### Project Development Phase Model Performance Test

Date	9 November 2023
Team ID	Team-592613
Project Name	Online Payments Fraud Detection Using ML
Maximum Marks	10 Marks
Team Members	Srajal Agarwal (Leader)
	Deepesh
	Dhruv
	Paras Garg

#### **Model Performance Testing:**

Since the outcome of our project needed to be predicted as either "is a fraud" or "is not a fraud," a classification-based model was necessary.

Decision Tree Classifier, Logistic Regression, Extra Tree Classifier and Random Forest Classifier were the models utilized in the projects.

The metrics reports for each model are as follows:

## **Decision Tree Classifier:-**

1. Test accuracy

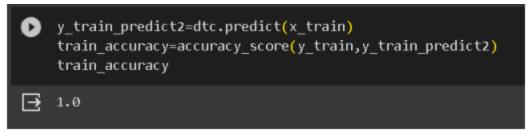
```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)

DecisionTreeClassifier
DecisionTreeClassifier()

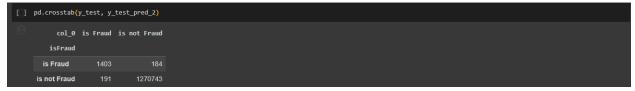
[] y_test_pred_2 = dtc.predict(X_test)

[] accuracy_test_2 = accuracy_score(y_test, y_test_pred_2)
accuracy_test_2
0.9997853093819277
```

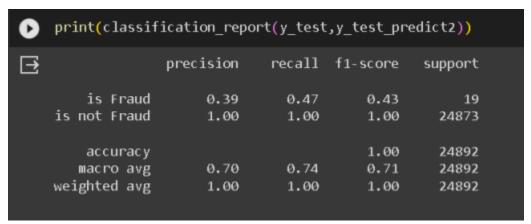
2. Train accuracy



Confusion Matrix



4. Classification Report



## **Random Forest Classifier:-**

Test accuracy



Testing Accuracy 0.9837896584810781

2. Train accuracy



3. Confusion Matrix



4. Classification Report

```
[ ] print(classification_report(y_test,y_pred))

precision recall f1-score support

is Fraud 0.07 0.91 0.12 1587
is not Fraud 1.00 0.98 0.99 1270934

accuracy 0.98 1272521
macro avg 0.53 0.95 0.56 1272521
weighted avg 1.00 0.98 0.99 1272521
```

### **Logistic Regression:-**

1. Test accuracy

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,roc_auc_score,roc_curve
lr=logisticRegression()
lr.fit(x_train_smote,y_train_smote)
y_pred=lr.predict(X_test)
print("Training Score",accuracy_score(y_train_smote,lr.predict(x_train_smote)))
print("Testing Accuracy",accuracy_score(y_test,y_pred))
Training Score 0.8966881500108292
```

2. Train accuracy

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,roc_auc_score,roc_curve
lr=LogisticRegression()
lr.fit(x_train_smote,y_train_smote)
y_red=lr.predict(X_test)
print("Training Score",accuracy_score(y_train_smote,lr.predict(x_train_smote)))
print("Testing Accuracy",accuracy_score(y_test,y_pred))

Testing Accuracy 0.9662284551689128
```

3. Confusion Matrix



4. Classification Report

```
print(classification_report(y_test,y_pred))

precision recall f1-score support

is Fraud 0.03 0.83 0.06 1587
is not Fraud 1.00 0.97 0.98 1270934

accuracy 0.97 1272521
macro avg 0.51 0.90 0.52 1272521
weighted avg 1.00 0.97 0.98 1272521
```

#### **Extra Tree Classifier:-**

1. Test accuracy

```
from sklearn.ensemble import ExtraTreesClassifier
etc=ExtraTreesClassifier()
etc.fit(x_train_smote,y_train_smote)
y_pred=etc.predict(X_test)
print("Training Score", accuracy_score(y_train_smote,etc.predict(x_train_smote)))
print("Testing Accuracy", accuracy_score(y_test,y_pred))

Testing Accuracy 0.9994451957963758
```

2. Train accuracy

```
from sklearn.ensemble import ExtraTreesClassifier
etc=ExtraTreesClassifier()
etc.fit(x_train_smote,y_train_smote)
y_pred=etc.predict(X_test)
print("Training Score",accuracy_score(y_train_smote,etc.predict(x_train_smote)))
print("Testing Accuracy",accuracy_score(y_test,y_pred))
Training Score 1.0
```

3. Confusion Matrix



4. Classification Report



# **Final Prediction:-**

```
• etc.predict([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])

etc.predict([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])
```