CSCI 2410 Introduction to Data Analytics Using Python
Homework Assignment #5

HW Programming #5: Data Analytics with KNN and Perceptron Techniques

Tasks: Experiment with the **KNN** and **Perceptron** classification techniques on 'iris' dataset loaded from sklearn datasets.

Assignment Instructions:

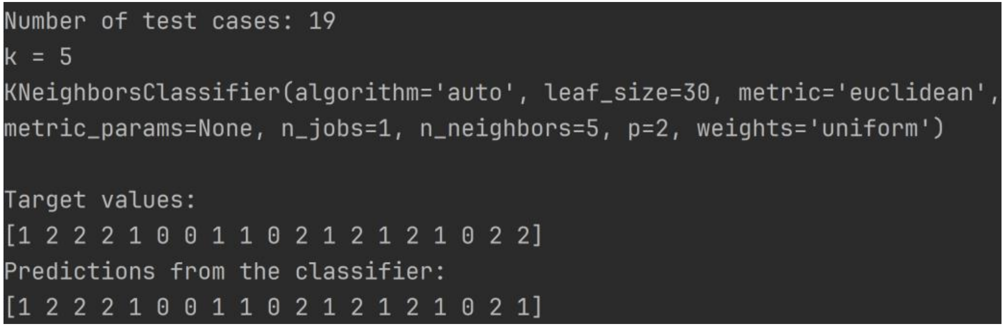
1. **[50%] KNN** classification on Iris dataset **load from sklearn datasets**

Note: the class labels for this data set are integers in [0, 1, 2]

Run with

(1) **[20%]** different **k** value,

(2) **[10%]** different number of test samples, and

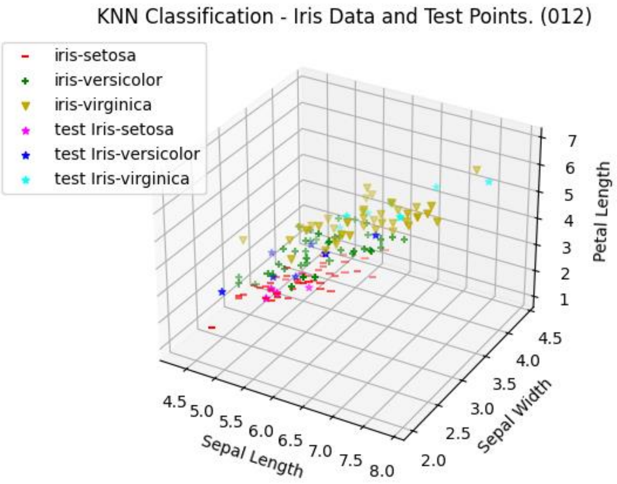


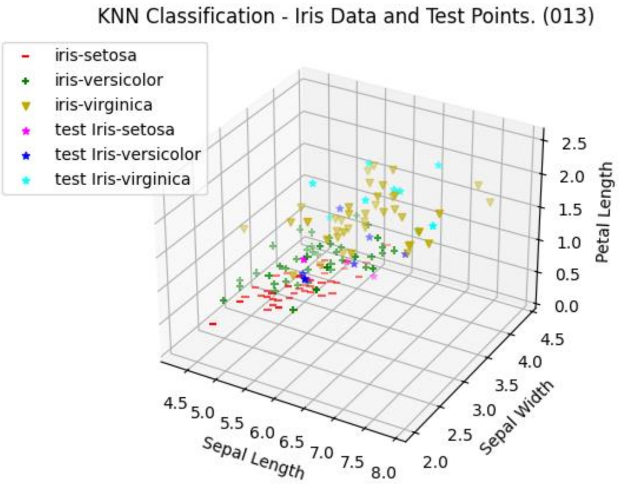
```
Number of test cases: 19
k = 5
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
metric_params=None, n_jobs=1, n_neighbors=5, p=2, weights='uniform')

Target values:
[1 2 2 2 1 0 0 1 1 0 2 1 2 1 2 1 0 2 2]
Predictions from the classifier:
[1 2 2 2 1 0 0 1 1 0 2 1 2 1 2 1 0 2 1]
```

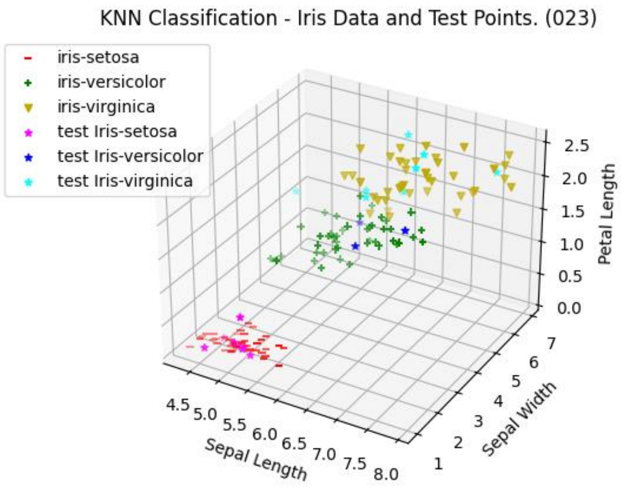
(3) **[20%]** 3D plots (Total 4 plots: (012, 013, 023, 123); where 012 means a 3D plot with the attributes of the 1st, 2nd, and 3rd columns of the data. Note: 123 means a 3D plot with the attributes of the 2nd, 3rd, and 4th columns of the data.

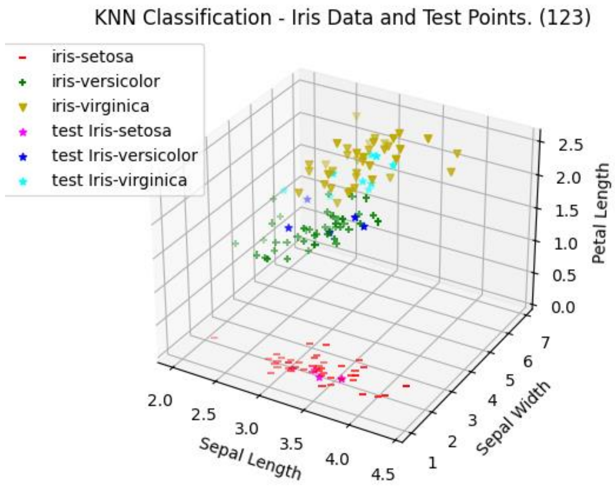
--- try to use different symbols/marks to distinguish the training data and test data points.

Python libraries needed: numpy, sklearn-datasets, sklearn.neighbors-KNeighborsClassifier, matplotlib.pyplot





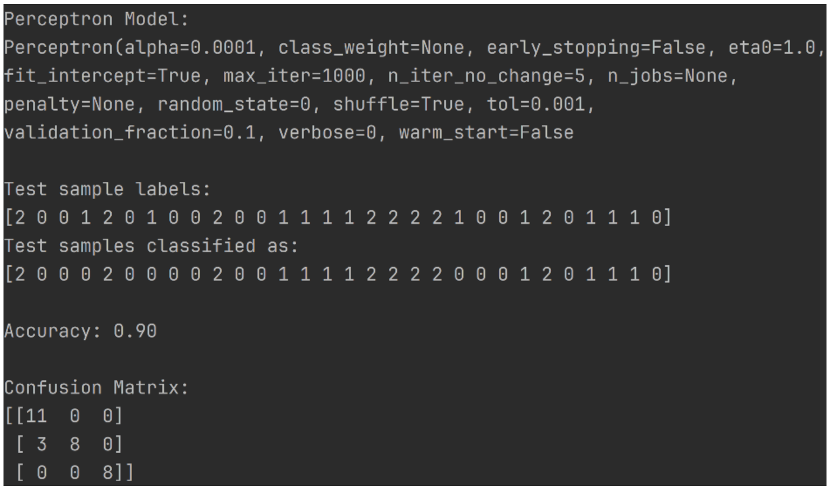




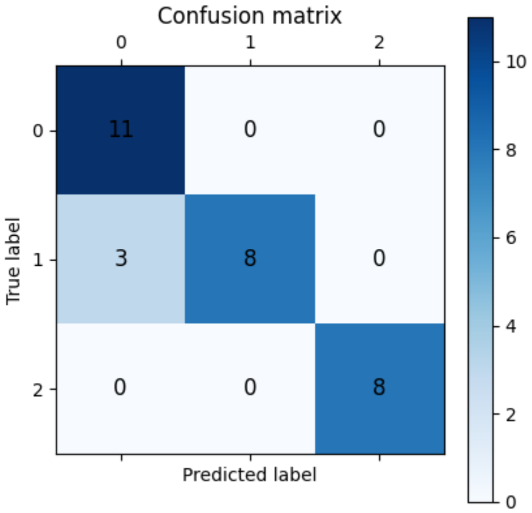
2. **[50%] Perceptron** on Iris dataset **load from sklearn datasets**

   Run with

   (1) **[20%]** all three classes, show and print the **confusion matrix** and accuracy score



```
Perceptron Model:
Perceptron(alpha=0.0001, class_weight=None, early_stopping=False, eta0=1.0,
fit_intercept=True, max_iter=1000, n_iter_no_change=5, n_jobs=None,
penalty=None, random_state=0, shuffle=True, tol=0.001,
validation_fraction=0.1, verbose=0, warm_start=False

Test sample labels:
[2 0 0 1 2 0 1 0 0 2 0 0 1 1 1 1 2 2 2 2 1 0 0 1 2 0 1 1 1 0]
Test samples classified as:
[2 0 0 0 2 0 0 0 0 2 0 0 1 1 1 1 2 2 2 2 0 0 0 1 2 0 1 1 1 0]

Accuracy: 0.90

Confusion Matrix:
[[11  0  0]
 [ 3  8  0]
 [ 0  0  8]]
```



   (2) **[30%]** 2-class classification on classes 1-2 (row 1-100 of the dataset) and classes 2-3(row 51-150 of the dataset).

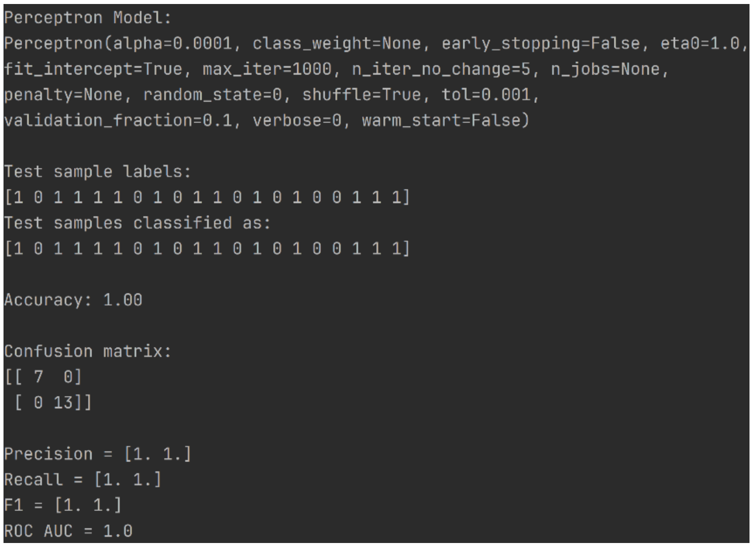
   Show and print the **confusion matrix**, and the scores of

   - accuracy,
   - precision,
   - recall,
   - F1, and
   - ROC_AUC scores

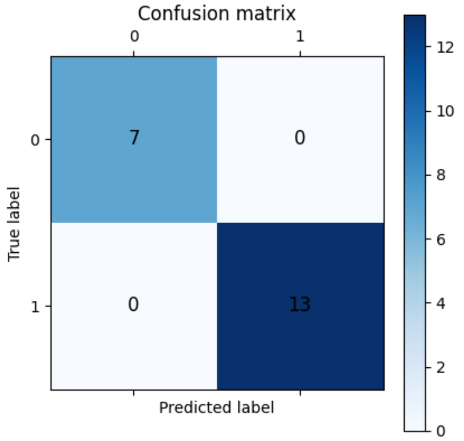
   for the two-class cases.

   Python libraries needed: sklearn-datasets, sklearn.preprocessing-StandardScaler, sklearn.linear_model-Perceptron, sklearn.model_selection-train_test_split, cross_val_score, sklearn.metrics-accuracy_score, sklearn.metrics-confusion_matrix, matplotlib.pyplot

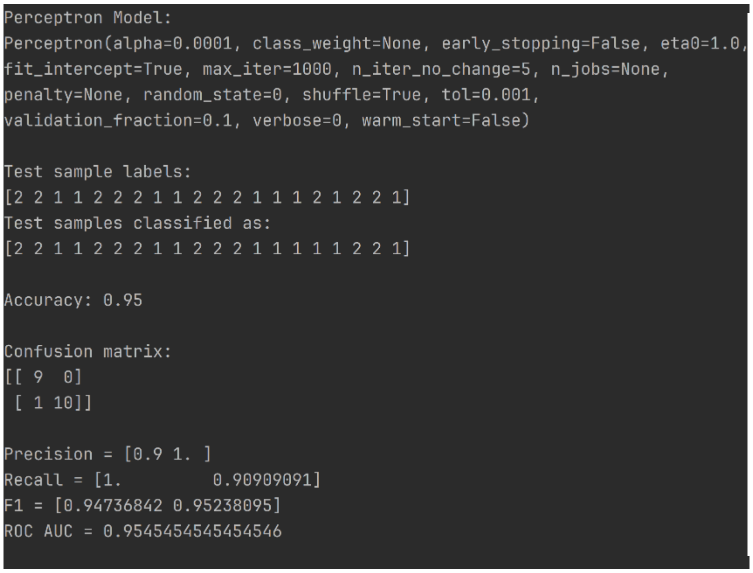
Classes 1-2 (row 1-100 of the dataset)



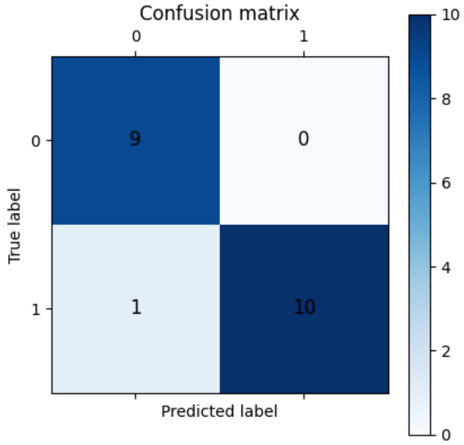
```
Perceptron Model:
Perceptron(alpha=0.0001, class_weight=None, early_stopping=False, eta0=1.0,
fit_intercept=True, max_iter=1000, n_iter_no_change=5, n_jobs=None,
penalty=None, random_state=0, shuffle=True, tol=0.001,
validation_fraction=0.1, verbose=0, warm_start=False)

Test sample labels:
[1 0 1 1 1 1 0 1 0 1 1 0 1 0 1 0 0 1 1 1]
Test samples classified as:
[1 0 1 1 1 1 0 1 0 1 1 0 1 0 1 0 0 1 1 1]

Accuracy: 1.00

Confusion matrix:
[[ 7  0]
 [ 0 13]]

Precision = [1. 1.]
Recall = [1. 1.]
F1 = [1. 1.]
ROC AUC = 1.0
```



Classes 2-3 (row 51-150 of the dataset)



```
Perceptron Model:
Perceptron(alpha=0.0001, class_weight=None, early_stopping=False, eta0=1.0,
fit_intercept=True, max_iter=1000, n_iter_no_change=5, n_jobs=None,
penalty=None, random_state=0, shuffle=True, tol=0.001,
validation_fraction=0.1, verbose=0, warm_start=False)

Test sample labels:
[2 2 1 1 2 2 2 1 1 2 2 2 1 1 1 2 1 2 2 1]
Test samples classified as:
[2 2 1 1 2 2 2 1 1 2 2 2 1 1 1 1 1 2 2 1]

Accuracy: 0.95

Confusion matrix:
[[ 9  0]
 [ 1 10]]

Precision = [0.9 1. ]
Recall = [1.         0.90909091]
F1 = [0.94736842 0.95238095]
ROC AUC = 0.9545454545454546
```



Requirements for the Submission of Programming/Homework Assignments

1. Well-documented program list (the .py files)

    20% of total points if no .py file submitted.

    Done

2. Three annotated program test and run examples (screenshots) that **show different and representative test cases** with **input, output, and the parameter settings of the program runs clearly marked/annotated**. You can do the annotations by

    (1) Pasting the screenshots into a WORD document,

    Done

    (2) Editing on the WORD document pages for the required marks and annotations,

    Done.

    Testing and running examples, as well as annotations, were provided inside the screenshots.

    (3) Converting the document to pdf for submission (it is ok to submit the WORD file directly without converting to pdf).

    Done

    20% of total points will be taken off if run examples are not representative.

    20% of total points will be taken off if run examples are not clearly marked/annotated.

3. A discussion page

    (a) Hardware and software used by your program,

    I completed this assignment using my personal computer with PyCharm Professional Version: 2023.2.1.

    (b) Features of your program, e.g., data structures, algorithms, programming styles, etc.

    The provided Python program performs two classification tasks using machine learning models: k-Nearest Neighbors (KNN) and a binary Perceptron. For KNN, the Iris dataset is loaded, and a 3D scatter plot is generated to visualize the data. The script then applies a KNN classifier, specifying the number of neighbors (k=5), and prints the true labels and predictions for a test set. The perceptron tasks involve both a general perceptron and a binary perceptron for classes 1, 2, and 3. The program splits the Iris dataset, standardizes features, trains the perceptron models, and evaluates their performance, including accuracy, confusion matrices, and precision-recall metrics.

    (c) Problems you encountered during your work, and

    None

    (d) Assigned discussion problems, if there is any.

    No assigned discussion problems

    (e) Fill in the following table and submit it along with your above submissions.

Total (approximate) time spent on the assignment	16 hours	Total (approximate) time for the correction part	1 hour
Problems and difficulties encountered	None		
Reflections (good and bad) on the assignment	Good: A snippet of lines of code was provided Bad: None		

Any comments and suggestions	None

20% of total points will be taken off if no discussion page is submitted.