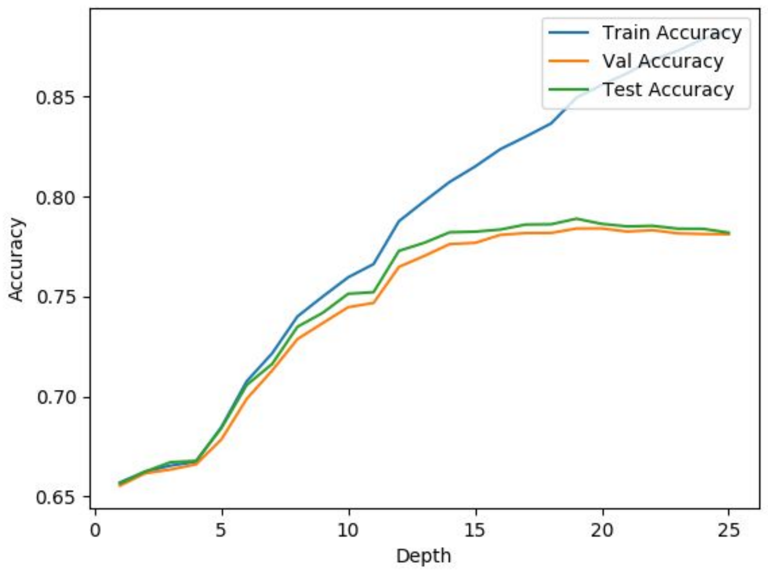
Machine Learning A3: Decision Trees

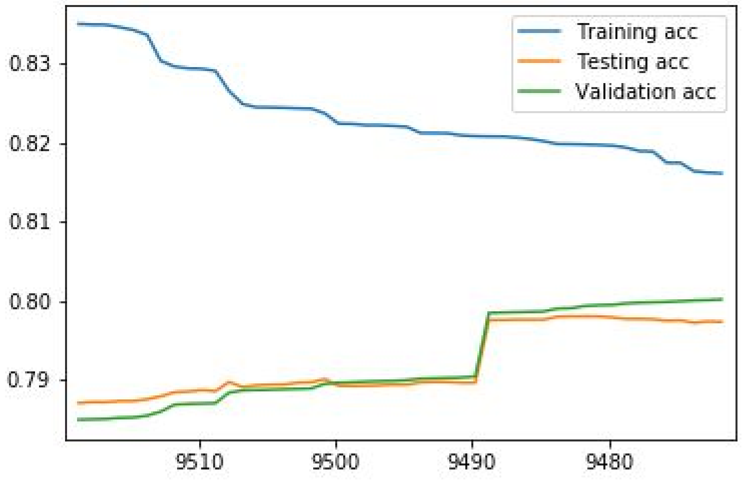
(A) Construction of DTree



- Increasing the number of tree nodes, hence the depth, increased the test and val accuracy, but **after a certain depth**, the tree overfits and testing and validation accuracy decreases.
- Although Training accuracy still increases, since the model is over-fitting on the training data set.
- **Maximum depth** of the tree was found to be 53.

- Accuracies:
  Train accuracy 0.8732557600482129
  Train accuracy 0.7785916276482314
  Val accuracy 0.7781846838494344

(B) Decision Tree Post Pruning



- The plot shows accuracy over data sets v/s decreasing number of nodes.
- Post pruning was performed by iterating over each node and its corresponding subtree in a bfs manner.
- There are improvements across pruning stages but test and valid accuracy **could be increased more than it did**. The setbacks in my program were that the decision tree implementation was using much higher time to build a full tree (**57.04122321 minutes**). Thus, I had to set my min_sample_split = 10. Would it be less than 10 (perhaps default=2), the improvement in accuracy would have been much higher. The problem statement dictates to post-prune the tree. **On the contrary if I was pruning at the time of tree construction, it would save a lot of time.**
- **The optimal depth** obtained after post-pruning the decision tree was **21**. Thus pruning decreased the tree size to a great amount.

Best Validation Accuracy before pruning: 0.783

Best Validation Accuracy after pruning: 0.8001

Best Testing Accuracy after pruning: 0.787

Best Testing Accuracy after pruning: 0.7973

(C) Random Forest

Train Accuracy 0.863721354287392

Test Accuracy: 0.8070647135175227

clf = RandomForestClassifier(n_estimators=250, criterion='entropy', max_depth=26, random_state=42, oob_score=True, n_jobs=-1, min_samples_split=5, min_samples_leaf=5, max_features=None)

Sklearn GridSearchCV

Following set of out of the bag parameters were used in the grid search:

```
parameters = [{
            'max_depth': [i for i in range(15, 20)],
            'min_samples_split': [i for i in range(2, 8)],
            'max_features': ('log2', 'sqrt'),
            'n_estimators': [150, 250]
          },
          {
            'max_depth': [i for i in range(20, 30)],
            'min_samples_split': [i for i in range(8, 12)],
            'max_features': ('log2', 'sqrt'),
            'n_estimators': [150, 250]
          }
         ]
```

Optimal set of params:

Max_depth: 26
Min_samples_split: 5
Max_features: ‘sqrt’
N_estimators: 150

{'max_depth': 29, 'max_features': 'sqrt', 'min_samples_split': 8, 'n_estimators': 250}

Accuracy over optimal set of hyper-parameters:

Val Accuracy: 0.81924373
Test Accuracy: 0.80812422

Observations:

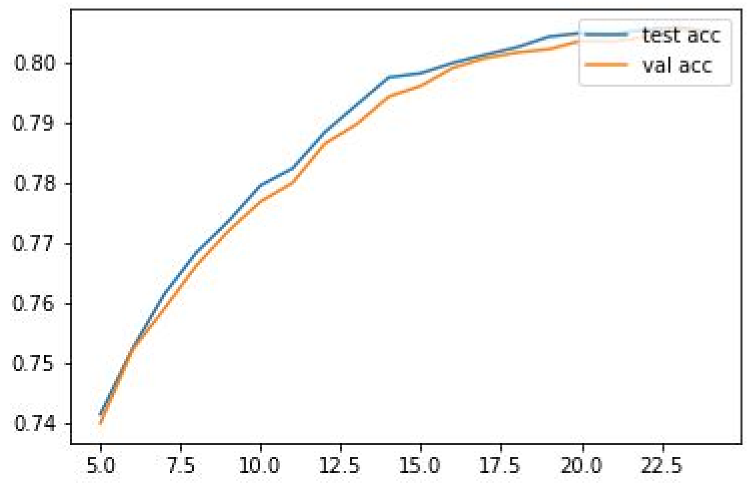
After using gridSearchCV, the test and validation accuracies improved but only on few cases (15 in test and 19 in validation data set).

The only difference (high level) in post pruning and GridSearchCV was the model's internal structure.The accuracies are almost similar.

Using GridSearchCV made the classification much faster than post pruning an entire tree.

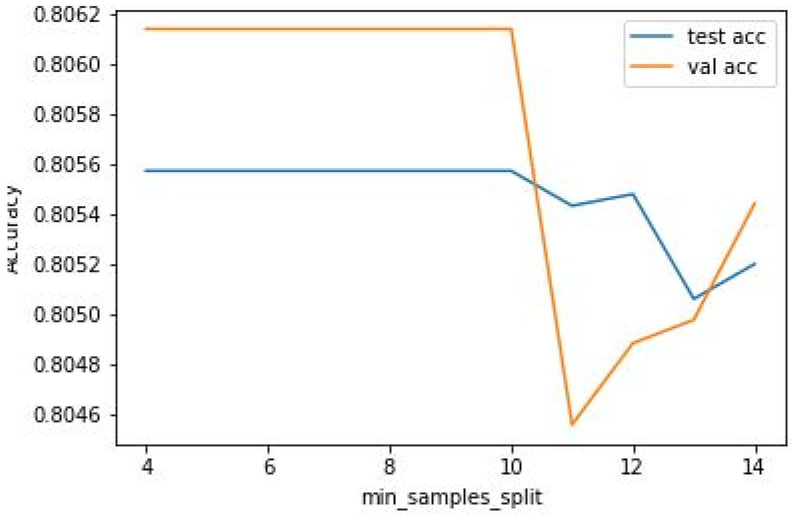
(D) Sensitivity Analysis

(1) Changing max_depth



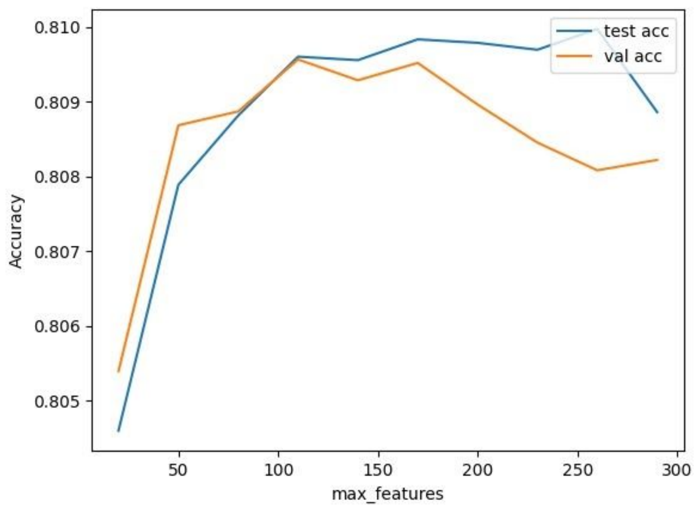
- max test accuracy: 0.8057113717491076 | Depth at max accuracy: 23
- max validation accuracy: 0.8054422399406638 | Depth at max accuracy: 23
- The model is sensitive enough, when max_depth is variable. The effect on accuracy can be approximately fitted as a logarithmic function of max_depth, w.r.t a particular scale.

(2) Changing min_samples_split



- max test accuracy: 0.8055722961383339 | Sample split value: 4
- max validation accuracy: 0.8061375857593176 | Sample split value: 4
- After a certain value of min_sample_split, the accuracy is changing variably but average accuracy is not affected as long as min_sample_split is less than a particular value, in this case, around 15.
- After 15, we should split the node.

(3) Changing max_features



- max test accuracy: 0.8099763571461684 | Optimal max_feature value: 260
- max validation accuracy: 0.8095679584646764 | Optimal max_feature value: 110

- The test and validation data are sensitive enough to max_features. Thus its extremely careful to tune the max_features carefully.