Supervised Learning Assignment

Nishant Jha
Njha30@gatech.edu

Abstract

The paper is intended to solve two classification problems using five supervised learning algorithms with intent of not only solving the problem but to analyze performance, complexity, efficiency as well as error rate of these algorithms being used. The first problem addressed is a generic problem to predict phishing website using online available dataset with 31 variables including URL length, prefix-suffix, port, redirect, etc. The second problem is to determine the wine category based on various defining factors like alcohol content, flavonoids etc. The analysis is done by training, validating, running test and finally plotting various curves for defined 5 algorithms.

1. Introduction to Problems and Datasets

1.1. The Phishing Website Dataset

Phishing has been one of the key hacking techniques since the beginning and has been into limelight even in current scenario. Identifying and disregarding a phishing website link is the only way to escape is hacking attempts. In past research [1-3], scientists have been able to come up with set of attributes that can be used effectively to identify a phishing website link.

The dataset used in the experiment consists of a few attributed derived from such research in order to prove it sound and effective in predicting phishing websites. There are 31 variables used to identify the result for around 11050 entries. The variables are mostly categorical and binary indicating whether a website has a particular attribute or not. For the non-binary attributes, dataset is one hot encoded for converting these attributes to binary. Some of these attributes includes having IP Address, URL Length, SSL final state, port, HTTPS status, links in tags etc.

1.2. The Wine Dataset

This dataset is defined over chemical compositions that are attributed to a particular wine belonging to one the 3 wine cultivators. Its again a classification dataset used to classify wine into these 3 cultivators class based on chemical properties like Alcohol, Malic acid, Alkalinity of ash, Magnesium, Flavonoids, etc. All in all there are 13 variables used for classifying 5000 entries into multi-class classification (3 classes). The data set is balanced, and no class dominates any other.

The major difference in the two problems is that the phishing website dataset classification is done over binary classes i.e., whether the URL is corrupt or not whereas the wine dataset is classified in 3 classes of 3 cultivators making it a multi-class classification problem. For the same reason, in order to make comparison better and on the same scale, the multiclass dataset is validated using with "f1-weighted" scoring with whereas binary phishing dataset is validated with "f1" scoring. Also, to calculate "f1" score for wine dataset, weighted average is used which is calculated based on data count.

Other differences in the dataset includes phishing data columns being categorical type where as wine data columns being numerical. There is also feature difference where phishing dataset is evaluated using 31 features, wine dataset has 13 features. Of all these differences however, both the data sets are balanced and there is no need to use and balancing technique.

2. Pre-Processing Datasets

In order to consume both the data generically, as said earlier, the categorical nonbinary field of phishing dataset is converted into binary field using 1-hot encoding with help of panda library get_dummies [4] method.

In case of wine dataset, since values were numerical with different floating-point values, normalization is done in order to make prediction easier and more efficient with improved accuracy.

Upon performing the corresponding data correcting tasks, the datasets are then divided into two stratified splits, where 80% of them are to be used for training/validation and 20% of them are kept for final testing. Stratified splits are used to keep ratio of targets and training dataset similar in order to avoid bias.

3. Experiments – Results and Analysis

Experiments are done by running 5 machine learning algorithms namely, Decision Tree, Neural Networks, Boosted Decision tree, SVM and K-NN. The results obtained post training are plotted based on data complexity (no. of samples) against prediction score, prediction time and error rate. Finally, comparison is of these algorithm performances is also done through different plots. All the results and analysis can be found in the section below:

3.1 Decision Tree

Initially a decision tree classifier is choses with criterion set to entropy, min sample leaf as 1 and iterated over max depth ranged between 1 to 25. The complexity curve is then plotted against depth range to check how classifier behaves on different depth.

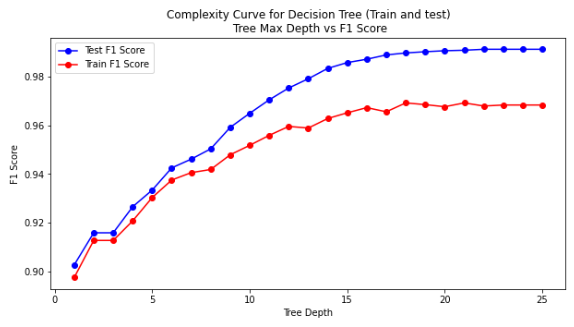
Post this GridSearchCV is used to determine best parameters (hyper tuning) using following param grid:

```
param_grid = {
    'criterion': ['gini', 'entropy'],
    'min_samples_leaf': range (5% to 
50% of sample size)
    'max_depth': range (1,20)
}
```

Using the parameters achieved post hyper tuning, a new classifier is created, and cross validation is done with training sample split over 20 folds with size between 5% to 100% of data. These validation interactions are plotted against various analysis metrics like prediction time, complexity curve and training and testing error. The same tuned classifier is then used for testing and final evaluation is done.

3.1.1 Phishing Dataset

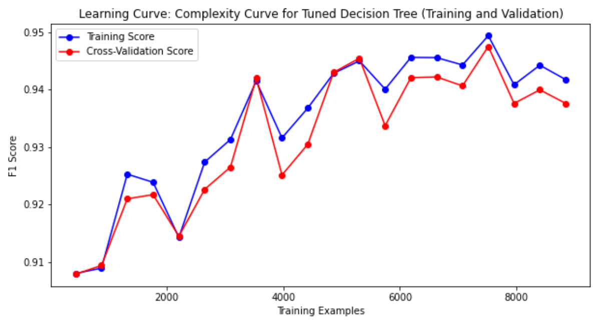
The data seems to perform well with low depth as f1 score seems to stabilize after depth 20 with test score higher than training score.



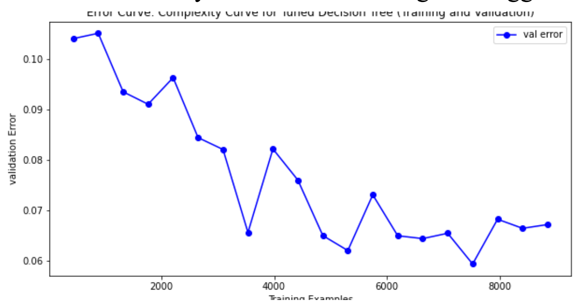
Post Hyper tuning, the parameters comes out to be as following:

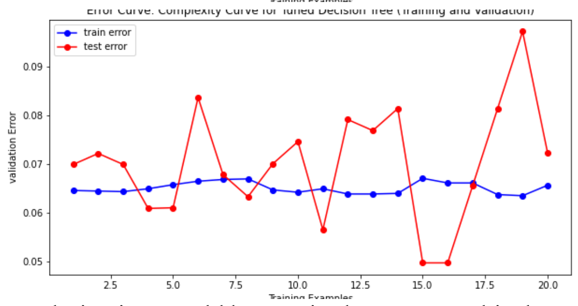
{'criterion':'entropy',
'max_depth': 11,
'min_samples_leaf': 44}

The complexity curve (f1 score vs training size) plotted for tuned decision tree is shown below:



The curve seems to fit quite well for some numbers of example. However bias and variance seems to increase after 5000 training set, indicating model fitting well with lower number of examples. In the error curve below as well, it is clear how error rate goes down till 5000 examples and destabilizes post that. All of this observation clearly indicates overfitting and suggesting

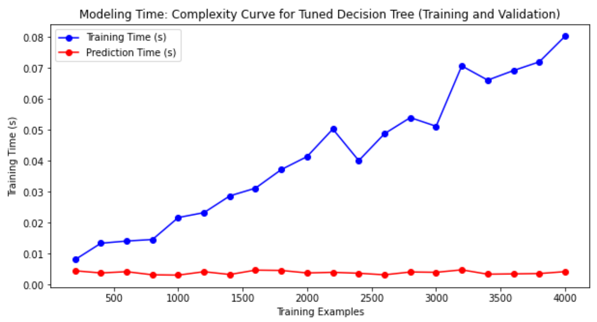




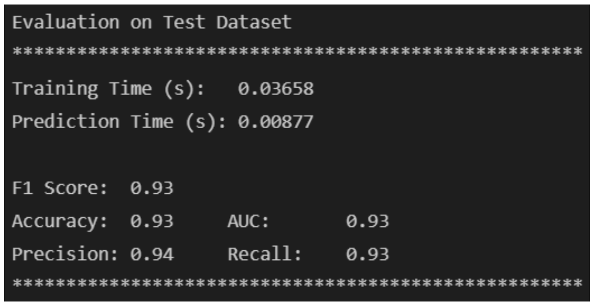
regularization would be required. However, this data set shows less variance and bias compared to wine dataset.

Finally Modeling time curve is plotted as shown below. The training time seems to increase with increasing num. of example whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.

.



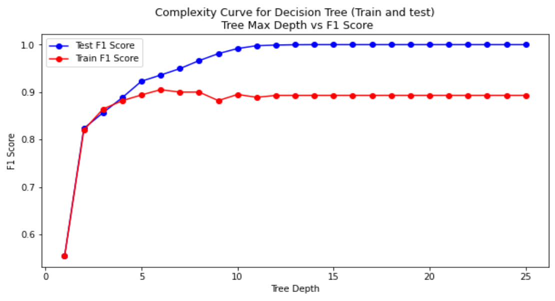
Overall the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.93.



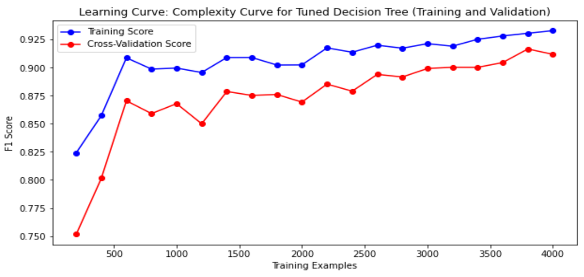
```
Evaluation on Test Dataset
*****************************************************

Training Time (s):   0.03658
Prediction Time (s): 0.00877

F1 Score:  0.93
Accuracy:  0.93     AUC:       0.93
Precision: 0.94     Recall:    0.93
*****************************************************
```

3.1.2 Wine Dataset

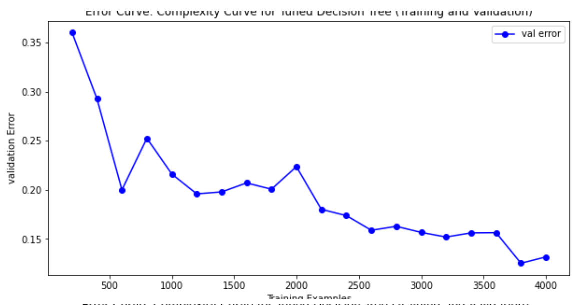
The data seems to perform well with low depth as f1 score seems to stabilize after depth 12 with test score higher than training score.

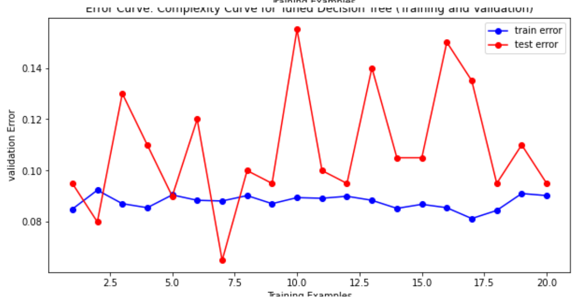


Post Hyper tuning, the parameters come out to be as following:
{'criterion':'entropy', 'max_depth': 7,
'min_samples_leaf': 20}



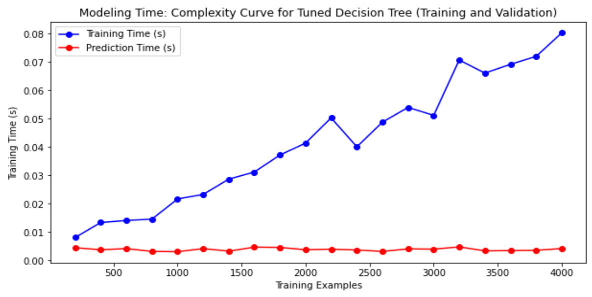
The complexity curve (f1 score vs training size) plotted for tuned decision tree is shown above. The accuracy seems to be low with lower number of training examples, indicating high bias. Also, variance too is greater in case of small number of samples as the score difference between training and validation is high. However, as the number of examples increases, the model seems to stabilize, score tends to improve also bias and variance improves.



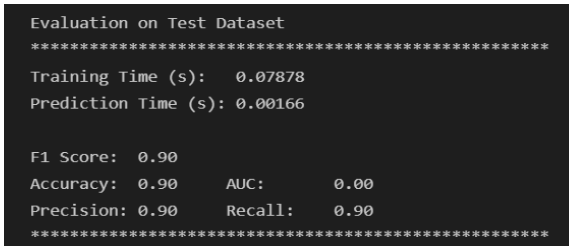


The above curve shows validation, training and testing error respectively. Overall validation error seems more than training error. Even in case of training and testing error, testing error seems greater than training error but it doesn't have a decreasing curve. Both observations suggest that model is overfitting in case of lower training example. However, the three curve seems to converge in case of higher num. of examples suggesting that model would behave better with greater number of training examples.

Finally Modeling time curve is plotted as shown below. The training time seems to increase with increasing num. of example whereas prediction time remains constant indication model efficiency remains intact with increasing training complexity.



Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.90.

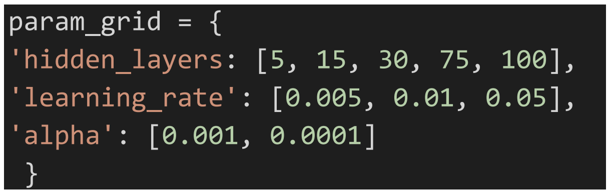


```
Evaluation on Test Dataset
*****************************************************

Training Time (s):   0.07878
Prediction Time (s): 0.00166


F1 Score:  0.90
Accuracy:  0.90     AUC:       0.00
Precision: 0.90     Recall:    0.90
*****************************************************
```

3.2 Neural Networks

Initially a Neural Network classifier is choses with solver set to sgd, activation as relu and iterated over hidden layer ranged between 1 to 150. The complexity curve is then plotted against hidden layer range to check how classifier behaves on different layers.

Post this GridSearchCV is used to determine best parameters (hyper tuning) using following param grid:

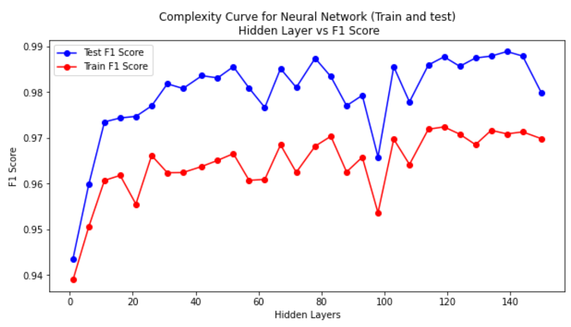


```
param_grid = {
'hidden_layers: [5, 15, 30, 75, 100],
'learning_rate': [0.005, 0.01, 0.05],
'alpha': [0.001, 0.0001]
 }
```

Using the parameters achieved post hyper tuning and using same solver and activation, a new classifier is created, and cross validation is done with training sample split over 20 folds with size between 5% to 100% of data. These validation and evaluation are done like as done in Decision tree experiment.

3.2.1 Phishing Dataset

The data seems to have better f1 score compared to decision tree however, the variance seems to be high though seems converging as hidden layer increases, however, not good enough to reduce variance significantly.

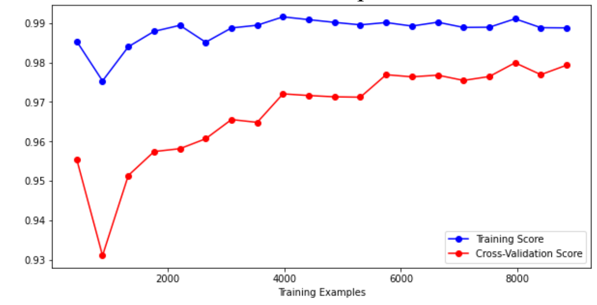


Post Hyper tuning, the parameters come out to be as following:

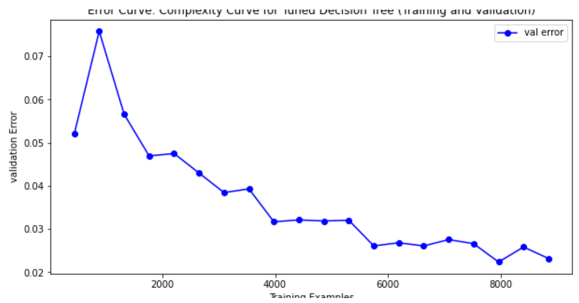
{'alpha':0.001, 'hidden_layer_sizes': 75, 'learning_rate_init': 0.05}

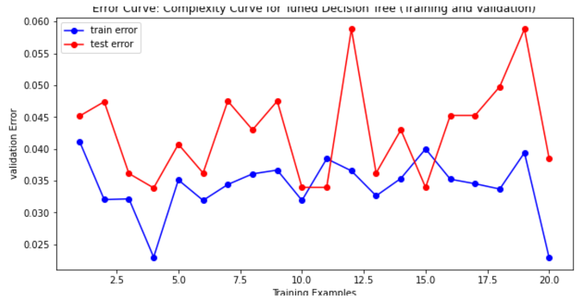
The complexity curve (f1 score vs training size) plotted for tuned neural network is shown below:

Bias and variance seem to improve with increase in

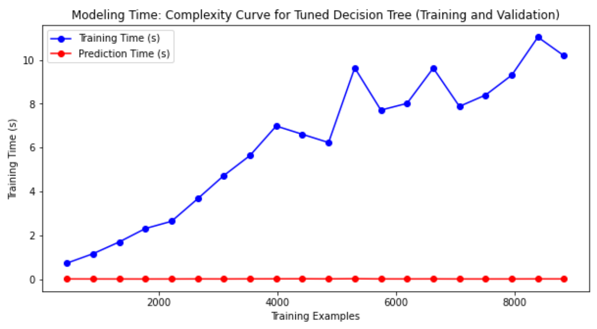


training examples. However, this time, the curve seems to converge quite well compared to phishing dataset reducing bias significantly as number of training examples increase. However, variance remains almost constant post 1000 examples. Even the F1 score seems to be quite high and stabilizing which indicates good performance accuracy.





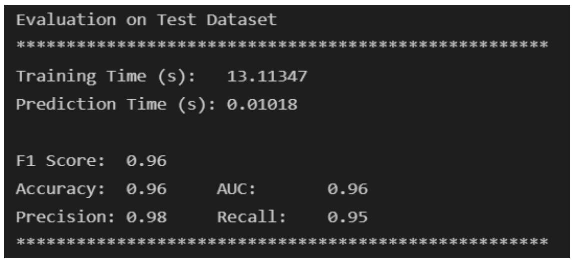
The above curves show validation, training and testing error respectively. Overall validation error seems less than training error. Even in case of training and testing error, testing error seems greater than training error but it doesn’t have a decreasing curve. However, the overall error rate is much lower compared to decision tree model, the stability seems missing in all the curve and shows regularization is needed.



Finally Modeling time curve is plotted as shown above. The training time seems to increase with increasing num. of example whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.

.

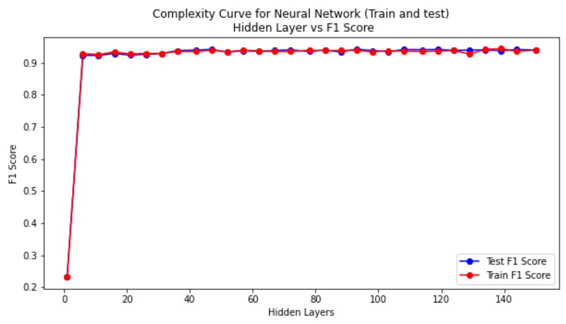
Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy 0.96 and precision coming as 0.98.



```
Evaluation on Test Dataset
******************************************************

Training Time (s):   13.11347
Prediction Time (s): 0.01018

F1 Score:  0.96
Accuracy:  0.96     AUC:       0.96
Precision: 0.98     Recall:    0.95
******************************************************
```

3.2.2 Wine Dataset

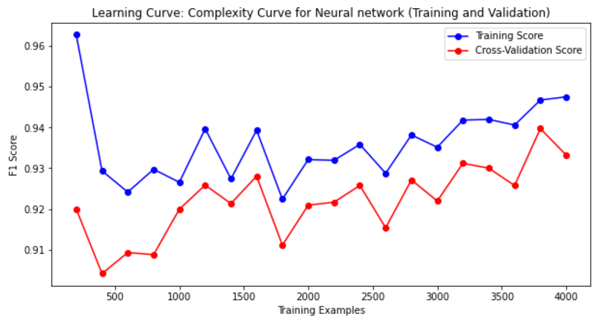
The data seems to have lower f1 score compared to phishing dataset. Also, the curve seems to be quite overlapping throughout the layers indicating clear overfitting.



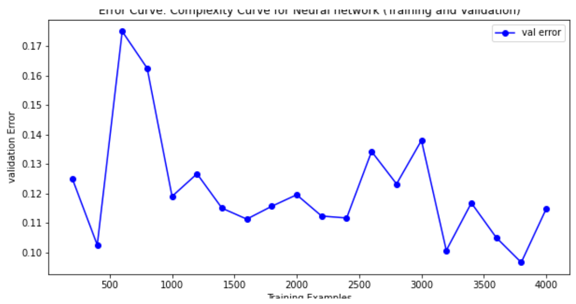
Post Hyper tuning, the parameters come out to be as following:

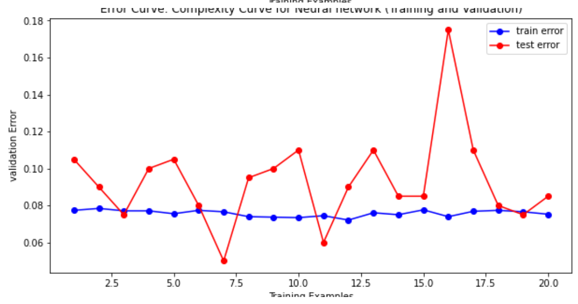
{'alpha':0.0001, 'hidden_layer_sizes': 100, 'learning_rate_init': 0.05}

The complexity curve (f1 score vs training size) plotted for tuned neural network is shown below:



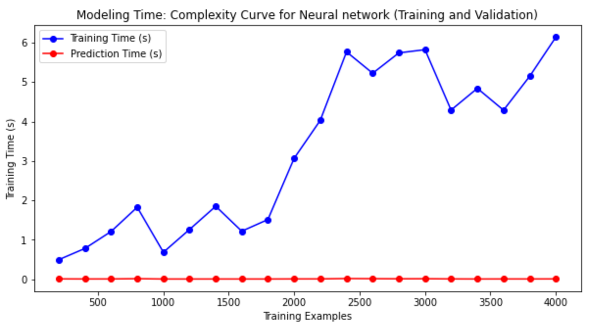
As seen before the bias and variance seems to be quite high even in case of validation data. However, this time, the curve seems to converge quite well reducing bias and variance significantly as number of training examples increase but later seems to diverge again. F1 score seems to be lower compared to phishing dataset but similar if not better than decision tree model. Overall underfitting is observed in the dataset and optimization seems to be required.





The above curves show validation, training and testing error respectively. Overall validation error is greater than training error. Even in case of training and testing error, testing error seems greater than training error but it doesn't have a decreasing curve. Also, error rate is greater compared phishing dataset, the stability seems missing in the curve and shows both generalization and regularization is required.

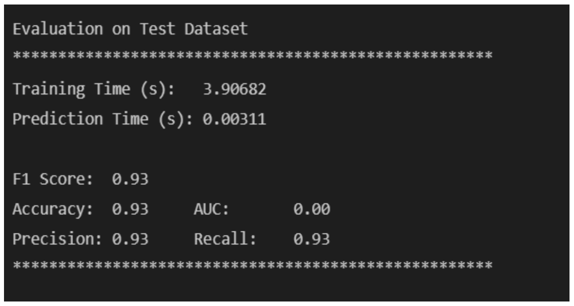
Finally Modeling time curve is plotted as shown below:



The training time seems to increase with increasing num. of example whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.

.

Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.93. The score seems lower compared to phishing data but similar to decision tree model.



```
Evaluation on Test Dataset
*******************************************************
Training Time (s):   3.90682
Prediction Time (s): 0.00311

F1 Score:  0.93
Accuracy:  0.93     AUC:       0.00
Precision: 0.93     Recall:    0.93
*******************************************************
```

3.3 Boosted Decision Tree

GradientBoostingClassifier is choses for this problem and is initially tested with min sample leaf 50, max depth 5 and iterated over n-estimator ranged between 1 to 250. The complexity curve is then plotted.

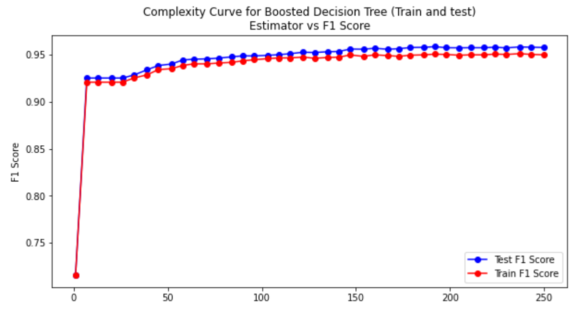
Post this GridSearchCV is used to determine best parameters (hyper tuning) using following param grid:

```
param_grid = {
'min_samples_leaf': range (5% to 50% of
sample size)
'max_depth': np.arange(1,4),
'n_estimators': range(10, 100)
'learning_rate': range (0.001, 0.1)
```

Using the parameters achieved post hyper tuning and new classifier is created, and cross validation is done in similar way as done in previous model.

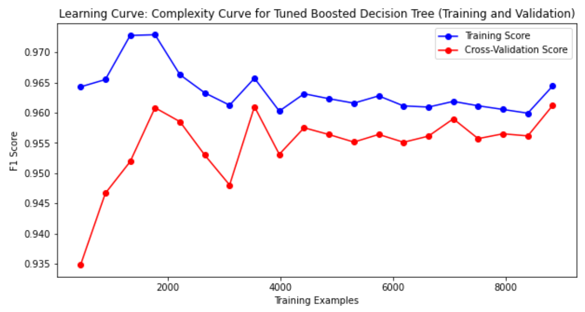
3.3.1 Phishing Dataset

The data seems to fit good with better f1 score and curve seem to be slightly overlapping. However as num. of estimator increases, the curve seem to slightly diverge although not very significantly.

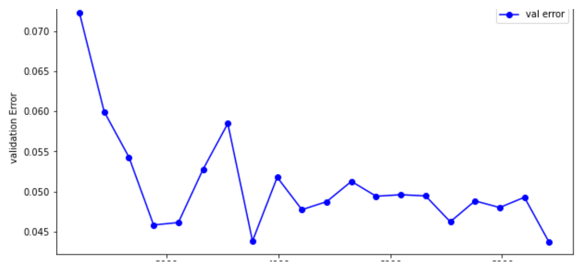


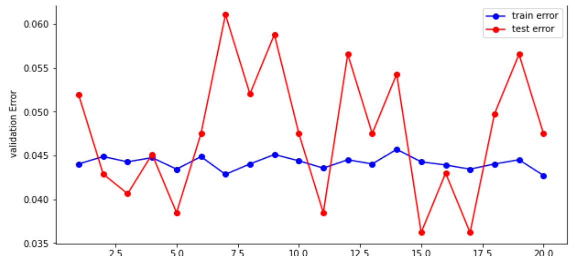
Post Hyper tuning, the parameters come out to be as following:

{'learning_rate': 0.1, 'max_depth': 3, 'min_samples_leaf': 44, 'n_estimators': 100}

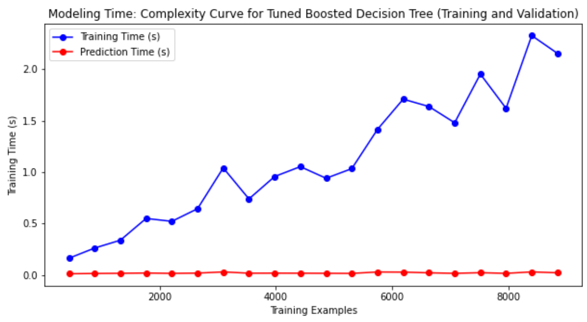


The complexity curve (f1 score vs training size) plotted for tuned neural network is shown above. Bias and variance seem to improve with increasing training examples. Even the F1 score seems to be quite high and stabilizing which indicates good performance accuracy.





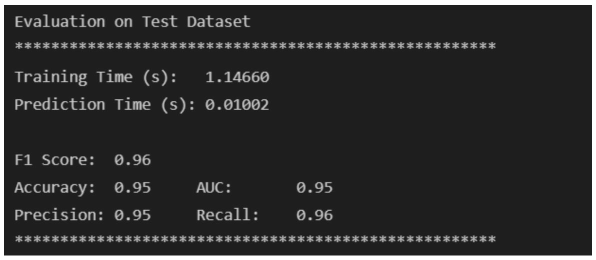
The above curves show validation, training and testing error respectively. Overall validation error is greater than training error but shows decreasing curve. Even in case of training and testing error, testing error seems greater than training error but it doesn't have a decreasing curve. This suggests some overfitting and requires regularization needed. However, error rate seems quite low compared to other models.



Finally Modeling time curve is plotted as shown above. The training time seems to increase with increasing num. of example whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.

.

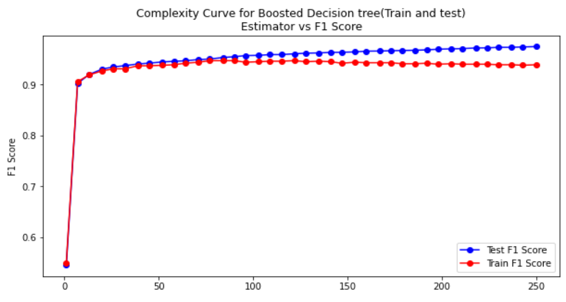
Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.95.



```
Evaluation on Test Dataset
*****************************************************

Training Time (s):   1.14660
Prediction Time (s): 0.01002

F1 Score:  0.96
Accuracy:  0.95     AUC:       0.95
Precision: 0.95     Recall:    0.96
*****************************************************
```

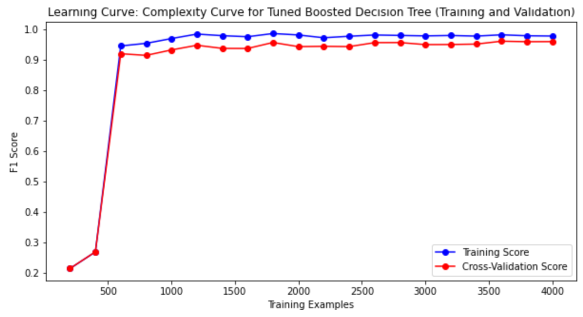
3.3.2 Wine Dataset

The data shows similar trend as that in phishing dataset. Just that the diversion seems to be greater in this case.



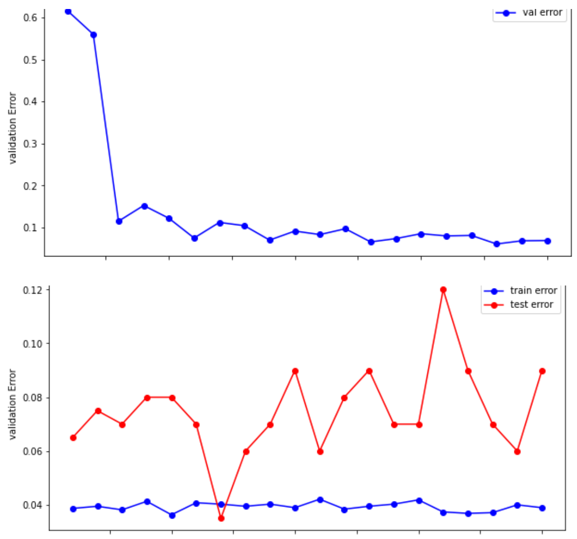
Post Hyper tuning, the parameters come out to be as following:

{'learning_rate': 0.1, 'max_depth': 3, 'min_samples_leaf': 200, 'n_estimators': 100}



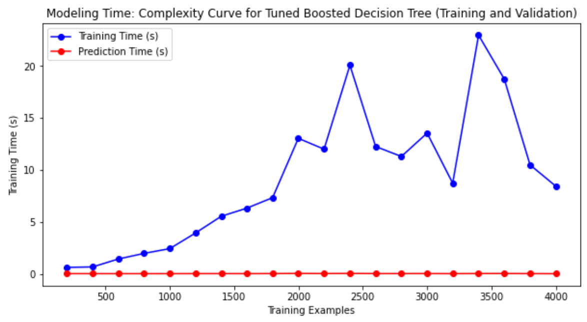
The complexity curve (f1 score vs training size) plotted for tuned boosted decision tree is shown above. The F1 score seems to be quite high and stabilizing which indicates good performance accuracy. Also, the curves seem to merge mostly indicating model fits the data very well.

The above curves show validation, training and testing

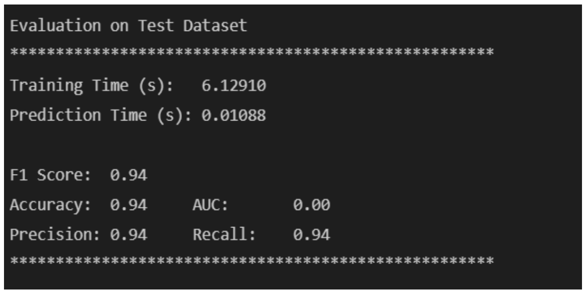


error respectively. Overall validation error is greater than training error but shows decreasing curve which stabilizes with increasing training example showing good fitting. Even in case of training and testing error, testing error seems greater than training error but it doesn't have a decreasing curve. Also, error rate seems big high than phishing data. This suggests some overfitting and requires regularization needed.

Finally Modeling time curve is plotted as shown Below. The training time seems to increase with increasing num. of example (however shows decreasing trend after 3500 examples), whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.



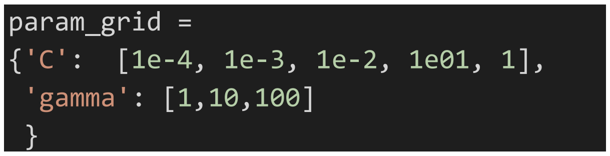
Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.94 which seems lesser that phishing data.



```
Evaluation on Test Dataset
*****************************************************
Training Time (s):   6.12910
Prediction Time (s): 0.01088

F1 Score:  0.94
Accuracy:  0.94     AUC:       0.00
Precision: 0.94     Recall:    0.94
*****************************************************
```

3.4 SVM

SVM is evaluated on two kernel functions ‘linear’ and ‘rbf’. GridSearchCV is used to determine best parameters (hyper tuning) using following param grid:



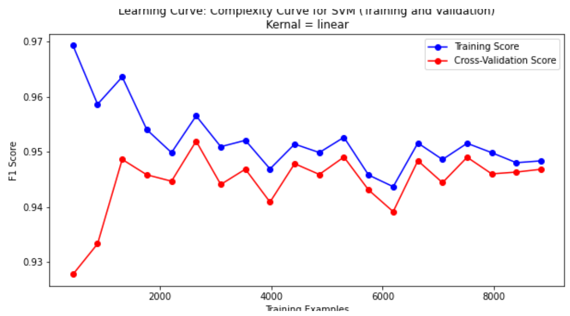
```
param_grid =
{'C':  [1e-4, 1e-3, 1e-2, 1e01, 1],
 'gamma': [1,10,100]
 }
```

Using the parameters achieved post hyper tuning and new classifier is created, and cross validation is done in similar way as done in previous model.

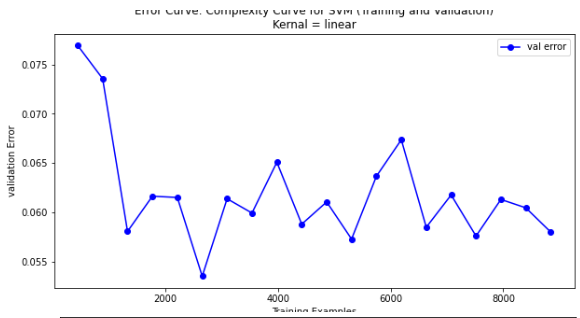
3.4.1 Phishing Dataset

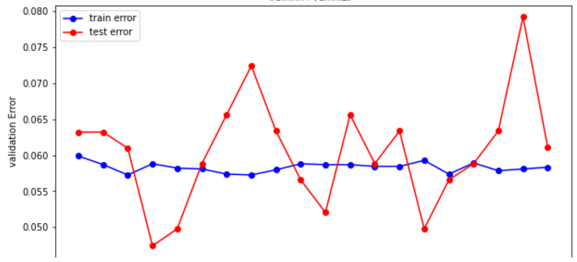
For kernel type linear, post Hyper tuning, the parameters come out to be as following:
{'C': 10.0, 'gamma': 1}

The complexity curve (f1 score vs training size) plotted for SVM with linear kernel is shown below. Bias and variance seem quite high initially however, it keeps improving with increasing training examples. Even the

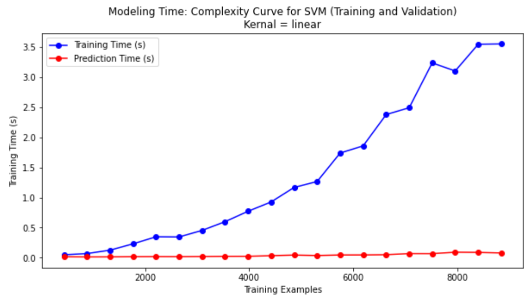


F1 score seems to stabilize around .95 which indicates good performance accuracy. The curves converse quite well with increasing example showing good fit.





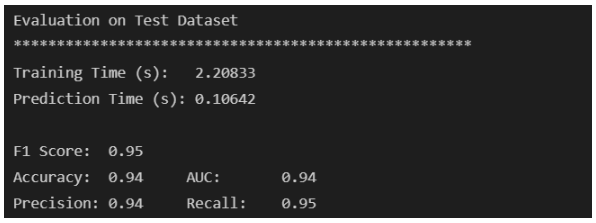
The above curves show validation, training and testing error respectively. Overall validation error is greater than training error but shows decreasing curve. Even in case of training and testing error, testing error seems greater than training error but it revolves around the training error line. This suggests some overfitting but also indicate model is performing well with more number of training examples. Error rate also seems quite low compared to other models.



Finally Modeling time curve is plotted as shown above. The training time seems to increase with increasing num. of example whereas prediction time remains almost constant with minimal variation indication model efficiency remains intact with increasing training complexity.

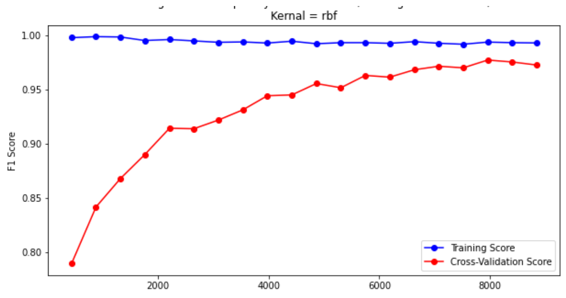
.

Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.94.

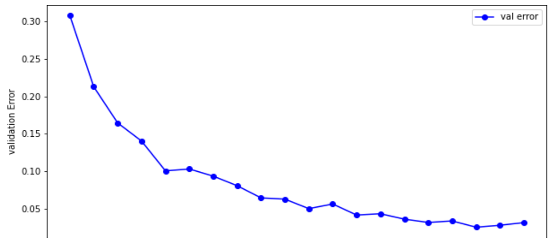


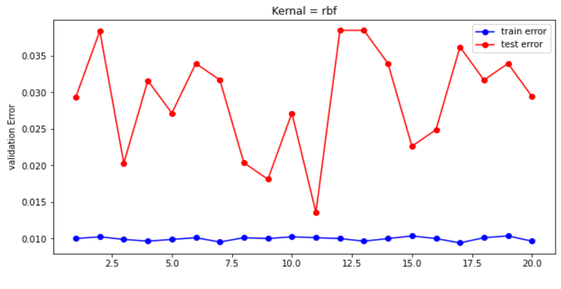
```
Evaluation on Test Dataset
*****************************************************
Training Time (s):   2.20833
Prediction Time (s): 0.10642

F1 Score:  0.95
Accuracy:  0.94     AUC:       0.94
Precision: 0.94     Recall:    0.95
```

For kernel type rbf, post Hyper tuning, the parameters come out to be as following:
{'C': 1, 'gamma': 1}

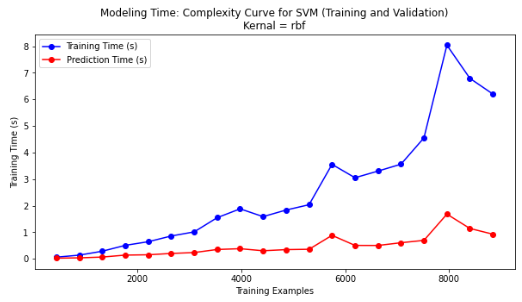


The complexity curve (f1 score vs training size) plotted for SVM with rbf kernel is shown above. Bias and variance seem quite high initially however, it keeps improving with increasing training examples. Even the F1 score seems to stabilize above .95 which indicates good performance accuracy. The curves converse quite well with increasing example showing good fit.





The above curves show validation, training and testing error respectively. Overall validation error is greater than training error but shows continuous decreasing curve which again is a good indicator. Also, it ends up at error rate lower that training error with

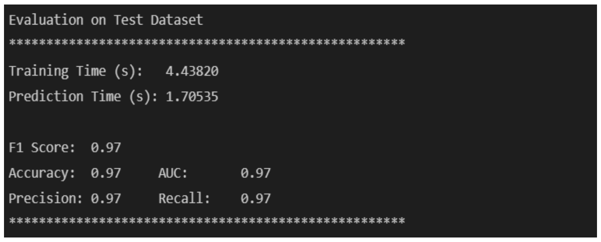


greater number of examples. Even in case of training and testing error, testing error seems greater than training error. Error rate also seems quite low compared to other models.

Finally Modeling time curve is plotted as shown above. The training time seems to increase with increasing num. of example and prediction time also increases but magnitude being quite less.

.

Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform very well with accuracy and precision coming as 0.97.

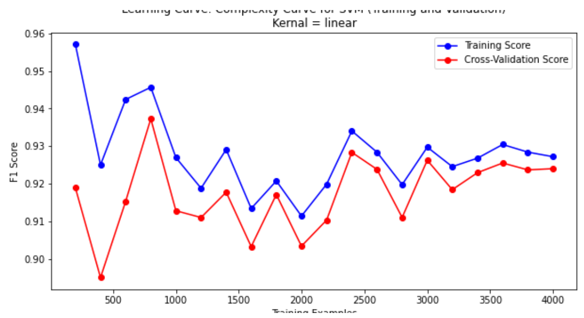


```
Evaluation on Test Dataset
*****************************************************
Training Time (s):   4.43820
Prediction Time (s): 1.70535

F1 Score:  0.97
Accuracy:  0.97     AUC:       0.97
Precision: 0.97     Recall:    0.97
*****************************************************
```

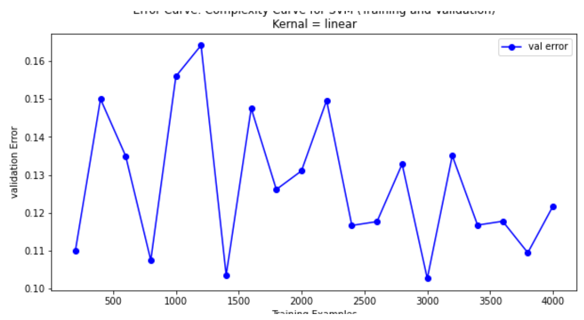
3.4.2 Wine Dataset

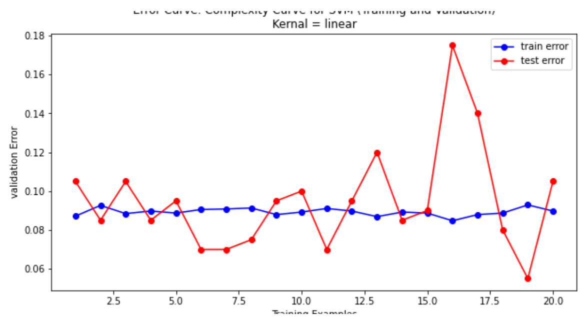
For kernel type linear, post Hyper tuning, the parameters come out to be as following:
{'C': 10.0, 'gamma': 1}

The complexity curve (f1 score vs training size) plotted for SVM with linear kernel is shown below. Bias and variance seem quite high initially and similar to phishing dataset, the curves do converges around 0.93 score. F1 score seems to be less compared to phishing dataset indicating more optimizations can be done to improve model performance.

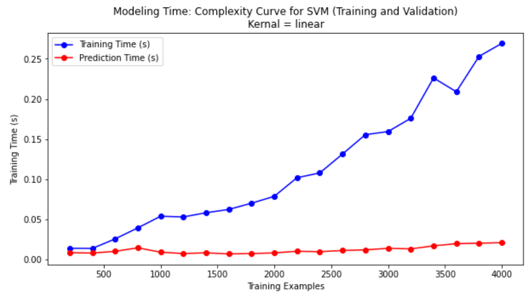


The below curves show validation, training and testing error respectively. Overall validation error is greater than training error and doesn't show decreasing curve in fact is fluctuating. Even in case of training and testing error, testing error seems greater than training error but it revolves around the training error line. This suggests some overfitting Error rate also seems a bit high compared to phishing model indicating overfitting.

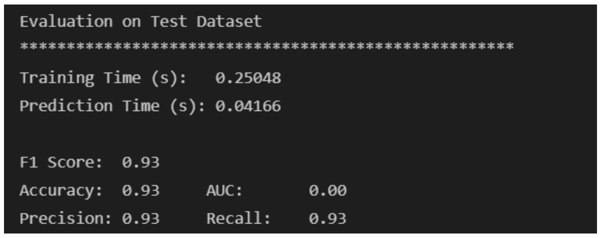




Finally Modeling time curve is plotted as shown below. The training time seems to increase with increasing num. of example but prediction time keeps stable.

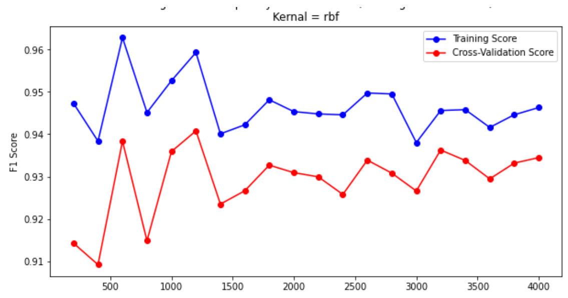


Overall, the final model is then tested over test set and following metric values are observed. Overall, model seems to perform well with accuracy and precision coming as 0.93 which is a bit lower compared to phishing dataset.

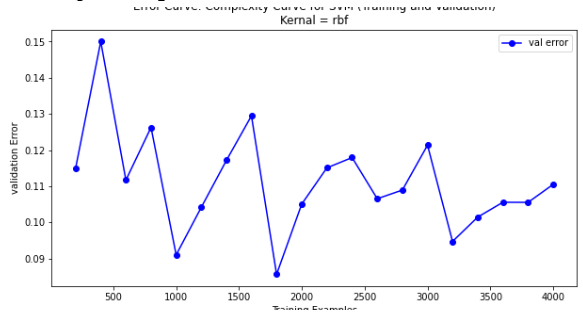


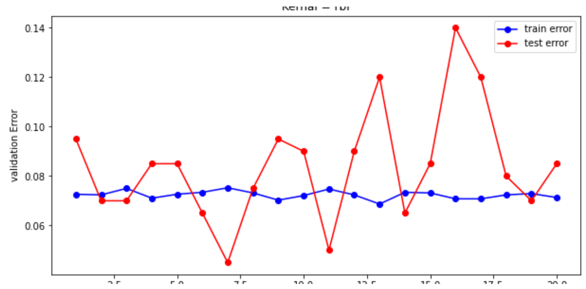
```
Evaluation on Test Dataset
*****************************************************

Training Time (s):   0.25048
Prediction Time (s): 0.04166

F1 Score:  0.93
Accuracy:  0.93     AUC:       0.00
Precision: 0.93     Recall:    0.93
```

For kernel type rbf, post Hyper tuning, the parameters come out to be as following:
{'C': 1, 'gamma': 1}

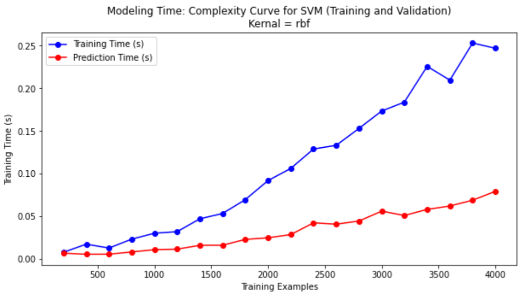


The complexity curve (f1 score vs training size) plotted for SVM with rbf kernel is shown above. Bias and variance seem quite high initially however, though it keeps improving with increasing training examples, the curves don't converge indicating underfitting. Even the F1 score seems to stabilize above .94 which indicates good performance accuracy but lesser than it was in the case of phishing dataset.



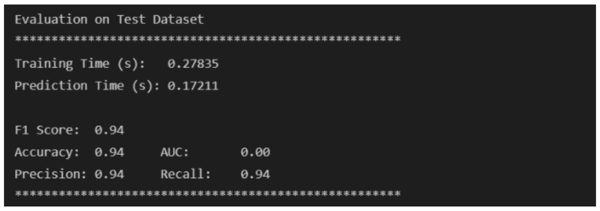


The above curves show validation, training and testing error respectively. Overall validation error is greater than training error and doesn't show sharp decreasing curve and is fluctuating. Even in case of training and testing error, testing error seems greater than training error but it revolves around the training error line. This suggests some overfitting. Error rate also seems a bit high compared to phishing model indicating overfitting.



Finally Modeling time curve is plotted as shown above. The training time seems to increase with increasing num. of example and so does prediction time which hasn't been the case with any models plotted earlier. This indicates poor efficiency.

Overall, the final model is then tested over test set and following metric values are observed. Model seems to perform well with accuracy and precision coming as 0.94 which is a lower compared to phishing dataset.


```
Evaluation on Test Dataset
*****************************************************
Training Time (s):   0.27835
Prediction Time (s): 0.17211

F1 Score:  0.94
Accuracy:  0.94     AUC:       0.00
Precision: 0.94     Recall:    0.94
*****************************************************
```

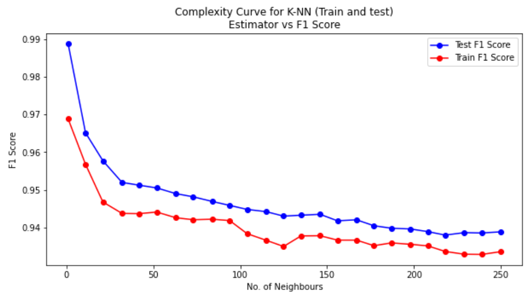
3.5 K-NN

Initially a K-NN classifier is to be iterated over neighbor ranged between 1 to 250. The complexity curve is then plotted against hidden layer range to check how classifier behaves on different layers.

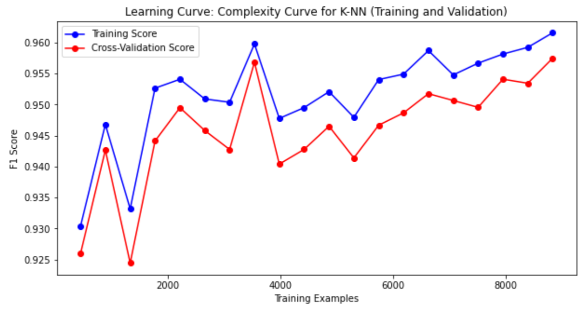
Post this the algorithm is ran with neighbor set to 20, and cross validation is done in similar way as done in previous model.

3.5.1 Phishing Dataset

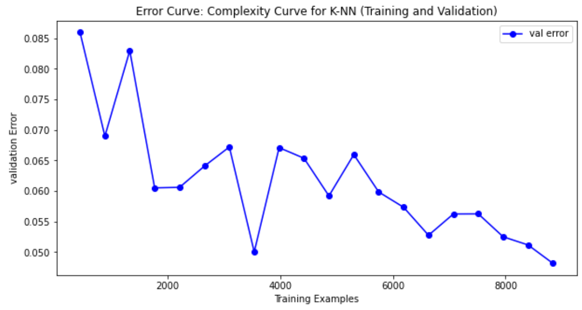
The data seems to not fit good with f1 score decreasing continuously for both training and validation with increasing neighbor as shown in plot below. Since the closest the plot comes is at 20. Neighbor number 20 is selected for further analysis.

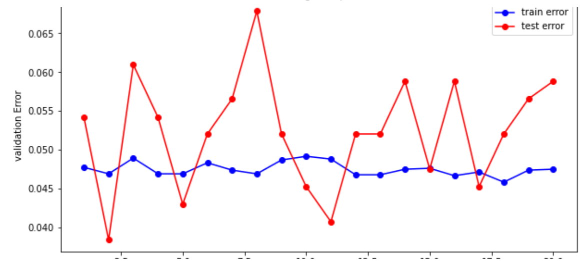


The complexity curve (f1 score vs training size) plotted for tuned decision tree is shown below:

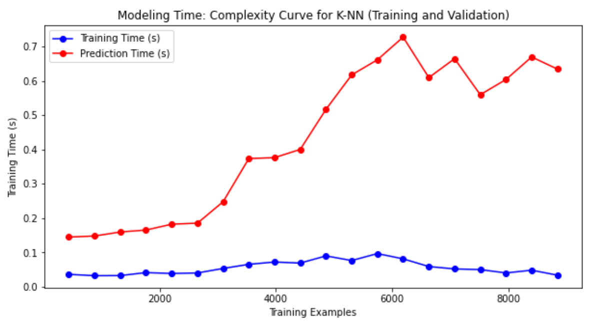


The complexity curve (f1 score vs training size) plotted for tuned KNN is shown above. Bias and variance seem to fluctuate with converging at some points. Even the F1 score seems to be quite high and stabilizing.





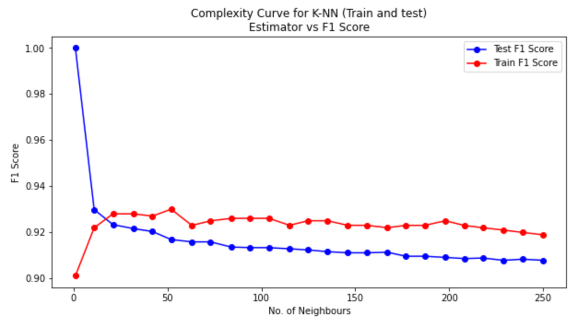
The training and validation error plotted above also indicates the same thing. Though the error is initially high, it shows a decreasing curve where validation and training error seems converging. This shows model is performing well with the dataset with minimal fitting issues. The only concern requiring regularization is fluctuating testing error which doesn't seem to converge.



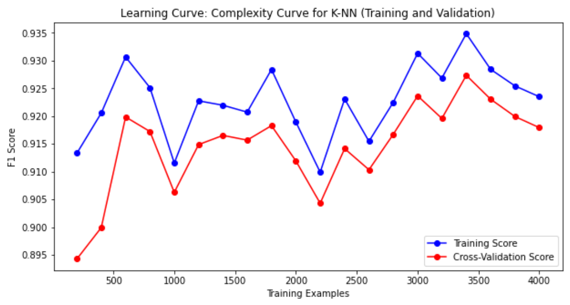
Finally Modeling time curve is plotted as shown above. The training time seems to be constant however, with this model, prediction time is showing continuous increasing curve with increase in training examples which questions model performance.

3.5.2 Wine Dataset

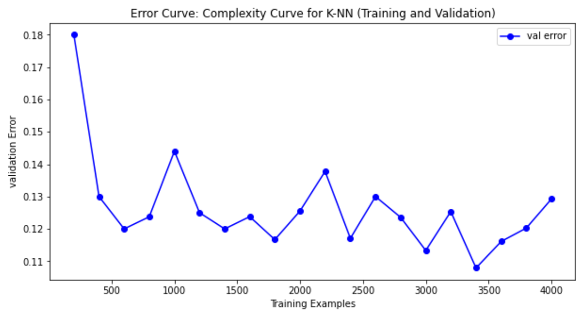
The data seems to fit well with neighbour less than 40. f1 score decreasing continuously stabilizes around 0.92 as shown in plot below. Neighbor number 20 is selected for further analysis.

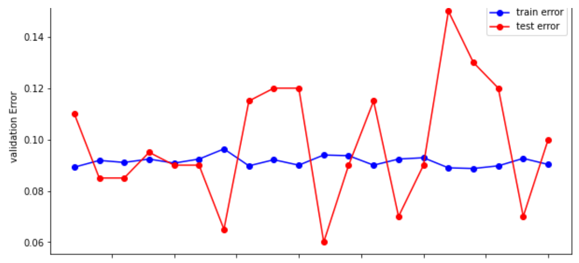


The complexity curve (f1 score vs training size) plotted for tuned decision tree is shown below:

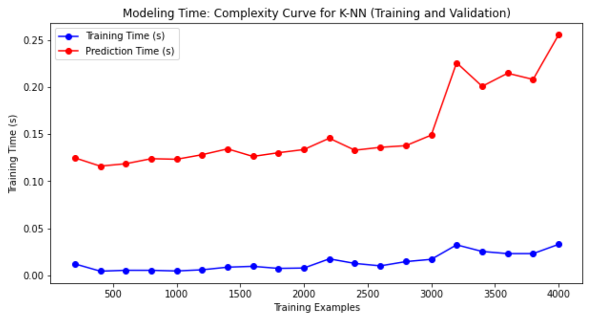


The complexity curve (f1 score vs training size) plotted for KNN is shown above. Bias and variance seem to fluctuate around 0.91 and doesn't seem converging. It can also be seen decreasing in trend with higher number of training examples clearing indicating underfitting.



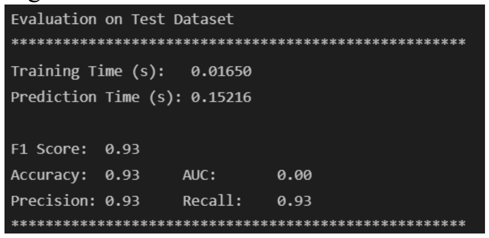


The training and validation error plotted above also indicates the same thing. Though the error is initially high, and a decreasing curve is seen in validation error, it doesn't seem stabilizing clearly indicating underfitting and requires regularization and generalization to solve this issue.



Finally Modeling time curve is plotted as shown above. The training time seems to be constant however, with this model, prediction time is showing continuous increasing curve with increase in training examples which questions model performance same as in case of phishing dataset.

Overall, model seems to perform well with accuracy and precision coming as 0.93 which is a same compared to phishing dataset.

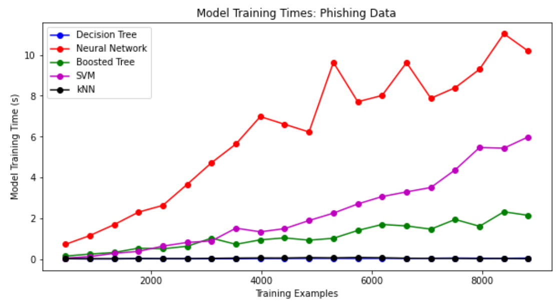


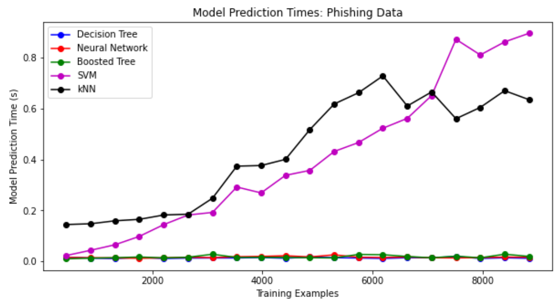
```
Evaluation on Test Dataset
*****************************************************

Training Time (s):   0.01650
Prediction Time (s): 0.15216

F1 Score:  0.93
Accuracy:  0.93     AUC:       0.00
Precision: 0.93     Recall:    0.93
*****************************************************
```

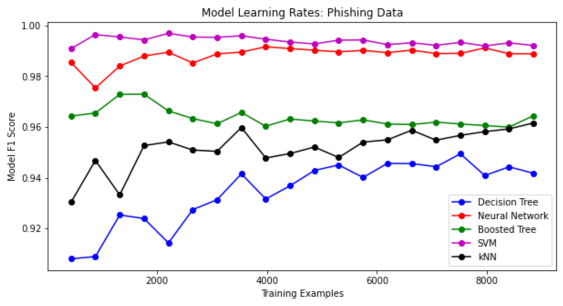
4. Model Performance Comparison

Finally, we tend to compare model performance based on training time, predicting time and F1 score. We came up with following analysis for respective dataset.

4.1 The Phishing Dataset





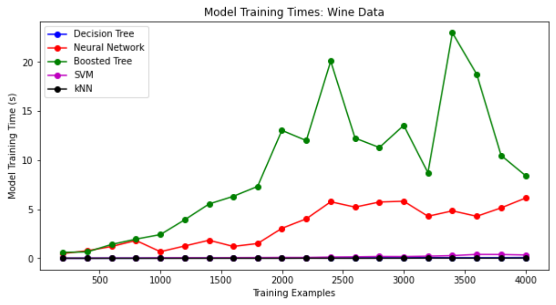


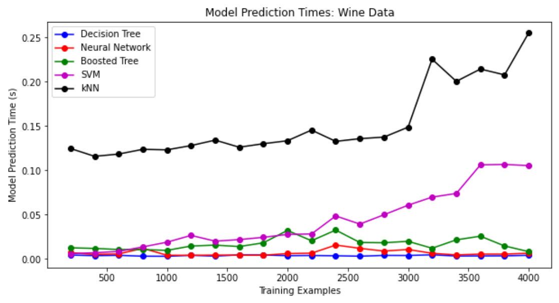
As seems above the neural network takes the maximum time in training where as K-NN and decision time takes the minimum. The SVM too shows incrementing curve however boosted tree stabilizes after some increment.

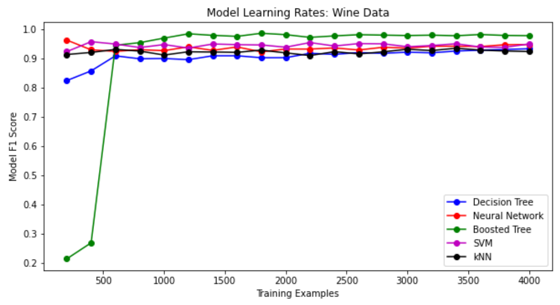
In case of prediction time, clearly SVM takes most followed by K-NN though the trend is opposite for the two when dataset size is small. All the other three tends to take minimal time to predict showing good performance.

Finally in terms of F1 score, SVM with rbf and Neural network seems to have best score followed by Boosted tree. Decision tree having minimal score among all. This means that the dataset has a very complex nature and complex algorithms (or combination of simple ones) are needed to learn the dataset.

4.2 The Wine Dataset







In case of wine dataset, Boosted tree seems to take more time training followed by neural network. Where as all the other three trains in around same time.

Incase of prediction time, k-NN is the one taking longest followed by SVM which shows increasing trend for larger dataset. All the other algorithm seems to take same time predicting the outcome.

Finally in terms of F1 score, boosted tree to starting of at very low score, shows best F1 score among all for larger dataset where as rest of the algorithms shows more or less similar score. This might be owing to lesser number of features models are being trained on compared to phishing dataset.

9. References

[1] "Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi (2012) An Assessment of Features Related to Phishing Websites using an Automated Technique. In: International Conferece For Internet Technology And Secured Transactions. ICITST 2012 . IEEE, London, UK, pp. 492-497. ISBN 978-1-4673-5325-0

[2] Mohammad, Rami, Thabtah, Fadi Abdeljaber and McCluskey, T.L. (2014) Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications, 25 (2). pp. 443-458. ISSN 0941-0643

[3] Mohammad, Rami, McCluskey, T.L. and Thabtah, Fadi Abdeljaber (2014) Intelligent Rule based Phishing Websites Classification. IET Information Security, 8 (3). pp. 153-160. ISSN 1751-8709

[4] https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html