**Lab1- Decision Tree vs Random Forest**

**Introduction**

The aim of this experiment is to build and compare Decision tree model and Random Forest model. Both these models would be used to predict the quality of wine. The model is build using a csv file containing 4898 samples with 11 features and one class label. In total there are 6 classes (6 different quality levels).

**Method**

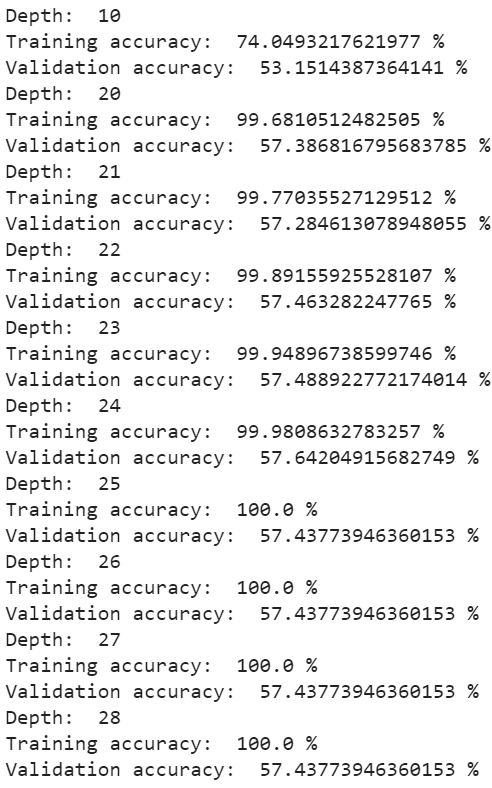
Each of the features helps to classify a sample into a quality grade, so the features are stored as an x variable and the quality is stored as a y variable. The train x and the train y were used to fit the Decision Tree Classifier model with the criterion set to entropy. The decision tree algorithm uses the criterion to make feature selection, which are then sued to compute the importance of each feature and build the model. There are two possible criterions for the Decision Tree model, they are, gini and entropy. The gini criterion is used by default, it measures impurity by measuring how often an element chosen randomly from the data set would be labeled incorrectly. It is essentially a probability. For lab one I used the entropy **criterion. Entropy is essentially used for information gain. It helps the algorithm decide which feature to split at every step while building the tree. ( Source:** <https://en.m.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity> ). In this lab I used the entropy criterion as it clearly split the data into the desired labels. The stopping criteria used was max\_depth. To tune the model, it is important to have a stopping criteria. Max depth limits the maximum number of levels the tree can have, in order to decide the max depth value a for loop which builds the decision tree with a range of depths was used. For each of the depth the training and validation accuracy was calculated. The depth for which the validation accuracy was highest was used as the max\_depth value to build the tree. However, it is important to note that there are many options for stopping criteria, some of them are: min\_samples , min\_samples\_leaf, max\_leaf\_nodes, min\_impurity\_split and more.

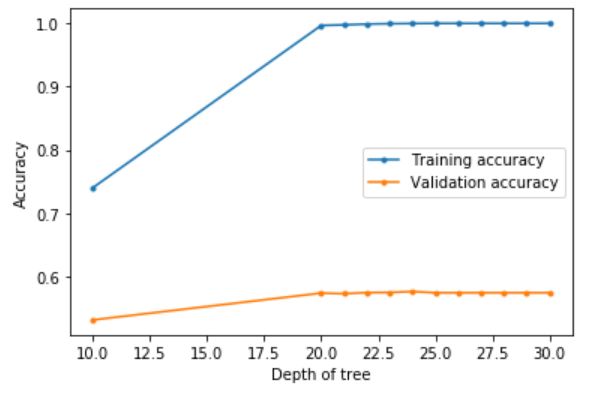
The second part of this lab was to build a random forest. A random forest is similar to decision tree one of the major differences between the two methods is that a random forest build multiple decision trees instead of just one tree. To tune the random forest randomization techniques, need to be applied. My program used the combination of random\_state and estimator value as a randomization technique to improve the diversity of the fitted trees. The method I used to tune the model is very similar to that used in decision tree. The main difference is that instead of the depth the optimum n\_estimator is tuned. The n\_estimator sets the number of trees in the forest. If this value is set too high there is a good chance of overfitting. The second technique was to set the random state to 0. Random state controls the randomness of the bootstrapping of the samples and the sampling of the features when deciding the best split (Source: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> ) . By setting it to 0 , the random number generator seeds the data by 0 (which is a common value used). N\_estimaor acts as a stopping criterion as it limits the number of trees produced. Additionally, the parameter min\_impurity\_decrease (with default = 0) helps to decide weather to create a split or not this again helps to set a threshold for stopping tree growth.

**Experiments:**

First step before building the model was to split the data into training and test. To do so the 80-20 rule was used, where 80% of the data was used as the training data and the rest 20% was used as the testing data. For this dataset, there were 3919 samples for training and 980 samples for test data.

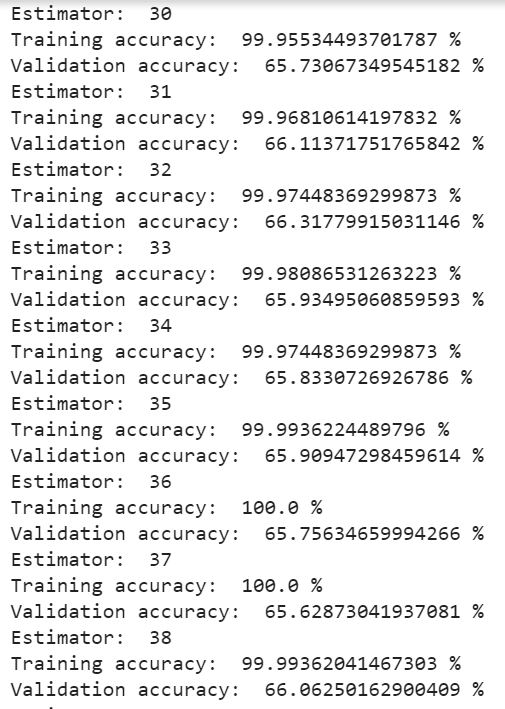
Next the parameters for decision tree were tuned to find the max\_depth at which the validation accuracy was at its highest. Below is a picture of the results of the each of the selected max\_depth values.

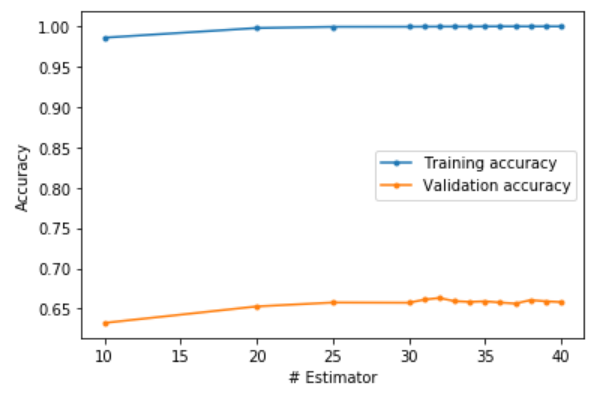




As the plot above and the picture of the validation scores show, the best max\_depth is 24. As the plot shows when the depth of the tree is increased from 10 to 20 there is a significant jump in validation accuracy value, but after 20 , the value of the validation increases till 24. At depth 25 the validation as reached its peak and the train accuracy is at 100% which could suggest at depth 25 and above the tree will overfit.

For the random forest a similar tuning approach was used.

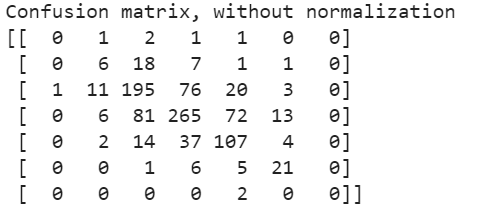




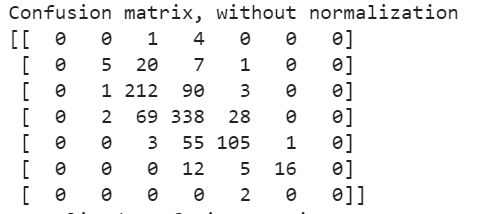
The best estimator was at 32 with the validation accuracy of 66.3178%, just like the decision tree model the validation accuracy increases at first significantly will estimator value of 30 after which it increases only marginally and even decreases. Another interesting observation is that training accuracy reaches 100% then decreases. This again shows that after estimator value of 32 the data is overfitted.

In order to compare the two classifiers, the mean accuracy was used. The mean value of the decision tree was 57.3485% and the mean for random forest was 65.6616%. This shows that on average the random forest was more accurate than the decision tree model.

Below is the confusion matrix for decision tree



Below is the confusion matrix for random forest



When comparing the two matrix it is clear the number of samples for which the model was accurate is higher for random forest. Confusion matrix compares the predicted data to the actual data to show how accurate the model is.

Next the visuals for each of the models were plotted (they both can be found in my lab1.ipynb file). Both plots of the trees shows that the first split was on the density. The first split rule for both the plots was if the density <=0.992. However, the main difference between the two tree is that at depth 3 the random forest tree is able to identify if a sample belongs to class 6 or 7 but the decision tree is still splitting for class 6. This shows that the random forest tree is able to classify samples not only more accurately but also faster than decision tree. To get a more detailed overview of the slit rule for each non-leaf and the class for each leaf look at the plots in the notebook.