1. **Random Forest and AdaBoost.M1**
2. **Random Forest**

Hyper – Parameter Values:

No. Trees: 100

Overall Accuracy: For Random Seed

Highest – 0.78

Average – 0.767

Confusion Matrix:

[[216 22]

[ 48 15]]

Experimental Results:

1. The overall accuracy was less when values of No. of trees were 10, 20, 50, 70, 80, 90, 95, 105, 110, 130, 150, 200, 250.
2. After 150, the training accuracy remained constant.

1. **AdaBoost.M1**

Hyper – Parameter Values:

n\_estimator = 210

learning\_rate = 0.9

Overall Accuracy: For Random Seed

0.827

Confusion Matrix:

[[227 11]

[ 41 22]]

Experimental Results:

1. The overall accuracy was less when values of n\_estimator were 10, 20, 30, 50, 70, 80, 85, 90, 200, 205, 215, 220, 230, 250, 270, 300, 400, 500, 600
2. The overall accuracy was less when values of Learning rate were 0.1, 0.2, 0.3, 0.5, 0.7, 0.8, 0.85, 0.95, 0.97, 1. The accuracy started decreasing as value was decreased below 0.9
3. **AdaBoost using Decision Stumps gives better overall and class conditional accuracy than RandomForest**
4. **NN, KNN, Logistic Regression, Naïve Bayes, Decision Tree.**
   1. **NN**

Hyper – Parameter Values:

solver='adam'

activation='relu'

alpha=0.001

max\_iter=200

early\_stopping=True

hidden\_layer\_sizes=(100,3)

random\_state=1

Overall Accuracy: For Random Seed

0.8073089700996677

Confusion Matrix:

[[208 30]

[ 28 35]]

Experimental Results: (Tried Values)

1. Solver = ‘sgd’,
2. Activation = relu, logistic
3. Alpha = 0.00001, 0.001
4. Early\_stopping = False
5. Hidden\_layer = (200, 3), (200, 2), (200, 4), (100, 2), (100, 4), (50, 3)
   1. **KNN**

Hyper – Parameter Values:

n\_neighbors = 5

Overall Accuracy: For Random Seed

0.7973421926910299

Confusion Matrix:

[[227 11]

[ 50 13]

Experimental Results: (Tried values)

1. n\_neighbors = 2, 4, 8, 10, 20, 100, 50
   1. **Logistic Regression**

Hyper – Parameter Values:

penalty='l2'

tol=0.01

Overall Accuracy: For Random Seed

0.8172757475083057

Confusion Matrix:

[[229 9]

[ 46 17]]

Experimental Results: (Tried values)

1. penalty='l1'
2. tol=0.001, 0.03, 0.005
   1. **Naïve Bayes**

Hyper – Parameter Values:

None

Overall Accuracy: For Random Seed

0.8006644518272426

Confusion Matrix:

[[226 12]

[ 48 15]]

* 1. **Decision Tree**

Hyper – Parameter Values:

max\_depth = 3

max\_features = 4

Overall Accuracy: For Random Seed

0.813953488372093

Confusion Matrix:

[[226 12]

[ 48 15]]

Experimental Results: (Tried values)

1. Max\_depth = 10, 20, 1, 2, 5
2. Max\_featuers = 3
   1. **Voting Classifier – Unweighted**

Hyper – Parameter Values;

Voting = ‘hard’

Overall Accuracy:

0.8305647840531561

Confusion Matrix:

[[227 11]

[ 40 23]]

Experimental Results:

1. Voting = soft
   1. **Voting Classifier – Weighted**

Hyper – Parameter Values;

Voting = ‘hard’

Weights=[1,1,2,1,2]

Overall Accuracy:

0.8305647840531561

Confusion Matrix:

[[227 11]

[ 40 23]]

Experimental Results:

1. Voting = soft
2. Weights = [1,1,3,1,2], [1,1,3,1,1]
   1. Result Discussion:
      1. Among all the different models, Logistic Regression gave the highest accuracy of 0.817
      2. However when all models were combined in an unweighted ensemble model, we got an even better result of 0.8305
      3. When weights corresponding to the accuracy of individual models was applied to the ensemble model, the accuracy improved to 0.8338
3. **Voting Classifier with 7 models**
   1. **Voting Classifier – unweighted**

Hyper – Parameter Values;

Voting = ‘hard’

Overall Accuracy:

0.8205980066445183

Confusion Matrix:

[[227 11]

[ 40 23]]

Experimental Results:

1. Voting = soft
   1. **Voting Classifier – weighted**

Hyper – Parameter Values;

Voting = ‘hard’

Weights = [1,1,2,1,2,1,3]

Overall Accuracy:

0.8272425249169435

Confusion Matrix:

[[227 11]

[ 40 23]]

Experimental Results:

1. Voting = soft
2. Weights = [1,1,2,1,3,1,3], [1, 1, 3, 1, 3, 1, 3]
   1. Result Discussion
      1. The overall accuracy without weights decreased due to the low accuracy of Random forest. However, the high accuracy of AdaBoost helped to have only a small difference
      2. The weights helped to increase the accuracy to 0.827

**References:**

1. https://machinelearningmastery.com/ensemble-machine-learning-algorithms-python-scikit-learn/

2. http://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion\_matrix.html

3. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.score

4. https://towardsdatascience.com/random-forest-in-python-24d0893d51c0

5. http://scikit-learn.org/stable/auto\_examples/ensemble/plot\_adaboost\_twoclass.html

6. https://machinelearningmastery.com/binary-classification-tutorial-with-the-keras-deep-learning-library/

7. https://towardsdatascience.com/logistic-regression-using-python-sklearn-numpy-mnist-handwriting-recognition-matplotlib-a6b31e2b166a

8. http://scikit-learn.org/stable/modules/generated/sklearn.linear\_model.LogisticRegression.html

9. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html