R249/419C

DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET, DUBLIN 8

BSc. (Honours) Degree in Information Systems/ Information Technology

Stage 4

SUPPLEMENTAL EXAMINATIONS 2014

*** *SOLUTIONS* ***

ARTIFICIAL INTELLIGENCE II

Dr. John Kelleher
Dr. Deirdre. Lillis
Mr. Rem. Collier

Duration: 2 Hours

Question 1 is **compulsory**

Answer Question 1 (40 marks) **and**

any 2 Other Questions (30 marks each).

****** SOLUTIONS ******

****** SOLUTIONS ******

1. (a) Explain what is meant by **inductive learning**.

(5 marks)

Inductive Learning involves the process of learning by example where a system tries to induce a general rule from a set of observed instances

(b) Distinguish between **supervised** and **unsupervised** learning.

(5 marks)

The distinction is that with **supervised learning** we know the actual label or category for each piece of data on which we train, whereas with **unsupervised learning** we do not know the classification of the data in the training sample. Unsupervised learning can thus often be viewed as a **clustering** task, while supervised learning can usually be seen as a **classification** task, or equivalently as a function-fitting task where one extrapolates the shape of a function based on some data points.

(c) Distinguish between **classification learning** and **regression learning**.

(5 marks)

- learning a discrete-valued function is called classification learning
- learning a continuous function is called regression.

(d) In the context of inductive learning explain what is meant by a **consistent hypothesis**.

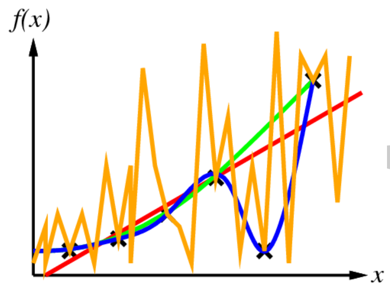
(5 marks)

A hypothesis is consistent if it agrees with the true function on all examples that we have.

(e) Inductive machine learning is often referred to as an **ill-posed problem**. What is meant by this?

(10 marks)

Inductive machine learning algorithms essentially search through a hypothesis space to find a the best hypothesis that is consistent with the training data used. It is possible to find multiple hypotheses that are consistent with a given training set (i.e. agrees with all training examples). It is for this reason that inductive machine learning is referred to as an ill-posed problem as there is typically not enough information in the training data used to build a model to choose a single best hypothesis. Inductive machine learning algorithms must somehow choose one of the available hypotheses as the *best*. An example like that shown in the figure below would be useful at this point



(f) Explain how do machine learning algorithms deal with the fact that machine learning is ill posed.

(10 marks)

Because inductive learning is ill-posed, we have to make some extra assumptions to have a unique solution with the data we have. The set of assumptions we make to have learning possible is called the **inductive bias** of the learning algorithm - this is the main implication of inductive machine learning being ill-posed.

Table 1: Dataset for the 3-Nearest Neighbor question

ID	Feature1	Feature2	Target
101	4	180000	C1
102	3	120000	C2
103	7	360000	C2
104	5	420000	C1
105	8	480000	C2

Table 2: Query instance for the 3-Nearest Neighbor question.

ID	Feature1	Feature2	Target
250	4	240000	?

2. (a) Table 1 lists a dataset containing examples described by two descriptive features, **Feature1** and **Feature2**, and labelled with a target class **Target**. Table 2 lists the details of a query for which we want to predict the target label. We have decided to use a **3-Nearest Neighbor** model for this prediction and we will use Euclidean distance as our distance metric:

$$d(x_1, x_2) = \sqrt{\sum_{i=1}^{n} \left((x_1.f_i - x_2.f_i)^2 \right)}$$

(i) What target class (**C1** or **C2**) will our **3-Nearest Neighbor** model label the query with? Provide an explanation for you answer.

(5 marks)

The first stage is to calculate the Euclidean distance between each of the examples and the query:

ID	Euclidean Distance
101	60000
102	120000
103	120000
104	180000
105	240000

From this table we can see that the three closest examples to the query are examples 101, 102, and 103. Example 101 has a target label of C1 and both 102 and 103 have target labels C2. Consequently C2 is the majority label in local model constructed by the 3-Nearest Neighbor classifier for this query instance and the query will be labelled with class C2.

(ii) There is a large variation in range between **Feature1** and **Feature2**. To account for this we decide to normalize the data. Compute the normalized data to four decimal points of precision using range normalization

$$x_i.f' = \frac{x_i.f - min(f)}{max(f) - min(f)}$$

(5 marks)

ID	Feature1	Feature2	Target
101	0.2	0.1667	C1
102	0	0.0000	C2
103	0.8	0.6667	C2
104	0.4	0.8333	C1
105	1	1.0000	C2

(iii) Assuming we use the normalized dataset as input, what target class (**C1** or **C2**) will our **3-Nearest Neighbor** model label the query with? Provide an explanation for you answer.

(5 marks)

The normalize query instance is:

ID	Feature1	Feature2	Target
250	0.2	0.3333	?

The Euclidean distances between the normalized data and normalized query are:

ID	Euclidean Distance
101	0.1667
102	0.3887
103	0.6864
104	0.5385
105	1.0414

From this table we can see that the 3 closest neighbors are: 101, 103 and 104. 101 and 104 are both labelled as class **C1**. So **C1** is the majority class in the neighborhood and the query will be labelled as belonging to it.

(b) Table 3 (on the next page) lists a data set with of 6 examples described in terms of 3 binary descriptive features (**A**, **B**, and **C**) and a target feature (**Target**). You are asked to create a decision tree model using this data. **Which of the descriptive features will the ID3 decision tree induction algorithm choose as the feature for the root node of the decision tree?** Support you anwer with appropriate calculations and dicussions of your results. Note that Table 4 (also on the next page) lists some equations that you may find useful for this question.

(15 marks)

The ID3 decision tree induction algorithm selects the decriptive feature with the highest information gain as the feature for the root node of the decision tree. The first step in calculating information gain is to calculate the entropy for the entire dataset:

$$\begin{aligned} H(DS) &= -\sum_{v\in\{C1,C2\}} p_v\, log_2 p_v \\ &= -\left(\frac{3}{6}\, log_2 \frac{3}{6}\right) + -\left(\frac{3}{6}\, log_2 \frac{3}{6}\right) \\ &= 1.00\, bits \end{aligned}$$

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

Split By	Feature Value	Partition	Examples	Entropy of Partition	Remainder	Info. Gain
A	1	DS_1	1,2,3	0.9183	0.9183	0.0817
	0	DS_2	4,5,6	0.9183		
B	1	DS_3	2,4,5,6	0.8113	0.5409	0.4591
	0	DS_4	1,3	0		
C	1	DS_5	1,2,3,4,6	0.9709	0.8091	0.1909
	0	DS_6	5	0		

From this table we can see the feature **B** has the highest information gain and consequently the ID3 algorithm will chose this feature as the feature tested at the root node of the tree.

ID	A	B	C	Target
1	1	0	1	C1
2	1	1	1	C2
3	1	0	1	C1
4	0	1	1	C2
5	0	1	0	C1
6	0	1	1	C2

Table 3: Dataset for the ID3 Algorithm Question

$$\begin{aligned} \text{Entropy(DS)} &= -\sum_{i=1}^{k} p_i \times log_2(p_i) \\ \text{Remainder(F)} &= \sum_{v\in Domain(F)} \frac{|DS_v|}{|DS|} Entropy(DS_v) \\ \text{InformationGain(F,DS)} &= Entropy(DS) - Remainder(F) \end{aligned}$$

Table 4: Equations from information theory.

Table 5: Query Document

Machine learning is fun

Table 6: Document counts from the corpus for the words in the query.

Document counts for the **dislike** data set

fun	is	machine	learning
415	695	35	70

Document counts for the **like** data set

fun	is	machine	learning
200	295	120	105

3. Lets assume we are given a set of **700** training documents that a friend has classified as **dislike** and another **300** documents that they have classified as **like**. We are now given a new document and asked to classify it. Table 5 lists the content of this query document and Table 6 gives the number of documents from each class (**dislike** and **like**) that the words in the query document occurred in. **What class will a Naive Bayes prediction model label the query document as belonging to?** (You must support your answer by showing the calculations that a Naive Bayes model will make.)

(30 marks)

A Naive Bayes model will label the query with the class 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 class given the evidence and assuming conditional independence across the evidence.
To carry out these calculation we need to convert the raw documents counts into conditional probabilities by dividing each count by the total number of documents occuring in class:

$P(w_k|C = dislike)$

w_k	Count	$P(w_k \vert C = dislike)$
fun	415	$\frac{415}{700} = .593$
is	695	$\frac{695}{700} = .99$
learning	35	$\frac{35}{700} = .05$
machine	70	$\frac{70}{700} = .10$

$P(w_k|C = like)$

w_k	Count	$P(w_k \vert C = like)$
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 probabilites of each class:

$P(dislike|Query\ Document)$

$$\begin{aligned} &= P(dislike) \times P(Machine|dislike) \times P(learning|dislike) \times P(is|dislike) \times p(fun|dislike). \\ &= 0.7 \times 0.593 \times 0.99 \times 0.5 \times 0.1 \\ &= 0.00205 \end{aligned}$$

$P(like|Query\ Document)$

$$\begin{aligned} &= P(like) \times P(Machine|like) \times P(learning|like) \times P(is|like) \times p(fun|like). \\ &= 0.3 \times 0.667 \times 0.983 \times 0.4 \times 0.35 \\ &= 0.00275 \end{aligned}$$

As $P(like|Query\ Document) > P(dislike|Query\ Document)$ the naive bayes classifier will return a label of **like**.

4. (a) The following model is commonly used for continuous prediction tasks:

$$y(x) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

(i) Provide the name for this model and explain all of the terms that it contains.

(5 marks)

Students should explain that this is a simple linear regression model which can be effectively used to make predictions. x is a vector of feature values for a query instance and w is a vector of feature weights. An diagram of a simple one dimensional linear function would help.

(ii) Explain how the following model can overcome some of the limitations of the model given above.

(8 marks)

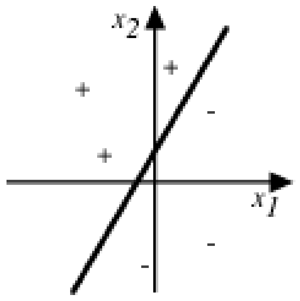
$$y(x) = \sum_{j=0}^{M-1} w_j \phi_j(x)$$

Students should explain that the simple linear regression model is attractive because it is linear with respect to w but has severe limitations because it is also linear with respect to x. These greatly limits the kinds of predictions that this model will be able to make. However, the introduction of *basis functions*, shown as ϕ above, goes some way towards solving this problem. The introduction of a non-linear basis function means that models can be made non-linear functions of input x but remain linear in w which makes them computationally easier to solve.

Students might give the example of polynomial regression in which $\phi_j(x) = x^j$ or some other suitable example.

(b) Explain why a **linear threshold perceptron** can represent the AND and OR functions but not the XOR function.

(7 marks)



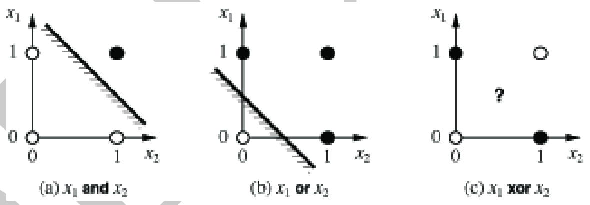
We can think of a 2 input threshold perceptron as representing a line separator in 2D input space.In other words, the function $\sum_j W_j a_j > 0$ or $\vec{w} \bullet \vec{a} > 0$ defines a line in the input space and the perceptron outputs a 1 for instances lying on one side of the line and -1 for instances lying on the other side of the line.

The figure above illustrates this concept. The equation for the line in this image is $\vec{w} \bullet \vec{a} > 0$. The inputs into the perceptron are x_1 and x_2; i.e. $\vec{a} = \{x_1, x_2\}$.

This concept of a linear perceptron defining a line scales up into higher dimensional input space. In 3D and higher inputs space (i.e. in situations where the linear perceptron has 3 or more inputs), the linear activation function $\vec{w} \bullet \vec{a} > 0$ defines a hyperplane decision surface in the n-dimensional space of inputs.

Data-sets of positive and negative examples that can be separated by a hyperplane are called *linearly separabe*.

Of course, not all data-sets of positive and negative examples are linearly separable. The XOR function is one example of a non-linearly separable function.



The figure above illustrates this by comparing it with the AND and OR function which are linearly separable. In this figure black dots represent a point in the input space where the value of the function is 1, and white dots indicate a point where the value is 0.

As is evident from the images, in the AND and OR inputs spaces it is possible to draw a line that separates the black dots from the white dots. However, in the XOR space no such line exists. *As a result, a threshold perceptron cannot represent the XOR function.*

(c) Describe how perceptron training can be viewed as gradient descent search through an error landscape.

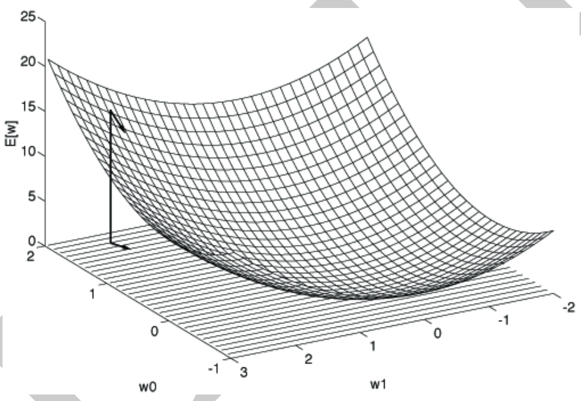
(10 marks)

Perceptrons learn by modifying the weights associated with their inputs in such a way as to minimize the error between the perceptrons output and the output expected by the training examples.

As a result the set of possible weight vectors that the perceptron can adopt during training can be viewed as its hypothesis space. In the figure below the axes w_0 and w_1 represent possible values for two weights of a simple linear unit. The w_0, w_1 plane therefore represents the entire hypothesis space.

Each position in this hypothesis space will have an associated error with. Mapping the error values as a error surface above the hypothesis allows us to view the error surface. In the figure below the the vertical axes indicates the error relative to some fixed set of training examples. The error surface shown in the figure thus summarises the desirability of every weight vector in the hypothesis space.

Perceptron learning adjusts the weights of the perceptron to minimise error. Consequently, it can be viewed as a gradient descent search for through the hypothesis space for the hypothesis (weight vector) with minimum error (i.e. the location in the hypothesis space whose associated error value is the global minimum in the error surface).


E[w]
25
20
15
10
5
0
w0
2
1
0
-1
w1
3
2
1
0
-1
-2