times -0.001930532) \\
&= 0.300193053
\end{aligned}$$

$$\begin{aligned}
w_{4,1}^{t+1} &= w_{4,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,1}} \\
&= 0.2 - (0.1 \times -0.000965266) \\
&= 0.200096527
\end{aligned}$$

$$\begin{aligned}
 w_{4,0}^{t+1} &= w_{4,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,0}} \\
 &= 0.1 - (0.1 \times -0.003217554) \\
 &= 0.100321755
 \end{aligned}$$

$$\begin{aligned}
 w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\
 &= -0.2 - (0.1 \times 0.000992812) \\
 &= -0.200099281
 \end{aligned}$$

$$\begin{aligned}
 w_{3,1}^{t+1} &= w_{3,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,1}} \\
 &= -0.1 - (0.1 \times 0.000496406) \\
 &= -0.100049641
 \end{aligned}$$

$$\begin{aligned}
 w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\
 &= 0.1 - (0.1 \times 0.001654686) \\
 &= 0.099834531
 \end{aligned}$$

- vi. Calculate the reduction in the sum of squared error of the network for this example using the new weights, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,1} \times a_1) + (w_{3,2} \times a_2)) \\
 &= ((0.099834531 \times 1) + (-0.100049641 \times 0.3) + (-0.200099281 \times 0.6)) \\
 &= -0.050239929
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{logistic}(-0.050239929) \\
 &= 0.487442659
 \end{aligned}$$

$$\begin{aligned}
z_4 &= ((w_{4,0} \times 1) + (w_{4,1} \times a_1) + (w_{4,2} \times a_2)) \\
&= ((0.100321755 \times 1) + (0.200096527 \times 0.3) + (0.300193053 \times 0.6)) \\
&= 0.340466545
\end{aligned}$$

$$\begin{aligned}
a_4 &= \text{logistic}(0.340466545) \\
&= 0.584303848
\end{aligned}$$

$$\begin{aligned}
z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\
&= ((0.103924135 \times 1) + (-0.098086974 \times 0.487442659) + (0.202292442 \times 0.584303848)) \\
&= 0.174312611
\end{aligned}$$

$$\begin{aligned}
a_5 &= \text{logistic}(0.174312611) \\
&= 0.543468144
\end{aligned}$$

$$\begin{aligned}
z_6 &= ((w_{6,0} \times 1) + (w_{6,3} \times a_3) + (w_{6,4} \times a_4)) \\
&= ((0.097301253 \times 1) + (0.098684354 \times 0.487442659) + (-0.201576582 \times 0.584303848)) \\
&= 0.027622244
\end{aligned}$$

$$\begin{aligned}
a_6 &= \text{logistic}(0.027622244) \\
&= 0.506905122
\end{aligned}$$

If we now calculate the sum of squared errors on this example for this network using the new weights we get:

$$\begin{aligned}
SSE &= (t_5 - a_5)^2 + (t_6 - a_6)^2 \\
&= (0.7 - 0.543468144)^2 + (0.4 - 0.506905122)^2 \\
&= (0.156531856)^2 + (-0.106905122)^2 \\
&= 0.035930927
\end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned}
\text{Error Reduction} &= 0.036647354 - 0.035930927 \\
&= 0.000716426
\end{aligned}$$

- (b) Assuming that the processing neurons use a **rectifier** activation functions, that the input to the network is Neuron 1 = 0.3 and Neuron 2 = 0.6 and that the desired output for this input is Neuron 5 = 0.7 and Neuron 6 = 0.4:
- i. Calculate the output generated by the network in response to this input.

$$\begin{aligned} z_3 &= ((w_{3,0} \times 1) + (w_{3,1} \times a_1) + (w_{3,2} \times a_2)) \\ &= ((0.1 \times 1) + (-0.1 \times 0.3) + (-0.2 \times 0.6)) \\ &= -0.05 \end{aligned}$$

$$\begin{aligned} a_3 &= \text{rectifier}(-0.05) \\ &= \max(0, -0.05) \\ &= 0 \end{aligned}$$

$$\begin{aligned} z_4 &= ((w_{4,0} \times 1) + (w_{4,1} \times a_1) + (w_{4,2} \times a_2)) \\ &= ((0.1 \times 1) + (0.2 \times 0.3) + (0.3 \times 0.6)) \\ &= 0.34 \end{aligned}$$

$$\begin{aligned} a_4 &= \text{rectifier}(0.34) \\ &= \max(0, 0.34) \\ &= 0.34 \end{aligned}$$

$$\begin{aligned} z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\ &= ((0.1 \times 1) + (-0.1 \times 0) + (0.2 \times 0.34)) \\ &= 0.168 \end{aligned}$$

$$\begin{aligned} a_5 &= \text{rectifier}(0.168) \\ &= \max(0, 0.168) \\ &= 0.168 \end{aligned}$$

$$\begin{aligned} z_6 &= ((w_{6,0} \times 1) + (w_{6,3} \times a_3) + (w_{6,4} \times a_4)) \\ &= ((0.1 \times 1) + (0.1 \times 0) + (-0.2 \times 0.34)) \\ &= 0.032 \end{aligned}$$

$$\begin{aligned}
 a_6 &= \text{rectifier}(0.032) \\
 &= \max(0, 0.032) \\
 &= 0.032
 \end{aligned}$$

- ii. Calculate the sum of squared errors for this network on this example.

$$\begin{aligned}
 SSE &= (t_5 - a_5)^2 + (t_6 - a_6)^2 \\
 &= (0.7 - 0.168)^2 + (0.4 - 0.032)^2 \\
 &= (0.532)^2 + (0.368)^2 \\
 &= 0.418448
 \end{aligned}$$

- iii. Calculate the δ values for each of the processing neurons in the network (i.e., $\delta_6, \delta_5, \delta_4, \delta_3$).

Calculations for δ_6 :

$$\begin{aligned}
 \frac{\partial \mathcal{E}}{\partial a_6} &= -(t_6 - a_6) \\
 &= -(0.4 - 0.032) \\
 &= -(0.368) \\
 &= -0.368
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial a_6}{\partial z_6} &= \begin{cases} 1 & \text{if } z_6 > 1 \\ 0 & \text{otherwise} \end{cases} \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \delta_6 &= \frac{\partial \mathcal{E}}{\partial a_6} \times \frac{\partial a_6}{\partial z_6} \\
 &= -0.368 \times 1 \\
 &= -0.368
 \end{aligned}$$

Calculations for δ_5 :

$$\begin{aligned}
 \frac{\partial \mathcal{E}}{\partial a_5} &= -(t_5 - a_5) \\
 &= -(0.7 - 0.168) \\
 &= -0.532
 \end{aligned}$$

$$\begin{aligned}\frac{\partial a_5}{\partial z_5} &= \begin{cases} 1 & \text{if } z_5 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 1\end{aligned}$$

$$\begin{aligned}\delta_5 &= \frac{\partial \mathcal{E}}{\partial a_5} \times \frac{\partial a_5}{\partial z_5} \\ &= -0.532 \times 1 \\ &= -0.532\end{aligned}$$

Calculations for δ_4 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_4} &= (w_{5,4} \times \delta_5) + (w_{6,4} \times \delta_6) \\ &= (0.2 \times -0.532) + (-0.2 \times -0.368) \\ &= -0.0328\end{aligned}$$

$$\begin{aligned}\frac{\partial a_4}{\partial z_4} &= \begin{cases} 1 & \text{if } z_4 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 1\end{aligned}$$

$$\begin{aligned}\delta_4 &= \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} \\ &= -0.0328 \times 1 \\ &= -0.0328\end{aligned}$$

Calculations for δ_3 :

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial a_3} &= (w_{5,3} \times \delta_5) + (w_{6,3} \times \delta_6) \\ &= (-0.1 \times -0.532) + (0.1 \times -0.368) \\ &= 0.0164\end{aligned}$$

$$\begin{aligned}\frac{\partial a_3}{\partial z_3} &= \begin{cases} 1 & \text{if } z_3 > 1 \\ 0 & \text{otherwise} \end{cases} \\ &= 0\end{aligned}$$

$$\begin{aligned}
\delta_3 &= \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \\
&= 0.0164 \times 0 \\
&= 0
\end{aligned}$$

- iv. Using the δ values you calculated above, calculate the sensitivity of the error of the network to changes in each of the weights of the network (i.e., $\partial \mathcal{E} / \partial w_{6,4}$, $\partial \mathcal{E} / \partial w_{6,3}$, $\partial \mathcal{E} / \partial w_{6,0}$, $\partial \mathcal{E} / \partial w_{5,4}$, $\partial \mathcal{E} / \partial w_{5,3}$, $\partial \mathcal{E} / \partial w_{5,0}$, $\partial \mathcal{E} / \partial w_{4,2}$, $\partial \mathcal{E} / \partial w_{4,1}$, $\partial \mathcal{E} / \partial w_{4,0}$, $\partial \mathcal{E} / \partial w_{3,2}$, $\partial \mathcal{E} / \partial w_{3,1}$, $\partial \mathcal{E} / \partial w_{3,0}$).

To calculate the sensitivity of the error of the network to changes in a weight we multiply the δ for the neuron that uses the weight in its weighted sum by the activation (or input) that the weight was applied to in the calculation of the weight sum. The calculations for each of the weights is shown below.

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{6,4}} &= \delta_6 \times a_4 \\
&= -0.368 \times 0.34 \\
&= -0.12512
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{6,3}} &= \delta_6 \times a_3 \\
&= -0.368 \times 0 \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,0}} &= \delta_6 \times a_0 \\
&= -0.368 \times 1 \\
&= -0.368
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{5,4}} &= \delta_5 \times a_4 \\
&= -0.532 \times 0.34 \\
&= -0.18088
\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{5,3}} &= \delta_5 \times a_3 \\ &= -0.532 \times 0 \\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{5,0}} &= \delta_5 \times a_0 \\ &= -0.532 \times 1 \\ &= -0.532\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{4,2}} &= \delta_4 \times a_2 \\ &= -0.0328 \times 0.6 \\ &= -0.01968\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{4,1}} &= \delta_4 \times a_1 \\ &= -0.0328 \times 0.3 \\ &= -0.00984\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{4,0}} &= \delta_4 \times a_0 \\ &= -0.0328 \times 1 \\ &= -0.0328\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{3,2}} &= \delta_3 \times a_2 \\ &= 0 \times 0.6 \\ &= 0\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{E}}{\partial w_{3,1}} &= \delta_3 \times a_1 \\ &= 0 \times 0.3 \\ &= 0\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial w_{3,0}} &= \delta_3 \times a_0 \\
&= 0 \times 1 \\
&= 0
\end{aligned}$$

- v. Assuming a learning rate of $\alpha = 0.1$, calculate the updated values for each of the weights in the network ($w_{6,4}, w_{6,3}, w_{6,0}, w_{5,4}, w_{5,3}, w_{5,0}, w_{4,2}, w_{4,1}, w_{4,0}, w_{3,2}, w_{3,1}, w_{3,0}$) after the processing of this single training example.

We update weights as follows:

$$w_{i,k}^{t+1} = w_{i,k}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{i,k}}$$

The calculation for each weight update is below:

$$\begin{aligned}
w_{6,4}^{t+1} &= w_{6,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,4}} \\
&= -0.2 - (0.1 \times -0.12512) \\
&= -0.187488
\end{aligned}$$

$$\begin{aligned}
w_{6,3}^{t+1} &= w_{6,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,3}} \\
&= 0.1 - (0.1 \times 0) \\
&= 0.1
\end{aligned}$$

$$\begin{aligned}
w_{6,0}^{t+1} &= w_{6,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{6,0}} \\
&= 0.1 - (0.1 \times -0.368) \\
&= 0.1368
\end{aligned}$$

$$\begin{aligned}
w_{5,4}^{t+1} &= w_{5,4}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,4}} \\
&= 0.2 - (0.1 \times -0.18088) \\
&= 0.218088
\end{aligned}$$

$$\begin{aligned}
 w_{5,3}^{t+1} &= w_{5,3}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,3}} \\
 &= -0.1 - (0.1 \times 0) \\
 &= -0.1
 \end{aligned}$$

$$\begin{aligned}
 w_{5,0}^{t+1} &= w_{5,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{5,0}} \\
 &= 0.1 - (0.1 \times -0.532) \\
 &= 0.1532
 \end{aligned}$$

$$\begin{aligned}
 w_{4,2}^{t+1} &= w_{4,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,2}} \\
 &= 0.3 - (0.1 \times -0.01968) \\
 &= 0.301968
 \end{aligned}$$

$$\begin{aligned}
 w_{4,1}^{t+1} &= w_{4,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,1}} \\
 &= 0.2 - (0.1 \times -0.00984) \\
 &= 0.200984
 \end{aligned}$$

$$\begin{aligned}
 w_{4,0}^{t+1} &= w_{4,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{4,0}} \\
 &= 0.1 - (0.1 \times -0.0328) \\
 &= 0.10328
 \end{aligned}$$

$$\begin{aligned}
 w_{3,2}^{t+1} &= w_{3,2}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,2}} \\
 &= -0.2 - (0.1 \times 0) \\
 &= -0.2
 \end{aligned}$$

$$\begin{aligned}
 w_{3,1}^{t+1} &= w_{3,1}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,1}} \\
 &= -0.1 - (0.1 \times 0) \\
 &= -0.1
 \end{aligned}$$

$$\begin{aligned}
 w_{3,0}^{t+1} &= w_{3,0}^t - \alpha \times \frac{\partial \mathcal{E}}{\partial w_{3,0}} \\
 &= 0.1 - (0.1 \times 0) \\
 &= 0.1
 \end{aligned}$$

- vi. Calculate the reduction in the sum of squared error of the network for this example using the new weights, compared with using the original weights.

The output for the network using the new weights is:

$$\begin{aligned}
 z_3 &= ((w_{3,0} \times 1) + (w_{3,1} \times a_1) + (w_{3,2} \times a_2)) \\
 &= ((0.1 \times 1) + (-0.1 \times 0.3) + (-0.2 \times 0.6)) \\
 &= -0.05
 \end{aligned}$$

$$\begin{aligned}
 a_3 &= \text{rectifier}(-0.05) \\
 &= \max(0, -0.05) \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 z_4 &= ((w_{4,0} \times 1) + (w_{4,1} \times a_1) + (w_{4,2} \times a_2)) \\
 &= ((0.10328 \times 1) + (0.200984 \times 0.3) + (0.301968 \times 0.6)) \\
 &= 0.344756
 \end{aligned}$$

$$\begin{aligned}
 a_4 &= \text{rectifier}(0.344756) \\
 &= \max(0, 0.344756) \\
 &= 0.344756
 \end{aligned}$$

$$\begin{aligned}
 z_5 &= ((w_{5,0} \times 1) + (w_{5,3} \times a_3) + (w_{5,4} \times a_4)) \\
 &= ((0.1532 \times 1) + (-0.1 \times 0) + (0.218088 \times 0.344756)) \\
 &= 0.228387147
 \end{aligned}$$

$$\begin{aligned}
 a_5 &= \text{rectifier}(0.228387147) \\
 &= \max(0, 0.228387147) \\
 &= 0.228387147
 \end{aligned}$$

$$\begin{aligned}
z_6 &= ((w_{6,0} \times 1) + (w_{6,3} \times a_3) + (w_{6,4} \times a_4)) \\
&= ((0.1368 \times 1) + (0.1 \times 0) + (-0.187488 \times 0.344756)) \\
&= 0.072162387
\end{aligned}$$

$$\begin{aligned}
a_6 &= \text{rectifier}(0.072162387) \\
&= \max(0, 0.072162387) \\
&= 0.072162387
\end{aligned}$$

If we now calculate the sum of squared errors on this example for this network using the new weights we get:

$$\begin{aligned}
SSE &= (t_5 - a_5)^2 + (t_6 - a_6)^2 \\
&= (0.7 - 0.228387147)^2 + (0.4 - 0.072162387)^2 \\
&= (0.471612853)^2 + (0.327837613)^2 \\
&= 0.329896184
\end{aligned}$$

Subtracting this new error from the error with the original weights (which we calculated as part of the calculation for δ_3 above) gives us the reduction in error:

$$\begin{aligned}
\text{Error Reduction} &= 0.418448 - 0.329896184 \\
&= 0.088551816
\end{aligned}$$

7. Assuming a fully connected feedforward network where all the neurons uses a linear activation function (i.e., $a_i = z_i$) and with the following topology:

- (a) 100 neurons in the input layer
- (b) 5 hidden layers with 2,000 neurons in each layer
- (c) 10 neurons in the output layer

If all the inputs to the network have been standardized to have a mean value of 0 and a standard deviation of 1, and the initial weights for the network are sampled from a normal distribution with mean 0.0 and standard deviation of $\sigma = 0.01$, then:

- (a) Calculate the variance of the z values across for the neurons in the first hidden layer in the first iteration of training.

$$\begin{aligned}
 \text{var}(Z^{(1)}) &= n_{in}^{(1)} \times \text{var}(W^{(1)}) \times \text{var}(\mathbf{d}^{(1)}) \\
 &= 100 \times 0.0001 \times 1 \\
 &= 0.01
 \end{aligned} \tag{8.1}$$

- (b) Calculate the variance of the z values across for the neurons in the last hidden layer in the first iteration of training.

$$\begin{aligned}
 \text{var}(Z^{(5)}) &= (n_{in}^{(2,\dots,5)}) \times \text{var}(W^{(2,\dots,5)})^4 \times \text{var}(\mathbf{d}^{(2)}) \\
 &= (2000 \times 0.0001)^4 \times 0.01 \\
 &= 0.00016
 \end{aligned} \tag{8.2}$$

Assuming that the variance of the δ s for the output layer is equal to 1:

- (a) Calculate the variance of the δ s across for the neurons in the last hidden layer in the first iteration of training.

$$\begin{aligned}
 \text{var}(\delta^{(5)}) &= n_{out}^{(5)} \times \text{var}(W^{(5)}) \times \text{var}(\mathbf{d}^{(5)}) \\
 &= 10 \times 0.0001 \times 1 \\
 &= 0.001
 \end{aligned} \tag{8.3}$$

- (b) Calculate the variance of the δ s values across for the neurons in the first hidden layer in the first iteration of training.

$$\begin{aligned}
 \text{var}(\delta^{(1)}) &= (n_{out}^{(1,\dots,4)}) \times \text{var}(W^{(1,\dots,4)})^4 \times \text{var}(\mathbf{d}^{(4)}) \\
 &= (2000 \times 0.0001)^4 \times 0.001 \\
 &= 0.0000016
 \end{aligned} \tag{8.4}$$

- (c) Is the training dynamic of this network stable, or is it suffering from vanishing or exploding gradients?

The training dynamic of this network is exhibiting vanishing gradients because the variance of the δ s is reducing through each layer that they are backpropagated through.

- * 8. Assuming a network architecture that has four neurons in a **softmax** output layer. If the one-hot encoding of the target for the current training example is $\mathbf{t} = [0, 0, 1, 0]$

and the logits for the four neurons in the softmax output layer for this example are $[0, 0.5, 0.25, 0.75]$, then what is the δ value for each of the four neurons?

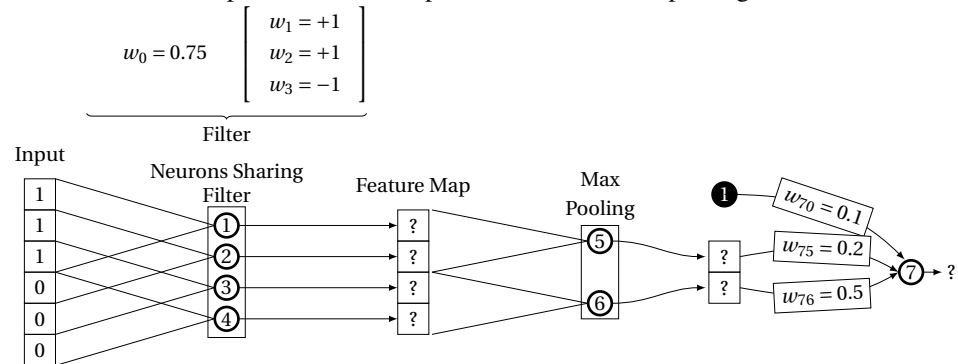
<i>logits</i>	0	0.5	0.25	0.75
e^{l_i}	1	1.648721271	1.284025417	2.117000017
$\sum_i e^{l_i}$				6.049746704
a_i	0	0.082648088	0.041324044	0.123972133
\mathbf{t}	0	0	1	0
δs	0	0.082648088	0.958675956	0.123972133

- * 9. Assuming a feedforward neural network that has 4 neurons in hidden layer k and that we are training this network using **inverted dropout** with $\rho = 0.5$. If the activations for the neurons in layer k are as follows: $[0.2, 0, 4, 0, 3, 0.1]$ and the **DropMask** for layer k is $[1, 0, 1, 0]$, calculate the activation vector that is actually propagated to layer $k + 1$ after inverted dropout has been applied.

\mathbf{a}	0.2	0.4	0.3	0.1
DropMask	1	0	1	0
\mathbf{a}'	0.2	0	0.3	0
$\mathbf{a}'' = \mathbf{a}' \times \frac{1}{\rho}$	0.4	0	0.6	0

- * 10. The figure below illustrates a layer of a convolutional neural network that is processing a one-dimensional input. For ease of reference each of the neurons in the network has been labeled: 1, 2, 3, 4, 5, 6, 7. The architecture of the network consists of ReLUs that share a filter (Neurons 1, 2, 3, 4), followed by a sub-sampling layer containing two max pooling units (Neurons 5, 6), and then a fully connected layer containing a single ReLU (Neuron 7). The ReLU in the first layer has a 3-by-1 receptive field, and there is a stride of 1 used in this layer. The max pooling units have a receptive field of 2-by-1,

and there is no overlap between the receptive fields of the max pooling units.



(a) What value will this network output?

$$\begin{aligned}
 z_1 &= 1.75 \\
 z_2 &= 2.75 \\
 z_3 &= 1.75 \\
 z_4 &= 0.75 \\
 a_1 &= 1.75 \\
 a_2 &= 2.75 \\
 a_3 &= 1.75 \\
 a_4 &= 0.75 \\
 a_5 &= 2.75 \\
 a_6 &= 1.75 \\
 z_7 &= 1.525 \\
 a_7 &= 1.525
 \end{aligned}$$

(b) Assuming the target output for this input is 1, calculate the δ for each neuron in the network.

$$Error = 1 - 1.525 = -0.525$$

Neuron 7

$$\frac{\partial \mathcal{E}}{\partial a} = -0.525 \times -1 = 0.525$$

$$\frac{\partial a}{\partial z} = 1 \text{ ReLU with output } > 1$$

$$\delta_7 = \frac{\partial \mathcal{E}}{\partial a} \times \frac{\partial a}{\partial z} = 0.525$$

Neuron 6

$$\frac{\partial \mathcal{E}}{\partial a} = \delta_7 \times w_{76} = 0.525 \times 0.5 = 0.2625$$

$$\frac{\partial a}{\partial z} = 1 \text{ ReLU with output } > 1$$

$$\delta_6 = \frac{\partial \mathcal{E}}{\partial a} \times \frac{\partial a}{\partial z} = 0.2625$$

Neuron 5

$$\frac{\partial \mathcal{E}}{\partial a} = \delta_7 \times w_{75} = 0.525 \times 0.2 = 0.105$$

$$\frac{\partial a}{\partial z} = 1 \text{ ReLU with output } > 1$$

$$\delta_5 = \frac{\partial \mathcal{E}}{\partial a} \times \frac{\partial a}{\partial z} = 0.105$$

Neurons 1 and 2 are in the local receptive field of Neuron 5 (a max pool unit) and Neurons 3 and 4 are in the local receptive field of neuron 6 (the other max pool unit).

Neuron 4

$$\delta_4 = 0 \text{ Not max output in receptive field of max pool 6}$$

Neuron 3

$$\frac{\partial \mathcal{E}}{\partial a} = \delta_6 \times w_{63} = 0.2625 \times 1 = 0.2625$$

$$\frac{\partial a}{\partial z} = 1 \quad \text{ReLU with output} > 1$$

$$\delta_3 = \frac{\partial \mathcal{E}}{\partial a} \times \frac{\partial a}{\partial z} = 0.2625$$

Neuron 2

$$\frac{\partial \mathcal{E}}{\partial a} = \delta_5 \times w_{52} = 0.105 \times 1 = 0.105$$

$$\frac{\partial a}{\partial z} = 1 \quad \text{ReLU with output} > 1$$

$$\delta_2 = \frac{\partial \mathcal{E}}{\partial a} \times \frac{\partial a}{\partial z} = 0.105$$

Neuron 1

$$\delta_1 = 0 \quad \text{Not max output in receptive field of max pool 5}$$

(c) Calculate the weight update for each weight in the filter: w_0, w_1, w_2, w_3 .

$$w_0$$

$$\delta_1 \times input = 0.0000 \times 1 = 0.0000$$

$$\delta_2 \times input = 0.1050 \times 1 = 0.1050$$

$$\delta_3 \times input = 0.2626 \times 1 = 0.2626$$

$$\delta_4 \times input = 0.0000 \times 1 = 0.0000$$

$$\Delta w_0 = 0.3675$$

$$w_1$$

$$\delta_1 \times input = 0.0000 \times 1 = 0.0000$$

$$\delta_2 \times input = 0.1050 \times 1 = 0.1050$$

$$\delta_3 \times input = 0.2626 \times 1 = 0.2626$$

$$\delta_4 \times input = 0.0000 \times 0 = 0.0000$$

$$\Delta w_1 = 0.3675$$

$$w_2$$

$$\delta_1 \times input = 0.0000 \times 1 = 0.0000$$

$$\delta_2 \times input = 0.1050 \times 1 = 0.1050$$

$$\delta_3 \times input = 0.2626 \times 0 = 0.0000$$

$$\delta_4 \times input = 0.0000 \times 0 = 0.0000$$

$$\Delta w_2 = 0.1050$$

$$w_3$$

$$\delta_1 \times input = 0.0000 \times 1 = 0.0000$$

$$\delta_2 \times input = 0.1050 \times 0 = 0.0000$$

$$\delta_3 \times input = 0.2626 \times 0 = 0.0000$$

$$\delta_4 \times input = 0.0000 \times 0 = 0.0000$$

$$\Delta w_3 = 0.000$$

- * 11. Assume a simple recurrent neural network architecture matching the one shown in the detailed schematic on the left of Figure 8.37^[502]. This network has two input neurons, three ReLUs in the hidden layer, and two ReLUs in the output layer. Also, all the bias terms in the network are equal to 0.1, and the weight matrices of the network (excluding bias terms) are listed in Equation (8.5)^[228].

$$\underbrace{\begin{bmatrix} -0.07 & 0.05 \\ -0.04 & 0.1 \\ -0.05 & -0.05 \end{bmatrix}}_{w_{hx}} \underbrace{\begin{bmatrix} -0.22 & -0.1 & 0.05 \\ -0.04 & -0.09 & -0.06 \\ -0.09 & 0 & -0.08 \end{bmatrix}}_{w_{hh}} \underbrace{\begin{bmatrix} 0.06 & -0.01 & -0.18 \\ -0.06 & 0.13 & 0.14 \end{bmatrix}}_{w_{yh}} \quad (8.5)$$

- (a) If $\mathbf{x}_t = [1, 0.5]$ and $\mathbf{h}_{t-1} = [0.05, 0.2, 0.15]$, calculate the value of \mathbf{y}_t .

$$\begin{aligned}
\mathbf{W}_{hx} \cdot \mathbf{x}_t &= \begin{bmatrix} -0.045 \\ 0.01 \\ -0.075 \end{bmatrix} \\
\mathbf{W}_{hh} \cdot \mathbf{h}_{t-1} &= \begin{bmatrix} -0.0235 \\ -0.029 \\ -0.0165 \end{bmatrix} \\
\mathbf{h}_t &= \varphi_{ReLU} \left(\begin{bmatrix} -0.045 & + & -0.0235 & + & 0.1 \\ 0.01 & + & -0.029 & + & 0.1 \\ -0.075 & + & -0.0165 & + & 0.1 \end{bmatrix} \right) = \begin{bmatrix} 0.0315 \\ 0.081 \\ 0.0085 \end{bmatrix} \\
\text{Augment } \mathbf{h}_t \text{ with the bias input then:} \\
\mathbf{y}_t &= \varphi_{ReLU} (\mathbf{W}_{yh} \cdot \mathbf{h}_t) \\
&= \begin{bmatrix} 0.1 & 0.06 & -0.01 & -0.18 \\ 0.1 & -0.06 & 0.13 & 0.14 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0.0315 \\ 0.081 \\ 0.0085 \end{bmatrix} = \begin{bmatrix} 0.09955 \\ 0.10983 \end{bmatrix}
\end{aligned}$$

- (b) Assuming that the target output for time $t_i = [0, 1]$, calculate the δ value for each neuron in the network.

$$Error_7 = 1 - 0.10983 = 0.89017$$

$$\frac{\partial \mathcal{E}}{\partial a_7} = -0.89017$$

$$\frac{\partial a_7}{\partial z_7} = 1$$

$$\delta_7 = \frac{\partial \mathcal{E}}{\partial a_7} \times \frac{\partial a_7}{\partial z_7} = -0.89017$$

$$Error_6 = 0 - 0.09955 = -0.09955$$

$$\frac{\partial \mathcal{E}}{\partial a_6} = 0.09955$$

$$\frac{\partial a_6}{\partial z_6} = 1$$

$$\delta_6 = \frac{\partial \mathcal{E}}{\partial a_6} \times \frac{\partial a_6}{\partial z_6} = 0.09955$$

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial a_5} &= (\delta_7 \times w_{7,5}) + (\delta_6 \times w_{6,5}) \\ &= (-0.89017 \times 0.14) + (0.09955 \times -0.18) \\ &= -0.1425428 \end{aligned}$$

$$\frac{\partial a_5}{\partial z_5} = 1$$

$$\delta_5 = \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} = -0.1425428$$

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial a_4} &= (\delta_7 \times w_{7,4}) + (\delta_6 \times w_{6,4}) \\ &= (-0.89017 \times 0.13) + (0.09955 \times -0.01) \\ &= -0.1167176 \end{aligned}$$

$$\frac{\partial a_4}{\partial z_4} = 1$$

$$\delta_4 = \frac{\partial \mathcal{E}}{\partial a_4} \times \frac{\partial a_4}{\partial z_4} = -0.1167176$$

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial a_3} &= (\delta_7 \times w_{7,3}) + (\delta_6 \times w_{6,3}) \\ &= (-0.89017 \times -0.06) + (0.09955 \times 0.06) \\ &= 0.0593832 \end{aligned}$$

$$\frac{\partial a_3}{\partial z_3} = 1$$

$$\delta_3 = \frac{\partial \mathcal{E}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} = 0.0593832$$

* 12. Assuming that the LSTM cell state propagated forward from the last time-step is

$$\mathbf{c}_{t-1} = \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix}$$

(a) What will be the value of \mathbf{c}_t if

$$\mathbf{f}_t = \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \quad \mathbf{i}_{\uparrow t} = \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \quad \mathbf{i}_{\downarrow t} = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix}$$

$$\begin{aligned} \mathbf{c}_{\uparrow}^{\dagger} &= \mathbf{c}_{t-1} \odot \mathbf{f}_t \\ &= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \odot \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} = \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \\ \mathbf{i}_t &= \mathbf{i}_{\uparrow t} \odot \mathbf{i}_{\downarrow t} \\ &= \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \odot \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} \\ \mathbf{c}_t &= \mathbf{c}_{\uparrow}^{\dagger} + \mathbf{i}_t \\ &= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} + \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} +0.25 \\ -0.25 \end{bmatrix} \end{aligned}$$

(b) What will be the value of \mathbf{c}_t if

$$\mathbf{f}_t = \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \quad \mathbf{i}_{\uparrow t} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \quad \mathbf{i}_{\downarrow t} = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix}$$

$$\begin{aligned}
\mathbf{c}_t^\dagger &= \mathbf{c}_{t-1} \odot \mathbf{f}_t \\
&= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \odot \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} = \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \\
\mathbf{i}_t &= \mathbf{i}_t^\dagger \odot \mathbf{i}_t^\ddagger \\
&= \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \odot \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \\
\mathbf{c}_t &= \mathbf{c}_t^\dagger + \mathbf{i}_t \\
&= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} = \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix}
\end{aligned}$$

(c) What will be the value of \mathbf{c}_t if

$$\mathbf{f}_t = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \quad \mathbf{i}_t^\dagger = \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \quad \mathbf{i}_t^\ddagger = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix}$$

$$\begin{aligned}
\mathbf{c}_t^\dagger &= \mathbf{c}_{t-1} \odot \mathbf{f}_t \\
&= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \odot \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \\
\mathbf{i}_t &= \mathbf{i}_t^\dagger \odot \mathbf{i}_t^\ddagger \\
&= \begin{bmatrix} +1.00 \\ +1.00 \end{bmatrix} \odot \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} \\
\mathbf{c}_t &= \mathbf{c}_t^\dagger + \mathbf{i}_t \\
&= \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} + \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix}
\end{aligned}$$

(d) What will be the value of \mathbf{c}_t if

$$\mathbf{f}_t = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \quad \mathbf{i}_t^\dagger = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \quad \mathbf{i}_t^\ddagger = \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix}$$

$$\begin{aligned}
\mathbf{c}_t^\dagger &= \mathbf{c}_{t-1} \odot \mathbf{f}_t \\
&= \begin{bmatrix} +0.5 \\ -0.5 \end{bmatrix} \odot \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \\
\mathbf{i}_t &= \mathbf{i}_t^\dagger \odot \mathbf{i}_t^\ddagger \\
&= \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \odot \begin{bmatrix} -0.25 \\ +0.25 \end{bmatrix} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} \\
\mathbf{c}_t &= \mathbf{c}_t^\dagger + \mathbf{i}_t \\
&= \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix}
\end{aligned}$$

- * 13. Equations (8.132)^[519] to (8.138)^[520] step through the calculation of the weight update for $\mathbf{W}^{(f)}$ in the context of the forward pass presented in Figure 8.41^[514] and under the assumption that the following error gradients are already calculated:

$$\frac{\partial \mathcal{E}_{t+1}}{\partial \mathbf{c}_t} = \begin{bmatrix} 0.35 \\ 0.50 \end{bmatrix} \quad \frac{\partial \mathcal{E}_{t+1}}{\partial \mathbf{h}_t} = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix} \quad \frac{\partial \mathcal{E}_t}{\partial \mathbf{o}_t} = \begin{bmatrix} 0.15 \\ 0.60 \end{bmatrix}$$

- (a) Given this context, calculate $\frac{\partial \mathcal{E}_t}{\partial \mathbf{c}_{t-1}}$.

From Equation (8.125)^[518] we know that:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{c}_{t-1}} = \frac{\partial \mathcal{E}}{\partial \mathbf{c}_t} \odot \mathbf{f}_t$$

From Equation (8.135)^[519] we know that:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{c}_t} = \begin{bmatrix} 0.867455753 \\ 0.941532531 \end{bmatrix}$$

From Figure 8.41^[514] we know that:

$$\mathbf{f}_t = \begin{bmatrix} 0.535440568 \\ 0.517492858 \end{bmatrix}$$

And so:

$$\frac{\partial \mathcal{E}}{\partial \mathbf{c}_{t-1}} = \begin{bmatrix} 0.867455753 \\ 0.941532531 \end{bmatrix} \odot \begin{bmatrix} 0.535440568 \\ 0.517492858 \end{bmatrix} = \begin{bmatrix} 0.464470915 \\ 0.48723636 \end{bmatrix}$$

- (b) Given this context, calculate the vector of error gradients with respect to the input $\mathbf{h}\mathbf{x}_t$ for the forget gate sigmoid layer.

To do this we multiply the δ values for the neurons in the layer by the weights the neurons use (see Equation (8.139)^[520]):

$$\mathbf{W}^{(f)\top} \cdot \delta_f = \begin{bmatrix} 0.00 & 0.00 \\ -0.26 & -0.03 \\ +0.12 & +0.08 \\ +0.08 & +0.01 \end{bmatrix} \cdot \begin{bmatrix} +0.064732317 \\ +0.141057014 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.172535735 \\ +0.088963342 \\ +0.053196424 \end{bmatrix}$$

9 Evaluation (Exercise Solutions)

1. The table below shows the predictions made for a categorical target feature by a model for a test dataset. Based on this test set, calculate the evaluation measures listed below.

ID	Target	Prediction	ID	Target	Prediction	ID	Target	Prediction
1	false	false	8	true	true	15	false	false
2	false	false	9	false	false	16	false	false
3	false	false	10	false	false	17	true	false
4	false	false	11	false	false	18	true	true
5	true	true	12	true	true	19	true	true
6	false	false	13	false	false	20	true	true
7	true	true	14	true	true			

- (a) A confusion matrix and the misclassification rate

The confusion matrix can be written as

		Prediction	
		true	false
Target	true	8	1
	false	0	11

Misclassification rate can be calculated as

$$\begin{aligned}
 \text{misclassification rate} &= \frac{(FP + FN)}{(TP + TN + FP + FN)} \\
 &= \frac{(0 + 1)}{(8 + 11 + 0 + 1)} \\
 &= 0.05
 \end{aligned}$$

- (b) The average class accuracy (harmonic mean)

First, we calculate the recall for each target level:

$$recall_{true} = \frac{8}{9} = 0.889$$

$$recall_{false} = \frac{11}{11} = 1.000$$

Then we can calculate a harmonic mean as

$$\begin{aligned} average\ class\ accuracy_{HM} &= \frac{1}{\frac{1}{|levels(t)|} \sum_{l \in levels(t)} \frac{1}{recall_l}} \\ &= \frac{1}{\frac{1}{2} \left(\frac{1}{0.889} + \frac{1}{1} \right)} \\ &= 0.941 \end{aligned}$$

(c) The precision, recall, and F_1 measure

We can calculate precision and recall as follows (assuming that the *true* target level is the positive level):

$$\begin{aligned} precision &= \frac{TP}{(TP + FP)} \\ &= \frac{8}{(8 + 0)} \\ &= 1.000 \end{aligned}$$

$$\begin{aligned} recall &= \frac{TP}{(TP + FN)} \\ &= \frac{8}{(8 + 1)} \\ &= 0.889 \end{aligned}$$

Using these figures, we can calculate the F_1 measure as

$$\begin{aligned} F_1\ measure &= 2 \times \frac{(precision \times recall)}{(precision + recall)} \\ &= 2 \times \frac{(1.000 \times 0.889)}{(1.000 + 0.889)} \\ &= 0.941 \end{aligned}$$

2. The table below shows the predictions made for a continuous target feature by two different prediction models for a test dataset.

ID	Target	Model 1 Prediction	Model 2 Prediction	ID	Target	Model 1 Prediction	Model 2 Prediction
1	2,623	2,664	2,691	16	2,570	2,577	2,612
2	2,423	2,436	2,367	17	2,528	2,510	2,557
3	2,423	2,399	2,412	18	2,342	2,381	2,421
4	2,448	2,447	2,440	19	2,456	2,452	2,393
5	2,762	2,847	2,693	20	2,451	2,437	2,479
6	2,435	2,411	2,493	21	2,296	2,307	2,290
7	2,519	2,516	2,598	22	2,405	2,355	2,490
8	2,772	2,870	2,814	23	2,389	2,418	2,346
9	2,601	2,586	2,583	24	2,629	2,582	2,647
10	2,422	2,414	2,485	25	2,584	2,564	2,546
11	2,349	2,407	2,472	26	2,658	2,662	2,759
12	2,515	2,505	2,584	27	2,482	2,492	2,463
13	2,548	2,581	2,604	28	2,471	2,478	2,403
14	2,281	2,277	2,309	29	2,605	2,620	2,645
15	2,295	2,280	2,296	30	2,442	2,445	2,478

- (a) Based on these predictions, calculate the evaluation measures listed below for each model.

- i. The sum of squared errors

The sum of squared errors for Model 1 can be calculated as follows.

ID	Target	Model 1					
		Pred.	Error	Error ²	Error	SST	SST ²
1	2,623.4	2,664.3	40.9	1,674.2	40.9	173.5	30,089.5
2	2,423.0	2,435.9	12.9	167.4	12.9	-54.9	3,017.9
3	2,423.3	2,398.5	-24.8	615.0	24.8	-92.3	8,528.0
4	2,448.1	2,447.1	-1.1	1.2	1.1	-43.8	1,918.8
5	2,761.7	2,847.3	85.7	7,335.9	85.7	356.4	127,043.9
6	2,434.9	2,411.2	-23.7	560.9	23.7	-79.6	6,341.4
7	2,519.0	2,516.4	-2.6	6.7	2.6	25.5	652.8
8	2,771.6	2,870.2	98.6	9,721.7	98.6	379.4	143,913.2
9	2,601.4	2,585.9	-15.6	242.0	15.6	95.0	9,028.8
10	2,422.3	2,414.2	-8.1	65.0	8.1	-76.7	5,875.6
11	2,348.8	2,406.7	57.9	3,352.0	57.9	-84.1	7,079.6
12	2,514.7	2,505.2	-9.4	89.3	9.4	14.4	206.2
13	2,548.4	2,581.2	32.8	1,075.2	32.8	90.3	8,157.2
14	2,281.4	2,276.9	-4.5	20.4	4.5	-214.0	45,776.8
15	2,295.1	2,279.7	-15.4	238.5	15.4	-211.2	44,597.1
16	2,570.5	2,576.6	6.1	37.2	6.1	85.7	7,346.2
17	2,528.1	2,510.2	-17.9	320.8	17.9	19.4	375.1
18	2,342.2	2,380.9	38.7	1,496.9	38.7	-110.0	12,093.6
19	2,456.0	2,452.1	-3.9	15.1	3.9	-38.8	1,501.8
20	2,451.1	2,436.7	-14.4	208.5	14.4	-54.2	2,934.9
21	2,295.8	2,307.2	11.4	129.8	11.4	-183.7	33,730.7
22	2,405.0	2,354.9	-50.1	2,514.9	50.1	-136.0	18,492.1
23	2,388.9	2,418.1	29.2	853.2	29.2	-72.8	5,297.2
24	2,629.5	2,582.4	-47.1	2,215.7	47.1	91.5	8,380.0
25	2,583.8	2,563.5	-20.3	411.7	20.3	72.7	5,281.6
26	2,658.2	2,662.0	3.9	15.1	3.9	171.2	29,298.7
27	2,482.3	2,491.8	9.4	88.6	9.4	0.9	0.8
28	2,470.8	2,477.7	6.9	47.7	6.9	-13.1	172.6
29	2,604.9	2,619.8	14.9	221.7	14.9	128.9	16,624.4
30	2,441.6	2,444.9	3.3	10.9	3.3	-46.0	2,117.8
Mean	2,490.9	2,497.3	6.5	1,125.1	23.7	6.5	19,529.1
Std Dev	127.0	142.0	33.5	2,204.9	24.1	142.0	34,096.6

$$\begin{aligned}
 \text{sum of squared errors} &= \frac{1}{2} \sum_{i=1}^n (t_i - \mathbb{M}(\mathbf{d}_i))^2 \\
 &= 16,876.6
 \end{aligned}$$

The sum of squared errors for Model 2 can be calculated as follows.

ID	Target	Model 1					
		Prediction	Error	Error ²	Error	SST	SST ²
1	2,623.4	2,690.6	67.2	4,511.2	67.2	199.7	39,884.8
2	2,423.0	2,367.4	-55.6	3,095.8	55.6	-123.5	15,255.8
3	2,423.3	2,412.2	-11.1	123.5	11.1	-78.7	6,187.8
4	2,448.1	2,439.9	-8.3	68.7	8.3	-51.0	2,602.4
5	2,761.7	2,693.1	-68.6	4,704.4	68.6	202.2	40,882.2
6	2,434.9	2,493.0	58.1	3,374.5	58.1	2.1	4.6
7	2,519.0	2,598.1	79.1	6,253.1	79.1	107.2	11,494.6
8	2,771.6	2,813.8	42.2	1,781.3	42.2	323.0	104,307.2
9	2,601.4	2,582.9	-18.5	343.9	18.5	92.0	8,469.5
10	2,422.3	2,485.1	62.8	3,940.2	62.8	-5.8	33.8
11	2,348.8	2,471.7	122.9	15,104.0	122.9	-19.1	366.3
12	2,514.7	2,583.6	68.9	4,749.9	68.9	92.7	8,598.5
13	2,548.4	2,604.0	55.6	3,091.3	55.6	113.1	12,797.6
14	2,281.4	2,309.4	28.0	783.9	28.0	-181.4	32,919.1
15	2,295.1	2,296.0	0.9	0.8	0.9	-194.8	37,962.7
16	2,570.5	2,611.7	41.2	1,697.4	41.2	120.8	14,595.0
17	2,528.1	2,557.1	29.0	839.9	29.0	66.3	4,390.0
18	2,342.2	2,420.5	78.3	6,135.2	78.3	-70.3	4,946.7
19	2,456.0	2,392.5	-63.5	4,027.2	63.5	-98.3	9,669.3
20	2,451.1	2,478.7	27.6	760.6	27.6	-12.2	147.8
21	2,295.8	2,290.0	-5.8	34.1	5.8	-200.9	40,358.2
22	2,405.0	2,490.4	85.4	7,286.1	85.4	-0.5	0.2
23	2,388.9	2,345.5	-43.3	1,878.7	43.3	-145.3	21,122.7
24	2,629.5	2,646.7	17.2	295.6	17.2	155.8	24,275.8
25	2,583.8	2,546.3	-37.6	1,410.9	37.6	55.4	3,069.4
26	2,658.2	2,759.3	101.1	10,227.4	101.1	268.4	72,047.6
27	2,482.3	2,462.8	-19.5	381.8	19.5	-28.1	787.8
28	2,470.8	2,403.4	-67.4	4,542.6	67.4	-87.4	7,645.6
29	2,604.9	2,644.9	40.0	1,601.7	40.0	154.1	23,736.8
30	2,441.6	2,478.0	36.4	1,327.0	36.4	-12.9	166.2
Mean	2,490.9	2,512.3	21.4	3,145.8	48.0	21.4	18,290.9
Std Dev	127.0	135.8	52.7	3,382.5	29.4	135.8	23,625.8

$$\begin{aligned}
 \text{sum of squared errors} &= \frac{1}{2} \sum_{i=1}^n (t_i - \mathbb{M}(\mathbf{d}_i))^2 \\
 &= 47,186.3
 \end{aligned}$$

ii. The R^2 measure

The R^2 measure is calculated as

$$R^2 = 1 - \frac{\text{sum of squared errors}}{\text{total sum of squares}}$$

The sum of squared error values from the previous part can be used. So, for Model 1,

$$\begin{aligned} \text{total sum of squares} &= \frac{1}{2} \sum_{i=1}^n (t_i - \bar{t})^2 \\ &= 292,937.1 \end{aligned}$$

and

$$\begin{aligned} R^2 &= 1 - \frac{16,876.6}{292,937.1} \\ &= 0.942 \end{aligned}$$

For Model 2,

$$\text{total sum of squares} = 274,363.1$$

and

$$\begin{aligned} R^2 &= 1 - \frac{47,186.3}{274,363.1} \\ &= 0.828 \end{aligned}$$

- (b) Based on the evaluation measures calculated, which model do you think is performing better for this dataset?

Model 1 has a higher R^2 value than Model 2, 0.942 compared to 0.828, which indicates that it is better able to capture the pattern in this dataset. An R^2 value this high suggests quite a powerful model.

3. A credit card issuer has built two different credit scoring models that predict the propensity of customers to default on their loans. The outputs of the first model for a test dataset are shown in the table below.

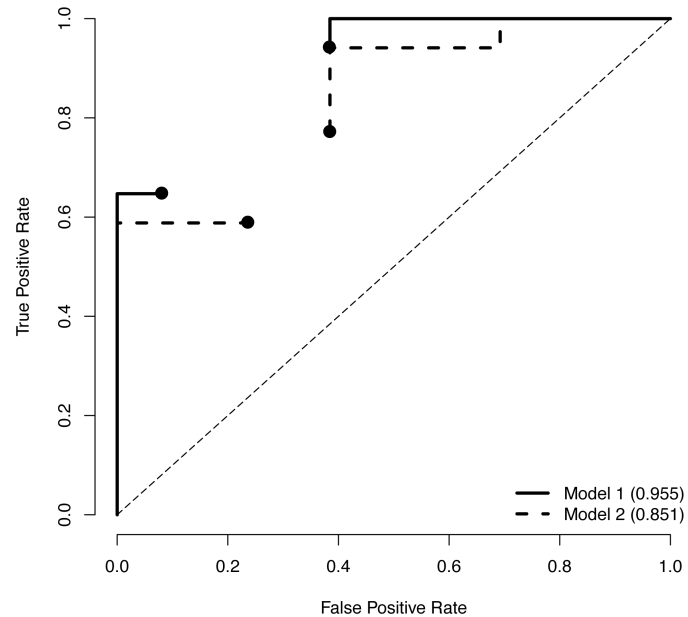
ID	Target	Score	Prediction	ID	Target	Score	Prediction
1	bad	0.634	bad	16	good	0.072	good
2	bad	0.782	bad	17	bad	0.567	bad
3	good	0.464	good	18	bad	0.738	bad
4	bad	0.593	bad	19	bad	0.325	good
5	bad	0.827	bad	20	bad	0.863	bad
6	bad	0.815	bad	21	bad	0.625	bad
7	bad	0.855	bad	22	good	0.119	good
8	good	0.500	good	23	bad	0.995	bad
9	bad	0.600	bad	24	bad	0.958	bad
10	bad	0.803	bad	25	bad	0.726	bad
11	bad	0.976	bad	26	good	0.117	good
12	good	0.504	bad	27	good	0.295	good
13	good	0.303	good	28	good	0.064	good
14	good	0.391	good	29	good	0.141	good
15	good	0.238	good	30	good	0.670	bad

The outputs of the second model for the same test dataset are shown in the table below.

ID	Target	Score	Prediction	ID	Target	Score	Prediction
1	bad	0.230	bad	16	good	0.421	bad
2	bad	0.859	good	17	bad	0.842	good
3	good	0.154	bad	18	bad	0.891	good
4	bad	0.325	bad	19	bad	0.480	bad
5	bad	0.952	good	20	bad	0.340	bad
6	bad	0.900	good	21	bad	0.962	good
7	bad	0.501	good	22	good	0.238	bad
8	good	0.650	good	23	bad	0.362	bad
9	bad	0.940	good	24	bad	0.848	good
10	bad	0.806	good	25	bad	0.915	good
11	bad	0.507	good	26	good	0.096	bad
12	good	0.251	bad	27	good	0.319	bad
13	good	0.597	good	28	good	0.740	good
14	good	0.376	bad	29	good	0.211	bad
15	good	0.285	bad	30	good	0.152	bad

Based on the predictions of these models, perform the following tasks to compare their performance.

- (a) The image below shows an **ROC curve** for each model. Each curve has a point missing.



Calculate the missing point in the ROC curves for Model 1 and Model 2. To generate the point for Model 1, use a threshold value of 0.51. To generate the point for Model 2, use a threshold value of 0.43.

To plot an ROC curve, it is easiest to sort the data according to the prediction scores generated. Based on the threshold being used to find a point for the ROC plot, we can create a new set of predictions from which we will calculate the true positive rate (TPR) and the false positive rate (FPR) that are used to plot the ROC curve. The table below shows the prediction scores for Model 1 in ascending order, the new predictions based on a threshold of 0.51, as well as the outcome of each of these predictions—whether it is a true positive (TP), false positive (FP), true negative (TN), or false negative (FN).

ID	Target	Score	Prediction	Outcome
28	good	0.064	good	TN
16	good	0.072	good	TN
26	good	0.117	good	TN
22	good	0.119	good	TN
29	good	0.141	good	TN
15	good	0.238	good	TN
27	good	0.295	good	TN
13	good	0.303	good	TN
19	bad	0.325	good	FN
14	good	0.391	good	TN
3	good	0.464	good	TN
8	good	0.500	good	TN
12	good	0.504	good	FP
17	bad	0.567	bad	TP
4	bad	0.593	bad	TP
9	bad	0.600	bad	TP
21	bad	0.625	bad	TP
1	bad	0.634	bad	TP
30	good	0.670	bad	FP
25	bad	0.726	bad	TP
18	bad	0.738	bad	TP
2	bad	0.782	bad	TP
10	bad	0.803	bad	TP
6	bad	0.815	bad	TP
5	bad	0.827	bad	TP
7	bad	0.855	bad	TP
20	bad	0.863	bad	TP
24	bad	0.958	bad	TP
11	bad	0.976	bad	TP
23	bad	0.995	bad	TP

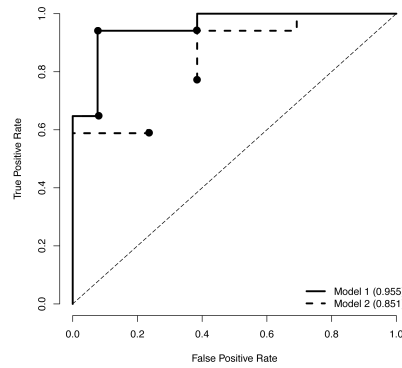
Based on these predictions, we can build a confusion matrix from which we can calculate the true positive rate and false positive rate that we use to plot a point in ROC space. The confusion matrix for Model 1 using a threshold of 0.48 is shown below.

		Prediction	
		<i>bad</i>	<i>good</i>
Target	<i>bad</i>	16	1
	<i>good</i>	1	12

So we can calculate the TPR and FPR as

$$\begin{aligned} TPR &= \frac{16}{16+1} = 0.9412 \\ FPR &= \frac{1}{12+1} = 0.0769 \end{aligned}$$

Using these figures, we can plot an extra point on the ROC curve and connect it to the existing points to complete the curve (other points for other thresholds are required to complete the curve, but they all result in the same TPR score and so a horizontal line).



The table below shows the prediction scores for Model 2 in ascending order, the new predictions based on a threshold of 0.43, as well as the outcome of each of these predictions—whether it is a true positive (TP), false positive (FP), true negative (TN), or false negative (FN).

ID	Target	Score	Prediction	Outcome
26	good	0.096	good	TN
30	good	0.152	good	TN
3	good	0.154	good	TN
29	good	0.211	good	TN
1	bad	0.230	good	FN
22	good	0.238	good	TN
12	good	0.251	good	TN
15	good	0.285	good	TN
27	good	0.319	good	TN
4	bad	0.325	good	FN
20	bad	0.340	good	FN
23	bad	0.362	good	FN
14	good	0.376	good	TN
16	good	0.421	good	TN
19	bad	0.480	bad	TP
7	bad	0.501	bad	TP
11	bad	0.507	bad	TP
13	good	0.597	bad	FP
8	good	0.650	bad	FP
28	good	0.740	bad	FP
10	bad	0.806	bad	TP
17	bad	0.842	bad	TP
24	bad	0.848	bad	TP
2	bad	0.859	bad	TP
18	bad	0.891	bad	TP
6	bad	0.900	bad	TP
25	bad	0.915	bad	TP
9	bad	0.940	bad	TP
5	bad	0.952	bad	TP
21	bad	0.962	bad	TP

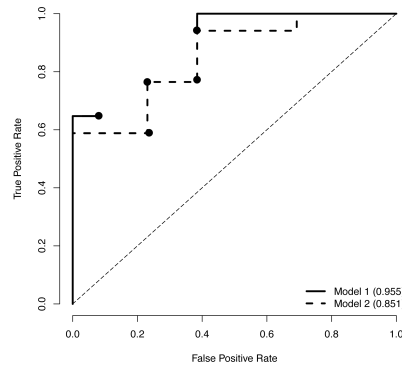
Based on these predictions, we can build a confusion matrix from which we can calculate the true positive rate and false positive rate that we use to plot a point in ROC space. The confusion matrix for Model 2 using a threshold of 0.48 is shown below.

		Prediction	
		<i>bad</i>	<i>good</i>
Target	<i>bad</i>	13	4
	<i>good</i>	3	10

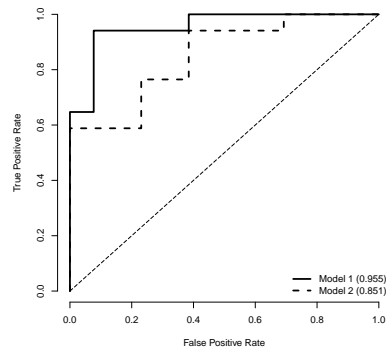
So we can calculate the TPR and FPR as

$$\begin{aligned} TPR &= \frac{13}{13+4} = 0.7647 \\ FPR &= \frac{3}{10+3} = 0.2308 \end{aligned}$$

Using these figures, we can plot an extra point on the ROC curve and connect it to the existing points to complete the curve (other points for other thresholds are required to complete the curve, but they all result in the same TPR score and so a horizontal line).



For completeness, we show both complete curves together below.



- (b) The **area under the ROC curve (AUC)** for Model 1 is 0.955 and for Model 2 is 0.851. Which model is performing best?

Based on the higher AUC, we can conclude that Model 1 is performing better at this task. Furthermore, the ROC curve for Model 1 dominates the curve for Model 2 (i.e., is always higher), which means that there is no operating point (or threshold value) for which Model 2 is better.

- (c) Based on the AUC values for Model 1 and Model 2, calculate the **Gini coefficient** for each model.

The Gini coefficient is calculated as

$$Gini\ coefficient = (2 \times ROC\ index) - 1$$

So for Model 1, the Gini coefficient is

$$Gini\ coefficient = (2 \times 0.955) - 1 = 0.91$$

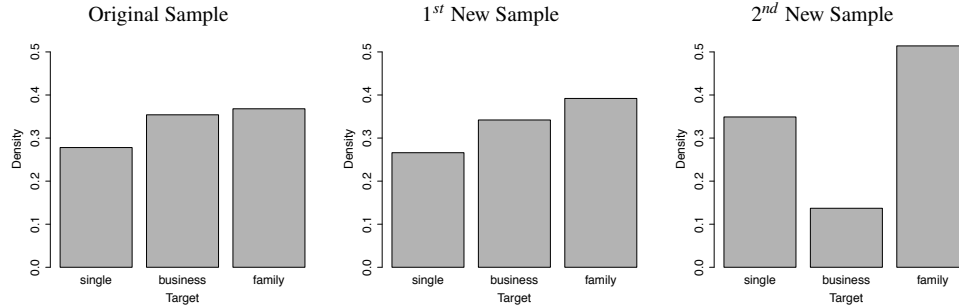
For Model 2, the Gini coefficient is

$$Gini\ coefficient = (2 \times 0.851) - 1 = 0.702$$

4. A retail supermarket chain has built a prediction model that recognizes the household that a customer comes from as being one of *single*, *business*, or *family*. After deployment, the analytics team at the supermarket chain uses the **stability index** to monitor the performance of this model. The table below shows the frequencies of predictions of the three different levels made by the model for the original validation dataset at the time the model was built, for the month after deployment, and for a monthlong period six months after deployment.

Target	Original Sample	1 st New Sample	2 nd New Sample
<i>single</i>	123	252	561
<i>business</i>	157	324	221
<i>family</i>	163	372	827

Bar plots of these three sets of prediction frequencies are shown in the following images.



Calculate the **stability index** for the two new periods and determine whether the model should be retrained at either of these points.

The stability index is calculated as

$$stability\ index = \sum_{l \in levels} \left(\left(\frac{|\mathcal{A}_{t=l}|}{|\mathcal{A}|} - \frac{|\mathcal{B}_{t=l}|}{|\mathcal{B}|} \right) \times \log_e \left(\frac{|\mathcal{A}_{t=l}|}{|\mathcal{A}|} / \frac{|\mathcal{B}_{t=l}|}{|\mathcal{B}|} \right) \right)$$

where l is a target level, $|\mathcal{A}|$ refers to the size of the test set on which performance measures were originally calculated, $|\mathcal{A}_{t=l}|$ refers to the number of instances in the original test set for which the model made a prediction of level l for target t , $|\mathcal{B}|$ and $|\mathcal{B}_{t=l}|$ refer to the same measurements on the newly collected dataset. The following table shows the components of calculating this for the two new periods.

Target	Original		1 st New Sample			2 nd New Sample		
	Count	%	Count	%	SI _t	Count	%	SI _t
single	123	0.2777	252	0.2658	0.00052	561	0.3487	0.01617
business	157	0.3544	324	0.3418	0.00046	221	0.1374	0.20574
family	163	0.3679	372	0.3924	0.00157	827	0.5140	0.04881
Sum	443		948		0.003	1,609		0.271

For the first new sample, the stability index is 0.003, which indicates that there is practically no difference between the distribution of target levels predicted for the original validation dataset and for the data in the newt period.

For the second sample, the stability index is 0.271, which indicates a massive difference between the distribution of target levels at this point in time compared to the distribution predicted for the original validation set. This suggests that concept drift has occurred and that the model should be retrained.

- * 5. Explain the problem associated with measuring the performance of a predictive model using a single accuracy figure.

A single accuracy figure can hide the real performance of a model. This is particularly evident when we are dealing with imbalanced test datasets. Consider a test dataset that contains 1,000 instances overall, 900 of which belong to the positive target level, and 100 of which belong to the negative target level.

Assuming that the model always predicts the positive level, then its accuracy on the given test dataset would be 90%, which is not at all an accurate reflection of the performance of the model. Generating a simple confusion matrix for this scenario shows how poorly the model is really performing.

		Prediction	
		<i>pos</i>	<i>neg</i>
Target	<i>pos</i>	900	0
	<i>neg</i>	100	0

While measures such as average class accuracy or the F_1 measure attempt to address the issues associated with a simple accuracy measure, no single measure is perfect. All single measures compress the actual information in a set of results from a test dataset in some way, so it is always a good idea to use multiple performance measures as part of an evaluation, even though ultimately we will use a single figure to choose between alternative models.

- * 6. A marketing company working for a charity has developed two different models that predict the likelihood that donors will respond to a mailshot asking them to make a special extra donation. The prediction scores generated for a test set for these two models are shown in the table below.

		Model 1	Model 2			Model 1	Model 2
ID	Target	Score	Score	ID	Target	Score	Score
1	false	0.1026	0.2089	16	true	0.7165	0.4569
2	false	0.2937	0.0080	17	true	0.7677	0.8086
3	true	0.5120	0.8378	18	false	0.4468	0.1458
4	true	0.8645	0.7160	19	false	0.2176	0.5809
5	false	0.1987	0.1891	20	false	0.9800	0.5783
6	true	0.7600	0.9398	21	true	0.6562	0.7843
7	true	0.7519	0.9800	22	true	0.9693	0.9521
8	true	0.2994	0.8578	23	false	0.0275	0.0377
9	false	0.0552	0.1560	24	true	0.7047	0.4708
10	false	0.9231	0.5600	25	false	0.3711	0.2846
11	true	0.7563	0.9062	26	false	0.4440	0.1100
12	true	0.5664	0.7301	27	true	0.5440	0.3562
13	true	0.2872	0.8764	28	true	0.5713	0.9200
14	true	0.9326	0.9274	29	false	0.3757	0.0895
15	false	0.0651	0.2992	30	true	0.8224	0.8614

- (a) Using a classification threshold of 0.5, and assuming that *true* is the positive target level, construct a **confusion matrix** for each of the models.

The first step to answering this is to apply the threshold to determine whether predictions are correct or not. For Model 1, the predictions and outcomes are as follows

ID	Target	Model 1 Score	Model 1 Prediction	Model 1 Outcome
1	false	0.1026	false	TN
2	false	0.2937	false	TN
3	true	0.5120	true	TP
4	true	0.8645	true	TP
5	false	0.1987	false	TN
6	true	0.7600	true	TP
7	true	0.7519	true	TP
8	true	0.2994	false	FN
9	false	0.0552	false	TN
10	false	0.9231	true	FP
11	true	0.7563	true	TP
12	true	0.5664	true	TP
13	true	0.2872	false	FN
14	true	0.9326	true	TP
15	false	0.0651	false	TN
16	true	0.7165	true	TP
17	true	0.7677	true	TP
18	false	0.4468	false	TN
19	false	0.2176	false	TN
20	false	0.9800	true	FP
21	true	0.6562	true	TP
22	true	0.9693	true	TP
23	false	0.0275	false	TN
24	true	0.7047	true	TP
25	false	0.3711	false	TN
26	false	0.4440	false	TN
27	true	0.5440	true	TP
28	true	0.5713	true	TP
29	false	0.3757	false	TN
30	true	0.8224	true	TP

Using this, we can build the following confusion matrix for Model 1

		Prediction	
		<i>true</i>	<i>false</i>
Target	<i>true</i>	15	2
	<i>false</i>	2	11

We then repeat the same process for Model 2.

ID	Target	Model 2 Score	Model 2 Prediction	Model 2 Outcome
1	false	0.2089	false	TN
2	false	0.0080	false	TN
3	true	0.8378	true	TP
4	true	0.7160	true	TP
5	false	0.1891	false	TN
6	true	0.9398	true	TP
7	true	0.9800	true	TP
8	true	0.8578	true	TP
9	false	0.1560	false	TN
10	false	0.5600	true	FP
11	true	0.9062	true	TP
12	true	0.7301	true	TP
13	true	0.8764	true	TP
14	true	0.9274	true	TP
15	false	0.2992	false	TN
16	true	0.4569	false	FN
17	true	0.8086	true	TP
18	false	0.1458	false	TN
19	false	0.5809	true	FP
20	false	0.5783	true	FP
21	true	0.7843	true	TP
22	true	0.9521	true	TP
23	false	0.0377	false	TN
24	true	0.4708	false	FN
25	false	0.2846	false	TN
26	false	0.1100	false	TN
27	true	0.3562	false	FN
28	true	0.9200	true	TP
29	false	0.0895	false	TN
30	true	0.8614	true	TP

Using this we can build the following confusion matrix for Model 2.

		Prediction	
		<i>true</i>	<i>false</i>
Target	<i>true</i>	14	3
	<i>false</i>	3	10

- (b) Calculate the simple accuracy and **average class accuracy** (using an **arithmetic mean**) for each model.

We can calculate average class accuracy simply from the confusion matrices. For Model 1 we calculate the recall for the *true* target level as

$$recall_{true} = \frac{14}{(14 + 3)} = 0.8235$$

For the *false* target level, recall is

$$recall_{false} = \frac{12}{(12 + 1)} = 0.9231$$

We can then calculate average class accuracy as

$$average\ class\ accuracy_{AM} = \frac{(0.8235 + 0.9231)}{2} = 0.8733$$

For Model 2 we calculate the recalls for the two target levels as

$$recall_{true} = \frac{14}{(14 + 3)} = 0.8235$$

and

$$recall_{false} = \frac{10}{(10 + 3)} = 0.7692$$

We can then calculate average class accuracy as

$$average\ class\ accuracy_{AM} = \frac{(0.8235 + 0.7692)}{2} = 0.7964$$

- (c) Based on the average class accuracy measures, which model appears to perform best at this task?

Based on average class accuracy, Model 1 appears to be doing a better job than Model 2. The main difference is in the better ability of Model 1 to identify instances of the *false* target level.

- (d) Generate a **cumulative gain chart** for each model.

To generate a cumulative gain chart, we first have to reorder the predictions in descending order of scores and divide the test instances into deciles. The table below shows this for Model 1 (because there are 30 instances in the test set, there are 3 instances per decile).

Decile	ID	Target	Model 1 Prediction	Model 1 Score	Model 1 Outcome
1 st	20	false	0.9800	true	FP
	22	true	0.9693	true	TP
	14	true	0.9326	true	TP
2 nd	10	false	0.9231	true	FP
	4	true	0.8645	true	TP
	30	true	0.8224	true	TP
3 rd	17	true	0.7677	true	TP
	6	true	0.7600	true	TP
	11	true	0.7563	true	TP
4 th	7	true	0.7519	true	TP
	16	true	0.7165	true	TP
	24	true	0.7047	true	TP
5 th	21	true	0.6562	true	TP
	28	true	0.5713	true	TP
	12	true	0.5664	true	TP
6 th	27	true	0.5440	true	TP
	3	true	0.5120	true	TP
	18	false	0.4468	false	TN
7 th	26	false	0.4440	false	TN
	29	false	0.3757	false	TN
	25	false	0.3711	false	TN
8 th	8	true	0.2994	false	FN
	2	false	0.2937	false	TN
	13	true	0.2872	false	FN
9 th	19	false	0.2176	false	TN
	5	false	0.1987	false	TN
	1	false	0.1026	false	TN
10 th	15	false	0.0651	false	TN
	9	false	0.0552	false	TN
	23	false	0.0275	false	TN

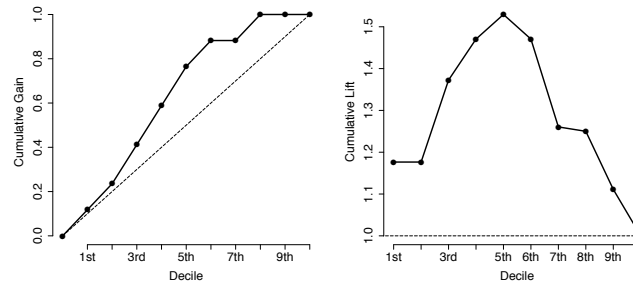
Based on this table, we can calculate the gain and lift (lift is not required to answer the question but is included for completeness) for each decile as well as the cumulative gain and cumulative lift for each decile. Recall that the definitions for gain, cumulative gain, lift, and cumulative lift always refer to the actual target levels rather than the predictions and are defined as follows:

$$\begin{aligned}
 \text{gain}(dec) &= \frac{\text{num positive test instances in decile } dec}{\text{num positive test instances}} \\
 \text{cumulative gain}(dec) &= \frac{\text{num positive test instances in all deciles up to } dec}{\text{num positive test instances}} \\
 \text{lift}(dec) &= \frac{\% \text{ of positive test instances in decile } dec}{\% \text{ of positive test instances}} \\
 \text{cumulative lift}(dec) &= \frac{\% \text{ of positive instances in all deciles up to } dec}{\% \text{ of positive test instances}}
 \end{aligned}$$

The measures for each decile for Model 1 are shown in the table below.

Decile	Positive (true) Count	Negative (false) Count	Gain	Cum. Gain	Lift	Cum. Lift
1 st	2	1	0.1176	0.1176	1.1765	1.1765
2 nd	2	1	0.1176	0.2353	1.1765	1.1765
3 rd	3	0	0.1765	0.4118	1.7647	1.3725
4 th	3	0	0.1765	0.5882	1.7647	1.4706
5 th	3	0	0.1765	0.7647	1.7647	1.5294
6 th	0	3	0.0000	0.7647	0.0000	1.2745
7 th	1	2	0.0588	0.8235	0.5882	1.1765
8 th	1	2	0.0588	0.8824	0.5882	1.1029
9 th	1	2	0.0588	0.9412	0.5882	1.0458
10 th	0	3	0.0000	0.9412	0.0000	0.9412

Using this table, we can now draw the required cumulative gain chart as follows (the cumulative lift chart is also shown).



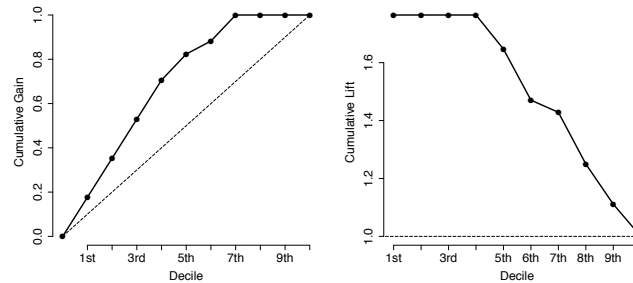
We then repeat for Model 2.

Decile	ID	Target	Model 1 Prediction	Model 1 Score	Model 1 Outcome
1 st	7	true	0.9800	true	TP
	22	true	0.9521	true	TP
	6	true	0.9398	true	TP
2 nd	14	true	0.9274	true	TP
	28	true	0.9200	true	TP
	11	true	0.9062	true	TP
3 rd	13	true	0.8764	true	TP
	30	true	0.8614	true	TP
	8	true	0.8578	true	TP
4 th	3	true	0.8378	true	TP
	17	true	0.8086	true	TP
	21	true	0.7843	true	TP
5 th	12	true	0.7301	true	TP
	4	true	0.7160	true	TP
	19	false	0.5809	true	FP
6 th	20	false	0.5783	true	FP
	10	false	0.5600	true	FP
	24	true	0.4708	false	FN
7 th	16	true	0.4569	false	FN
	27	true	0.3562	false	FN
	15	false	0.2992	false	TN
8 th	25	false	0.2846	false	TN
	1	false	0.2089	false	TN
	5	false	0.1891	false	TN
9 th	9	false	0.1560	false	TN
	18	false	0.1458	false	TN
	26	false	0.1100	false	TN
10 th	29	false	0.0895	false	TN
	23	false	0.0377	false	TN
	2	false	0.0080	false	TN

We can now calculate gain, cumulative gain, lift, and cumulative lift for each decile for Model 2.

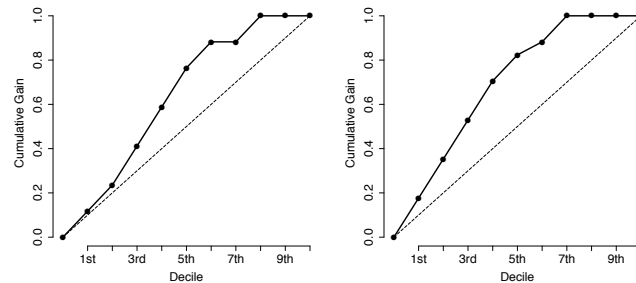
Decile	Positive (true) Count	Negative (false) Count	Gain	Cum. Gain	Lift	Cum. Lift
1 st	3	0	0.1765	0.1765	1.7647	1.7647
2 nd	2	1	0.1176	0.2941	1.1765	1.4706
3 rd	3	0	0.1765	0.4706	1.7647	1.5686
4 th	3	0	0.1765	0.6471	1.7647	1.6176
5 th	3	0	0.1765	0.8235	1.7647	1.6471
6 th	0	3	0.0000	0.8235	0.0000	1.3725
7 th	1	2	0.0588	0.8824	0.5882	1.2605
8 th	1	2	0.0588	0.9412	0.5882	1.1765
9 th	1	2	0.0588	1.0000	0.5882	1.1111
10 th	0	3	0.0000	1.0000	0.0000	1.0000

Using this table, we can now draw the required cumulative gain chart as follows (the cumulative lift chart is also shown).



- (e) The charity for which the model is being built typically has only enough money to send a mailshot to the top 20% of its contact list. Based on the cumulative gain chart generated in the previous part, would you recommend that Model 1 or Model 2 would perform best for the charity?

The cumulative gain charts are repeated here.



The key thing to notice here is that at the 2nd decile, Model 2 has identified almost 40% of the positive instances in the dataset, whereas Model 1 has identified only just over 20%. In a scenario like the one in this question, the typical approach would be to present every contact in the contact list to a model and then rank the contacts by the output of the model. Mailshots would be sent to the top 20% of this list, or whatever percentage the people sending the mailshot could afford. Because of this, a model that ranks well should be preferred over one that simply makes correct predictions. So in this case, even though Model 2 performs worse than Model 1 based on average class accuracy, Model 2 should be preferred for this task. The issue for Model 1 arises because it gives very high prediction scores to test instances 10 and 20.

- * 7. A prediction model is going to be built for in-line quality assurance in a factory that manufactures electronic components for the automotive industry. The system will be integrated into the factory's production line and determine whether components are of an acceptable quality standard based on a set of test results. The prediction subject is a component, and the descriptive features are a set of characteristics of the component that can be gathered on the production line. The target feature is binary and labels components as *good* or *bad*.

It is extremely important that the system not in any way slow the production line and that the possibility of defective components being passed by the system be minimized as much as possible. Furthermore, when the system makes a mistake, it is desirable that the system can be retrained immediately using the instance that generated the mistake. When mistakes are made, it would be useful for the production line operators to be able to query the model to understand why it made the prediction that led to a mistake. A large set of historical labeled data is available for training the system.

- (a) Discuss the different issues that should be taken into account when evaluating the suitability of different machine learning approaches for use in this system.

For this system, classification accuracy will obviously be important. Beyond simple accuracy, however, there is also the issue of a balance between the types of errors that the system will make, that is, false positives and false negatives. For this scenario, if we assume that *good* is the positive target level, then we should avoid false positives as much as possible.

Beyond performance of the algorithm, other considerations are important for this scenario. The first of these is prediction speed. The model will make predictions as part of a production line, which it cannot in any way slow down. Depending on the throughput speed of the production line, the model will likely need to make predictions very quickly.

The next issue that applies in this scenario is the ability to retrain the model. This ability should be taken into account in evaluating the performance of models for this task. There are two ways of thinking about this. The first is the ability to easily modify a model as new data becomes available. The second is the speed at which a model can be completely rebuilt if it cannot be easily modified.

Finally, explanation capability is another important factor to consider when evaluating models. Once the model gives a prediction, how easy is it for someone to understand this prediction? This is key in this scenario so that production line operators can understand why mistakes are being made.

- (b) For this task, discuss the suitability of the decision tree, k nearest neighbor, naive Bayes, and logistic regression models. Which one do you think would be most appropriate?

It is not possible to say anything about the likely performance in terms of prediction accuracy of the three model types for this problem without knowing much more about the data involved, and probably actually performing experiments. We can, however, comment on the model types in terms of the other characteristics: prediction speed, capacity for retraining, training speed, and explanation capability.

For k -NN models, prediction speed is always an issue. Although techniques such as k - d trees can help in reducing the time that it takes to find the nearest neighbors for a model, k -NN models will take more time than the other approaches to make a prediction. This can make them unsuitable for high-throughput scenarios like the production line. In terms of retraining, k -NN models excel. To allow the model to take new data into account, we simply add the new instances to the data used. Similarly, full retraining is almost

instant—the training dataset used by the model is simply replaced with a new one—and k -NN models provide some ability to explain the predictions that they make—the neighbors on which a prediction is based can be provided for explanation. Taking these characteristics together, a k -NN model is a reasonably good fit for this task. The main advantage is the ease with which a k -NN model can be retrained. The main disadvantage of using a k -NN model for this task is that prediction time may be too slow for integration into the production line.

A decision tree model would be a good candidate for this scenario. A decision tree can make predictions very quickly. Decision trees also have excellent explanation capacity—by tracing the path through the decision tree, it is very easy to understand how a prediction has been made. The main disadvantage of using a decision tree for this problem is its lack of capacity for retraining. There are techniques that can be used to modify a decision tree based on some newly available data, but this remains an open research problem. On the positive side, though, decision tree training is quite fast, so if a model needs to be completely rebuilt, this does not take as long as it can for other model types—for example, regression models.

A naive Bayes model can make predictions very quickly—this simply involves evaluating the naive Bayes equation—so prediction speed is not likely to be an issue. Like decision trees, there are approaches to adapting naive Bayes models to take into account new data, but there is not a standard, widely used approach. Training time, however, is not excessive, so retraining a naive Bayes model is not a significant problem. The explanation capacity of naive Bayes models is only modest. While the conditional and prior probabilities used to make a prediction can be presented, interpreting these is not something that people find intuitively easy. For this scenario, having probabilities returned would, however, be an advantage, as it makes it very easy to tune a model to favor false precision over recall.

Finally, a logistic regression model would have no problems with prediction speed for this scenario—evaluating the regression equation is not a significant computational task. Retraining, however, is a problem. It is not easy to adapt an existing logistic regression model to take into account new data, and logistic regression models can take a very long time to train—the longest time of the four modeling approaches considered here. Logistic regression models have some explanation capability. By considering model weights together with descriptive feature values, it is possible to get a good understanding of what has contributed to a particular prediction. In some industries

score cards are used to aid explanation (? provides a good overview of the use of score cards for financial credit risk scoring).

Taking the four approaches together, decision trees for the explanation capability and k -NN models for their capacity for retraining are probably the most attractive for this problem. Given the importance of prediction speed in this scenario, decision trees would probably be slightly more suitable.

It is important to reiterate, however, that this discussion has not taken into account the ability of these different model types to cope with the actual data in this problem.

- * 8. The following matrices list the confusions matrices for two models and the **profit matrix** for this prediction task. Calculate the overall profit for each model.

$$\mathbb{M}_1 = \begin{bmatrix} 50 & 10 \\ 20 & 80 \end{bmatrix} \quad \mathbb{M}_2 = \begin{bmatrix} 35 & 25 \\ 40 & 60 \end{bmatrix} \quad Profit = \begin{bmatrix} +100 & -20 \\ -110 & +10 \end{bmatrix}$$

Multiplying each element in the confusion matrices by the corresponding element in the profit matrix give us the following:

$$\mathbb{M}_1 = \begin{bmatrix} 5000 & -200 \\ -2200 & 800 \end{bmatrix}$$

$$\mathbb{M}_2 = \begin{bmatrix} 3500 & -500 \\ -4400 & 600 \end{bmatrix}$$

Summing each of these matrices gives us the profit for each model:

$$\mathbb{M}_1 = +3400$$

$$\mathbb{M}_2 = -800$$

- * 9. The following table lists the scores returned by a prediction model for a test set of 12 examples. The prediction task is a binary classification task, and the instances in the test set are labeled as belonging to the *positive* or *negative* class. For ease of reading, the instances have been ordered in descending order of the score the model assigned to each instance.

ID	Target	Score
1	positive	0.75
2	positive	0.72
3	positive	0.64
4	negative	0.62
5	negative	0.55
6	positive	0.48
7	negative	0.45
8	negative	0.44
9	negative	0.38
10	negative	0.35
11	negative	0.32
12	negative	0.31

- (a) Calculate the **ROC index** for this model using the **trapezoidal method** and the following set of thresholds: 1.0, 0.5, and 0.0.

ID	Target	Score	Predictions by Threshold		
			T= 1.0	T= 0.5	T= 0.0
1	positive	0.75	negative	positive	positive
2	positive	0.72	negative	positive	positive
3	positive	0.64	negative	positive	positive
4	negative	0.62	negative	positive	positive
5	negative	0.55	negative	positive	positive
6	positive	0.48	negative	negative	positive
7	negative	0.45	negative	negative	positive
8	negative	0.44	negative	negative	positive
9	negative	0.38	negative	negative	positive
10	negative	0.35	negative	negative	positive
11	negative	0.32	negative	negative	positive
12	negative	0.31	negative	negative	positive

The true positive rate and false positive rate for threshold 1 (1.0) are:

$$TPR = \frac{0}{4} = 0$$

$$FPR = \frac{0}{8} = 0$$

The true positive rate and false positive rate for threshold 2 (0.5) are:

$$TPR = \frac{3}{4} = 0.75$$

$$FPR = \frac{2}{8} = 0.25$$

The true positive rate and false positive rate for threshold 3 (0.0) are:

$$TPR = \frac{4}{4} = 1.00$$

$$FPR = \frac{8}{8} = 1.00$$

The ROC index is then calculated as:

$$\begin{aligned} ROC\ index &= \frac{(0.25 - 0) \times (0.75 + 0)}{2} + \frac{(1.00 - 0.25) \times (1.00 + 0.75)}{2} \\ &= 0.09375 + 0.65625 \\ &= 0.75 \end{aligned}$$

- (b) The following table lists the scores returned by the same prediction model for a new test set of 12 examples. Again, the prediction task is a binary classification task, and the instances in the test set are labeled as belonging to the *positive* or *negative* class. For ease of reading, the instances have been ordered in descending order of the score the model assigned to each instance. Calculate the ROC index for this model using the **trapezoidal method** and the following set of thresholds: 1.0, 0.5, and 0.0.

ID	Target	Score
1	positive	0.71
2	positive	0.70
3	positive	0.66
4	positive	0.65
5	positive	0.62
6	positive	0.60
7	negative	0.58
8	positive	0.48
9	positive	0.34
10	negative	0.30
11	negative	0.28
12	negative	0.25

ID	Target	Score	Predictions by Threshold		
			T= 1.0	T= 0.5	T= 0.0
1	positive	0.71	negative	positive	positive
2	positive	0.70	negative	positive	positive
3	positive	0.66	negative	positive	positive
4	positive	0.65	negative	positive	positive
5	positive	0.62	negative	positive	positive
6	positive	0.60	negative	positive	positive
7	negative	0.58	negative	positive	positive
8	positive	0.48	negative	negative	positive
9	positive	0.34	negative	negative	positive
10	negative	0.30	negative	negative	positive
11	negative	0.28	negative	negative	positive
12	negative	0.25	negative	negative	positive

The true positive rate and false positive rate for threshold 1 (1.0) are:

$$TPR = \frac{0}{8} = 0$$

$$FPR = \frac{0}{4} = 0$$

The true positive rate and false positive rate for threshold 2 (0.5) are:

$$TPR = \frac{6}{8} = 0.75$$

$$FPR = \frac{1}{4} = 0.25$$

The true positive rate and false positive rate for threshold 3 (0.0) are:

$$TPR = \frac{8}{8} = 1.00$$

$$FPR = \frac{4}{4} = 1.00$$

The ROC index is then calculated as:

$$\begin{aligned} \text{ROC index} &= \frac{(0.25 - 0) \times (0.75 + 0)}{2} + \frac{(1.00 - 0.25) \times (1.00 + 0.75)}{2} \\ &= 0.09375 + 0.65625 \\ &= 0.75 \end{aligned}$$

- (c) The ROC index is insensitive to changes in class distribution within the test set. This means that if the proportion of positive to negative instances changes in a test set, the ROC index will remain the same if the performance of the models on each class is constant. Consequently, the ROC index is robust to class imbalance or skew in the test set. Why do you think this is the case?¹

In a binary confusion matrix the proportion of positive and negative instances is represented by the ratio between the first row (containing the true positive and false negatives counts) to the second row (containing the false positives and true negative counts).

The ROC index is based on the true positive rate (TPR) and the false positive rate (FPR), and each of these rates is calculated as a ratio within a single class: calculating TPR involves dividing the number of true positives by the total number of positive instances in the test set, and calculating FPR involves dividing the number of false positive by the total number of negative instances in the test set. Consequently, if the performance of the model on both the positive and negative classes remain constant (in terms of the ratio of positive examples it get correct to the total number of positive examples, and the ratio of the negative examples it get wrong to the total number of negative examples) then the ROC index will remain constant.

We can see this if we compare the ROC indices we calculated in the first two parts of this question. The ROC indices are the same for this model on both of these test sets even though in the first test set the ratio of positive to negative instances was 4:8 and in the second test set the ratio was 8:4.

1. We recommend ? as an excellent introduction and overview to ROC analysis that covers the topic of imbalance in the test set. Note also that in very highly imbalanced data where there is a very large number of negative examples, the false positive rate is not very sensitive to changes in the number of false positives (because the denominator is so large) and in these contexts, if the focus of the model is on detecting the positive class, it is probably advisable to use precision as the evaluation metric, as it focuses more on the ability to detect the positive class, rather than on the ability to distinguish between classes (which the ROC index captures).

This is because the model's TPR and FPR was constant across both test sets (TPR=0.75 and FPR=0.25).

This contrasts with other metrics such as accuracy, precision and F-score which use counts from both classes in their calculation. For example, calculating precision involves the true positive count (which is a count of a portion of the set of positive instances in the test set) and the false positive count (which is a count of the portion of negative instances in the test set). In fact, if we calculate the F_1 for this model on both of these test sets assuming a threshold of 0.5 we see that the F_1 measure for the same model is different on both these datasets even though its TPR and FPR is consistent across the test sets. This shows that the F_1 is sensitive to class distribution in the test set. The calculations are as follows:

F_1 for test set 1:

$$\begin{aligned} F_1 &= 2 \times \frac{\frac{3}{3+2} \times \frac{3}{3+1}}{\frac{3}{3+2} + \frac{3}{3+1}} \\ &= 2 \times \frac{0.6 \times 0.75}{0.6 + 0.75} \\ &= 0.6667 \end{aligned}$$

F_1 for test set 2:

$$\begin{aligned} F_1 &= 2 \times \frac{\frac{6}{6+1} \times \frac{6}{6+2}}{\frac{3}{3+2} + \frac{3}{3+1}} \\ &= 2 \times \frac{0.857142857 \times 0.75}{0.857142857 + 0.75} \\ &= 0.8 \end{aligned}$$

- * 10. As part of a natural language processing project, a company is creating a dictionary of idiomatic phrases.² The company has used an automatic process to extract a set of 50,000 candidate idioms from a large corpus and now are planning to use a machine learning model to filter this set of candidates before presenting them to a human annotator who decides whether a candidate phrase should be added to the dictionary or not. In order to evaluate which machine learning model to use as the pre-annotator

2. This question is inspired by the work reported in ?.

filter, the company created a test set of 10 phrases extracted at random from the set of 50,000 candidates.

- (a) The following table presents the scoring by two models of the test set of candidate idioms. Which model would be chosen to filter candidate idioms if the decision were taken on the basis of the F_1 score for each model, assuming both models use a threshold of > 0.5 for classifying a candidate as an idiom.

ID	Idiom	Model Scoring	
		M_1	M_2
1	true	0.70	0.80
2	true	0.56	0.80
3	true	0.55	0.70
4	true	0.54	0.45
5	true	0.45	0.44
6	false	0.73	0.55
7	false	0.72	0.54
8	false	0.35	0.40
9	false	0.34	0.38
10	false	0.33	0.30

Applying a threshold of 0.5 to the model scores gives us the following set of predictions for the models:

ID	Idiom	Model Scoring	
		M_1	M_2
1	true	true	true
2	true	true	true
3	true	true	true
4	true	true	false
5	true	false	false
6	false	true	true
7	false	true	true
8	false	false	false
9	false	false	false
10	false	false	false

Based on these predictions the confusion matrix, precision, recall and F_1 for \mathbb{M}_1 is:

$$\mathbb{M}_1 = \begin{bmatrix} 4 & 1 \\ 2 & 3 \end{bmatrix}$$

$$precision = \frac{4}{4+2} = 0.6667$$

$$recall = \frac{4}{4+1} = 0.8$$

$$F_1 = 2 \times \frac{0.6667 \times 0.8}{0.6667 + 0.8} = 0.727272727$$

And, the confusion matrix, precision, recall and F_1 for \mathbb{M}_2 is:

$$\mathbb{M}_2 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$$

$$precision = \frac{3}{3+2} = 0.6$$

$$recall = \frac{3}{3+2} = 0.6$$

$$F_1 = 2 \times \frac{0.6 \times 0.6}{0.6 + 0.6} = 0.6$$

Based on the F_1 score \mathbb{M}_1 is the better option.

- (b) There is a cost associated with each item presented to the human annotator, and the company wants to maximize the number of items that end up in the dictionary. The company estimates that it has an annotation budget that will cover the human annotation of 20,000 phrases (i.e., 40% of the set of candidate phrases). Calculate the **cumulative gain** of each of the models for the 4th decile. Again, assume both models use a threshold of > 0.5 for the idiom class. Finally, on the basis of cumulative gain scores, which model would you recommend the company use for the pre-annotation filtering task?

ID	Idiom	\mathbb{M}_1	\mathbb{M}_1	Outcome	Decile
		Score	Prediction		
6	false	0.73	true	FP	1 st
7	false	0.72	true	FP	2 nd
1	true	0.70	true	TP	3 rd
2	true	0.56	true	TP	4 th
3	true	0.55	true	TP	5 th
4	true	0.54	true	TP	6 th
5	true	0.45	false	FN	7 th
8	false	0.35	false	TN	8 th
9	false	0.34	false	TN	9 th
10	false	0.33	false	TN	10 th

The number of positive instances in the first four deciles for \mathbb{M}_1 is 2, and the number of positive instances overall is 5, so the cumulative gain for \mathbb{M}_1 for the 4th decile is:

$$\text{cumulative gain}(4th)_{\mathbb{M}_1} = \frac{2}{5} = 0.4$$

ID	Idiom	\mathbb{M}_1	\mathbb{M}_1	Outcome	Decile
		Score	Prediction		
1	true	0.80	true	TP	1 st
2	true	0.80	true	TP	2 nd
3	true	0.70	true	TP	3 rd
6	false	0.55	true	FP	4 th
7	false	0.54	true	FP	5 th
4	true	0.45	false	FN	6 th
5	true	0.44	false	FN	7 th
8	false	0.40	false	TN	8 th
9	false	0.38	false	TN	9 th
10	false	0.30	false	TN	10 th

The number of positive instances in the first four deciles for \mathbb{M}_2 is 3, and the number of positive instances overall is 5, so the cumulative gain for \mathbb{M}_3 for the 4th decile is:

$$\text{cumulative gain}(4th)_{\mathbb{M}_2} = \frac{3}{5} = 0.6$$

Comparing the cumulative gains for the two models under the annotation budget constraints for the project, \mathbb{M}_2 is the better model to use as it is likely to present more true positive instances to the annotator within the first four

deciles, as sorted by the models scores, and ultimately resulting in more instances in the idiom dictionary.

III BEYOND PREDICTION

10 Beyond Prediction: Unsupervised Learning (Exercise Solutions)

1. The following table shows a small dataset in which each instance describes measurements taken using three sensors when a valve in an oil well was opened. The three descriptive features, PRESSURE, TEMPERATURE, and VOLUME measure characteristics of the oil flowing through the valve when it was opened. The ***k*-means clustering** approach is to be applied to this dataset with $k = 3$ and using **Euclidean distance**. The initial cluster centroids for the three clusters C_1 , C_2 , and C_3 are $\mathbf{c}_1 = \langle -0.929, -1.040, -0.831 \rangle$, $\mathbf{c}_2 = \langle -0.329, -1.099, 0.377 \rangle$, and $\mathbf{c}_3 = \langle -0.672, -0.505, 0.110 \rangle$. The following table also shows the distance to these three cluster centers for each instance in the dataset.

ID	PRESSURE	TEMPERATURE	VOLUME	Cluster Distances Iter. 1		
				$Dist(\mathbf{d}_i, \mathbf{c}_1)$	$Dist(\mathbf{d}_i, \mathbf{c}_2)$	$Dist(\mathbf{d}_i, \mathbf{c}_3)$
1	-0.392	-1.258	-0.666	0.603	1.057	1.117
2	-0.251	-1.781	-1.495	1.204	1.994	2.093
3	-0.823	-0.042	1.254	2.314	1.460	1.243
4	0.917	-0.961	0.055	2.049	1.294	1.654
5	-0.736	-1.694	-0.686	0.697	1.284	1.432
6	1.204	-0.605	0.351	2.477	1.611	1.894
7	0.778	-0.436	-0.220	1.911	1.422	1.489
8	1.075	-1.199	-0.141	2.125	1.500	1.896
9	-0.854	-0.654	0.771	1.650	0.793	0.702
10	-1.027	-0.269	0.893	1.891	1.201	0.892
11	-0.288	-2.116	-1.165	1.296	1.848	2.090
12	-0.597	-1.577	-0.618	0.666	1.136	1.298
13	-1.113	-0.271	0.930	1.930	1.267	0.960
14	-0.849	-0.430	0.612	1.569	0.879	0.538
15	1.280	-1.188	0.053	2.384	1.644	2.069

- (a) Assign each instance to its nearest cluster to generate the clustering at the first iteration of *k*-means on the basis of the initial cluster centroids.

The clusters to be assigned to each instance are shown below.

ID	PRESSURE	TEMP.	VOLUME	Cluster Distances Iter. 1			Iter. 1 Cluster
				$Dist(\mathbf{d}_i, \mathbf{c}_1)$	$Dist(\mathbf{d}_i, \mathbf{c}_2)$	$Dist(\mathbf{d}_i, \mathbf{c}_3)$	
1	-0.392	-1.258	-0.666	0.603	1.057	1.117	C_1
2	-0.251	-1.781	-1.495	1.204	1.994	2.093	C_1
3	-0.823	-0.042	1.254	2.314	1.460	1.243	C_3
4	0.917	-0.961	0.055	2.049	1.294	1.654	C_2
5	-0.736	-1.694	-0.686	0.697	1.284	1.432	C_1
6	1.204	-0.605	0.351	2.477	1.611	1.894	C_2
7	0.778	-0.436	-0.220	1.911	1.422	1.489	C_2
8	1.075	-1.199	-0.141	2.125	1.500	1.896	C_2
9	-0.854	-0.654	0.771	1.650	0.793	0.702	C_3
10	-1.027	-0.269	0.893	1.891	1.201	0.892	C_3
11	-0.288	-2.116	-1.165	1.296	1.848	2.090	C_1
12	-0.597	-1.577	-0.618	0.666	1.136	1.298	C_1
13	-1.113	-0.271	0.930	1.930	1.267	0.960	C_3
14	-0.849	-0.430	0.612	1.569	0.879	0.538	C_3
15	1.280	-1.188	0.053	2.384	1.644	2.069	C_2

- (b) On the basis of the clustering calculated in Part (a), calculate a set of new cluster centroids.

The instances that have been assigned to the first cluster, C_1 , are

ID	PRESSURE	TEMPERATURE	VOLUME
1	-0.392	-1.258	-0.666
2	-0.251	-1.781	-1.495
5	-0.736	-1.694	-0.686
11	-0.288	-2.116	-1.165
12	-0.597	-1.577	-0.618

And, so, the new cluster centroid is the average of these values:

$$\mathbf{c}_1 = \langle -0.453, -1.685, -0.926 \rangle.$$

The instances that have been assigned to the second cluster, C_2 , are

ID	PRESSURE	TEMPERATURE	VOLUME
4	0.917	-0.961	0.055
6	1.204	-0.605	0.351
7	0.778	-0.436	-0.220
8	1.075	-1.199	-0.141
15	1.280	-1.188	0.053

And, so, the new cluster centroid is the average of these values:

$$\mathbf{c}_2 = \langle 1.051, -0.878, 0.020 \rangle$$

The instances that have been assigned to the third cluster, C_3 , are

ID	PRESSURE	TEMPERATURE	VOLUME
3	-0.823	-0.042	1.254
9	-0.854	-0.654	0.771
10	-1.027	-0.269	0.893
13	-1.113	-0.271	0.930
14	-0.849	-0.430	0.612

And, so, the new cluster centroid is the average of these values:

$$\mathbf{c}_3 = \langle -0.933, -0.333, 0.892 \rangle.$$

2. The following table shows details of two different clusterings of the dataset from Question 1—one with $k = 2$ and one with $k = 3$ —and partial workings to calculate the silhouette for the clusterings.

$k = 2$ clustering						$k = 3$ clustering					
Nearest						Nearest					
ID	Cluster	Cluster	$a(i)$	$b(i)$	$s(i)$	ID	Cluster	Cluster	$a(i)$	$b(i)$	$s(i)$
d_1	C_1	C_2	??	1.898	??	d_1	C_1	C_2	0.732	1.681	0.565
d_2	C_1	C_2	1.608	2.879	0.442	d_2	C_1	??	??	??	??
d_3	C_2	C_1	0.624	2.594	0.76	d_3	C_3	C_2	0.624	2.422	0.742
d_4	C_1	C_2	1.261	2.142	0.411	d_4	C_2	C_1	0.482	1.884	??
d_5	C_1	C_2	1.452	2.098	0.308	d_5	C_1	C_3	0.619	2.098	0.705
d_6	C_1	??	??	??	??	d_6	C_2	C_3	0.68	2.24	0.697
d_7	C_1	C_2	1.42	2.061	0.311	d_7	C_2	C_1	0.777	1.935	0.598
d_8	C_1	C_2	1.272	2.432	??	d_8	C_2	C_1	0.558	1.842	0.697
d_9	C_2	C_1	0.496	2.067	0.76	d_9	C_3	C_1	0.496	2.04	0.757
d_{10}	C_2	C_1	0.344	2.375	??	d_{10}	C_3	??	0.344	??	??
d_{11}	C_1	C_2	1.565	2.802	0.441	d_{11}	C_1	C_2	0.769	2.201	0.651
d_{12}	C_1	C_2	1.338	??	??	d_{12}	C_1	C_2	0.592	1.935	0.694
d_{13}	C_2	C_1	0.379	2.444	0.845	d_{13}	C_3	C_1	0.379	2.436	0.844
d_{14}	C_2	C_1	??	2.056	??	d_{14}	C_3	C_1	0.459	2.038	??
d_{15}	C_1	C_2	1.425	2.53	0.437	d_{15}	C_2	C_1	0.579	2.101	0.725

- (a) A number of values are missing from these workings (indicated by ??). Calculate the missing values. The distances between each instance in the dataset from Question 1 (using **Euclidean distance**) are shown in the following distance matrix, and will be useful for this exercise.

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}
d_1	0.00														
d_2	0.99	0.00													
d_3	2.31	3.30	0.00												
d_4	1.52	2.11	2.30	0.00											
d_5	0.56	0.95	2.55	1.95	0.00										
d_6	2.00	2.63	2.29	0.55	2.46	0.00									
d_7	1.50	2.12	2.21	0.61	2.02	0.73	0.00								
d_8	1.56	1.98	2.62	0.35	1.96	0.78	0.82	0.00							
d_9	1.63	2.60	0.78	1.94	1.79	2.10	1.92	2.20	0.00						
d_{10}	1.95	2.93	0.47	2.23	2.15	2.32	2.13	2.52	0.44	0.00					
d_{11}	1.00	0.47	3.23	2.07	0.78	2.61	2.20	1.94	2.49	2.86	0.00				
d_{12}	0.38	0.96	2.43	1.77	0.19	2.26	1.83	1.78	1.69	2.04	0.83	0.00			
d_{13}	2.01	2.98	0.49	2.32	2.19	2.41	2.22	2.61	0.49	0.09	2.91	2.09	0.00		
d_{14}	1.59	2.57	0.75	1.93	1.81	2.08	1.83	2.21	0.28	0.37	2.51	1.70	0.44	0.00	
d_{15}	1.82	2.26	2.68	0.43	2.21	0.66	0.94	0.28	2.31	2.62	2.19	2.03	2.71	2.33	0.00

To illustrate how this exercise is completed the complete workings for calculating the silhouette with for instance d_6 will be shown. In this clustering

\mathbf{d}_6 is a member of C_1 . The first step is calculating the silhouette value for \mathbf{d}_6 is to calculate $a(i)$, the average distance between \mathbf{d}_6 and the other members of cluster C_1 . The members of C_1 are $\{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6, \mathbf{d}_7, \mathbf{d}_8, \mathbf{d}_{11}, \mathbf{d}_{12}, \mathbf{d}_{15}\}$. From the distance matrix provided, the distance from \mathbf{d}_6 to each other member of C_1 is:

$$\mathbf{d}_6 \begin{bmatrix} \mathbf{d}_1 & \mathbf{d}_2 & \mathbf{d}_4 & \mathbf{d}_5 & \mathbf{d}_6 & \mathbf{d}_8 & \mathbf{d}_{11} & \mathbf{d}_{12} & \mathbf{d}_{15} \\ 2.00 & 2.63 & 0.55 & 2.46 & 0.73 & 0.78 & 2.61 & 2.26 & 0.66 \end{bmatrix}$$

The average distance between \mathbf{d}_6 and the other members of C_1 , $a(i)$, is the average of these values: 1.631.

The next step in the algorithm is to calculate the average distance from \mathbf{d}_6 to each member of the other clusters in the clustering. In this case there is just one, C_2 . The distances from \mathbf{d}_6 to the members of C_2 are:

$$\mathbf{d}_6 \begin{bmatrix} \mathbf{d}_3 & \mathbf{d}_9 & \mathbf{d}_{10} & \mathbf{d}_{13} & \mathbf{d}_{14} \\ 2.29 & 2.10 & 2.32 & 2.41 & 2.08 \end{bmatrix}$$

which gives an average of 2.240 which is the value for $b(i)$ as there is only one other cluster. The silhouette width for \mathbf{d}_6 is then calculated as

$$\frac{2.240 - 1.631}{\max(2.240, 1.631)} = 0.272$$

Other values are calculated similarly. The completed table of all values is shown below.

$k = 2$ clustering					
Nearest					
ID	Cluster	Cluster	$a(i)$	$b(i)$	$s(i)$
\mathbf{d}_1	C_1	C_2	1.259	1.898	0.337
\mathbf{d}_2	C_1	C_2	1.608	2.879	0.442
\mathbf{d}_3	C_2	C_1	0.624	2.594	0.76
\mathbf{d}_4	C_1	C_2	1.261	2.142	0.411
\mathbf{d}_5	C_1	C_2	1.452	2.098	0.308
\mathbf{d}_6	C_1	C_2	1.631	2.24	0.272
\mathbf{d}_7	C_1	C_2	1.42	2.061	0.311
\mathbf{d}_8	C_1	C_2	1.272	2.432	0.477
\mathbf{d}_9	C_2	C_1	0.496	2.067	0.76
\mathbf{d}_{10}	C_2	C_1	0.344	2.375	0.855
\mathbf{d}_{11}	C_1	C_2	1.565	2.802	0.441
\mathbf{d}_{12}	C_1	C_2	1.338	1.991	0.328
\mathbf{d}_{13}	C_2	C_1	0.379	2.444	0.845
\mathbf{d}_{14}	C_2	C_1	0.459	2.056	0.776
\mathbf{d}_{15}	C_1	C_2	1.425	2.53	0.437

$k = 3$ clustering					
Nearest					
ID	Cluster	Cluster	$a(i)$	$b(i)$	$s(i)$
\mathbf{d}_1	C_1	C_2	0.732	1.681	0.565
\mathbf{d}_2	C_1	C_2	0.843	2.219	0.62
\mathbf{d}_3	C_3	C_2	0.624	2.422	0.742
\mathbf{d}_4	C_2	C_1	0.482	1.884	0.744
\mathbf{d}_5	C_1	C_3	0.619	2.098	0.705
\mathbf{d}_6	C_2	C_3	0.68	2.24	0.697
\mathbf{d}_7	C_2	C_1	0.777	1.935	0.598
\mathbf{d}_8	C_2	C_1	0.558	1.842	0.697
\mathbf{d}_9	C_3	C_1	0.496	2.04	0.757
\mathbf{d}_{10}	C_3	C_2	0.344	2.363	0.855
\mathbf{d}_{11}	C_1	C_2	0.769	2.201	0.651
\mathbf{d}_{12}	C_1	C_2	0.592	1.935	0.694
\mathbf{d}_{13}	C_3	C_1	0.379	2.436	0.844
\mathbf{d}_{14}	C_3	C_1	0.459	2.038	0.775
\mathbf{d}_{15}	C_2	C_1	0.579	2.101	0.725

- (b) On the basis of the completed table, calculate the silhouette for each clustering.

The silhouette for each clustering is simply the average of the silhouette entries. For the clustering with $k = 2$ this is 0.517, and for the clustering with $k = 3$ this is 0.711.

- (c) On the basis of the silhouette, would you choose 2 or 3 for the value of k for this dataset?

As it has a higher silhouette there is more evidence that three clusters exist in this dataset than two. So $k = 3$ should be chosen.

3. A city tax service has performed a clustering of individual taxpayers using *k*-means clustering in order to better understand groups that might exist within their taxpayer base. The clustering has divided the taxpayers into three clusters. Four descriptive features have been used to describe each taxpayer:

- AGE: The age of the taxpayer.
- YEARSINCURRENTEMPLOYMENT: The number of years that the taxpayer has been in their current job.
- TOTALINCOME: The taxpayer's total income for the current tax year.
- EFFECTIVETAXRATE: The effective tax rate paid by the taxpayer (this is simply tax paid divided by total income).

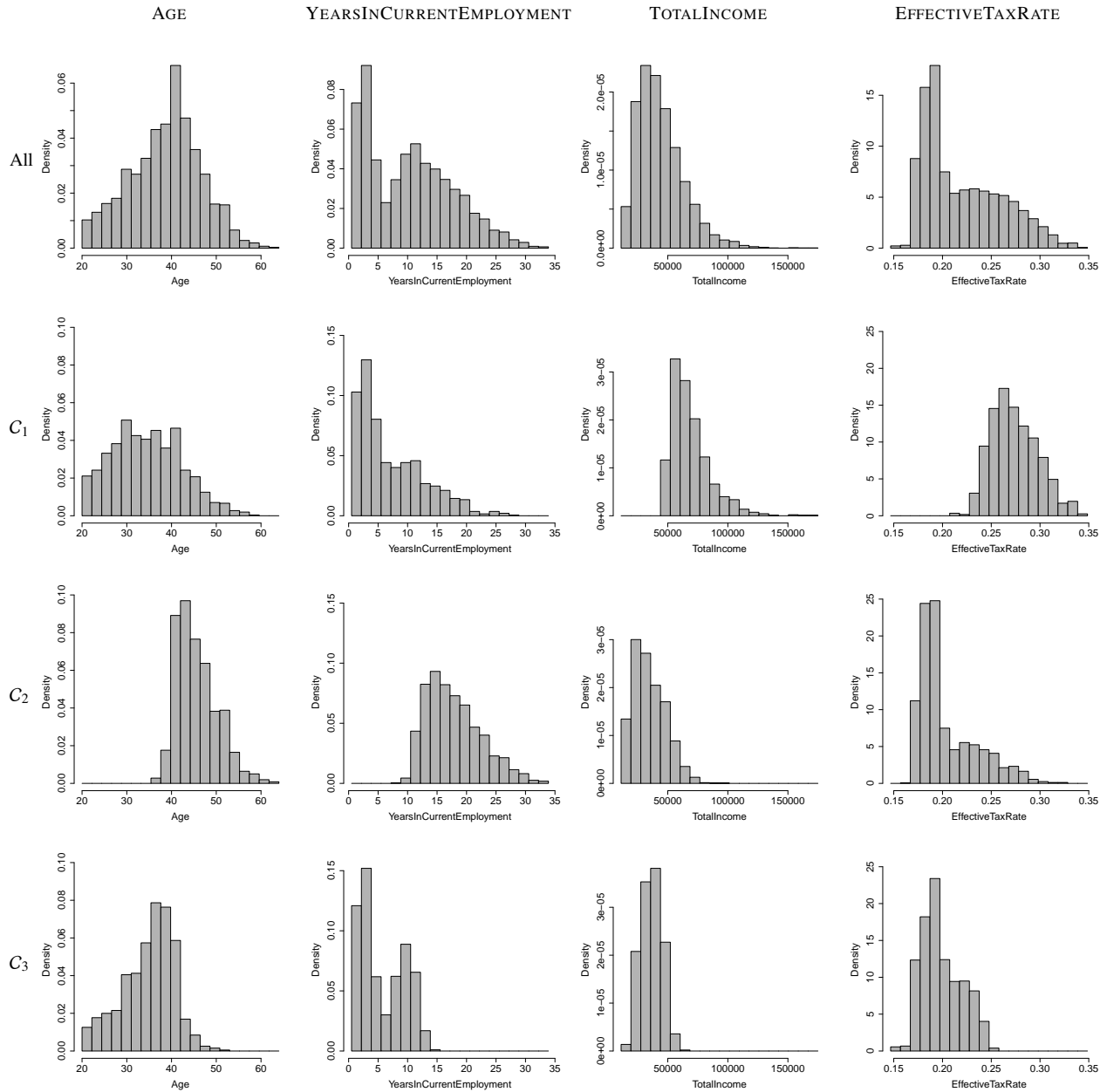
The following table shows summary statistics of the four descriptive features for each of the three clusters found.

Feature	Cluster	1 st				3 rd		Std. Dev.
		Min.	Qrt.	Mean	Median	Qrt.	Max	
AGE	C ₁	20	28	34.6	34	40	59	7.8
	C ₂	36	43	45.8	45	48	64	4.5
	C ₃	20	32	34.9	36	39	52	5.8
YEARSIN CURRENT EMPLOYMENT	C ₁	0.50	2.74	7.18	5.11	10.76	27.40	5.56
	C ₂	8.16	14.25	17.81	17.04	20.71	33.89	4.60
	C ₃	0.50	2.44	5.73	4.38	9.32	14.12	3.73
TOTALINCOME	C ₁	46 247.70	57 355.06	68 843.26	64 977.64	75 967.11	175 000	16 387.77
	C ₂	11 182.46	24 222.04	34 711.67	32 637.42	44 102.08	93 800.98	13 412.08
	C ₃	15 505.02	29 636.07	36 370.00	36 421.53	42 912.04	64 075.62	8 744.26
EFFECTIVE TAXRATE	C ₁	0.210	0.256	0.274	0.271	0.291	0.349	0.024
	C ₂	0.167	0.183	0.204	0.192	0.220	0.321	0.030
	C ₃	0.147	0.183	0.199	0.194	0.214	0.252	0.021

The following table shows the information gain calculated when each descriptive feature is used to predict the membership of a single cluster versus the rest of the population.

Feature	Information Gain		
	C ₁	C ₂	C ₃
AGE	0.0599	0.4106	0.1828
YEARSINCURRENTEMPLOYMENT	0.0481	0.5432	0.3073
TOTALINCOME	0.5015	0.0694	0.1830
EFFECTIVETAXRATE	0.5012	0.0542	0.2166

The following images show histograms of the values of the four descriptive features both for the full dataset and when divided into the three clusters found.



Using the information provided, write a description of what it means for a taxpayer to be a member of each of the clusters.

To perform this task the best place to start is to analyse the information gain values provided in the final table. These indicate which descriptive features are likely to be most useful in describing each cluster. For C_1 the descriptive features with the highest information gain values are TOTALINCOME and EFFECTIVETAXRATE; for C_2 it is YEARSINCURRENTEMPLOYMENT and AGE; and for C_3 it is YEARSINCURRENTEMPLOYMENT and EFFECTIVETAXRATE. The selection of two descriptive features for each cluster is arbitrary here but seems to be suggested by the information gain values. We will always use the other descriptive features in our descriptions too.

Using each of these pairs of descriptive features an examination of the summary statistics in the table provided and the histograms can help understand how to define each of the clusters found. For example, the first cluster, C_1 , can be largely defined as containing taxpayers with relatively large incomes who pay a high effective tax rate. They can also be seen to be relatively young and not long in their current positions. It is common in this type of application of clustering to name clusters so as to easily capture their meaning. We could describe these taxpayers as *high flyers* given their youth and high earnings.

Similarly, we can describe the members of the second cluster, C_2 , as being older and having stayed in their current positions for a long time. They also tend towards lower incomes and pay slightly low tax rates, although this distribution has a long tail. We might describe these as *golden years* taxpayers.

Finally, members of the the third cluster, C_3 , are largely defined by not having been long in their current jobs and paying a fairly low effective tax rate. They also tend to be in their thirties and have relatively low incomes. We might describe these taxpayers as being those in the *middle*.

- * 4. The following table shows a *customer-item matrix* describing items from an online store that customers have bought.

ID	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}
1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
2	1	1	1	1	0	1	1	1	0	0	0	0	0	0
3	1	1	0	1	0	1	1	1	1	0	0	0	1	0
4	1	0	1	0	1	1	1	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	1	1	1	1	1	1	1
6	0	1	0	0	0	0	0	0	1	1	1	1	1	0
7	0	0	0	1	0	0	1	1	1	1	0	1	0	1
8	0	1	0	0	0	0	0	0	0	1	0	1	1	1

The online store would like to cluster their customers to see if they could define meaningful groups to whom they could target special offers. The table below shows a distance matrix calculated using the **Jaccard similarity measure** (see Section 5.4.5^[211]). A number of items have been left out of this matrix (indicated by ??).

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8
d_1	0.000							
d_2	??	0.000						
d_3	0.600	??	0.000					
d_4	0.429	0.500	0.700	0.000				
d_5	1.000	0.923	0.750	1.000	0.000			
d_6	0.909	??	0.727	1.000	0.375	0.000		
d_7	0.917	0.727	0.636	0.909	0.444	??	0.000	
d_8	0.900	0.909	0.818	1.000	0.500	0.429	0.667	0.000

- (a) Using the Jaccard similarity index (reproduced here from Section 5.4.5^[211])

$$dist_J(\mathbf{q}, \mathbf{d}) = 1 - \frac{CP(\mathbf{q}, \mathbf{d})}{CP(\mathbf{q}, \mathbf{d}) + PA(\mathbf{q}, \mathbf{d}) + AP(\mathbf{q}, \mathbf{d})}$$

calculate these missing distances in the preceding distance matrix (note that because this is a distance (or dissimilarity) matrix rather than a similarity matrix, the values shown are $1 - sim_J(\mathbf{q}, \mathbf{d})$).

The distances can be calculated as follows:

$$\begin{aligned}
 dist_J(\mathbf{d}_1, \mathbf{d}_2) &= 1 - \frac{CP(\mathbf{d}_1, \mathbf{d}_2)}{CP(\mathbf{d}_1, \mathbf{d}_2) + PA(\mathbf{d}_1, \mathbf{d}_2) + AP(\mathbf{d}_2, \mathbf{d}_1)} \\
 &= 1 - \frac{5}{5 + 1 + 2} \\
 &= 1 - 0.625 = 0.375
 \end{aligned}$$

$$\begin{aligned}
 dist_J(\mathbf{d}_2, \mathbf{d}_3) &= 1 - \frac{CP(\mathbf{d}_2, \mathbf{d}_3)}{CP(\mathbf{d}_2, \mathbf{d}_3) + PA(\mathbf{d}_2, \mathbf{d}_3) + AP(\mathbf{d}_2, \mathbf{d}_1)} \\
 &= 1 - \frac{6}{6 + 2 + 1} \\
 &= 1 - 0.67 = 0.33
 \end{aligned}$$

$$\begin{aligned}
 dist_J(\mathbf{d}_2, \mathbf{d}_6) &= 1 - \frac{CP(\mathbf{d}_2, \mathbf{d}_6)}{CP(\mathbf{d}_2, \mathbf{d}_6) + PA(\mathbf{d}_2, \mathbf{d}_6) + AP(\mathbf{d}_2, \mathbf{d}_1)} \\
 &= 1 - \frac{1}{1 + 5 + 6} \\
 &= 1 - 0.08 = 0.92
 \end{aligned}$$

$$\begin{aligned}
 dist_J(\mathbf{d}_6, \mathbf{d}_7) &= 1 - \frac{CP(\mathbf{d}_6, \mathbf{d}_7)}{CP(\mathbf{d}_6, \mathbf{d}_7) + PA(\mathbf{d}_6, \mathbf{d}_7) + AP(\mathbf{d}_6, \mathbf{d}_7)} \\
 &= 1 - \frac{3}{3 + 4 + 3} \\
 &= 1 - 0.30 = 0.70
 \end{aligned}$$

The completed table showing missing values is shown below.

	1	2	3	4	5	6	7	8
1	0.000							
2	0.375	0.000						
3	0.600	0.333	0.000					
4	0.429	0.500	0.700	0.000				
5	1.000	0.923	0.750	1.000	0.000			
6	0.909	0.917	0.727	1.000	0.375	0.000		
7	0.917	0.727	0.636	0.909	0.444	0.700	0.000	
8	0.900	0.909	0.818	1.000	0.500	0.429	0.667	0.000

- (b) **Agglomerative hierarchical clustering** (AHC) can easily be applied to this distance matrix. If **single linkage** is used with AHC, which agglomerations will be made in the first three iterations of the algorithm?

Instances \mathbf{d}_2 and \mathbf{d}_3 are closest according to Jaccard similarity index and will still be the first pair of instances combined into a cluster if average linkage were used. The table below shows the revised distance matrix after this combina-

tion is made. Note now that distances are updated to be the average distance between the members of the new cluster C_9 and other instances rather than the minimum as was the case previously.

	1	C_9	2	3	4	5	6	7	8
1	0.000								
C_9	0.375	0.000	-						
2	-	-	-						
3	-	-	-	-					
4	0.429	0.500	-	-	0.000				
5	1.000	0.750	-	-	1.000	0.000			
6	0.909	0.727	-	-	1.000	0.375	0.000		
7	0.917	0.636	-	-	0.909	0.444	0.700	0.000	
8	0.900	0.818	-	-	1.000	0.500	0.429	0.667	0.000

Examining this matrix the next agglomeration to made is a tie between instances \mathbf{d}_5 and \mathbf{d}_6 and instance \mathbf{d}_1 and the newly created cluster C_9 . When the Jaccard similarity index is used ties are common. We will break ties by choosing the items earliest in the dataset (it makes little difference) and so merge \mathbf{d}_1 and C_9 . The table below shows the revised distance matrix after this combination is made.

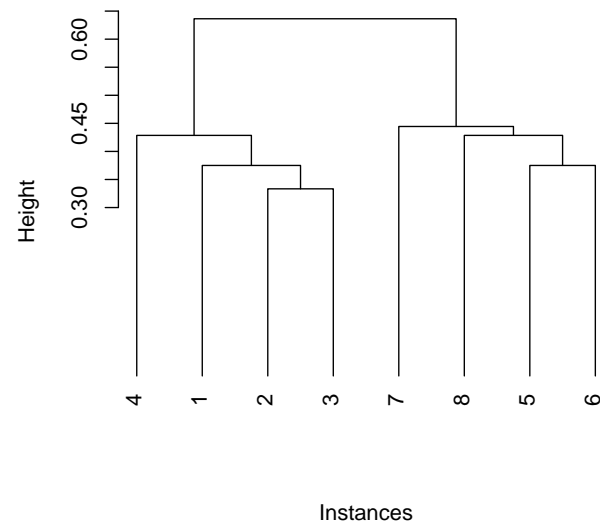
	C_{10}	1	C_9	2	3	4	5	6	7	8
C_{10}	0.000									
1	-	-								
C_9	-	-	-	-						
2	-	-	-	-						
3	-	-	-	-	-					
4	0.429	-	-	-	-	0.000				
5	0.750	-	-	-	-	1.000	0.000			
6	0.727	-	-	-	-	1.000	0.375	0.000		
7	0.636	-	-	-	-	0.909	0.444	0.700	0.000	
8	0.818	-	-	-	-	1.000	0.500	0.429	0.667	0.000

The pair from the tie mentioned previously, \mathbf{d}_5 and \mathbf{d}_6 , are the next agglomeration made. The table below shows the revised distance matrix after this combination is made.

	C_{10}	1	C_9	2	3	4	C_{11}	5	6	7	8
C_{10}	0.000										
1	-										
C_9	-	-									
2	-	-	-								
3	-	-	-	-							
4	0.429	-	-	-	-	0.000					
C_{11}	0.727	-	-	-	-	1.000	0.000				
5	-	-	-	-	-	-	-				
6	-	-	-	-	-	-	-	-			
7	0.636	-	-	-	-	0.909	0.444	0.44	0.700	0.000	
8	0.82	-	-	-	-	1.000	0.429	0.500	0.429	0.667	0.000

For completeness the dendrogram showing the remaining steps in the algorithm is shown below.

Cluster Dendrogram



- (c) If **average linkage** were used with AHC instead of single linkage, which agglomerations would be made in the first three iterations of the algorithm?

Instances d_2 and d_3 are closest according to Jaccard similarity index and will still be the first pair of instances combined into a cluster. The table below shows the revised distance matrix after this combination is made. Note that this is different to the table from this step in the previous example as now average distance linkage is used. So, when calculating the distance between

C_9 and instances from the dataset we calculate the average distance between the members of C_9 (\mathbf{d}_2 and \mathbf{d}_3) and the relevant instances. So, for example the distance between C_9 and \mathbf{d}_4 is the average of the distances between \mathbf{d}_2 and \mathbf{d}_4 , 0.500, and \mathbf{d}_3 and \mathbf{d}_4 , 0.700. This is equal to 0.600 which is the value shown in the table below.

	1	C_9	2	3	4	5	6	7	8
1	0.000								
C_9	0.488	0.000	-						
2	-	-	-						
3	-	-	-	-					
4	0.429	0.600	-	-	0.000				
5	1.000	0.837	-	-	1.000	0.000			
6	0.909	0.822	-	-	1.000	0.375	0.000		
7	0.917	0.682	-	-	0.909	0.444	0.700	0.000	
8	0.900	0.864	-	-	1.000	0.500	0.429	0.667	0.000

Examining this matrix the next agglomeration to made is clearly between instances \mathbf{d}_5 and \mathbf{d}_6 this time. The distance between \mathbf{d}_1 and the newly created cluster C_9 is now higher because average linkage is used. The table below shows the revised distance matrix after this combination is made.

	1	C_9	2	3	4	C_{10}	5	6	7	8
1	0.000									
C_9	0.488	0.000	-							
2	-	-	-							
3	-	-	-	-						
4	0.429	0.600	-	-	0.000					
C_{10}	0.954	0.829	-	-	1.000	0.000				
5	-	-	-	-	-	-	-			
6	-	-	-	-	-	-	-	-		
7	0.917	0.682	-	-	0.909	0.572	-	-	0.000	
8	0.900	0.864	-	-	1.000	0.464	-	-	0.667	0.000

The entry defining the distance between C_9 and C_{10} in this table is interesting. As average linking is used, this is the average of the distances between all of the members of these two clusters. $C_9 = \{\mathbf{d}_2, \mathbf{d}_3\}$ and $C_{10} = \{\mathbf{d}_5, \mathbf{d}_6\}$. The pairwise distances between instances are:

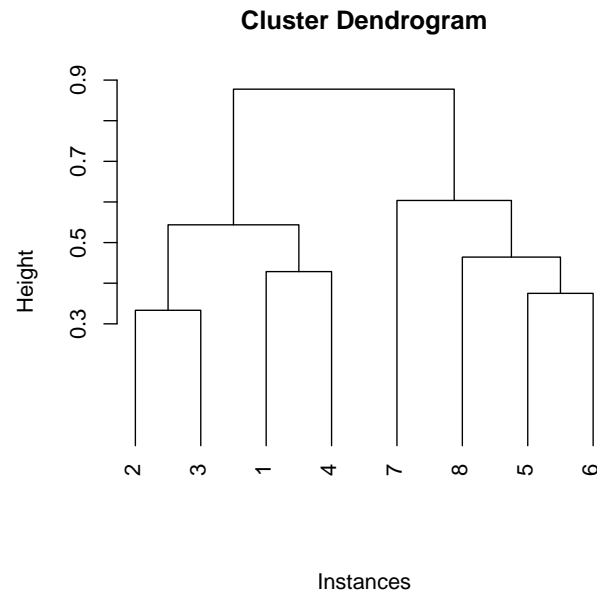
	\mathbf{d}_2	\mathbf{d}_3
\mathbf{d}_5	0.923	0.750
\mathbf{d}_6	0.917	0.727

So the distance between C_9 and C_{10} is the average of these values, 0.829.

On the third iteration of the algorithm instances \mathbf{d}_1 and \mathbf{d}_4 , are closest and so are the the next agglomeration made. Note her that this is now different from when single linkage is used. The table below shows the revised distance matrix after this combination is made.

	C_{11}	1	C_9	2	3	4	C_{10}	5	6	7	8
C_{11}	0.000	-									
1	-	-									
C_9	0.544	0.488	0.000	-							
2	-	-	-	-							
3	-	-	-	-	-						
4	-	-	-	-	-	-					
C_{10}	0.977	0.954	0.829	-	-	-	0.000				
5	-	-	-	-	-	-	-	-			
6	-	-	-	-	-	-	-	-	-		
7	0.913	-	0.682	-	-	-	0.572	-	-	0.000	
8	0.950	-	0.864	-	-	-	0.464	-	-	0.667	0.000

For completeness the dendrogram showing the remaining steps in the algorithm is shown below.



- * 5. The following table shows a small dataset used for human activity recognition from a wearable accelerometer sensor.¹ Each instance describes the average acceleration in the X, Y, and Z directions within a short time window. There are no labels, so this data is being clustered in an attempt to recognize different activity from this simple data stream. The **k-means clustering** approach is to be applied to this dataset with $k = 2$ and using **Euclidean distance**. The initial cluster centroids for the two clusters C_1 and C_2 are $\mathbf{c}_1 = \langle -0.235, 0.253, 0.438 \rangle$ and $\mathbf{c}_2 = \langle 0.232, 0.325, -0.159 \rangle$. The following table also shows the distance to these three cluster centers for each instance in the dataset.

ID	X	Y	Z	Cluster Distances Iter. 1	
				$Dist(\mathbf{d}_i, \mathbf{c}_1)$	$Dist(\mathbf{d}_i, \mathbf{c}_2)$
1	-0.154	0.376	0.099	0.370	0.467
2	-0.103	0.476	-0.027	0.532	0.390
3	0.228	0.036	-0.251	0.858	0.303
4	0.330	0.013	-0.263	0.932	0.343
5	-0.114	0.482	0.014	0.497	0.417
6	0.295	0.084	-0.297	0.922	0.285
7	0.262	0.042	-0.304	0.918	0.319
8	-0.051	0.416	-0.306	0.784	0.332

- (a) Assign each instance to its nearest cluster to generate the clustering at the first iteration of k -means on the basis of the initial cluster centroids.

The clusters to be assigned to each instance are shown below.

ID	X	Y	Z	Cluster Distances Iter. 1		Iter. 1 Cluster
				$Dist(\mathbf{d}_i, \mathbf{c}_1)$	$Dist(\mathbf{d}_i, \mathbf{c}_2)$	
1	-0.154	0.376	0.099	0.370	0.467	C_1
2	-0.103	0.476	-0.027	0.532	0.390	C_2
3	0.228	0.036	-0.251	0.858	0.303	C_2
4	0.330	0.013	-0.263	0.932	0.343	C_2
5	-0.114	0.482	0.014	0.497	0.417	C_2
6	0.295	0.084	-0.297	0.922	0.285	C_2
7	0.262	0.042	-0.304	0.918	0.319	C_2
8	-0.051	0.416	-0.306	0.784	0.332	C_2

- (b) On the basis of the clustering calculated in Part (a), calculate a set of new cluster centroids.

1. The data in this question has been artificially created but is inspired by the **Human Activity Recognition Using Smartphones Dataset** first described by ? and available from the UCI Machine Learning Repository (?).

The instances that have been assigned to the first cluster, C_1 , are

ID	X	Y	Z
1	-0.154	0.376	0.099

And, so, the new cluster centroid is just this single value:

$$\mathbf{c}_1 = \langle -0.154, 0.376, 0.099 \rangle.$$

The instances that have been assigned to the second cluster, C_2 , are

ID	X	Y	Z
2	-0.103	0.476	-0.027
3	0.228	0.036	-0.251
4	0.330	0.013	-0.263
5	-0.114	0.482	0.014
6	0.295	0.084	-0.297
7	0.262	0.042	-0.304
8	-0.051	0.416	-0.306

And, so, the new cluster centroid is the average of these values:

$$\mathbf{c}_2 = \langle 0.121, 0.221, -0.205 \rangle$$

- (c) Calculate the distances of each instance to these new cluster centers and perform another clustering iteration.

The distances to the two cluster centres, and the resulting clustering are shown below.

ID	X	Y	Z	Cluster Distances Iter. 2		Iter. 2 Cluster
				$Dist(\mathbf{d}_i, \mathbf{c}_1)$	$Dist(\mathbf{d}_i, \mathbf{c}_2)$	
1	-0.154	0.376	0.099	0.370	0.467	C_1
1	-0.154	0.376	0.099	0.000	0.438	C_1
2	-0.103	0.476	-0.027	0.169	0.383	C_1
3	0.228	0.036	-0.251	0.620	0.219	C_2
4	0.330	0.013	-0.263	0.705	0.301	C_2
5	-0.114	0.482	0.014	0.142	0.414	C_1
6	0.295	0.084	-0.297	0.666	0.240	C_2
7	0.262	0.042	-0.304	0.669	0.248	C_2
8	-0.051	0.416	-0.306	0.420	0.279	C_2

11

Beyond Prediction: Reinforcement Learning (Exercise Solutions)

1. An agent in an environment completes an episode and receives the following rewards:

$$\{r_0 = -33, r_1 = -11, r_2 = -12, r_3 = 27, r_4 = 87, r_5 = 156\}$$

- (a) Calculate the discounted return at time $t = 0$ on the basis of this sequence of rewards using a discounting factor of 0.72.

We can apply Equation (11.8)^[642] to this sequence of rewards to calculate the discount rate, as viewed from $t = 0$ where $\gamma = 0.72$:

$$\begin{aligned} G &= (0.72^0 \times -33) + (0.72^1 \times -11) + (0.72^2 \times -12) \\ &\quad + (0.72^3 \times 27) + (0.72^4 \times 87) + (0.72^5 \times 156) \\ &= (-33) + (0.72 \times -11) + (0.5184 \times -12) \\ &\quad + (0.3732 \times 27) + (0.2687 \times 87) + (0.1935 \times 156) \\ &= 16.4985 \end{aligned}$$

- (b) Calculate the discounted return at time $t = 0$ on the basis of this sequence of rewards using a discount rate of 0.22.

We can apply Equation (11.8)^[642] to this sequence of rewards to calculate the discounted return, as viewed from $t = 0$ where $\gamma = 0.22$:

$$\begin{aligned} G &= (0.22^0 \times -33) + (0.22^1 \times -11) + (0.22^2 \times -12) \\ &\quad + (0.22^3 \times 27) + (0.22^4 \times 87) + (0.22^5 \times 156) \\ &= (-33) + (0.22 \times -11) + (0.0484 \times -12) \\ &\quad + (0.0106 \times 27) + (0.0023 \times 87) + (0.0005 \times 156) \\ &= -35.4365 \end{aligned}$$

2. To try to better understand the slightly baffling behavior of her new baby, Maria—a scientifically minded new mother—monitored her baby girl over the course of a day

recording her activity at 20 minute intervals. The activity stream looked like this (with time flowing down through the columns):

SLEEPING	SLEEPING	SLEEPING	CRYING	SLEEPING	SLEEPING
CRYING	SLEEPING	HAPPY	HAPPY	CRYING	HAPPY
SLEEPING	SLEEPING	CRYING	HAPPY	SLEEPING	HAPPY
SLEEPING	CRYING	SLEEPING	HAPPY	SLEEPING	HAPPY
HAPPY	SLEEPING	HAPPY	HAPPY	SLEEPING	HAPPY
HAPPY	SLEEPING	HAPPY	SLEEPING	HAPPY	HAPPY
HAPPY	HAPPY	HAPPY	SLEEPING	HAPPY	SLEEPING
SLEEPING	SLEEPING	HAPPY	SLEEPING	HAPPY	SLEEPING
SLEEPING	HAPPY	HAPPY	SLEEPING	SLEEPING	SLEEPING
SLEEPING	HAPPY	HAPPY	SLEEPING	HAPPY	SLEEPING
SLEEPING	CRYING	CRYING	SLEEPING	SLEEPING	SLEEPING

Maria noticed that her baby could occupy one of three states—HAPPY, CRYING, or SLEEPING—and moved quite freely between them.

- (a) On the basis of this sequence of states, calculate a transition matrix that gives the probability of moving between each of the three states.

The first step to building the transition matrix is to count the frequency of each possible state transition. Working down through the list of states we can count the number of times we move from one state to the next. For example, in the first five states {SLEEPING, CRYING, SLEEPING, SLEEPING, SLEEPING} there is one transition from SLEEPING \rightarrow CRYING, one transition from CRYING \rightarrow SLEEPING, and two transitions from SLEEPING \rightarrow SLEEPING. This would give the following partial transition frequency matrix:

	SLEEPING	CRYING	HAPPY
SLEEPING	2	1	0
CRYING	1	0	0
HAPPY	0	0	0

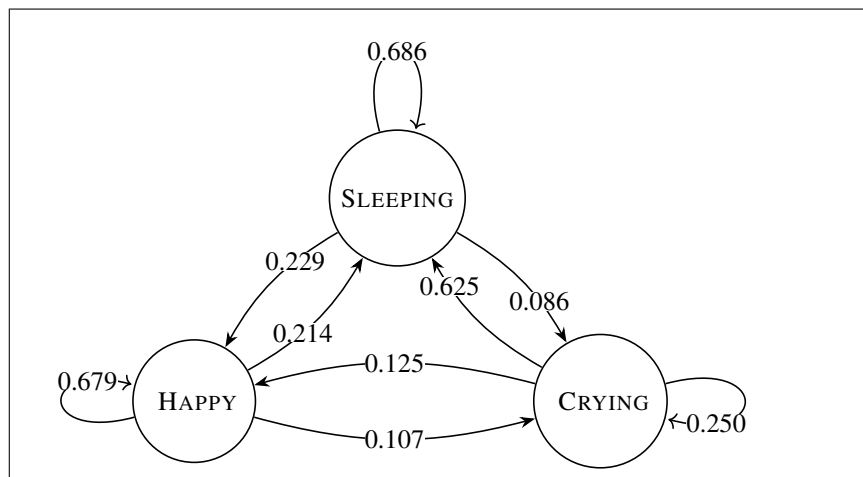
The final frequency table is shown in the table below:

	SLEEPING	CRYING	HAPPY
SLEEPING	24	3	8
CRYING	5	2	1
HAPPY	6	3	19

By normalising each row in the table (dividing each value by the sum of values in the row) we can calculate the final transition matrix:

	SLEEPING	CRYING	HAPPY
SLEEPING	0.686	0.086	0.229
CRYING	0.625	0.250	0.125
HAPPY	0.214	0.107	0.679

- (b) Draw a **Markov process** diagram to capture the behavior of a small baby as described.



3. The following table shows the action-value table for a reinforcement learning agent learning to play the **TwentyTwos** game after 20 episodes of training have elapsed.

State	Action	Value	State	Action	Value	State	Action	Value
PL-DL	<i>Twist</i>	0.706	PH-DL	<i>Twist</i>	−0.038	PM-DH	<i>Twist</i>	0.533
PL-DL	<i>Stick</i>	0.284	PH-DL	<i>Stick</i>	0.164	PM-DH	<i>Stick</i>	−0.526
PM-DL	<i>Twist</i>	−0.985	PL-DH	<i>Twist</i>	0.386	PH-DH	<i>Twist</i>	0.154
PM-DL	<i>Stick</i>	0.589	PL-DH	<i>Stick</i>	−0.832	PH-DH	<i>Stick</i>	0.103
BUST	<i>Twist</i>	0.000	TIE	<i>Twist</i>	0.000	WIN	<i>Twist</i>	0.000
BUST	<i>Stick</i>	0.000	TIE	<i>Stick</i>	0.000	WIN	<i>Stick</i>	0.000
LOSE	<i>Twist</i>	0.000				TWENTYTWO	<i>Twist</i>	0.000
LOSE	<i>Stick</i>	0.000				TWENTYTWO	<i>Stick</i>	0.000

In the answers to the following questions, assume that after the initial cards have been dealt to the player and the dealer, the following cards are coming up next in the deck: 10 ♥, 2 ♣, 7 ♣, K ♥, 9 ♦.

- (a) At the beginning of the first episode the player is dealt (2 ♥, K ♣), the dealer is dealt (A ♦, 3 ♦), and the dealer's visible card is the A ♦. Given these cards, what state is the TwentyTwos playing agent in?

The value of the player's hand is 12 and the value of the dealer's visible card is 11 so the agent finds itself in the PL-DH state.

- (b) Assuming that the next action that the agent will take is selected using a **greedy action selection policy**, what action will the agent choose to take (*Stick* or *Twist*)?

To answer this question we need to examine the action-value table to find the expected discounted return for each possible action from this state:

- $Q(\text{PL-DH}, \text{Twist}) = 0.386$
- $Q(\text{PL-DH}, \text{Stick}) = -0.832$

As the expected return for the *Twist* action is higher this is the action that the agent will select using a greedy action selection policy.

- (c) Simulate taking the action that the agent selected in Part (b) and determine the state that the agent will move to following this action and the reward that they will receive. (**Note:** If cards need to be dealt to the player or dealer, use cards from the list given at the beginning of this question.)

The agent selected the *Twist* action and the next card in the deck that will be dealt is the 10 ♥. So, the cards in the player's hand will now be (2 ♥, K ♣, 10 ♥)

which have a value of 22. This means that the agent will move to the PH-DH state. As this is a non-terminal state the agent will receive a reward of 0.

- (d) Assuming that **Q-learning** is being used with $\alpha = 0.2$ and $\gamma = 0.9$, update the entry in the action-value table for the action simulated in Part (c).

To update the $Q(\text{PL-DH}, \text{Twist})$ entry in the action-value table we can use Equation 11.24^[658] as follows:

$$\begin{aligned}
 Q(\text{PL-DH}, \text{Twist}) & \\
 &\leftarrow Q(\text{PL-DH}, \text{Twist}) + \\
 &\quad \alpha \times \left(R(\text{PL-DH}, \text{Twist}) + \gamma \times \max_a Q(\text{PH-DH}, a) - Q(\text{PL-DH}, \text{Twist}) \right)
 \end{aligned}$$

To fill in all value in this table we need to determine $\max_a Q(\text{PH-DH}, a)$, the action from PH-DH with the highest expected return. The action value function entries for the two possible actions are:

- $Q(\text{PH-DH}, \text{Twist}) = 0.154$
- $Q(\text{PH-DH}, \text{Stick}) = 0.103$

At this point in the learning process the *Twist* has the higher expected return and so $Q(\text{PH-DH}, \text{Twist})$ is used in the action-value function entry update equation:

$$\begin{aligned}
 Q(\text{PL-DH}, \text{Twist}) & \\
 &\leftarrow Q(\text{PL-DH}, \text{Twist}) + \\
 &\quad \alpha \times (R(\text{PL-DH}, \text{Twist}) + \gamma \times Q(\text{PH-DH}, \text{Twist}) - Q(\text{PL-DH}, \text{Twist})) \\
 &\quad 0.386 + 0.2 \times (0 + 0.9 \times 0.154 - 0.386) \\
 &\quad 0.361
 \end{aligned}$$

- (e) Assuming that a **greedy action selection policy** is used again and that **Q-learning** is still being used with $\alpha = 0.2$ and $\gamma = 0.9$, select the next action that the agent will perform, simulate this action, and update the entry in the action-value table for the action. (**Note:** If cards need to be dealt to the player or dealer, continue to use cards from the list given at the beginning of this question.)

The agent finds itself in the PH-DH state. So, assuming again that a greedy action selection policy is being used it will choose the action with the highest

value in the action-value function table. The entries for *Twist* and *Stick* from this state are:

- $Q(\text{PH-DH}, \text{Twist}) = 0.154$
- $Q(\text{PH-DH}, \text{Stick}) = 0.103$

At this point in the learning process the *Twist* has the higher expected return and so $Q(\text{PH-DH}, \text{Twist})$ will be selected. In this case, unfortunately, this is not a good idea. The player's hand has a total value of 22 so any new card will send them bust. In this instance the card dealt from the deck described above is the 2♣ which takes the player agent in the the BUST state with a reward of -1 .

The action-value function table entry for $Q(\text{PH-DH}, \text{Twist})$ can be updated as:

$$\begin{aligned} &Q(\text{PH-DH}, \text{Twist}) \\ &\leftarrow Q(\text{PH-DH}, \text{Twist}) + \\ &\quad \alpha \times \left(R(\text{PH-DH}, \text{Twist}) + \gamma \times \max_a Q(\text{BUST}, a) - Q(\text{PH-DH}, \text{Twist}) \right) \end{aligned}$$

The BUST state is a terminal state so it doesn't matter which action is selected next as all return an expected return of 0.0. So the update becomes:

$$\begin{aligned} &Q(\text{PH-DH}, \text{Twist}) \\ &\leftarrow Q(\text{PH-DH}, \text{Twist}) + \\ &\quad \alpha \times (R(\text{PH-DH}, \text{Twist}) + \gamma \times Q(\text{BUST}, \text{Twist}) - Q(\text{PH-DH}, \text{Twist})) \\ &\quad 0.154 + 0.2 \times (-1 + 0.9 \times 0 - 0.154) \\ &\quad 0.039 \end{aligned}$$

- (f) On the basis of the changes made to the TwentyTwos playing agent's action-value table following the two actions taken in the previous parts of this question, how has the agent's **target policy** changed?

The first action taken didn't change the overall policy in a noticeable way. The action-value function table entry for $Q(\text{PL-DH}, \text{Twist})$ reduced slightly, but at 0.36 it is still higher than the value for $Q(\text{PL-DH}, \text{Stick})$ and so a greedy target policy will still select the *Twist* action in that state.

The results of the changes resulting from the second action are more impactful. The choice to *Twist* in the PH-DH state led to the player going bust

and reduced the action-value function table entry for $Q(\text{PH-DH}, \textit{Twist})$ from 0.154 to 0.039. This is less than the entry for $Q(\text{PH-DH}, \textit{Stick})$, at 0.103, and so the target policy from that state will not swap from the *Twist* action to the *Stick* action—probably a moire sensible choice.

- * 4. As part of a project to develop a self-driving taxi system, the behavior of a taxi driver has been observed over a work day. During their shift the taxi driver can CRUISE looking for work, wait for a fare at a taxi RANK, take a FARE and deliver a passenger to their destination, or take a BREAK. The behavior of the taxi driver over the shift looked like this (with time flowing down through the columns):

CRUISE	RANK	RANK	FARE	CRUISE	CRUISE
FARE	CRUISE	BREAK	FARE	FARE	BREAK
CRUISE	CRUISE	CRUISE	FARE	FARE	RANK
CRUISE	FARE	CRUISE	FARE	FARE	RANK
CRUISE	FARE	CRUISE	FARE	CRUISE	RANK
FARE	CRUISE	FARE	FARE	CRUISE	FARE
FARE	RANK	FARE	CRUISE	FARE	FARE
FARE	FARE	FARE	CRUISE	FARE	CRUISE
CRUISE	CRUISE	FARE	CRUISE	FARE	CRUISE
CRUISE	FARE	FARE	CRUISE	CRUISE	CRUISE
CRUISE	FARE	FARE	CRUISE	FARE	CRUISE
CRUISE	RANK	FARE	CRUISE	CRUISE	CRUISE

- (a) On the basis of this behavior sequence, calculate a transition matrix that gives the probability of moving between all the four states.

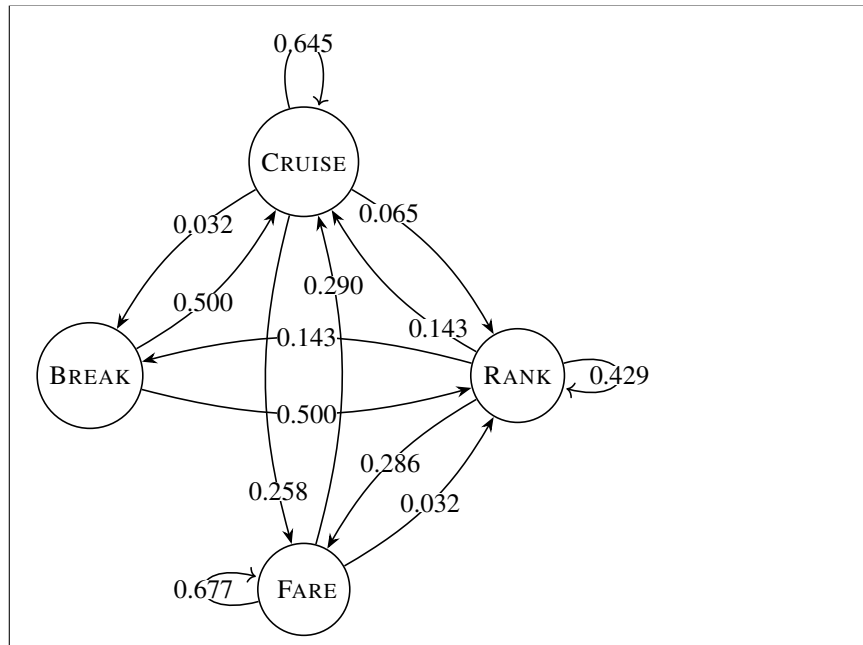
The first step to building the transition matrix is to count the frequency of each possible state transition. Working down through the list of states we can count the number of times we move from one state to the next. The frequency table calculated from this behavior sequence is:

	CRUISE	RANK	FARE	BREAK
CRUISE	20	2	8	1
RANK	1	3	2	1
FARE	9	1	21	0
BREAK	1	1	0	0

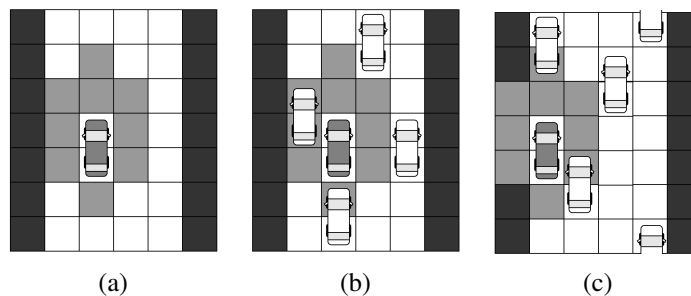
By normalising each row in the table (dividing each value by the sum of values in the row) we can calculate the final transition matrix:

	CRUISE	RANK	FARE	BREAK
CRUISE	0.645	0.065	0.258	0.032
RANK	0.143	0.429	0.286	0.143
FARE	0.290	0.032	0.677	0.000
BREAK	0.500	0.500	0.000	0.000

- (b) Draw a **Markov process** diagram to capture the behavior of the taxi driver as described.



- * 5. The following image labeled (a) shows a simple schematic of a system used to train a self-driving car to drive on a four-lane highway. The car has sensors on the front, the rear, and the sides that indicate the presence of other cars or lane barriers in the area immediately surrounding the car. The shaded cells in Image (a) below show the region that these sensors cover. The region around the car is divided into cells that can be empty, occupied by another car, or occupied by a barrier. Cars occupy an area covered by two cells (one above the other as shown in Image (a)). Images (b) and (c) show the car in other positions where other cars and barriers are sensed by the car, but other cars and barriers are out of range of the sensors.



The car can move at three speeds: *stationary*, *slow*, and *fast*. When moving fast, the car moves forward two cells per time-step; when moving slowly the car moves forward one cell per time-step; and when stationary does not move forward at all. The actions that the car can take are to (1) maintain its current speed, (2) increase its speed (move up one level in the speed categories stationary, slow, and fast), (3) decrease its speed (move down one level in the speed categories), (4) move to the left, and (5) move to the right.

When the car changes speed, the action has an immediate effect on the car's progress in the current time-step. If the car is stationary, taking the action to move left or right has no effect on the car's position. If the car is moving slowly, then moving left or right moves the car one cell in that direction at that time-step, but not any distance forward. If the car is moving fast, then moving left or right moves the car one cell in that direction at that time-step and one cell forward.

The goal that the agent is being trained to achieve is to learn to drive as far as possible in the shortest amount of time possible without crashing. The car moves forward along an infinite highway, and an episode ends if the car crashes into a barrier or another car.

- (a) Design a state representation for the car agent in this scenario. How many states will exist in the representation?

The most straight-forward representation for this agent would be one in which the cells sensed by the car define a set of states corresponding to whether they are occupied or empty. Given that the sensors on the car cover 9 cells that means that there are $2^9 = 512$ possible configurations as each cell can be either occupied or empty. The images labeled (a), (b) and (c) above show three possible configurations. In (a) all cells are empty, in (b) two of the cells to the left of the car and the cell directly behind the car are occupied, and in (c) all cells to the left are occupied as well as one cell in front and one to the right and behind. As the car can move at three speeds so this would also need to be taken into account as well and so this would mean $512 \times 3 = 1536$ states.

The number of states in the representation could be reduced by simplifying the level of data captured - for example just indicating if there was a car to the left or right or not rather than capturing exactly which cells were occupied. The representation could also be made more rich, for example by distinguishing between cells occupied by barriers and cells occupied by cars, or by taking account of the speed at which other cars are moving. The key in designing state representations is to find the simplest representation that captured everything that is important to the goal the agent is trying to achieve.

- (b) Given the state representation that you have defined in Part (a) and the actions available to the agent, how many entries would the action-value function table for a tabular reinforcement learning agent trained for this task have?

There are five actions available to the agent and the state representation in part (a) had 1 536 states. Therefore there would need to be $5 \times 1\,536 = 7\,680$ entries in an action-value function table.

- (c) Design a reward function for this scenario.

The goal in this scenario is to train an agent to cover as much ground as possible in the shortest time possible without crashing. A simple reward scheme would be to reward the agent +1 for every time-step for which it doesn't crash. This might seem like it would encourage safe driving behavior, but in fact would result in an agent learning to stay stationary—the only sure way not to crash (assuming other cars don't crash into it). A simple and effective reward function would be to reward the agent according to the number of cells travelled at each time-step. This would be 0 when stationary, +1 when moving slow, and +2 when travelling fast. This simple reward scheme would result in an agent that could learn to drive along the highway.

Bibliography

- Alimoglu, Fevzi, and Ethem Alpaydin. 1996. Methods of combining multiple classifiers based on different representations for pen-based handwritten digit recognition. In *Proceedings of the fifth Turkish artificial intelligence and artificial neural networks symposium (TAINN'96)*.
- Anguita, Davide, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. 2013. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the 21st international European symposium on artificial neural networks, computational intelligence, and machine learning (ESANN'13)*, 437–442.
- Bache, K., and M. Lichman. 2013. UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml>.
- Bejar, J., U. Cortés, and M. Poch. 1991. LINNEO+: A classification methodology for ill-structured domains, Research report RT-93-10-R, Dept. Llenguatges i Sistemes Informatics, Universitat Politècnica de Catalunya.
- Berk, Richard A., and Justin Bleich. 2013. Statistical procedures for forecasting criminal behavior. *Criminology & Public Policy* 12 (3): 513–544.
- Bray, Freddie, Jacques Ferlay, Isabelle Soerjomataram, Rebecca L. Siegel, Lindsey A. Torre, and Ahmedin Jemal. 2018. Global cancer statistics 2018: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: A Cancer Journal for Clinicians* 68 (6): 394–424. doi:10.3322/caac.21492.
- Cleary, Duncan, and Revenue Irish Tax. 2011. Predictive analytics in the public sector: Using data mining to assist better target selection for audit. In *The proceedings of the 11th European conference on e-government: Faculty of administration, University of Ljubljana, Ljubljana, Slovenia, 16–17 June 2011*, 168. Academic Conferences Limited.
- Dua, Dheeru, and Casey Graff. 2017. UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml>.
- Fawcett, Tom. 2006. An introduction to ROC analysis. *Pattern Recognition Letters* 27 (8): 861–874.
- Hirschowitz, Anton. 2001. Closing the CRM loop: The 21st century marketer's challenge: Transforming customer insight into customer value. *Journal of Targeting, Measurement and Analysis for Marketing* 10 (2): 168–178.
- Kansagara, Devan, Honora Englander, Amanda Salanitro, David Kagen, Cecelia Theobald, Michele Freeman, and Sunil Kripalani. 2011. Risk prediction models for hospital readmission: A systematic review. *JAMA* 306 (15): 1688–1698.

- Klubička, Filip, Giancarlo D. Salton, and John D. Kelleher. 2018. Is it worth it? Budget-related evaluation metrics for model selection. In *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*.
- Kohavi, Ron. 1996. Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid. In *Proceedings of the twenty-fifth ACM SIGKDD international conference on knowledge discovery and data mining KDD*, 202–207.
- Mac Namee, B., P. Cunningham, S. Byrne, and O. I. Corrigan. 2002. The problem of bias in training data in regression problems in medical decision support. *Artificial Intelligence in Medicine* 24 (1): 51–70.
- Mangasarian, Olvi L., and William H. Wolberg. 1990. Cancer diagnosis via linear programming. *SIAM News* 23 (5): 1–18.
- Mishne, Gilad, and Natalie S. Glance. 2006. Predicting movie sales from blogger sentiment. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, 155–158.
- Osowski, Stanislaw, Linh Tran Hoai, and T. Markiewicz. 2004. Support vector machine-based expert system for reliable heartbeat recognition. *IEEE Transactions on Biomedical Engineering* 51 (4): 582–589. doi:10.1109/TBME.2004.824138.
- Palaniappan, Sellappan, and Rafiah Awang. 2008. Intelligent heart disease prediction system using data mining techniques. *International Journal of Computer Science and Network Security* 8 (8): 343–350.
- Rubin, Daniel J. 2015. Hospital readmission of patients with diabetes. *Current Diabetes Reports* 15 (4): 17.
- Siddiqi, Naeem. 2005. *Credit risk scorecards: Developing and implementing intelligent credit scoring*. Wiley.
- Tsanas, Athanasios, and Angeliki Xifara. 2012. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings* 49: 560–567.