**Assignment 2**

**Introduction:**

The objective of this project is to predict high intensive GPU loads and telecom customer churn using the selected two dataset. We will be using SVM, decision tree and gradient boosted tree algorithms to find the best model to perform each classification task. We will be using learning curves to fine tune parameters like kernel for SVM, depth for decision tree and number of estimators for gradient boosted tree.

**Part 1**

**Telecom Customer Churn Prediction**

**Dataset:**

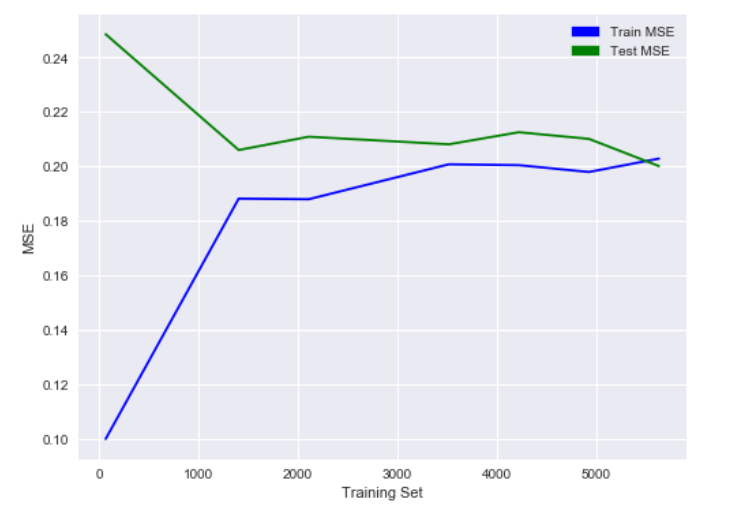
The dataset we will be using for this project has a total of 21 features and 7043 records. We will be dropping customerId and PaymentMethod from the dataset and will be using churn as our target variable. Partner, Dependents, PhoneService, PaperlessBilling, Churn, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies variables are converted to binary. Dummy variables have been added to convert InternetService, Contract and Gender to categorical variables. After the above changes we have 20 features we will be using as our independent variables to predict out target variable which is churn.

**Experiment 1:**

In this experiment we will be analyzing the performance of three different SVM models using learning curves and cross fold validation. We will be choosing the best model for predicting the customer churn based of the bias and variance seen in the learning curves and the average accuracy on the validation set of each model. The SVM kernels we will be using for this experiment are linear, sigmoid and RBF.

**Linear Kernel performance:**

To plot the learning curve, we will be running the SVM model for different sizes of the training set. Running the model for training size 70, 1408, 2112, 3521, 4225, 4930 and 5634, we get the following learning curve.



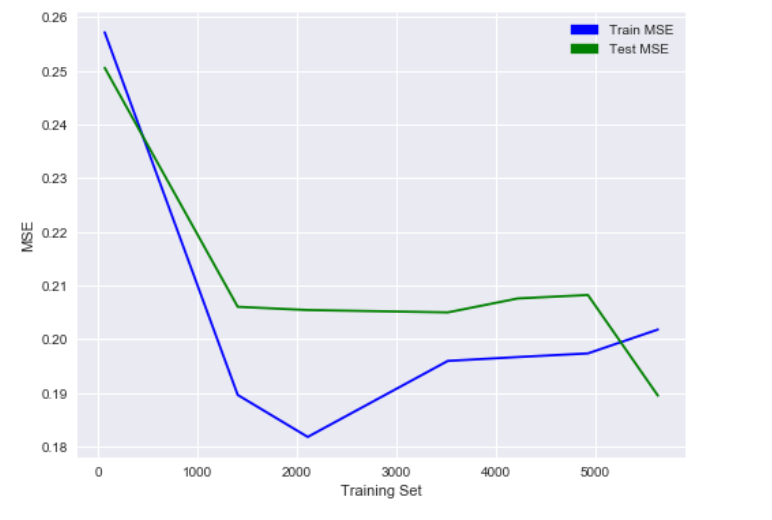
We can see from the graph that the variance becomes low when training size is above 1500. The bias seems a bit high from the sudden increase in training error as the training set size moves towards 1500. But we need to see how this fares in comparison with the other 2 kernels.

Next, we run cross fold validation on 90% of the training set with 5 folds and check the average accuracy of our validation sets. This will give use a good idea of how well this model will perform on the churn dataset as all the datapoints will be used in training and test atleast once. The average accuracy of this test using a linear kernel SVM is given below:

|  |  |
| --- | --- |
| Mean Validation Set accuracy | 79.36% |

**Sigmoid Kernel performance:**

Repeating the above experiment with a sigmoid kernel SVM gives the below learning curve:



From the above curve we can see that the Bias reduces as the training set size reduces. Therefore, this kernel seems to generalize well for this dataset only when dataset size is bigger than 2000.

The average validation set accuracy of this model using cross validations with 5 folds and 90% of the data is:

|  |  |
| --- | --- |
| Mean Validation Set accuracy | 79.73% |

Although this average accuracy of this model is high, we can see from the learning curve that this model doesn’t generalize well when bias is high.

**RBF Kernel performance:**

The RBF kernel gives the below learning curve:



From the curve we can see that the performance of the RBF kernel is similar to the linear kernel. The Bias is high as the training set size increase, but the variance is between train and test error is low for training set size above 1500.

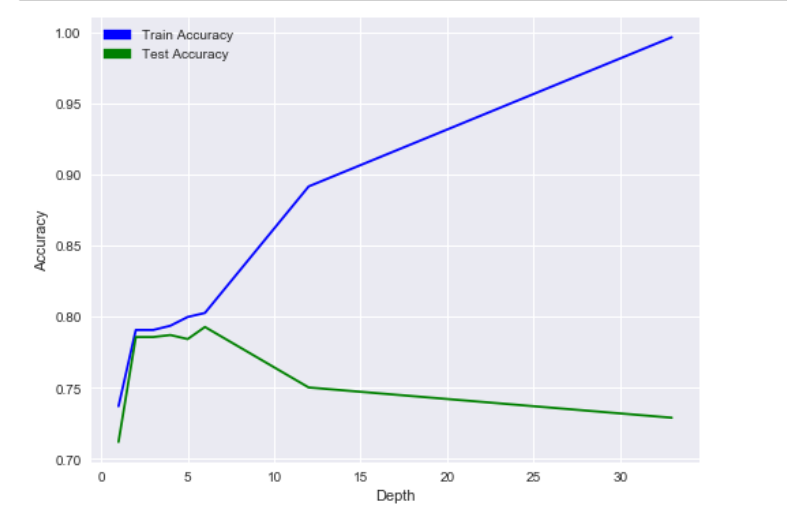
Since the performance of this model is similar to linear kernel, we can compare the average accuracy of the two models to see which model generalizes better for the give dataset.

|  |  |
| --- | --- |
| Mean Validation Set accuracy | 79.36% |

The accuracy of these models also seems to be the same when rounding to 2 digits. Therefore, we can say that both the linear and RBF kernels give us good models with 79.36% accuracy for this dataset.

**Experiment 2:**

In this experiment we will be predicting the customer churn using a decision tree. We will be running the models for different depths and compare the training and test accuracy to select the best depth of the tree. Plotting the test and train accuracy against depth we get the following plot.



From the plot we can see that the variance is low at depths below 7 and as the depth increases, we can see that the decision tree is overfitting with high variance and high bias. So, the ideal depth for the tree should be below 7.

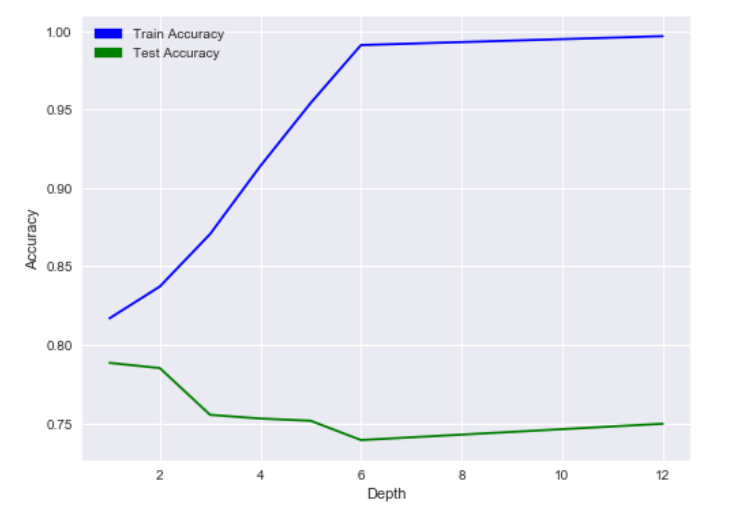
Running the cross-fold validation with the depth from 1-7 we can see which model is the best by checking the average validation accuracy. Running cross validation with 5 folds we get the below average accuracies:

|  |  |
| --- | --- |
| Max Depth | Accuracy |
| 1 | 74.06% |
| 2 | 79.27% |
| 3 | 79.27% |
| 4 | 79.43% |
| 5 | 79.11% |
| 6 | 78.03% |
| 7 | 77.61% |

From the above table we can see that the best depth parameter for the decision tree is 4.

**Experiment 3:**

In this experiment we will be predicting the customer churn using a gradient boosted tree. We will be running the models for different depths and number of estimators and compare the training and test accuracy to select the best parameters. Plotting the test and train accuracy against depth we get the following graph.

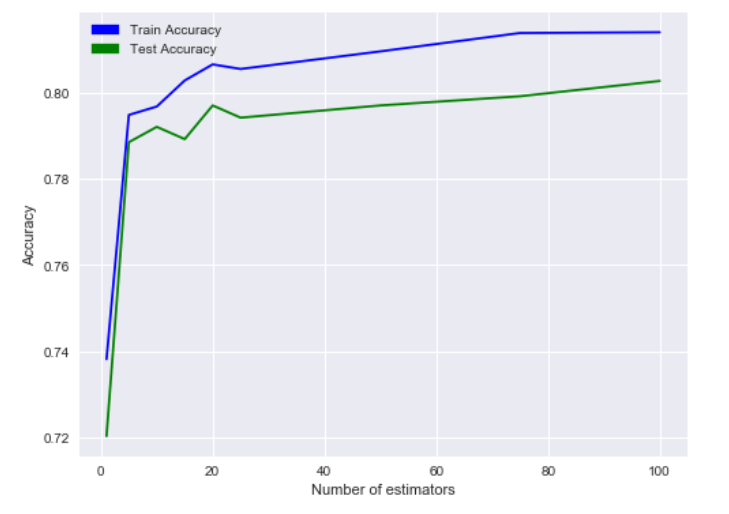


From the plot we can see that the best value for depth is 1 as the bias and variance are very low. As the depth increases the model starts to overfit and has high bias and variance. We can confirm this by running cross validation with 5 folds and taking the average accuracy for depths 1,2,5 and 7.

|  |  |
| --- | --- |
| Max Depth | Accuracy |
| 1 | 80.32% |
| 2 | 78.86% |
| 5 | 76.25% |
| 7 | 76.21% |

From the results of the cross validation run we can see that we get the best results for depth 1.

Plotting the test and train accuracy against number of estimators:



From the plot we can see that ideal number of estimators is around 50. As the variance is lowest at this point. We can also see that the number of estimators has only a small influence on the bias and therefore the model doesn’t overfit at high number of estimators. We can verify our pick for the number of estimators by running cross validation with 5 folds and number of estimators as 1, 5, 10, 15, 20, 50, 75 and 100.

From the above results we can choose 50 as our ideal number of estimators.

**Conclusion:**

In order to select the best algorithm for predicting the telecom customer churn we need to compare the accuracy of the best models from each algorithm.

|  |  |
| --- | --- |
| Algorithm | Accuracy |
| SVM with linear kernel | 79.36% |
| SVM with sigmoid kernel | 79.36% |
| Decision tree | 79.43% |
| Gradient boosted tree | 80.32% |

From the above results we can see that the gradient boosted tree performs well for this classification. We can also be sure that this model will generalize well for the dataset as the above results are from cross validation runs.

**Part 2**

**GPU Average Runtime Prediction**

**Dataset:**

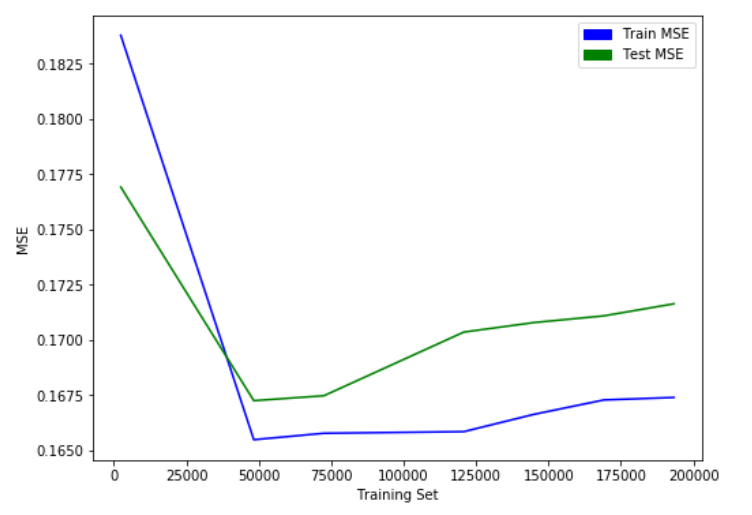
The dataset we will be using for this project has a total of 18 features and 241600 records. We will be taking the average of 4 of these features Run1, Run2, Run3 and Run4 as AverageRun and will be using the remaining 14 for our prediction. For the logistic problem we will be predicting if the average GPU run time is greater than the median. This splits our data almost equally with 50% of ones and 50% zeroes.

**Experiment 1:**

In this experiment we will be analyzing the performance of three different SVM models using learning curves and cross fold validation. We will be choosing the best model for predicting the high intensive GPU loads based of the bias and variance seen in the learning curves and the average accuracy on the validation set of each model. The SVM kernels we will be using for this experiment are linear, sigmoid and RBF.

**Linear Kernel performance:**

To plot the learning curve, we will be running the SVM model for different sizes of the training set. Running the model for different training size we get the following learning curve.



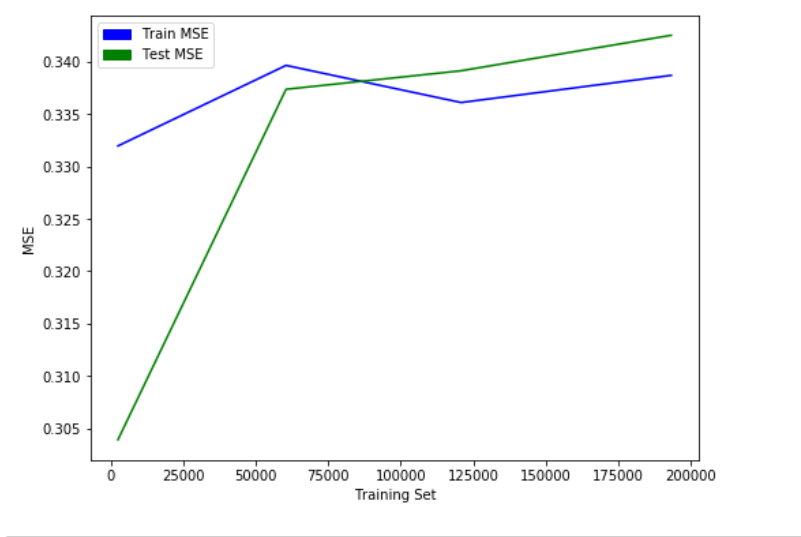
We can see from the graph that the variance and bias becomes low when training size is around 50000 but keeps increasing as the training set size increase beyond 75000. We can also see that the bias is very high when the dataset is less than 50000 causing a lot of underfitting.

Next, we run cross fold validation on 90% of the training set with 5 folds and check the average accuracy of our validation sets. This will give use a good idea of how well this model will performs on the dataset as all the datapoints will be used in training and test atleast once. The average accuracy of this test using a linear kernel SVM is given below:

|  |  |
| --- | --- |
| Mean Validation Set accuracy | 83.21% |

**Sigmoid Kernel performance:**

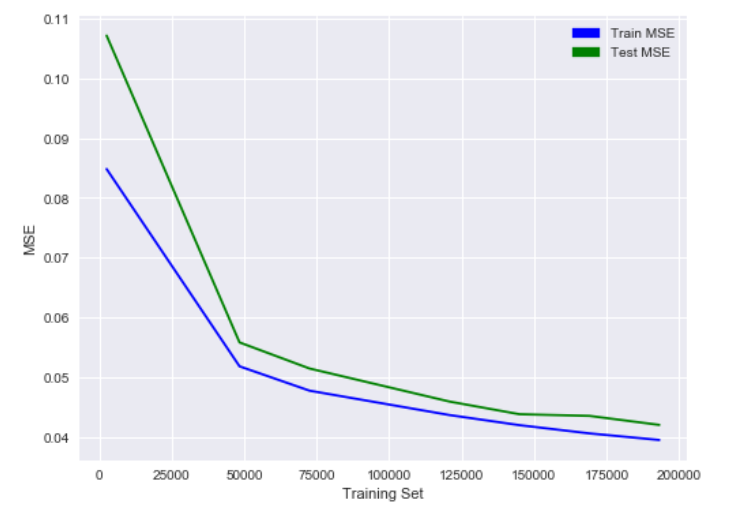
The sigmoid kernel gives the below learning curve:



From the curve we can see that the sigmoid kernel doesn’t generalize well for this dataset. The model seems to only work when training set size is above 90,000. Before this the testing error seems to suggest that the model works better on the test set than on the training set. This suggests that this model is not a good fit for this dataset.

**RBF Kernel performance:**

The RBF kernel gives the below learning curve:



From the curve we can see that the variance is very low for the RBF kernel and that the bias decreases for dataset greater than 50,000. We can see that the performance improves as the training set size increases.

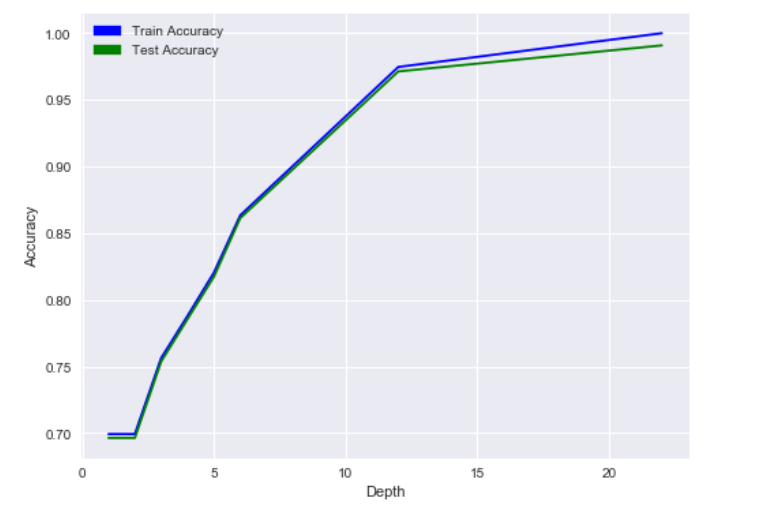
Running cross fold validation, we get the average accuracy of the model. This will give us a good indication of how well the model will generalize as each datapoint is used in both training and testing.

|  |  |
| --- | --- |
| Mean Validation Set accuracy | 95.67% |

Since the model has a very high average accuracy, we can say that this model generalizes better than the other SVM models.

**Experiment 2:**

In this experiment we will be predicting the GPU loads using a decision tree. We will be running the models for different depths and compare the training and test accuracy to select the best depth of the tree. Plotting the test and train accuracy against depth we get the following plot.



From the plot we can see that the variance is low and that the bias is high so there is a lot of underfitting. Only when the depth is more than 12 is does the tree seem to fit for most of the data.

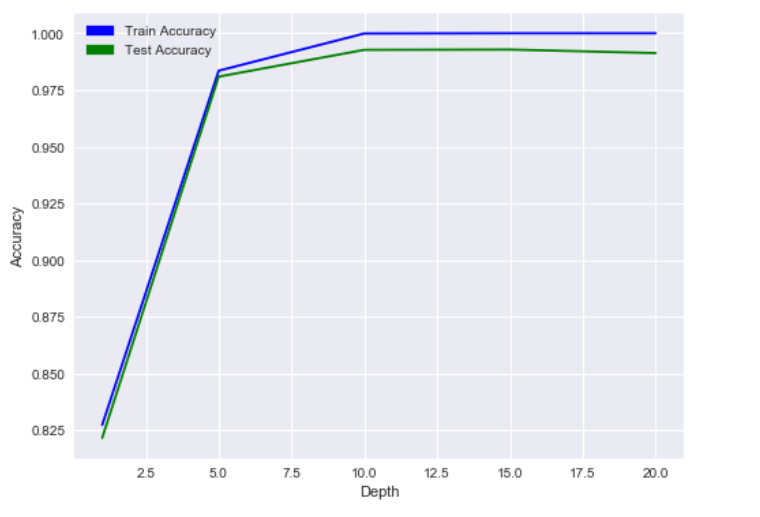
So, running the cross-fold validation with the depths 10, 15 and 20 will give us the best model by checking the average validation accuracy. Running cross validation with 5 folds we get the below average accuracies:

|  |  |
| --- | --- |
| Max Depth | Accuracy |
| 10 | 95.68% |
| 15 | 98.62% |
| 20 | 98.91% |

From the above table we can see that the best depth parameter for the decision tree is 20.

**Experiment 3:**

In this experiment we will be predicting the GPU loads using a gradient boosted tree. We will be running the models for different depths and number of estimators and compare the training and test accuracy to select the best parameters. Plotting the test and train accuracy against depth we get the following graph.

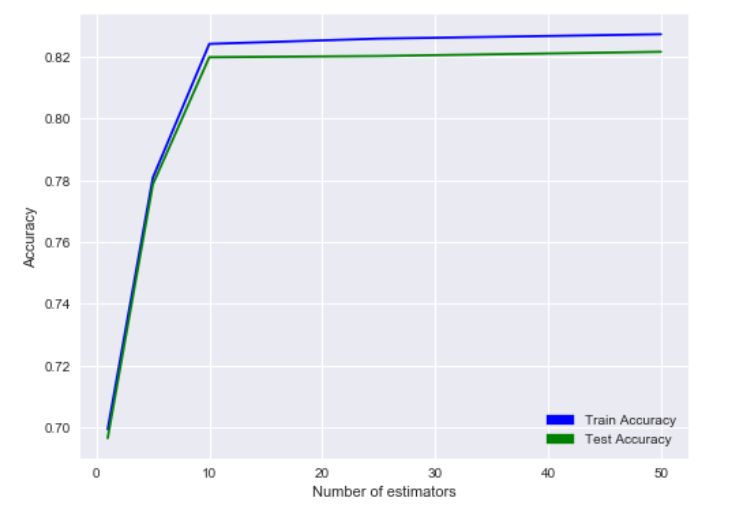


From the plot we can see that the best value for depth is 10 as the variance and bias is very low. For depths below 10 the bias is very high so there is a lot of underfitting. We can confirm our selection by running cross validation with 5 folds and taking the average accuracy for depths 5 and 10.

|  |  |
| --- | --- |
| Max Depth | Accuracy |
| 5 | 98.11% |
| 10 | 99.17% |

From the results of the cross validation run we can confirm that we get the best results for depth 10.

Plotting the test and train accuracy against number of estimators:



From the plot we can see that ideal number of estimators is 10 and above. As the variance and bias is lowest at this point. We can verify our pick for the number of estimators by running cross validation with 5 folds and number of estimators as 5,10 and 25.

|  |  |
| --- | --- |
| Number of estimators | Accuracy |
| 5 | 78.08% |
| 10 | 82.22% |
| 25 | 82.64% |

From the above results we can choose 25 as our ideal number of estimators.

**Conclusion:**

In order to select the best algorithm for predicting high intensive GPU load we need to compare the accuracy of the best models from each experiment.

|  |  |
| --- | --- |
| Algorithm | Accuracy |
| SVM with RBF kernel | 95.67% |
| Decision tree | 98.91% |
| Gradient boosted tree | 82.64% |

From the above results we can see that the decision tree performs well for this classification. We can also be sure that this model will generalize well for the dataset as the above results are from cross validation runs.