ICE - 4701 Advanced Machine Learning

Assignment 2

Due date: 5 pm, Friday, 5th May 2023

Instructions.

- Solve all problems.
- Possible marks: 100.
- Submit ONE file with your Python code. The code should produce the desired output. If you need to include a comment, place it as a comment in the Python file. You can submit a Jupyter Notebooks or a plain py file. Submit also all images that your code needs in order to run. You don't have to submit the csv files with the data.
- Note: Points will be taken off for code which is long, unreadable, or inefficient.

Problem 1. Multi-label classification

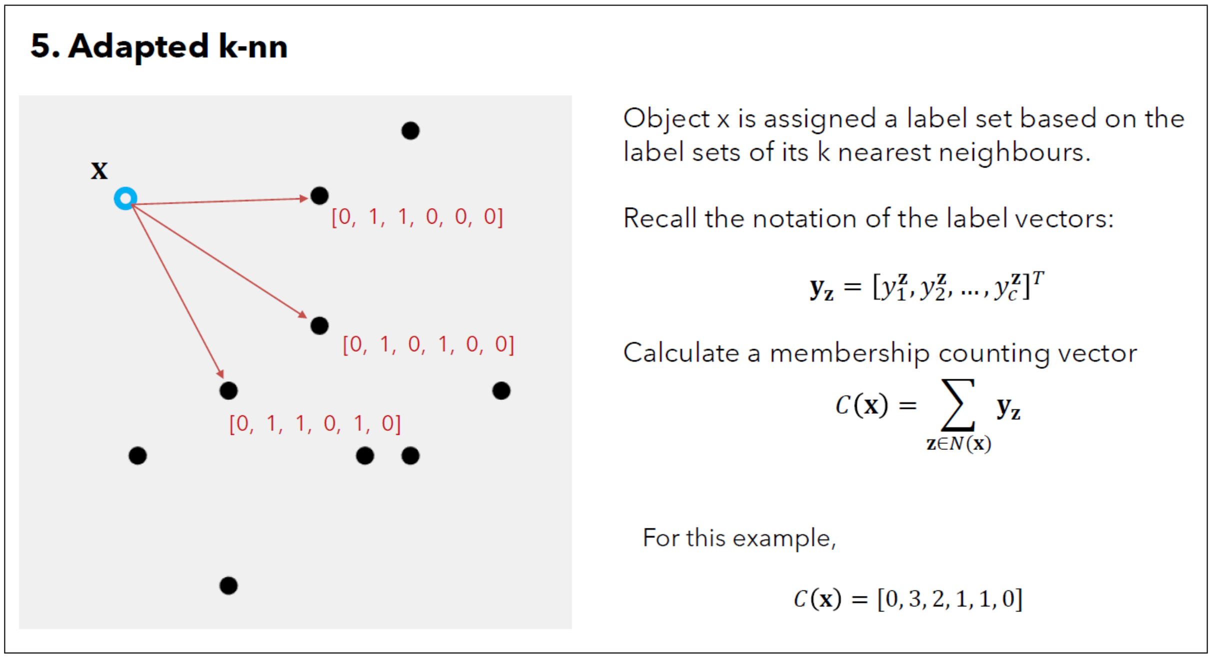
This time, to avoid confusion, you are only allowed to import the following Python libraries:

- numpy
- matplotlib.pyplot
- pandas
- LinearDiscriminantAnalysis from `sklearn.discriminant_analysis`

(a) Prepare a function named `binary_relevance_ldc` which will implement the Binary Relevance Method for multilabel classification. The classifier model for the binary classification should be the Linear Discriminant Classifier (LDC). The input should be: training data ($N \times n$), training labels ($N \times c$, containing integers 1, 2, 3, ...), and testing data ($N_{test} \times n$). The function should return a binary matrix of assigned labels of size $N_{test} \times c$. [5]

(b) Subsequently, write a function named `adaptive_knn` which will implement the Adaptive knn methods for multilabel classification as explained in the lectures (The slide is reprodiced in Figure 1).

Assume that we apply the following rule: after calculating the membership counting vector $C(\mathbf{x})$, our classifier returns only the casses which score value larger than 3 in the counting vector. Use $k = 10$ neighbours. The input and the output should be the same as for the binary relevance function in part (a) of this assignment. [20]



5. Adapted k-nn

x
[0, 1, 1, 0, 0, 0]
[0, 1, 0, 1, 0, 0]
[0, 1, 1, 0, 1, 0]

Object x is assigned a label set based on the label sets of its k nearest neighbours.

Recall the notation of the label vectors:

$$\mathbf{y_z} = [y_1^{\mathbf{z}}, y_2^{\mathbf{z}}, \dots, y_c^{\mathbf{z}}]^T$$

Calculate a membership counting vector

$$C(\mathbf{x}) = \sum_{\mathbf{z} \in N(\mathbf{x})} \mathbf{y_z}$$

For this example,

$$C(\mathbf{x}) = [0, 3, 2, 1, 1, 0]$$

Figure 1: Lecture slide with the explanation of the Addapted-nn method for multi-label classification

(c) Use the Bird data set from Lab 6. (The csv files are also provided with this script for convenience.) Apply the two functions you programmed in (a) and (b) to compare the two classifiers for multilabel data on this dataset. Program and use the hamming loss for your comparison. [15]

(d) Find out the bird that is most well recognised and the one that is least well recognised. Find images of these birds and show them in a figure. You may use plt.imread() and plt.imshow(). An example of the expected output is shown in Figure 2 [20]

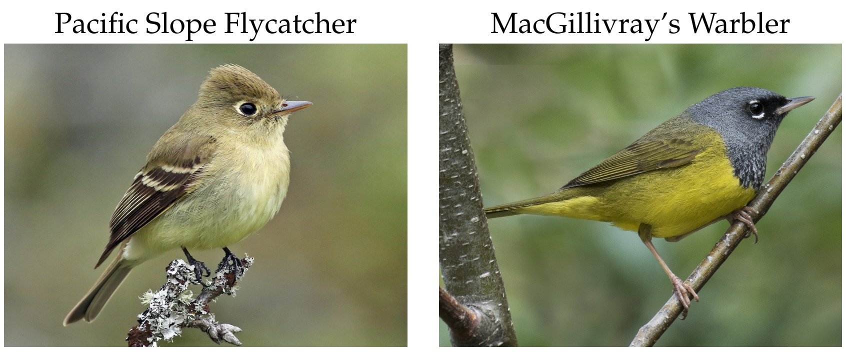


Figure 2: Expected output for the best and worst recognised birds. The two chosen birds are not the correct answer; they are used to show the format of the output.

Problem 2. Streaming data

Consider the following experiment. A data stream comes from two classes with normal distributions. The distributions are static; their parameters don't change with time. The parameters are:

$$\mathbf{m}_1 = [0,1]^T, \qquad \Sigma_1 = \begin{bmatrix} 7 & -1 \\ -1 & 3 \end{bmatrix}, \qquad \mathbf{m}_2 = [4,2]^T, \qquad \Sigma_2 = \begin{bmatrix} 2 & 1 \\ 1 & 6 \end{bmatrix}$$

(a) Generate a large data set from the two-class distribution to serve as the testing data. Use prior probabilities 0.5 and 0.5. Plot the data. [5]

(b) Prepare and run a data stream of 300 data points coming from the distribution of the problem. Sample each point by first choosing which class label the point should come from. To do this, compare a random number generated from a uniform distribution within [0,1] with threshold 0.5. If larger, sample a point from Class 1, otherwise, sample a point from Class 2. Apply the adaptive nearest mean classifier from Lab 7 (supervised version) to the streaming data. Start with means $\mathbf{m}_1 = [1,2]^T$ and $\mathbf{m}_2 = [-1,-1]^T$. Store the accuracy of the classifier along the data stream as well as the class means. Prepare two plots: (1) The accuracy of the classifier along the time steps, (2) The migration of the two means. An example of a figure of migration of two means for streaming data classification was shown in Lecture 11, slide 27. You don't need to plot the streaming points, just the mean trajectory. [20]

(c) Include, in the streaming data experiment in part (b) of this question, a calculation of the *running* error rate. This error rate is obtained ONLY using the streaming data. For example, suppose that the first object is classified incorrectly. Then your running error at time 1 is 1 (100%). Suppose that the next object is classified correctly by the updated NMC. The running error at time 2 becomes $\frac{1}{2} = 0.5$. Now, let the third object be labelled correctly. The running error at time 3 is $\frac{1}{3} = 0.33$, and so on. Plot the running error rate together with the "true" error rate which you plotted in part (b). Give a comment on the two curves you plotted. [15]