BUAN 6341 Applied Machine Learning **Date : 03/09/2020**

Assignment 2

Implementing learning Algorithms for classification

Prepared by : Kamini Bokefode

DATASETS

Two datasets have been used to evaluate the performance of classification algorithms. Details of the first dataset are as follows:

Dataset1: This dataset has been referred as DT1 throughout the report. The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed.

Dataset has been downloaded from the following link. Only bank-full.csv has been used for this assignment.

http://archive.ics.uci.edu/ml/datasets/Bank+Marketing

Dataset2: This dataset has been referred as DT2 throughout the report. This data set measures the running time of a matrix-matrix product A*B = C, where all matrices have size 2048 x 2048, using a parameterizable SGEMM GPU kernel with 241600 possible parameter combinations. For each tested combination, 4 runs were performed and their results are reported as the 4 last columns. All times are measured in milliseconds*. There are 14 parameter, the first 10 are ordinal and can only take up to 4 different powers of two values, and the 4 last variables are binary. Dataset can be downloaded from the following link :

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

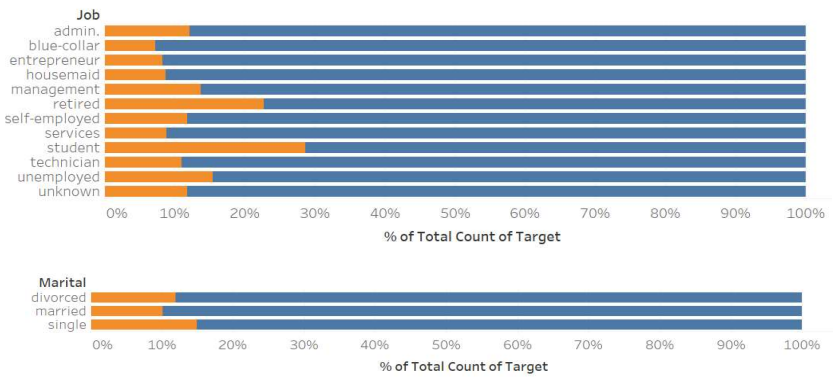
DATA EXPLORATION

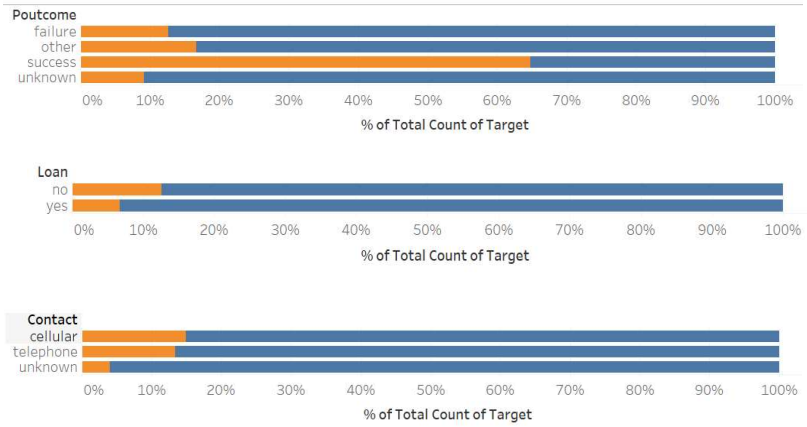
Data preprocessing:

Features like 'marital','contact','poutcome','default','housing','loan' have been transformed to dummy variables and unique types of the following variables "job","education","month" have been mapped to numbers using map function.

Visualization:

Percentage of client who have subscribed to the bank term deposit and those who have declined upon contacting for the term deposit have been plotted below.





Cross validation :

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model. It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

MACHINE LEARNING ALGORITHM

Following Machine Learning Algorithm has been used for the classification problem for both datasets.

- Support Vector Machine (Linear Kernel, RBF Kernel, Sigmoid Kernel)
- Decision Tree (without hyperparameter tuning)
- Decision Tree (with hyperparameter tuning)
- Adaptive Boosting
- Adaptive Boosting (Hyperparameter tuning)

Evaluation Metric:

Accuracy has been used to evaluate performance of the different models. It is the ratio of number of correct predictions to the total number of input samples. It works well only if there are equal number of samples belonging to each class. Therefore, cross validation has been used to avoid problems caused by the imbalanced data.

SUPPORT VECTOR MACHINE

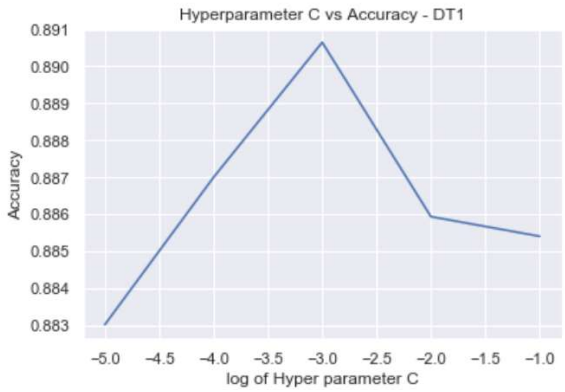
C is the regularization parameter that controls the trade-off between the slack variable penalty (misclassifications) and width of the margin. It controls the tradeoff between smooth decision boundary and classifying the training points correctly. Gamma is a parameter for nonlinear hyperplanes. The higher the gamma value it tries to exactly fit the training data set.

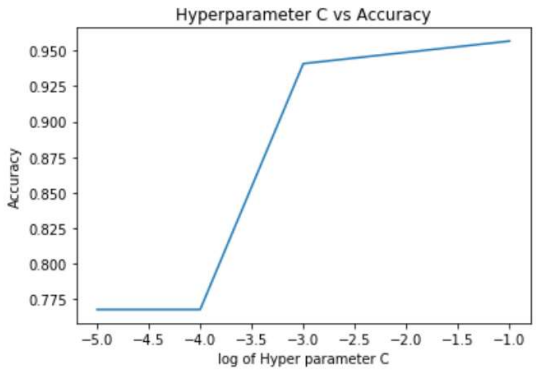
Support Vector Machine (Linear Kernel)

K(x,xi) = sum(x*xi)
Hyperparameters : C = [0.0001,0.0005,0.001,0.01,0.1]

For the linear kernel function with the increase in value of C parameter, the boundary between classification areas should become finer and should result in increasing in the accuracy for the training data.

For DT1 we can see that the accuracy increases from 0.883 to 0.891 however upon further increasing the C value, the accuracy drops down to 0.885. For DT2, we can see the expected behavior, i.e. accuracy continues to increase with higher values of Parameter C. The highest accuracy attained is 0.955 at C = 0.1





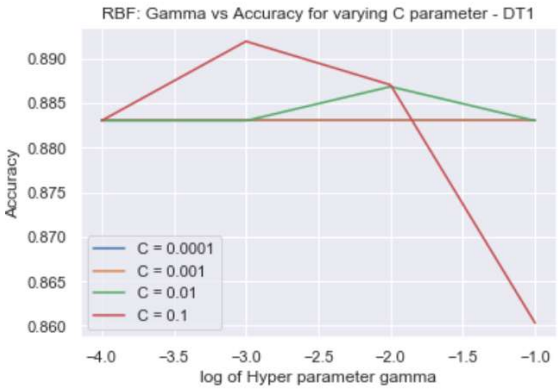
Support Vector Machine (RBF Kernel)

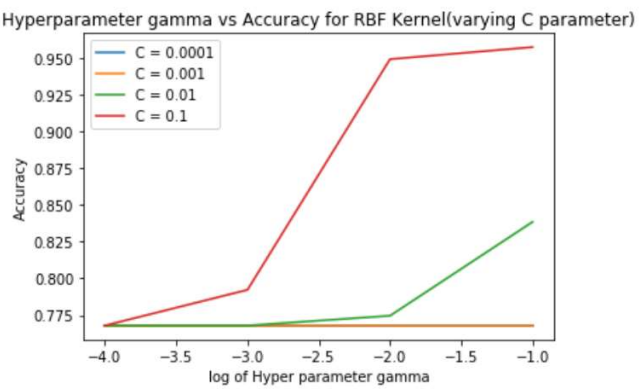
The radial basis function kernel is popular kernel function commonly used in support vector machine classification. RBF can map an input space in infinite dimensional space.

K(x,xi) = exp(-gamma*sum((x-xi^2))
Hyperparameters: Gamma: [0.0001,0.001,0.01,0.1] **, C:** [0.0001,0.001,0.01,0.1]

In this experiment accuracy was calculated for 16 possible combinations of the hyperparameter C and Gamma. The accuracy remains the same for most of the combinations of C and gamma for both the datasets. It can be observed in both the datasets that there is significant variation in the accuracy with gamma for C=0.1. It can be inferred that C=0.1 and Gamma = .001 gives highest accuracy of 0.8919 for DT1. And that C=0.1 and Gamma = .0 gives highest accuracy of 0.9572 for DT2.

	DATASET 1				DATASET 2			
	Gamma				Gamma			
C	0.0001	0.001	0.01	0.1	0.0001	0.001	0.01	0.1
0.0001	0.883015	0.883015	0.883015	0.883015	0.76792	0.76792	0.76792	0.76792
0.001	0.883015	0.883015	0.883015	0.883015	0.76792	0.76792	0.76792	0.76792
0.01	0.883015	0.883015	0.886842	0.883015	0.76792	0.76792	0.77456	0.83836
0.1	0.883015	0.891929	0.887041	0.860321	0.76792	0.7922	0.949	0.9572



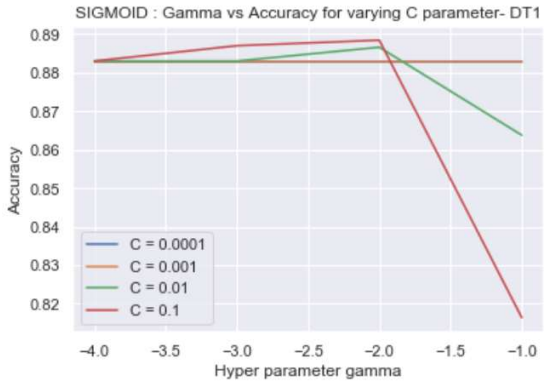


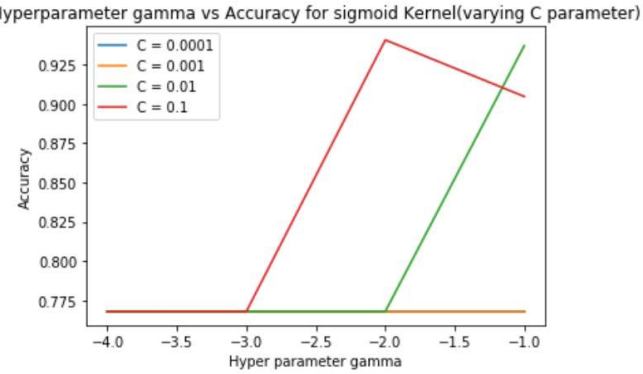
Support Vector Machine (Sigmoid Kernel)

K(x,y) = tanh(ax^t*y+c)
Hyperparameters: Gamma: [0.0001,0.001,0.01,0.1] , **C:** [0.0001,0.001,0.01,0.1]

In this experiment accuracy was calculated for 16 possible combinations of the hyperparameter C and Gamma. The accuracy remains the same for most of the combinations of C and gamma for both the datasets. It can be observed in both the datasets that there is significant variation in the accuracy with gamma for C=0.1. It can be inferred that C=0.1 and Gamma = .001 gives highest accuracy of 0.887 for DT1. And that C=0.1 and Gamma = .01 gives highest accuracy of 0.94076 for DT2.

	DATASET 1				DATASET 2			
	Gamma				Gamma			
C	0.0001	0.001	0.01	0.1	0.0001	0.001	0.01	0.1
0.0001	0.88302	0.88302	0.88302	0.88302	0.76792	0.76792	0.76792	0.76792
0.001	0.88302	0.88302	0.88302	0.88302	0.76792	0.76792	0.76792	0.76792
0.01	0.88302	0.88302	0.88684	0.88302	0.76792	0.76792	0.76792	0.93708
0.1	0.88302	0.887	0.88843	0.81637	0.76792	0.76792	0.94076	0.90476





- **Gridsearch:** It creates a grid of hyper-parameters and just try all of their combinations (hence, this method is called Gridsearch, But don't worry! we don't have to do it manually because Scikit-learn has this functionality built-in with GridSearchCV. GridSearchCV takes a dictionary that describes the parameters that could be tried on a model to train it. The grid of parameters is defined as a dictionary, where the keys are the parameters and the values are the settings to be tested. Upon using GridSearchCV for both the datasets, following output has been received.

	Data Set1	Dataset2
C	0.1	0.1
Gamma	0.001	0.1
Kernel	rbf	rbf
Accuracy	0.8919	0.9546

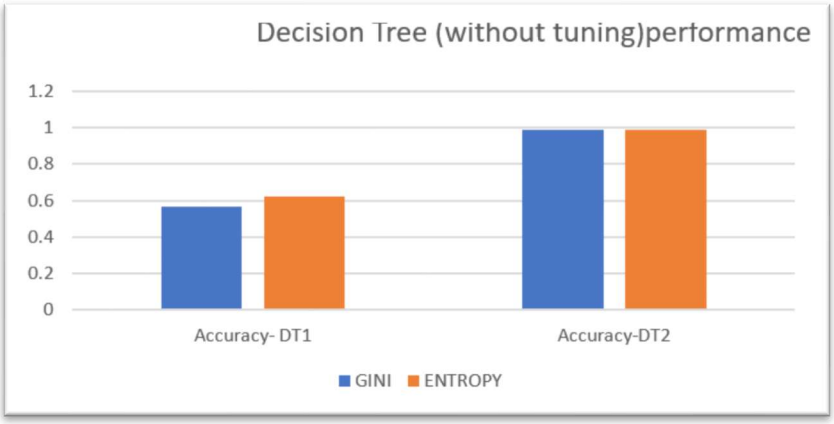
Best Kernel function?

It can be concluded that for both the datasets, rbf kernel has performed best with parameter C = 0.1 resulting in accuracy of 0.8919 for DT1 and 0.9546 for DT2.

DECISION TREE (WITHOUT HYPERPARAMETER TUNING)

Decision tree breaks down a data set into smaller and smaller subsets while ar the same time an associated decision tree us incrementaly developed. Decision tree with default values for all the parameters was executed using python library sklearn for both the datasets.

- It can be inferred that without hypertuning, decision tree with criterion = entropy outperforms in both the datasets.
- The accuracy of the decision tree for DT2 is very high
- The performance of the decision free for the DT1 is less than 0.63



Metric to split on variables	Accuracy- DT1	Accuracy-DT2
GINI	0.5687	0.9889
ENTROPY	0.6228	0.989

DECISION TREE (WITH HYPERPARAMETER TUNING)

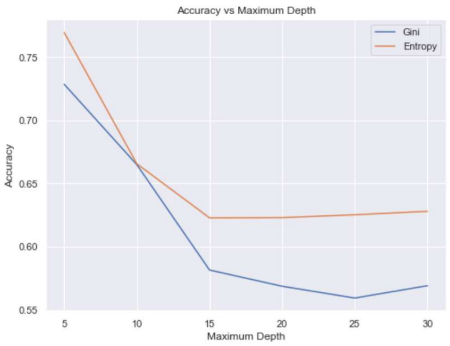
Growing the tree beyond a certain level of complexity leads to overfitting.In our data, age doesn't have any impact on the target variable.Growing the tree beyond Gender is not going to add any value. Need to cut it at Gender.This process of trimming trees is called Pruning.

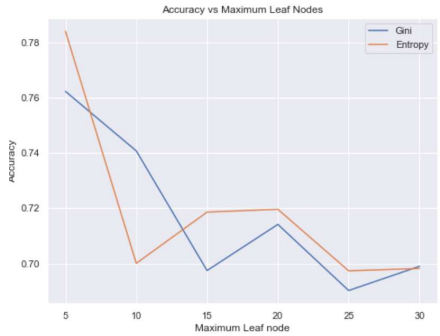
We can avoid overfitting by changing the parameters like:

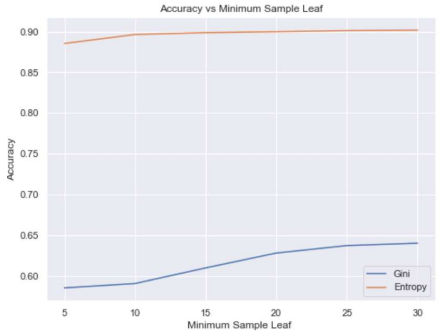
- max_leaf_nodes : Reduce the number of leaf nodes
- min_samples_leaf : Restrict the size of sample leaf
- max_depth : Reduce the depth of the tree to build a generalized tree

For both the dataset in this experiment, range of values used for pruning parameters are same and are 5,10,15,20,25 & 30.

The accuracy for both gini and entropy criterion for different pruning parameters has been plotted for DT1 below.

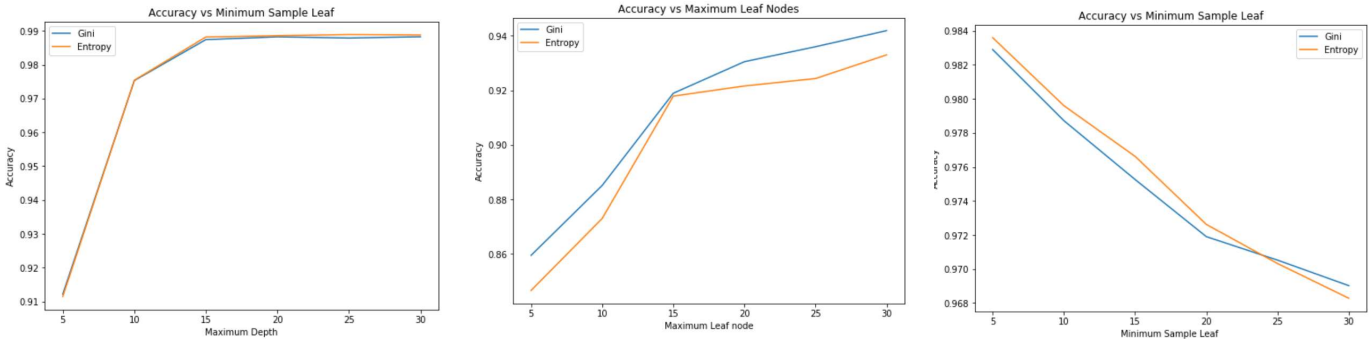






- With the increase in parameter values for max_depth and maximum_leaf_nodes, accuracy is decreasing.
- For pruning parameters like max_depth and maximum_leaf_nodes, the performance by criterion gini and entropy are not significantly different.
- For parameter max_sample_leaf, criterion entropy has outperformed ober gini by almost 50%

The accuracy for both gini and entropy criterion for different pruning parameters has been plotted for DT2 below



- In contrast to the pattern observed in DT1, the performance of both entropy and gini criterion is almost same with changes in all the pruning parameters used in this experiment.
- There is increase in the accuracy with the increase in max_depth and max_leaf_nodes.
- The accuracy of the model decreases with the increase in the min_sample_leaf. However the decrease is not that significant.

GridSearchCV has been used to evaluate performance of decision trees with all possible combinations given values pruning parameters. the best values for the parameters given by GridSearchCV method are as follows:

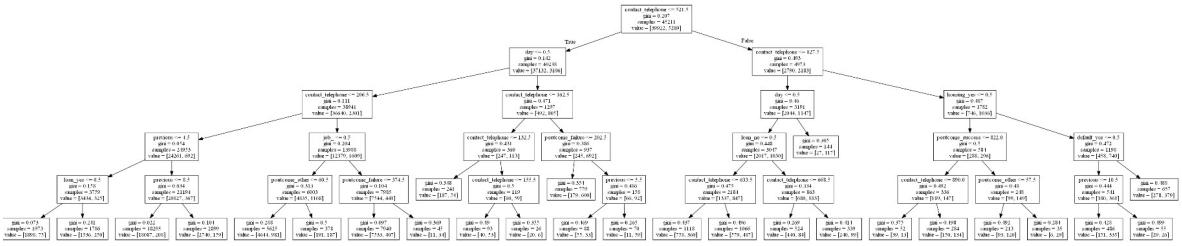
Decision Tree		
Dataset	DT1	DT2
criterion	gini	gini
max_depth	5	8
max_leaf_nodes	26	29
min_samples_leaf	24	15
Accuracy	0.7306	0.9422

Based on the values given by grid search result, Decision tree has been drawn using graphviz for both the datasets.

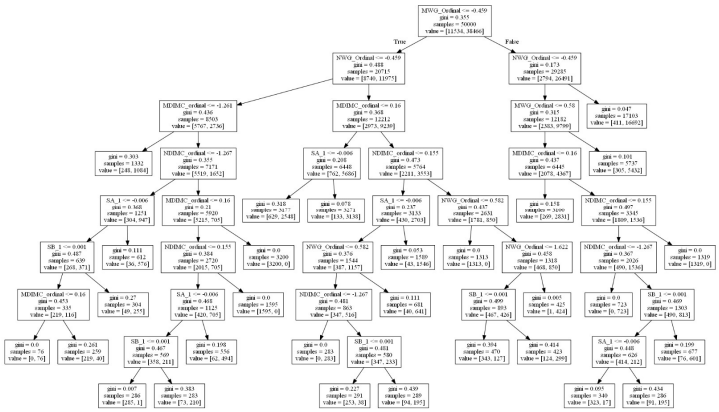
For DT1, contact has the highest information gain and is the root node followed by day and previous outcome.

For DT2, MWG_Ordinal is the root node and followed by NWG and MDIMC. This outcome is inline with our observation that MWG_ordinal has some values which always result in low runtime compared to other parameters in the dataset.

Decision tree diagram for DT1



Decision Tree diagram for DT2:



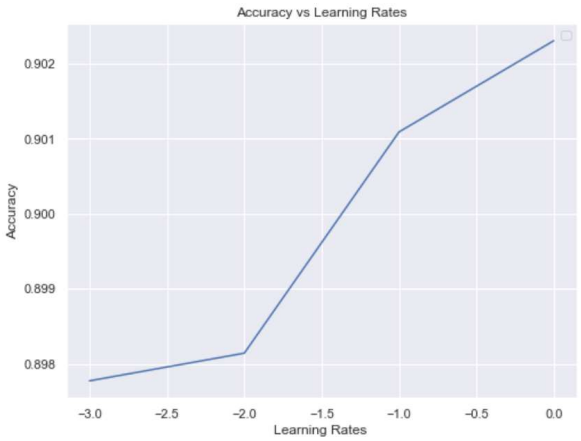
ADAPTIVE BOOSTING:

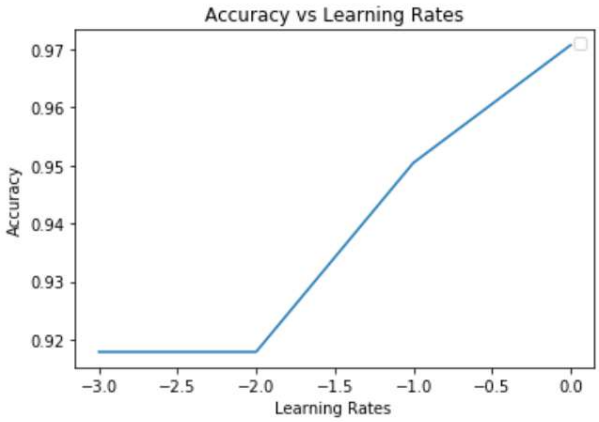
Boosted Decision Tree(without pruning)

AdaBoost works on improving the areas where the base learner fails. The base learner is a machine learning algorithm which is a weak learner and upon which the boosting method is applied to turn it into a strong learner. Any machine learning algorithm that accept weights on training data can be used as a base learner.

Using the parameter values from pruned decision tree, adaptive boosted model was evaluated for different learning rates for both the datasets. The plot with grid lines in the background is the log(learning rate) vs Accuracy plot for DT1 and the other one is for DT2.

It can be clearly inferred that learning rate of 1 gives best accuracy for both datasets of 0.902 and 0.97 for DT1 and DT2.

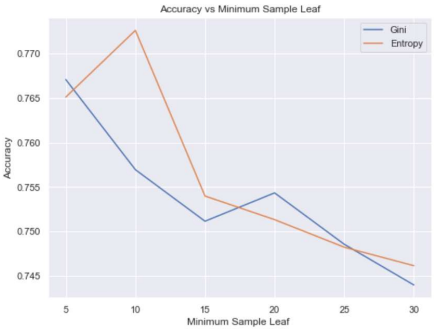


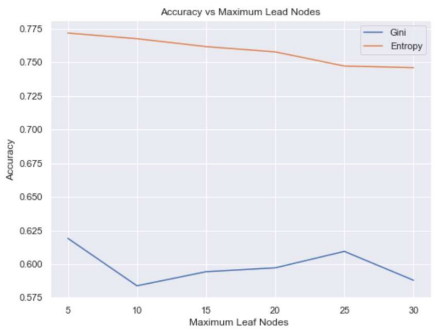


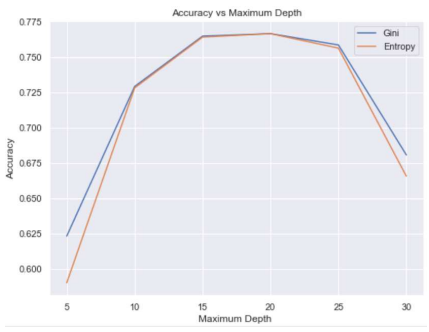
Boosted Decision Tree(pruning)

For both the dataset in this experiment, range of values used for pruning parameters are same and are 5,10,15,20,25 & 30.

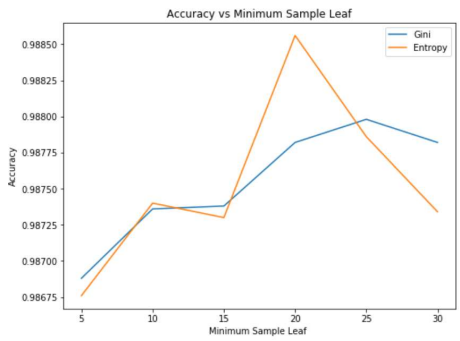
The accuracy of both gini and entropy criterion for different pruning parameters has been plotted for DT1 below

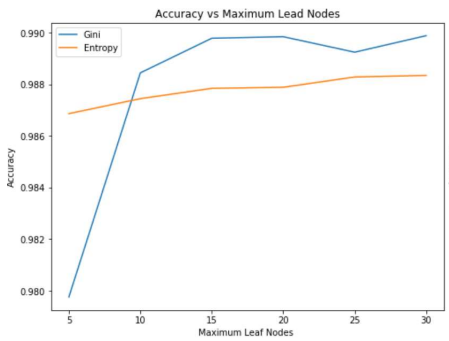


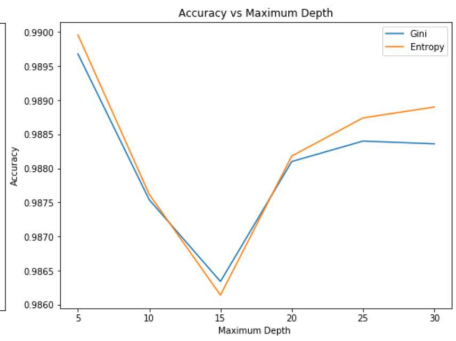




The accuracy of both gini and entropy criterion for different pruning parameters has been plotted for DT2 below







Parameter tuning analysis

- The adaptive boosting performed really for DT2 as the accuracy for the decision tree increased from 0.9422 to .98
- In contrast to DT2, for DT1 adaptive boosting has underperformed as the accuracy of the model dropped from 0.736 to 0.5698.
- For DT2 and DT1 the accuracy for both gini and entropy criterion are almost same for different pruning parameters except for max_leaf_nodes of DT1 where entropy has outperformed gini criterion.

GridSearchCV has been used to evaluate performance of decision trees with all possible combinations given values pruning parameters. the best values for the parameters given by GridSearchCV method are as follows:

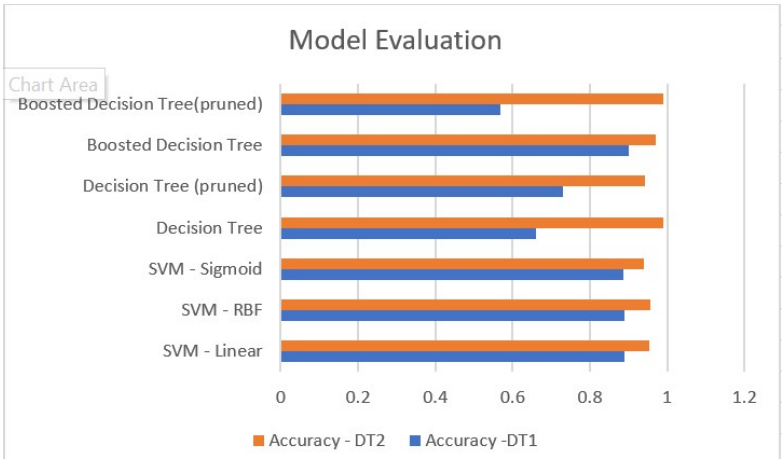
Adaptive Boosted	Decision tree	
	DT1	DT2
criterion	gini	gini
max_depth	5	8
max_leaf_nodes	22	29
min_samples_leaf	24	15
Accuracy	0.5698	0.9896

MODEL EVALUATION

Below is the summary of performances of models for both datasets.

Algorithm	Accuracy -DT1	Accuracy - DT2
SVM - Linear	0.891	0.955
SVM - RBF	0.891	0.9572
SVM - Sigmoid	0.887	0.9407
Decision Tree	0.662	0.989
Decision Tree (pruned)	0.7306	0.94228
Boosted Decision Tree	0.902	0.97
Boosted Decision Tree(pruned)	0.5698	0.989

Here is quick overview of accuracy of all the models evaluated in this experiment.



Performance analysis of all algorithms:

- The performance of all the algorithms for DT2 is close to each other with model accuracy lying between (0.94 to 0.98). However, the Boosted decision tree(pruned) has performed best with accuracy of 0.98.
- The performance of all the algorithms for DT1 have been inconsistent. It can be clearly seen that Boosted decision tree without any pruning has performed really well in comparison with other algorithms.
- In contrast to performance of boosted decision tree with pruning for DT2, in DT1 the algorithm has performed very poorly. It has resulted in the least accuracy of the model for DT1
- The performance of SVM with different kernels have almost similar performance for both the dataset.

Best algorithm? Why?

For both the Datasets boosted version of Decision tree has performed best resulting in highest accuracy. Ensemble methods always to improve the accuracy of your model. There are two good reasons for this:

a) They are generally more complex than traditional methods.

b) The traditional methods give a good base level from which you can improve and draw from to create your ensembles.

Scope of improvement

Performance analysis of all algorithms have been performed based on accuracy only. Although accuracy is a decent performance parameter it cannot be sufficient in to evaluate a model performance. Based on the dataset and prompt other parameters like precision , rank and F1 score can also be very helpful in deciding which model to use for unseen data.

In this experiment only three pruning parameters have been used. Another pruning technique parameterized by the cost complexity parameter, ccp_alpha could have been used.

Feature preprocessing of dataset resulted in values which were difficult to interpret especially in decision tree. It is therefore recommended to perform feature scaling manually based on the data.