Col 774: Assignment3

Vivek Singh

2019MCS2574

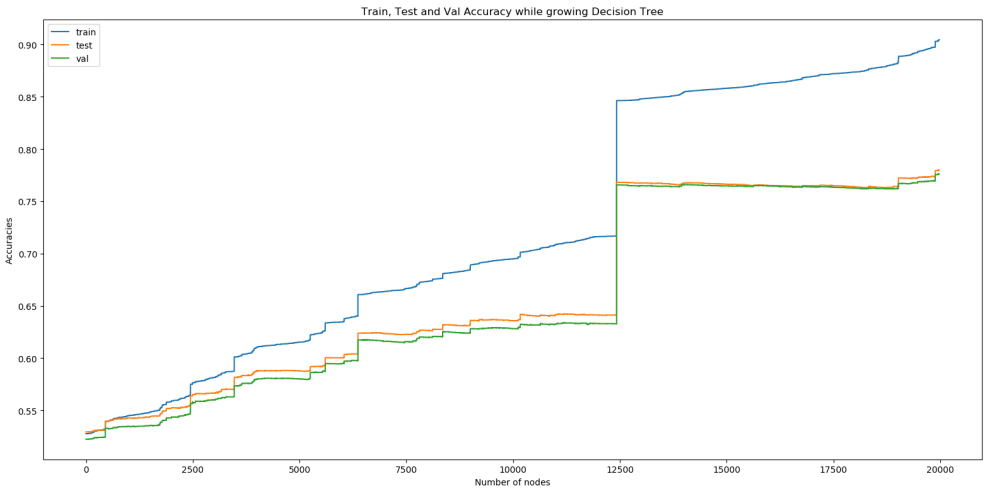
1 Decision Trees (and Random Forests)

1.1 Decision tree algorithm for predicting the virus infected files based on a variety of features

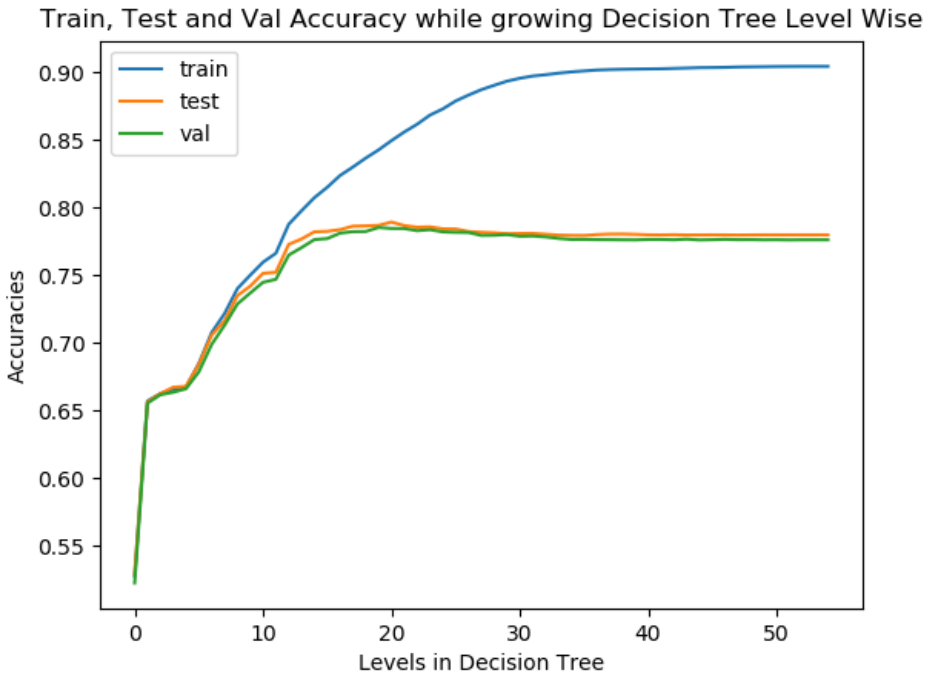
- The Virus data-set contains 482 different parameters used to distinguish as infected executable or not.
- Since all the attributes variable are continuous. So for any given attribute at a node, for two way split we are using median of that attribute.
- At a Decision node we will choose a attribute which results in maximum mutual information gain for a class variable.
- To check the impurity at each node we are calculating the entropy to gain knowledge of uncertainty.
- Split the data-set at each node which provides maximum mutual information gain.
- We Create Decision tree recursively. Each time a new node is added to the tree, we calculate the accuracy over train, test and validate dataset.
- Final accuracy over the data set after fully grown tree are:

Training	Validation	Testing
90.439 %	77.619 %	77.984 %

- **Recursively grown tree Graph showing the Nodes VS Accuracy**:



- **Level-Wise grown tree Graph showing the Levels VS Accuracy**:



Total Nodes	Decision Nodes	Leaf Nodes
19977	9988	9989

- **Comment/Observation :**

  As we keep adding new nodes to the tree accuracy of all three data-set i.e Train, Test and Validation increases. But after a point accuracy over Test and Validation data-set starts decreasing giving us the point of over-fitting.

  As we grow the tree by fitting on training data-set it starts to over-fit on training set which can be visualized in both the graphs plotted above.

  This is because we are growing the tree till we get the best entropy at each node according to the training data-set which lead to overfit.

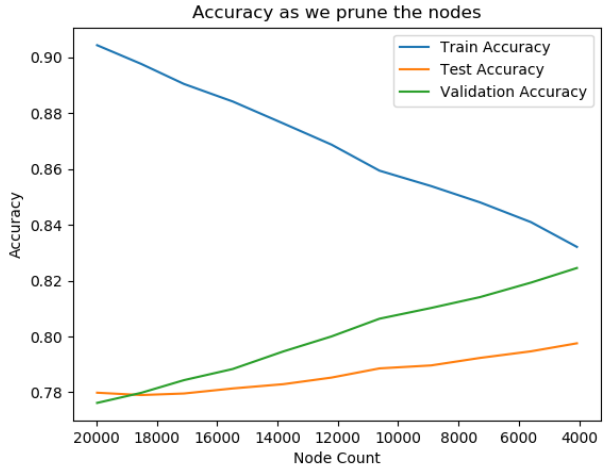
  This fully grown tree does not represent a better estimate of accuracy achieved on a data-set due to high overfit on training data.

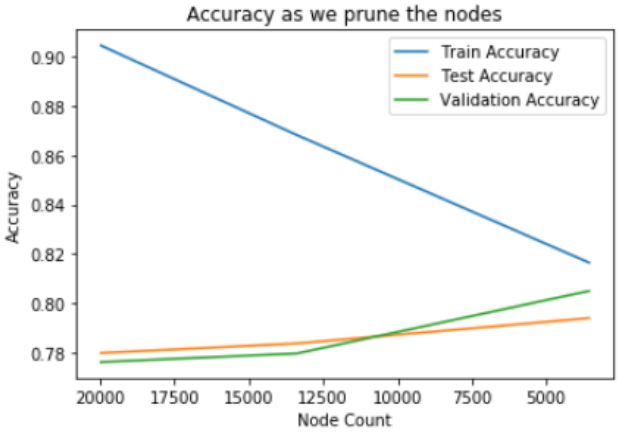
1.2 Decision Tree Post Pruning

- Pruning the tree post growing it fully.
- Pruning is one of methods to reduce over-fit in the data.
- We greedily prune away a node at a time, whose removal gives us best accuracy on validation data.
- Pruning stops when the accuracy gain on validation set is zero.
- This helps in reducing the over-fitting as this validation data was not used to build the tree.
- Following is the graph for Nodes Vs Accuracy as we prune the tree.

  Left Figure: Pruned Bottom-Up

  Right Figure: Pruned Iterative.





- Accuracy over the data after pruning the decision tree are:

Tree	Training	Validation	Testing	Total Nodes
Fully Grown: Tree	90.439 %	77.619 %	77.984 %	19977
Pruned Tree: Bottom Up	83.21 %	82.45 %	79.75 %	4079
Pruned Tree: Iterative	81.64 %	80.49 %	79.39 %	3559

- **Comment/Observation :**

  Here we can see that as we remove the nodes and subtree below that node, we are able to reduce the over-fit on training data-set.

  Pruning also results in increase of accuracy over Test data-set as well as validation data-set. Which can be seen in the table above.

  Hence a pruned tree gives us better estimate of the accuracy achieved over the data-set rather than fully grown tree.

1.3 Random Forests

- Used sklearn.ensemble.RandomForestClassifier for this part.
- Used sklearn.model_selection.GridSearchCV for calculating the best parameter.
- Parameters which are used for grid search are:

  n_estimators: [50, 150, 250, 350, 450]

  max_features: [0.1, 0.3, 0.5, 0.7, 0.9]

  min_samples_split: [2, 4, 6, 8, 10]

- This part was executed on Google Colab, 35 GB RAM and 40 cores.
- Used 3-fold Cross validation
- 125 candidates, totalling 375 fits in grid-search.
- Total time taken to calculate the optimal Parameters : **108 min**
- **Optimal parameters :**

  n_estimators: 350

  max_features: 0.1

  min_samples_split: 10

- **Accuracies :**

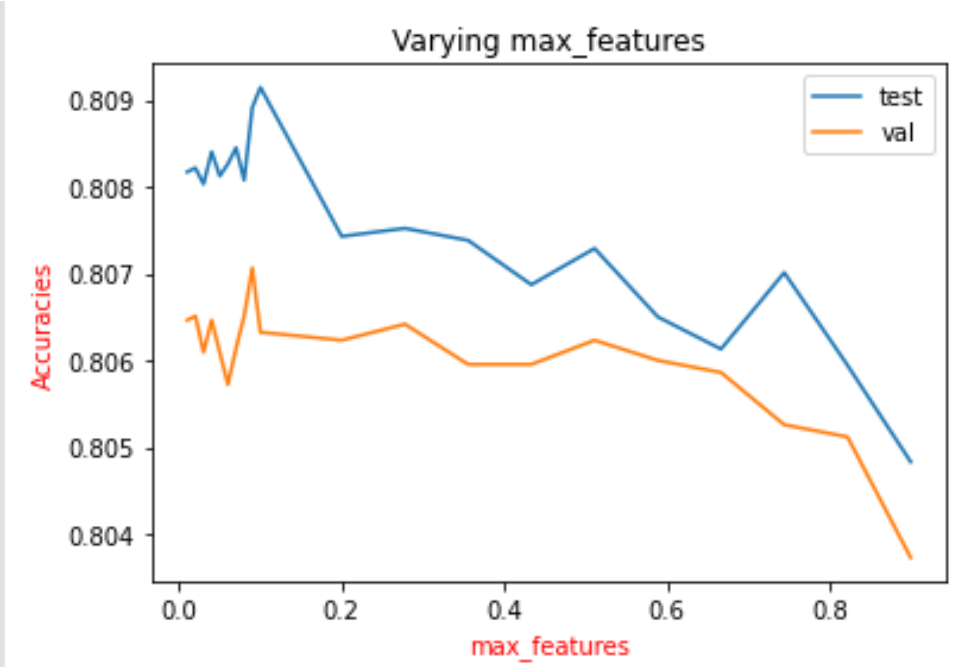
Training	Out-Of-Bag	Validation	Testing
87.37 %	81.05 %	80.70 %	80.90 %

- **Comparing with pruned tree:**

Model	Training	Validation	Testing
Random-Forest	87.37 %	80.70 %	80.90 %
Pruned DT	83.21 %	82.45 %	79.75 %

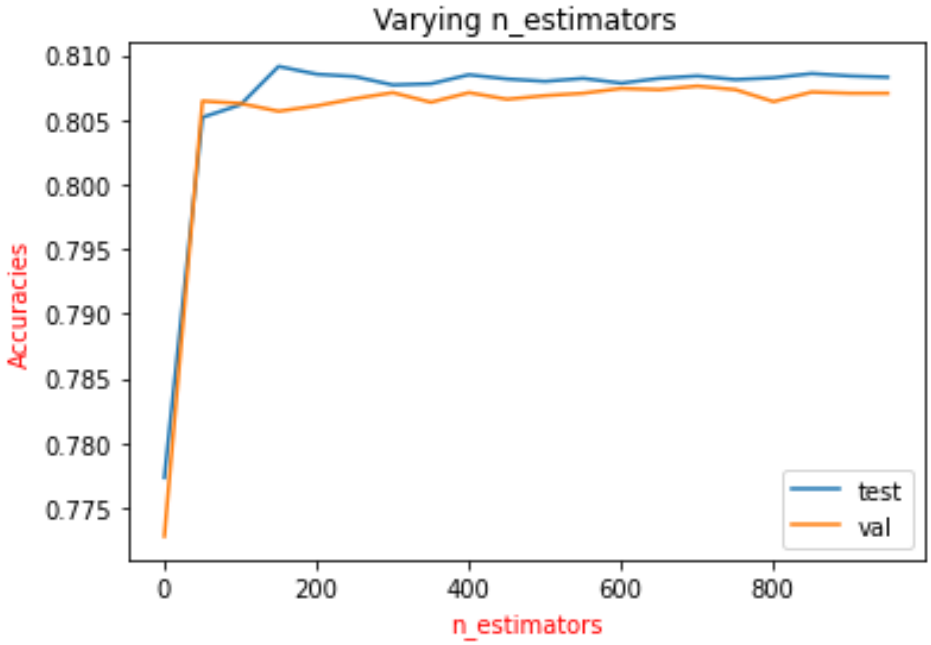
1.4 Random Forests - Parameter Sensitivity Analysis

- Varying parameter : **max_features**



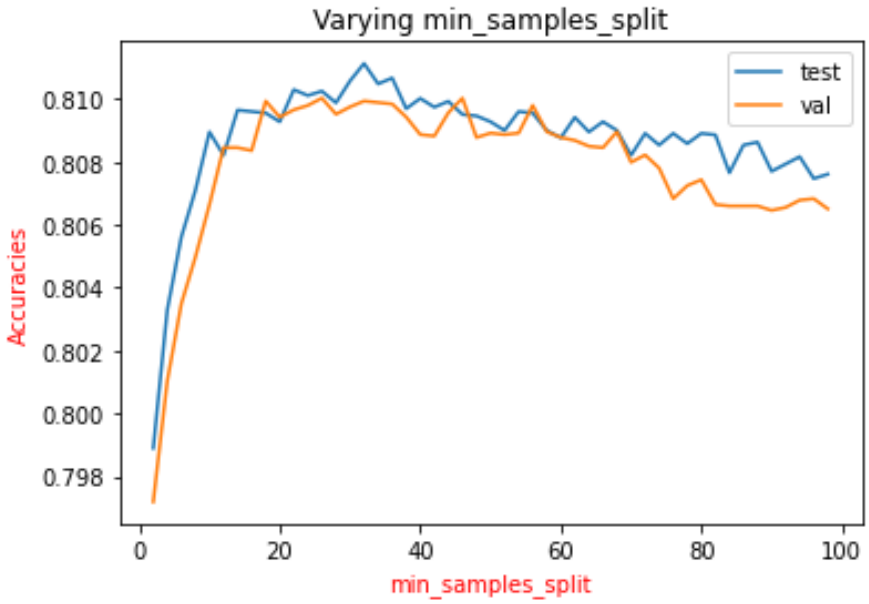
- **COMMENT :**
- max_features : The number of features to consider when looking for the best split.
- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- So we can observe from the graph that before max_features is < 0.1 Accuracy is changing arbitrarily.
- At max_feature $= 0.1$, we achieve maximum accuracy. 0.1*482 i.e 48 features considered.
- After that when max_feature > 0.1 we see a decline in accuracy.
- This result is also confirmed by GridSearch.
- Selecting random feature for each split ensures the randomness of RandomForest required to create a ensemble technique.

- Varying parameter : **n_estimators**



- **COMMENT :**
- n_estimators : The number of trees in the forest.
- As n_estimators = 1 it behaves just as Decision tree.
- But as we increase the number of trees in forest we can see exponential rise in the test an validation accuracy.
- The increase signifies the ensemble advantage over decision tree.
- As we keep increasing the number of trees it get saturated and no significant accuracy gain is achieved.

- Varying parameter : **min_samples_split**



COMMENT :

- min_samples_split : The minimum number of samples required to split an internal node.

- If we keep sample very small i.e 2, we achieve very low accuracy.
- As we keep increasing the samples we notice increase in accuracy.
- Range 20 to 30 seems to give the best result.
- After 30, we can observe a general decrease in accuracy.
- The above results can be explained as if samples as very small then we are unnecessary splitting the nodes and nodes with higher samples we are moving toward majority class.