Machine Learning – Assignment 1

Submitted by

Debonil Ghosh (M21AIE225)

Question 1.

There are two models, M1 and M2, used to predict the scores for some input data. Suppose M1 predicts the score for input data as score1.npy and M2 predicts the score for the same data as score2.npy. Actual labels for a given score is label.npy

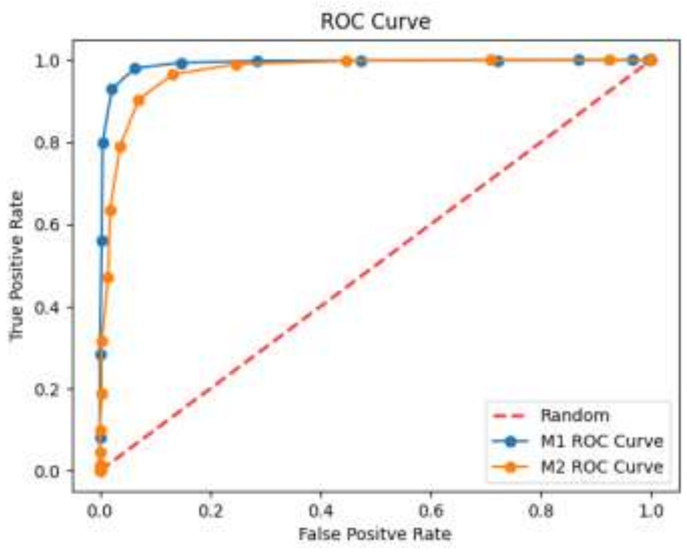
(use np.load to load .npy files)

1. Plot ROC curve (from scratch) for both the models in a single plot. (10 marks)
2. Explain which model performs better on this data and why? (5 marks)
3. Compute AUC for both the ROC curves. (5 marks)
4. Calculate true positive rate for both models when false acceptance rate is 10% (5 marks)
5. Draw your analysis on (3) and (4) (5 marks)

Note: Scores here represent the distance between two samples using two different models. 0 in the label represents similar samples and 1 represents different samples.

Solution:

1.



2. **Model M1** performs better on this data.
The curve that reaches closer to the top left corner of the graph denotes the better performing model. By visually comparing ROC graphs of given two models, it is found that ROC graph of M1 is slightly in top-left position with ROC curve of M2. It confirms that M1 will always give better (lesser False Positive points compare to True positive points) or same results than M2.
It is found that AUC of M1 is greater than AUC of M2 and that also supports the above statement.

3.

Area Under the curve (AUC) for M1: **0.9894237454500838**
Area Under the curve (AUC) for M2: **0.9681889933184838**

4. When false acceptance rate is 10%,

True positive rate for M1: **0.9858447507259417**
True positive rate for M2: **0.9334093356781591**

5. The Area Under the Curve is the measure of the ability of a model to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.
Here it is found that the area under the ROC curve of model M1 is **0.98** (approx.) and the same of model M2 is **0.96** (approx.). It indicates capability to distinguish between two classes is quite good for both of the models. Though Model 1 is little better performance as compared to model M2.

By intercepting both the ROC Curve, True positive rate is calculated when the false acceptance rate is 10%. When false acceptance rate is 10%, TPR for M1 is **0.98** (approx.) and for M2: **0.93** (approx.). These means, if we allow 10% false positive cases, then we can get about 98% and 93% sensitivity out of these two models respectively.

Question 2.

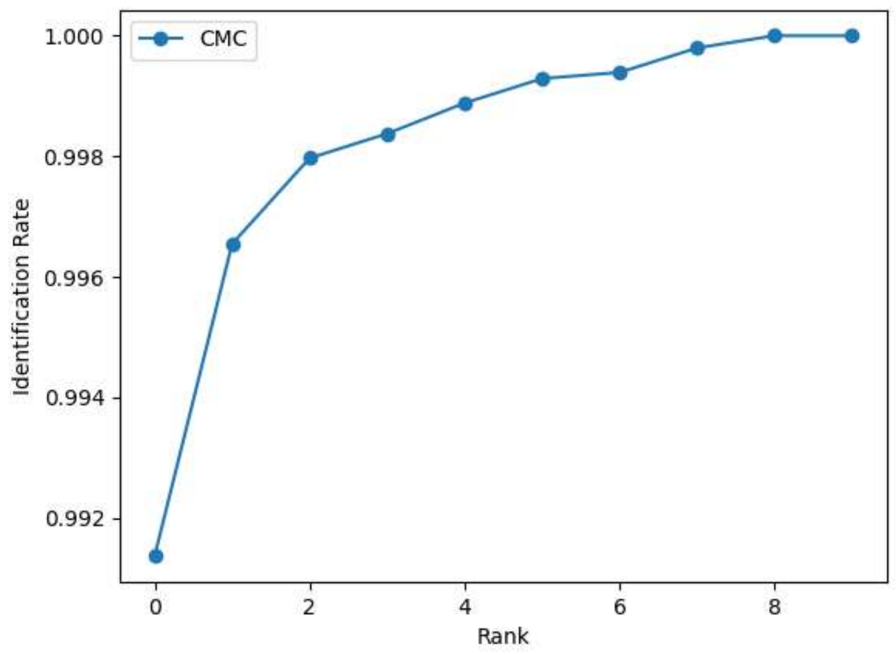
Dataset link: Link

Consider a fingerprint recognition dataset, having 600 images in the gallery and 9854 images in the probe. A model is used to classify probe images into 600 classes. The probabilities predicted by the model for all 600 gallery images are given in score.npy. The correct labels are given in label.npy.

1. Plot CMC curve up to rank 10. (10 marks)
2. Comment on the results

Solution:

1.



2. **Cumulative Match Characteristic** or **CMC** curve is a tool to compare Rank vs identification accuracy of a multi class classifier. Here Rank is index of sorted array of predicted probabilities of matching a sample with different classes by the given classifier. That means the class that matches most with the sample will have rank 1, next will come at 2 and so on, as per predicted results. But in reality, rank 1 predicted class may not match to the actual, we may need to check next predicted classes for it. In this case CMC curve helps to visualise the performance of the model. It shows how much rank we should visit to get a certain accuracy or more precisely an identification rate.
In plotted graph, identification rate started below 99.2% then with increasing cumulative ranks, it tends to 99.99% near rank 10. That could be interpreted like, top ten predicted results match with 99.99% accuracy.

Question 3.

You are requested to solve the fruit classification problem based on the features in the given dataset using decision trees. Load this dataset for your decision tree classification problem. The dataset has 3 features and one target variable. The target variable takes either Papaya (0) or Banana (1). The features are “Size” in cm, “Weight” in kg, and “SkinColor” (100-green, 200-yellow, and 300 orange).

- ● Load (Train-Test Split) and prepare required packages and shuffle the dataset. (2 marks)
- ● Build and Train a DecisionTree classifier. (5 marks)
- ● Don't stick to a single configuration for your model. Try different hyperparameters. (At least 5) (3 marks)
- ● Test the model for each configuration (5 marks)
- ● Visualize the tree, evaluate it based on the metrics given in previous questions. (3 marks)
- ● Report the confusion matrix for your best model (don’t use inbuilt function) (2 marks)

If the hyperlink doesn't work copy-paste the URL below -

https://drive.google.com/file/d/1O-Txgca54gFn0cTszrYq3n7OIKnz5o2m/view?usp=sharing

Solution:

Testing Model with Different Hyper Parameters:

- All same with increasing **min_samples_split** from **3 to 10:**

```
Decision Tree Classifier Created with min_samples_split=3, max_depth=10,train_size=0.8, random_state=80
Accuracy Score: 0.9
Decision Tree Classifier Created with min_samples_split=5, max_depth=10,train_size=0.8, random_state=80
Accuracy Score: 0.925
Decision Tree Classifier Created with min_samples_split=8, max_depth=10,train_size=0.8, random_state=80
Accuracy Score: 0.925
Decision Tree Classifier Created with min_samples_split=10, max_depth=10,train_size=0.8, random_state=80
Accuracy Score: 0.975
```

- All same with increasing **max-depth** from **1 to 10:**

```
Decision Tree Classifier Created with min_samples_split=3, max_depth=0,train_size=0.8,
random_state=67
Accuracy Score: 0.9
Decision Tree Classifier Created with min_samples_split=3, max_depth=1,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=2,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=3,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=4,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=5,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=6,train_size=0.8,
random_state=67
Accuracy Score: 0.975
Decision Tree Classifier Created with min_samples_split=3, max_depth=7,train_size=0.8,
random_state=67
Accuracy Score: 0.925
Decision Tree Classifier Created with min_samples_split=3, max_depth=8,train_size=0.8,
random_state=67
Accuracy Score: 0.9
Decision Tree Classifier Created with min_samples_split=3, max_depth=9,train_size=0.8,
random_state=67
Accuracy Score: 0.9
```

- All same with increasing **training-test ratio** from **0.05 to 0.95:**

```
Train Ratio: 0.05            Accuracy Score: 0.8052631578947368
 Train Ratio: 0.1            Accuracy Score: 0.8722222222222222
 Train Ratio: 0.15           Accuracy Score: 0.8529411764705882
 Train Ratio: 0.2            Accuracy Score: 0.925
 Train Ratio: 0.25           Accuracy Score: 0.9466666666666667
 Train Ratio: 0.3            Accuracy Score: 0.9428571428571428
 Train Ratio: 0.35           Accuracy Score: 0.9384615384615385
 Train Ratio: 0.4            Accuracy Score: 0.9416666666666667
 Train Ratio: 0.45           Accuracy Score: 0.9545454545454546
 Train Ratio: 0.5            Accuracy Score: 0.97
 Train Ratio: 0.55           Accuracy Score: 0.9666666666666667
 Train Ratio: 0.6            Accuracy Score: 0.9625
 Train Ratio: 0.65           Accuracy Score: 0.9714285714285714
 Train Ratio: 0.7            Accuracy Score: 0.9833333333333333
 Train Ratio: 0.75           Accuracy Score: 0.98
 Train Ratio: 0.8            Accuracy Score: 0.975
Train Ratio: 0.85                    Accuracy Score: 0.9666666666666667
Train Ratio: 0.9              Accuracy Score: 0.95
Train Ratio: 0.95                    Accuracy Score: 1.0
```

Testing with 5 different set of parameters:

1.

Decision Tree Classifier Created with min_samples_split=3, max_depth=3,train_size=0.1, random_state=80

Accuracy Score: 0.8944444444444445

Confusion Matrix:

```
[[78 15]
 [ 4 83]]
```

Classification report -

```
           precision    recall  f1-score   support


         0       0.85      0.95      0.90        87
         1       0.95      0.84      0.89        93


  accuracy                           0.89       180
   macro avg       0.90      0.90      0.89       180
weighted avg       0.90      0.89      0.89       180
```

2.

Decision Tree Classifier Created with min_samples_split=3, max_depth=3,train_size=0.5, random_state=80

Accuracy Score: 0.96

Confusion Matrix:

```
[[54  1]
 [ 3 42]]
```

Classification report -

```
           precision    recall  f1-score   support
           0       0.98      0.93      0.95        45
           1       0.95      0.98      0.96        55

    accuracy                           0.96       100
   macro avg       0.96      0.96      0.96       100
weighted avg       0.96      0.96      0.96       100
```

3.

Decision Tree Classifier Created with min_samples_split=3, max_depth=5,train_size=0.6, random_state=80

Accuracy Score: 0.95

Confusion Matrix:

```
[[43  2]
 [ 2 33]]
```

Classification report -

```
              precision    recall  f1-score   support

           0       0.94      0.94      0.94        35
           1       0.96      0.96      0.96        45

    accuracy                           0.95        80
   macro avg       0.95      0.95      0.95        80
weighted avg       0.95      0.95      0.95        80
```

4.

Decision Tree Classifier Created with min_samples_split=5, max_depth=5,train_size=0.75, random_state=80

Accuracy Score: 0.98

Confusion Matrix:

```
[[31  1]
 [ 0 18]]
```

Classification report -

```
              precision    recall  f1-score   support
```

```
           0       0.95      1.00      0.97        18
           1       1.00      0.97      0.98        32

    accuracy                           0.98        50
   macro avg       0.97      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```

5.

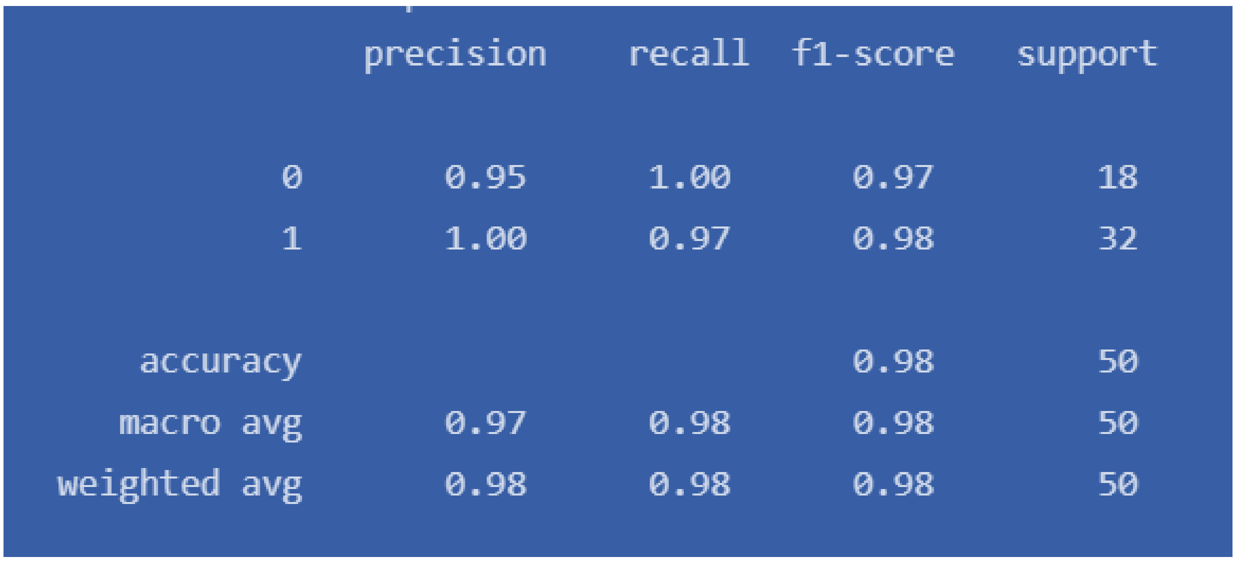
Decision Tree Classifier Created with min_samples_split=5, max_depth=10,train_size=0.9, random_state=80

```
Accuracy Score: 0.95
Confusion Matrix:
[[13  1]
 [ 0  6]]
Classification report -
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       1.00      0.93      0.96        14

    accuracy                           0.95        20
   macro avg       0.93      0.96      0.94        20
weighted avg       0.96      0.95      0.95        20
```

Visualize the tree:

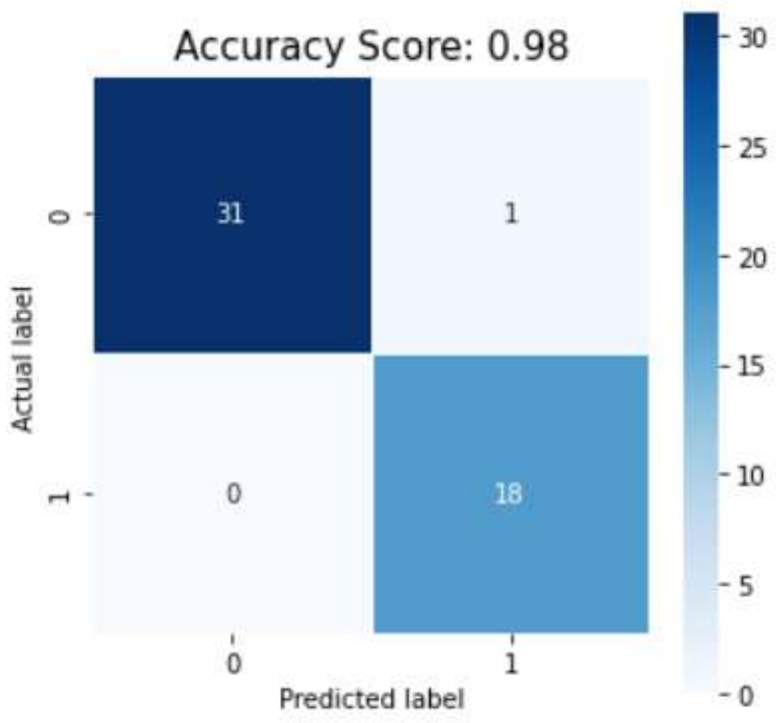
```
X_1 <= 0.865816898 ? 0.3408476964769649
 left:X_0 <= 25.0 ? 0.1695663866094057
  left:X_0 <= 15.0 ? 0.02410291695335795
    left:0.0
    right:X_2 <= 111.0 ? 0.025126065032168327
        left:0.0
        right:X_0 <= 23.0 ? 0.003983697318353158
                left:X_1 <= 0.369486225 ? 0.0014778804083617064
                                left:1.0
                                right:1.0
                right:X_2 <= 193.0 ? 0.040178571428571425
                                left:1.0
                                right:1.0
  right:0.0
 right:0.0
```

Classification report -



```
              precision    recall  f1-score   support

           0       0.95      1.00      0.97        18
           1       1.00      0.97      0.98        32

    accuracy                           0.98        50
   macro avg       0.97      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```

Confusion Matrix:


Accuracy Score: 0.98
Actual label
Predicted label