Faculty of Computers and AI
Cairo University

ML: Assignment 2

Decision Trees and KNN

[6 marks]

Dataset Description:

Whenever you go to the bank to deposit some cash money, the cashier places banknotes in a machine that tells whether a banknote is real or not. In the "BankNote_Authentication.csv" you have four features: variance, skew, curtosis and entropy and the class attribute refers to whether or not the banknote is real or forged.

Problem 1 [Decision Trees using Scikit-learn]:

Use the Banknote Authentication data attached with the assignment to implement the following requirements:

1. Experiment with a fixed train_test split ratio: Use 25% of the samples for training and the rest for testing.
    a. Run this experiment five times and notice the impact of different random splits of the data into training and test sets.
    b. Print the sizes and accuracies of these trees in each experiment.
2. Experiment with different range of train_test split ratio: Try (30%-70%), (40%-60%), (50%-50%), (60%-40%) and (70%-30%):
    a. Run the experiment with five different random seeds for each of split ratio.
    b. Calculate mean, maximum and minimum accuracy for each split ratio and print them.
    c. Print the mean, max and min tree size for each split ratio.
    d. Draw two plots: 1) shows mean accuracy against training set size and 2) the mean number of nodes in the final tree against training set size.

Note: the size of the tree is number of its nodes.

Problem 2 [KNN]:

Use the Banknote Authentication data to implement your own simple KNN classifier using python, (Don’t use any built-in functions):

1. Divide your data into 70% for training and 30% for testing.
2. Each feature column should be normalized separately from all other features. Specifically, for both training and test objects, each feature should be transformed using the function: *f(v) = (v - mean) / std*, using the mean and standard deviation of the values of that feature column on the training data.
3. If there is a tie in the class predicted by the k-nn, then among the classes that have the same number of votes, you should pick the one that comes first in the Train file.
4. Use Euclidean distance to compute distances between instances.
5. Experiment with different values of k=1,2,3, …., 9 and print the following:
    - The value of k used for the test set.
    - The number of correctly classified test instances.
    - The total number of instances in the test set. • The accuracy.

SUBMISSION RULES:

1. Deadline: 07 May 2023 @11:59 PM
2. No late submission is allowed.
3. Deliver your code as “.py” files not notebook extensions.
4. Name your zipped folder that contains [problem1.py and problem2.py] as follows: ID1_ID2_ID3_ID4_ID5. Zip

   You will lose 0.5 mark from the assignment grade if you did not write the correct naming convention.
5. Team members are Max: 5
6. Team members can be from different labs (but they must all attend the discussion together). If anyone did not attend the discussion will take zero in the assignment. You should understand every point in your code.
7. ONLY one member should upload the zipped folder to avoid multiple submissions for the same team.

8. We will run a plagiarism tool to check any cheating. Cheaters will take ZERO in the assignment and no excuses will be accepted.

GRADING CRITERIA:

Problem 1 [DT]	[8]
Experiment with Fixed Training Set size	2
Experiment with a range of train_test split ratio	4
Plotting	2
Problem 2 [KNN]	[10]
KNN algorithm (+ handling tie case)	4
Normalization + Euclidean distance Calc.	4
Experiment with different k values and print summary info as required	2
Total [18/3]	6