SECTION A

1. (a) Chess is a two-player strategy board game that is played on a checkered board with 64 squares arranged in an 8x8 grid. The game does not have any hidden information (i.e. both players are aware of the full state of the game board). Each player begins the game with 16 pieces; one of these pieces is called the king. The goal of the game is to checkmate the opponent's king by placing it under an inescapable threat of capture. A game of chess ends when a checkmate is reached, when one of the players resigns, or if a draw is declared. Games of chess often have a long duration (e.g. 100+ moves).

A fellow student is working on an agent to play chess, and has asked you to look over their code. You see that they have implemented the Q-learning algorithm, with the following parameter values:

- Learning rate = 0.1
- Discount factor = 0.0
- Exploration rate = 0.7

Evaluate the student's choices for each of the parameters above. If appropriate, recommend alternative parameter values. Justify your evaluation and recommended alternative parameter values (if any) by explaining the impact of each parameter on the Q-learning process. [12]

(b) Explain what is meant by the **Markov property**. Use the game of chess as an example in your answer. [3]

(c) You have received the following email message from a non-technical manager who works for a company that runs an online platform for discussion forums. Prepare a detailed reply.

"My company wishes to develop a system to identify forum posts that contain spam. The content of spam posts on our forums changes quite frequently, and currently our moderators spent a large amount of time reviewing forum posts. I have heard that machine learning models could be useful for automatic identification of suspicious posts. A colleague mentioned that machine learning algorithms may be categorised as ***lazy*** *or* ***eager*** *– could you tell me what the difference is? What are the main strengths and weaknesses of each of these two categories? Which category of algorithm would be more suitable for our application? It would also be helpful if you could provide a specific example of an algorithm from each category."* [10]

1) (a)

The discount factor controls how well future rewards are regarded of how important they are. The discount factor is in the range $[0,1]$. The greater it is, the most we value future rewards.
In the game of chess, the game doesn't finished until there is 'check mate'. Therefore, we should consider that reward the highest one. This means that the discount factor should be 1.0 and not 0.0.

The exploration rate is used to find alternatives moves to the ones we are considering good. This is a great idea as it makes sure that we are considering all possibilities. However, if we have the exploration rate set to 0.7, we are going to use a random move 70% of the time. The Q-algorithm should be using instead a low value such as 0.01. In this way, we would usually follow the path we think is optimal and provide an alternative move to explore 1% of the time.

The learning rate, α, ranges between 0 and 1. The higher the learning rate is, the less important we consider previous Q's. Because of this, is advisable to start with a high alpha, close to 1, and reduce with each iteration of the algorithm. This means that alpha will need to be updated during the algorithm and doesn't necessarily need to be fixed.

(b)

We say that a stochastic process has the Markov property if its next state depends only upon its current state, not any events that preceded current state. The game of chess is a clear example of a proces with Markov Property. In the n-th position of a game, the agent will need to based its best move on the current position. Previous moves on the $n-1, n-2, \ldots, 1$th positions do not have an impact on the decision from that round.

Because of this, a player could perform an optimal move on a game that has already started without the need of having the knowledge of previous moves.

(c)

If you decide to go for eager learning, it will take more time to build the model for first time, however the model will be able to provide predictions in a quicker way. In eager learning, the model summarises the data, therefore the model itself can be informative and helps to provide some useful information about how it is filtering spam. The only reason why I'd not recommend an eager learning in this case is due to 'concept drift' in which with time, the spam received may change and therefore we would need to retrain the model again with this new data.

On the other hand, in lazy learning, you will need a large space to store the model as if does not summarise the data. As well, this will make the model more susceptible to noise. The main advantadge of using lazy learning over eager for this application is 'concept drift'. As spam changes with time, there will not be any need to retrain the model.

In this case, I would recommend to use eager learning unless you have concerns on the time and money investment that may come when retraining the model in the future.

Examples of lazy: KNN - neighbours
Examples of eager: Naive Bayes Classifier

2. (a) Explain the difference between the following machine learning tasks: **classification**, **regression**. For each task type, list a specific application, and an algorithm which may be used to learn a suitable model. [7]

(b) Recommend a procedure to select a suitable value of the parameter k when applying the k Nearest Neighbours algorithm to a regression task. Your answer should also discuss the impact of using a weighting scheme. [6]

(c) A colleague with very little prior experience of machine learning has asked for your advice on selecting an appropriate **hypothesis language**. Describe to your colleague with the aid of diagrams **what a hypothesis language is**, and include discussions of relevant issues (including **underfitting** and **overfitting**). [12]

[PTO]

(2) (a)

In a classification task, we aim to predict the class of given data points. The target class is a discrete variable.
On the other hand, a regression task aims to predict a continuous outcome variable.

An example of application for a classification task would be trying to predict the music genere of a given song, e.g. jazz, rock, ...

An example of application for a regression task would be predicting the price of a house given features such as number of rooms, size, etc.

A suitable model for a classification task would be a linear classifier while for a regression task, we could use linear regression.

(b)

In a k-NN algorithm, using a low number for k could lead to underfitting while using a high number could overfit.
As we are using k-NN for a regression task, we can consider both, even and odd numbers. If we are not using a weighted algorithm, then a reasonable approach would be starting at a low number of neighbours, e.g. $k = 1$ and then increase k one at a time. At some stage we should see that increasing k does not improve the final score

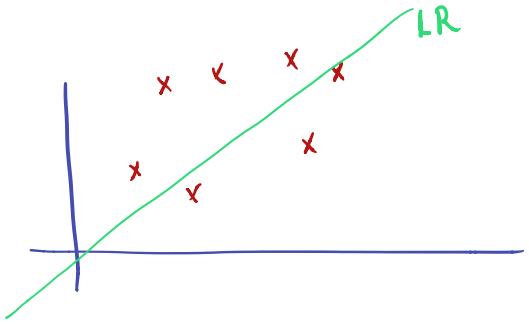
or improves it very slightly. In this case we can use the last one as our final candidate.

On the other hand, for a weighting scheme, neighbours that are far will have a lower impact on the final prediction. Because of this, it is fine using all neighbours at once.

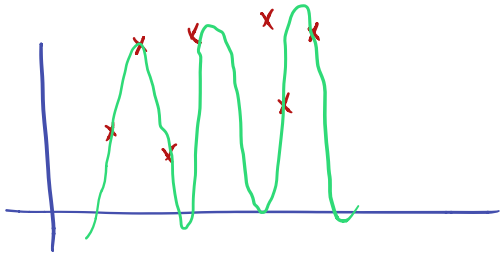
A1 This is application-dependant, it seems reasonable to take the best from the non-weighting scheme vs the weighting one.

CV

The hypothesis language is the language used to describe the hypothesis function. In this way, we can imagine that the more simple it is, the more difficult will be for our hypothesis function to exactly match the target. A clear example of this is the linear regression seen below:



As a linear regression uses a hypothesis language that is capable to use only hyperplanes, is not able to perfectly match datapoints that do not have a linear correlation between them. On the other hand, a polynomial regression would provide a more complex hypothesis language, the greater the polynomial degree is, the more complex it will be:



If using a degree enough large, we would be able to exactly predict all points.

In practice, a hypothesis language that is too simple will make the model underfit while a too complex one will make the model to overfit.

Therefore, ideally we are trying to find a balance in between. Bias of a model is usually related to simple hypothesis language while variance is related to complex hypothesis language.

3. (a) Explain what is meant by the term **Curse of Dimensionality** in the context of machine learning tasks.

Discuss how **heuristics** may be used to mitigate the effects of the Curse of Dimensionality. Your answer should include a specific example of a heuristic used in conjunction with an ML algorithm. [6]

(b) Identify a **scale invariant** method to measure similarity between instances in a dataset. Explain how this method may be applied using the aid of a diagram and an appropriate equation. [4]

(c) Briefly explain what is meant by the term **pure inductive learning**.

Describe in detail an algorithm which can be used for pure inductive learning of decision trees. Use detailed pseudocode and a diagram showing a sample decision tree to aid your explanation. [12]

(d) Decision tree learning can be "unstable". Explain what this means and why it happens. Briefly outline a modification that can reduce instability. [3]

(3) (a)

While adding more attributes to our dataset increases the amount of information we have, does not necessarily makes models more efficient. This is due to the fact that some attributes are more relevant than others. Most machine learning algorithms rely on having a good sampling density throughout the feature space, if this density is not high enough there is a good chance that our model will make worse predictions.

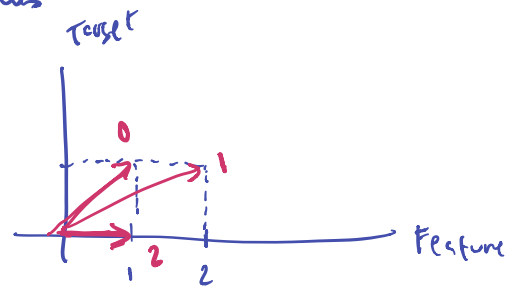
A good heuristic to mitigate this is to do feature selection. This is, selecting a subset of features that will maintain the prediction performance. In the case of a decision tree, information gain can be used to get the top X% features that have highest importance for this algorithm.

(b) The cosine similarity is an example of a scale invariant method to measure similarity between instances.

In the following dataset:

	Feature	Target
0	1	1
1	2	1
2	1	0

we can use the following vector representation of the examples



The cos similarity of two given examples can be calculated as $\cos(\theta_1, \theta_2) = \frac{|\theta_1 \theta_2|}{\|\theta_1\| \|\theta_2\|}$.

The closer this similarity is to 1, the more similar we expect the examples to be.

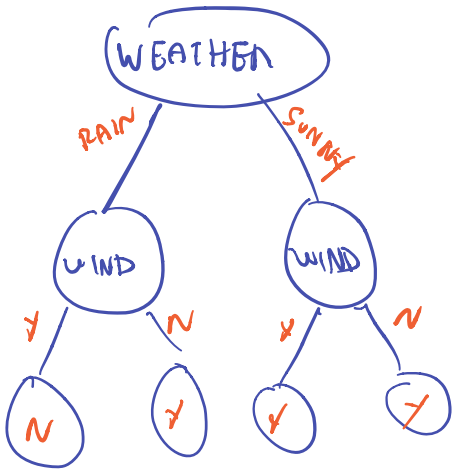
In this example, I would expect:

$$\cos(\theta_0, \theta_1) \geq \cos(\theta_3, \theta_2) > \cos(\theta_0, \theta_2)$$

(c)

In machine learning, pure inductive learning refers to the approach of building a hypothesis function only from observed data. In general, any supervised learning model is using pure inductive learning. A decision tree classifier would be an example of algorithm using pure inductive learning with decision trees.

WEATHER	WIND	TENNIS
RAIN	Y	N
SUNNY	Y	Y
SUNNY	N	Y
RAIN	N	Y



This decision tree classifier can be represented as seen in the diagram. Given a new example it can followed logical propositions based on the observations to give

a prediction for the example.

(d)

Instability of decision trees refers to the fact that the hypothesis function found is sensitive to training set used and therefore, adding or changing an example, could change the hypothesis function (structure of the tree).

One idea would be to alter the attribute selection procedure so that the tree learning algorithm is less sensitive to some % of the training dataset being replaced.

4. (a) A botanist is working on classifying trees as deciduous (lose their leaves in winter) or not. Describe in detail the process of constructing a learning curve. As part of the description, provide an example of a learning curve, ensuring that axes and all other parts are clearly labelled. In addition, explain for the botanist what can be determined from comparing two learning curves. [7]

(b) Are learning curves applicable to: (1) multi-class classification problems; (2) regression problems; (3) non-probabilistic classifiers? Explain your reasoning. [3]

(c) Referring back to the example of Part (a), describe in detail the process of constructing a ROC curve. Provide an example of a ROC curve, with axes and all other parts of the curve clearly labelled. Explain for the botanist what can be determined from comparing two ROC curves. [8]

(d) Are ROC curves applicable to: (1) multi-class classification problems; (2) regression problems; (3) non-probabilistic classifiers? Explain your reasoning. [3]

(e) Write a short note for the botanist on assembling a training dataset for the task, including what data are needed, the training dataset coverage, and how to know when enough data is collected. [4]

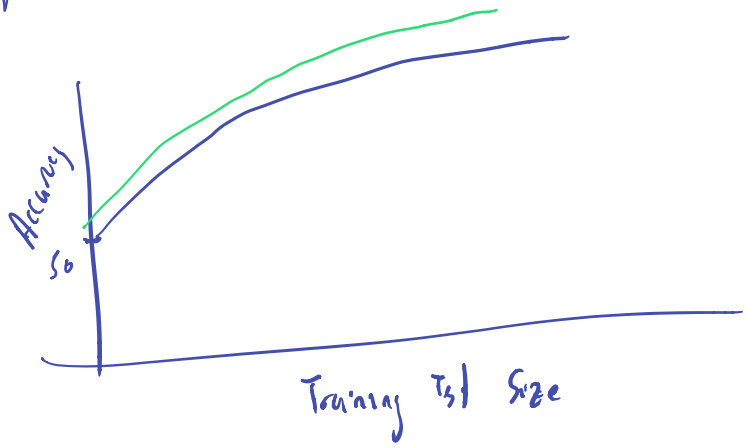
(4) (a)

A learning curve is a plot model learning performance over experience or time.

Repeat for various values of X from 0% to 100%:

- Repeat several times:
  - Train on X% of data (randomly sampled)
  - Test on the rest
- Average the results at X%.

Example



(b)

The concept of 'accuracy' of a model can be generalise to a multi-class classificatio- problem. This is the number of correct predictions of our model against the total size of the training set. Therefore, the previous steps could be apply to build a learning curve.

The same idea can be applicable to a regression problem, however the axis y will have to use a different metric such as root mean squared error. As in this case we would seek to reduce the RMSE. The curve may have a different shape, lower the better.

The learning curve doesn't use the probability of an example for a given class. It just uses the final prediction. Therefore, a non-probabilistic classifier could be equally used.

1c) In a binary classification, a probabilistic classifier usually uses by default a threshold of 0.5 to predict if an example belongs to a class or not.

This is, x will be classified as POS if $h_\theta(x) \geq 0.5$, otherwise x will be classified as NEG.

By predicting all examples of the training set, we get a confusion matrix

		PREDICTED CLASS: POS	PREDICTED CLASS: NEG
ACTUAL CLASS	POS	TP	FN
	NEG	FP	TN

This confusion matrix helps us to visualize the number of true positives, true negatives, false positives and false negatives. From here we can compute the TP Rate and FP rate, this is:

$$TP\% = TP/(TP+FN)$$

$$FP\% = FP/(FP+TN)$$

Now, the idea of ROC curve is based on changing this threshold and therefore getting different counts on the confusion matrix. By doing this we will get different values of TP% and FP%.

In our example:

True Positive = Trees that have been predicted as deciduous and that are deciduous

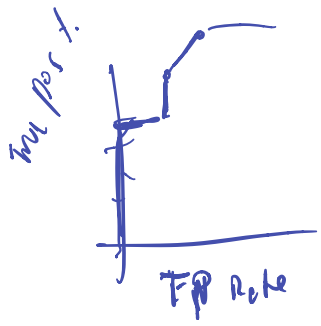
True Negative = Trees that have been predicted as not deciduous and that are not deciduous

False Positives: Trees that have been predicted as deciduous but are not deciduous

False negatives: Trees that have been predicted as not deciduous but are deciduous.

Let's imagine that after changing the threshold function we get the following table:

Threshold	FP	TP
1	0	0
0.9	0	30
0.8	0	50
0.7	10	50
0.6	10	70
0.5	20	80
0.4	40	80
0.3	40	100
0.2	70	100
0.1	86	100
0	100	100



Ideally we would like to maximize true positives and minimize false positive.

Comparing to ROC curve:

- ROC AUC
- closer ROC curve is to the upper corner

(d)

- They are not applicable to multi-class problems. Alternatives could be provided such as providing multiple ROC curves, one per class, where we consider class POS = being of class c_j, and class NEG = not being of class c_j.

- Not applicable to regression problems as they are totally based on confusion matrix

- Not applicable to non-probabilistic classifiers as we need to change threshold

(5)

(a)

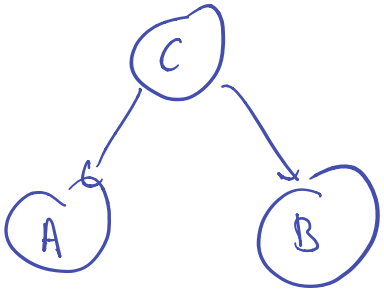
We say that an event A and B are independent if neither one influences or causes the other.

Given a third event C, we say that A and B are conditionally independent if the occurrence of A and the occurrence of B are independent events in their conditionally probability distribution given C.

In other words, given knowledge that C occurs, knowledge of whether A occurs provide no information on the likelihood of B occurring, and knowledge of whether B occurs, provides no information on the likelihood of A occurring.

In a bayesian network notation, A and B being independent could be represented by two non-related nodes:

while, A and B being conditionally independent given C could be represented as



(b)

An atomic event is an event which contains only a single outcome in the sample space.

An elementary proposition is a statement that can be true or false and is constructed by assigning a value to a random variable.

For a set of random variables, joint probability distribution gives probability of every atomic event on those variable.

A bayesian network takes into account the different dependencies between variables while the joint probability does not, as it considers all variables when calculating probabilities. (taking advantage of conditional independence)

(c)

	off-topic	on-topic
Funny	2	1
cats	3	1
video	2	0
jump	1	0
dogs	1	2
tree	1	2
classify	0	3
type	0	2
product	0	2
nuts	0	1

$3/10 \cdot 1/10 \cdot 1/10 = 3/1000$ ← prob in off-topic

$P(\text{cats jump tree}) = p(\text{cats})\,p(\text{jump})\,p(\text{tree})$

$P(\text{funny cats}) = p(\text{funny})\,p(\text{cats})$

$6/100 = 2/10 \cdot 3/10$ ← prob in on-topic

6) (a)

In order to predict the growth rate of future trees we could not be using a logistic regression. The reason for this is that a logistic regression is aimed to be used for a classification task. However, the target in this case is continuous.

A linear and polynomial regression could be used to predict the growth of future trees. However, if the input data has categorical variables, these should be first encoded to numerical values.

This may not be ideal for categorical variables that are nominal such as weather conditions - ideally we would like some kind of ordering.

Both algorithms are perfectly suitable for this case. Linear Regression has higher biased than polynomial and higher degree or polynomial will lead to overfitting.

(b)

Gradient descent has the purpose of finding a local minimum of any continuous differentiable function. It can be applied to multiple linear regression by minimizing the partials derivatives obtained from minimizing the mean squared error between hypothesis function and target, this is

$$\min_{\theta_1, \theta_2, \ldots, \theta_N} \frac{1}{2N} \sum_{i=1}^{N} \left(h_\theta(x_i) - y^{(i)} \right)^2 \quad \text{— cost function}$$

→ Initialize θ to any set of valid initial values

→ repeat till convergence (or time limit reached)

Simultaneously for each θ_j in θ do:

$$\theta \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

(c) Linear Classifier. Perceptron rule uses

Stochastic Gradient Descent