**Implementation of Support Vector Machines, Decision Trees and Ensembles using k-fold cross validation**

**Dataset description and goal:**

The goal of the project is to:

1. Implement Support Vector Machines with different kernels
2. Evaluate performance of Decision Trees and experiment with Pruning
3. Compare effectiveness of ensemble methods (Bagging, Random Forest, Boosting) and report optimal parameter values using learning curves.
4. Use K-fold cross validation with all the above algorithms to obtain best accuracy scores.

Two different datasets are chosen to evaluate and compare model performances. The best model and optimal parameter values are reported.

**Dataset 1:**

Contains information on SGEMM GPU Kernel Performance that captures the running time of a matrix-matrix product operation. The dataset information can be viewed and downloaded at UCI Machine learning repository.

<https://archive.ics.uci.edu/ml/datasets/SGEMM+GPU+kernel+performance>.

On quick examination, the following characteristics are observed from the dataset.

* There are 18 columns and 241600 rows
* There are no missing values
* The target variable for the model is average of the four GPU run times
* The independent variables are not highly correlated. Hence there no variables are dropped from the model.
* There are few outliers present, and they need to be removed.

**Dataset 2:**

Contains information on 23 different species of gilled mushrooms that are classified as edible or poisonous based on given features. This is a very interesting dataset that allows to experiment with multiple algorithms like PCA, Logistic Regression, Classification Trees, Ensembles and Neural Networks that readily translate to real-world observations. Also, the dataset has good volume, quality and does not have any aggregations.

In modern world, Plant Biologists classify the different plant species by observing distinctive features and appearances. Drug Specialists, Nutritionists and Agencies like FDA further enhance the models by adding decision variables like adult & child health, interactions of food constituents with other chemicals in body, allergic reactions, animal testing, clinical trials, etc. to predict the positive and negative effects of the plant food item before approving and releasing to the consumer market. A similar simulation of the classification algorithm on multiple predictor variables is carried out in this dataset.

The dataset information can be viewed and downloaded at Kaggle or UCI Machine learning repository.

<https://www.kaggle.com/uciml/mushroom-classification>

<https://archive.ics.uci.edu/ml/datasets/mushroom>

On quick examination, the following characteristics are observed from the dataset.

* There are 23 columns and 8124 rows
* The target variable for the model is ‘class’
* There are 2480 missing values for the ‘stalk-root’ column.
* There are more than 2 categories for most of the predictor variables.

**Assumptions and initial dataset preparation:**

The following assumptions and initial operations are made on the datasets for the implementation of algorithms.

* Cross validation is used with all the algorithms implemented and is written from scratch.
* For plotting SVM, PCA is used to reduce the dimensions to a 2D plane
* Randomized Search is used to search the best values of parameters of tree algorithms

**Dataset 1:**

* Columns 1-10 and 15-18 are continuous and columns 11-14 are categorical variables.
* Scaling has been performed for columns 1-10 using Standard Scaler.
* For binary classification, column 15 is converted to either 1 or 0 using its mean value as cut off.
* The number of outliers (above or below 3 standard deviations from mean) is 5251 (accounts for 2.17 % of data) and they have been removed from the given dataset.
* For computing SVM, only first 20,000 records are selected to reduce time complexity.

**Dataset 2:**

* For binary classification, column ‘class’ is converted to 1 if species is poisonous, and 0 if edible.
* Missing values for the ‘stalk-root’ column is imputed with the frequent value ‘stalk-root’ = ‘b’
* There are more than 2 categories for most predictors, and ordering them (1, 2, 3...) doesn't make any sense because they are not ordinal. Hence pivot is done on categories. This increases the number of columns.

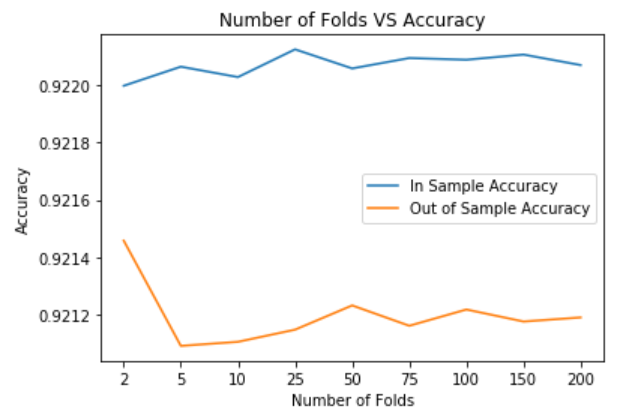
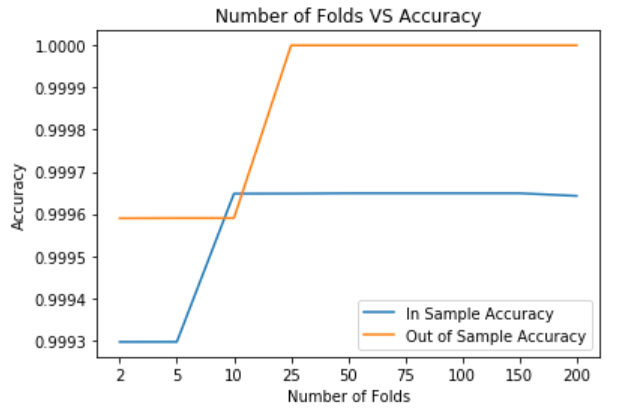
**Tasks 1 & 5:**

The dataset is scaled and partitioned into train set with 70% data, and test set with 30% data. Cross validation is used to split the train and test sets further into k partitions and the algorithm is applied on each partition. The final accuracy of model is the mean of all the k different accuracy scores. An initial value of k=4 is assumed to reduce time complexity.

**Experiment with different number of folds:**

Various values of k are experimented, and the optimal k value is found by applying simple logistic regression model.

***Sgemm Product* *Mushroom***

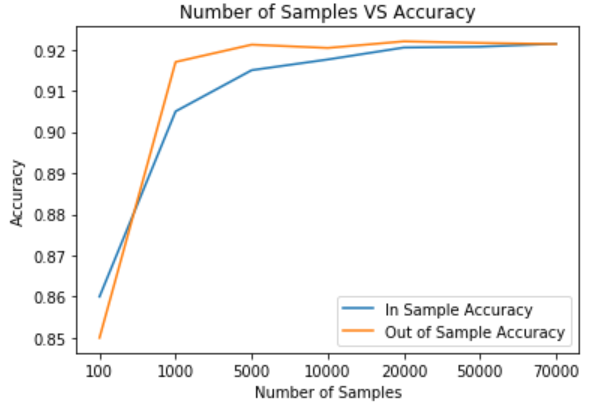
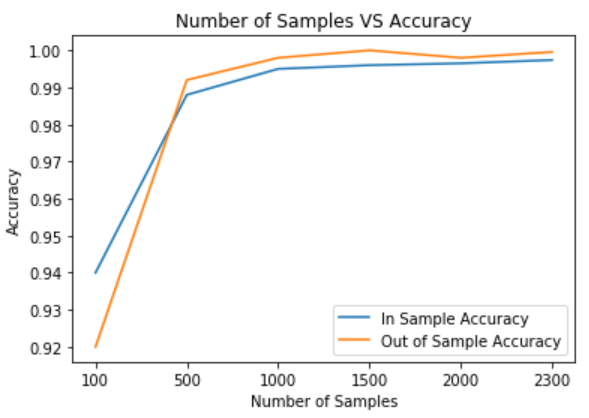
 

It is observed that k=2 is best for sgemm product dataset and k=10 is better for Mushroom dataset. K is chosen such that the variance in train and test set are minimal and accuracy is high.

**Experiment with different sample size:**

Various sample sizes of data are experimented, and the accuracy of train and test sets are found.

***Sgemm Product* *Mushroom***

It is observed accuracy increases with number of samples (N). N=70000 is best for sgemm product dataset and N=2000 is better for Mushroom dataset.

**Tasks 2 & 5:**

Next, instead of logistic regression, SVM algorithm is applied on each of the partitions. The final accuracy of the SVM Kernel model is the mean of all the 4 different accuracy scores. 3 different kernels are used - Linear, Polynomial and Radial Basis Function. These kernels are chosen because they can be applied on binary classification datasets with very less prior knowledge on the dataset. PCA is used to reduce the number of dimensions when plotting SVM decision boundaries. Initial values of parameters are chosen arbitrarily: C=1, gamma=’auto’, degree=3

***Sgemm product***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SVM Kernel** | **K=1** | **K=2** | **K=3** | **K=4** | **Mean Accuracy** |
| Linear |  |  |  |  | In sample:  0.99507  Out of sample:  0.99416 |
| Polynomial |  |  |  |  | In sample:  0.99507  Out of sample:  0.99416 |
| RBF |  |  |  |  | In sample:  0.99507  Out of sample:  0.99416 |

For the sgemm product dataset, all 3 kernels give the same degree of accuracy and there is not much difference in the decision boundary.

***Mushroom***

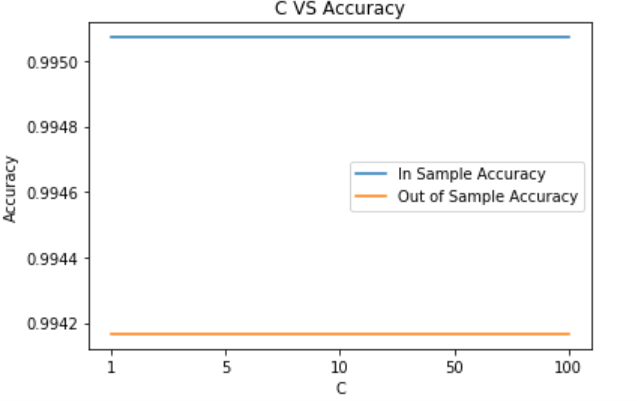
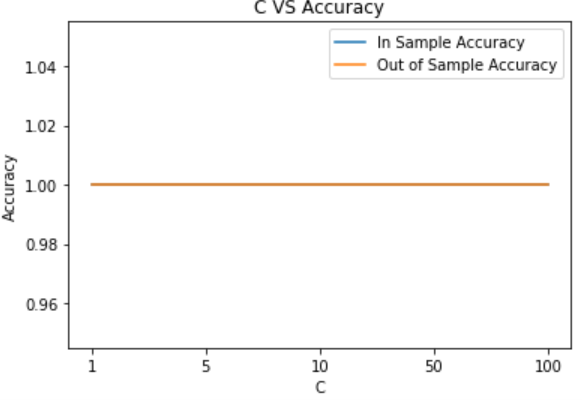
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SVM Kernel** | **K=1** | **K=2** | **K=3** | **K=4** | **Mean Accuracy** |
| Linear |  |  |  |  | In sample:  1.0  Out of sample:  1.0 |
| Polynomial |  |  |  |  | In sample:  0.91399  Out of sample:  0.924109 |
| RBF |  |  |  |  | In sample:  0.99613  Out of sample:  0.99795 |

For mushroom dataset, we find different contour levels and decision boundaries defined for each kernel type. The best accuracy is given by Linear Kernel, followed by Radial Basis Function kernel, followed by Polynomial kernel.

**Experiment with different values of C:**

For Linear kernel, various optimization parameter ( C) values are experimented.

***Sgemm Product* *Mushroom***

It is seen that for various values of C, the accuracy remains constant for both datasets and they don’t change.

**Experiment with different gamma, degree of polynomial and C:**

A randomized search is performed to get the optimal values of parameters for RBF and polynomial kernels for both datasets.

**Sgemm product**: Best parameter values for polynomial kernel are 'gamma': 1, 'degree': 3, 'C': 500

Best parameter values for rbf kernel are 'gamma': 0.1, 'C': 10

**Mushroom**: Best parameter values for polynomial kernel are 'gamma': 1, 'degree': 2, 'C': 1000

Best parameter values for rbf kernel are 'gamma': 0.1, 'C': 1000

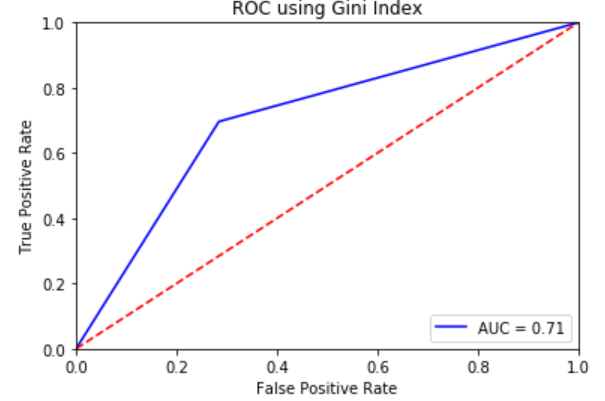
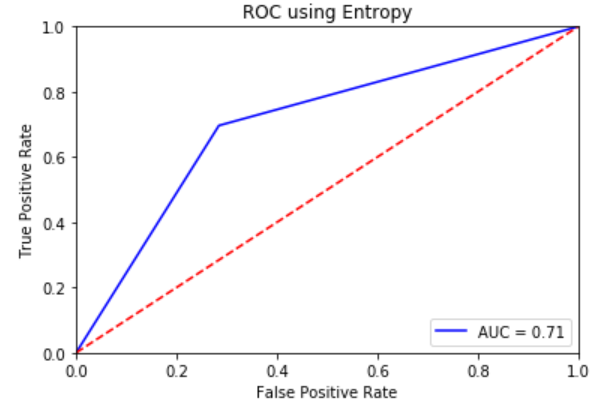
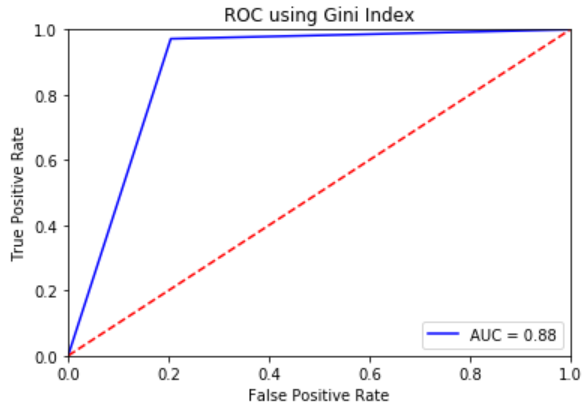
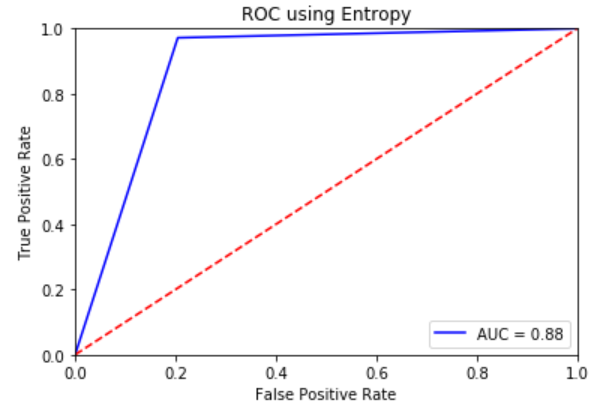
**Tasks 3 & 5:**

Next, Decision Tree algorithm is applied on each of the partitions. The final accuracy of the Decision Tree model is the mean of all the 4 different accuracy scores. Decision trees are implemented using either Gini index or Entropy. Initially the number of trees is arbitrarily set to 100.

**Sgemm Product** **Mushroom**

|  |  |  |  |
| --- | --- | --- | --- |
| **Measure** | **In Sample Accuracy** | **Out of Sample Accuracy** | **Execution time (ms)** |
| Gini | 0.99417 | 0.99404 | 0.31515 |
| Entropy | 0.99416 | 0.99376 | 0.33909 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Measure** | **In Sample Accuracy** | **Out of Sample Accuracy** | **Execution time (ms)** |
| Gini | 0.99982 | 0.99959 | 0.20146 |
| Entropy | 0.99982 | 0.99959 | 0.24734 |

For sgemm product dataset, it is found that Gini Index has higher accuracy and lower computational time than entropy. There is not much difference in area under curve between Gini Index and Entropy. The AUC value is 0.71 for both criteria.

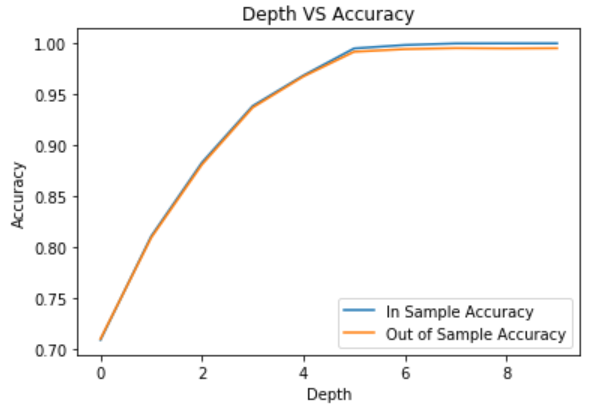
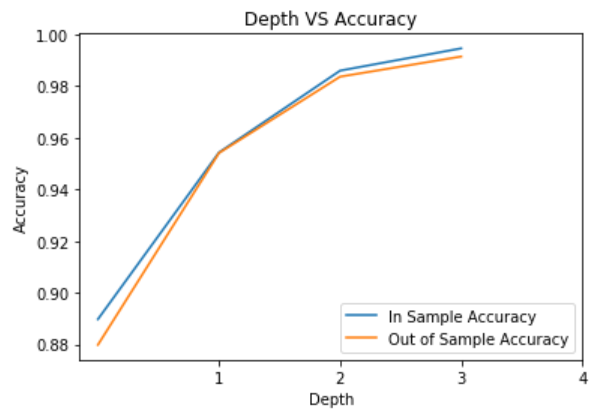
For mushroom dataset, it is found that Gini Index and Entropy have similar accuracy values, but Gini Index has lower computational time than entropy. There is not much difference in area under curve between Gini Index and Entropy. The AUC value is 0.88 for both criteria.

Hence, it is concluded that **Gini Index is better criterion than Entropy** for both the classification datasets.

**Experiment with pruning:**

**Changing depth of tree:** Pruning is experimented by changing values for depth of the tree and the optimal tree depth is evaluated.

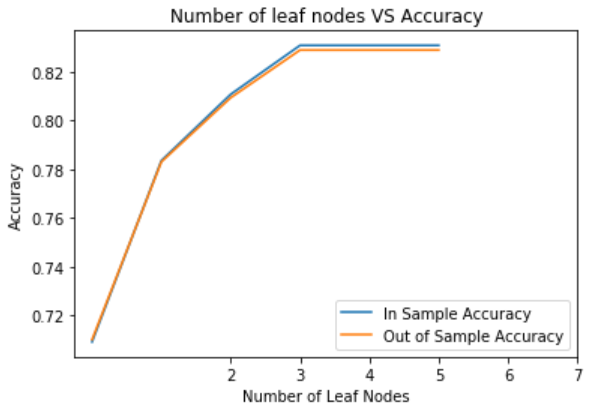
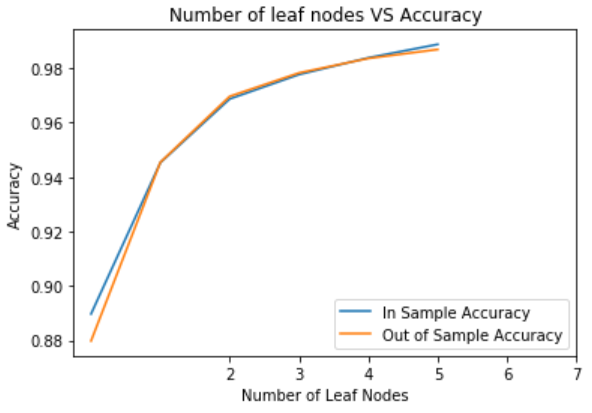
**Sgemm Product** **Mushroom**

For sgemm product dataset, the optimal depth is 4, after which the plots for train and test accuracies begin to diverge. For Mushroom dataset, the optimal depth of tree is 1, after which the plots for train and test accuracies begin to diverge. We look for the depth of tree where train and test accuracies are as close as possible (train and test errors are as low as possible) with minimal variance. This model will generalize well among all the other models.

**Changing number of leaf nodes:** Pruning is experimented by changing number of leaf nodes of the tree and the optimal number is evaluated.

**Sgemm Product** **Mushroom**

For sgemm product dataset, the optimal number of leaf nodes is 1, after which the plots for train and test accuracies begin to diverge. For Mushroom dataset as well, the optimal number of leaf nodes is 1, after which the plots for train and test accuracies begin to diverge. We look for the number of leaf nodes where train and test accuracies are as close as possible (train and test errors are as low as possible) with minimal variance. This model will generalize well among all the other models.

**Tasks 4 & 5:**

**Bagging**

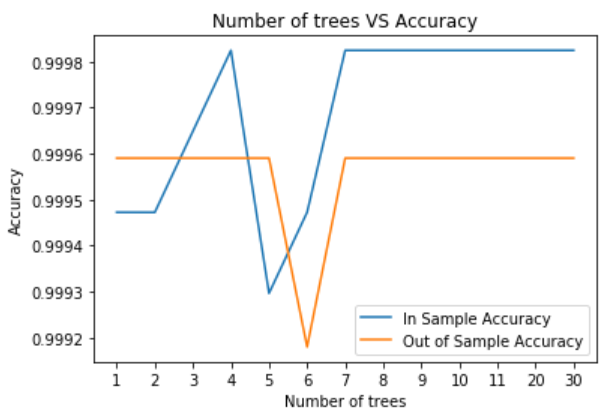
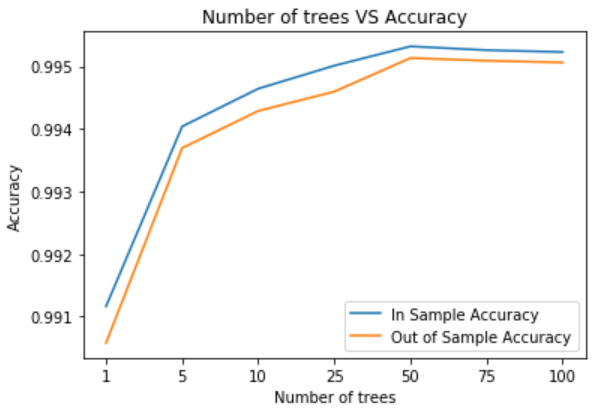
Boostrap Aggregation is performed with cross validation with base classifier as decision trees. The final accuracy of the Bagging model is the mean of all the 4 different accuracy scores. Initially the number of trees is arbitrarily set to 100.

|  |  |  |
| --- | --- | --- |
| **Data set** | **In Sample Accuracy** | **Out of Sample Accuracy** |
| Sgemm Product | 0.995297 | 0.995063 |
| Mushroom | 0.99982 | 0.99959 |

**Experiment with different number of trees:**

The performance of bagging is experimented by changing the number of trees. A graph is plotted with number of trees and corresponding accuracy values.

**Sgemm Product** **Mushroom**



For sgemm product dataset, the optimal number of decision trees used for bagging is 50, after which the accuracy of the model begins to decline. For mushroom dataset, the optimal number of decision trees used for bagging is 7, after which the accuracy of the model begins to remain constant.

**Random Forest**

Random Forest is performed with cross validation with base classifier as decision trees. The final accuracy of the Random Forest model is the mean of all the 4 different accuracy scores. Initially the number of trees is arbitrarily set to 100, and number of predictor features is set to 5.

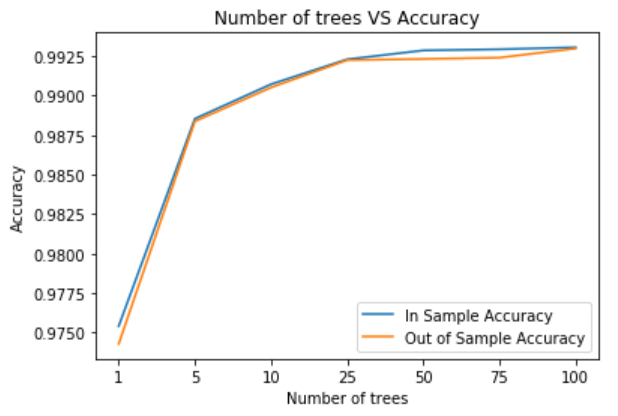
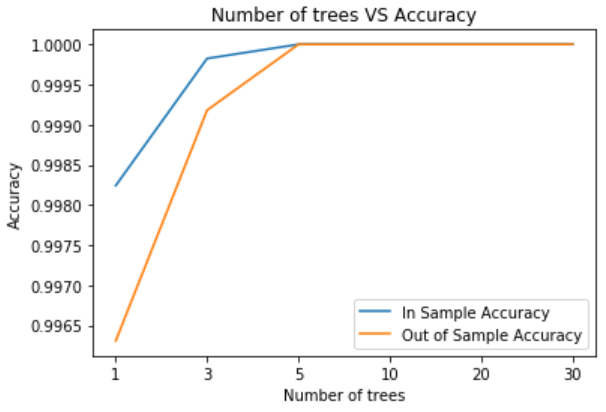
|  |  |  |
| --- | --- | --- |
| **Data set** | **In Sample Accuracy** | **Out of Sample Accuracy** |
| Sgemm Product | 0.993018 | 0.99279 |
| Mushroom | 1.0 | 1.0 |

For sgemm product dataset, bagging gives more accuracy than random forest. But for Mushroom dataset, Random Forest gives more accuracy than bagging.

**Experiment with different number of trees:**

The performance of Random Forest is experimented by changing the number of trees. A graph is plotted with number of trees and corresponding accuracy values.

**Sgemm Product** **Mushroom**

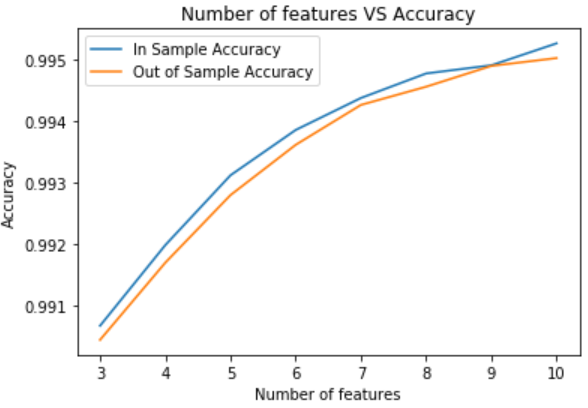
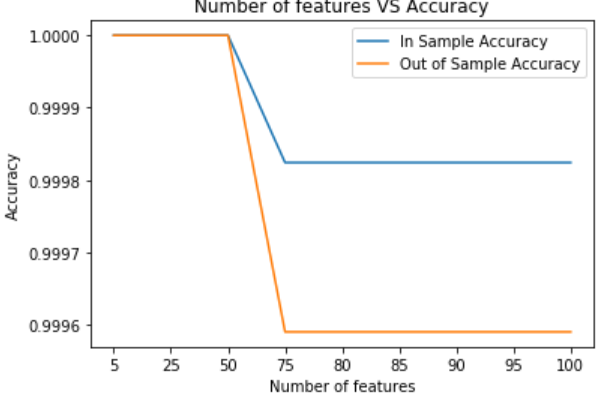
 

For sgemm product dataset, the optimal number of decision trees used for Random Forest is 25, after which the plots of train and test accuracies begin to diverge or become constant. For mushroom dataset, the optimal number of decision trees used for random forest is 5, after which the accuracy of the model begins to remain constant.

**Experiment with different number of features:**

The performance of Random Forest is experimented by changing the number of features selected. A graph is plotted with number of features and corresponding accuracy values.

**Sgemm Product** **Mushroom**

For sgemm product dataset, the optimal number of features used for Random Forest is 9, after which the plots of train and test accuracies begin to diverge. For mushroom dataset, the optimal number of features used for random forest is 50 or less, after which the accuracy of the model begins to decline.

**Boosting:**

Boosting is performed using Adaptive Boost (AdaBoost), Stochastic Gradient Boost and XG Boost

**AdaBoost:**

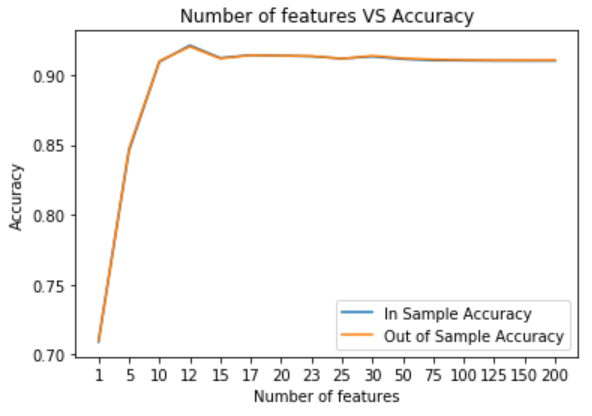
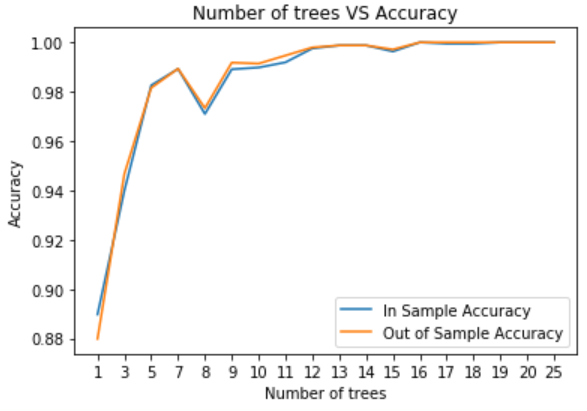
Adaptive Boost is performed with cross validation with base classifier as decision trees. The final accuracy of the Adaptive Boost model is the mean of all the 4 different accuracy scores. Initially the number of trees is arbitrarily set to 100.

|  |  |  |
| --- | --- | --- |
| **Data set** | **In Sample Accuracy** | **Out of Sample Accuracy** |
| Sgemm Product | 0.910362 | 0.910767 |
| Mushroom | 1.0 | 1.0 |

**Experiment with different number of trees:**

The performance of AdaBoost is experimented by changing the number of trees. A graph is plotted with number of trees and corresponding accuracy values.

**Sgemm Product** **Mushroom**

For sgemm product dataset, the optimal number of decision trees used for AdaBoost is 12, after which the plots of train and test accuracies begin to decline. For mushroom dataset, the optimal number of decision trees used for AdaBoost is 20, after which the accuracy of the model begins to remain constant.

**Stochastic Gradient Boost:**

Stochastic Gradient Boost is performed with cross validation. The final accuracy of the Stochastic Boost model is the mean of all the 4 different accuracy scores. Initially the number of trees is arbitrarily set to 100.

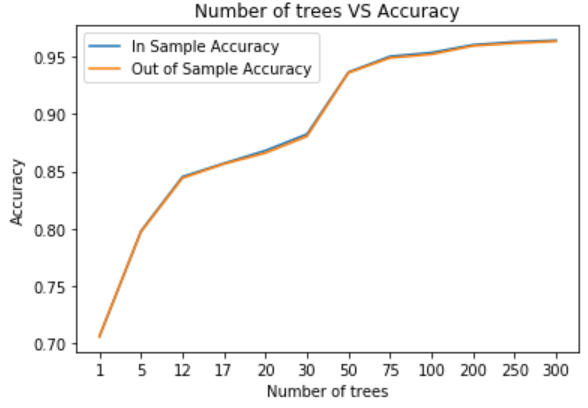
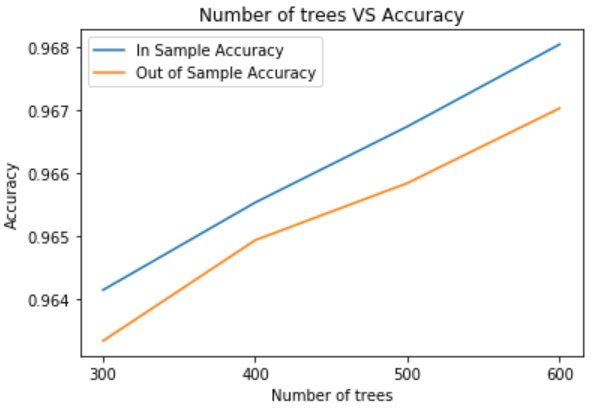
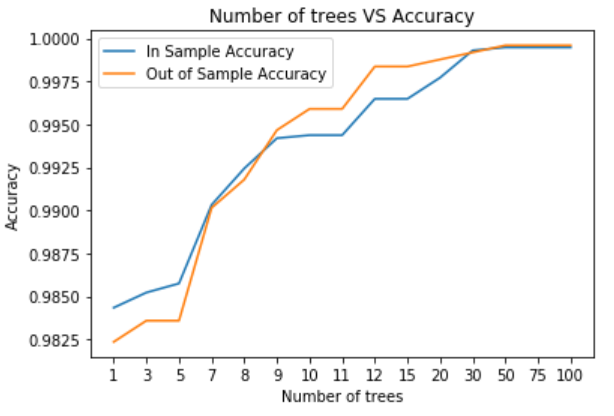
|  |  |  |
| --- | --- | --- |
| **Data set** | **In Sample Accuracy** | **Out of Sample Accuracy** |
| Sgemm Product | 0.95346 | 0.95197 |
| Mushroom | 0.99947 | 0.99959 |

For sgemm product dataset, Stochastic Gradient Boost gives more accuracy than AdaBoost. But for Mushroom dataset, AdaBoost gives more accuracy than Stochastic Gradient Boost.

**Experiment with different number of trees:**

The performance of Stochastic Gradient Boost is experimented by changing the number of trees. A graph is plotted with number of trees and corresponding accuracy values.

**Sgemm Product**  **Mushroom**

For sgemm product dataset, the accuracy increases with increase in number of decision trees. But when number of trees reach 400, then the test accuracy begins to decline. Hence the optimal number of decision trees used for Stochastic Gradient Boost is 400, after which the model begins to overfit data. For mushroom dataset, the optimal number of decision trees used for Stochastic Gradient Boost is 50, after which the accuracy of the model begins to remain constant.

**XG Boost:**

XG Boost is performed with cross validation. The final accuracy of the XG Boost model is the mean of all the 4 different accuracy scores. Initially the number of trees is arbitrarily set to 100.

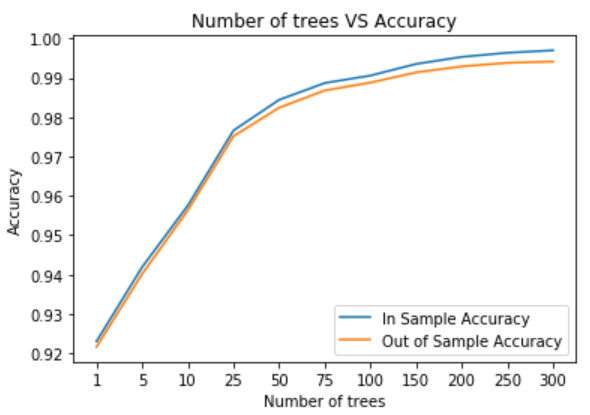
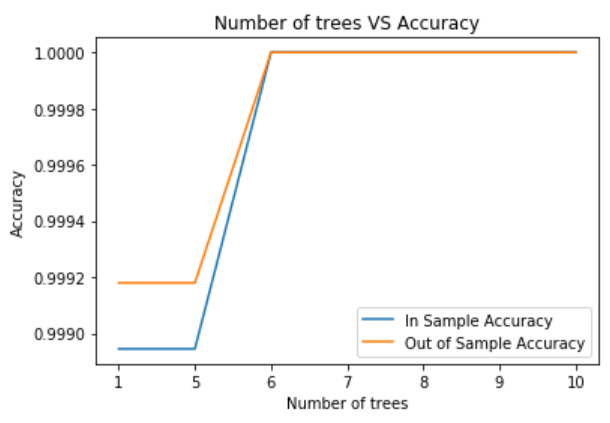
|  |  |  |
| --- | --- | --- |
| **Data set** | **In Sample Accuracy** | **Out of Sample Accuracy** |
| Sgemm Product | 0.99060 | 0.988816 |
| Mushroom | 1.0 | 1.0 |

Among all the boosting algorithms, XG Boost gives the maximum accuracy for both the datasets. Hence it is the best model to use.

**Experiment with different number of trees:**

The performance of XG Boost is experimented by changing the number of trees. A graph is plotted with number of trees and corresponding accuracy values.

**Sgemm Product**  **Mushroom**

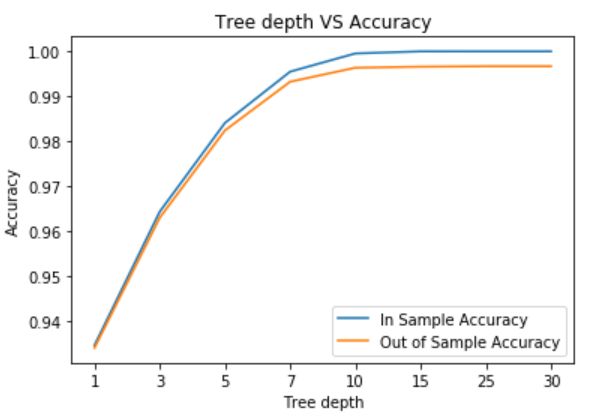
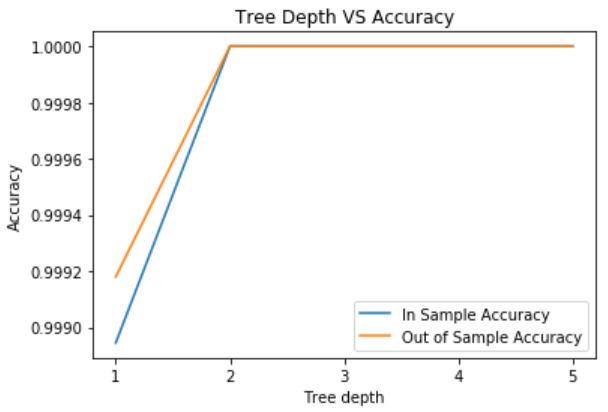
 

For sgemm product dataset, the optimal number of decision trees used for XG Boost is 25, after which the model begins to overfit data. For mushroom dataset, the optimal number of decision trees used for XG Boost is 6, after which the accuracy of the model begins to remain constant.

**Experiment with depth of trees:**

The performance of XG Boost is experimented by changing the depth of trees. A graph is plotted with depth of trees and corresponding accuracy values.

**Sgemm Product**  **Mushroom**

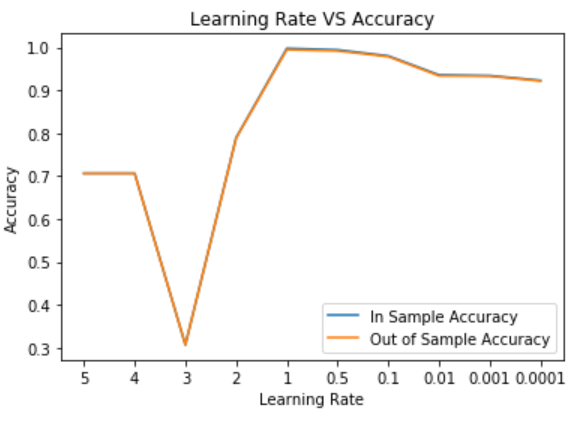
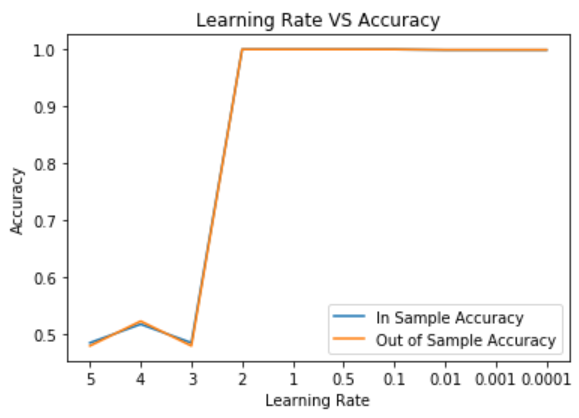
 

For sgemm product dataset, the optimal depth of decision trees used for XG Boost is 3, after which the model begins to overfit data. For mushroom dataset, the optimal depth of decision trees used for XG Boost is 2, after which the accuracy of the model begins to remain constant.

**Experiment with multiple learning rates:**

The performance of XG Boost is experimented by changing learning rate. A graph is plotted with learning rate and corresponding accuracy values.

**Sgemm Product**  **Mushroom**

For sgemm product dataset, the optimal learning rate used for XG Boost is 1, after which the model accuracy begins to decline. For mushroom dataset, the optimal learning rate used for XG Boost is 2, after which the accuracy of the model begins to decline slightly.

**Comparison of different models:**

1. Cross validation improves the model accuracy scores and reduces the bias by averaging the accuracy/error scores.
2. For categorical classifications, Gini Index scores better in terms of computing accuracy, error and time complexity. Entropy includes a log term, and it involves more computational complexity than Gini Index.
3. A decision tree is pruned to improve accuracy and avoid overfitting. Pruning is done by limiting depth of tree and number of leaf nodes.
4. Ensembles (Bagging, Random Forest, Boosting) are combination of weak learners (decision trees in this case) which provide more accuracy than individual weak learners. Hence accuracy from Ensembles is much better than any other model.
5. For Sgemm product dataset, maximum accuracy is given by Bagging, followed by SVM (gini), SVM (Entropy), Random Forest, XG Boost, Stochastic Gradient Boost and AdaBoost.
6. For Mushroom dataset, maximum accuracy is given by 3 algorithms – Random Forest, AdaBoost and XG Boost. It is followed by SVM (gini), SVM (Entropy), Bagging, Stochastic Gradient Boost.
7. Among SVM Kernels, best accuracy is given by Linear Kernel, followed by Radial Basis Function kernel, followed by Polynomial kernel.
8. PCA and feature selection can further be used to reduce the number of dimensions in the dataset and improve time complexity.