**Context:**

This documentation explains results after application of various classification algorithms on the credit card payment data. There is an output feature in this case which can take values of 0 or 1. This column means that a value of 1 indicates that the customer (record) has more chances to pay the bill in the coming weeks. Each of the classification algorithm gives a model with different accuracy and recall scores etc. Main idea is to find the combination which gives the best and consistent results.

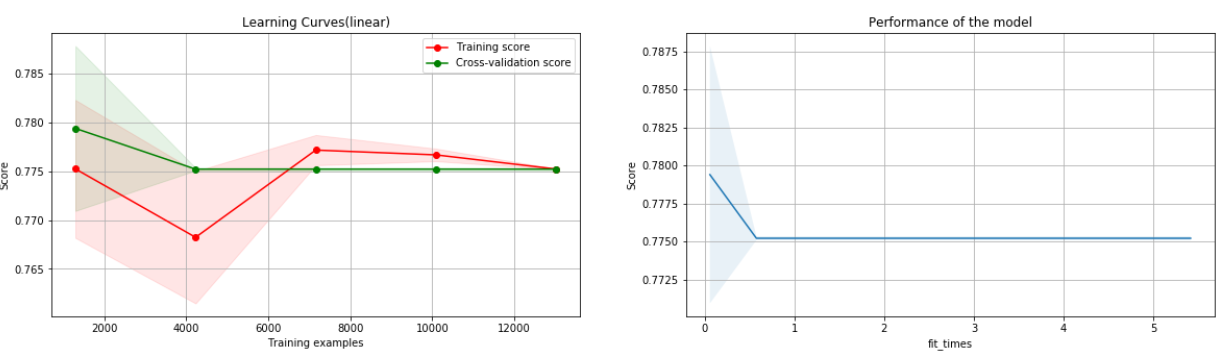
**About the Dataset:**

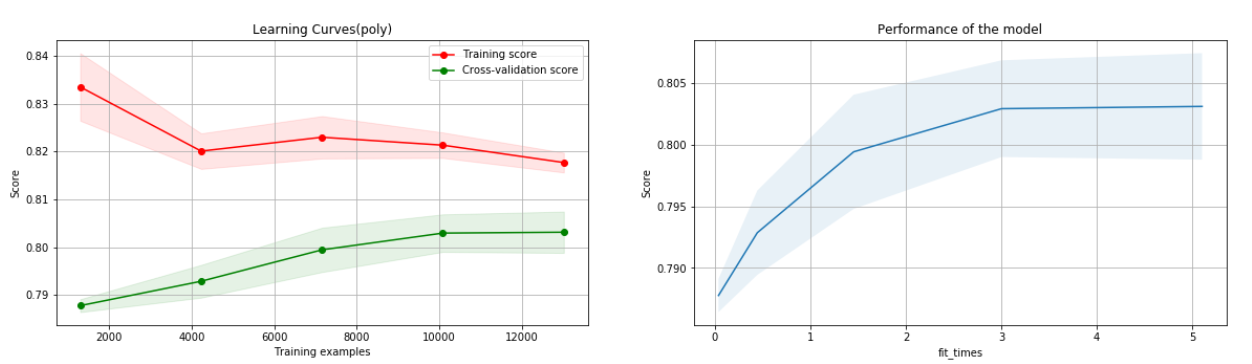
Credit card payment have a total of 30000 records and 25 features. There are 24 input features and most of the features are numeric.

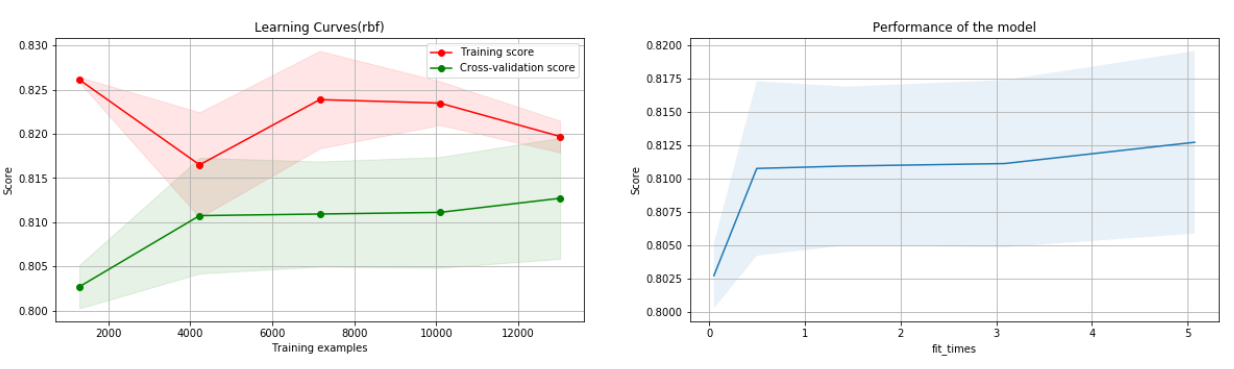
**SVM:**

In this part of the experiment, SVM classification is used on the datasets. SVM from sklearn package is used here. To get an efficient classification set of steps are performed. Cross validation score for each model helps in picking the best model for the data. Learning curves and performance curves gives an idea of how different models work with the dataset. By hyperparameter tuning, each model will have the best parameters to work with given dataset. After hyper parameter tuning, we will have the results from the best models possible out of which we choose our best model based on metrics. We are experimenting with linear, rbf and polynomial kernels and experimentation is performed on 2 datasets

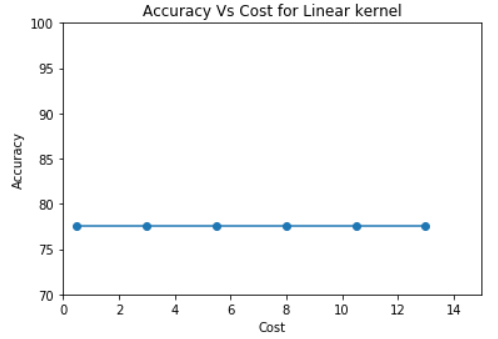
Cross validation scores for SVM with linear, polynomial and rbf kernels are 77.52, 78.75, 81.27 respectively. This indicates that rbf gives the best model with this data. Learning curves below gives us an idea about how model is being trained with train and then fit with test data. In case of linear, there is fair chance of overfitting as learning with test and train data is very close. Polynomial kernel and rbf kernel show some good learning curves in terms of alignment with test and train data as observed in the plots below.



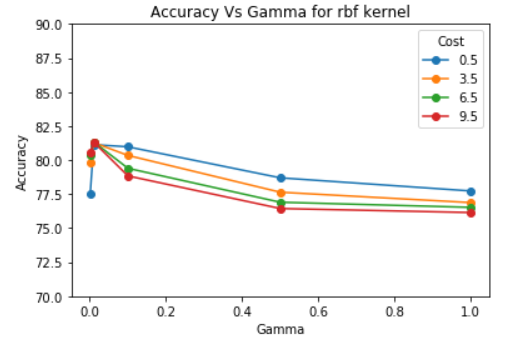
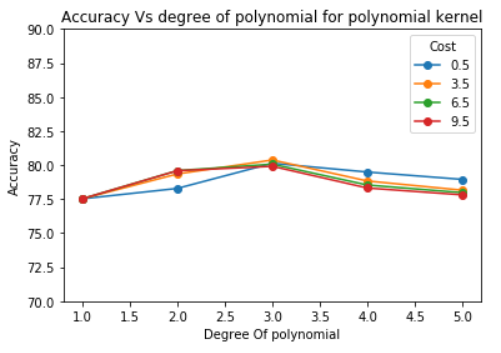




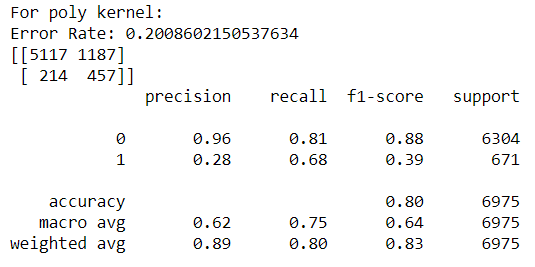
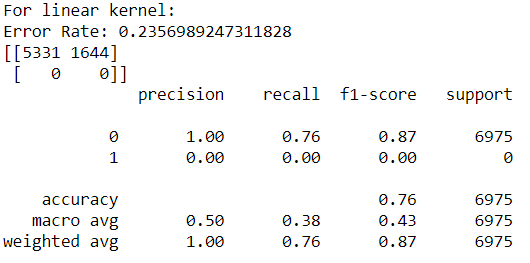
**Hyper parameter tuning:**

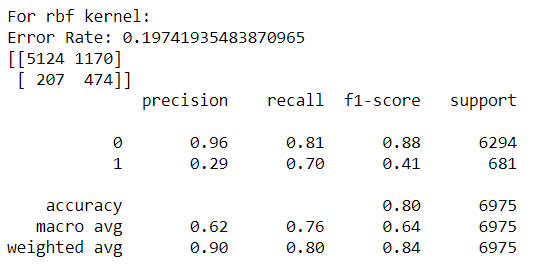


As observed from the graphs, cost is not affecting the accuracy in case of linear kernel. So, we can any cost value and here we work with 1. In case of polynomial kernel, finding the best cost and degree combination is required. From the graph, it is evident that a combination of 3 for degree and 3.5 for cost is giving the best accuracy. For rbf kernel, gamma value and cost combination are required. From results, gamma of 0.1 and cost of 0.5 is giving the best accuracy.



From hyper parameter tuning step, best models are built with the respective parameters. Classification report, confusion matrix and error rate for all the models are as below. After fitting the models with train data and predicting the test data, we verified the test error and train error. In this case, rbf is the best model that can be used as the error rate of rbf is 0.198 whereas error rates of linear and poly kernel models are 0.235 and 0.200 respectively.



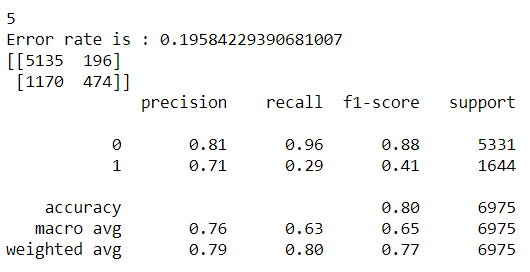
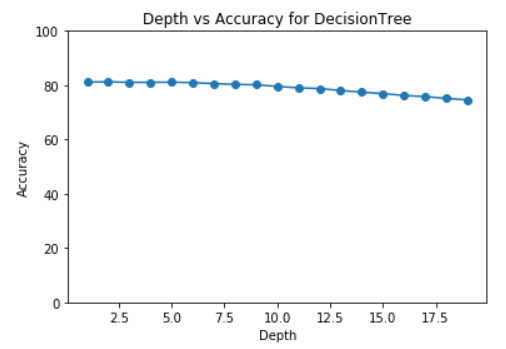


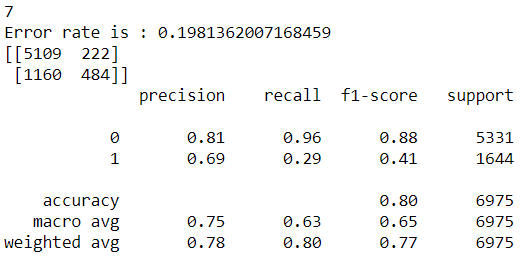
Best SVM model in this case is with parameters as rbf kernel, cost of 0.5 and gamma of 0.1.

**Decision Tree:**

To implement decision tree, sklearn package is used. Hyper parameter tuning is done to find the best values for parameters like depth, features and alpha. Decision tree is experimented with pruning to see if the accuracy is improved. Then model is also experimented with gini and entrophy criterion and compare the metrics.

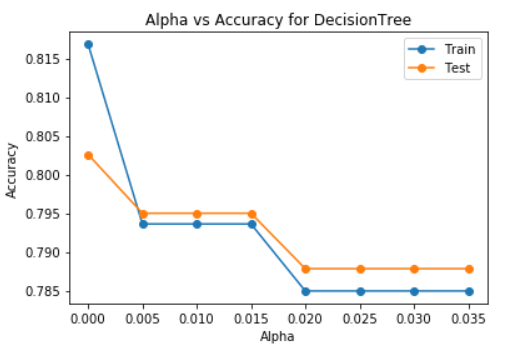
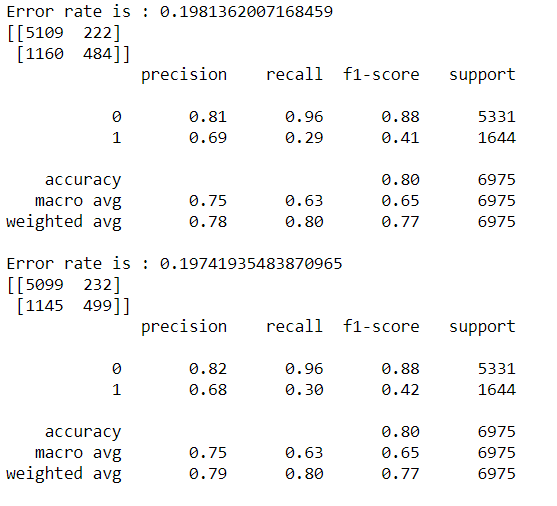
After experimenting with different depth values to find the best one, depth of 5 gave good results compared with other depth values.





Model is experimented with different feature values. Out of the various values, feature value of 7 gave the best results.

Experimenting with two different criterions didn’t improve the accuracy and other metrics. Error rate has got down to 0.197 when the criterion is changed to entropy. That can also be inferred from the metrics below. 0.198 is the error rate when gini is the index used. Till now we found the best parameter values for the depth, features and criterion. Experimentation with pruning gave the results which are shown in the plot below. From the plot, we see more accuracy for test than train when we experiment with pruning. So pruning is not appropriate with this data and the model.

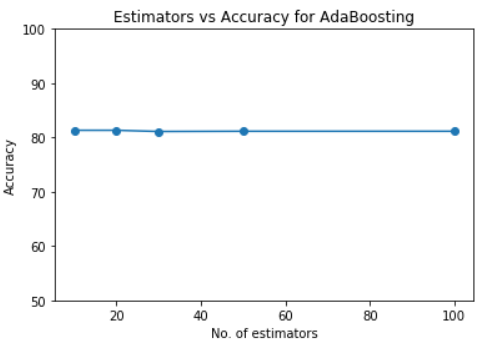
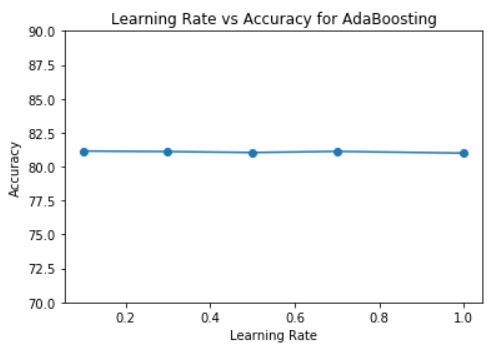


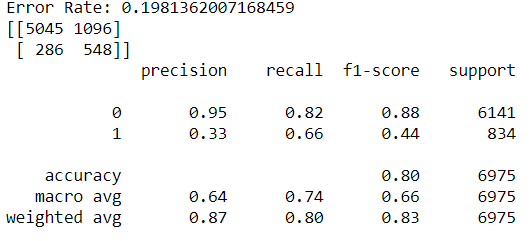
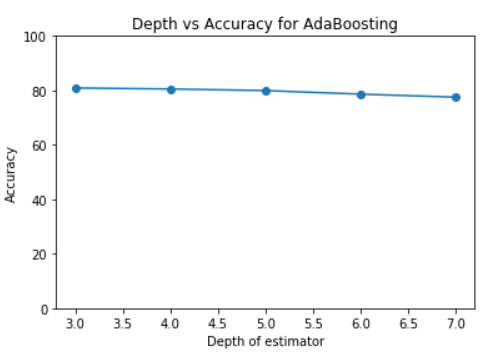
The best decision tree model after hyperparameter tuning has an error rate of 0.197.

**Boosting:**

Boosting techniques are used with decision tree in this part of the experiment. Hyper parameter tuning is carried out to find the best values for the learning rate, estimators, and depth. We carry out these experiments and build the models.

As observed from the plots, learning rate is not affecting the accuracy at any value. Cost value chosen here is 0.1. Number of estimators is affecting the accuracy but after 50 features, these is not much change in the accuracy. So, features are 50. When experimenting with depth, a depth of 4 can be considered the best choice.





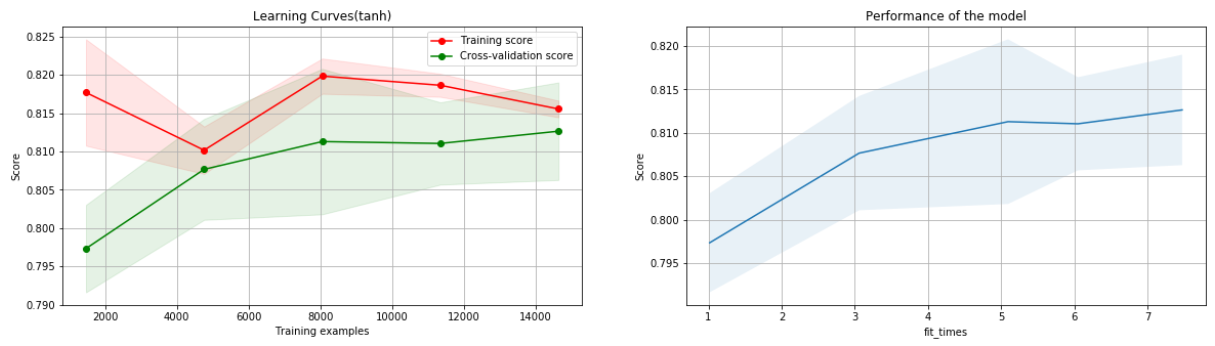
Final boosting model, with the parameters as above, gives an accuracy of 0.87 and also an error rate of 0.198.

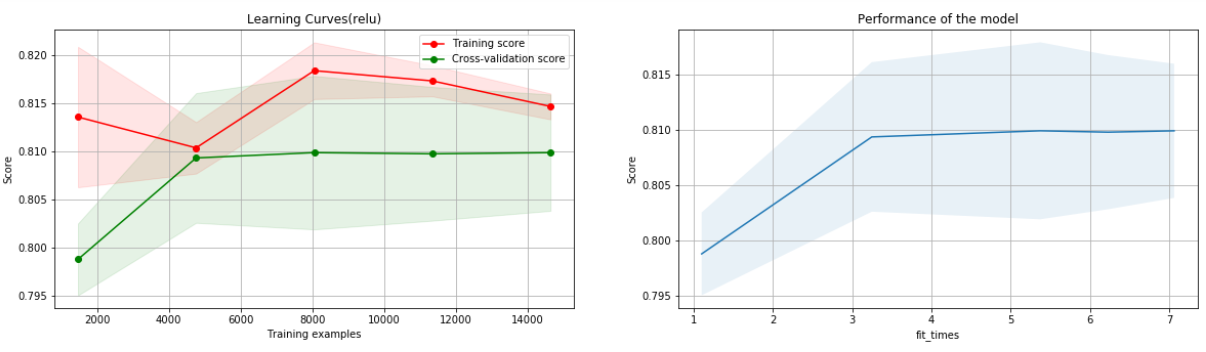
**ANN:**

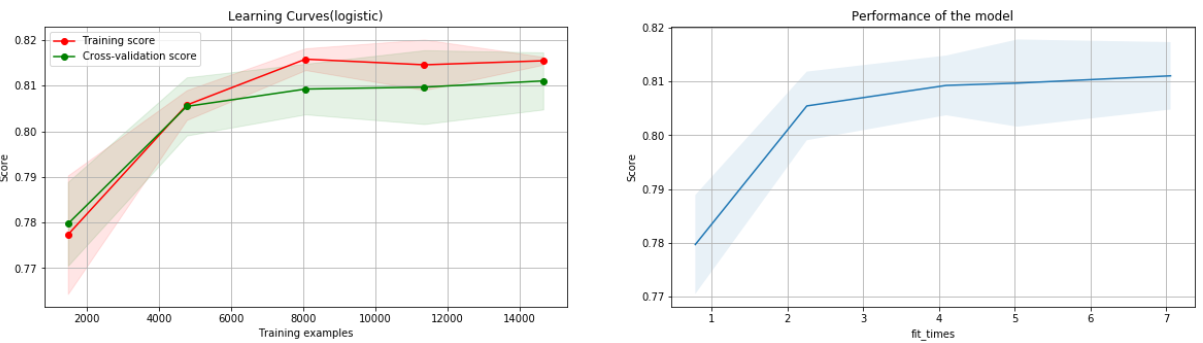
To implement ANN, first step is to know the learning and performance of models with different activation functions. After knowing the level of learning, confirm the inferences with the cross-validation scores. To find the best model for the given dataset, A classifier must be constructed. To do that we experiment with different values for hidden layers, number of nodes, activation function and solver type. With different combinations of these parameters, we have different accuracy and error rates. Required Classifier is constructed from the results of the experimentation and that is used to fit a model which is used for classification.

Learning curves of ANN classifier with different values of activation function (tanh, relu, logistic) are shown below. ‘Score’ vs ‘training Examples’ graphs shows the level of learning the model is going through. Logistic activation function has the best cross validation scores comparatively. Also, the performance of the model for 3 activation functions reveals that activation function have good performance. So, we will choose logistic as activation and then experiment with hidden layers.

**Learning Curves:**

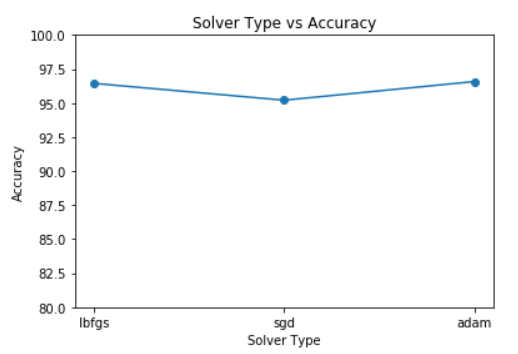
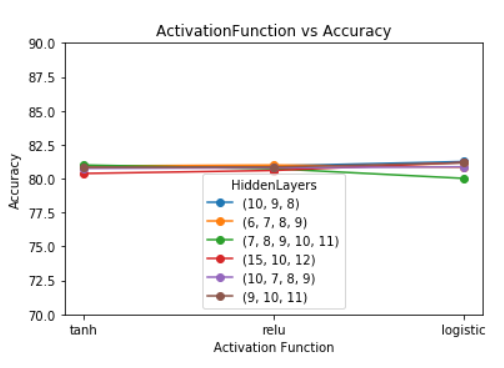




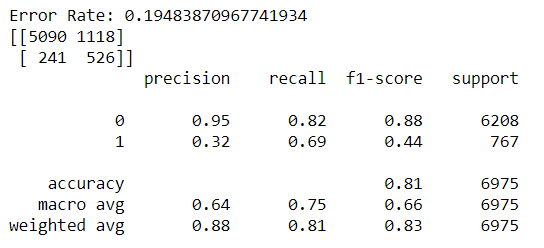


**Hyper Parameter tuning:**

After finding the best activation for the given case we need to experiment with hidden layers and number of nodes. After trying various combinations relu with (10, 9, 8) gave the best results. It is shown in the Activation function vs Accuracy plot. After fixing on activation function and hidden layers combination we experiment with solver type and find the best possible solver for this case. From the second graph below, accuracy is similar between lbfgs and adam. We’ll choose adam because of good accuracy rates.



After hyper parameter tuning, we are left with the best possible combination for this case. Logistic activation function, (10, 9, 8) as 3 hidden layers with respective numbers as nodes and adam solver gives the best possible model. Train and fit the classifier with training data set and predict the test dataset values. Comparison of the predicted and actual results gives the error rate, accuracy and other metrics which can assess the model.



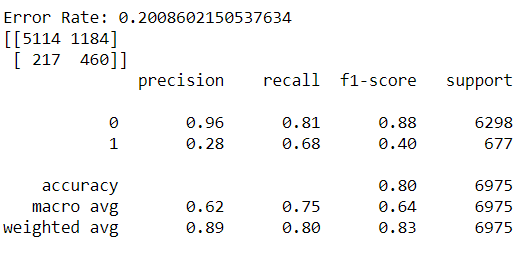
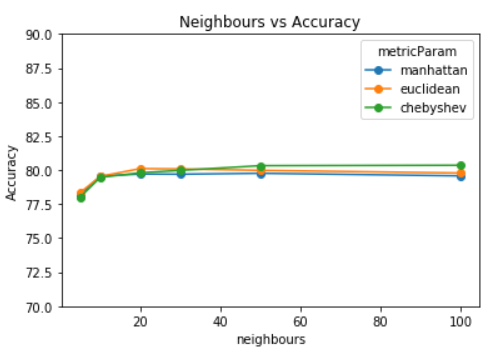
ANN could fit a model with accuracy of 88 and f1 score of 83. Error rate of the model is 0.194.

**KNN:**

Cross-validation scores gives an idea of how the accuracy of the model varies with different models. In this case we initially check the cross-validation scores by building different models with varying neighbors’ number. Hyper parameter tuning is done and required model is extracted. To do so, experimentation is conducted with varying combinations of number of neighbors and distance metrics. Each combination is unique with at least one different value of parameters. After hyper parameter tuning, we are left with the classifier and the best possible parameter values for this dataset.

Cross validation scores of the KNN classifier with neighbor values of 5, 10, 20 are 78.35208679093128, 79.5380548453689, 80.10947780769537. Number of neighbors affect the model. Hence, we experiment with different values of number of neighbors and with the distance metric to be chosen. Results of the experiment are as

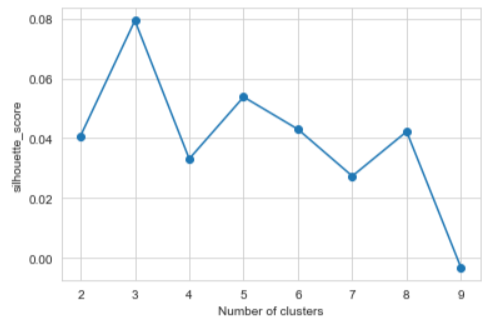
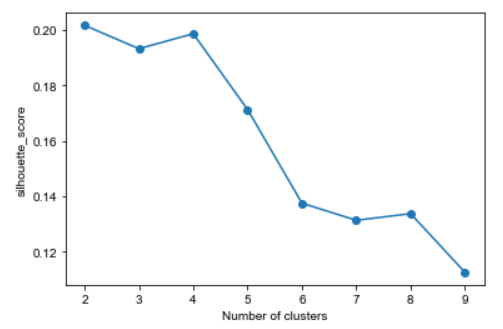
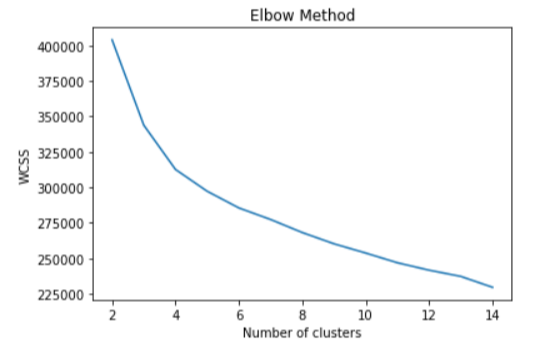
below. From graph, Chebyshev distance is the best choice comparatively. Though accuracy scores are almost similar, there is a significant difference in values for different metrics. Comparing the number of neighbors for the metric Chebyshev, accuracy scores increased with neighbors but there is very less difference between the models with neighbor values of 50 and 100. Considering computational overhead, 50 is a good choice here as doubling the neighbors didn’t have significant effect on the scores.



With number of neighbor value of 50 and distance metric as Chebyshev (because of its good scores) we build a KNN classifier which is the best possible result concluded. Train and fit the KNN classifier with train set and then predict on the test Dataset. Final model has a recall score of .80 and error rate of 0.20.

**KM and EM:**

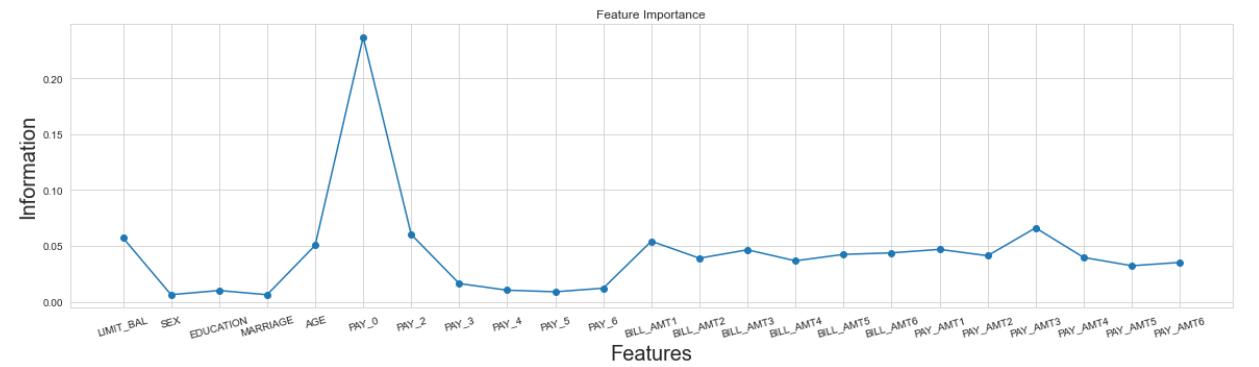
K means (KM) and expectation maximization (EM) are applied on both the data sets. In either of the cases, to find the number of clusters to be chosen, elbow method and silhouette scores are used. From the plots, cluster value with highest silhouette scores are to be chosen. From graph2, K means have a cluster value of 2 and for EM, cluster value of 3 can be chosen based on 3rd plot below.



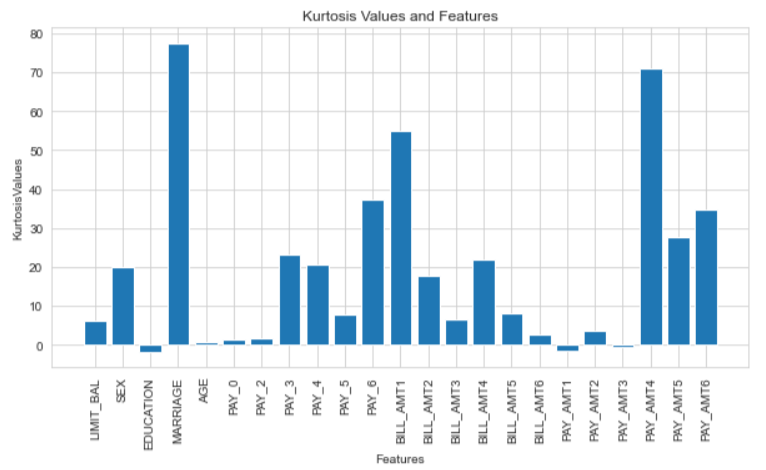
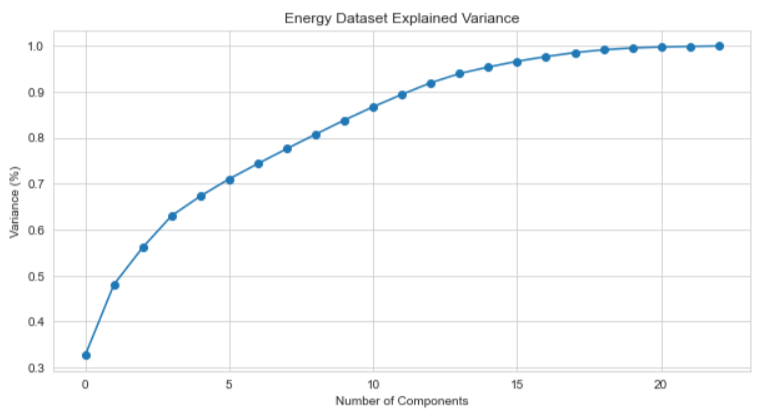
**Dimensionality Reduction:**

Dimensionality reduction algorithms are applied on the data to extract the best features that explains majority about the data. Decision Tree, PCA, ICA and random optimizations (RO) are the algorithms used. Decision tree gives the best features without any transformations. PCA, ICA perform their own feature transformation techniques and gives the best features. PCA transforms features so that variance can be explained. ICA makes new features independent. RO selects the features randomly enabling the combinations which might not be possible with traditional methods.

**Decision Tree:** Maximum informationis given by PAY\_0 variable in this case. We consider the variables which give at least 5% information about the data. In this case, we have 6 features which explains at least 5% about the data.



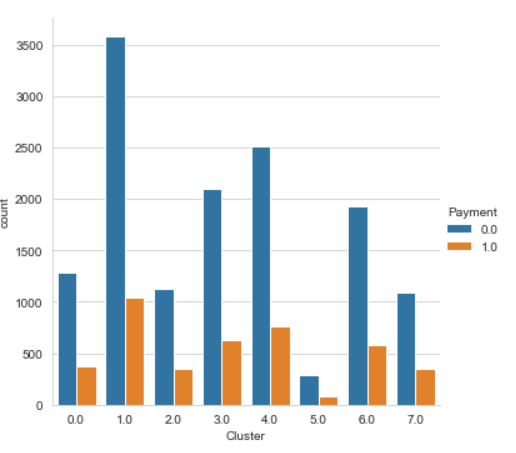
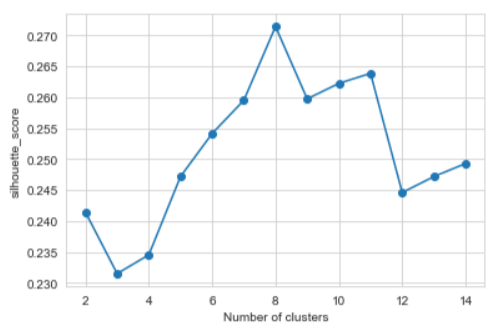
**PCA and ICA:** After PCA, number of components must be chosen such that maximum variance can be explained. In this case the number of components to be chosen are 15. For ICA, kurtosis values help in finding the best features to work with. Excluding the features with negative kurtosis values gives the best combination.



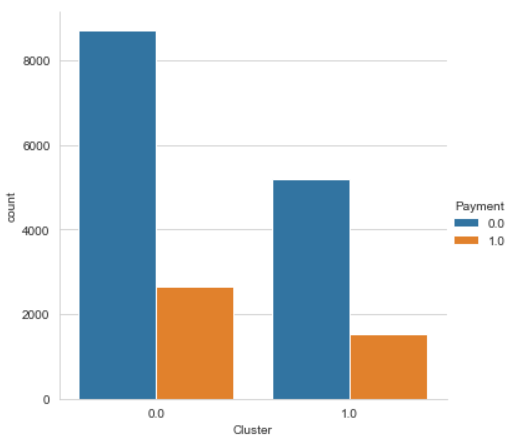
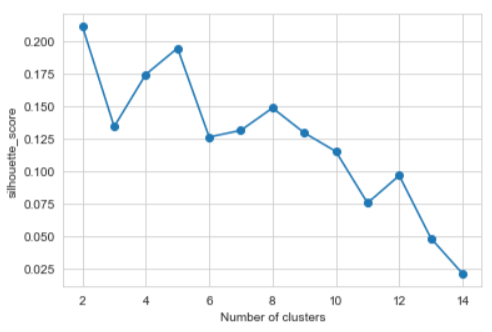
In case of RO, 5 random components are chosen in this case. Results of RO help in better understanding of the data.

**Clustering and Dimensionality Reduction**

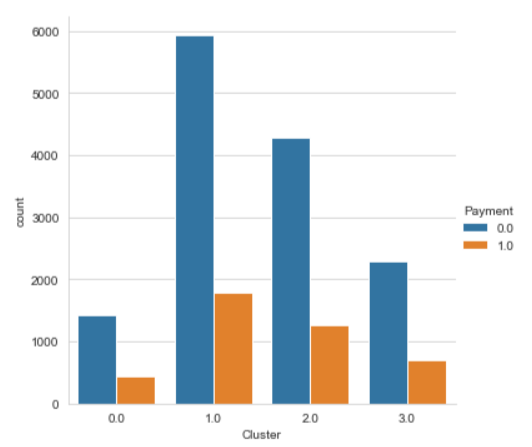
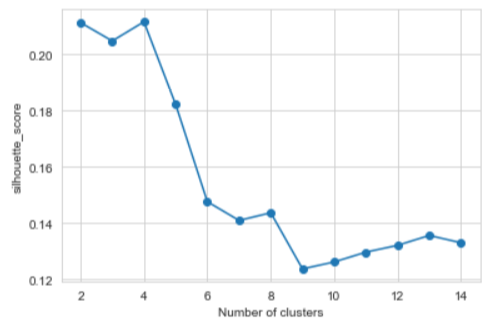
Applying clustering to data after dimensionality reduction gave different results. Finding the clusters for each case and plotted the clusters and output variable classification. Some cases did not give the same number of clusters as the target variables. In such cases the pattern of how the target variable is distributed tells details about data.

**DT K Means:** Cluster value of 8 gave the best results in this case. Output variable has only two values and those are distributed across these clusters. Cluster with 1 and 4 have more 0’s of the output variable.

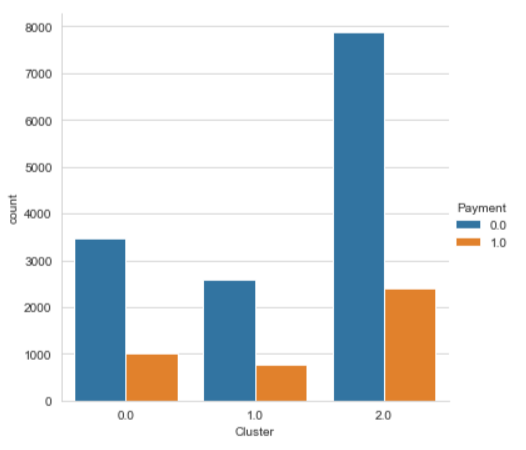
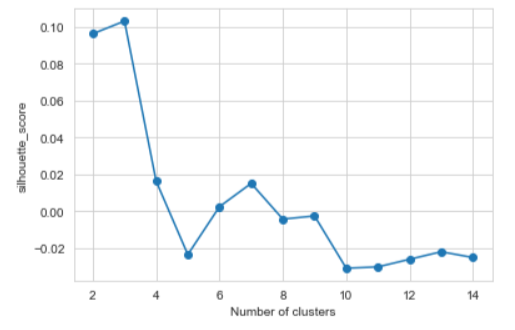
**DT EM:** Cluster value of 2 gave the best silhouette score. Plotting the clusters against the output variable showed that both clusters 0 and 1 have more 0’s which isn’t an ideal case.



**PCA KM:** Cluster value of 4 gave the best silhouette score. Cluster 1 has a greater number of 1’s and 0’s of the target variable.

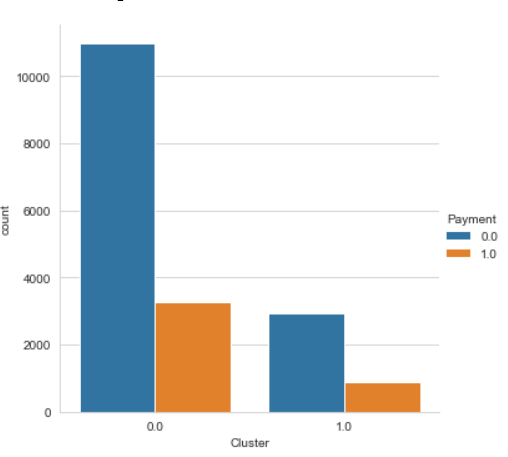
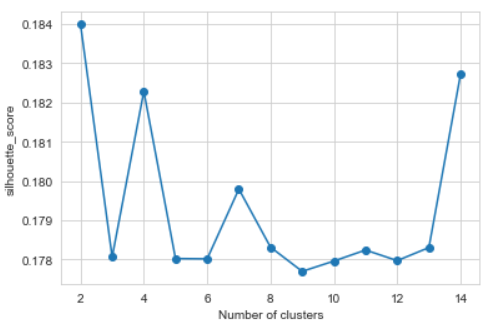


**PCA EM:** Cluster size of 3 is ideal in case of PCA EM. Cluster 2 have most of the 0’s. The distribution across the clusters is as shown.

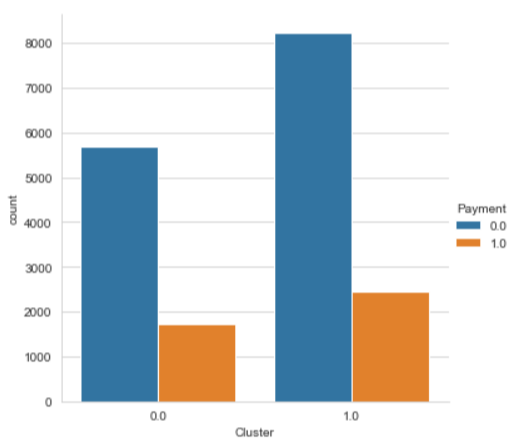
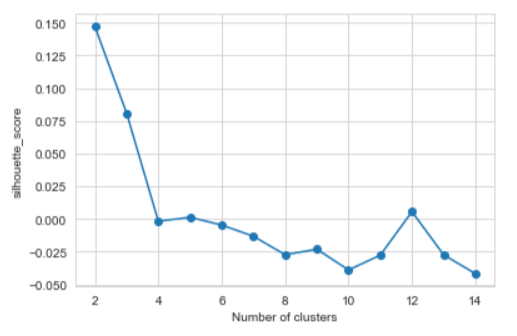


**ICA:** Apply K means and EM methods on the data after ICA. For K means, cluster size of 2 gave the best silhouette scores. Distribution shows that majority of 0’s is part of Cluster 0. In case of EM, ideal cluster size is 2 but the distribution has more 0’s as part of cluster 1.

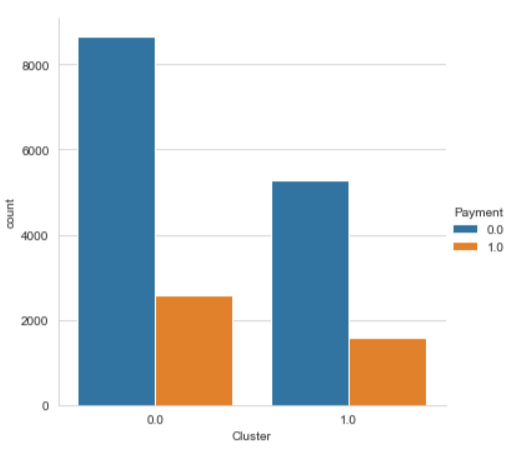
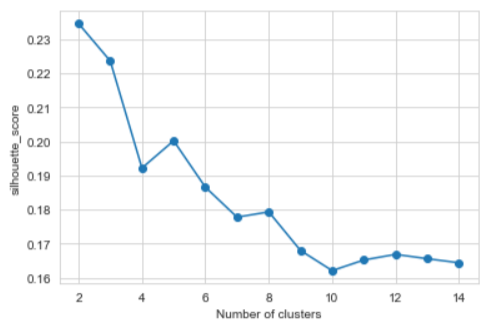
**KM:**



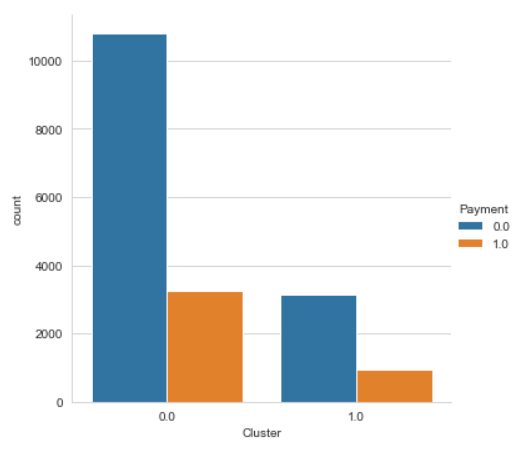
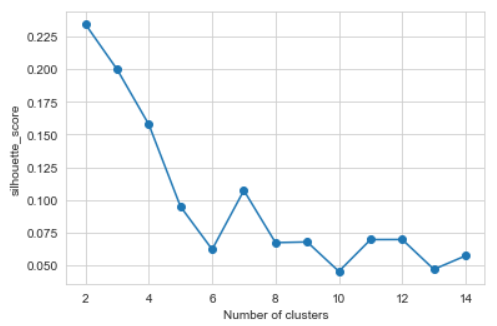
**EM:**



**RO KM:** Apply K means on the data obtained after random optimization. Finding the cluster size with higher silhouette score is the optimal. In this case, it is 2. Distribution is as shown below.



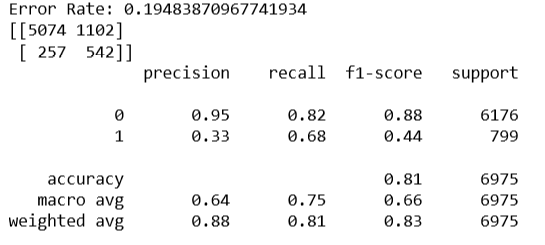
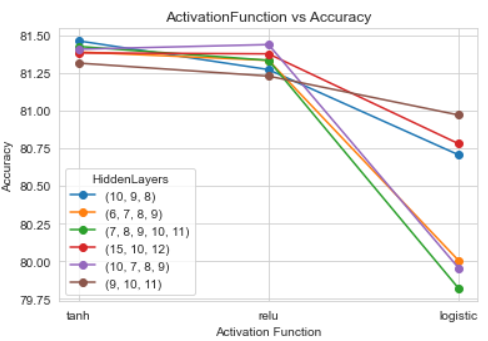
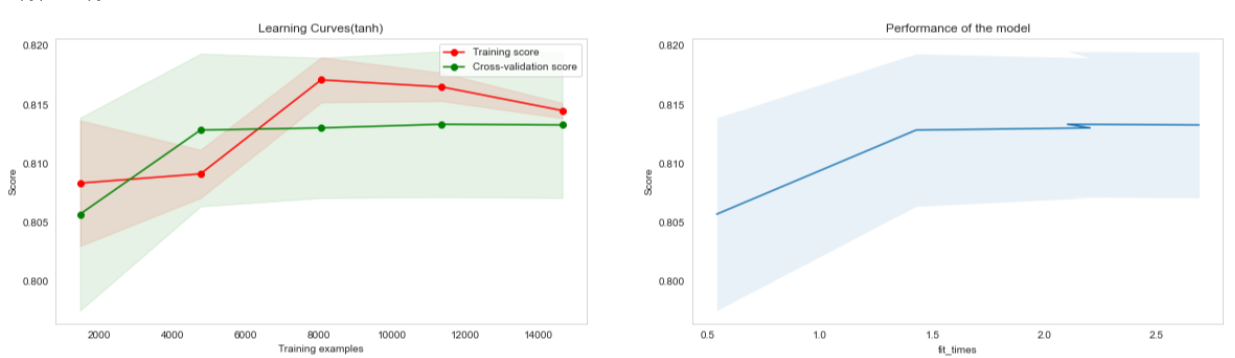
**RO EM:** Apply Expectation maximization on the RO data and best cluster size is 2 in this case. Distribution is as below.



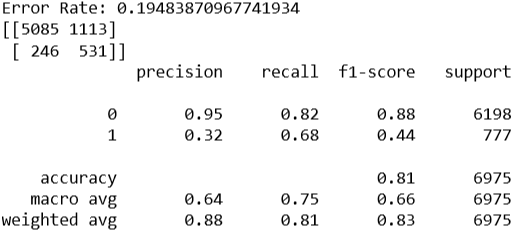
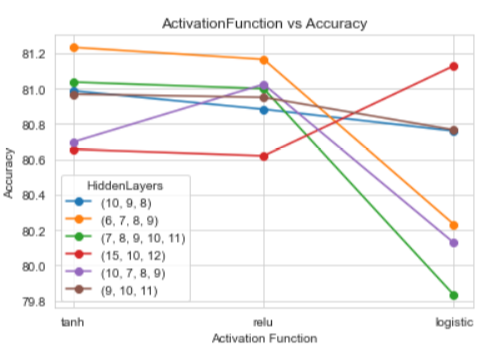
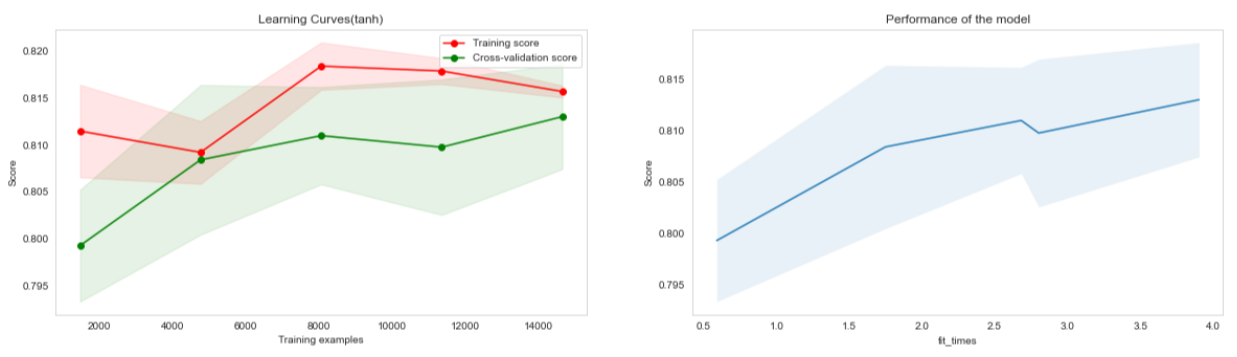
**Neural networks and Dimensionality Reduction:**

This part deals with application of neural network algorithm on the data after dimensionality reduction algorithms. For each of the datasets, ANN is implemented on decision tree data, PCA data and ICA data. Learning curve of the best activation function is included. Hyper parameter tuning is done to find the best parameters in every case. Model is fit and error rate along with confusion matrix and classification report is provided for every case.

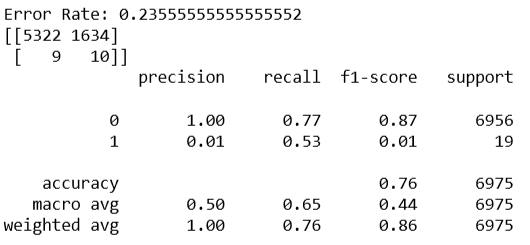
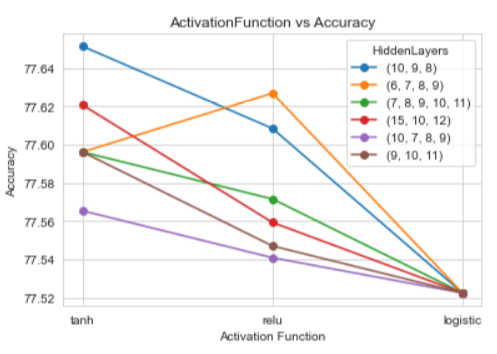
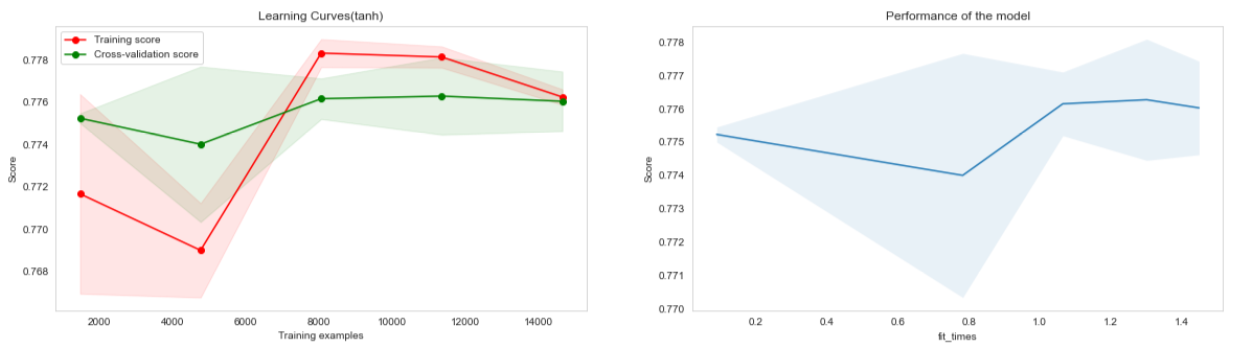
**ANN and DT:** Tanh is the best parameter for the activation in this case. An Error rate of 0.19 is observed when ANN is applied on decision tree data. F1 score of 0’s is 0.88 and for 1’s the F1 score is 0.44.



**ANN PCA:** Tanh is the best parameter for the activation in this case. Number of hidden layers is 4. An Error rate of 0.194 is observed when ANN is applied on PCA data. F1 score of 0’s is 0.88 and for 1’s the F1 score is 0.44.



**ANN ICA:** Tanh is the best parameter for the activation in this case. An Error rate of 0.23 is observed when ANN is applied on ICA data. F1 score of 0’s is 0.87 and for 1’s the F1 score is 0.01. This combination is useful when finding all 0’s is important.



**Clustering and neural network:**

Data consisting of only clustering results as features and class label as the output is taken and neural network algorithm is implemented for either of the data sets. Finding the best combination of activation function, learner, number of hidden layers and number of nodes in each hidden layer is done as part of hyper parameter tuning. Once the model is trained with train data, predictions are done on the test data. Classification report, confusion matrix and error rate for each model are included.

Tanh is the best activation function for this dataset when the clustering results and neural networks are combined. An error rate of 0.22 is observed with this model. This model classified all the values as 0 leading to a precision of 100 but and error score of 0.229 and f1 score for 0 as 0.87.

