# Classifiers Comparison

The Boolean satisfiability (SAT) problem consists in determining whether a Boolean formula F is satisfiable or not. F is represented by a pair (X, C), where X is a set of Boolean variables and C is a set of clauses in Conjunctive Normal Form (CNF). Each clause is a disjunction of literals (a variable or its negation). This problem is one of the most widely studied combinatorial problems in computer science. It is the classic NP-complete problem. Over the past number of decades a significant amount of research work has focused on solving SAT problems with both complete and incomplete solvers.

Recent advances in supervised learning have provided powerful techniques for classifying problems. In this project, we see the SAT problem as a classification problem; given a Boolean formula (and represented by a vector of features) we are asked to predict it as:

- SAT or (ebglucose_solved = 1) if the problem is satisfiable.
- UNSAT or (ebglucose_solved = 2) if the problem is unsatisfiable.
- UNKNOW or (ebglucose_solved =0) if current solvers are not capable of identifying the satisfiability status.

In this project, we represent SAT problems with a vector of 126 features with general information about the problem, e.g., number of variables, number of clauses, fraction of horn clauses in the problem, etc. Complete descriptions of the vector of features are available at http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/.

## 1. Basic Models:

Without observing the dataset and without any tuning or preprocessing the data, two classifies (K-Nearest Neighbors and Random Forest) are executed. In this basic evaluation 70% of the data are used for the training and 30% of the data used for testing.

**KNN classifier:**
**Test Accuracy Score: 0.594**

**Random Forest:**
**Test Accuracy Score: 0.783**

**Classification Report:**

| | precision | recall | f1-score | support | precision | recall | f1-score | support |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 0 | 0.47 | 0.56 | 0.51 | 93 | 0.69 | 0.81 | 0.75 | 93 |
| 1 | 0.58 | 0.43 | 0.50 | 116 | 0.82 | 0.60 | 0.78 | 116 |
| 2 | 0.69 | 0.75 | 0.72 | 141 | 0.82 | 0.91 | 0.9 | 141 |
| | | | | | | | | |
| accuracy | | | 0.59 | 350 | | | 0.78 | 350 |
| macro avg | 0.58 | 0.58 | 0.58 | 350 | 0.78 | 0.77 | 0.77 | 350 |
| weighted avg | 0.60 | 0.59 | 0.59 | 350 | 0.79 | 0.78 | 0.78 | 350 |

**Table 1: Test Accuracy and Classification Report for Classifiers (KNN and Random Forest)**

The classification results can be seen in **Table1** above. It can be seen Random forest has produced better testing accuracy (78.3%) than KNN (59.4%). The confusion matrices of the classifiers are given below in **Fig1.**

# Classifiers Comparison

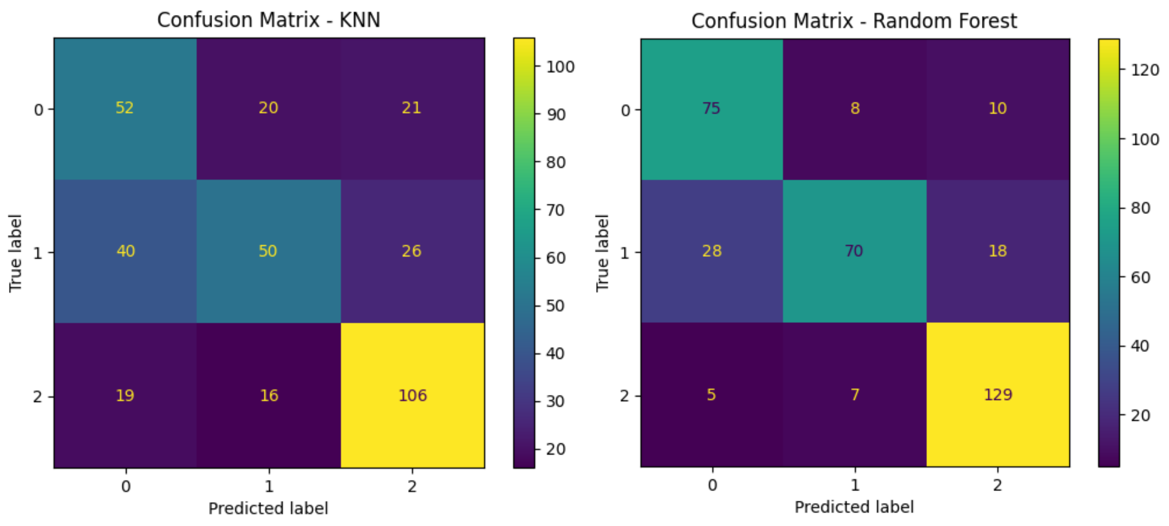


**Fig 1: Confusion matrices for KNN and Randeon forest  classifier**

## 2. Dataset Observation:

From the data types of the dataset it can be seen that there are only numetric  values, encoding is not needed in this case. From the featureset distributions there might be possibility of outliers, few feature distributions are given in **Fig2.** It is clear that the features are in different scale.

**Dataset datatypes:**

```
float64    105
int64       22
dtype: int64
```

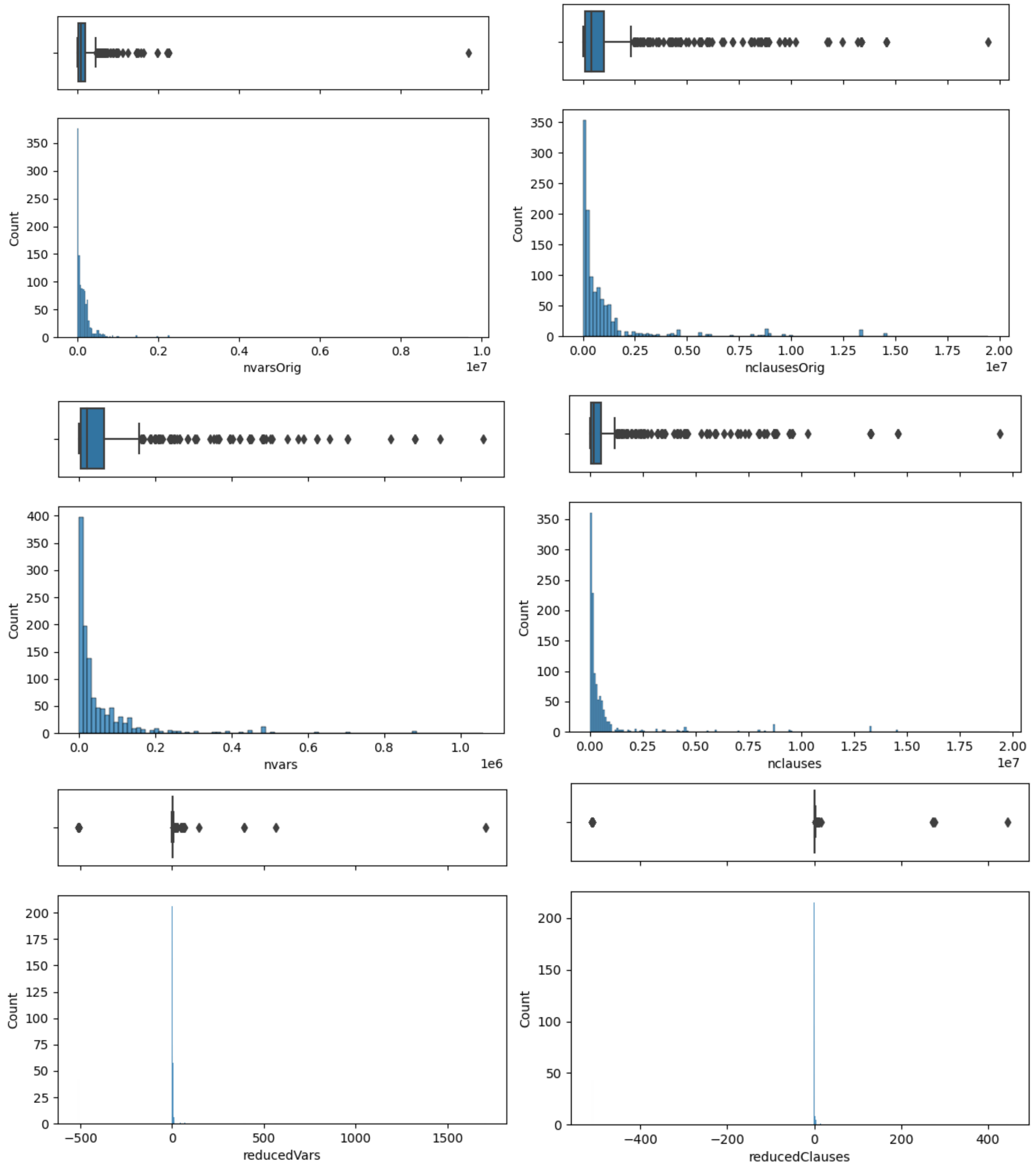


**Fig:2 Data distribution for few of the Features**

On checking missing vaues it can be seen there are no missing values present.

The class level counts are as below.

**Class Labels:  Name: ebglucose_solved, dtype: int64**

| | |
|---|---|
| **2** | **494** |
| **0** | **347** |
| **1** | **325** |

# Classifiers Comparison

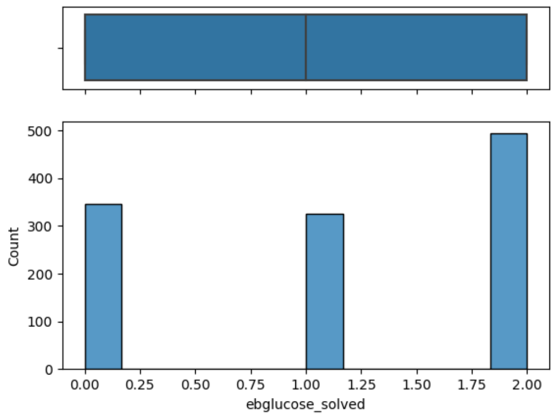


**Fig3: Class levels**

## 3. Rubust Evaluation:

A detailed analysis performed with  multiple classifiers.
The classifers which are choosen are **Naive Bayes, Ridge Classifier, K-Nearest Neighbours, Decision Tree, Bagging, Random Forest, Suport Vector Machines**.
The classifiers are executed with scaled data using Standard Normalization with Repeated Stratified Cross Validation with 5 splits and 3 repeates. Grid search is used for hyper parameter optimization .
Only for Naïve Bayes classifer scaling is done Power transformation .
Classification reports are generated for the total number of observations(outlier inclusive).

### 3.1 Data Preprocessing:

As the dataset has no missing values no need
Sandard Normalization is used for feature normalization only for Naive Bayes Power Transformation is used.
From the dataset observation it is clear there is no need for handling categorical data .
As the features were numerical data and it is a classification problem for feature selection ANOVA corelation coeffients is used and top k variables are used for classification. Principle componet analysys is used for dimensionality reduction.
For outlier detection Local Outlier Factor is used with neighbor and the lowest 0.5% data was removed.

The Label and the features are seperated and ‘INSTANCE _ID’ was removed from the featureset.

## 3.2 Naive Bayes Evaluation:

**Hyper parameters:**
For execution of Naive Bayes classifier a list of **smoothing parameter** values beetween ‘0.001’ to ‘1’ are choosen for hyper parameter optimization. For dimensionality reduction PCA was used and the n components were choosen are 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125 for optimization.

**Evaluation:**
From **Fig4** it is seen that with increasing smoothing parameter value classifier accuracy reduces.

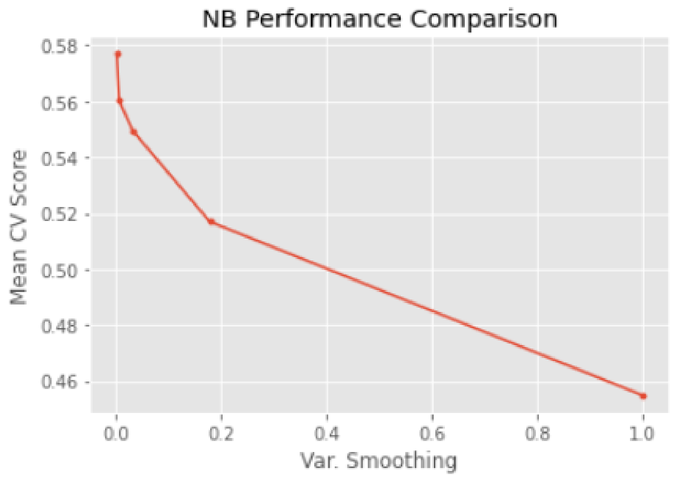


**Fig4: Max CV scores for different Var. Smoothing parameter values**

**Classification report:**

```
              precision    recall  f1-score   support
           0       0.46      0.72      0.56       347
           1       0.52      0.28      0.36       325
           2       0.70      0.64      0.67       494

    accuracy                           0.56      1166
   macro avg       0.56      0.55      0.53      1166
weighted avg       0.58      0.56      0.55      1166
```

**Confusion Matrix:**

```
     0    1    2
0  250   44   53
1  152   90   83
2  138   39  317
```

# Classifiers Comparison

**Learning Curve:**

From the learing curve in **Fig5** it is visible that the training and cross-valiadtion curves are close together and the accuracy is not great. Maybe the model is sufering from high bias different models can be tried .

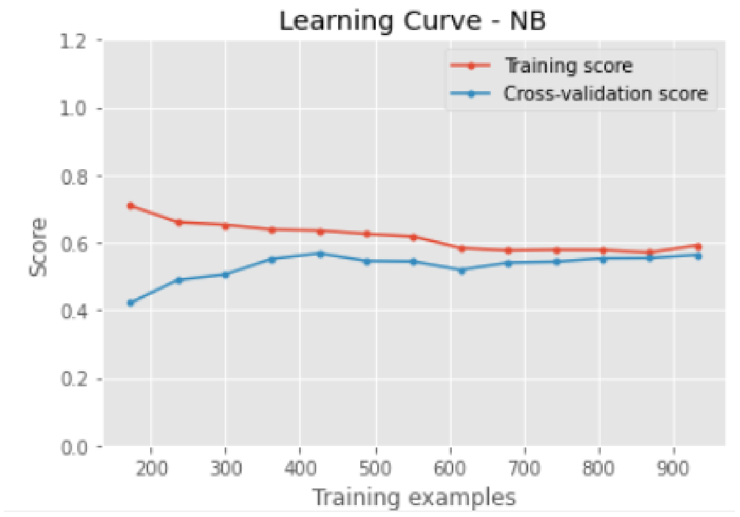


**Fig5: Learning curve for Naive Bayes**

**The best Accuracy is achieved is 57.7% for Smoothing parameter value 0.001 and 115 principle components .**

## 3.2 Ridge Classifier Evaluation:

**Hyper parameters:**

For execution of Ridge classifier a list of **alpha values** between '0.0001' to '1' are choosen for hyper parameter optimization. For dimensionality reduction PCA was used and the n components were choosen are 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125 for optimization.

**Evaluation:**

From **Fig6** it is seen that with increasing Alpha value classifier accuracy reduces.

# Classifiers Comparison

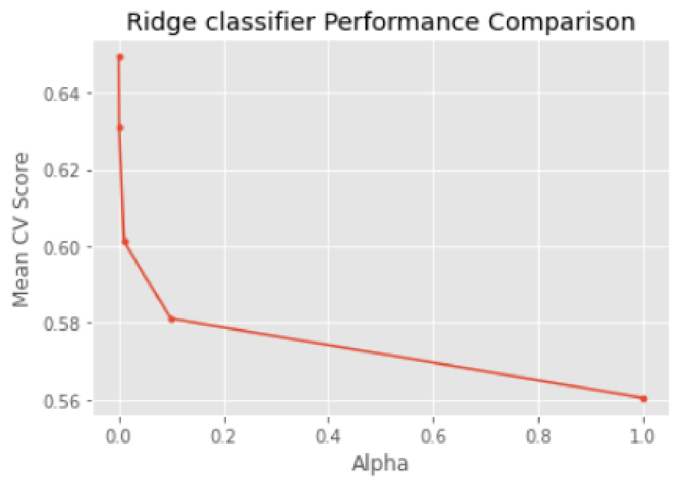


**Fig6: Max CV scores for different Alpha values**

**Classification report:**

```
              precision    recall  f1-score   support
           0       0.59      0.66      0.62       347
           1       0.59      0.46      0.51       325
           2       0.73      0.77      0.75       494

    accuracy                           0.65      1166
   macro avg       0.63      0.63      0.63      1166
weighted avg       0.65      0.65      0.65      1166
```

**Confusion Matrix:**

```
      0     1     2
0     230   51    66
1     101   148   76
2     61    53    380
```

**Learing curve:**

From **Fig7** it is noticed training and cross-validation accuracy is less and it doesn't seem to improve with more training examples.it suggests that model is overfitting. Maybe reducing the alpha value will produce better results, trying with other models also an option.

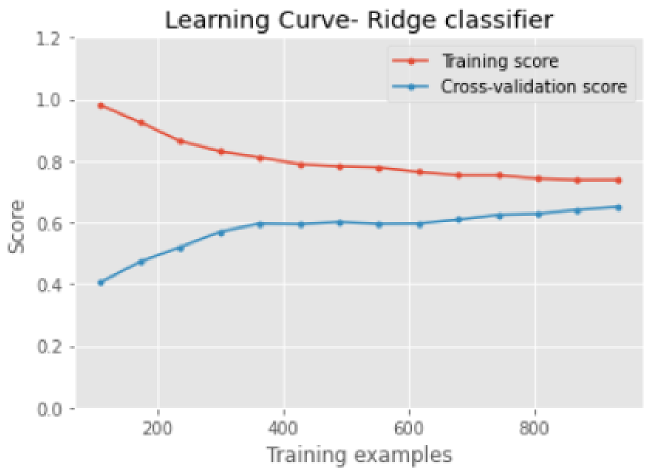


**Fig7: Learning curve for Ridge classifier**

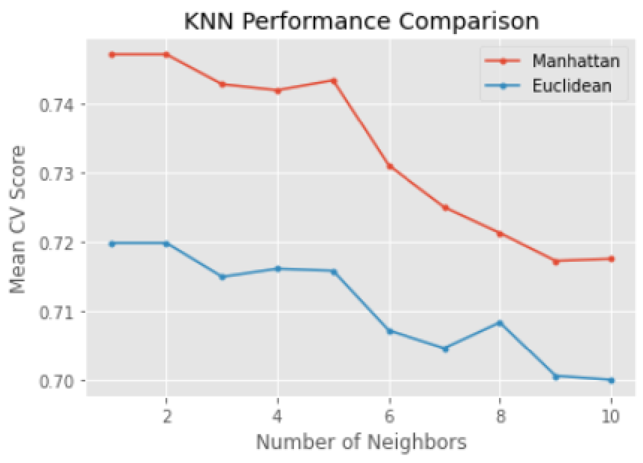
**The best Accuracy is achieved is 64.9% for alpha value 0.0001 and 105 principle components .**

### 3.3 KNN Robust Evaluation:

**Hyper Parameters:**

For execution of KNN a list of **K neighbour** values between 1 to 10; **distance metric** 'Manhattan' and 'Euclidean' ; **weights** 'uniform' and 'distance' are choosen for hyper parameter optimization. Best features are choosen using ANOVA statistics with k values 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125 for optimization.

**Evaluation:**



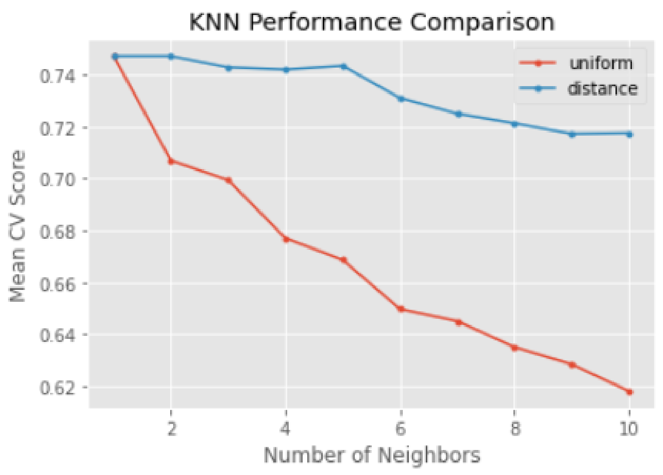


**Fig8: Max CV scores for different K –neighbors values for Manhattan /Euclidean distance and Uniform/distance weight**

From **Fig8** Cross validation scores are better with Manhattan distance compared to Euclidean and scores are better with 'distance ' weight compared to 'uniform' weight though the best score is generated with 'uniform weight'.

**Classification Report:**

```
             precision    recall  f1-score   support
           0       0.67      0.72      0.70       347
           1       0.68      0.60      0.64       325
           2       0.77      0.79      0.78       494

    accuracy                           0.72      1166
   macro avg       0.71      0.70      0.70      1166
weighted avg       0.71      0.72      0.71      1166
```

**Confusion Matrix:**

```
     0    1    2
0  250   48   49
1   59  196   70
2   62   44  388
```

**Learning Curve:**

For the learing curve in **Fig9** high variance can be seen though the acuuracy of CV-score increases with more training samples. May be more data(smaples) could give better results .

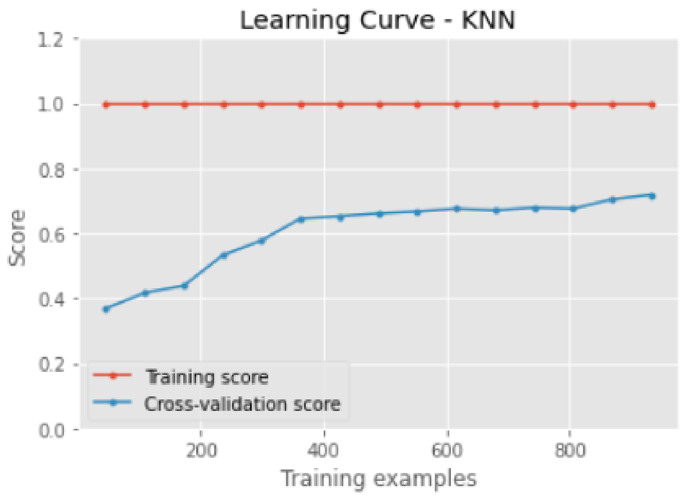


**Fig9: Learning Curve for KNN**

**The best Accuracy is achieved is 74.7% for k neighbour value 1, distance metric 'Manhattan' , 'Uniform' weight and for 75 selected best features .**

### 3.4 Decision Tree Evaluation:

**Hyper Parameters:**

For execution of Decision Tree **loss function** 'Gini' and 'Entropy'; **max depth** of the tree 10,20,30,40,50; **minimim sample split** 2, 3, 4 are choosen for hyper parameter optimization.
Best features are choosen using ANOVA statistics with k values 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125 for optimization.

**Evaluation:**

**Fig10** below shows that for max depth of tree above 20 the accuracy is similar and better accuracy is acieved with 'entropy' function compared to 'gini'.

# Classifiers Comparison

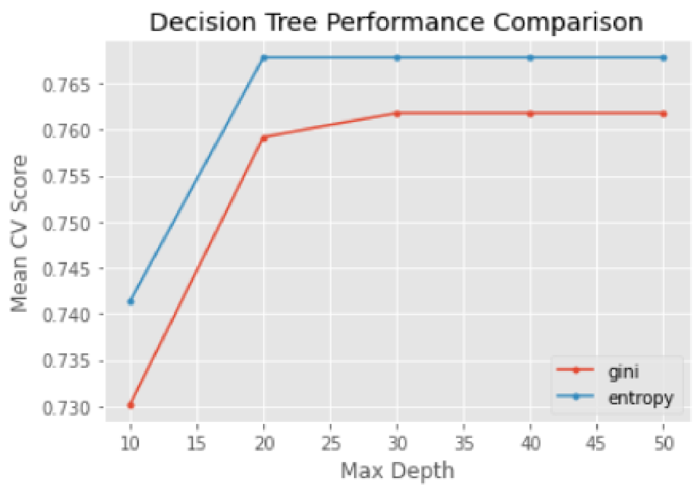


**Fig10: Max Cross valiadtion scores for tree depth values for loss function ‘gini’ and ‘entropy’**

**Classification report:**

```
              precision    recall  f1-score   support
           0       0.75      0.77      0.76       347
           1       0.69      0.68      0.69       325
           2       0.81      0.80      0.80       494

    accuracy                           0.76      1166
   macro avg       0.75      0.75      0.75      1166
weighted avg       0.76      0.76      0.76      1166
```

**Confusion Matrix:**

```
     0    1    2
0  268   48   31
1   41  222   62
2   48   53  393
```

**Learning Curve:**

For the learing curve in **Fig11** high variance can be seen though the acuuracy of CV-score increases with more training samples. May be more data(smaples) could give better results .

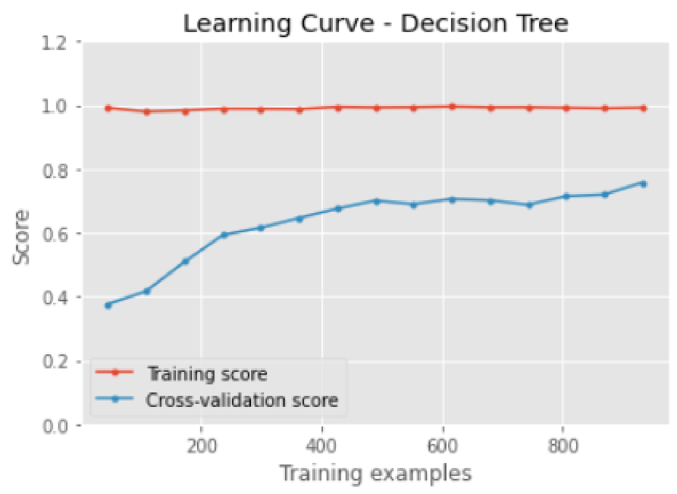


**Fig11: Learing curve for Decision Tree**

**The best Accuracy is achieved is 76.8% for loss function ‘Entropy’, Max depth 20, minimum sample split 4 and for 105 selected best features .**

## 3.5 Bagging Evaluation:

**Hyper Parameters:**

For execution of Bagging **n estimators**( number of trees) 10, 50, 100, 150, 200, 250 are choosen for hyper parameter optimization. Best features are choosen using ANOVA statistics with k values 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125 for optimization.

**Evaluation:**

From **Fig12** it can be seen that for more than 100 number of trees the per formance is similar and the best CV-score achieved for n estimatore value 200.

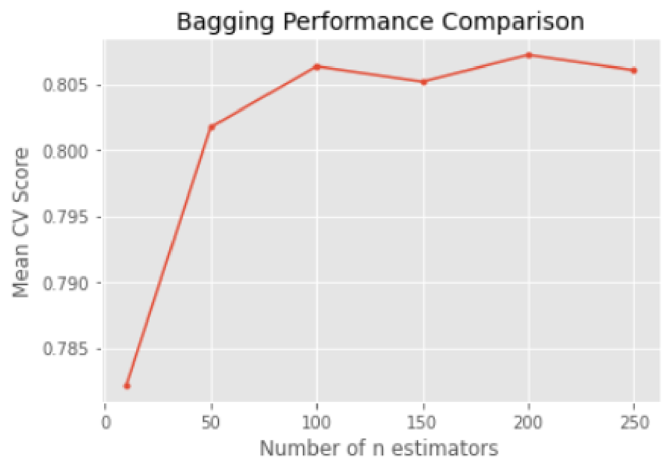


**Fig12: Max Cross validation score for different number of trees**

**Classification Report:**

```
              precision    recall  f1-score   support
           0       0.78      0.81      0.80       347
           1       0.78      0.71      0.74       325
           2       0.85      0.87      0.86       494

    accuracy                           0.81      1166
   macro avg       0.80      0.80      0.80      1166
weighted avg       0.81      0.81      0.81      1166
```

**Confusion Matrix:**

```
     0    1    2
0  282   35   30
1   48  230   47
2   32   31  431
```

**Learning Curve:**

For the learing curve in **Fig13** high variance can be seen though the acuuracy of CV-score increases with more training samples. May be more data(smaples) could give better results .

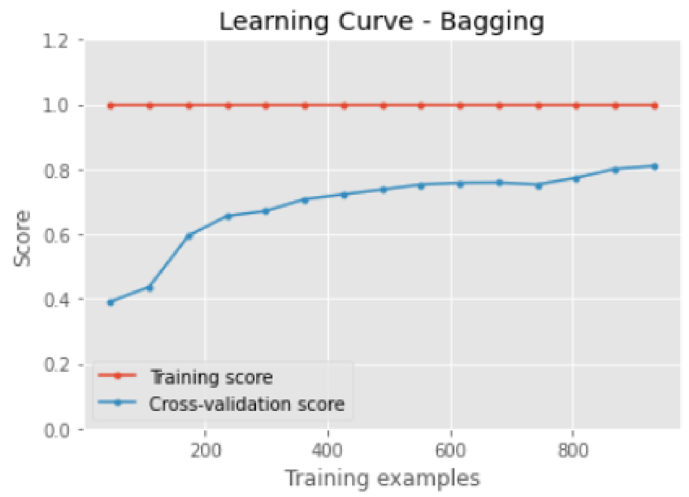


**Fig13: Learing curve for Bagging**

**The best Accuracy is achieved is 80.7% for 200 trees and for 120 selected best features .**

## 3.6 Random Forest Evaluation:

**Hyper Parameters:**

For execution of Random forest **n estimators**( number of trees) 100, 200; **loss function** ‘Gini’ and ‘Entropy’; **max depth** of the tree 20, 30, 40, 50; **minimum sample split** 2,3 are choosen for hyper parameter optimization. Best features are choosen using ANOVA statistics with k values 60, 70, 80, 90, 100, 110, 120 for optimization.

**Evaluation:**

**Fig14** below shows that for max depth of tree above 30 the accuracy is similar and better accuracy is acieved with ‘entropy’ function compared to ‘gini’.

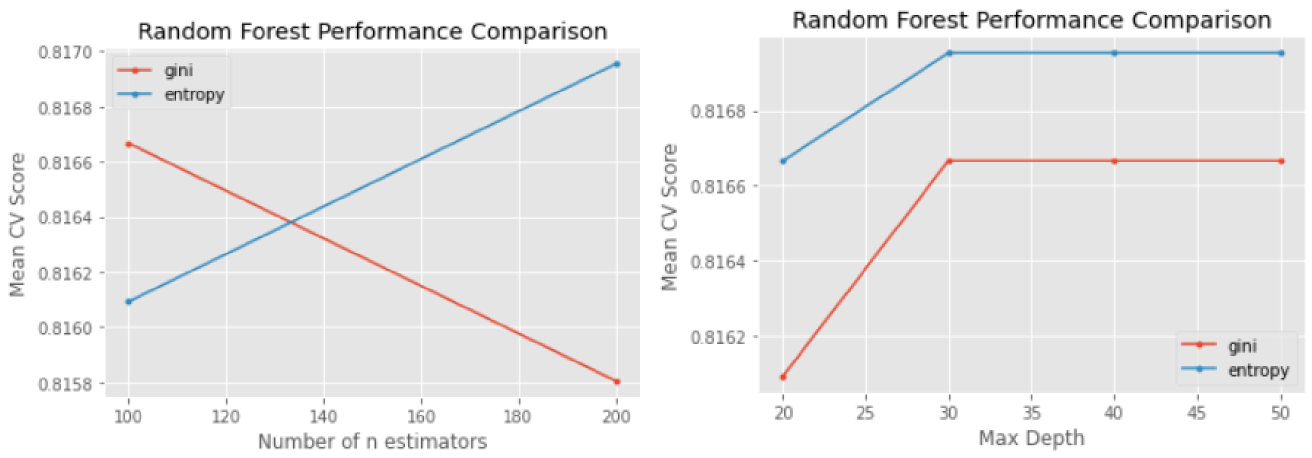


**Fig:14 Max CV score for number of trees(n estimatore) and max tree depth with Gini and Entropy function**

# Classifiers Comparison

**Classification Report:**

```
              precision    recall  f1-score   support
           0       0.79      0.80      0.79       347
           1       0.77      0.71      0.74       325
           2       0.85      0.88      0.87       494

    accuracy                           0.81      1166
   macro avg       0.80      0.80      0.80      1166
weighted avg       0.81      0.81      0.81      1166
```

**Confusion Matrix:**

```
     0    1    2
0  278   40   29
1   46  230   49
2   30   28  436
```

**Learing Curve:**

For the learing curve in **Fig15** high variance can be seen though the acuuracy of CV-score increases with more training samples. May be more data(smaples) could give better results .

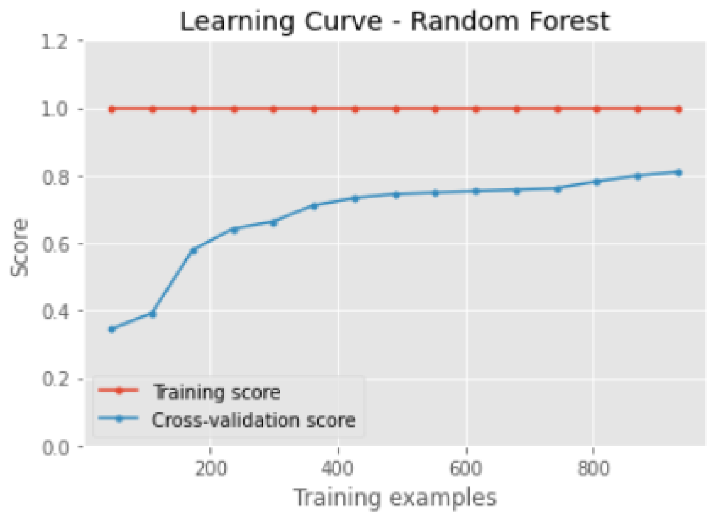


**Fig15: Learning curve for Random forest**

**The best Accuracy is achieved is 81.7% for 200 trees, loss function 'Entropy', Max depth 30, minimum sample split 2 and for 100 selected best features .**

# Classifiers Comparison

## 3.7 Support Vector Machine Evaluation:

**Hyper Parameters:**

For execution of SVM **kernel** RBF, Polynomial, Sigmoid; **Penalty parameter**(C) 0.01, 0.1, 1, 10, 50, 100; **gamma** 'auto', 'scale' are choosen for hyper parameter optimization. Best features are choosen using ANOVA statistics with k values 60, 70, 80, 90, 100, 110, 120 for optimization.

**Evaluation:**

In Fig16 it can be observed RBF kernel has better results compared to th other kernals used. With increasing penalty value kernals except Sigmoid produces better results.

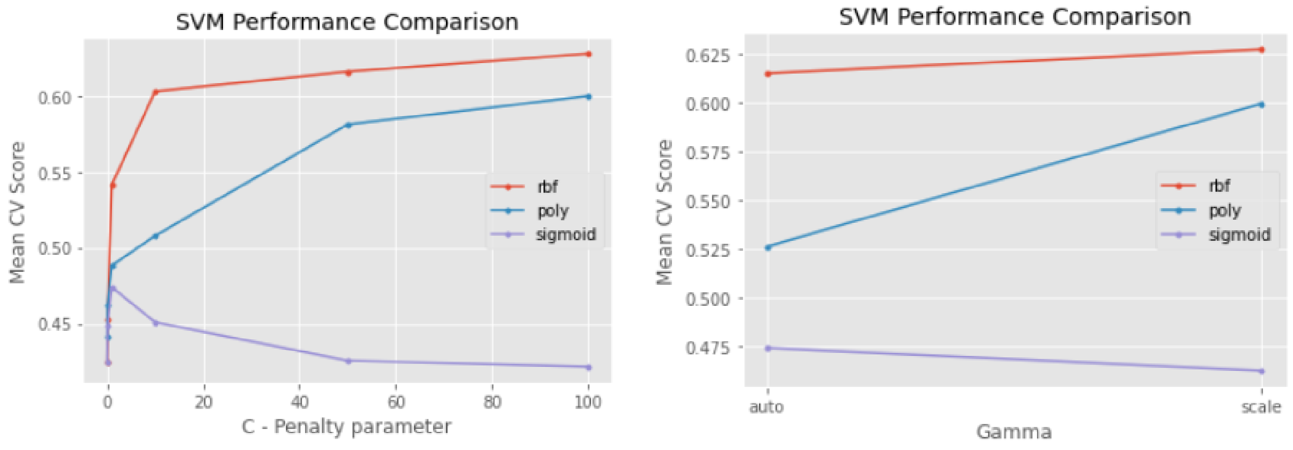


**Fig16: Max CV score for Penalty parameter and Gamma with kernal RBF, Polynomial and Sigmoid**

**Classification Report:**

```
              precision    recall  f1-score   support
           0       0.53      0.59      0.56       347
           1       0.55      0.33      0.41       325
           2       0.67      0.79      0.72       494

    accuracy                           0.60      1166
   macro avg       0.58      0.57      0.56      1166
weighted avg       0.59      0.60      0.59      1166
```

**Confusion Matrix:**

```
     0    1    2
0  205   44   98
1  121  106   98
2   61   43  390
```

**Learning Curve:**

From Fig17 it is noticed training and cross-validation accuracy is less and it doesn't seem to improve with more training examples.it suggests that model is overfitting. Maybe changing the hyper parameters will produce better results or .a different model needed.

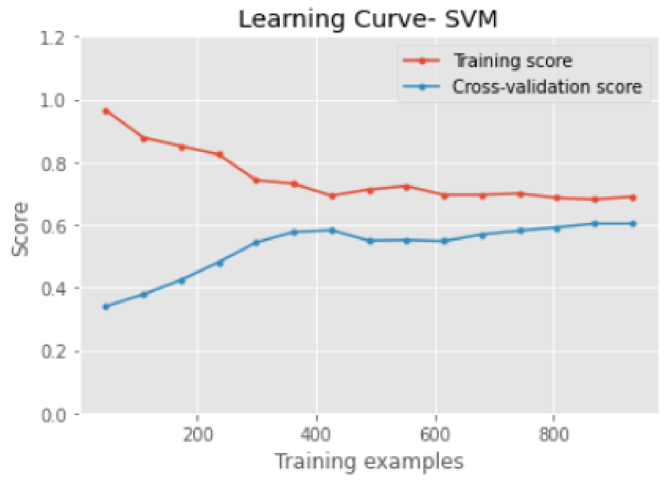


**Fig17: Learing curve for SVM**

**The best Accuracy is achieved is 62.8% for kernel RBF, penalty(c) 100, gamma scale and for 120 selected best features .**

## 4. Summary:

| Models | **Naive Bayes** | **Ridge Classifier** | **KNN** | **Decision Tree** | **Bagging** | **Random Forest** | **SVM** |
|---|---|---|---|---|---|---|---|
| Accuracy (%) | **57.7** | **64.9** | **74.7** | **76.8** | **80.7** | **81.7** | **62.8** |

**Table2: Highest CV- accuracy for different models created.**

Random forest has produced the highest accuracy (81.7% - from **Table2**) among all the models are used for this classification. More Hyper parameters can be used for tuning but that definitely has computational cost as algorithms takes time with increased hyper parameters. Other algorithms can also be tried to increase accuracy or maybe with more sample a better result can be produced.

## References:

1) Jason Brownlee, November 27, 2019. How to Choose a Feature Selection Method For Machine Learning. https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/
2) Carl Dawson, Sep 6, 2019. SVM Parameter Tuning. https://towardsdatascience.com/a-guide-to-svm-parameter-tuning-8bfe6b8a452c
3) Mahbubul Alam, Sep26, 2020. Anomaly detection with Local Outlier Factor (LOF) https://towardsdatascience.com/anomaly-detection-with-local-outlier-factor-lof-d91e41df10f2
4) Jason Brownlee, July 8, 2020. 4 Automatic Outlier Detection Algorithms in Python https://machinelearningmastery.com/model-based-outlier-detection-and-removal-in-python/
5) https://www.dataquest.io/blog/learning-curves-machine-learning/
6) **https://www.datatechnotes.com/2020/04/anomaly-detection-with-local-outlier-factor-in-python.html**
7) **https://www.featureranking.com/tutorials/machine-learning-tutorials/sk-part-3-cross-validation-and-hyperparameter-tuning/**
8) **https://www.datatechnotes.com/2020/04/anomaly-detection-with-local-outlier-factor-in-python.html**