MODEL PERFORMANCE TESTING

Date	15 November 2022
Team ID	PNT2022TMID32563
Project Name	Web Phishing Detection
Maximum Marks	4 Marks

Performance Metrics

S.No	Parameter	Values
1.	Metrics	**Classification Model:** **Random Forest Classifier** Accuracy Score-95%
2.	Tune the model	Hyperparameter Tuning-97% Validation Method-KFOLD& Cross Validation Method

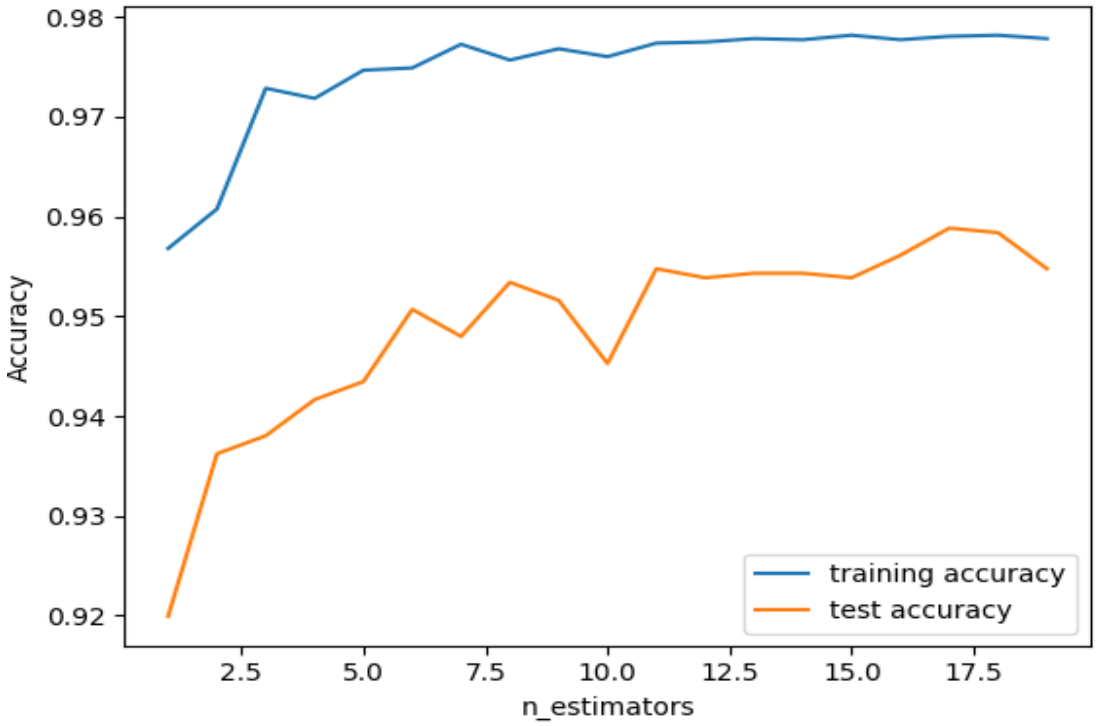
1.METRICS:

CLASSIFICATION REPORT:

```
In [54]: #computing the classification report of the model

         print(metrics.classification_report(y_test, y_test_forest))
```

```
              precision    recall  f1-score   support

          -1       0.96      0.99      0.97      1933
           1       0.87      0.70      0.78       278

    accuracy                           0.95      2211
   macro avg       0.92      0.84      0.88      2211
weighted avg       0.95      0.95      0.95      2211
```

PERFORMANCE:



Out[92]:

	ML Model	Accuracy	f1_score	Recall	Precision
0	Logistic Regression	0.884	0.264	0.133	0.585
1	K-Nearest Neighbors	0.943	0.778	0.902	0.898
2	Support Vector Machine	0.916	0.526	0.457	0.964
3	Naive Bayes Classifier	0.559	0.362	0.993	0.242
4	Decision Tree	0.949	0.785	0.869	0.968
5	Random Forest	0.950	0.778	0.864	0.956
6	Gradient Boosting Classifier	0.945	0.750	0.756	0.921
7	CatBoost Classifier	0.960	0.827	0.875	0.948
8	XGBoost Classifier	0.087	0.087	0.782	0.924
9	Multi-layer Perceptron	0.085	0.085	0.766	0.917
10	Multi-layer Perceptron	0.085	0.085	0.766	0.917

2. TUNE THE MODEL – HYPERPARAMETER TUNING

In [58]:
```
#HYPERPARAMETER TUNING
grid.fit(X_train, y_train)
```

Out[58]:

GridSearchCV

```
GridSearchCV(cv=5,
             estimator=GradientBoostingClassifier(learning_rate=0.7,
                                                  max_depth=4),
             param_grid={'max_features': array([1, 2, 3, 4, 5]),
                         'n_estimators': array([ 10,  20,  30,  40,  50,  60,  70,  80,  90, 100, 110, 120, 130,
       140, 150, 160, 170, 180, 190, 200])})
```

estimator: GradientBoostingClassifier

```
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

GradientBoostingClassifier

```
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

In [59]:
```
print("The best parameters are %s with a score of %0.2f"
      % (grid.best_params_, grid.best_score_))
```

```
The best parameters are {'max_features': 5, 'n_estimators': 200} with a score of 0.97
```

VALIDATION METHODS: KFOLD & Cross Folding

Wilcoxon signed-rank test

In [78]:
```
#KFOLD and Cross Validation Model

from scipy.stats import wilcoxon
from sklearn.datasets import load_iris
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import cross_val_score, KFold

# Load the dataset
X = load_iris().data
y = load_iris().target

# Prepare models and select your CV method
model1 = GradientBoostingClassifier(n_estimators=100)
model2 = XGBClassifier(n_estimators=100)
kf = KFold(n_splits=20, random_state=None)
# Extract results for each model on the same folds
results_model1 = cross_val_score(model1, X, y, cv=kf)
results_model2 = cross_val_score(model2, X, y, cv=kf)
stat, p = wilcoxon(results_model1, results_model2, zero_method='zsplit');
stat
```

Out[78]: 95.0

5x2CV combined F test

In [89]:
```
from mlxtend.evaluate import combined_ftest_5x2cv
from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from mlxtend.data import iris_data

# Prepare data and clfs
X, y = iris_data()
clf1 = GradientBoostingClassifier()
clf2 = DecisionTreeClassifier()

# Calculate p-value
f, p = combined_ftest_5x2cv(estimator1=clf1,
                            estimator2=clf2,
                            X=X, y=y,
                            random_seed=1)

print('f-value:', f)
print('p-value:', p)
```

```
f-value: 1.727272727272733
p-value: 0.2840135734291782
```