CS451 Assignment 2 Report

Can Yılankıran S011483

In this assignment, the goal was successfully implementing a machine learning algorithms such as KNearest Neighbour Classifier, Naïve Bayes Classifier and Decision Tree Classifier over the MINST dataset.

Implementation:

I have created a main class and 3 classes for those algorithms. Splitting the data to train and test data is done only once at the main class and passed to the 3 classes __init__ functions. Furthermore, the validation data it split only once at the main class.

For Naïve Bayes there is no parameter can be changed other than the split ratio of the data. I split the data into %80 for train data, %20 for test data.

For KNearest Neighbour Classifier, we can select the n_neighbour parameter. To be able to observe the effect of this parameter I have a for loop starting from 1 to 30 growing by 2.

For Decision Tree Classifier we can select the criterion and splitter parameters. For criterion there is 2 options which are “gini” and “entropy”. The criterion parameter consists of 2 options which are “best” and “random”. It changes the behavior of algorithm while splitting.

Results:

NaïveBayesClassifier:

The results of naïve bayes classifier’s accuracy were approx. 0.85. Accuracy and confusion matrix are given below. The best performing parameters were %80 for train data and %20 for test data.

- **Confusion Matrix and Accuracy:**

	precision	recall	f1-score	support
0	1.00	1.00	1.00	15
1	0.83	0.83	0.83	12
2	1.00	0.71	0.83	14
3	1.00	0.75	0.86	16
4	1.00	0.86	0.92	14
5	0.86	0.92	0.89	13
6	0.95	1.00	0.97	18
7	0.74	1.00	0.85	17
8	0.41	0.64	0.50	11
9	0.90	0.64	0.75	14
accuracy			0.85	144
macro avg	0.87	0.84	0.84	144
weighted avg	0.88	0.85	0.85	144

```
[[15  0  0  0  0  0  0  0  0  0]
 [ 0 10  0  0  0  0  1  0  1  0]
 [ 0  1 10  0  0  0  0  0  3  0]
 [ 0  0  0 12  0  1  0  0  2  1]
 [ 0  0  0  0 12  0  0  2  0  0]
 [ 0  0  0  0  0 12  0  0  1  0]
 [ 0  0  0  0  0  0 18  0  0  0]
 [ 0  0  0  0  0  0  0 17  0  0]
 [ 0  0  0  0  0  1  0  3  7  0]
 [ 0  1  0  0  0  0  0  1  3  9]]
```

KNearestNeighbourClassifier:

The results of KNearestNeighbourClassifier's accuracy were approx. 0.96. Accuracy and confusion matrix are given below. The best performing parameters were %80 for train data, %20 for test data, and n_neighbour (printed as K at output) was 3.

- **Confusion Matrix and Accuracy:**

```
Train test for kNearest
K value: 3
0.9611111111111111
0.9611111111111111
[[34  0  0  0  1  0  0  0  0  0]
 [ 0 36  0  0  0  0  0  0  0  0]
 [ 1  0 34  0  0  0  0  0  0  0]
 [ 0  0  0 33  0  1  0  1  2  0]
 [ 0  0  0  0 34  0  0  0  1  2]
 [ 0  0  0  0  0 37  0  0  0  0]
 [ 0  0  0  0  0  0 37  0  0  0]
 [ 0  0  0  0  0  0  0 36  0  0]
 [ 0  2  0  0  0  0  0  0 31  0]
 [ 0  0  0  1  0  2  0  0  0 34]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        35
           1       0.95      1.00      0.97        36
           2       1.00      0.97      0.99        35
           3       0.97      0.89      0.93        37
           4       0.97      0.92      0.94        37
           5       0.93      1.00      0.96        37
           6       1.00      1.00      1.00        37
           7       0.97      1.00      0.99        36
           8       0.91      0.94      0.93        33
           9       0.94      0.92      0.93        37

    accuracy                           0.96       360
   macro avg       0.96      0.96      0.96       360
weighted avg       0.96      0.96      0.96       360
```

DecisionTreeClassifier:

The results of DecisionTreeClassifier's accuracy were approx. 0.84. Accuracy and confusion matrix are given below. The best performing parameters were %80 for train data, %20 for test data, and used criterion is "entropy" and used split method is "best".

- **Confusion Matrix and Accuracy:**

```
0.8472222222222222
[[32  0  0  0  2  0  0  1  0  0]
 [ 1 23  0  1  0  1  2  0  3  5]
 [ 1  2 28  0  0  0  1  1  2  0]
 [ 0  1  1 22  0  3  0  2  7  1]
 [ 0  0  0  1 33  0  0  3  0  0]
 [ 0  1  0  0  0 33  2  0  1  0]
 [ 0  1  0  1  0  1 33  0  1  0]
 [ 0  1  0  0  3  0  0 32  0  0]
 [ 1  3  1  0  0  2  0  1 23  2]
 [ 0  1  0  1  1  1  0  0  4 29]]
              precision    recall  f1-score   support

           0       0.88      0.93      0.90        15
           1       0.81      0.87      0.84        15
           2       1.00      0.43      0.60        14
           3       0.72      0.93      0.81        14
           4       0.88      1.00      0.93        14
           5       0.71      0.86      0.77        14
           6       0.92      0.73      0.81        15
           7       0.93      0.93      0.93        15
           8       0.87      0.93      0.90        14
           9       0.92      0.86      0.89        14

    accuracy                           0.85       144
   macro avg       0.86      0.85      0.84       144
weighted avg       0.86      0.85      0.84       144
```

Conclusion:

I have learned how to use the scikit library effectively with Python. I understand how parameters can affect the performance of a learning algorithm. Also, we can state that, the best performing algorithm among these 3 is, KNearestNeighbourClassifier algorithm. NaïveBayesClassifier came second according to my tests, I feel sad about naiveBayesClassifer because I really like this approach considering the ease of use and non-complex structure. Lastly, DecisionTreeClassifier was the worst one among these algorithms.