# CART_RandomForest

September 2, 2021

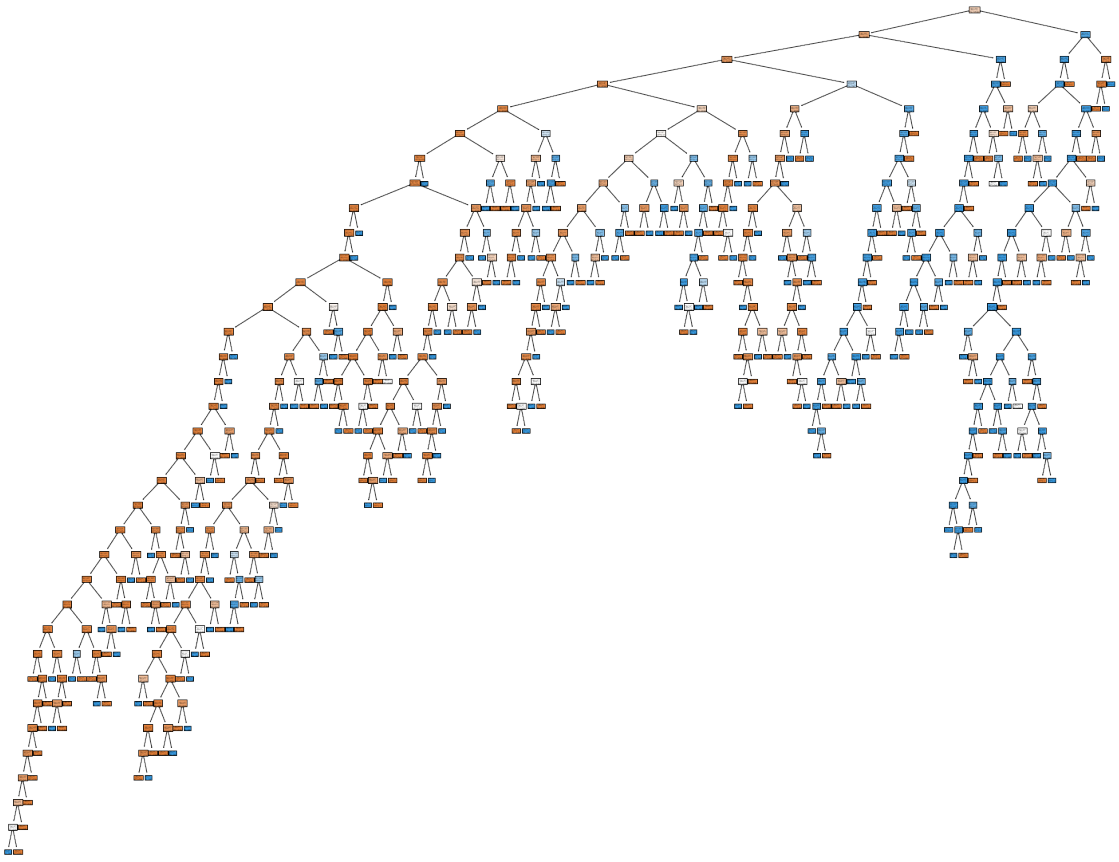
### 0.0.1 Random forest and one-class SVM for email spam classifier (25 points)

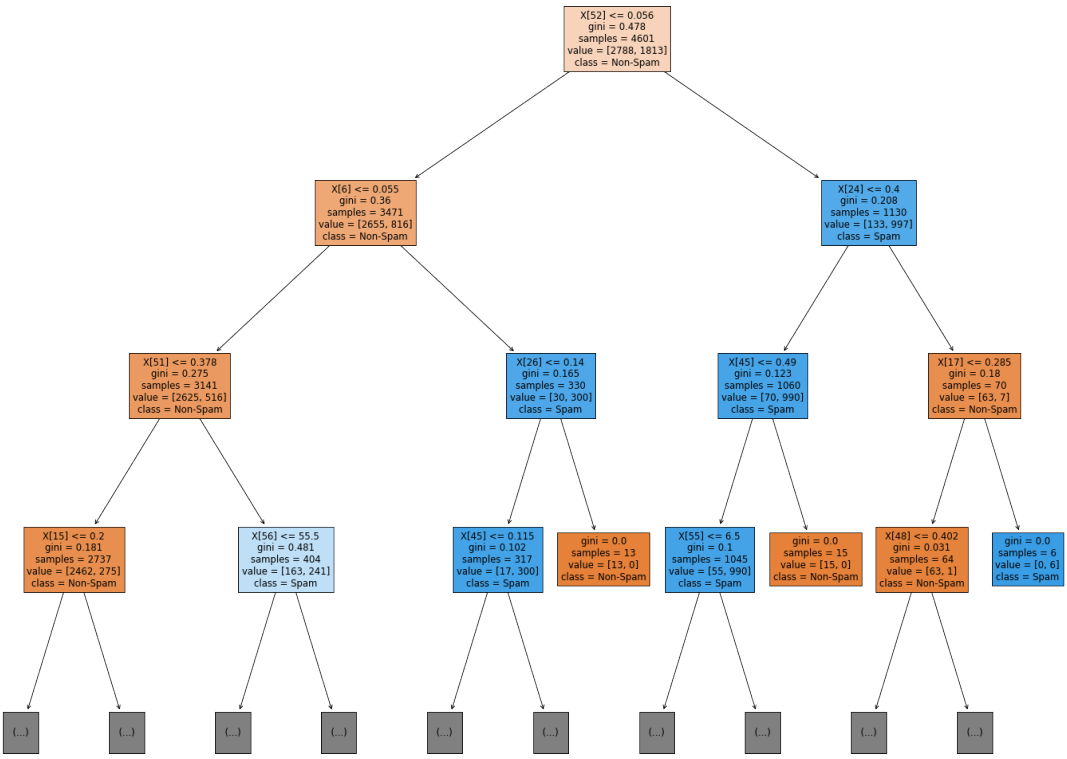
The task for this analysis is to build a spam classifier using the UCR email spam dataset https://archive.ics.uci.edu/ml/datasets/Spambase came from the postmaster and individuals who had filed spam. Please download the data from that website. The collection of non-spam emails came from filed work and personal emails, and hence the word 'george' and the area code '650' are indicators of non-spam. These are useful when constructing a personalized spam filter. You are free to choose any

### (a) Build a CART model and visualize the fitted classification tree.

```
Missing Values in Dataset : False
```

```
DecisionTreeClassifier(random_state=123)
```






**(b) Building a random forest model. Randomly shuffling the data and partitioning to use 80% for training and the remaining 20% for testing. Compare and report the test error for your classification tree and random forest models on testing data. Plot the curve of test error (total misclassification error rate) versus the number of trees for the random forest, and plot the test error for the CART model (which should be a constant with respect to the number of trees).**

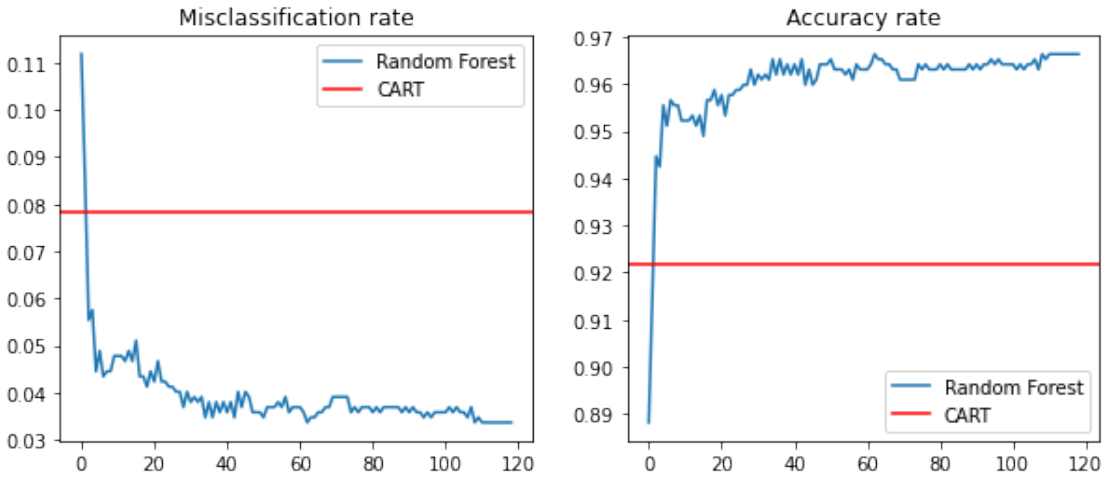
```
Accuracy for CART is :  0.9218241042345277

Confusion Matrix is :
[[522  31]
 [ 41 327]]
Misclassification rate for CART is :  7.81758957654723
```

**A random forest model was fit on the same data set. The data was partitioned to 80% training and 20% for testing. The same methodology was applied in building a CART classifier as well. The accuracy and misclassification rates for each model were calculated and plotted. It was observed that random forest model performed**

**better with a greater number of trees. A single CART classifier had a constant misclassification rate of 7.8% (accuracy of 92%). Below is the plot of accuracy rate and misclassification rate for both the models.**

```
Text(0.5, 1.0, 'Accuracy rate')
```




**(c) Now we will use a one-class SVM approach for spam filtering. Randomly shuffle the data and partition to use 80% for training and the remaining 20% for testing.Extract all non-spam emails from the training block (80% of data you have selected) to build the one-class kernel SVM using RBF kernel (you can turn the kernel bandwidth to achieve good performance). Then apply it on the 20% of data reserved for testing (thus this is a novelty detection situation), and report the total misclassification error rate on these testing data** A one-class SVM approach was used for spam filtering. With a similar test/train split, all non-spam messages were filtered to constitute the training data for a one-class RBF kernel model. The 20% test data (including the spam messages were applied on this model to identify the misclassification error rate. This would be of use as a novelty detection technique. Refer to the code for implementation steps. The mis-classification error rate is 36.6%

```
[(0.05, 0.6330076004343105),
 (0.1, 0.5407166123778502),
 (0.5, 0.4256243213897937),
 (1.0, 0.4223669923995657)]

Misclassification rate is :  0.3669923995656895
```