Report

This report depicts the implementation of three supervised learning methods namely, Support Vector Machine (SVM), Decision Tree, and Boosting on two distinct data sets. The goal is to analyze and understand the behavior of these supervised learning methods on the datasets. The algorithms are performed with the help of Sckitlearn library in Python language (on Jupyter notebook).

Dataset: Mushroom Classification

This dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family Mushroom drawn from The Audubon Society Field Guide to North American Mushrooms (1981). Each species is identified as definitely edible or poisonous. You can download the dataset from here.

My inclination towards working on this dataset is because mushrooms classification into edible and poisonous makes huge impact in shrooming. Also, I have not worked on a large dataset containing categorical features. This will help me learn various data processing steps before feeding data to the algorithms.

Following are some preliminary steps performed on the dataset:

1. Converting the categorical variables into dummy/indicator variables
2. Performing Principle Component Analysis to reduce 95 one-hot-encoded features to only 2 Principle Components
3. Splitting the dataset into train and test set with split ratio of 70:30 respectively
4. Training different classification models on these two components

I have chosen 2 Principal Components as they describe 94% of variance in my dataset and keeping in mind the computational time and resources. One can increase the number of columns if more accuracy is required.

Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. Keeping the same thing in mind, the data has been trained for different hyperparameter.

1. Linear Model:

For linear model, the data has been trained for three different values of hyperparameter C [0.001, 0.1, 1] and the accuracy is reported for the best performing model along with the confusion matrix. Cross validation is also performed in parallel with 10-fold number of splits and mean training and validation scores are reported. Refer to the Table 1.0 for all the detail scores.

For better understanding of the increase in accuracy for different C values, A plot of training, validation, and testing accuracy is shown in Fig 1.0. From the plot it can be inferred that $Accuracy_{training} > Accuracy_{validation} > Accuracy_{testing}$. This is an expected result as we have trained our model on training data and hence it's accuracy will be highest followed by the rest. From the plot, as C value approaches 1, the accuracy of the three datasets is not varying much and for higher values of C, there is no major impact on the accuracy.

Dataset accuracy	Linear (C=0.1)	Polynomial (degree 3)	RBF(C=1)
Training	0.8991	0.8783	0.9287
Validation	0.8987	0.8788	0.9281
Test	0.8999	0.8679	0.9258
Test confusion matrix	[[1224 33] [211 970]]	[[1244 13] [309 872]]	[[1232 25] [156 1025]]

Table 1.0 Comparison across SVM models

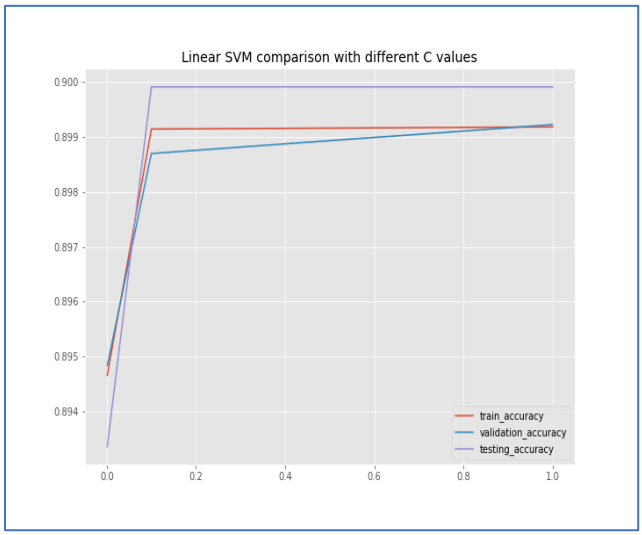

Figure 1.0 Train, validation and test plot – Linear model

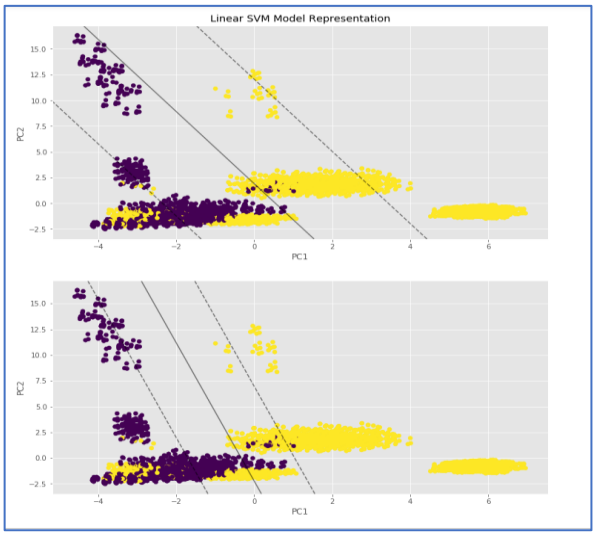

Fig 1.1 Boundary of Linear SVM

Figure 1.1 shows the boundary variation as we change the values of C. For higher value i.e., C = 0.0001 we get the boundary which is more flexible (it treats all values that falls inside or on the boundary as 0). Whereas for larger values of C, the model becomes more stringent in classifying and hence the boundary width decreases.

2. Polynomial Model:

For Polynomial model implementation, the data has been trained over different degree (i.e. 2, 3, and 4) of polynomial function (by invoking the classifier and initializing with different degrees) and the accuracy is reported for the best performing model along with the confusion matrix. Cross validation is also performed in parallel with 10-fold number of splits and mean training and validation scores are reported. Refer to Table 1.0 for those values.

From Figure 2.0, we can infer that the accuracy of train and validation dataset are almost similar for this model and the testing accuracy falls below both. As the degree of polynomial increases from 2 to 4, the accuracy of all the three datasets decreases. The reduction in the accuracy is very sharp after degree 3 which is also a sign that higher degree models may not provide fruitful results.

3. RBF Model:

This model provided the best accuracy for the dataset in all aspects (for e.g. training, validation and test). The Figure 3.0 shows the accuracy plot for different values of C hyperparameter [0.0001, 0.1, 1] for all three datasets. With increase in values of C, the accuracy for all the three datasets becomes constant even if you increase C parameter.

Figure 3.1 shows the variation in error rates for both train and validation data. It is observed that validation error toggles more than the training set data error. This shows that on passing new unknown values to the

model, the model error drops for different folds and is better for certain fold making the average error less overall. Please refer Table 1.0 for accuracy measures and confusion matrix.

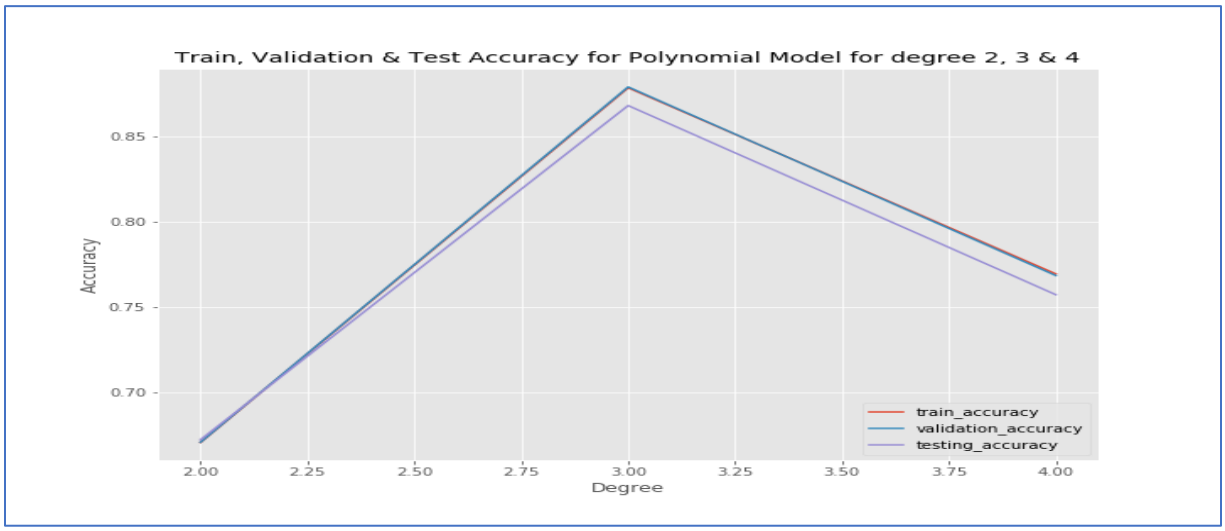

Figure 2.0 Train, Validation & Test accuracy plot - Polynomial

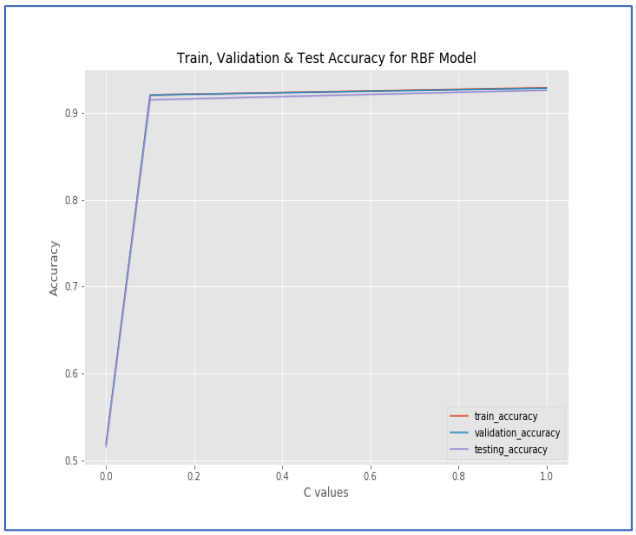

Figure 3.0 Train, Validation & Test accuracy plot – RBF

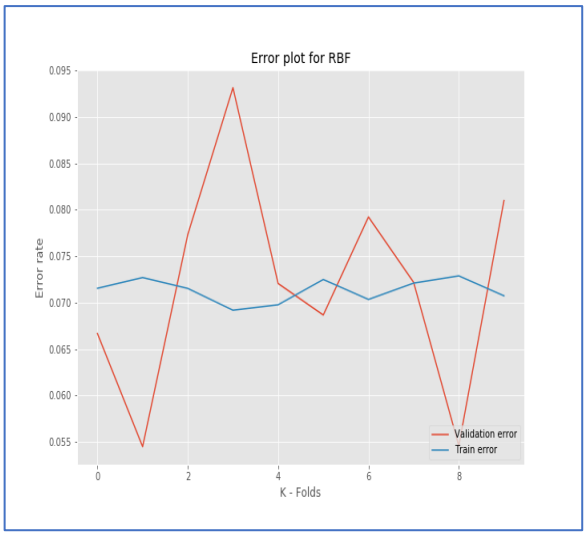

Figure 3.1 Error Plot RBF

Comparison of different SVM model's performance

The below figures show the performance of the three best SVM models in terms of error plots on the 10-fold training dataset. Figure 4.0 depicts the performance of model on validation dataset and it is evident that RBF has the lowest error curve compared to other two models leading to giving more accuracy. Figure 4.1 depicts the similar nature of showing error plot on the trained data. We can see that for RBF it is lowest. Hence, we can confirm that RBF performance on this dataset is best for the chosen parameters.

Decision Tree

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. The data was trained for two types of Decision Tree algorithm namely; Gini Index & Entropy with max_depth parameter varying at values 2, 5, and 10. It was observed that for both the algorithms, the dataset was performing equally well with the test data. So,

the choice made by me was for Entropy model since splitting data based on information gain results in better decision making. Table 2.0 shows the best values for both the models. There is not significant

Figure 4.0 Model Performance on Test data

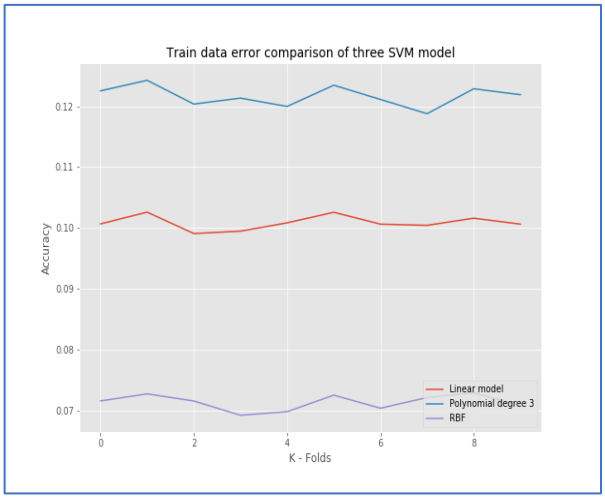

Figure 4.1 Model Performance on Train data

difference between the test accuracy of both the algorithms but choosing Entropy model was a personal choice.

I have tested three pruning values by limiting the max_depth parameter. After training models for these (i.e. 2, 5 and 10) values, it is observed that the model with max_depth as 10 generalizes the data appropriately giving good model accuracy during cross validation. Moreover, if you keep increasing the depth of the tree, the model will start overfitting the data (i.e. high variance and high bias). The Figure 5.0 shows the accuracy plot of train, validation and testing. The training accuracy rises linearly up to 95%,

Dataset accuracy	Entropy (k=10 fold, depth=10)	Gini (k=10 fold, depth=10)
Train	0.9521	0.9421
Validation	0.9210	0.9255
Test	0.9225	0.9266
Confusion matrix	[[1194 63] [126 1055]]	[[1224 33] [146 1035]]

Table 2.0 Accuracy Measures of Decision tree

however, the test data accuracy becomes constant. The difference between train and test accuracy is small and thus, it represents the data well in classifying the categorical variable viz. visible in confusion matrix

AdapBoost

I am using Adaptive boosting algorithm to implement ensemble supervised learning on the dataset. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. For this, I have performed cross validation on the data set with 10 folds. The base classifier used is Decision tree with different values of max_depth to experiment pruning results. After testing for max_depth values 3, 5, and 10; it is observed that the tree pruned to 10 depth is giving promising results by the model. The variation for the same is shown in the Figure 6.0. As you may see that the training accuracy of the model has reached 100%. This can happen because boosting weights

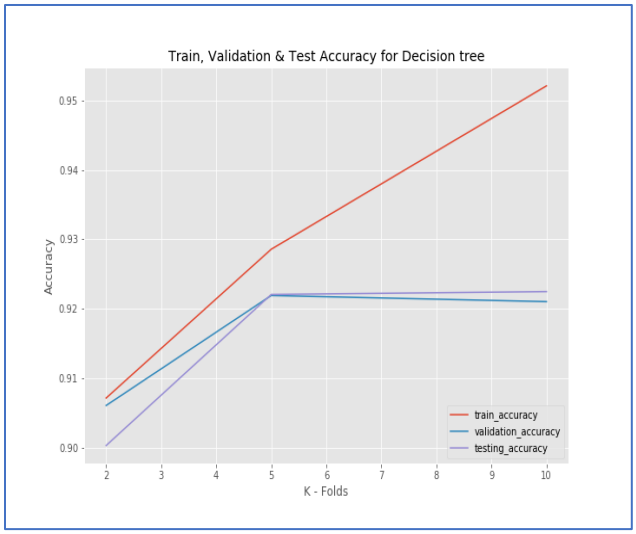

Figure 5.0 Train, Validation & Test accuracy plot – D tree

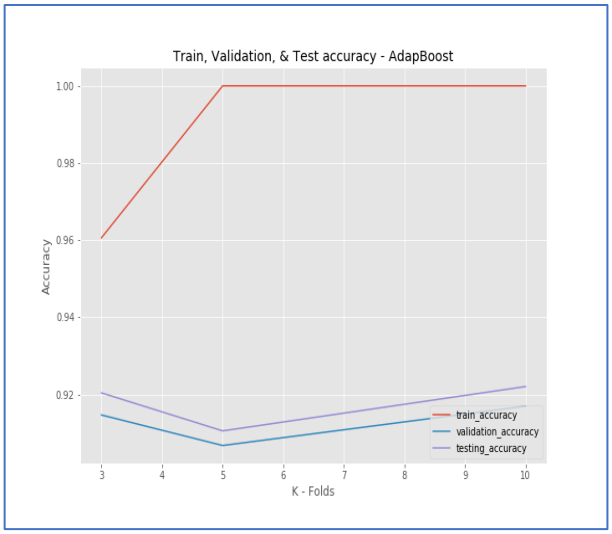

Figure 6.0 Train, Validation & Test accuracy - AdaBoost

the unclassified data more in the next iteration and performs this process until everything is classified correctly. The testing accuracy is 92.21% thus classifying the data aptly. The Confusion matrix result are Confusion Matrix: [[2951 0]
[0 2735]]

Overall Performance Comparison

The figure below shows the accuracy of the different models that gave best results on the test data. As you can figure out, RBF has shown the best accuracy followed by Decision Tree and AdaBoost. For Dataset 2, the best model is RBF Gaussian SVM learner.

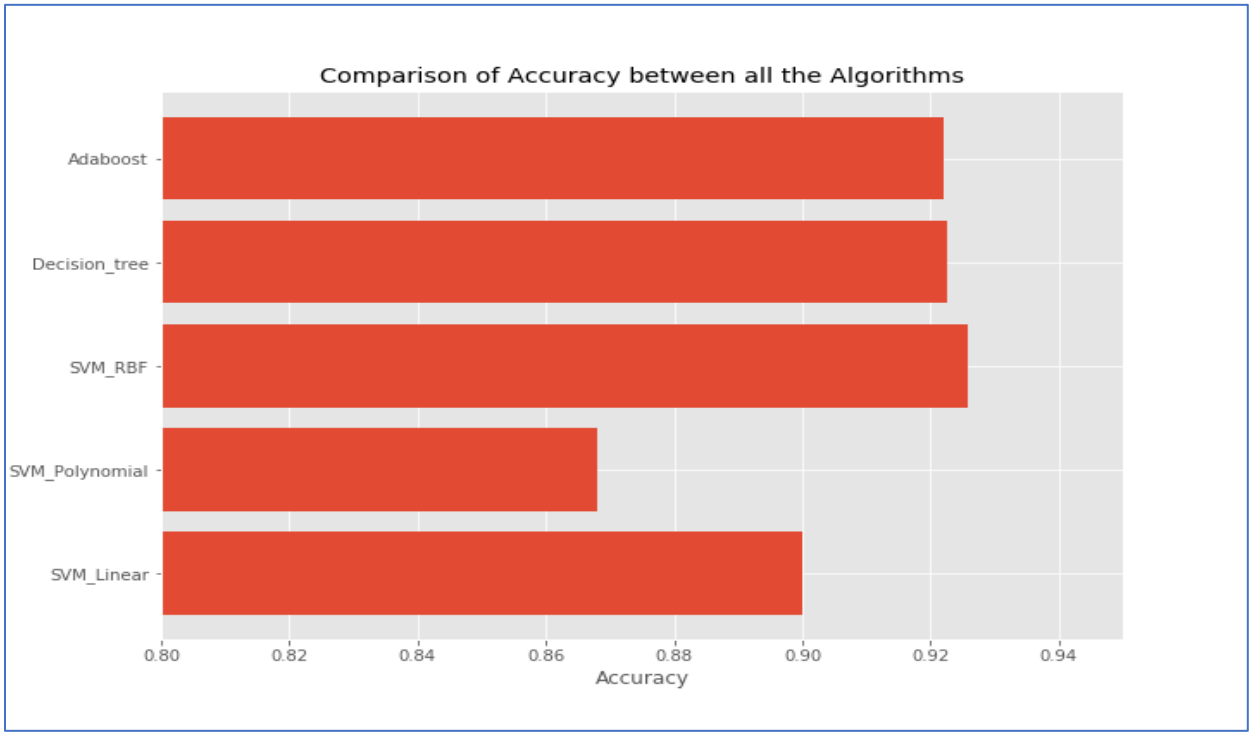

Figure 7.0 Overall Performance plot

Learning Curve – RBF Kernel

The learning curve is plotted for the best performing model i.e. RBF kernel in case of dataset 2. As you can see, the difference between training and cross-validation is not much. They have actually merged and that can also be verified from Table 1.0. With increase in the sample sizes at each split, the error reduces for the training dataset. This shows that the model is able to learn and generalize the data pretty well.

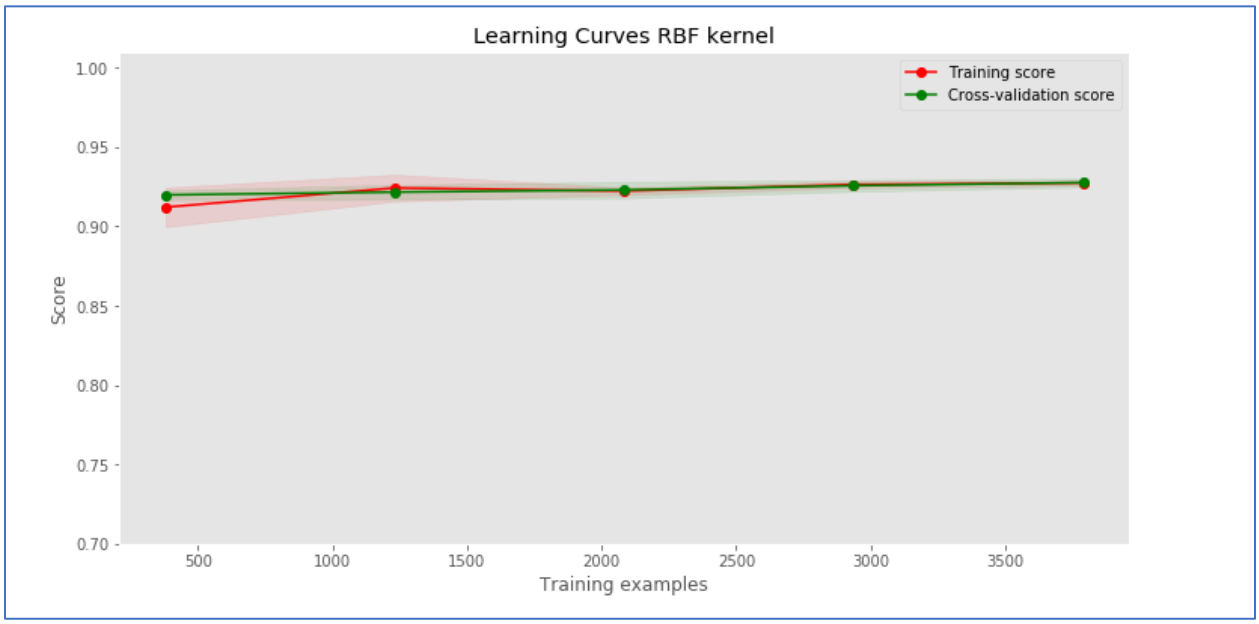

Figure 8.0 Learning curve RBF kernel

Conclusion

We can conclude that for Dataset1; the best accuracy is obtained by Decision tree and that for Dataset 2 it is obtained from the RBF Kernel. This assignment has made me think on datasets from a broader aspect regarding data handling, feature selection and thinking from different point of views. Overall, it was a great practice to work on this report. Both the datasets were very diverse in terms of the information provided and it was really challenging to work on both datasets.

Additional things that might have resulted in better accuracy could be including more than 2 Principle components for training the models would have increased the model accuracy.