Project Development Phase

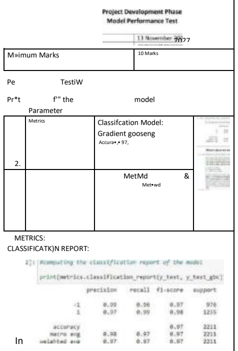
Model Performance Test

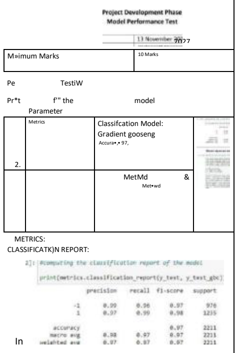
Date	12 November 2022
Team ID	PNT2022TMID23940
Project Name	WEB PHISHING DETECTION
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

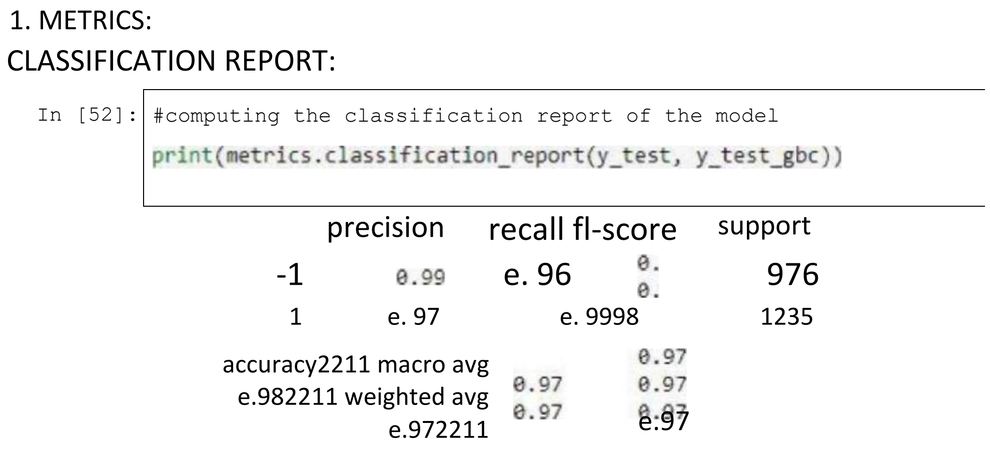
S.No.	Parameter	Values	Screenshot

	Metrics	ClassificationModel :
		Gradian Boosting Classification.
		Accuracy: 97.4%
			
			Project Development Phase
			Model Performance Test
			13 November 2022
			M»imum Marks 10 Marks
			Pe TestiW
			Pr*t f'" the model
			Parameter
			2. Metrics Classifcation Model: Gradient gooseng Accura•,• 97,
			MetMd & Met•wd
			METRICS:
			CLASSIFICATK)N REPORT:
			2]: #computing the classification report of the model
			print(metrics.classification_report(y_test, y_test_gbc))
			precision recall f1-score support
			-1 0.99 0.96 0.97 976
			1 0.97 0.99 0.98 1235
			accuracy 0.97 2211
			macro avg 0.98 0.97 0.97 2211
			In weighted avg 0.97 0.97 0.97 2211

	Tune the Model	Hyperparameter Tuning
		97%
		Validation Method - KFOLD and Cross validation Method.
			
			Project Development Phase
			Model Performance Test
			13 November 2022
			M»imum Marks 10 Marks
			Pe TestiW
			Pr*t f'" the model
			Parameter
			2. Metrics Classifcation Model: Gradient gooseng Accura•,• 97,
			MetMd & Met•wd
			METRICS:
			CLASSIFICATK)N REPORT:
			2]: #computing the classification report of the model
			print(metrics.classification_report(y_test, y_test_gbc))
			precision recall f1-score support
			-1 0.99 0.96 0.97 976
			1 0.97 0.99 0.98 1235
			accuracy 0.97 2211
			macro avg 0.98 0.97 0.97 2211
			In weighted avg 0.97 0.97 0.97 2211

1. METRICS:

Classification Reports:



1. METRICS:

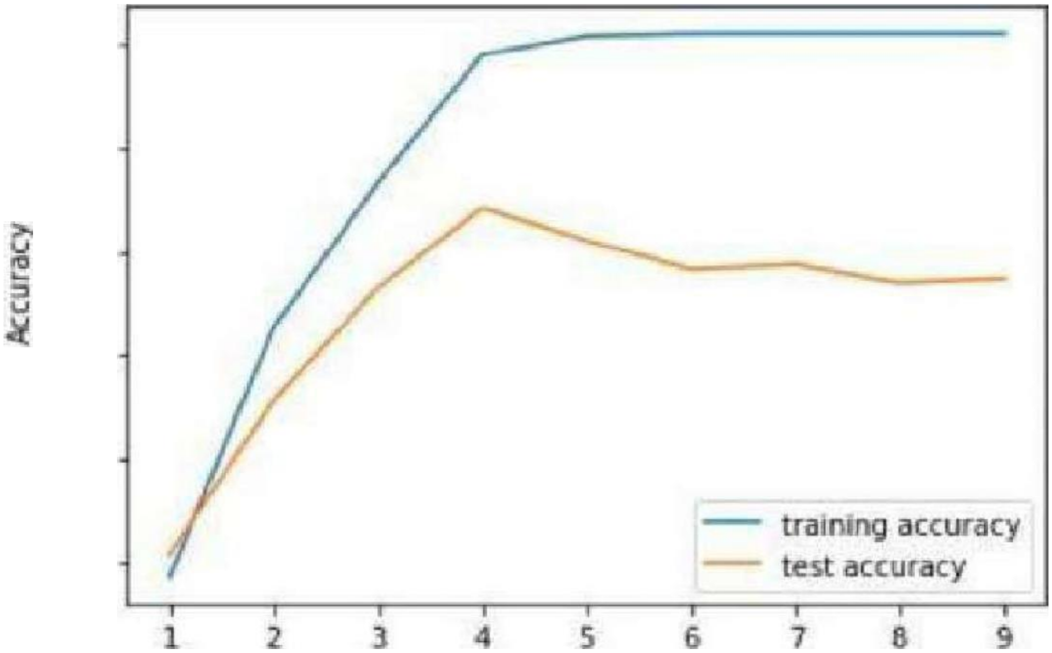
CLASSIFICATION REPORT:

```
In [52]: #computing the classification report of the model
         print(metrics.classification_report(y_test, y_test_gbc))
```

```
              precision    recall fl-score    support
      -1          0.99      e. 96      0.         976
                                       0.
       1          e. 97          e. 9998         1235

accuracy2211 macro avg                 0.97
e.982211 weighted avg      0.97        0.97
           e.972211        0.97        0.97
                                       e:97
```

PERFORMANCE:



099

0.98

0.97

096

0.95

094

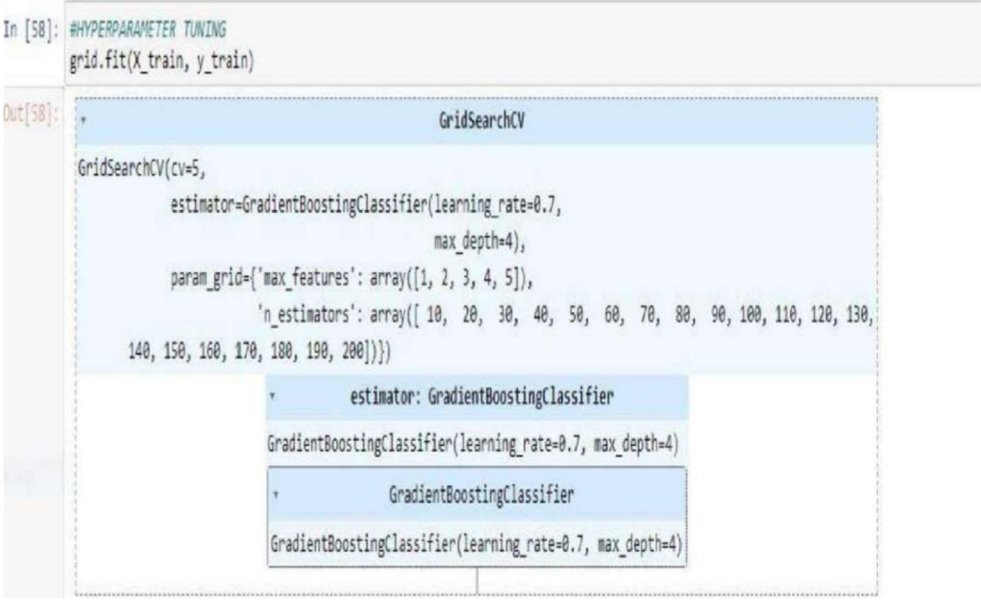
max_depth

Out

	ML Model	Accuracy	fi_score	Recall	Precision
0	Gradient Boosting Classifier	0.974	0.977	0.994	0.986
	CatBoost Classifier	0072	0075	0.994	0.989
2	Random Forest	0069	0.972	0.992	0.991

3	Support Vector Machine	0064	0.968	0.980	0.965
4	Decision Tree	0.958	0.962	0.991	0.9
5	K*Nearest Neighbors	0.956	0.961	0.991	0.989
6	Logistic Regression	0.934	0.941	0.943	0.927
7	Naive Bayes Classifier	0.605	01454	0.292	0.997
8	XGBoost CIB55ifier	0.548		93	0.984
9	Multi-layer Perceptron	0.543	0.543	0.989	0.983

2.TUNE THE MODEL HYPER PARAMETER TURNING

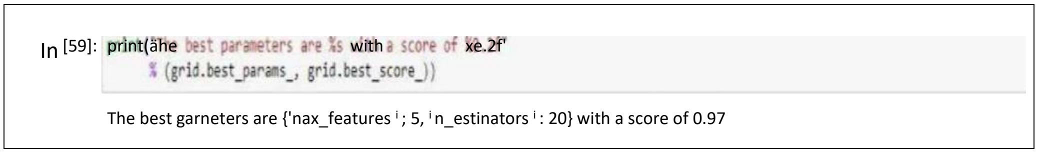


```
In [58]: #HYPERPARAMETER TUNING
         grid.fit(X_train, y_train)

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(ăhe best parameters are %s with a score of xe.2f'
               % (grid.best_params_, grid.best_score_))

The best garneters are {'nax_features ⁱ ; 5, ⁱ n_estinators ⁱ : 20} with a score of 0.97
```

3.VALlDATlON METHOD KFOLD AND CROSS VALIDATION

Wilcoxon signed-rank test

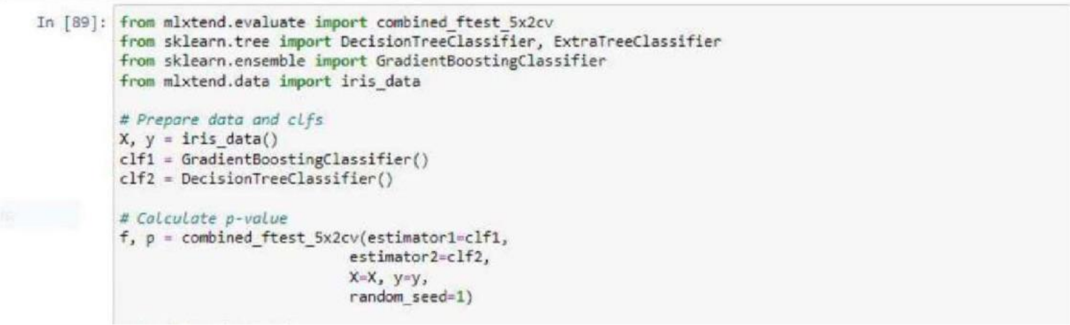
```
In [78]: #KFOLO and Cross Val i dotion talodeL
            from scipy.stats import Wilcoxon from sklearn.datasets
            import load_irås from skiearn.ensemble import
            GradientBoostingC1assifier from xgb005t import
            XGöC1assifier from sklearn.model_selection import
            cross_val_score, KFoId

           # Load the dataset
            X = load iris() edata
            y = load_iris()*target

            # Prepare models and select your CV
            method modell mode12 XGBC1assifier(n
            estimators=1ØØ) kf — KFoId(n splits—20,
            random state—None)
            * Extract results for each model on the same folds results modeli =
            cross y, cv=kf) results_mode12 X, y, cv=kf) stat, p wilcoxon(results
            modell, results mode12, zero method- 'zsplit• ) ; stat

out[78J : 9S.ø
```

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:
ø.2B4ø13S734291782
```