Machine Learning Algorithms

Click on a question number to see how your answers were marked and, where available, full solutions.

Question Number	Score
Supervised Learning Algorithms	
Question 1	0 / 0
Question 2	1 / 1
Question 3	3 / 3
Question 4	1 / 1
Question 5	1 / 1
Question 6	1 / 1
Unsupervised Learning Algorithms	
Question 7	1 / 1
Question 8	1 / 1
Reinforcement Learning Algorithms	
Question 9	1 / 1
Evolutionary Algorithms	
Question 10	1 / 1
Total	**11 / 11 (100%)**

The pass rate for the questions in the tutorial is 50%, if you score less than this you might want to revisit the questions you had difficulty with and read some of the resources pertaining to that topic.

Thank you for using this tool, in order to improve the system please complete the questionnare linked below:

https://forms.ncl.ac.uk/view.php?id=6719176 (https://forms.ncl.ac.uk/view.php?id=6719176)

Performance Summary

Exam Name:	Machine Learning Algorithms
Session ID:	15416011971
Exam Start:	Thu Dec 10 2020 11:51:33
Exam Stop:	Thu Dec 10 2020 11:52:18
Time Spent:	0:00:45

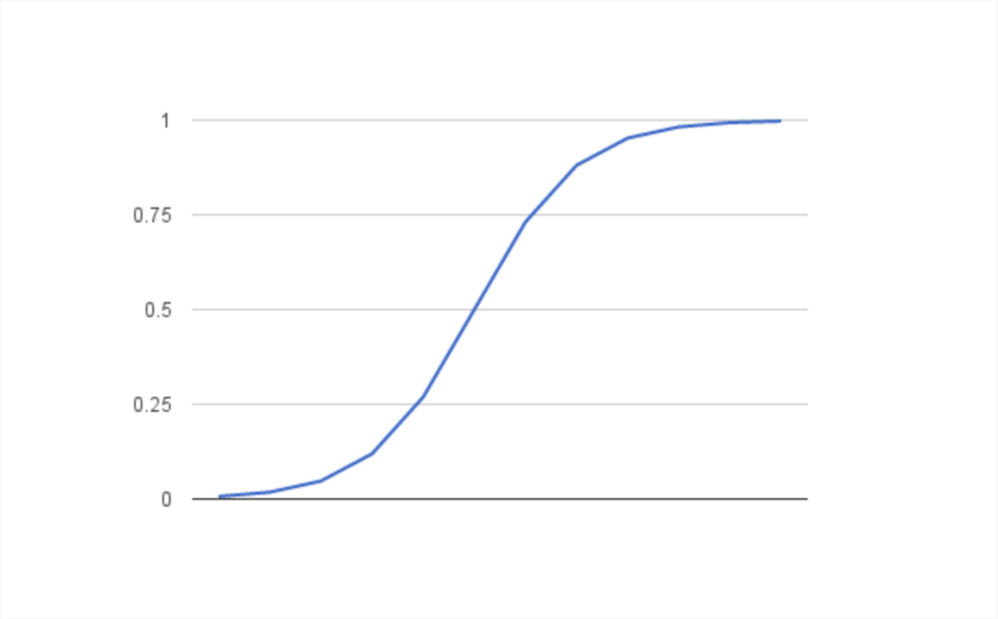
Question 1

Logistic Regression

In spite of its name, logistic regression is a model for **classification**, not regression and is one of the most widely used classification algorithms, particularly for binary classfication (problems with two class values).

Examples of classification tasks where logisitic regression is used include spam detection and predicting whether a user will click on an advertisement.

Logistic regression is named after the logistic function or sigmoid function as it is also known, this function is an S shaped curve which can take any real-valued number and map it onto a value between 0 and 1 (but never exactly at those limits).



Logistic Function

Input values (x) are linearly combined using weights or coefficient values (B- Greek captical letter Beta) to predict an output value (y). A *coefficient* is a number used to multiply a variable.

The output value being modelled is a binary value (0 or 1).

You must estimate the coefficients of the logistic regression model, this is done using maximum likelihood estimation- this is a common learning algorithm used by a number of machine learning algorithms.

The main idea behind maximum likelihood estimation for linear regression is that a search procedure seeks values for the coefficients that minimise the error in the probabilities predicted by the model to those in the data.

The ideal coefficients would result in a model which predicts a value close to 1 for the default class and a close value to 0 for the other class.

For a step by step tutorial of how to implement logistic regression in Python visit: https://datatofish.com/logistic-regression-python/ (https://datatofish.com/logistic-regression-python/)

Resources:

Image from: https://machinelearningmastery.com/logistic-regression-for-machine-learning/ (https://machinelearningmastery.com/logistic-regression-for-machine-learning/)

https://towardsdatascience.com/probability-concepts-explained-maximum-likelihood-estimation-c7b4342fdbb1 (https://towardsdatascience.com/probability-concepts-explained-maximum-likelihood-estimation-c7b4342fdbb1)- Further information on Maximum Likelihood Estimation

https://www.mathsisfun.com/definitions/coefficient.html (https://www.mathsisfun.com/definitions/coefficient.html) - Further information on coefficients

https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python
(https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python)

Question 2

Linear Regression

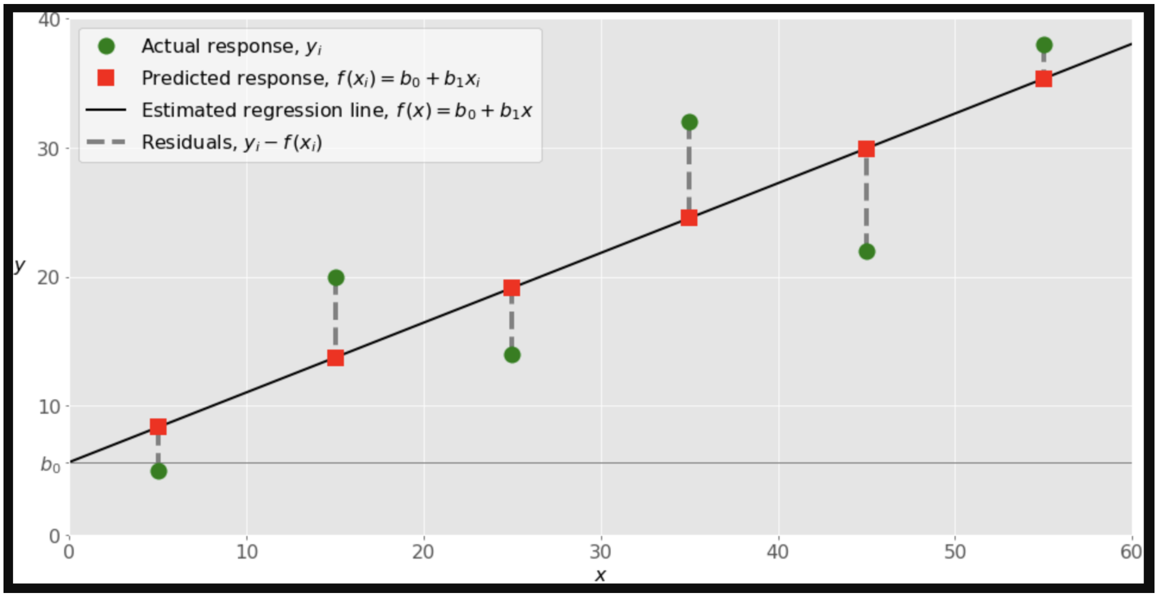
As the name of the machine learning algorithm suggest, linear regression solves regression problems. It is a good foundation algorithm to aid with your understanding of machine learning.

One of the objectives of a linear regression model is to determine a relationship between one or more features (independent variables) and a continous target variable (dependent variable).

If there is only one feature it is termed *univariate linear regression,* if there is more than one variable it is *multiple linear regression.*

The main idea of a linear regression algorithm is to achieve a line that best fits the data. The best fit line is the one which has the lowest total prediction error over all data points. The error is the distance between the point to the regression line.

We use the intercept and slope learned from the data to predict the outcome of new data.



The figure above shows a simple linear regression.

For a step by step tutorial on implementing linear regression in Python, visit: https://realpython.com/linear-regression-in-python/ (https://realpython.com/linear-regression-in-python/)

Resources:

https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86 (https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86)

Deep Learning Book- Chapter 5: http://www.deeplearningbook.org/contents/ml.html (http://www.deeplearningbook.org/contents/ml.html)

https://machinelearningmastery.com/linear-regression-for-machine-learning/ (https://machinelearningmastery.com/linear-regression-for-machine-learning/)

https://medium.com/velotio-perspectives/a-step-towards-machine-learning-algorithms-univariate-linear-regression-7b5f4bc9a01c (https://medium.com/velotio-perspectives/a-step-towards-machine-learning-algorithms-univariate-linear-regression-7b5f4bc9a01c)

Image from: https://realpython.com/linear-regression-in-python/ (https://realpython.com/linear-regression-in-python/)

Link to the Kaggle House Prices dataset: https://www.kaggle.com/c/house-prices-advanced-regression-techniques (https://www.kaggle.com/c/house-prices-advanced-regression-techniques)

If there is more than one variable, what is this type of regression called?

○ Univariate Linear Regression ● Multiple Linear Regression

Expected answer:

○ Univariate Linear Regression ● Multiple Linear Regression

✓

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 3

Naive Bayes Classifier

This algorithm is mostly used for tasks such as sentiment analysis, spam filtering and in recommendation systems.

The main idea behind a naive bayes classfier is the assumption that the features are independent- the presence of one particular feature is unrelated to the presence of any other feature- this is why it is termed Naive.

Reminder on Bayes Theorem:

Bayes theorem provides a method for us to calculate the probability of some data belonging to a given class, provided our prior knowledge.

$$P(class|data) = (P(data|class)^{\star} P(class))/P(data)$$

Where $P(class|data)$ is the probability of $class$ given the provided $data$

Example application:

We can use a naive bayes classifier to determine whether the day is suitable to play sport, given the weather.

We have the following dataset:

Weather	Temperature	Play
Rainy	Hot	No
Rainy	Cool	Yes
Sunny	Hot	Yes
Overcast	Hot	No
Sunny	Mild	Yes
Sunny	Mild	Yes

The columns represent the features and the rows represent the entries.

We assume that the features are independent, for example if the weather is sunny it does not neccesarily mean the temperature will be hot.

We can now rewrite Bayes theorem according to the dataset as:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

y = the class variable 'play'

X = the features

X can be given as - $X = (x_1, x_2, x_3, \ldots., x_n)$ x_i represents the features, e.g. weather, temp.

We can now use the dataset to answer the question- will sport be played if the weather is sunny?

We first create a frequency table from the dataset:

Weather	**No**	**Yes**
Rainy	1	1
Sunny		3
Overcast	1	
Total	2	4

We then create a likelihood table from the dataset:

Weather	**No**	**Yes**		
Rainy	1	1	= 2/6	0.33
Sunny		3	= 3/6	0.5
Overcast	1		=1/6	0.16
Total	2	4		
	= 2/6	=4/6		
	0.33	0.66		

We can then work out the solution to our question using the theorem discussed above:

$$P(Yes|Sunny) = P(Sunny|Yes)^{\star}P(Yes)/P(Sunny)$$

$$P(Sunny|Yes) = 3 \div 6 = 0.5$$

$$P(Sunny) = 3 \div 6 = 0.5$$

$$P(Yes) = 4 \div 6 = 0.66$$

$P(Yes|Sunny) = 0.5^{\star}0.66 \div 0.5 = 0.66$ Which is a higher probability.

Refer to the resources for examples of implementation in Python.

Resources and examples from:

https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c (https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c)

https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/ (https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/)

https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/ (https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/)

What task would you be most likely to use the Naive Bayes Classifier?

This is a multiple choice question

☑ Spam Filtering ☑ Sentiment Analysis ☑ Recommendation Systems

✓

Expected answer:

☑ Spam Filtering ☑ Sentiment Analysis ☑ Recommendation Systems

✓ You chose a correct answer. You were awarded **1** mark.

✓ You chose a correct answer. You were awarded **1** mark.

✓ You chose a correct answer. You were awarded **1** mark.

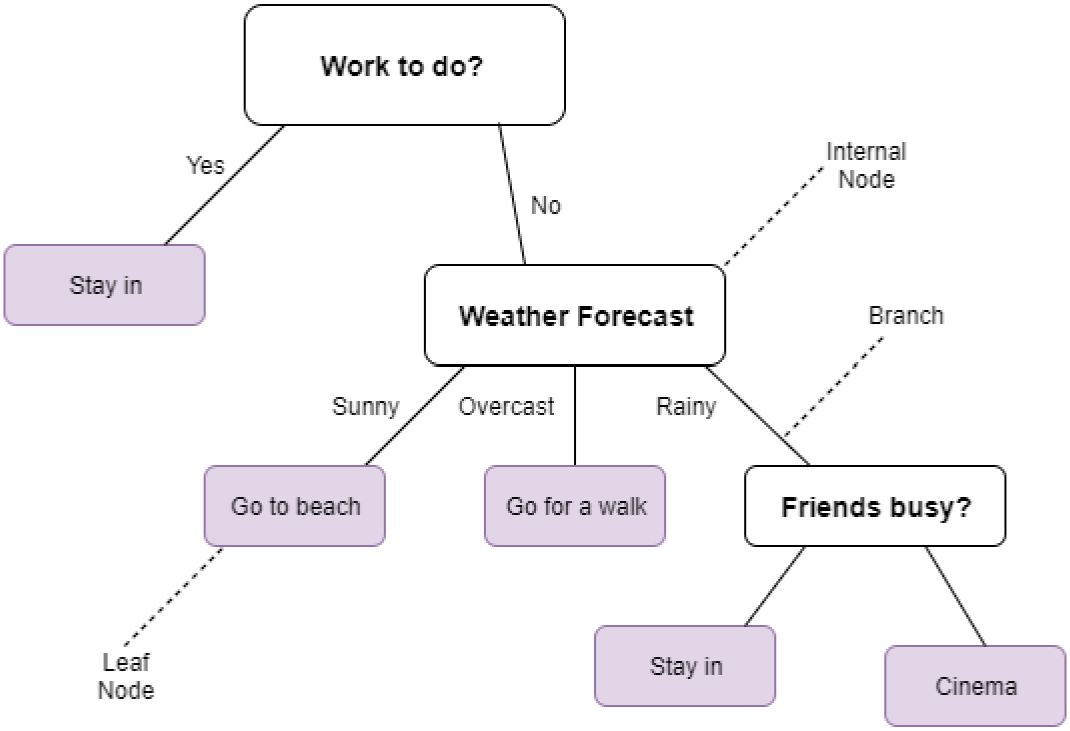
You scored **3** marks for this part.

Score: 3/3 ✓

Question 4

Decision Trees

We can think of decision trees as a model which breaks down our data by making decisions based on asking a series of questions. We want to build a tree with a set of hierarchical decisions which eventually give us a final result. The decisions will be chosen so that the tree is as small as possible, while aiming for high classification/regression accuracy.



Decision trees are created using two steps: **Induction** and **Pruning**.

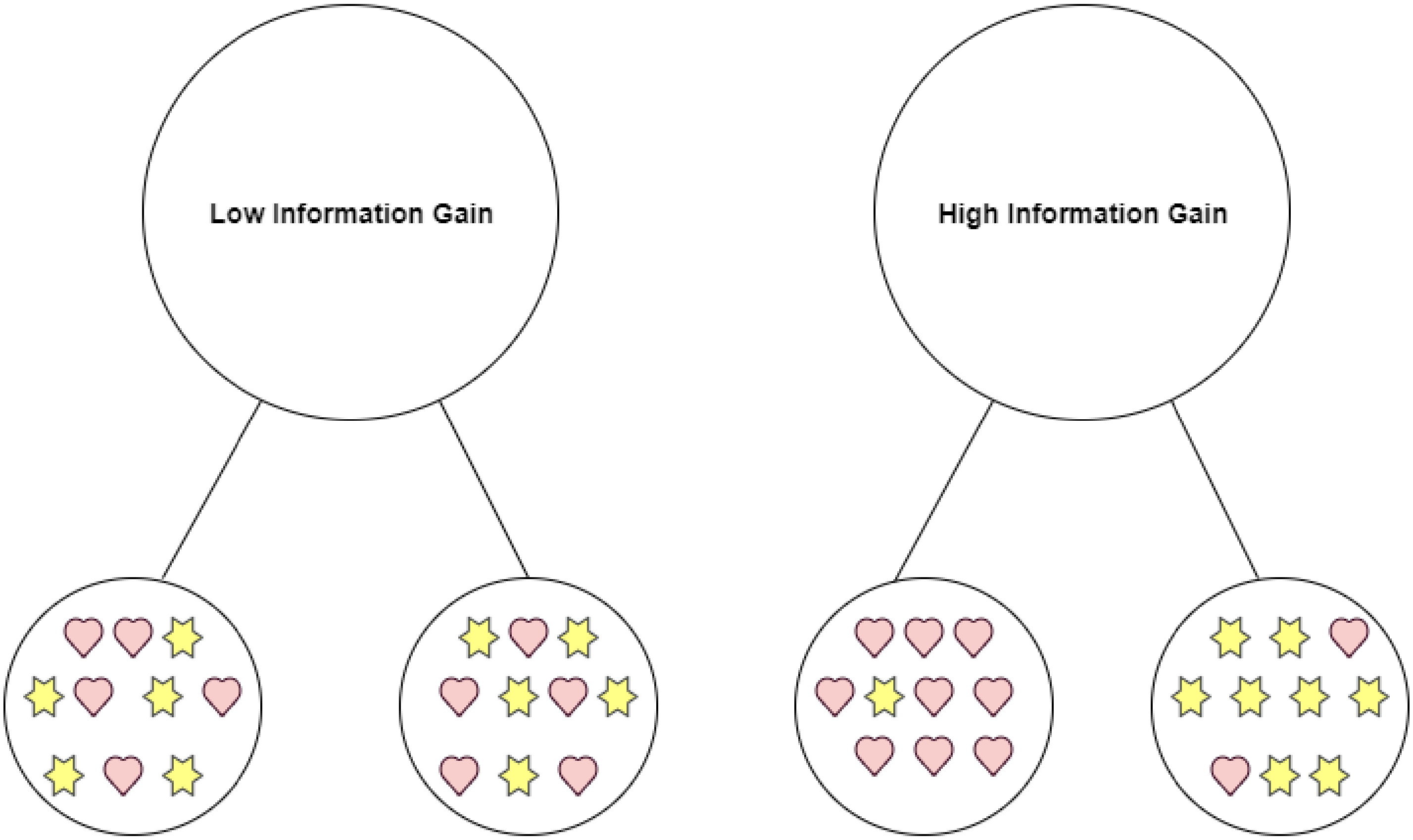
Induction - is the process where we actually build the tree (i.e. set all of the hierarchical decision boundaries based on the data)

We need to determine the "best feature" in the dataset on which to split the data on, this is usually chosen by using a greedy algorithm to minimise a cost function. This is an iterative process where we try out different split points and select the one with the lowest cost.

For *regression* we would most commonly use squared error as the cost function.

For *classification* we would more commonly use the Gini Index function.

Ideally, a node should have an error value of zero, meaning that each split outputs a single class- this is ideal becasue we know that once we get to that particular decision node, we know what exactly our output will be. The concept of having a single class per-split across our dataset is known as **information gain.** We start at the tree root and split the data on the feature that results in the largest information gain.



As shown in the figure above, if we choose a split where each output has a mix of classes, then we haven't gained any information (low information gain), as we still do not know whether a particular node (i.e. feature) has an influence in classifying our data.

If we have a high percentage of each class for the outputs in our split, then we have gained information (high information gain) that splitting in that particular way, on that feature gives us a particular output.

Pruning - Tree pruning is a technique that is necessary due to the training of decision trees being prone to overfitting.

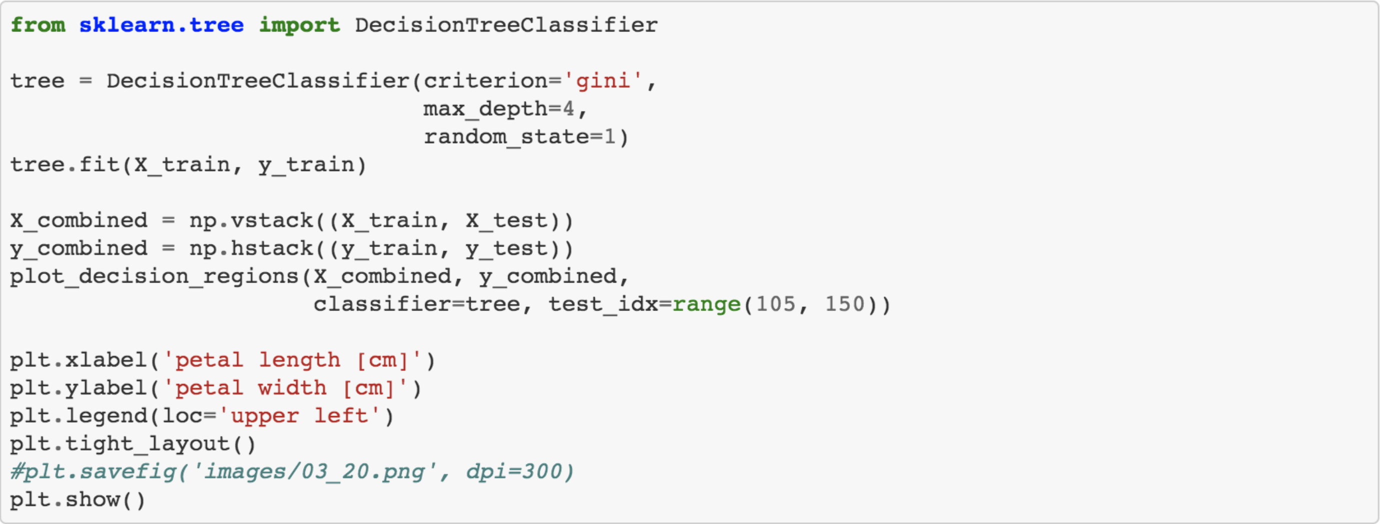
Large and complex decision trees may have many splits which are redundant and unnecessary, treepruning removes (prunes) the unnessesary parts of the tree.

Pruning compressess part of the tree from strict and rigid decision boundaries into ones that are smoother and generalise better, therefore reducing the tree complexity.

Building a decision tree in python

We can use `scikit-learn` to implement a decision tree.

Building a decision tree

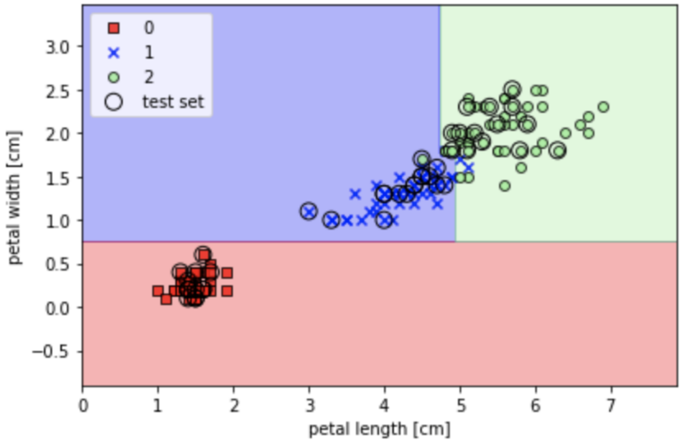


```
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(criterion='gini',
                              max_depth=4,
                              random_state=1)
tree.fit(X_train, y_train)

X_combined = np.vstack((X_train, X_test))
y_combined = np.hstack((y_train, y_test))
plot_decision_regions(X_combined, y_combined,
                      classifier=tree, test_idx=range(105, 150))

plt.xlabel('petal length [cm]')
plt.ylabel('petal width [cm]')
plt.legend(loc='upper left')
plt.tight_layout()
#plt.savefig('images/03_20.png', dpi=300)
plt.show()
```



Code example from: https://github.com/rasbt/python-machine-learning-book-2nd-edition/blob/master/code/ch03/ch03.ipynb (https://github.com/rasbt/python-machine-learning-book-2nd-edition/blob/master/code/ch03/ch03.ipynb)

Resourses and references:

Image altered from- S. Raschka , V. Mirjalili (2017) Python Machine Learning. Birmingham: Packt Publishing

https://towardsdatascience.com/a-guide-to-decision-trees-for-machine-learning-and-data-science-fe2607241956 (https://towardsdatascience.com/a-guide-to-decision-trees-for-machine-learning-and-data-science-fe2607241956)

https://scikit-learn.org/stable/modules/tree.html (https://scikit-learn.org/stable/modules/tree.html)

(https://towardsdatascience.com/a-guide-to-decision-trees-for-machine-learning-and-data-science-fe2607241956)What is the process called in which we build the decision tree?

● Induction ○ Inducement ○ Regression

✓ Expected answer:

● Induction ○ Inducement ○ Regression

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 5

Random Forests

Random forests are a popular model due to their scalability, ease of use and good classification performance. This model can be thought of as a collection of decision trees that work together as an ensemble. All of the trees in the random forest come up with a class prediction and the class with the majority of votes becomes the models prediction.

Random forests can be applied to both classification and regression problems.

One of the main ideas regarding random forests is that the individual trees need to be relatively uncorrelated with each other. There are two methods which can be used to ensure that each individual tree is not greatly correlated with any of the other trees in the model. These are:

Bagging (Bootstrap Aggregation)

With decision trees, any small changes to the training set can significantly change the tree structures. Random forests use this to their advantage by letting each individual tree randomly sample from the dataset with replacement, this results in differing trees. This is the process known as bagging.

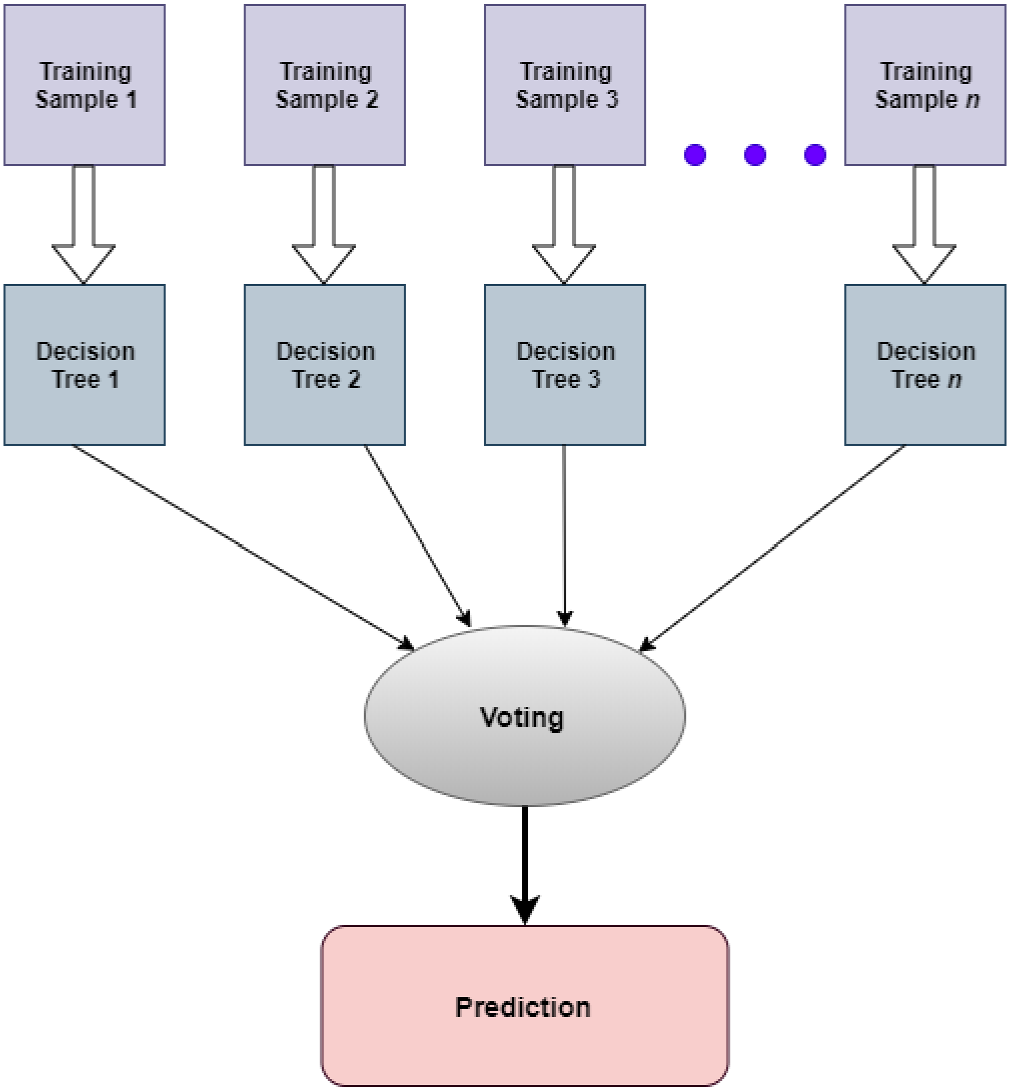
For example- from our training set we have a sample size of n and we are feeding each tree with the same sample size.

However instead of the original training data, we take a random sample with replacement.

So if we have a set of training data which is [5, 6, 7, 8, 9, 10], then we might give one of the trees the sample [6, 6, 5, 9, 9, 10]. Both sets are the same length, however some of the numbers are repeated as a consequence of random selection and replacement.

Feature Randomness

In a random forest, the trees are limited to picking from a random subset of features, this enforces even more variation among the trees, resulting in lower correlation. This differs from the process of splitting a node in decision trees where we consider all possible features and choose the one which produces the most separation.



Resources and references:

S. Raschka , V. Mirjalili (2017) Python Machine Learning. Birmingham: Packt Publishing

https://towardsdatascience.com/understanding-random-forest-58381e0602d2 (https://towardsdatascience.com/understanding-random-forest-58381e0602d2)

https://towardsdatascience.com/why-random-forest-is-my-favorite-machine-learning-model-b97651fa3706 (https://towardsdatascience.com/why-random-forest-is-my-favorite-machine-learning-model-b97651fa3706)

Image altered from-https://www.datacamp.com/community/tutorials/random-forests-classifier-python (https://www.datacamp.com/community/tutorials/random-forests-classifier-python)

Which machine learning model are random forests built upon?

○ Support Vector Machine ● Decision Tree ○ Naive Bayes Classifier

✓

Expected answer:

○ Support Vector Machine ● Decision Tree ○ Naive Bayes Classifier

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 6

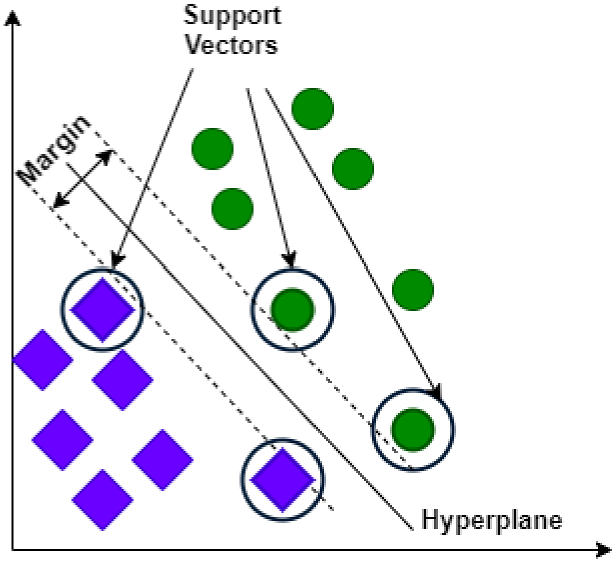
Support Vector Machine (SVM)

Support Vector Machines (SVM) can be applied to both classification and regression problems, however they are mostly used for classification purposes.

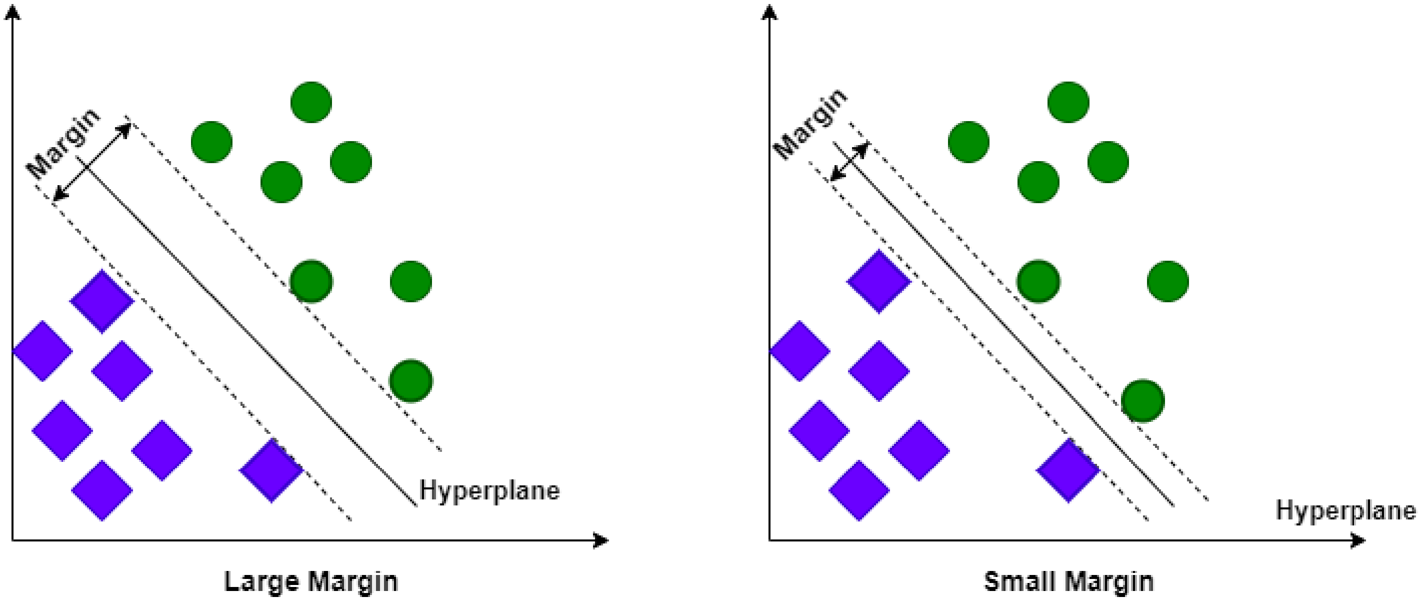
The main objective of the SVM is to find a hyperplane which clearly differentiates the data points (training samples). A hyperplane is a decision boundary which helps with the classification of data points. Datapoints which are positioned on opposite sides of the hyperplane can be thought of as belonging to different classes.

To separate the data points into different classes, many hyperplanes can be chosen, **the SVM aims to maximise the margin.** The margin is the difference between the decision boundary and the datapoints that are closest to the hyperplane. The datapoints closest to the hyperplane are called *support vectors*.

Support vectors can influence the hyperplane in relation to its position and have influence on determining the best position of the hyperplane. The data points that constitute the support vectors can be thought of as the most difficult to classify.



A loss function which helps maximise the margin is hinge loss- the cost is 0 if the predicted and actual value are the same, if the values are not the same we calculate the loss value. We also need to add a regularization parameter to blance the loss and margin maximisation.



The idea behind having a hyperplane with a large margin is that it tends to have lower generalisation error and is less prone to overfiiting. A large margin also gives more scope that future data points can be classified with greater confidence.

The support vector machine can be implemented in Python using `svm` from `scikit learn`.

Resources and References:

https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47)

https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/ (https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/)

http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf (http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf)

https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html (https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html)

https://scikit-learn.org/stable/modules/svm.html (https://scikit-learn.org/stable/modules/svm.html)

S. Raschka , V. Mirjalili (2017) Python Machine Learning. Birmingham: Packt Publishing

Which type of margin is less prone to overfitting?

● Large margin ○ Small margin

✓

Expected answer:

● Large margin ○ Small margin

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 7

Clustering

Clustering is an unsupervised learning technique which organises data into groups based on similarity which can expose relationships within your dataset. Clustering can be used for a number of applications including the grouping of films or music to a particular genre as well as for recommendation systems based on frequent customer purchase behaviour.

There are two types of clustering, hard and soft clustering.

Hard clustering assigns each sample in the dataset to only one cluster.

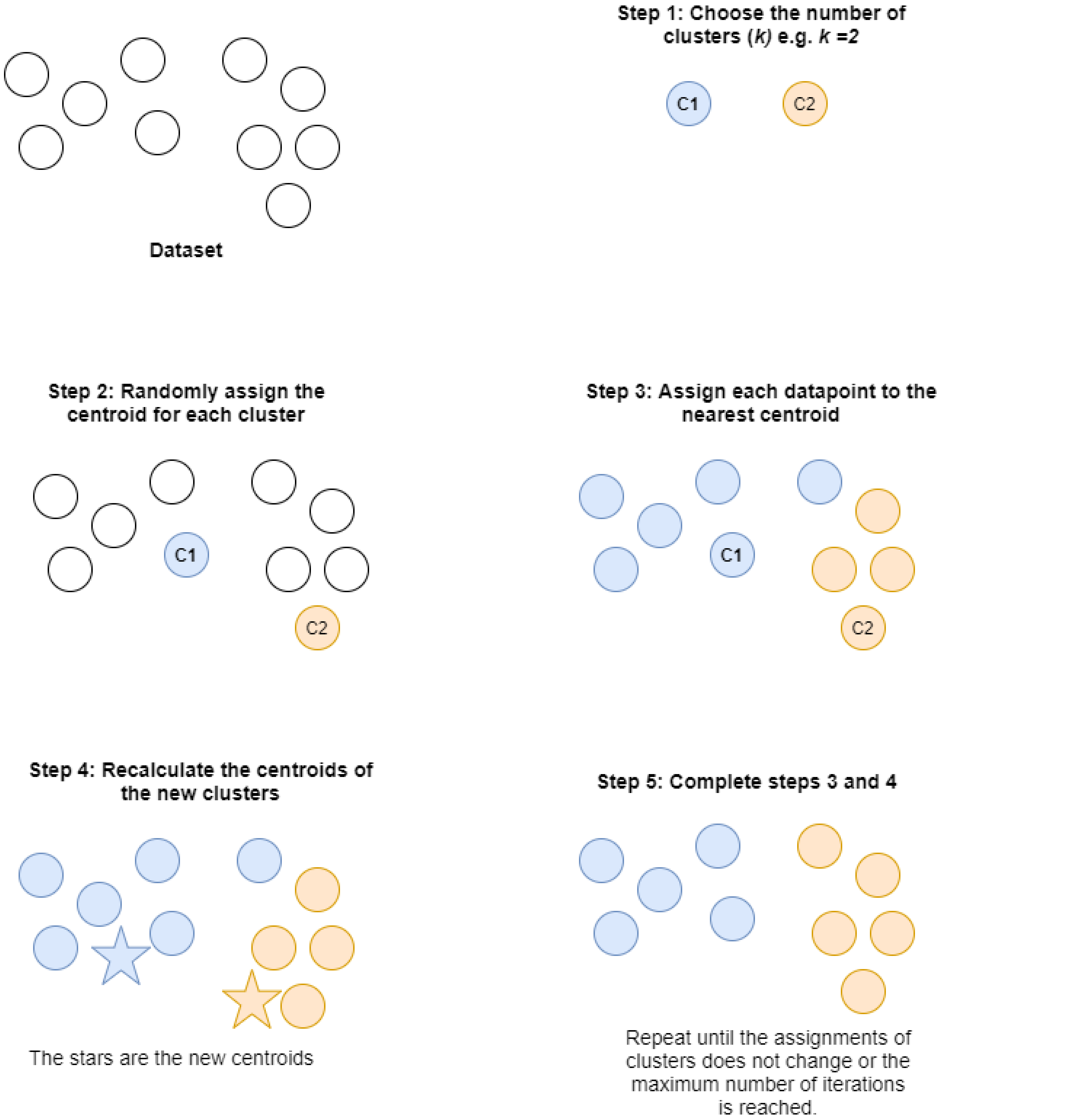
Soft clustering can assign each sample to more than one cluster.

K-Means

The K-means algorithm is an example of a hard clustering algorithm, in which the each individual sample gets assigned to only one cluster.

The K-means algorithm is iterative and attempts to separate the data into *k* pre-defined groups (clusters). These groups are independent- they do not overlap and each datapoint can only belong in one group.

The steps for implementing a K-means algorithm are set out in the diagram below:



Step 1: Choose the number of clusters (*k*) e.g. *k* =2

Step 2: Randomly assign the centroid for each cluster

Step 3: Assign each datapoint to the nearest centroid

Step 4: Recalculate the centroids of the new clusters

The stars are the new centroids

Step 5: Complete steps 3 and 4

Repeat until the assignments of clusters does not change or the maximum number of iterations is reached.

The K-means algorithm can be implemented in Python using the KMeans class from `scikit learn`.

Resources and References:

S. Raschka , V. Mirjalili (2017) Python Machine Learning. Birmingham: Packt Publishing

https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a (https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a)

Step diagram created and altered from: https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/ (https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/)

In the K-means algorithm, how many clusters can a sample be assigned to?

○ None ● One ○ Multiple

Expected answer:

○ None ● One ○ Multiple

✓

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 8

Dimensionality Reduction

The curse of dimensionality occurs when we have a large number of features. *Feature extraction* (projecting the data into a new feature space) can be used as an approach to dimensionality reduction/data compression with the aim of keeping the most relevant/important information.

Principle Component Analysis (PCA)

PCA is one method of dimensionality reduction, it is based on the use of linear-algebra algorithms.

PCA is a good method for identifying which variables can be removed from your dataset due to their low information influence. It can also be applied to determine if your variables are independent of one another.

PCA can also be used to determine any patterns in the data based on the correlation between the different features.

How to apply PCA

- Construct a covariance matrix - this matrix details how the variables in our dataset relate to one another.

  Postive covariance - the features increase or decrease together, there is a correlation.

  Negative covariance - the features vary in opposite directions, there is no correlation.

- Deconstruct the covariance matrix into its eigenvectors (direction) and eigenvalues (magnitude). The eigenvectors represents the *principal components*. We then know the direction of the data and and how important (magnitude) that direction is. As we want to reduce the dimensionality of our data, we only select the eigenvectors (principal components) that hold the most information (the variance).
- We can make an assumption that the more variance in a certain direction can indicate an important element in the data. Therefore we can drop the eigenvectors which appear relatively unimportant.

PCA can be applied in Python using the `sklearn PCA` class.

Resources and References:

S. Raschka , V. Mirjalili (2017) Python Machine Learning. Birmingham: Packt Publishing

https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c (https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c)

https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60 (https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60)

Which out of eigenvectors and eigenvalues represents the direction of the data?

● Eigenvector ○ Eigenvalue

✓

Expected answer:

● Eigenvector ○ Eigenvalue

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

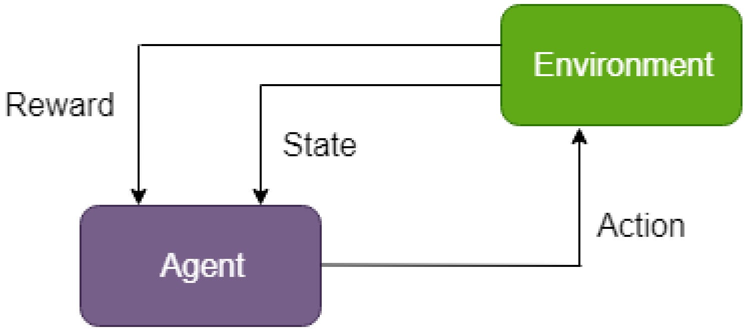
Score: 1/1 ✓

Question 9

Reinforcement Learning Algorithms

Reinforcement Learning Recap:

A reinforcement learning algorithm aims to develop a system (referred to as an agent) that improves its performance based on interactions with the environment.



Reinforcement algorithms can be used in a number of applications such as business strategy planning, aircraft control and robot motion control.

There are a number of terms within a reinforcement learning algorithm which are important to understand:

- **State:** the current state/position of the environment.
- **Agent:** attempts to maximise the reward through a series of interactions with the environment.
- **Action:** refers to the possibilities of what the agent can do in each state.
- **Reward:** the concept which describes feedback from the environment.
- **Policy:** the agent employs the policy to determine the next action which will be based upon the current state. A policy can be thought of as a type of strategy.
- **Value:** the expected long-term return with discount.
- **Q value or action value:** similar to value, however it takes an extra parameter which accounts for the current action.

Different Implementation Approaches

Value-Based: the aim is to try and maximise a value function.

Policy-Based: the aim is to try and create a policy which aids you in recieving maximum reward based upon the action performed in each state.

Model-Based: the creation of a virtual model is required for each environment and the aim is that the agent learns to perform in each specific environment.

Reinforcement Types

Positive reinforcement - an event which happens because of a specific behaviour. This then increases the strength and frequency of that behaviour and has a positive impact on the action taken by the agent.

Negative reinforcement - a stengthening of behaviour which has occurred because of a negative condition which should/has been stopped or avoided.

Example of a reinforcement learning algorithm

Q-Learning- is a value-based approach which informs the agent of what action it should take.

We create a Q table of the shape [state, action] where we store and update our Q value based upon the interactions with the environment.

Steps of a Q-Learning algorithm:

- Initialise the Q table
- Choose an action
- Perform the action
- Measure the reward
- Update the Q table

Resources and References:

https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-i-q-learning-sarsa-dqn-ddpg-72a5e0cb6287 (https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-i-q-learning-sarsa-dqn-ddpg-72a5e0cb6287)

https://www.guru99.com/reinforcement-learning-tutorial.html (https://www.guru99.com/reinforcement-learning-tutorial.html)

https://medium.com/@SmartLabAI/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc (https://medium.com/@SmartLabAI/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc)

https://www.geeksforgeeks.org/what-is-reinforcement-learning/ (https://www.geeksforgeeks.org/what-is-reinforcement-learning/)

https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56 (https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56)

What is the name of the concept which describes the long-term return with discount?

● Value ○ State ○ Agent

✓ Expected answer:

● Value ○ State ○ Agent

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Question 10

What is an Evolutionary Algorithm?

Evolutionary algorithms are so-called as they are loosely modelled on the process of natural selction. In an evolutionary algorithm, the members which are determined as the fittest will survive and generate, while members deemed unfit will no longer contribute to the gene pool of further generations.

One of the ways we can use an evolutionary algorithm is to find the best combination of elements that maximises a fitness function, we can decide on a solution once we have either ran the algorithm for a specified number of interations or we have reached a fitness threshold.

What is a fitness function?

The fitness function can be defined as a measure of how well an algorithm performs against its predictive goal.

Genetic Algorithms

Genetic algorithms are a type of evolutionary algorithm, they are a probabilistic search method where we consider a set of possible solutions for our problem and determine the best ones out of them.

In comparison to other types of machine learning algorithms, genetic algorithms rely more on blind search than information and assumptions.

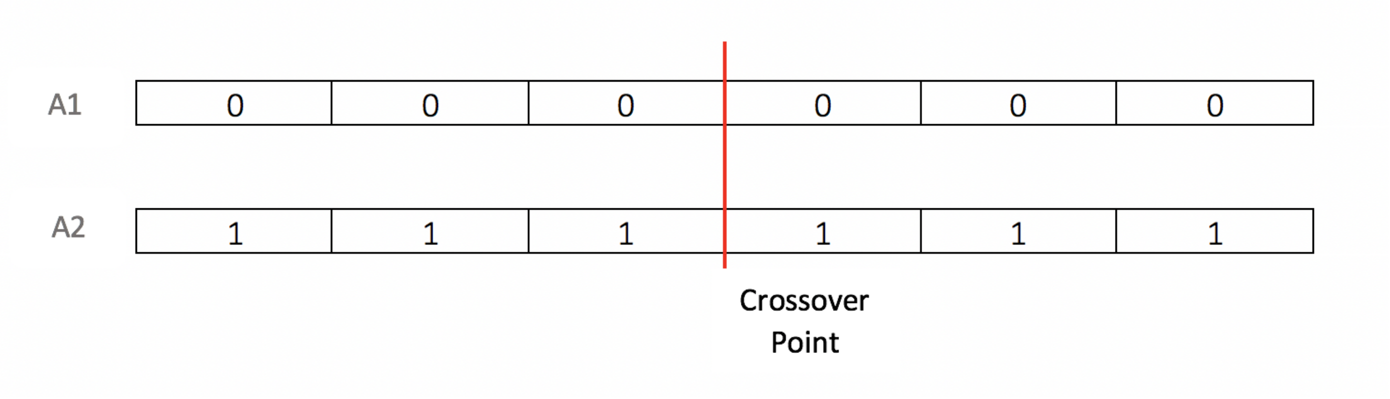
The process of the genetic algorithm begins by encoding the possible solutions to the problem as strings of characters, the strings are called *chromosomes* and the components of the strings are called *genes*.

The main task of the genetic algorithm is to find good strings. How good a string is, is represented by the fitness function.

A genetic algorithm works on a *population* of strings, starting with a random population which evolves into a population containing good strings. The main aspect of the algorithm are operations which take the population at the current generation and produce the population at the next timestep in a way which increases the overall fitness of the population.

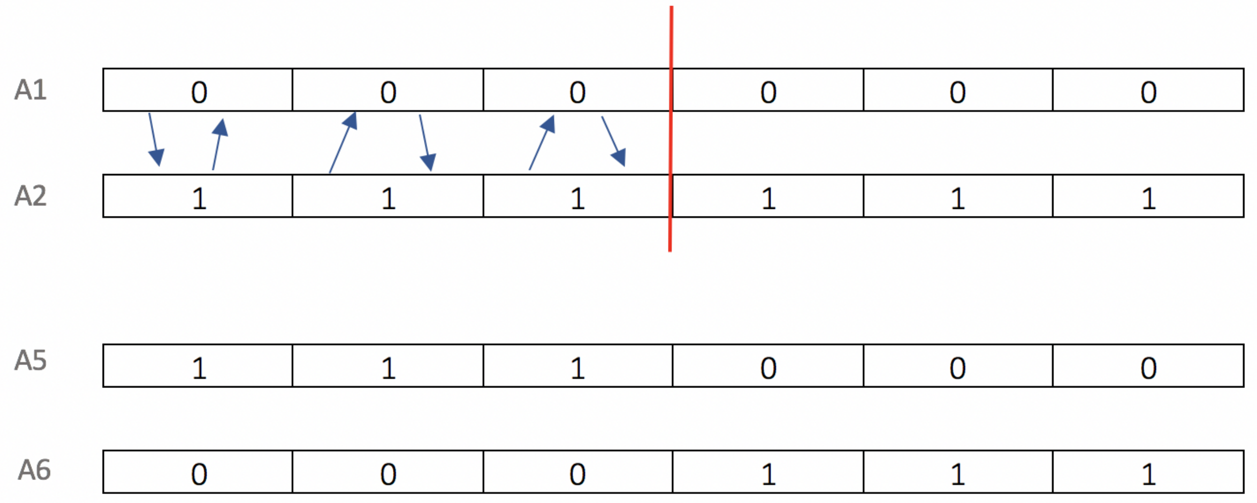
The next step in the algorithm is *selection.* This phase selects the fittest individuals (based on their fitness score) and allows them to pass their genes to the next generation.

A *crossover* point is then chosen at random from within the genes.



A1	0	0	0	0	0	0
A2	1	1	1	1	1	1

The next stage is to create *offspring / crossover* which are created by exchanging the genes of the parents until the crossover point is reached.

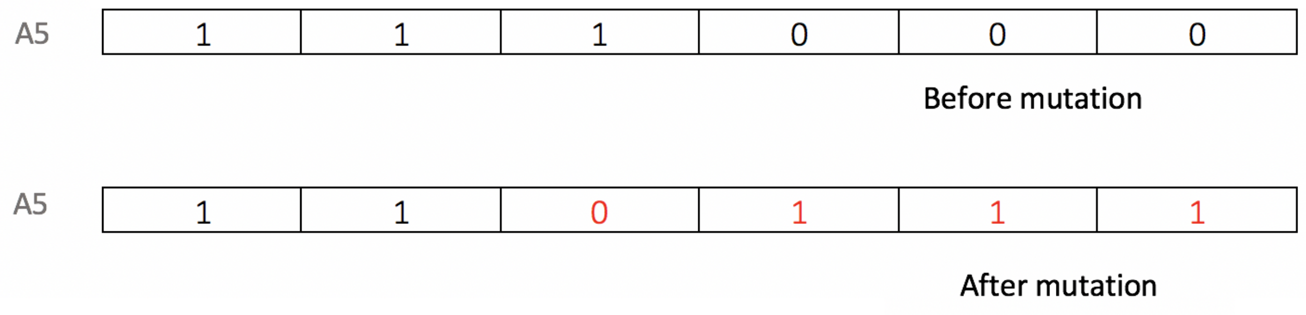


A1	0	0	0	0	0	0
A2	1	1	1	1	1	1

A5	1	1	1	0	0	0
A6	0	0	0	1	1	1

The offspring are added to the population

Mutation also occurs which changes characters in the *offspring* randomly. The characters are changed with a small probability.



A5	1	1	1	0	0	0

Before mutation

A5	1	1	0	1	1	1

After mutation

The genetic algorithm *terminates* when the population has converged (the offspring produced are no longer significantly different from the previous generation) - the genetic algorithm has now provided a set of solutions to our problem.

Pseudocode example of a simple genetic algorithm:

```
START
Generate the initial population
Compute fitness
REPEAT
  Selection
  Crossover
  Mutation
  Compute fitness
UNTIL Population has converged
STOP
```

When genetic algorithms are applied to learning they turn the task into a reinforcement learning problem, this is one of the main ways in which these types of algorithms are applied.

Advantage of genetic algorithms: by using a search method, a wide array of potential solutions can be tested, which may lead to the resolution of more difficult problems.

Disadvantage of genetic algorithms: by testing a wide array of potential solutions, this increases our computational cost.

Resources:

https://towardsdatascience.com/introduction-to-evolutionary-algorithms-a8594b484ac (https://towardsdatascience.com/introduction-to-evolutionary-algorithms-a8594b484ac)

https://skymind.ai/wiki/evolutionary-genetic-algorithm (https://skymind.ai/wiki/evolutionary-genetic-algorithm)
(https://skymind.ai/wiki/evolutionary-genetic-algorithm)

https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3 (https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3) (Images recreated from this source)

Genetic Algorithms in Machine Learning Paper: https://link.springer.com/content/pdf/10.1007/3-540-44673-7_7.pdf (https://link.springer.com/content/pdf/10.1007/3-540-44673-7_7.pdf)

What is the name of the process which changes characters in the offspring?

○ Termination ● Mutation ○ Selection

Expected answer:

○ Termination ● Mutation ○ Selection

✓

✓ You chose a correct answer. You were awarded **1** mark.

You scored **1** mark for this part.

Score: 1/1 ✓

Created using Numbas (https://www.numbas.org.uk), developed by Newcastle University (http://www.newcastle.ac.uk).