Intro to Machine Learning [EE769]

Assignment 1

Report by Sufiyan Adhikari (173190009)

https://github.com/dumbPy/Intro_to_ML-EE769-

Pre-processing:

After importing the test.csv file into pandas dataframe object **df**, **df.isnull().any()** is used to see which feature columns had NaN values.
df.dropna() was used to drop all rows with any NaN values in order get the most accurate data.
df.fillna('method=ffill') was also tried that fills be last seen value of that feature in place of Nan Values. But the data becomes less acurate, hence not used finally.

Other tried method was to find categorical columns and to search for non-categorical type elements in them, like *float* that were actually Nan values in these categorical columns. Complete rows with any such non-string or non-categorical (float) entry in categorical (type *str* for 1st element) columns were deleted. On discovering the above stated method ***df.dropna()***, this method was deemed unnecessary and hence not used in final code.

Classifiers Tried:

Random Forest Classifier was tried and worked well. Cross Validation accuracy after feature dropped to 10 features and hyperparameter tuning was around 95-96%

Gradient Boosting Classifier was tried and worked well. Cross Validation accuracy after feature dropped to 10 features and hyperparameter tuning was around 97-98%

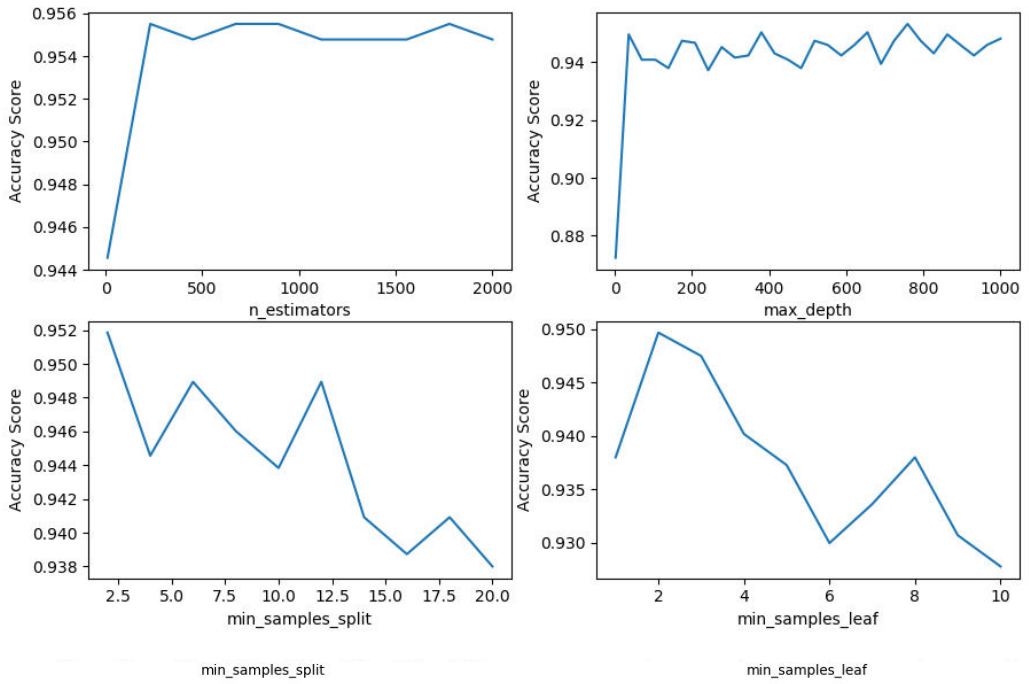
Multi Layer Perceptron with default 1 hidden layer with 100 nodes was used and had the best accuracy of about 97-98% in CrossValidation by Kfold method with k=5.

Support Vector Classifier was tried and had the worst Cross Validation accuracy of about 55%. No Hyperparameter tuning was tried. Gamma = np.exp(-16.75) was found to be best by manually trying a few ranges and manual binary searching the value. Still not up to other classifiers, hence not considered in test.py

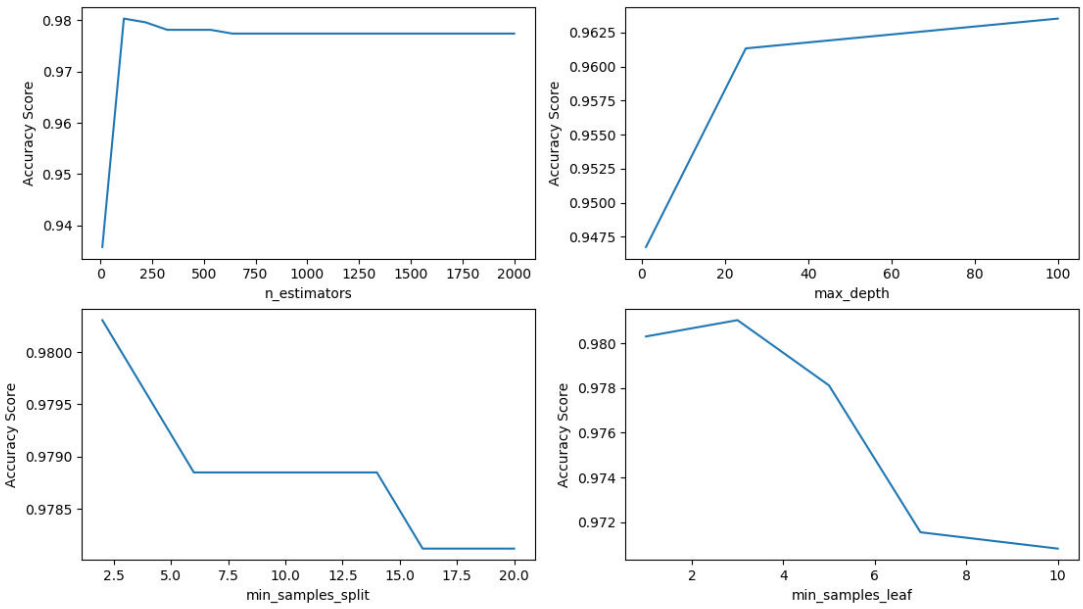
Decision Tree Classifier had an accuracy of about 96% with 10 features selected by *classifier.feature_importances_* from Gradient Boosting Classifier. No hyperparameter tuning tried for Decision Tree Classifier.

HyperParameter Tuning:

Hyperparameter tuning was tried for **Gradient Boosting Classifier** and **Random Forest Classifier**. The Cross Validation accuracy was seen to increase by a few percentage after hyperparameter tuning as well as feature drop.



Random Forest Classifier Hyperparameter Tuning Graphs



Gradient Boosting Classifier HyperParameter Tuning Graphs

Outputs

train.py

```
tinkerman@dumbpy:~/Documents/Intro to ML [EE769]$ python3 train.py
**********
Model = Random Forest Classifier
**********
**********
Features Selected ::  105
Train Accuracy    ::  0.995437956204
Test Accuracy     ::  0.883636363636
Average CrossValidation Score of 5.00 runs: 92.12254

**********
     Dropping Features by feature_importances_
        Training after Feature Drop
**********
Features Selected ::  10
Train Accuracy    ::  0.998175182482
Test Accuracy     ::  0.938181818182
Average CrossValidation Score of 5.00 runs: 94.82358

Tuning Parameters.. ETA 1Mins... Wait....
**********
Best Parameters decided by Parameter Tuning:

         Parameter  Value
0      n_estimators    231
1      max_features   sqrt
2         max_depth    759
3  min_samples_split      2
4   min_samples_leaf      2
5         bootstrap  False
Training Model after Feature Drop and Hyperparameter Tuning
**********
Features Selected ::  10
Train Accuracy    ::  0.999087591241
Test Accuracy     ::  0.963636363636
Average CrossValidation Score of 5.00 runs: 95.77116

**********
Model = Gradient Boosting Classifier
**********

**********
Features Selected ::  105
Train Accuracy    ::  0.999087591241
Test Accuracy     ::  0.970909090909
Average CrossValidation Score of 5.00 runs: 97.66580
```

```
**********************************************************
     Dropping Features by feature_importances_
        Training after Feature Drop
**********************************************************
Features Selected ::  10
Train Accuracy    ::  0.997262773723
Test Accuracy     ::  0.978181818182
Average CrossValidation Score of 5.00 runs: 98.02944

Tuning Parameters.. ETA 2Mins... Wait....
**********************************************************
Best Parameters decided by Parameter Tuning:

          Parameter Value
0       n_estimators   114
1       max_features  auto
2          max_depth    50
3  min_samples_split    10
4   min_samples_leaf     5
Training Model after Feature Drop and Hyperparameter Tuning
**********************************************************
Features Selected ::  10
Train Accuracy    ::  0.997262773723
Test Accuracy     ::  0.989090909091
Average CrossValidation Score of 5.00 runs: 98.10270


  Training on 10 best Features from Here On.

**********************************************************
                  Model = SVC
**********************************************************
Train Accuracy    ::  54.9270072993
Test Accuracy     ::  49.4545454545
Average CrossValidation Score of 5.00 runs: 53.82937

**********************************************************
Model =Decision Tree Classifier
**********************************************************
Features Selected ::  10
Train Accuracy    ::  100.0
Test Accuracy     ::  96.7272727273
Average CrossValidation Score of 5.00 runs: 96.64336

**********************************************************
      Model = Multi Layer Perceptron
**********************************************************
Features Selected ::  10
Train Accuracy    ::  97.8102189781
Test Accuracy     ::  97.8181818182
Average CrossValidation Score of 5.00 runs: 97.51875
```

OUTPUT
test.py with dummy gt.csv

```
tinkerman@dumbpy:~/Documents/Intro to ML [EE769]$ python3 test.py
**********************************************************
Model = RandomForestClassifier
Test Accuracy     ::  0.498286497601
**********************************************************



**********************************************************
Model = GradientBoostingClassifier
Test Accuracy     ::  0.496915695682
**********************************************************



**********************************************************
Model = MLPClassifier
Test Accuracy     ::  0.502398903358
**********************************************************



Best Classifier by Accuracy Score : MLPClassifier
**********************************************************
Model = MLPClassifier
Test Accuracy     ::  0.502398903358
**********************************************************



Prediction of MLPClassifier written to out.csv
```