COMP 2211 Midterm Exam - Spring 2024 - HKUST

Date: March 16, 2024 (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 **28** pages (including this cover page, honor code, and 4 blank pages at the end).
3. Write your answers in the space provided in black/blue ink. *NO pencil please, otherwise you are not allowed to appeal for any grading disagreements.*
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	
Student ID	
Venue and Seat Number	

Problem	**Topic**	**Score**
1	True/False Questions	/ 5
2	Advanced Python for Artificial Intelligence	/ 19
3	Model Evaluation & Advanced Python Programming	/ 16
4	Naïve Bayes Classifier	/ 16
5	K-Nearest Neighbors	/ 18
6	Leader Clustering	/ 16
7	D-fold Cross Validation	/ 10
	Total	/ 100

The HKUST Academic Honor Code

Honesty and integrity are central to the academic work of HKUST. Students of the University must observe and uphold the highest standards of academic integrity and honesty in all the work they do throughout their program of study.

As members of the University community, students have the responsibility to help maintain the academic reputation of HKUST in its academic endeavors.

Sanctions will be imposed if students are found to have violated the regulations governing academic integrity and honesty.

Declaration of Academic Integrity

I confirm that I have answered the questions using only materials specifically approved for use in this examination, that all the answers are my own work, and that I have not received any assistance during the examination.

Signature: _____

Problem 1 [5 points] True/False Questions

Indicate whether the following statements are true or false by putting **T** or **F** in the given table. You get 0.5 point for each correct answer.

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

(a) In the following code

```
import numpy as np
a = np.array([0, 1, 1, 2, 5, 5, 3])
b = np.array([0, 1, 2, 3, 4, 5])
c = (b == a.reshape(7, 1))
```

The array `c` has the shape (7, 1).

(b) After executing the following block of code:

```
import numpy as np
a = np.array([[1, 2], [3, 4], [5, 6]])
b = np.array([[1, 2, 3], [0, 0, 0], [1, 0, 0]])
c = a.dot(b)
```

The array `c` is `array([[22, 28], [0, 0], [1, 2]])`.

(c) The Naïve Bayes Classifier operates under the assumption that the presence of a particular feature in a class is independent of the presence of any other feature.

(d) In Naïve Bayes, we assume that $P(B|e1, e2) = P(B|e1)P(B|e2)$ where B is our belief and $(e1, e2)$ are evidence.

(e) In Naïve Bayes, given $P(B = b)$, $P(e1|B = b)$, and $P(e2|B = b)$ for each possible belief b, we can compute $P(B = b'|e1, e2)$ for any b'.

(f) K-Nearest Neighbors Classifier **CANNOT** handle data with categorical features since it is difficult to find the distance between categorical features.

(g) In K-Nearest Neighbors for binary classification, odd values of k are usually preferred.

(h) A 6-fold cross validation for K-nearest neighbors algorithm means that for each value of K, we randomly select 1/6 of the training data as the validation set to evaluate the model which is trained by the remaining (5/6) of the training data.

(i) The result of the K-Means Clustering **DOES NOT** depend on the initial centroids.

(j) It is possible that after new cluster centroids are computed by the K-Means Clustering Algorithm, a cluster centroid may be associated with an empty cluster (i.e., with zero points in it).

Problem 2 [19 points] Advanced Python for Artificial Intelligence

(a) [13 points] Consider the following NumPy arrays:

```
import numpy as np
# np.arange(start, stop)
# - return an array of evenly spaced values within the half-open interval
#   [start,stop), the default step size is 1.
# np.ones(shape):
# - return a new array of given shape, filled with ones.
A = np.arange(5,15)
B = np.ones((3,2))
C = np.array([[[0,1,2,3],
               [4,5,6,7]],
              [[8,9,10,11],
               [12,13,14,15]],
              [[16,17,18,19],
               [20,21,22,23]]])
```

Suppose the following Python statements are **running consecutively**. Write the output for each of the following Python statements. If the output is an empty array, write "Empty Array". If an error occurs, write "Error".

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

(ii)
```
# np.ndarray.reshape(shape)
# - return an array containing the same data with a new shape.
print(B.reshape(3,-1,2))
```

(iii) Please write a line of Python code to get the same result in part (a)(ii) by using `np.expand_dims` on `B`.

```
# np.expand_dims(a, axis)
# - insert a new axis to a that will appear at the axis position in the
#   expanded array shape. Return an array that is the view of a with the
#   number of dimensions increased.
```

(iv) Please write a line of Python code to create the array `C` by using the functions `np.ndarray.reshape` and `np.arange()`.

(v)
```
# np.mean(a, axis)
# - return a new array containing the average values of a over the specified axis.
print(np.mean(C,axis = 2))
```

(vi)
```
# np.transpose(a, axes)
# - return an array with axes transposed in a.
print(np.transpose(C, (1, 0, 2)))
```

(vii)
```
# a@b
# - return the matrix multiplication of the two arrays a and b.
D = A.reshape(2, 5)
print(B@D)
```

(viii)
```
# np.dot(a,b)
# - return the dot product of the two arrays a and b. If both a and b are
#   1-D arrays, it is the inner product of vectors and returns a scalar.
#   If both a and b are bool arrays, the output is in bool datatype.
# np.ndarray.astype(dtype)
# - return the copy of the array with a specified dtype.
E = A < 12
F = A % 5 == 0
print(np.dot(E, F).astype(int))
```

(ix)
```
print(np.dot(E, F.astype(int)))
```

(x)
```
print( C / B )
```

(xi)
```
# np.newaxis
# - increase the dimension of an array by adding new axis.
print(C / B[..., np.newaxis])
```

(xii)
```
G = C[0, :, 2:]
print(G)
```

(xiii)
```
C[0, 1, 2] = 100
print(G)
```

Scheme:

- 1 point for giving the correct answer for each part. 13 points in total.

(b) [6 points] In recommendation systems, we often recommend items similar to what the user likes. This is known as content-based filtering. In content-based filtering, an item is represented as a feature vector (or a 1D array). Given the following code which computes the cosine similarity between feature vectors of items in explicit loops:

```
def compute_cosine_similarity_loops(X):
    num_items, num_features = X.shape
    # --- BLOCK TO REWRITE ---
    X_normalized = np.zeros((num_items, num_features))
    for i in range(num_items):
        feature_norm = np.sqrt(np.sum(X[i] ** 2))
        X_normalized[i] = X[i] / feature_norm
    similarities = np.zeros((num_items, num_items))
    for i in range(num_items):
        for j in range(num_items):
            similarities[i, j] = np.sum(X_normalized[i] * X_normalized[j])
    # --- BLOCK TO REWRITE ---
    return similarities


X = np.array([[0, 2], [1, -1], [1, 1]])
print(compute_cosine_similarity_loops(X))
# Output:
# [[ 1.         -0.70710678  0.70710678]
#  [-0.70710678  1.          0.        ]
#  [ 0.70710678  0.          1.        ]]
```

Rewrite the block of code between the comment lines "`# --- BLOCK TO REWRITE ---`" using **no explicit loops** in the space provided. You may find the following functions useful for this question.

- Element-wise square of an array:
  `np.square(A)`
  - `A` is the input array

  This is equivalent to `A ** 2`.
- Element-wise square root of an array:
  `np.sqrt(A)`
  - `A` is the input array
- Sum of array elements over a given axis:
  `np.sum(A, axis)`
  - `A` is the input array
  - `axis` is the axis across which the array is summed

- Insert a new axis of size 1 to an array:
  `np.expand_dims(A, axis)`
  - `A` is the input array
  - `axis` is the position where the axis is to be inserted

  If `axis` is 0, this is equivalent to `A[np.newaxis]` and `A[None]`. If `axis` is 1, this is equivalent to `A[:, np.newaxis]` and `A[:, None]`.
- Transpose of an array:
  `np.transpose(A)`
  - `A` is the input array

  This is equivalent to `A.T`.
- Matrix multiplication:
  `np.matmul(A, B)`
  - `A` is the left array for matrix multiplication
  - `B` is the right array for matrix multiplication

  This is equivalent to `A @ B`.
  More information on matrix multiplication: suppose `A.shape[1] == B.shape[0]`, then
  `C = np.matmul(A, B)`
  means that for each `i` and `j`,
  `C[i, j] == np.sum(A[i] * B[:, j])`

Write your code in the space below.

Problem 3 [16 points] Model Evaluation and Advanced Python Programming

In this problem, you need to implement the evaluation metrics for multi-class classifiers. Specifically, you need to implement the confusion matrix, accuracy, precision, recall, and macro F1 score. We provide the related definitions and formulas as follows.

(a) [5.5 points] Suppose there is a test dataset consisting of 10 data points, their actual classes are [2, 1, 1, 2, 0, 1, 0, 0, 1, 1], and their predicted classes by a classifier model are [2, 1, 2, 1, 0, 2, 1, 0, 0, 0]. What is the confusion matrix for the prediction results? What are TP, TN, FP, FN for each class?

Confusion matrix: a table that summarized actual labels and the predictions of classification. For multi-class classification, the confusion matrix has the shape (num classes, num classes) that records the number of occurrences between actual labels and the predictions. The classes are listed in the same order in the rows as in the columns, therefore the correctly classified elements are located on the main diagonal.

For each class i, TP_i, TN_i, FP_i, FN_i represent the instance numbers of true positive, true negative, false positive, and false negative, respectively.

- True positive: A test result where the classifier correctly predicts the positive class as positive.
- True negative: A test result where the classifier correctly predicts the negative class as negative.
- False positive: A test result where the classifier incorrectly predicts the negative class as positive.
- False negative: A test result where the classifier incorrectly predicts the positive class as negative.

Please fill in the confusion matrix (the rows represent actual class and the columns represent predicted class) and the TP, TN, FP, FN table.

	Predicted			
Actual	Class	0	1	2
	0			
	1			
	2			

Class	TP	TN	FP	FN
0				
1				
2				

(b) [1 point] What is the accuracy score of the classifier model on the above test data? Accuracy = $\frac{\sum_{i=1}^{N} TP_i}{num\_testdata}$, where N is the number of classes.

(c) [2.5 points] What are the precisions, recalls, and F1 scores for each class of the classifier model on the above test data? Please fill in the table using fractions or keep 3 decimals.

For each class i, $Precision_i = \frac{TP_i}{TP_i + FP_i}$
For each class i, $Recall_i = \frac{TP_i}{TP_i + FN_i}$
For each class i, $F1_i = \frac{2 \cdot Precision_i \cdot Recall_i}{Precision_i + Recall_i}$

Class	Precision	Recall	F1 score
0			
1			
2			

(d) [1 point] What is the macro F1 score of the classifier model on the above test data? Please keep 3 decimals in your answer.
The macro F1 score is the unweighted mean of the F1 scores of all classes:
$Macro\text{-}F1 = \frac{\sum_{i=1}^{N} F1_i}{N}$, where N is the number of classes.

(e) [6 points] Given two NumPy 1D arrays with the same shape `(num_testdata,)`: `test_actual` and `test_predict`, representing the actual class labels and predicted class labels for the test data, please implement the following functions by filling in the blanks. For each TODO, please use a **one-line Python expression**.

```
import numpy as np

def generate_confusion_matrix(test_actual, test_predict):
    # TODO 1: Get num_classes, the number of classes in the test data.
    # Note that the classes in the test_actual and test_predict are represented
    # in integer indices from [0, 1, ..., num_classes - 1].
    num_classes = ________________________________________
    confusion_matrix = np.zeros((num_classes, num_classes))

    # TODO 2: Get the values of confusion_matrix, where the rows represent
    # actual class and the columns represent predicted class.
    for i in range(0, num_classes):
        for j in range(0, num_classes):
            confusion_matrix[i, j] = ____________________________________
    return confusion_matrix

def calculate_evaluation_metrics(test_actual, test_predict):
    confusion_matrix = generate_confusion_matrix(test_actual, test_predict)
    # TODO 3: Get the accuracy score, which is a scalar value.
    accuracy = ___________________________________________________
    # TODO 4: Get the precisions for all classes, which is a 1D array
    # with shape (num_classes, ).
    precision = __________________________________________________
    # TODO 5: Get the recalls for all classes, which is a 1D array
    # with shape (num_classes, ).
    recall = _____________________________________________________
    # TODO 6: Get the macro F1 score, which is a scalar value.
    macro_f1 = ___________________________________________________
    return accuracy, precision, recall, macro_f1
```

Note:

- An expression is a combination of values, variables, operators, and calls to functions.
- There must be no explicit loops in your expression.
- Your implemented functions should work with any number of test data points and any number of classes.
- You cannot use any variable that is not defined inside the function or any global variable.

You may find the following attribute or functions useful for this problem.

- `np.max(a, axis = None)`
  - return the maximum of the array a along the given axis. If axis is None, the result is a scalar value.
- `np.ndarray.sum(axis = None)`
  - return the sum of the array over the given axis. If axis is None, the result is a scalar value.
- `np.ndarray.diagonal()`
  - if the array is 2D, then a 1D array containing the diagonal elements is returned.
- `np.mean(a, axis)`
  - return a new array containing the average values of a over the specified axis. If axis is None, the result is a scalar value.

Write your code in the space below.

Problem 4 [16 points] Naïve Bayes Classifier

Based on the given training data in the table below, which includes both numerical and categorical attributes, make a prediction about the degree classification (DC) (i.e., First-Class Honors or Second-Class Honors D1 or Second-Class Honors D2, or Third-Class Honors) of the student with the following attribute values using Naïve Bayes classifier.

- Study Attitude (SA): Serious
- Part-time Job (PTJ): No
- Average Energy Level (AEL): 7.2
- Courses Taught by Desmond and Pearl (CDP): Yes
- Number of Friends in the Study Group (NFSG): 4

Student	Study Attitude (SA) **Categorical**	Part-time Job (PTJ) **Categorical**	Average Energy Level (AEL) **Numerical**	Courses Taught by Desmond and Pearl (CDP) **Categorical**	Number of Friends in the Study Group (NFSG) **Categorical**	Degree Classification (DC)
1	Serious	No	8.5	Yes	5	First
2	Moderate	Yes	6.2	No	3	Second D1
3	Serious	No	7.8	Yes	4	Second D1
4	Moderate	Yes	5.9	No	2	Third
5	Serious	No	8.1	Yes	6	First
6	Casual	Yes	6.5	Yes	1	Second D2
7	Serious	No	7.3	No	4	Second D1
8	Serious	No	7.9	Yes	5	First
9	Moderate	Yes	5.7	No	3	Third
10	Serious	No	8.2	Yes	6	First
11	Casual	Yes	6.1	Yes	2	Second D2
12	Serious	No	7.6	No	4	Second D2
13	Moderate	Yes	6.4	Yes	3	Third
14	Serious	No	8.0	Yes	5	First
15	Casual	Yes	5.8	No	1	Second D2
16	Serious	No	7.5	Yes	4	Second D1
17	Serious	No	7.7	Yes	6	First
18	Moderate	Yes	6.0	No	3	Third
19	Serious	No	7.8	Yes	5	First
20	Casual	Yes	6.3	Yes	2	Second D2

Assume each categorical attribute has the following possible values:

- Study Attitude: Serious, Moderate, Casual
- Part-time Job: Yes, No
- Courses Taught by Desmond and Pearl: Yes, No
- Number of Friends in the Study Group: 1, 2, 3, 4, 5, 6

Assume that the numerical data follow a Gaussian distribution:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

where

$$\texttt{numerical training data} = (x_1, x_2, \ldots, x_n)$$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2}$$

If needed, apply 1-Laplace Smoothing to the likelihood probabilities of the **affected feature only**. The affected feature means that the categorical feature has a category given some belief in the test dataset, which was **NOT** observed in the training dataset. Please provide all the steps.

You may find the following equation useful for this question:

$$B_{NB} = argmax_{B_i} P(B_i)(P(e_1|B_i)P(e_2|B_i)P(e_3|B_i)\cdots P(e_d|B_i))$$

(a) [4 points] Calculate the mean (μ) and standard deviation (σ) of the Average Energy Level (AEL) of Degree Classification: First-Class Honors and Second-Class Honors D1.

(b) [3 points] Calculate the test data sample's likelihoods of Average Energy Level (AEL) of First-Class Honors and Second-Class Honors D1.

(c) [4 points] Calculate the test data sample's likelihoods of Study Attitude (SA), Part-time Job (PTJ), Courses Taught by Desmond and Pearl (CDP), and Number of Friends in the Study Group (NFSG) of all Degree Classification(s) (DC).

(d) [2 points] Calculate the prior probabilities.

(e) [3 points] Finally, calculate the posterior probabilities and make the prediction.

Assume that the likelihood of Average Energy Level (AEL) of Second-Class Honors D2 and Third-Class Honors are:

- P(AEL=7.2|Second-Class Honors D2) = 0.32515
- P(AEL=7.2|Third-Class Honors) = 0.000334158

Problem 5 [18 points] K-Nearest Neighbors

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

{ ((4,7), A), ((3,4), B), ((5,7), B), ((5,10), A), ((8,6), A) }

We use a weighted K-Nearest Neighbors Classifier with K = 3, the distance between two data samples is defined as Manhattan distance (sum of absolute difference on each attribute), and the weight is defined as the inverse of the distance, that is $\frac{1}{distance}$.

Given a test sample with attribute values (4,6), fill in the table below and determine the class label:

x_1	x_2	y	Distance	Weight
4	7	A		
3	4	B		
5	7	B		
5	10	A		
8	6	A		

The class label: _____

(b) [13 points] Now, given a new training dataset with 6 samples with a K-Nearest Neighbors Classifier where Manhattan distance is employed, calculate the 6-cross-validation error for K = 1, 3 respectively. What K should we use based on the cross-validation results? Note: When selecting a neighbor, to resolve ties, choose the neighbor with the lowest index. When evaluating the error, use

- Number of wrong predictions / Number of test data points

during the cross-validation.
Please provide all the steps to choose K.

index	Attribute 1	Attribute 2	Class
1	2.6	2	A
2	2.5	3.7	A
3	3.4	3.8	B
4	2.7	3	A
5	3.2	2.9	B
6	3.5	4	B

(c) [2 points] Based on the chosen K and those 6 training samples in part (b), calculate the test error for the test dataset below.

Note:

- When selecting a neighbor, to resolve ties, choose the neighbor with the lowest index.
- When evaluating the error, use
  - Number of wrong predictions / Number of test data points

.

Attribute 1	Attribute 2	Class
3.4	3	B
3	2.8	A
2	3.5	A
2.5	7	B

Problem 6 [16 points] Leader Clustering

Consider the following cluster method called **Leader Clustering**. It receives two parameters: an integer K and a real-value threshold T. Similar to K-means clustering, it starts by selecting K instance (which will be called leaders) and assigns each training instance to the cluster of the closest leader. During this assignment step, however, if the distance of a training instance to its **closest** leader is greater than the input threshold T, then this training instance forms a new cluster and becomes the initial leader of this new cluster. After all the training instances have been assigned to a cluster, the leader of each cluster is updated as the mean of the cluster. The process is then repeated until the cluster assignments do not change.

(a) [6 points] Given a 1-dimensional data set { 1, 3, 5, 9, 11, 13, 15 }, use the Leader Clustering algorithm and Euclidean distance to cluster the given points in the data set into 2 clusters. Assume c1 = 3 and c2 = 11 are chosen as the initial K = 2 leaders and the threshold for forming new clusters T = 5. Fill in the following table of the first assignment iteration with your completed values and what are the leaders after the first assignment iteration? (If new clusters are formed in the process, named their leaders as c3, c4, ... based on the order. Leave the distance c3/c4/... blank if the new cluster(s) are not formed yet.)

Data point	Distance between the data point and c1	Distance between the data point and c2	Distance between the data point and c3 (if needed)	Distance between the data point and c4 (if needed)	Closest Centroid
1					
3					
5					
9					
11					
13					
15					

The leaders after the first assignment iteration: _____

(b) [6 points] Given a 1-dimensional data set {5, 9, 11, 13, 17, 19}, use the Leader Clustering algorithm and Euclidean distance to cluster the given points in the data set into 2 clusters. Assume c1 = 5 and c2 = 11 are chosen as the initial K = 2 leaders and the threshold for forming new clusters T = 5. Fill in the following table of the first assignment iteration with your computed values and what are the leaders after the first assignment iteration? (If new clusters are formed in the process, named their leaders as c3, c4, ... based on the order. Leave the distance c3/c4/... blank if the new cluster(s) are not formed yet.)

Data point	Distance between the data point and c1	Distance between the data point and c2	Distance between the data point and c3 (if needed)	Distance between the data point and c4 (if needed)	Closest Centroid
5					
9					
11					
13					
17					
19					

The leaders after the first assignment iteration: _____

(c) [2 points] Which of the two methods, K-Means Clustering or Leader Clustering, will be better at dealing with outliers? Please briefly explain.

(d) [2 points] During lectures, we have learned that one drawback of the K-Means Clustering algorithm is that we need to specify K for the algorithm, but usually, we don't know how many clusters there should be for an unlabeled dataset. Will the Leader Clustering mitigate this drawback? Will there be any related limitations of the Leader Clustering algorithm? Please briefly explain.

Problem 7 [10 points] D-fold Cross Validation

Suppose you have a dataset of 2-dimensional points consisting of 4 classes with 20 points each, arranged as follows:

- $\mathbf{x_0}$ to $\mathbf{x_{19}}$, where all of them with class label 0
- $\mathbf{x_{20}}$ to $\mathbf{x_{39}}$, where all of them with class label 1
- $\mathbf{x_{40}}$ to $\mathbf{x_{59}}$, where all of them with class label 2
- $\mathbf{x_{60}}$ to $\mathbf{x_{79}}$, where all of them with class label 3

Someone has performed D-fold cross-validation with a multi-class classifier and obtained the following results:

- D = 2, average accuracy = 0%
- D = 3, average accuracy = 50%
- D = 5, average accuracy = 100%

Describe the issue with this implementation of D-fold cross-validation. Provide a concise explanation of the result obtained for each value of D. Also, suggest improvements to the implementation.

Assumption: If there are samples from a specific class, labeled as i, in the training data, then any samples in the test set that also belong to class i will be correctly classified. On the other hand, if a class is NOT present in the training data, then any test sample of this class will be misclassified.

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

/* Rough work */

/* Rough work */

/* Rough work */

/* Rough work */