Machine Learning: Classification Models
(SEP 787)

COURSE PROJECT: COMPARING CLASSIFIERS

Submitted to:

Dr. Jeff Fortuna
Assistant Professor,
W Booth School of Engineering Practice and Technology

Submitted by:

Neelanjan Goswami
Student Id: 400414867

Sumanth Mahabaleshwar Bhat
Student Id: 400351124

1. DATASET DESCRIPTION:

HTRU2 is a data set which describes a sample of pulsar candidates collected during the High Time Resolution Universe Survey (South). Pulsars are a rare type of Neutron star that produce radio emission detectable here on Earth. They are of considerable scientific interest as probes of space-time, the inter-stellar medium, and states of matter. Each candidate is described by 8 continuous variables, and a single class variable. The first four are simple statistics obtained from the integrated pulse profile (folded profile). This is an array of continuous variables that describe a longitude-resolved version of the signal that has been averaged in both time and frequency. The remaining four variables are similarly obtained from the DM-SNR curve. This project's dataset can be found on UCI ML Datasets at http://archive.ics.uci.edu/ml/datasets/HTRU2. Dataset attributes are:

1. Mean of the integrated profile.
2. Standard deviation of the integrated profile.
3. Excess kurtosis of the integrated profile.
4. Skewness of the integrated profile.
5. Mean of the DM-SNR curve.
6. Standard deviation of the DM-SNR curve.
7. Excess kurtosis of the DM-SNR curve.
8. Skewness of the DM-SNR curve.
9. Class

HTRU 2 Summary

17,898 total examples.

1,639 positive examples.

16,259 negative examples.

Algorithms Used

1. KNN
2. Naive Bayes
3. SVM
4. Adaboost

1.1 Dataset Pre-processing:

- The pre-processing involved checking for the null and NaN values. The following table has the Percentage of null values under each attribute.

```
Mean of the integrated profile                0
Standard deviation of the integrated profile  0
Excess kurtosis of the integrated profile     0
Skewness of the integrated profile            0
Mean of the DM-SNR curve                      0
Standard deviation of the DM-SNR curve        0
Excess kurtosis of the DM-SNR curve           0
Skewness of the DM-SNR curve                  0
Class                                         0
dtype: int64
```

- The following table has the Percentage of NaN values under each attribute.

	Column_Name	% Missing
Mean of the integrated profile	Mean of the integrated profile	0.0
Standard deviation of the integrated profile	Standard deviation of the integrated profile	0.0
Excess kurtosis of the integrated profile	Excess kurtosis of the integrated profile	0.0
Skewness of the integrated profile	Skewness of the integrated profile	0.0
Mean of the DM-SNR curve	Mean of the DM-SNR curve	0.0
Standard deviation of the DM-SNR curve	Standard deviation of the DM-SNR curve	0.0
Excess kurtosis of the DM-SNR curve	Excess kurtosis of the DM-SNR curve	0.0
Skewness of the DM-SNR curve	Skewness of the DM-SNR curve	0.0
Class	Class	0.0

- Since all the attributes played a vital role in the classification, elimination of any of the attributes was not ideal.

1.2 Project Objectives:

- To apply data classification methods such as Naïve Bayes, KNN (KNearest Neighbor), AdaBoost, and SVM (Support Vector Machine) in order to classify the sample of pulsar candidates collected during the High Time Resolution Universe Survey (South).
- To compare the results derived from the classifications and obtain the below results:
    1. Computational times for both training and testing
    2. Cross Validation for parameter selection
    3. Plot and compare ROC (Receiver Operating Characteristic)
    4. Confusion matrix of results from each classifier

2. Machine Learning Model Training and Testing:

2.1 Training and Testing Data:

- Before performing classification techniques, the data is split into testing and training data. The data split is 75 percent of observations as training data and 25 percent testing data from the total of 3276 observations.

```
X_train.shape  X_test.shape
(13423, 8)     (4475, 8)
```

2.2 Parameters calculated for evaluating the model:

Training Time (in seconds): Time taken for the classifier to be trained on the trained data.

Testing Time (in seconds): Time taken for the classifier to be evaluated on the testing data

Confusion Matrix:

True Negative	False Positive
False Negative	True Positive

Cross Validation: A method of training numerous models on subsets of the input data and then assessing them on a different subset of the data

Accuracy: Percentage of correct predictions for the test data.

$$\frac{True\ Positive + True\ Negative}{Total\ Observations}$$

$$Total\ Observations = True\ Pos. + True\ Neg. + False\ Pos. + False\ Pos.$$

Recall: Measure of the model identifying True Positives

$$\frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1 Score: Weighted average of Precision and Recall

$$2 \times \frac{Precision * Recall}{Precision + Recall}$$

Precision: Quality of a positive prediction made by the model

$$\frac{True\ Positive}{True\ Positive + False\ Positive}$$

R^2 : Metric used for evaluating the performance of a model. It operates by calculating the amount of variation in the dataset's predictions.

3. CLASSIFICATION ALGORITHMS:

We have applied four classifier algorithms to the dataset and are listed below:

1. K-Nearest Neighbours (KNN)
2. Naive Bayes
3. AdaBoost
4. Support Vector Machines (SVM)

3.1 Classification with K-Nearest Neighbours (KNN)

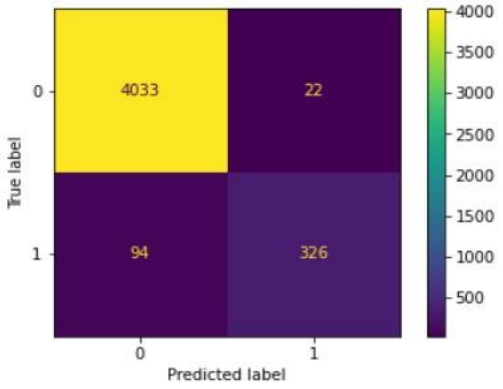
- K-Nearest Neighbours (KNN) is a supervised learning algorithm. It assumes that things with similar attributes tend to cluster together. This similarity is also referred to as proximity. The KNN plots the data points in clusters, having the least distance from each other. When a new data point appears, it is easily categorized in its respective class. It works both on classification and regression problems.

- One issue inherent with the KNN algorithm is the possibility of neglecting classes with fewer samples in the training dataset, but our dataset requires classification in only two categories (Class 0 & Class 1), and this issue is not impactful for our dataset. Through trial and error, we found that the ideal number of neighbours to include is 50, as anything greater or lesser resulted in less score value and less cross-validation.

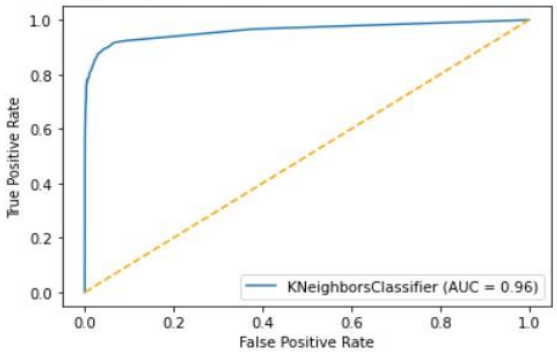
- Observations on applying KNN classifier is depicted below:

```
Training Time     : 0.038
Testing Time  : 0.32
```

```
True Negative    :  4033
False Negative   :  94
True Positive    :  326
False Positive   :  22

Accuracy         :  0.9740782122905028
Recall           :  0.9740782122905028
Precision        :  0.9734275325175813
F1 Score         :  0.9740782122905027

Test-set R-square value : 0.97

Cross Validation :  [0.96872673 0.97691735 0.97468354 0.9709389  0.97168405 0.97391952
 0.98211624 0.96348733 0.96944858 0.96721311]

Confusion Matrix :
[[4033   22]
 [  94  326]]
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fc1f5d43190>
```



Plotting KNN ROC Curve:



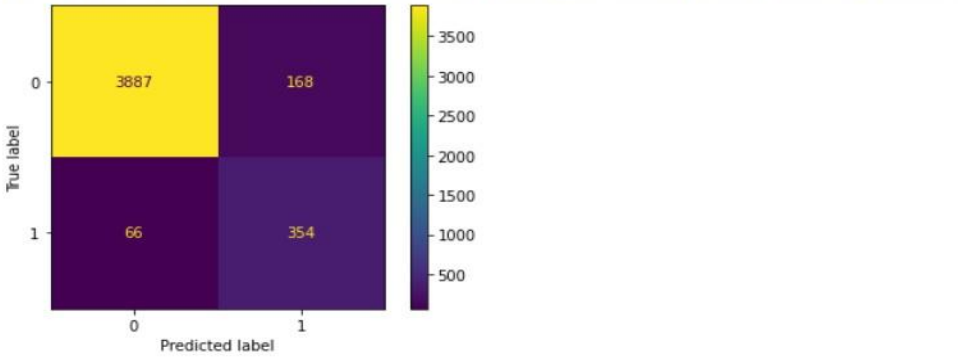
3.2 Classification with Naive Bayes

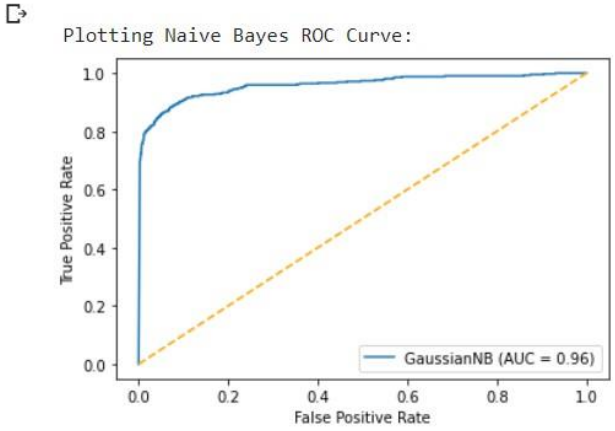
- Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values. It is called naive Bayes or idiot Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable.

- Rather than attempting to calculate the values of each attribute value P(d1, d2, d3|h), they are assumed to be conditionally independent given the target value and calculated as P(d1|h) * P(d2|H) and so on.

- Observations on applying Naive Bayes classifier is depicted below:

```
Training Time: 0.012
Testing Time: 0.011

True Negative    :  3887
False Negative   :  66
True Positive    :  354
False Positive   :  168

Accuracy         :  0.9477094972067039
Recall           :  0.9477094972067039
Precision        :  0.9546647096838049
F1 Score         :  0.9477094972067039

Test-set R-square value : 0.95
Cross Validation :  [0.94936709 0.95160089 0.93968727 0.9485842  0.94113264 0.94560358
 0.94634873 0.93368107 0.94411326 0.94411326]

Confusion Matrix :
[[3887  168]
 [  66  354]]
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fc1ded90090>
```





3.3 Classification with AdaBoost

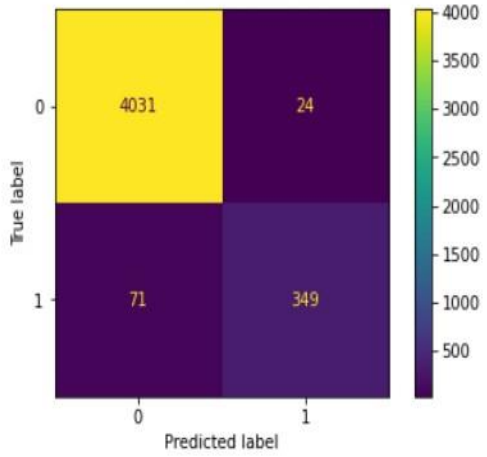
- AdaBoost, short for Adaptive Boosting, is a machine learning model which is highly effective in getting good accuracy. In AdaBoost, a model is built, and equal weights are assigned to every single data point. It then allocates better weights to the data that are incorrectly classified. So, after competition of this cycle all the incorrectly classified data is prioritized more in the next model. This will continue in iterations, until least error is received.

- Overall, the boosting model works in such a way that a new model is generated every time and errors are rectified in each model. AdaBoost provides an accuracy of 97.8% on this dataset, and we have used 100 estimators (the number of models to iteratively train).

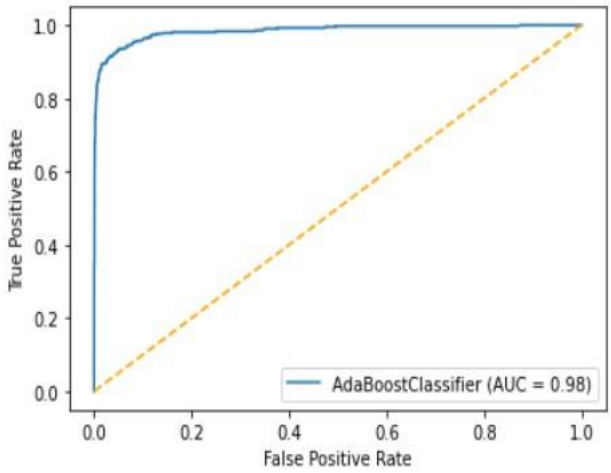
- Observations on applying AdaBoost classifier is depicted below:

```
Training Time    : 3.0
Testing Time  : 0.09
```

```
True Negative    :  4031
False Negative   :  71
True Positive    :  349
False Positive   :  24

Accuracy         :  0.9787709497206704
Recall           :  0.9787709497206704
Precision        :  0.9782769557364679
F1 Score         :  0.9787709497206704

Test-set R-square value : 0.98
Cross Validation :  [0.97691735 0.98212956 0.97840655 0.97839046 0.97913562 0.98137109
 0.98435171 0.97242921 0.97913562 0.97317437]

Confusion Matrix :
[[4031   24]
 [  71  349]]
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fc1deb5f410>
```



AdaBoost ROC Curve:



3.4 Support Vector Machine (SVM)

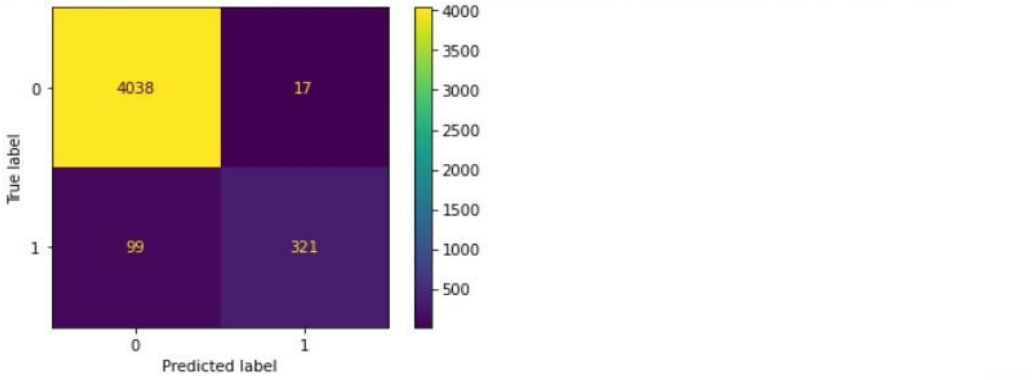
- In SVM, a support vector is utilized to develop soft margins based on the input variables to classify new samples. It incrementally increases the number of dimensions of the dataset until an optimal support vector can be used to classify the data.

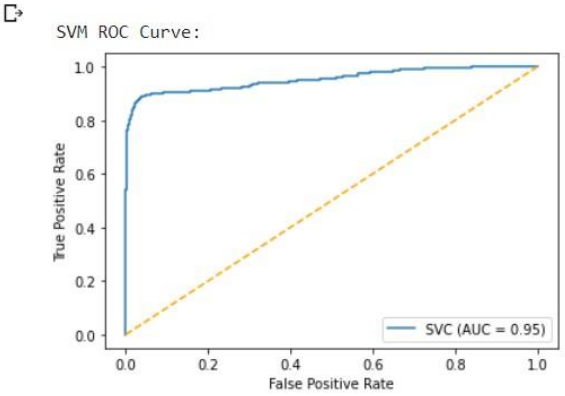
- This way, a new data point can be categorized into its respective class faster. It is a supervised machine learning algorithm SVM provided an accuracy of 97.4 percent on the dataset.

- Observations on applying SVM classifier is depicted below:

```
Training Time    : 2.0
Testing Time  : 0.72
```

```
True Negative    :  4038
False Negative   :  99
True Positive    :  321
False Positive   :  17

Accuracy         :  0.9740782122905028
Recall           :  0.9740782122905028
Precision        :  0.9735950907621367
F1 Score         :  0.9740782122905027

Test set R-square value : 0.97
Cross Validation :  [0.96351452 0.97766195 0.97393894 0.97019374 0.97391952 0.97615499
 0.98137109 0.96497765 0.97019374 0.96944858]

Confusion Matrix :
[[4038   17]
 [  99  321]]
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fc1dedf2510>
```

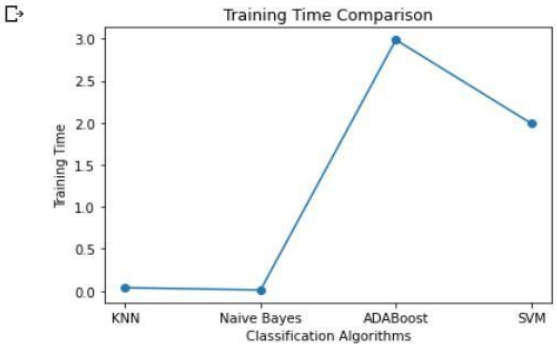




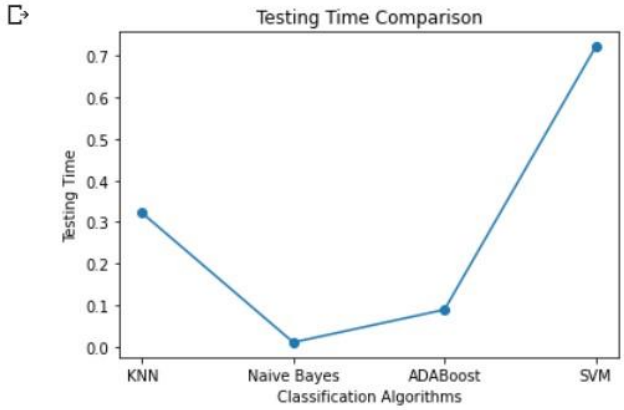
4 RESULTS - Comparison & Inference

4.1 Analysing the Computational Times for both training and testing:

- With respect to the computational time, Naive Bayes is quicker in testing the model compared to all the other algorithms, taking a time of 0.011 seconds in predicting results. However, Adaboost gives the best result in training the model, as it took about 3 seconds of training time. SVM took 2 seconds of training time and 0.72 seconds of testing time which is comparatively higher, as the algorithm itself is a bit complex for this dataset. KNN classifier had the highest training time compared to all the other classification algorithms. It was about 0.038 seconds in training the model.
- The comparison results of the training and evaluation time for the classifiers are shown below:

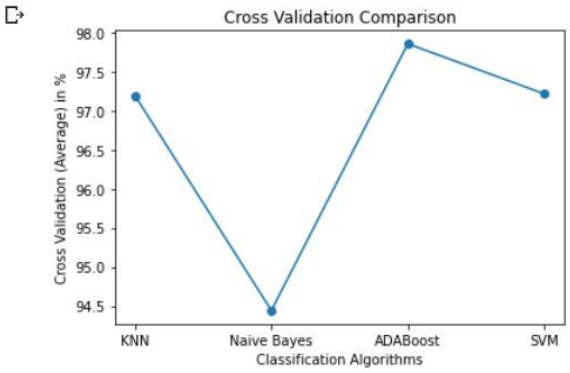


-



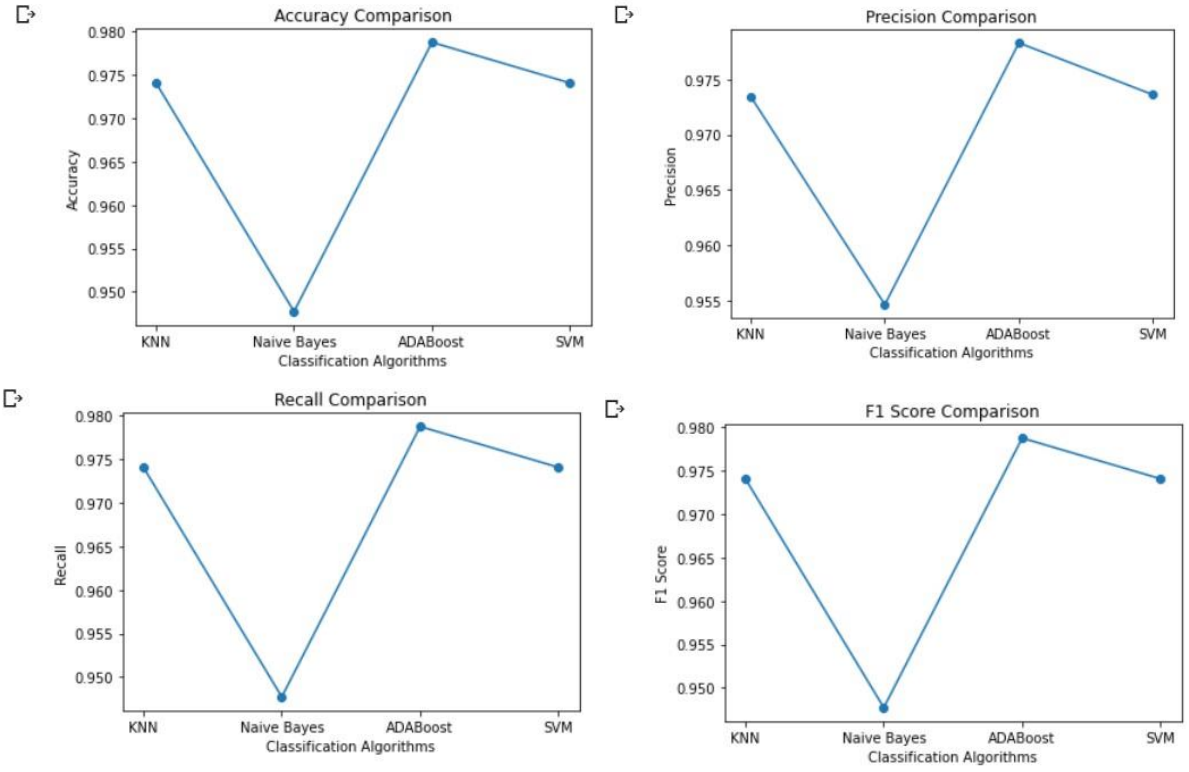
4.2 Cross Validation:

- Based on the initial analysis we could say that KNN was giving the best result with respect to time and accuracy combined. But on performing cross-validation, the analysis differed a bit, KNN had about 97.3% (approx.) of cross-validation average score, whereas SVM gave the cross-validation average score of 97.4% (approx.) which is the highest of all the classification models.
- AdaBoost gave the cross-validation average score of 97.7% (approx.). However, the least cross-validation average score was given by KNN, which is 97.3%.
- The comparison of cross-validation (average) results for all the approaches are presented below:



4.3 Classifier Comparison based on Accuracy, Precision, F-1 Score and Recall:

- Looking into the accuracy, precision, F-1 score and recall of all the algorithms that were implemented, we could say that SVM results the highest accuracy, f-1 score and recall, but provides the least precision.
- Naive Bayes gave 94.7% of accuracy, f-1 score and recall, and the precision was about ~94%, which is the least of all other classification algorithms.
- However, we could say that AdaBoost and SVM were the best in giving the good results for this dataset, but we cannot neglect the fact for AdaBoost that it's way costlier in terms of its performance timings.
- The results for accuracy, precision, and recall scores comparison between the classifiers is given below:



5. CONCLUSION

- From the above results, we can conclude that there isn't much difference in accuracy, F-1 score, recall, and cross-validation scores of AdaBoost, KNN and SVM, except for the precision of SVM which is the lowest. However, the most important and noticeable difference is the computation time for the SVM and AdaBoost.

- All the four Classification Algorithms suited well for this dataset. Although, Naive Bayes had the best training time, it had the least accuracy, recall and cross validation score comparing to others.

- Therefore, KNN is the best fit algorithm for this dataset with respect to all the attributes - computational time, accuracy, precision and cross-validation scores.