

---

## COMP 2211 Midterm Exam - Spring 2022 - HKUST

---

Date: April 2, 2022 (Saturday)

Time Allowed: 2 hours, 2:00–4:00 pm

- Instructions:
1. This is a closed-book, closed-notes examination.
  2. There are 7 questions on **19** pages.
  3. Write your answers in the space provided.
  4. All programming codes in your answers must be written in the Python version as taught in the class.
  5. For programming questions, unless otherwise stated, you are **NOT** allowed to define additional classes, helper functions and use global variables, nor any library functions not mentioned in the questions.

Student Name	<b>SOLUTIONS</b>
Student ID	
Email Address	

For T.A.  
Use Only

Problem	Topic	Score
1	True/False Questions	/ 15
2	Python Fundamentals	/ 16
3	Conditional Probability and Bayes Classifier	/ 11
4	K-Nearest Neighbors	/ 14
5	K-Means Clustering	/ 12
6	Perceptron	/ 20
7	Perceptron and Multilayer Perceptron	/ 12
	Total	/ 100

### Problem 1 [15 points] True/False Questions

Indicate whether the following statements are true or false by putting **T** or **F** in the given table. You get 1.5 points for each correct answer.

- (a) Machine learning is a sub-field of artificial intelligence.
- (b) Tensorflow is easy to use and less flexible than Keras.
- (c) Suppose we wish to calculate  $P(B|e_1, e_2)$  and we have no conditional independence information. Having  $P(e_1, e_2)$ ,  $P(B)$ ,  $P(e_1, e_2|B)$  are sufficient for the calculation.
- (d) Training a K-nearest neighbors classifier takes more computational time than applying it.
- (e) K-nearest neighbors cannot be used for regression.
- (f) Consider D-fold cross-validation. A higher number of folds (i.e. larger value of D), the estimated error will be lower on average.
- (g) K-means clustering algorithm is guaranteed to converge.
- (h) Consider a two-class classification problem. Suppose we have trained a perceptron model on a linearly separable training set, and now we get a new labeled data point which is correctly classified by the model, and far away from the decision boundary. If we add this new point to our earlier training set and re-train with the same set of initial weights and bias, the learnt decision boundary will be changed for sure.
- (i) Gradient descent is usually **NOT** guaranteed to converge at global minimum.
- (j) If the learning rate is too small, gradient descent may take a very long time to converge and computationally expensive.

Question	Answer (T/F)
(a)	T
(b)	F
(c)	T
(d)	F
(e)	F
(f)	T
(g)	T
(h)	F
(i)	T
(j)	T

## Problem 2 [16 points] Python Fundamentals

For each of the Python expressions below, write the output when the expression is evaluated. If the output is an empty array, write “Empty Array”. If an error occurs, write “Error”.

(a) [5 points] Consider the following NumPy arrays:

```
import numpy as np
A = np.array([10, 20, 30, 40, 50, 60, 70, 80, 90])
B = np.array([ [0, 1, 2, 3],
               [4, 5, 6, 7],
               [8, 9, 10, 11],
               [12, 13, 14, 15] ])
```

(i) `print(A[2:6:3])`

**Answer:** [30 60]

(ii) `print(B[0:2, 1:3])`

**Answer:**

```
[[1 2]
 [5 6]]
```

(iii) `print(A[-1:-3])`

**Answer:**

Empty Array

(iv) `print(B[:, :-1])`

**Answer:**

```
[[12 13 14 15]
 [ 8  9 10 11]
 [ 4  5  6  7]
 [ 0  1  2  3]]
```

(v) `print(B[:3, 2:])`

**Answer:**

```
[[ 2  3]
 [ 6  7]
 [10 11]]
```

- (b) [2 points] What is the output of the following code?

```
import numpy as np
X = np.array([2,2,0,2,2,0,1,1,0,1,1,0])
Y = X[X != 0]
print(Y[:2])
```

**Answer:**

```
[2 2 1 1]
```

- (c) [9 points] Given the following code which computes the distance between each training point in  $X_{train}$  and each test point in  $X_{test}$  using a nested loop over both the training data and test data.

```
import numpy as np

def compute_distances_nested_loops(X_train, X_test):
    num_test = X_test.shape[0]
    num_train = X_train.shape[0]
    distances = np.zeros((num_test, num_train))
    # --- BLOCK TO REWRITE ---
    for i in range(num_test):
        for j in range(num_train):
            distances[i,j] = np.sqrt(np.sum(np.square(X_test[i,:]-X_train[j,:])))
    # --- BLOCK TO REWRITE ---
    return distances
```

```
Train = np.array([[0,1], [1,2], [2,3], [3,4]])
Test = np.array([[5,6], [7,8]])
print(compute_distances_nested_loops(Train, Test))
# Output:
# [[7.07106781 5.65685425 4.24264069 2.82842712]
#  [9.89949494 8.48528137 7.07106781 5.65685425]]
```

Rewrite the block of code between the comment lines `# --- BLOCK TO REWRITE ---` using no explicit loops in the space provided.

Hints:

- To compute the distance between training data point (0, 1) and test data point (5, 6), we do

$$dist = (0 - 5)^2 + (1 - 6)^2$$

- Expand it

$$\begin{aligned}
 dist &= 0^2 - 2(0)(5) + 5^2 + 1^2 - 2(1)(6) + 6^2 \\
 &= 0^2 + 1^2 + 5^2 + 6^2 - 2(0)(5) - 2(1)(6) \\
 &= 0^2 + 1^2 + 5^2 + 6^2 - 2((0)(5) - (1)(6))
 \end{aligned}$$

You may find the following functions useful for this question.

- Dot product of two arrays:  
`numpy.dot(a, b)`
  - It returns the product of matrix multiplication.
- Return the element-wise square of the input  
`numpy.square(x)`
  - x is the input data
- Return the sum of array elements over a given axis.  
`numpy.sum(a, axis=None)`
  - a is an array with elements to sum.
  - axis: None or int or tuple of ints. axis = 0 means along the column and axis = 1 means working along the row.
- Return the non-negative square-root of an array, element-wise.  
`numpy.sqrt(x)`
  - x is the array with values whose square-roots are required.
- Return a matrix (or a 2D array) from an 1D array.  
`numpy.matrix(data)`
  - data is the 1D array.
  - Example: `numpy.matrix([1, 2, 3])`, output is `[[1, 2, 3]]`
- Insert a new axis that will appear at the axis position in the expanded array shape.  
`numpy.expand_dims(a,axis)`
  - a is the input array.
  - axis is an int or tuple of ints that represents position in the expanded axes where the new axis (or axes) is placed.

### Answer:

```

M = np.dot(X_test, X_train.T)
te = np.square(X_test).sum(axis=1)
tr = np.square(X_train).sum(axis=1)
dists = np.sqrt(-2*M + np.matrix(tr) + np.matrix(te).T)

```

Alternative answer:

```

distances = np.sqrt(np.sum(np.square(np.expand_dims(X_test, 1) - X_train), axis=2))
distances = np.sqrt(np.sum(np.square(X_test[:,None] - X_train[None]), axis=-1))

```

### Problem 3 [11 points] Conditional Probability and Bayes Classifier

(a) [2 points] Given the following probabilities:

- $P(\text{Good course} \mid \text{Desmond is in the course}) = 0.5$
- $P(\text{Good course} \mid \text{Desmond is not in the course}) = 0.3$
- $P(\text{Desmond in a randomly chosen course}) = 0.1$

What is  $P(\text{Desmond is in the course} \mid \text{Not a good course})$ ? If your answer is not an integer, write your answer in fraction form (use / to separate your numerator and denominator), e.g.  $1/2$ .

**Answer:**

Let  $D$  be Desmond is in the course,  $G$  be a good course

$$\begin{aligned} P(D \mid \text{NOT } G) &= \frac{P(D, \text{NOT } G)}{P(\text{Not } G)} \\ &= \frac{P(\text{Not } G \mid D)P(D)}{P(\text{Not } G \mid D)P(D) + P(\text{Not } G \mid \text{Not } D)P(\text{Not } D)} \\ &= \frac{(1 - 0.5) \times 0.1}{(1 - 0.5) \times 0.1 + (1 - 0.3) \times (1 - 0.1)} \\ &= \frac{5}{68} \end{aligned}$$

(b) [7 points] Suppose you are given the following set of data with three Boolean input variables  $A$ ,  $B$ ,  $C$ , and a single Boolean output variable  $D$ .

A	B	C	D
1	0	1	1
1	1	1	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	1
0	0	0	1
0	0	1	0

- (i) According to the Naive Bayes classifier, what is  $P(D = 1 \mid A = 1, B = 1, C = 0)$ ? If your answer is not an integer, write your answer in fraction form (use / to separate your numerator and denominator), e.g.  $1/2$ .

**Answer:**

$$\begin{aligned} P(D = 1|A = 1, B = 1, C = 0) &= \frac{P(D = 1)P(A = 1|D = 1)P(B = 1|D = 1)P(C = 0|D = 1)}{\sum_{j=0}^1 P(D = j)P(A = 1|D = j)P(B = 1|D = j)P(C = 0|D = j)} \\ &= \frac{(4/8)(2/4)(1/4)(2/4)}{(4/8)(2/4)(2/4)(1/4) + (4/8)(2/4)(1/4)(2/4)} = \frac{1}{2} \end{aligned}$$

- (ii) According to the Naive Bayes classifier, what is  $P(D = 0|A = 1, B = 1)$ ? If your answer is not an integer, write your answer in fraction form (use / to separate your numerator and denominator), e.g. 1/2.

**Answer:**

$$\begin{aligned} P(D = 0|A = 1, B = 1) &= \frac{P(D = 0)P(A = 1|D = 0)P(B = 1|D = 0)}{\sum_{j=0}^1 P(D = j)P(A = 1|D = j)P(B = 1|D = j)} \\ &= \frac{(4/8)(2/4)(2/4)}{(4/8)(2/4)(2/4) + (4/8)(2/4)(1/4)} = \frac{2}{3} \end{aligned}$$

- (iii) According to the general Bayes classifier (i.e. without independence assumption), what is  $P(D = 1|A = 1, B = 1, C = 0)$ ? If your answer is not an integer, write your answer in fraction form (use / to separate your numerator and denominator), e.g. 1/2.

**Answer:**

$P(D = 1|A = 1, B = 1, C = 0) = 0$  as there is no data with  $A = 1, B = 1, C = 0$  and  $D = 1$  in the dataset.

- (iv) According to the general Bayes classifier (i.e. without independence assumption), what is  $P(D = 1|A = 1, B = 1)$ ? If your answer is not an integer, write your answer in fraction form (use / to separate your numerator and denominator), e.g. 1/2.

**Answer:**

$P(D = 1|A = 1, B = 1) = 1/2$  as number of data with  $A = 1, B = 1, D = 1$  is 1, and number of data with  $A = 1, B = 1$  is 2.

- (c) [2 points] The Naive Bayes algorithm selects the class  $c$  for an example  $x$  that maximizes  $P(c|x)$ . Suppose one of your classmates stated that it is equivalent to selecting the  $c$  that maximizes  $P(x|c)$  under an assumption. What is the assumption that he has made?

**Answer:**

$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$ , so finding the  $c$  that maximizes  $P(c|x)$  is equivalent to finding  $c$  that maximizes  $P(x|c)$ , if the prior  $P(c)$  is uniform.

**Problem 4 [14 points] K-Nearest Neighbors**

- (a) [3 points] Consider a set of 5 training data given as  $((x_1^{\text{Train}}, x_2^{\text{Train}}), c^{\text{Train}})$  values, where  $x_1^{\text{Train}}$  and  $x_2^{\text{Train}}$  are the two attribute values (positive integers) and  $c^{\text{Train}}$  is the binary class label,  $A$  or  $B$ :

$$\{ ((6,7),A), ((4,8),B), ((6,5),B), ((7,9),A), ((2,4),A) \}$$

Classify a test example  $(x_1^{\text{Test}}, x_2^{\text{Test}})$  with attribute values  $(4,7)$  using a KNN classifier with  $K = 3$  and Manhattan distance defined by

$$\text{distance}(\mathbf{x}^{\text{Train}}, \mathbf{x}^{\text{Test}}) = \sum_{i=1}^2 |x_i^{\text{Train}} - x_i^{\text{Test}}|$$

where  $|\cdot|$  denote absolute value.

Complete the following table by filling in the computed Manhattan distance between each training data point and the test example. Determine the class label based on the results.

$x_1^{\text{Train}}$	$x_2^{\text{Train}}$	$c^{\text{Train}}$	Distance
6	7	A	
4	8	B	
6	5	B	
7	9	A	
2	4	A	

**Answer:**

$x_1^{\text{Train}}$	$x_2^{\text{Train}}$	$c^{\text{Train}}$	Distance
6	7	A	2
4	8	B	1
6	5	B	4
7	9	A	5
2	4	A	5

The predicted class label of the test example is B.

(b) [6 points] Judge whether each of the following student's claims is correct or not. Explain why.

- (i) [3 points] A student claims that the results of a general KNN classifier that uses Euclidean distance will change if we multiply all attribute values of each training and test data point by 0.5.

**Answer:**

The claim is false, because  $K$  nearest neighbors will remain unchanged after multiplying all attribute values of each training and test data points by 0.5.

- (ii) [3 points] Another student claims that the classification accuracy of the training set will always increase if the value of  $K$  used in KNN classifier is incrementally increased from 1 to  $N$ , where  $N$  is the total number of training examples.

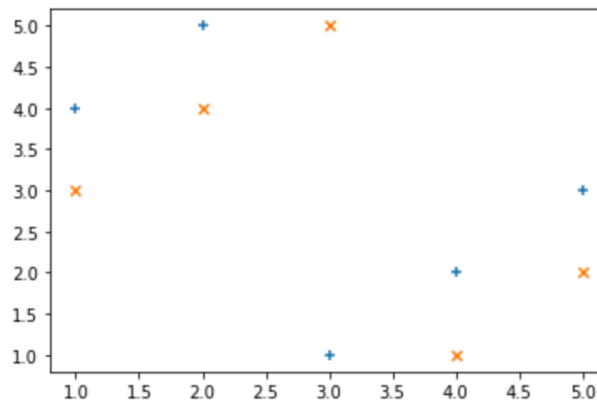
**Answer:**

The claim is false. A counterexample is as follows:

The training set accuracy when  $K = 1$  will be 100%.

As  $K$  approaches the total number of training examples more and more examples influence the class, and eventually the class will always be the majority class in the training set.

- (c) [5 points] Consider KNN using Euclidean distance on the following data set. Each point belongs to one of the two classes:  $+$  and  $x$ .



- (i) [2 points] Perform 10-fold cross validation on the given data set (i.e. the 10 data points as shown in the figure), what is the validation error when using 1-nearest neighbor?

**Answer:**

Every point is misclassified. So, the validation error is 10/10.

- (ii) [3 points] Which of the following values of K leads to the minimum 10-fold cross validation error: 3, 5 or 9? What is the error for that K? If there is a tie, please elaborate.

**Answer:**

All 3 values of K mis-classify all of the points and have the same classification errors, 10/10.

**Problem 5 [12 points] K-Means Clustering**

Given a 1-dimensional data set  $\{0, 3, 6, 9, 27, 30\}$ , use the K-means clustering algorithm and Euclidean distance to cluster the given points in the data set into 2 clusters. Assume  $c1 = 3$  and  $c2 = 4$  are chosen as the initial cluster centers.

- (a) [4.5 points] Perform one iteration of K-means clustering by finding the Euclidean distance between each data point in the data set and the centroids, and assign each data point to the closest centroid according to the distance found. Fill in the following table with your computed values. If your answer is not an integer, write your answer in decimal form, e.g. 1.234.

Data Point	Distance between the data point and $c1 = 3$	Distance between the data point and $c2 = 4$	Closest Centroid ( $c1$ or $c2$ ?)
0			
3			
6			
9			
27			
30			

**Answer:**

Data Point	Distance between the data point and $c1 = 3$	Distance between the data point and $c2 = 4$	Closest Centroid ( $c1$ or $c2$ ?)
0	3	4	$c1$
3	0	1	$c1$
6	3	2	$c2$
9	6	5	$c2$
27	24	23	$c2$
30	27	26	$c2$

- (b) [1.5 points] What are the values of  $c1$  and  $c2$  after one iteration of K-means? If your answer is not an integer, write your answer in decimal form, e.g. 1.234.

**Answer:**

$$c1 = \frac{0 + 3}{2} = 1.5$$

$$c2 = \frac{6 + 9 + 27 + 30}{4} = 18$$

- (c) [4.5 points] Perform the second iteration of K-means clustering by finding the Euclidean distance between each data point in the data set and the computed centroids in part (b), and assign each data point to the closest centroid according to the distance found. Fill in the following table with your computed values. If your answer is not an integer, write your answer in decimal form, e.g. 1.234.

Data Point	Distance between the data point and the c1 computed in part (b)	Distance between the data point and the c2 computed in part (b)	Closest Centroid (c1 or c2?)
0			
3			
6			
9			
27			
30			

**Answer:**

Data Point	Distance between the data point and the c1 computed in part (b)	Distance between the data point and the c2 computed in part (b)	Closest Centroid (c1 or c2?)
0	1.5	18	c1
3	1.5	15	c1
6	4.5	12	c1
9	7.5	9	c1
27	25.5	9	c2
30	28.5	12	c2

- (d) [1.5 points] What are the values of c1 and c2 after the second iteration of K-means? If your answer is not an integer, write your answer in decimal form, e.g. 1.234.

**Answer:**

$$c1 = \frac{0 + 3 + 6 + 9}{4} = 4.5$$

$$c2 = \frac{27 + 30}{4} = 28.5$$

**Problem 6 [20 points] Perceptron**

Given the following training dataset:

$x_1$	10	0	8	3	4	0.5	4	2
$x_2$	10	0	4	3	8	0.5	3	5
$T$	1	-1	1	-1	1	-1	1	1

- (a) [8 points] Show the action of the perceptron algorithm for the above sequence of data points by completing the following table. Assume  $\eta = 1$  and we start with the following initial weights and bias

$$w_1 = 1$$

$$w_2 = 1$$

$$\theta = 0$$

and use the following activation function.

$$f(z) = \begin{cases} 1 & z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

Updating rules:

$$\Delta w_i = \eta(T - O)x_i$$

$$\Delta \theta = \eta(T - O)$$

$$w_i = w_i + \Delta w_i$$

$$\theta = \theta + \Delta \theta$$

where  $i \in \{1, 2\}$ .

If your answer is not an integer, write your answer in decimal form, e.g. 1.234.

$x_1$	$x_2$	$T$	$O$	$\Delta w_1$	$w_1$	$\Delta w_2$	$w_2$	$\Delta \theta$	$\theta$
-	-	-	-	-	1	-	1	-	0

**Answer:**

$x_1$	$x_2$	$T$	$O$	$\Delta w_1$	$w_1$	$\Delta w_2$	$w_2$	$\Delta \theta$	$\theta$
-	-	-	-	-	1	-	1	-	0
10	10	1	1	0	1	0	1	0	0
0	0	-1	1	0	1	0	1	0	-2
8	4	1	1	0	1	0	1	0	-2
3	3	-1	1	-6	-5	-6	-5	-2	-4
4	8	1	-1	8	3	16	11	2	-2
0.5	0.5	-1	1	-1	2	-1	10	-2	-4
4	3	1	1	0	2	0	10	0	-4
2	5	1	1	0	2	0	10	0	-4

- (b) [2 points] According to the values in the table above, state whether the perceptron algorithm is converged in 1 epoch. If not, explain why.

**Answer:**

The algorithm is not converged in 1 epoch. Since there are changes of weights and biases.

- (c) [10 points] Write a Python program to verify your answers of Part (a). In your program, you need to define the following variables
- A 2D NumPy array,  $X$ , to store all the attribute values  $x_1, x_2$ , where the shape is (8,2)
  - A 1D NumPy array,  $T$ , to store the target values, where the shape is (8,)
  - A 1D NumPy array,  $W$ , to store the weights, where the shape is (2,)
  - A float bias value,  $b$ .

and perform the required computations. Also, print the following sequence of values (in exact order) for each iteration:

$x_1 < s > x_2 < s > T < s > O < s > \Delta w_1 < s > w_1 < s > \Delta w_2 < s > w_2 < s > \Delta \theta < s > \theta < end >$

where  $< s >$  refers to an empty space, and  $< end >$  refers to an end of line character.

The following shows a line of sample output.

0 0 0 0 0 0 0 0 0 0

Remark: You cannot use any other libraries other than NumPy in your program.

**Answer:**

```
import numpy as np

X = np.array([[10,10], [0,0], [8,4], [3,3], [4,8], [0.5,0.5], [4,3], [2,5]])
T = np.array([1,-1,1,-1,1,-1,1,1])
W = np.array([1,1], dtype=np.float32)
b = 0.0

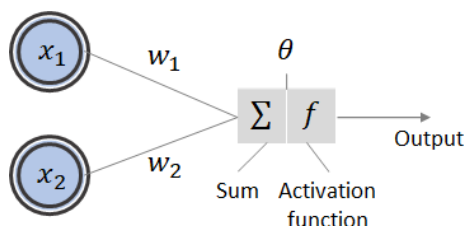
for i in range(X.shape[0]):
    y = X[i].dot(W) + b
    if y >= 0:
        O = 1
    else:
        O = -1
    DW = (T[i] - O) * X[i]
    W += DW
    Db = (T[i] - O)
    b += Db

print(X[i][0], X[i][1], T[i], O, DW[0], W[0], DW[1], W[1], Db, b)
```

## Problem 7 [12 points] Perceptron and Multilayer Perceptron

- (a) [4 points] Can we represent the given boolean function with a single neuron as shown below?

A	B	f(A,B)
0	0	0
0	1	0
1	0	1
1	1	0



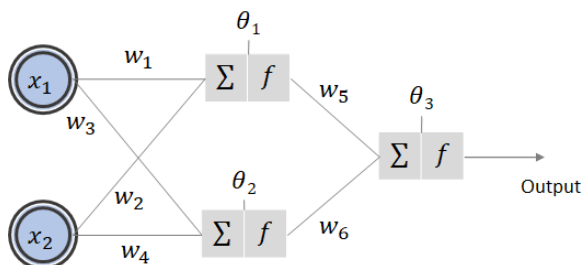
If yes, show the possible weights, bias and activation function that can be used to compute the output of all the data points correctly. If not, explain why not in 1-2 sentences.

**Answer:**

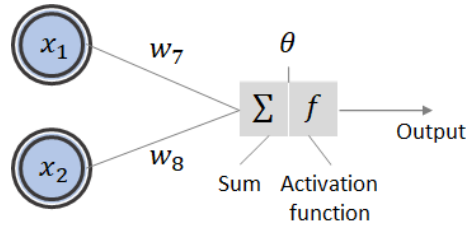
Yes, we can represent this function with a single neuron, since it is linearly separable. One set of possible weights and bias is:  $w_1 = 1, w_2 = -1, \theta = -0.5$ , and the activation function is

$$f(x) = \begin{cases} 1 & x > 0 \\ 0 & \text{otherwise} \end{cases}$$

- (b) [6 points] Suppose we have a neural network as shown below with  $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0$ , and linear activation function (i.e.  $f(x) = Cx$ , where  $C$  is a constant).



Can any function that is represented by the above network be represented by a single unit of artificial neural network in the following diagram? If so, detail the weights ( $w_7$  and  $w_8$ ), bias ( $\theta$ ), and the activation function  $f(x)$ . Otherwise, explain why not.



**Answer:**

Yes, the network can be represented by a single unit of artificial neural network by setting its weights,  $w_7 = w_1w_5 + w_3w_6$ ,  $w_8 = w_2w_5 + w_4w_6$ , and the activation function  $f(x) = C^2x$ .

- (c) [2 points] Given a multilayer perceptron with 3 layers (input layer, hidden layer and output layer). The number of units in each of these layers are 3, 4, 2. Assume each input or neuron is fully-connected. Calculate the number of trainable parameters of this network.

**Answer:**

$$3 * 4 + 4 + 4 * 2 + 2 = 26.$$

----- END OF PAPER -----