

Supervised Learning

Anshuta Awasthi

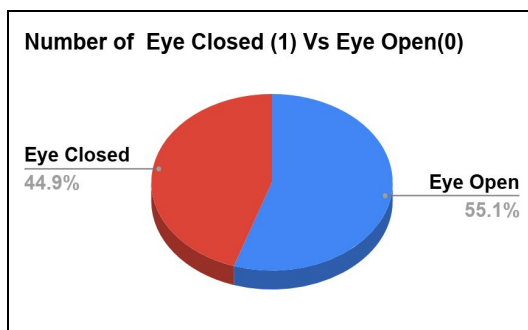
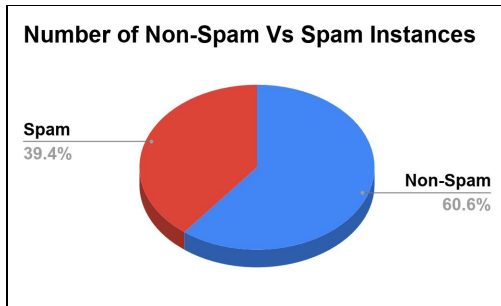
OMSCS -CS7641-Assignment 1

GTID - aawasthi32

1. About Data:

I have chosen below mentioned datasets from UCI machine Learning Repository:

- **Spambase Dataset:** This dataset is binary classification dataset ,to characterize the email whether it is spam or not .Most of the attributes indicate whether a particular word or character was frequently occurring in the email. The run-length attributes measure the length of sequences of consecutive capital letters. I found this dataset interesting to study because in recent years, spam emails have become a serious security threat, and act as a prime medium for phishing of sensitive information . Addition to this, it also spreads malicious software to various users. Therefore, email classification becomes an important research area to automatically classify original emails from spam emails. The set includes a total of 4601 observations, 2788 messages are classified as *Non-Spam(0)* and 1813 were effectively *Spam(1)* . There are 57 continuous real attributes.



EEG Eye State Dataset : An electroencephalogram (EEG) is a noninvasive test that records electrical patterns in your brain. In this dataset ,all data is taken from one continuous EEG measurement with the EEG Neuroheadset. The eye state was detected via a camera during the EEG measurement and added later manually to the file after analysing the video frames. '1' indicates the eye-closed and '0' the eye-open state. It is particularly very interesting dataset to study because the test results can also be used to detect driver's drowsiness. With increasing casualties, it is prudent to develop and deploy automated drowsiness detection

systems.<https://www.forbes.com/sites/tanyamohn/2016/08/08/nearly-83-6-million-american-drivers-are-sleep-deprived-new-report-highlights-dangers-high-cost/#8c3f3c74007a>

EEG results alone have been proved to be successful in detecting the state between alertness and drowsiness This dataset has 14,980 instances and 14 attributes(all continuous real).

8,257 records are classified as eye-open(0) and 6,723 are classified for eye-closed(1) state.

Why these are interesting from machine learning point of view ?:

Both the datasets have some similarities and some differences. Spambase dataset has less number of instances , while more number of features . In contrast to EEG dataset which has sufficiently large number of instances and comparatively less number of features. Computational complexity of ML algorithms is largely dependent of these 2 factors.Hoping to find interesting contrast when same algorithm is applied to these different datasets.

Both the datasets are similar in a way that both are binary classified datasets with both have continuous real attributes. Intent behind selecting the datasets with such similarities is it is better to compare “Apples” with “Apples . Meaning we can focus more on the specific differences between the two. While EEG signals are known to have noise and might contain outliers too ,identification of a best algorithm to separate spam and non spam emails is a challenging problem due to unstructured information and more number of features.

2. Decision Tree: We are not doing any feature scaling for decision tree . Tree based models are not distance based models and can handle varying ranges of features. Hence, Scaling is not required while modelling trees. The data is divided into train and test datasets in the ratio of 70:30.First of all I analysed the outcomes of simple decision tree , without applying any pruning . Training set and testing set accuracy were very high ,which led to suspicion about data . There are no missing values .On close observation , I found that there were several duplicate rows in dataset. On removal of duplicate rows below are the observations .In scikit learn default split criteria is Gini Index , so used that in simple decisionTreeClassifier . For Pruning , I chose to do pre-pruning with gridsearchCV on both the datasets, by limiting the max_depth of tree.

Passed following parameters in GridSearchCV

Parameters: {'criterion':['gini','entropy'],'max_depth':[4,5,6,7,8,9,10,11,12,15,20,30,40,50,70,90,120,150]}

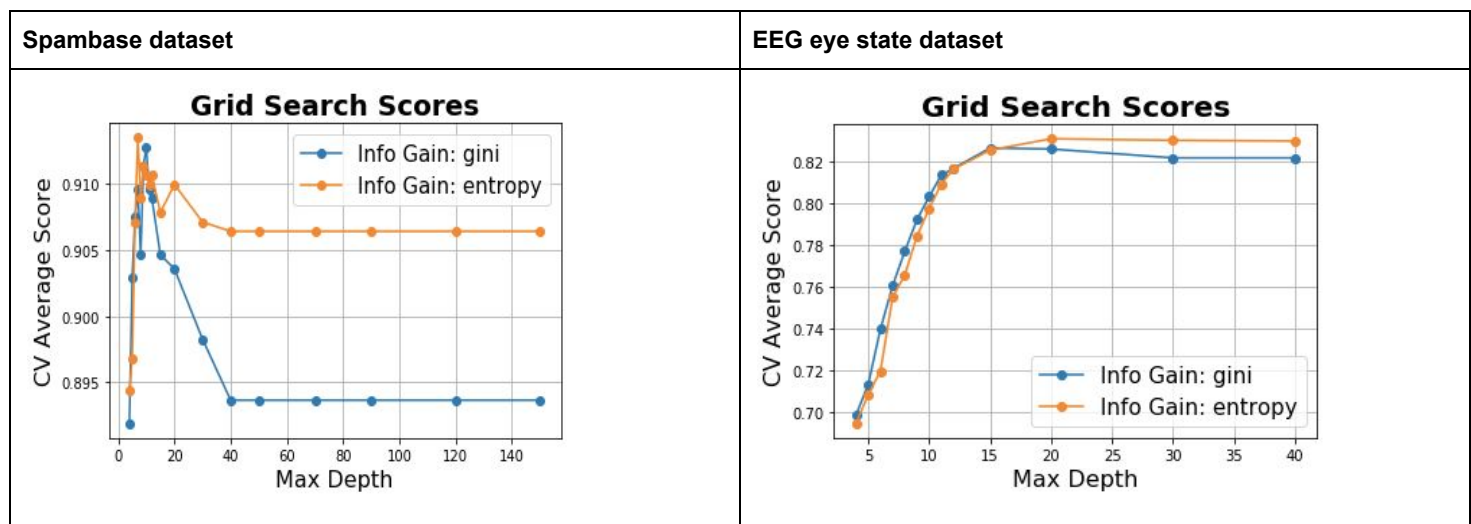
According to the top 2 rows in the below table , for decision trees ,without any form of pruning we have very high values of training set accuracies and tree depth. As depth increases , model complexity also increases and model tends to overfit the training data for both Spambase and EEG eye state datasets.

Last two rows show the outcome (after pre-pruning applied by limiting the max_depth) of best tree resulted from gridSearchCV. The smaller value of tree depth parameter shows that model complexity has been reduced significantly , inturn reducing the accuracy on training data .

Dataset	Pruning	Time To train	Time To Test	Training set Accurac y	Testing set Accuracy	Tree Depth	Impurity Measure
Spambase	No	0.515 Sec	0.00201 Sec	0.99964	0.88502	33	Gini
EEG Eye dataset	No	0.137 Sec	0.00184 Sec	0.954	0.84779	34	Gini
Spambase with Grid searchCV	Yes	13.11 Sec	0.00125 Sec	0.9649	0.9214	8	Entropy
EEG Eye dataset with GridSearchCV	Yes	23.03 Sec	0.00145 Sec	1	0.8474	30	Entropy

Last two rows display the outcome of best resulted Decision Tree after GridSeachCV. There are following observations

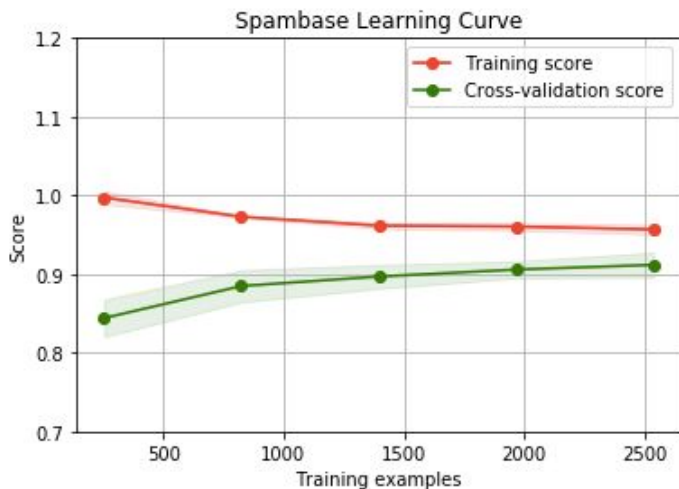
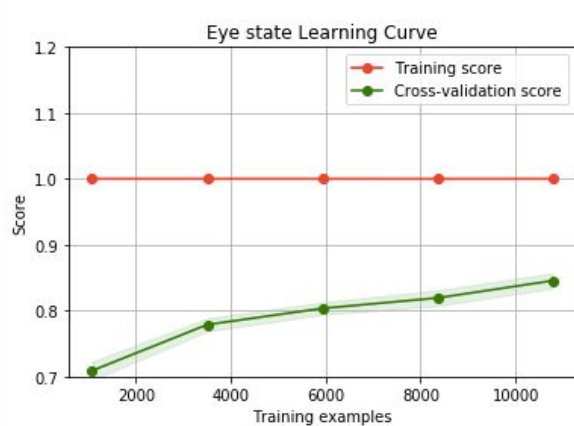
1. It shows that optimum decision tree gives best result with “Entropy” . but there is a drawback in using Entropy for deciding the split , Entropy is logarithmic and it takes more time than “Gini”index calculation. This might be the reason why scikit learn uses it's default criteria as “Gini”.
2. However there is not much difference,below graphs show the scores variation .With both “Gini” and “Entropy”, optimum tree depth achieved is approx same and little difference in score.
3. Also below graphs validate the results of above table that optimum accuracy is achieved at the max tree depth of 8 and 30 for spambase and EEG eye dataset respectively after pre-pruning.



Learning curves after Pruning:

For **spambase dataset**, we see overfitting is reduced after pruning, both the curves(training and test set) show slight convergence. The training set accuracy is reduced, that displays the reduction in overfitting too.

EEG Eye Dataset does not show improvement after pruning. It is still overfitting with training set accuracy as high as 1. Also the best fitted DT outcome of gridSearchCV shows Max depth as 30, which is clearly indicative of complex model. This can be explained as EEG dataset being noisy and we are getting such distorted results due to the attempt to fit every training data instance, including noisy ones, into the model descriptions.

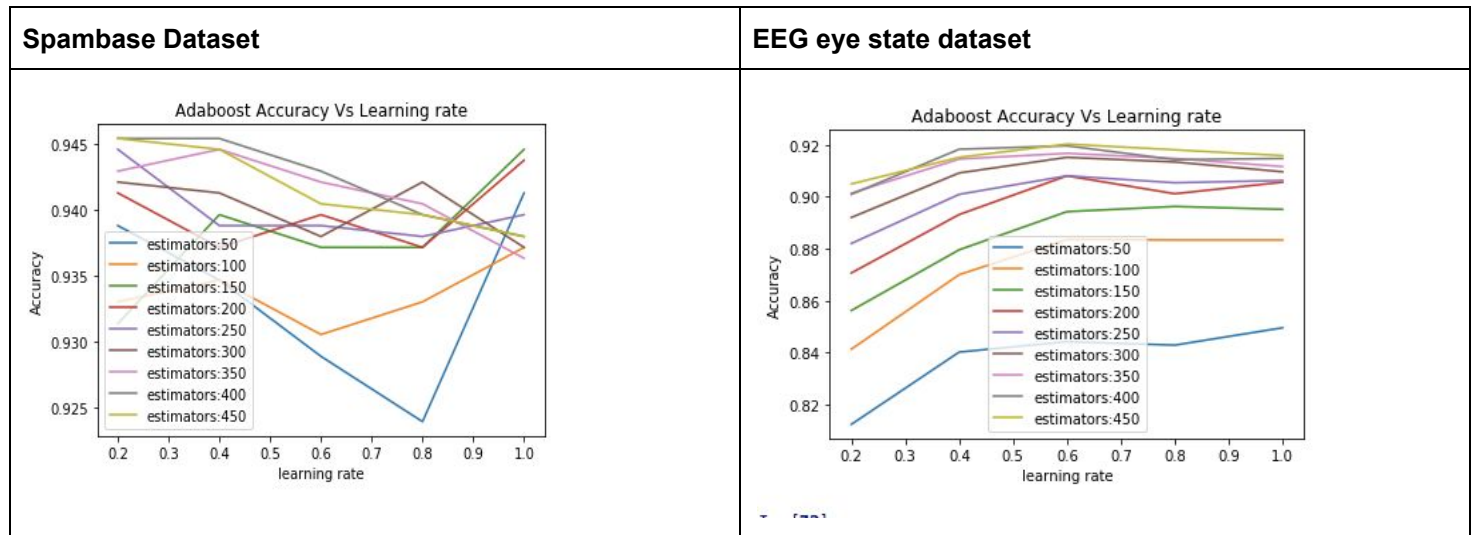
Spambase dataset(with Pruning)	EEG eye state dataset(With Pruning)																														
																															
Confusion Matrix: <pre>[[692 27] [83 407]]</pre>	Confusion Matrix: <pre>[[1451 216] [242 1087]]</pre>																														
Classification Report: <table><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr><tr><td>0</td><td>0.89</td><td>0.96</td><td>0.93</td><td>719</td></tr><tr><td>1</td><td>0.94</td><td>0.83</td><td>0.88</td><td>490</td></tr></table>		precision	recall	f1-score	support	0	0.89	0.96	0.93	719	1	0.94	0.83	0.88	490	Classification Report: <table><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr><tr><td>0</td><td>0.86</td><td>0.87</td><td>0.86</td><td>1667</td></tr><tr><td>1</td><td>0.83</td><td>0.82</td><td>0.83</td><td>1329</td></tr></table>		precision	recall	f1-score	support	0	0.86	0.87	0.86	1667	1	0.83	0.82	0.83	1329
	precision	recall	f1-score	support																											
0	0.89	0.96	0.93	719																											
1	0.94	0.83	0.88	490																											
	precision	recall	f1-score	support																											
0	0.86	0.87	0.86	1667																											
1	0.83	0.82	0.83	1329																											

3. Adaptive Boosting : Outcome of Adaptive Boosting classifier on both the datasets has been studied as a parameter of test set accuracy. Adaptive Boosting has one clear advantage over decision trees that it improves on the time taken to train the model.

Plotted following graphs to see the impact of hyperparameters like learning rate and n_estimator values on test set accuracy. This graph was helpful in choosing the values for hyperparameters. Although accuracy improves as we increase the number of estimators, one drawback of having more number of estimators is that it significantly increases the complexity and training time as well as testing time.

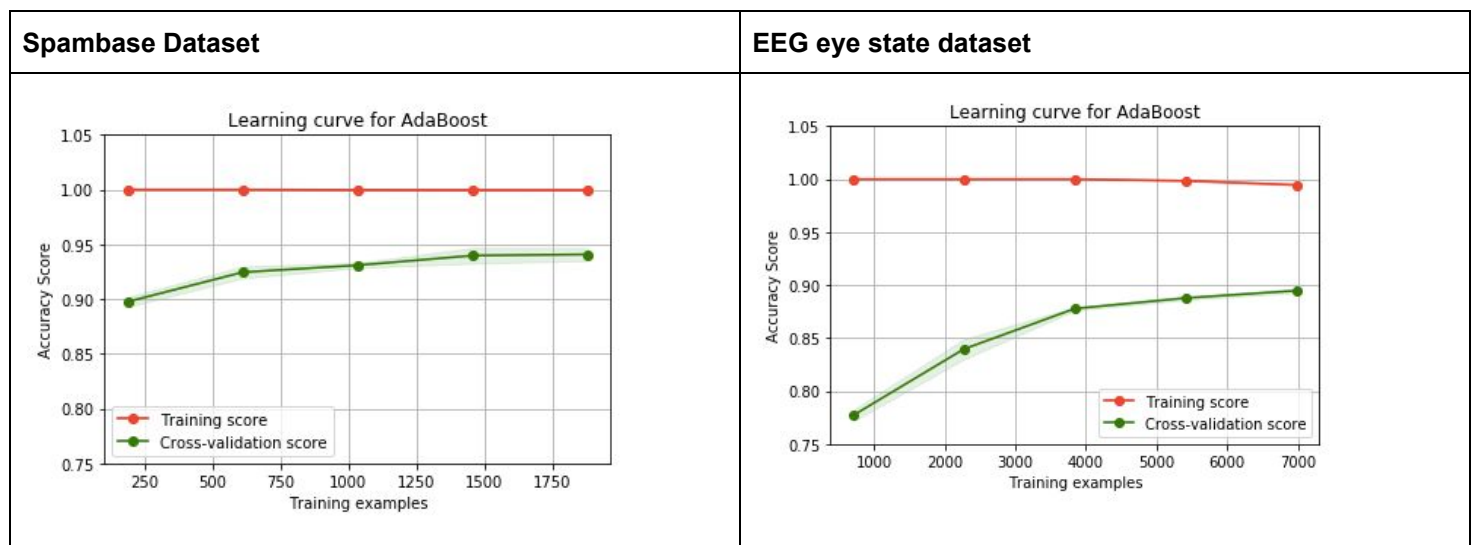
For spambase dataset, N-estimator =150 with learning_rate = 1.0 provides the good improvement in accuracy as well as training time when compared to decision tree with GridsearchCV. It will be good idea to not chose very high value for n_estimator, for getting more accuracy, since the trade off will be with time taken to train the model.

Similarly for EEG eye state dataset, $N_estimator = 200$ and $learning_rate = 0.6$ provides a very good improvement in accuracy and training time when compared with the results of best decision tree returned by gridsearchCV in previous model.



Below are the learning curve which show the performance in terms of cross validation accuracy score on different training set sizes. As it looks this model is also suffering from slight “overfitting”. Cross validation score is showing gradual upward trend in both the datasets. In EEG eye state dataset, the training accuracy shows a slight downward bend at the end. Although the cross validation score gives good value of accuracy. Looks like there can be few solutions to get better results

a). Addition of more data is going to solve the problem of overfitting here. b). Another one is increasing the number of estimators as shown in curves above. c). The last one is to apply other algorithms on these datasets and see which is giving best results. We are going to choose the last one and see if there is any model which outperforms these results!!



Dataset	Time To test	Time To train	Training set Accuracy	Testing set Accuracy	N Estimator	Learning Rate
Spambase Dataset	0.04309	1.9847	0.99220	0.939619	150	1
EEG eye state dataset	0.1196 Sec	5.4864 Sec	0.98168	0.908322	200	0.6

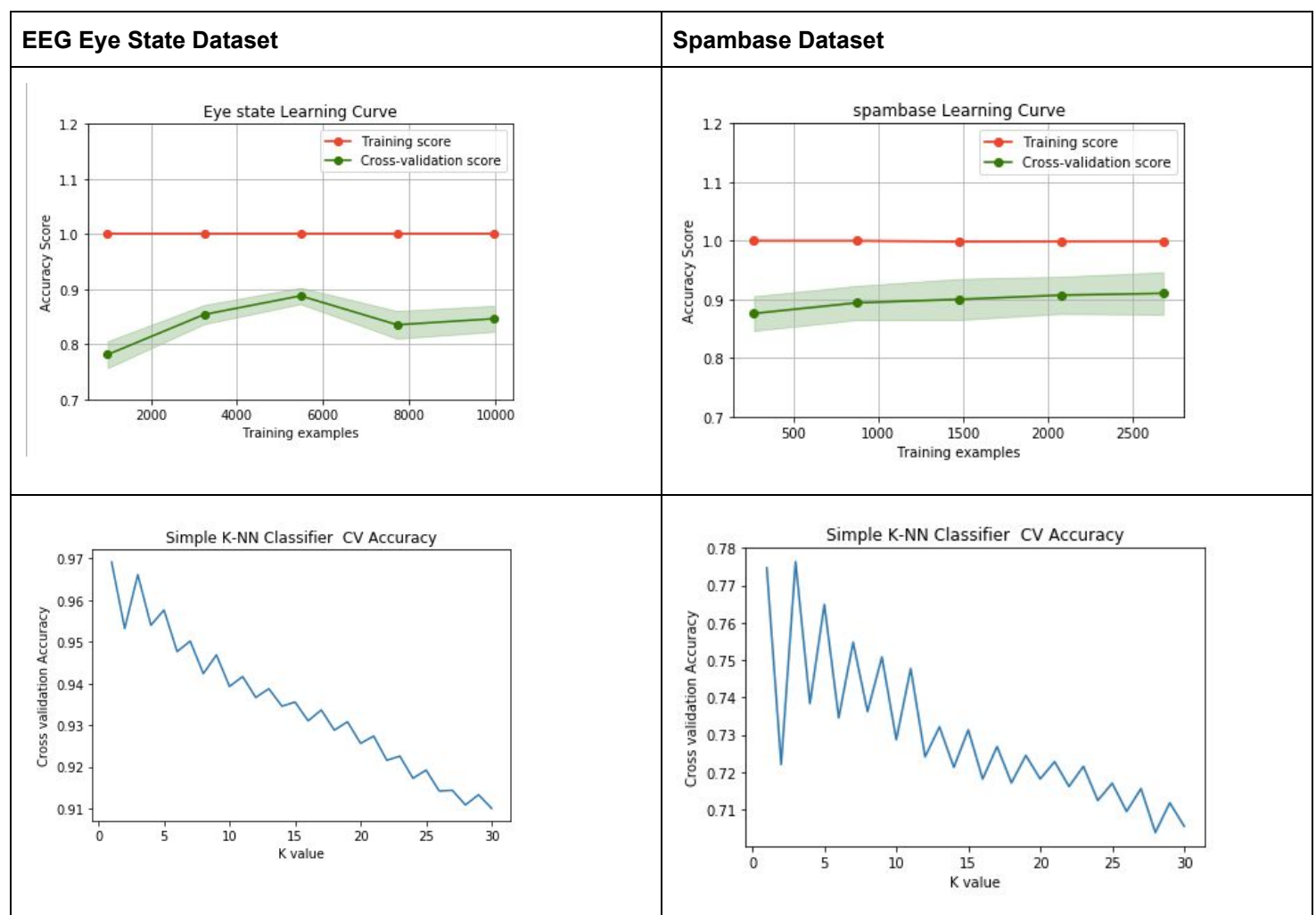
If we compare AdaBoost results with best resulted Decision Tree from GridSearchCV, we see that for spambase dataset training time has been reduced from 13 secs to 2 secs (approx.) while improvement in test set accuracy is from 0.92 to 0.94.

Similarly , for EEG eye state dataset also training time has been reduced to 5.5 secs from 23 secs , while showing significant improvement in accuracy from 0.85 to 0.91

4. K-NN: Feature scaling of data is necessary for K-NN algorithm since it uses Euclidean distance between two data points in its computation.

Value of K : To determine the best value and to understand the effect of number of neighbors on the accuracy in K-NN classifier, below graph is plotted . As we increase the value of K , “CV Accuracy” is going down , since increasing K will increase the bias and decrease the variance. For very low values of K(such as K=1) , our model will suffer from high variance but low bias, because model will fit only to the 1-nearest point. This means model is really close to the training data, not a generalized one hence - overfitting.

K-NN Learning curves : This model is clearly overfitting on both the datasets. Training set accuracy (shown in the curve and table) for both the datasets is nearly 100% , which is not resulting in good test set accuracy. Optimum values of K has been chosen by gridsearchCV are 6 and 8 for EEG and spambase dataset respectively. The curve in second row of the table shows that increasing values of K results in degrading performance .Spambase learning curve cross validation accuracy is gradually improving , hence adding more data will help in this case . While cross validation score in eye state dataset is showing unpredictable results. This might be due to noise present in EEG signals and KNN is not giving satisfactory results because KNN is sensitive to noise especially for small values of K



	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.85	0.85	0.85	2491	0	0.92	0.92	0.92	730
1	0.82	0.81	0.81	2003	1	0.88	0.88	0.88	479

Dataset	Test set Accuracy	Train set Accuracy	Testing Time	Training time	Confusion Matrix	Best value of K
spambase	0.901819	0.999085	0.21208 Sec	271.2195 Sec	<pre>[[672 58] [56 423]]</pre>	8
EEG eye state	0.830380	1.0	0.33266 Sec	374.8086	<pre>[[2126 365] [386 1617]]</pre>	6

Above data shows , that KNN model is not a best choice , since it performed fairly poor on both the datasets. Not only in terms of accuracy , but also in terms of time complexity, it is giving worse results than other algorithms. There are, of course, many ways to improve upon this base algorithm. Common modifications include weighting, and specific preprocessing to reduce computation and reduce noise, such as various algorithms for feature extraction and dimension reduction.

5.SVM:

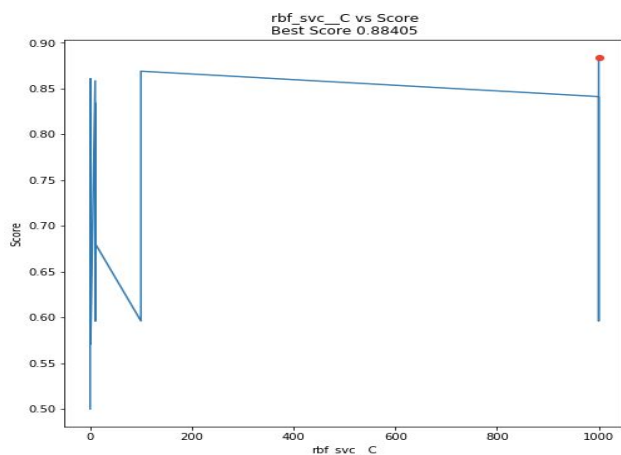
Feature scaling of data is necessary for SVM also since SVM tries to maximize the distance between the separating plane and the support vectors. If one feature (i.e. one dimension in this space) has very large values, it will dominate the other features when calculating the distance. Rescaling all features (e.g. to [0, 1]), they all have the same influence on the distance metric.

Tuning C and gamma in gridSearchCV: It is very important to adjust the C and gamma values to get the optimum performance from our model. If the value of C is large then model tries to minimize the misclassification and we get the higher variance and lower bias, which may lead to the problem of overfitting. If the value of C is small then model is lenient to misclassification and we get lower variance/high bias. For simplicity , I have taken exponential (10^x) values of C and gammas in parameter grid to cover wide range. A better choice would have been values in the exponential of 2 to get more precise results .

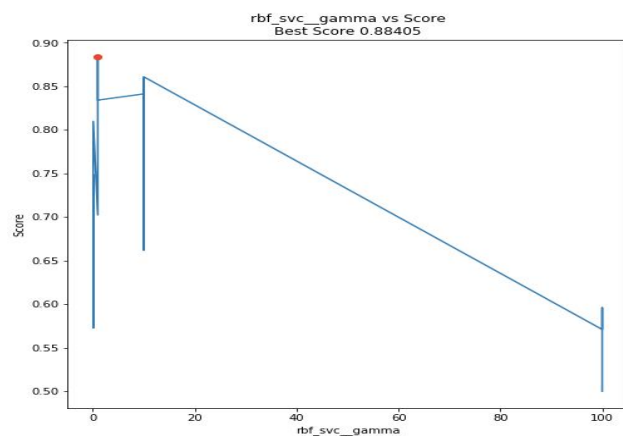
This Graph shows how different C values and Gamma values affect the “cross validation accuracy” for SVM with kernel = “rbf” calculations when used with gridSearchCV on EEG eye state dataset. Range of parameter used in gridSearchCV:

Cs = [0.1, 1, 10, 100, 1000]

gammas = [0.1, 1, 10, 100]



Best C = 1000



Best gamma = 1

Red spot in graphs show the value of C and gamma when the best score is achieved.

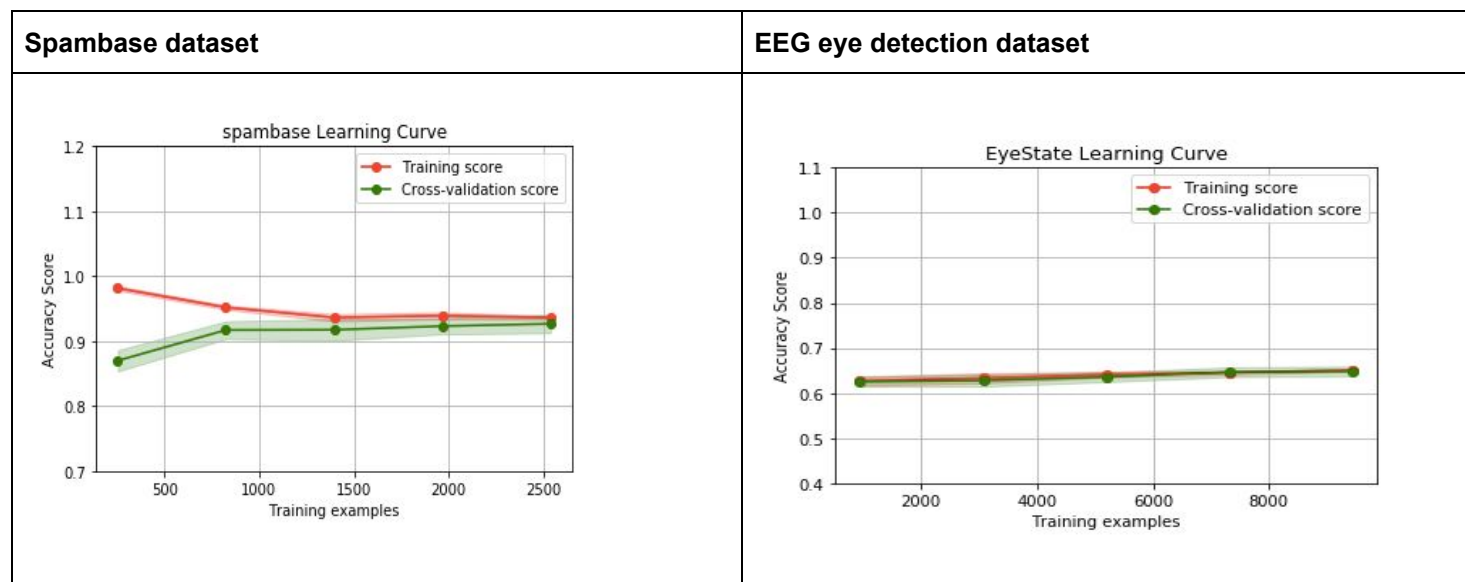
Note: Not including other graphs showing C and gamma variations for spambase dataset and for linear SVM to keep the report concise. 1 graph is enough for demonstration. Best parameters resulted for each case are given in the table ahead.

Learning Curve and result for SVM with Linear kernel:

For spambase dataset: Training accuracy decreases close to the level achieved from the cross validation accuracy. The two curves are converging for accuracy of 0.92, which is by far the best we have achieved so far.

For EEG eye state dataset: The results obtained from SVC with kernel -"Linear" for this dataset are the worst one. This is pretty reasonable as we have around 15000 instances and SVM is not feasible for large datasets. In SVM storing the kernel matrix requires memory that scales quadratically with the number of data points. **Training time for linear SVM algorithm also scales superlinearly with the number of data points. So, these algorithms aren't feasible for large data sets.** As already stated that EEG signals are prone to noise. Noisy data is adding to the bad performance of Linear SVM in the case of EEG eye state dataset. I don't think that the results are reliable, both the test and cross validation scores converged too fast with not very good value of accuracy. This might be due choice of incorrect hyperparameter range. As shown in the table that it has taken longest time for training, also results in confusion matrix are skewed. It is better to reject this model outright for EEG eye state dataset.

"Linear SVM produced the quite contrasting results for these 2 datasets, while it gave the best results for spambase data in terms of both, testing time and accuracy. It clearly did not work for EEG eye state dataset."

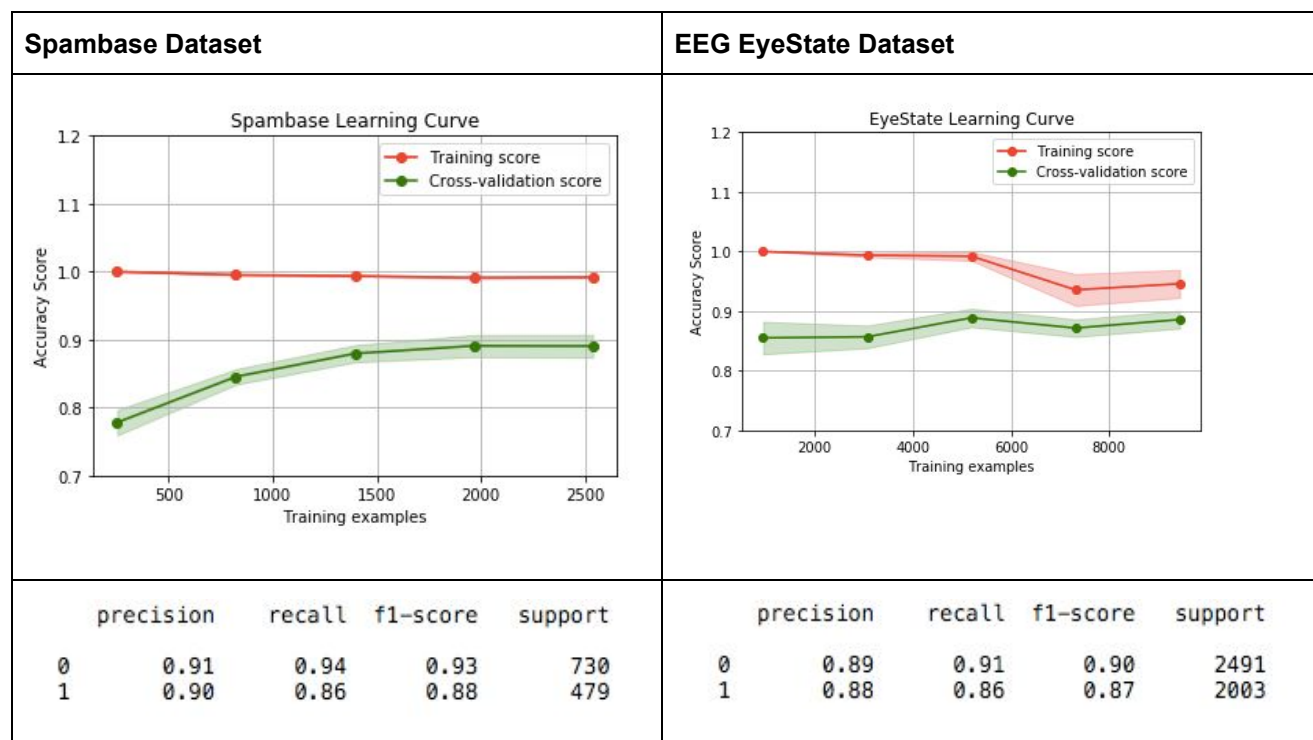


	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.94	0.93	0.93	730	0	0.65	0.83	0.73	2491
1	0.90	0.91	0.90	479	1	0.68	0.45	0.54	2003

Learning Curve and result for SVM with RBF kernel: RBF kernel is gaussian function, and gamma is inverse of gaussian here. Larger the “gamma”, narrower the gaussian bell. Below learning curves are obtained with the best model outcome from gridSearchCV.

EEG curve is showing the converging tendency, As table below shows that for EEG eye state data set our model is using $C = 1000$, which is at the higher side. As discussed, larger values of C results in the case of overfitting due to high variance and low bias, which is justified with the learning curve for EEG. On the other hand, spambase curve shows “overfitting” while the value of C used is 10 (which does not look at the very higher side). For spambase RBF kernel did not give better results than linear kernel, but for EEG Eye state dataset, it performed way better than linear kernel.

In this case model can be improved by either adding more data or tuning values of C and gamma by selecting wider range. The range currently used is 10^{-1} to 10^3 .



Results For SVM (Linear and RBF):

Dataset	kernel	Test set Accuracy	Train set Accuracy	Testing Time (in Secs)	Training time(in Secs)	Confusion Matrix	“C”	“Gamma”
spambase	Linear	0.91878	0.928	0.0425	11834.76	$\begin{bmatrix} 680 & 50 \\ 45 & 434 \end{bmatrix}$	10	0.1
EEG eye state	Linear	0.638	0.631	0.6140	18053.21	$\begin{bmatrix} 2069 & 422 \\ 1111 & 892 \end{bmatrix}$	1000	0.1

spambase	RBF	0.889	0.9013	0.1371	212.077	$\begin{bmatrix} 685 & 45 \\ 65 & 414 \end{bmatrix}$	10	0.1
EEG eye state	RBF	0.8869	0.93125	0.435	2413.083	$\begin{bmatrix} 2266 & 225 \\ 272 & 1731 \end{bmatrix}$	1000	1

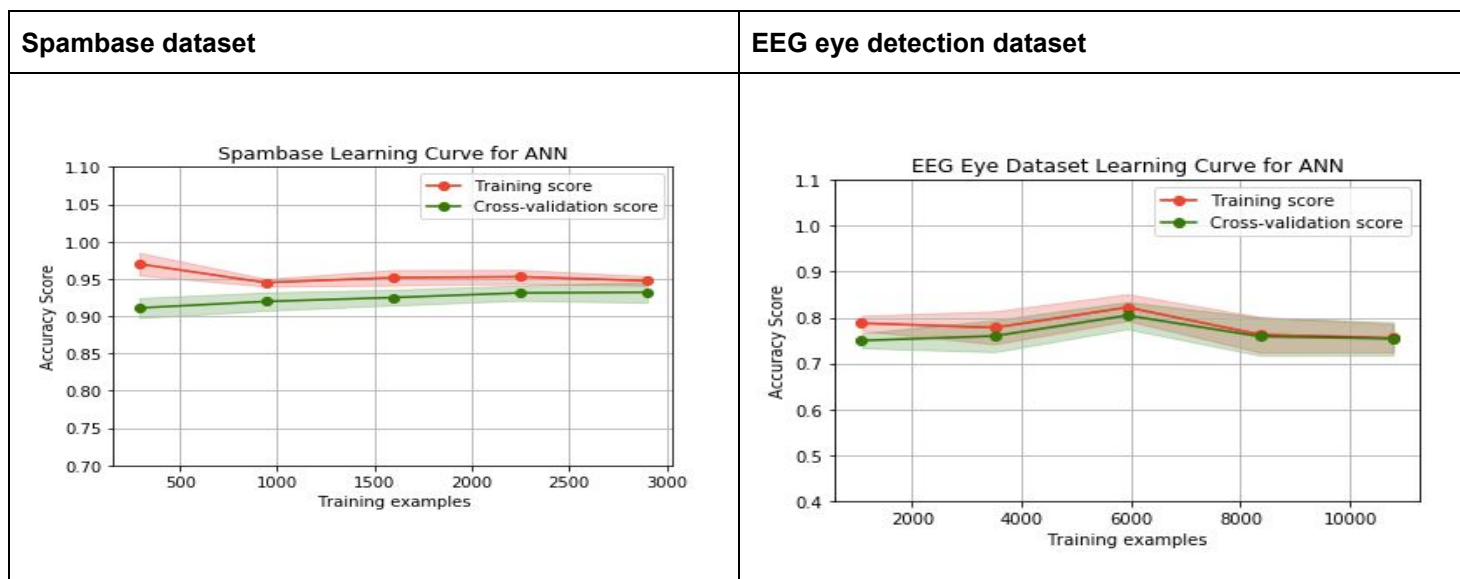
6. Artificial Neural Networks:For ANN also, feature scaling is done first then proceeded with MLPClassifier implementation with gridSearchCV used for hyperparameter tuning and getting best outcome.

Activation function used for both the datasets is “Relu” .Since both the datasets used here are binary classification ones , my first choice was “logistic” but it was taking longer (logistic function for example = $1/(1+e^{(-a)})$) uses an exponent, which is computational slow when done often. I am not discussing here the results obtained from “logistic” activation, to keep the report concise. But the results were not better than “Relu” , the reason can be vanishing gradient issue with sigmoid.The derivative of the sigmoid function is always smaller than 1. In backpropagation,If we have many layers, and we multiply these gradients, and the product of many smaller than 1 values goes to zero very quickly. However I am not sure about that this is the exact reason for sigmoid performing worse than relu in this case , but got better results with ‘Relu’.

Learning Curve and result for Neural Network: In NN we have many hyperparameters to tune. Here we have tried to tune alphas , number of hidden layers and number of neurons in each layer . It was difficult to choose the appropriate ranges of parameters , which need to be passed for tuning.Alpha is a regularization parameter , large values of alpha encourage smaller weights, results in less complex model - reduces overfitting. Hence the wide range for alpha has been used. As described in udacity lectures , most of the functions can be represented by 1 hidden layer , and any arbitrary function(even if it is discontinuous) can be represented by adding 1 more layer to the model. So, at the most 2 layers can give us best results.For deciding the range of number of neurons in each layer , referred to the book *Introduction to Neural Networks for Java* by **Jeff Heaton**. There are several rule of thumbs, like it can be chosen as mean of nodes in input layer and output layer and appropriate range was chosen accordingly.In tuple range , we defined the several combination of number of neurons with hidden layer size 1 and 2 .

For spambase dataset: We have received moderately good results for spambase , the training and test accuracy curves are converging for satisfactory values of accuracy. Neural network has resulted as good fitting model for this dataset. Both the curves show different patterns on changing the learning rate On increasing the value of learning rate passed to the MLPClassifier , the curves show gradual learning as we increase dataset size and are better to visualize and understand. ANN was much faster than SVM in terms of training and testing times also.

For EEG eye state dataset:



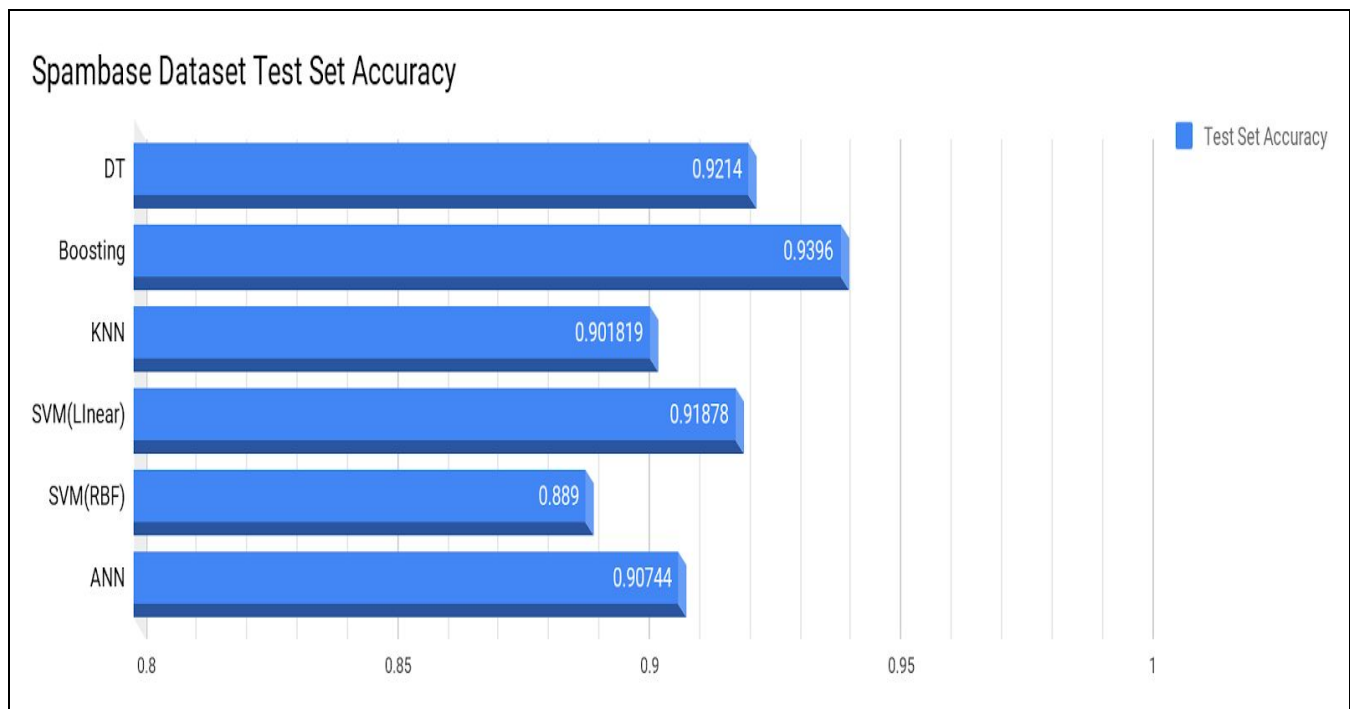
	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.95	0.89	0.92	481	0	0.78	0.74	0.76	1667
1	0.85	0.93	0.89	325	1	0.69	0.74	0.72	1329

Results for ANN after GridSearchCV:

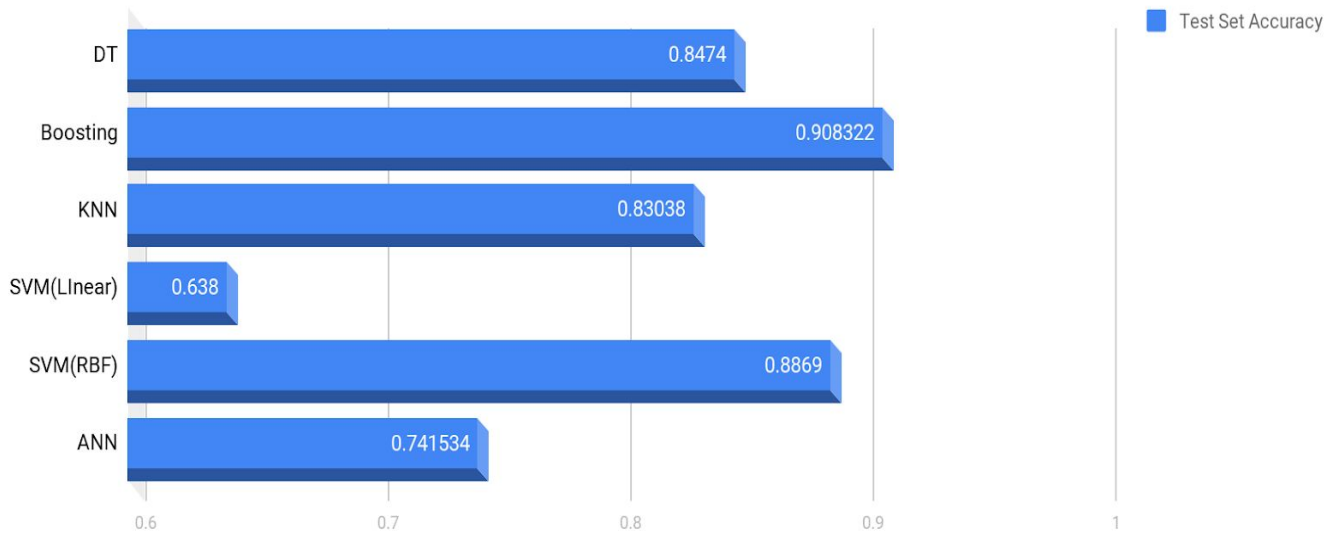
Dataset	Activation	Test set Accuracy	Train set Accuracy	Testing Time (in Secs)	Training time(in Secs)	Confusion Matrix	"Alpha"	Layers and number of neurons
spambase	relu	.90744	0.9386	0.00452	331.519	$\begin{bmatrix} 426 & 55 \\ 23 & 302 \end{bmatrix}$	0.01	Layer =2, Neuron =9
EEG eye state	relu	0.741534	0.7421	0.00447	1080.818	$\begin{bmatrix} 1233 & 434 \\ 341 & 988 \end{bmatrix}$	0.0001	Layer =1, Neurons =14

Conclusion: Below barcharts show the comparative performance of all models on both the datasets. The charts show the comparison in terms of Accuracy and Training Time complexity .

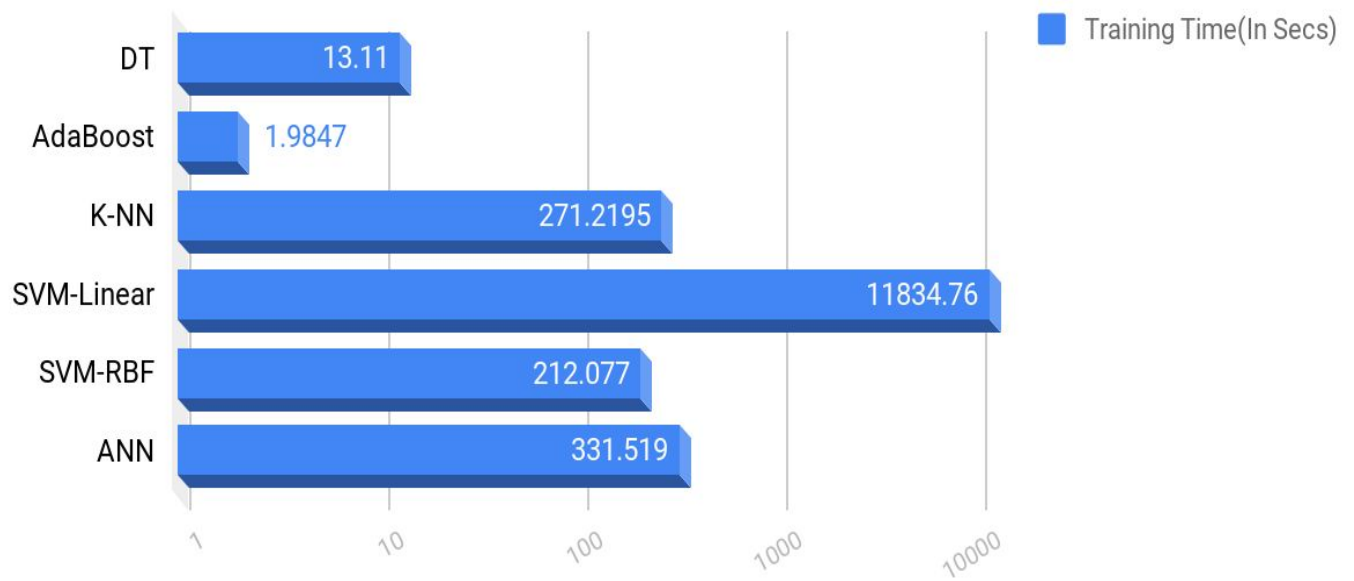
The observations are followed :



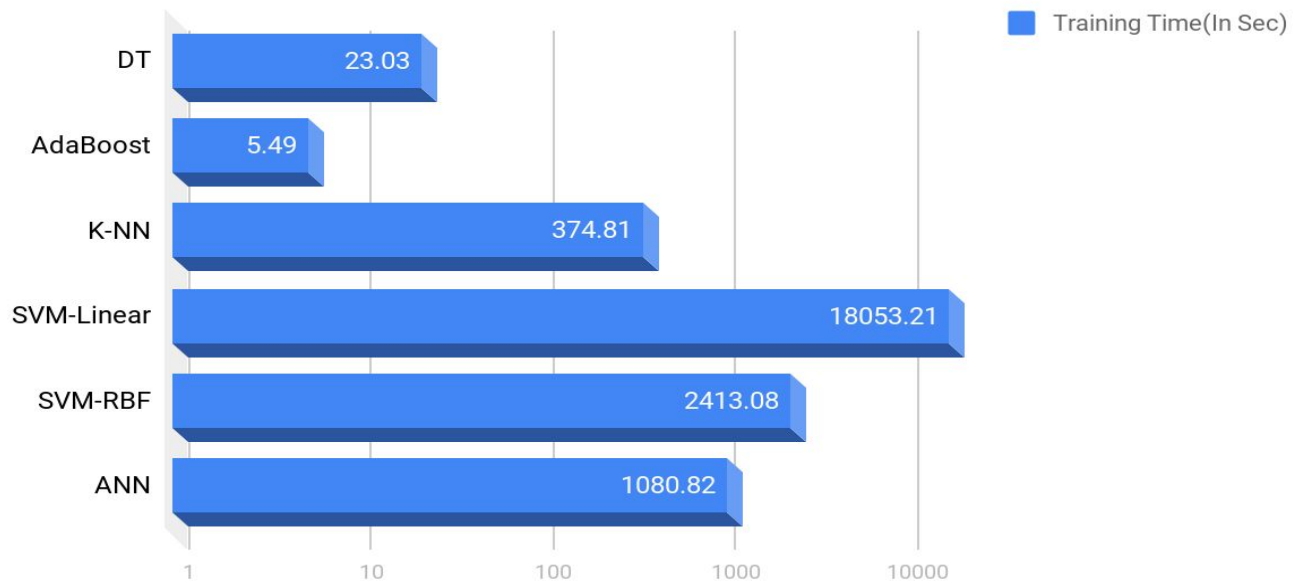
EEG Eye State Dataset Testing Accuracy



Training Time Comparison for Spambase Dataset



Training Time Comparison for EEG Eye State Dataset



1. In all the models, “AdaBoost with decision tree” is giving best results, both in terms of training time as well as accuracy for both the datasets, since both are binary classification. AdaBoost is proven to give best results for binary classification datasets, while it may not produce best results for multiclass classification problems. For Spambase dataset, Decision tree itself gave good results and ensemble of decision trees in boosting improved the performance further.
For EEG eye state dataset, although Decision tree did not give the best results, but ensemble of decision trees in boosting significantly improved performance from single decision tree.
2. For spambase dataset, second best model is decision tree both in terms of accuracy and training time complexity. However for EEG eye state dataset this is not the case. In terms of accuracy, SVM with RBF kernel is performing better than decision tree. But if training time complexity is also taken into account, then I would say that Decision Tree is better choice than SVM(kernel =RBF), since decision tree model has lower time complexity.
3. SVM(kernel = Linear) is giving contrasting results for “Accuracy” in both the datasets. While for spambase dataset it converges to good accuracy of test dataset, but for EEG eye state dataset it shows worst results due to large number of records and reason being stated before in SVM Linear section of the report. In terms of training time complexity SVM (kernel =Linear) turns out to be bad choice.
4. SVM(kernel =RBF) gives very satisfactory results in terms of accuracy for EEG eye state dataset, in fact after boosting, SVM RBF gives the best score for EEG eye state dataset. SVM Linear has performed bad and SVM RBF performed very good. RBF kernel generally outperforms linear or polynomial kernels because RBF is squared exponential kernel which is non-parametric model while linear and polynomial kernels are parametric models. In a way nonparametric model means that the complexity of the model is potentially infinite, its complexity can grow with the data. If you give it more and more data, it will be able to represent more and more complex relationships. In contrast a parametric model's size is fixed, so after a certain point your model will be saturated, and giving it more and more data won't help.
5. ANN model (with MLP classifier) has produced results with slightly lesser accuracy than SVM(for spambase, SVM Linear outperformed ANN and for EEG eye state dataset, SVM RBF outperformed ANN), but still ANN can be better choice than SVM since it performs much better in terms of training time complexity. The memory space required for SVM will be the number of support vectors (nSV) multiply by the number of feature values. This is significantly large compared to NN which only need to store the weight.