Project Development Phase
Model Performance Test

Date	**09-11-2023**
Team ID	**Team-593025**
Project Name	**Project - Online Payments Fraud Detection Using ML**
Maximum Marks	**10 Marks**

Model Performance Testing:-

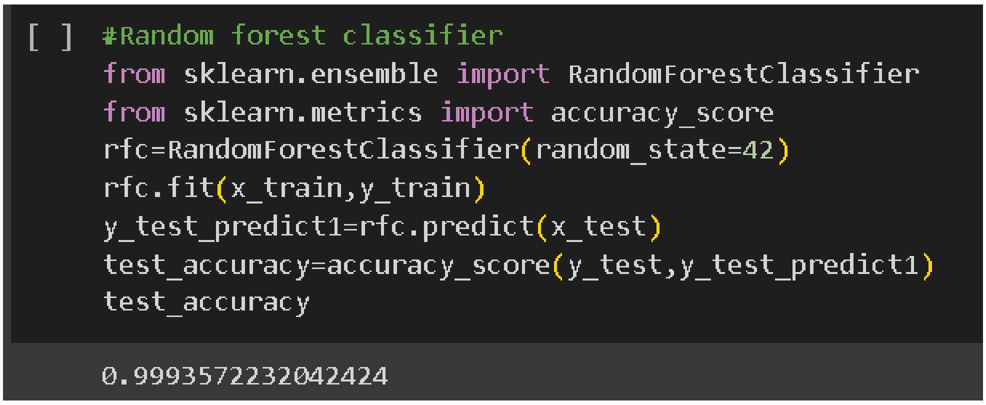
Our Project required a classification based model as the result to be predicted was either “is a fraud” or “is not a fraud”

Models used in the projects were, Random forest Classifier, Decision Tree Classifier, Extra Tree Classifier, Support Vector Machine, xgBoost Classifier

Following are the Metrics report for each of the model:-

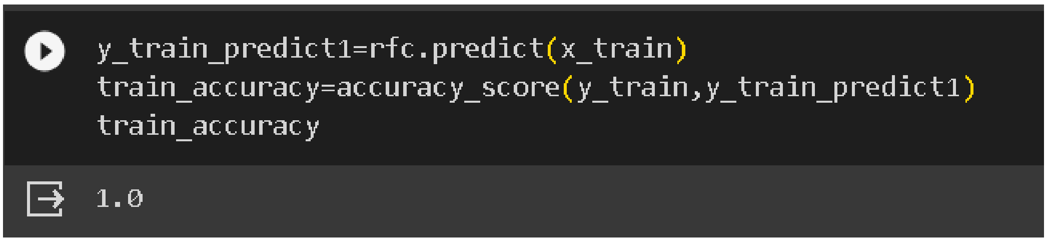
Random Forest Classifier:-

1. Test accuracy



```
[ ]  #Random forest classifier
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score
     rfc=RandomForestClassifier(random_state=42)
     rfc.fit(x_train,y_train)
     y_test_predict1=rfc.predict(x_test)
     test_accuracy=accuracy_score(y_test,y_test_predict1)
     test_accuracy

     0.9993572232042424
```

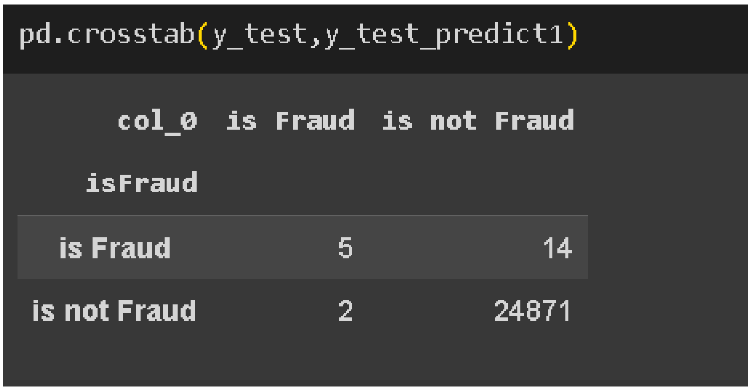
2. Train accuracy



```
y_train_predict1=rfc.predict(x_train)
train_accuracy=accuracy_score(y_train,y_train_predict1)
train_accuracy
```

```
1.0
```

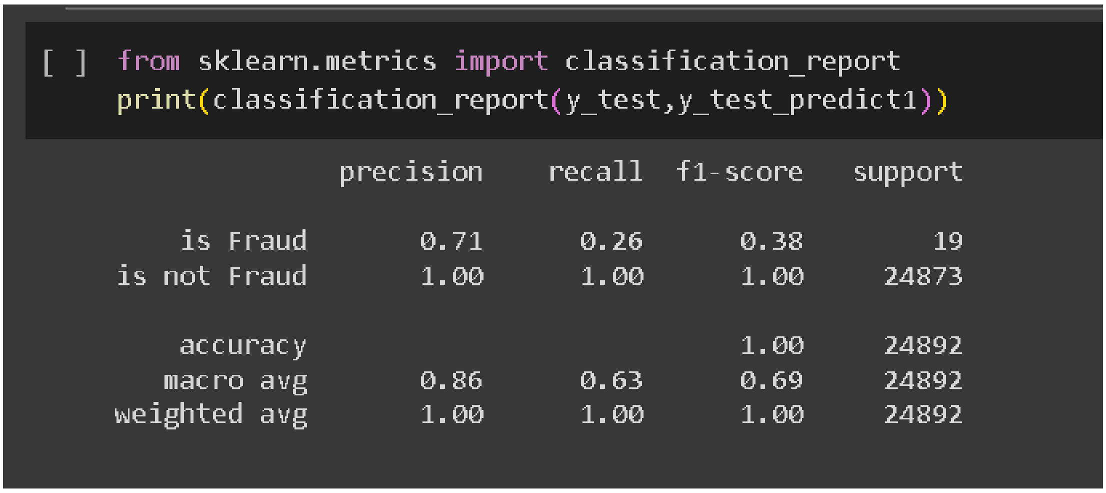
3. Confusion Matrix



```
pd.crosstab(y_test,y_test_predict1)
```

col_0 isFraud	is Fraud	is not Fraud
is Fraud	5	14
is not Fraud	2	24871

4. Classification Report

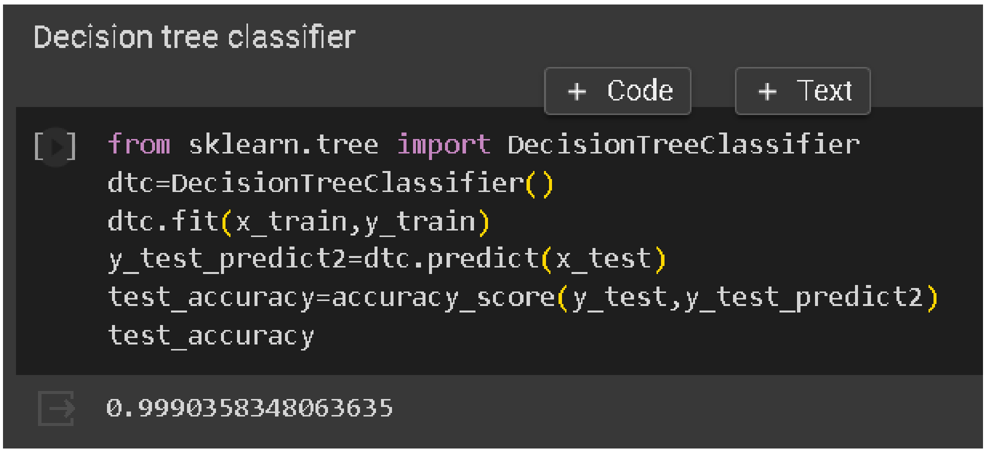


```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_test_predict1))
```

```
              precision    recall  f1-score   support

    is Fraud       0.71      0.26      0.38        19
is not Fraud       1.00      1.00      1.00     24873

    accuracy                           1.00     24892
   macro avg       0.86      0.63      0.69     24892
weighted avg       1.00      1.00      1.00     24892
```

Decision Tree Classifier:-

1. Test accuracy

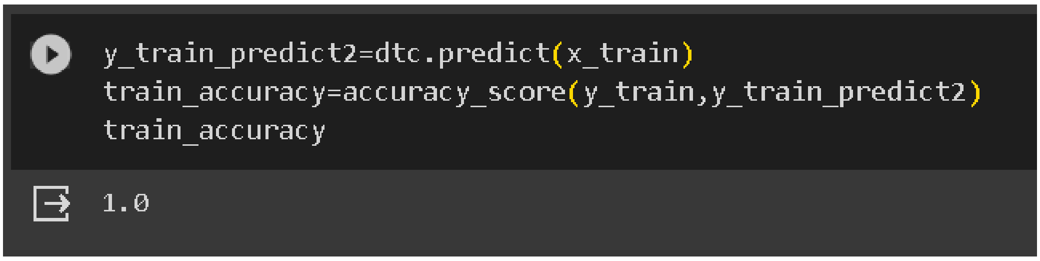


Decision tree classifier

```
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier()
dtc.fit(x_train,y_train)
y_test_predict2=dtc.predict(x_test)
test_accuracy=accuracy_score(y_test,y_test_predict2)
test_accuracy
```

```
0.9990358348063635
```

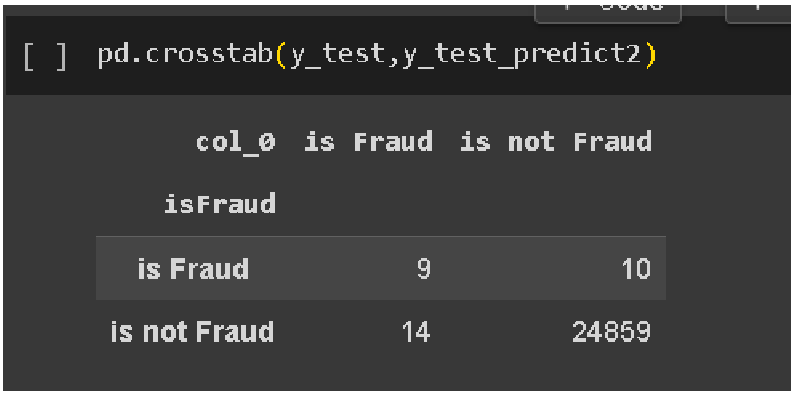
2. Train accuracy



```
y_train_predict2=dtc.predict(x_train)
train_accuracy=accuracy_score(y_train,y_train_predict2)
train_accuracy
```

```
1.0
```

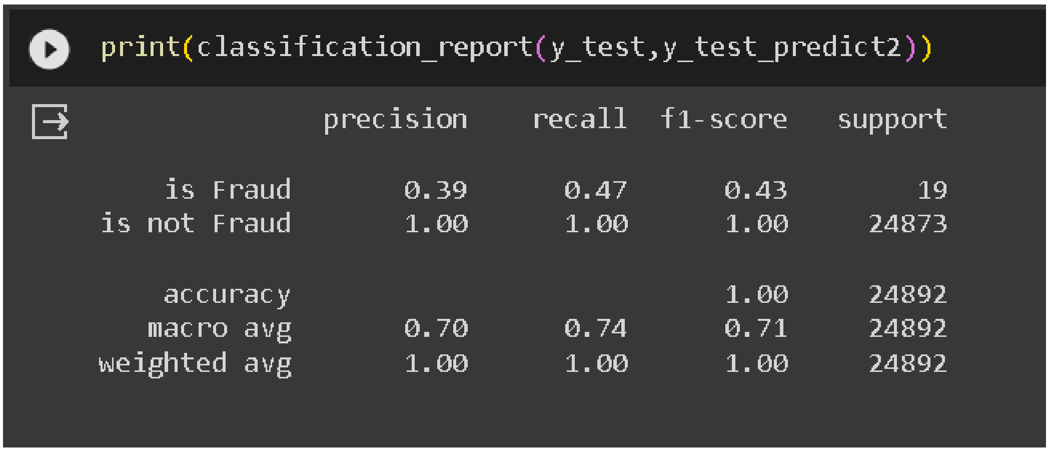
3. Confusion Matrix



```
[ ] pd.crosstab(y_test,y_test_predict2)
```

col_0	is Fraud	is not Fraud
isFraud		
is Fraud	9	10
is not Fraud	14	24859

4. Classification Report

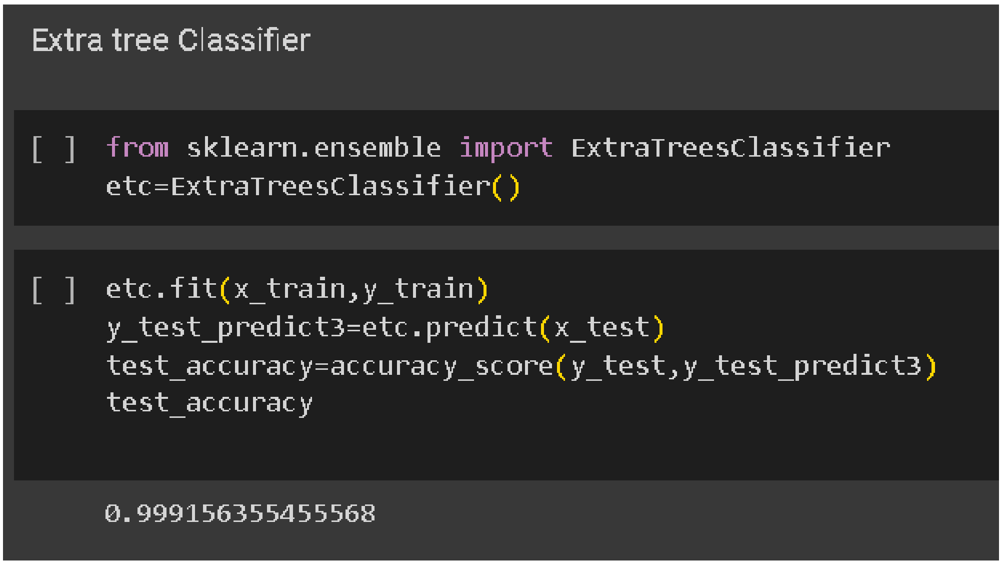


```
print(classification_report(y_test,y_test_predict2))
```

```
              precision    recall  f1-score   support

    is Fraud       0.39      0.47      0.43        19
is not Fraud       1.00      1.00      1.00     24873

    accuracy                           1.00     24892
   macro avg       0.70      0.74      0.71     24892
weighted avg       1.00      1.00      1.00     24892
```

Extra Tree Classifier:-

1. Test accuracy



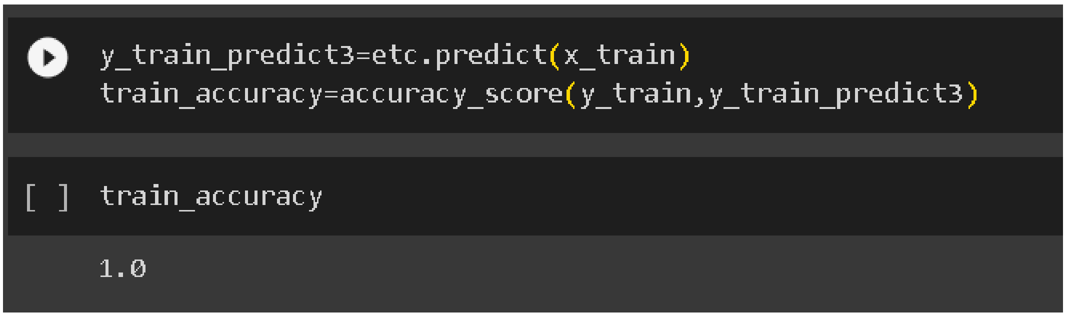
Extra tree Classifier

```
[ ] from sklearn.ensemble import ExtraTreesClassifier
    etc=ExtraTreesClassifier()
```

```
[ ] etc.fit(x_train,y_train)
    y_test_predict3=etc.predict(x_test)
    test_accuracy=accuracy_score(y_test,y_test_predict3)
    test_accuracy
```

```
    0.9991563554555568
```

2. Train accuracy

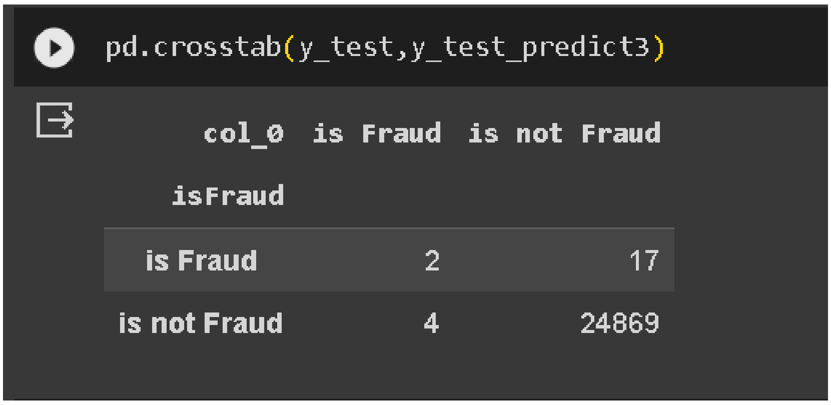


```
y_train_predict3=etc.predict(x_train)
train_accuracy=accuracy_score(y_train,y_train_predict3)
```

```
[ ] train_accuracy
```

```
    1.0
```

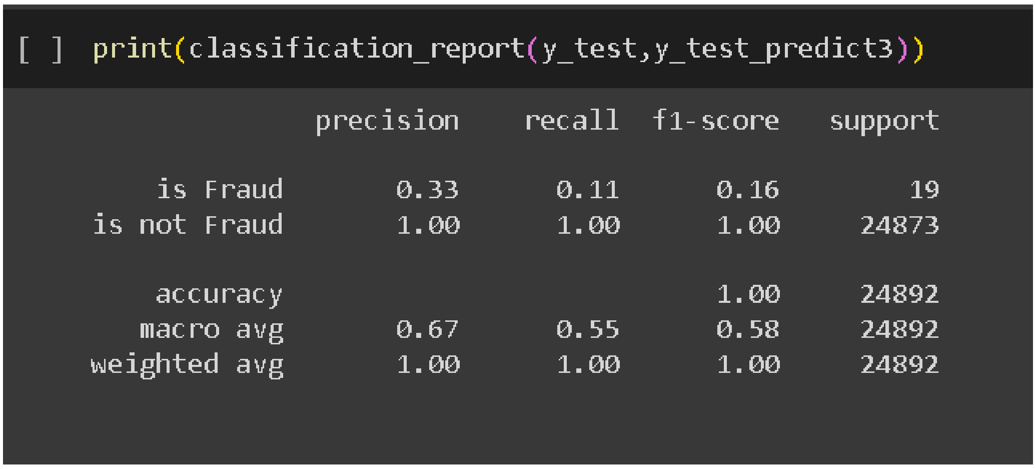
3. Confusion Matrix



```
pd.crosstab(y_test,y_test_predict3)
```

col_0	is Fraud	is not Fraud
isFraud		
is Fraud	2	17
is not Fraud	4	24869

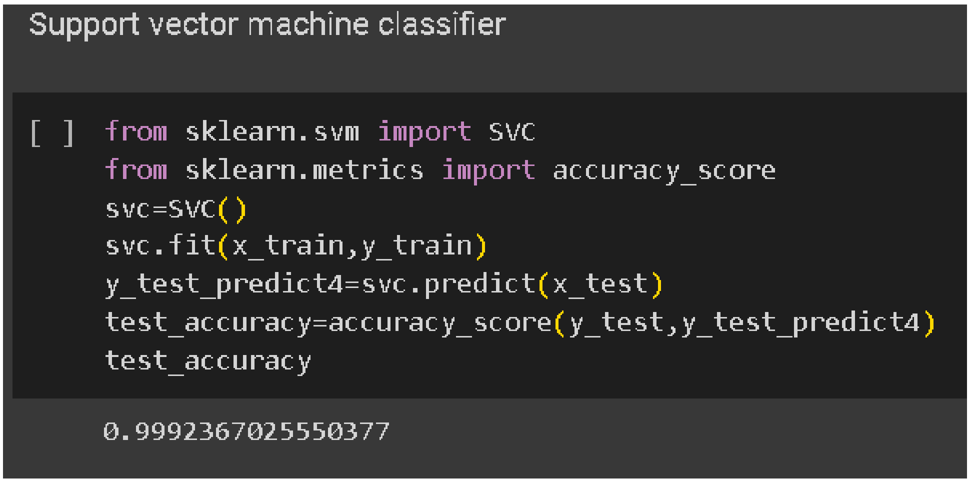
4. Classification Report



```
[ ]  print(classification_report(y_test,y_test_predict3))

                  precision    recall  f1-score   support

        is Fraud       0.33      0.11      0.16        19
    is not Fraud       1.00      1.00      1.00     24873

        accuracy                           1.00     24892
       macro avg       0.67      0.55      0.58     24892
    weighted avg       1.00      1.00      1.00     24892
```

Support Vector Machine:-

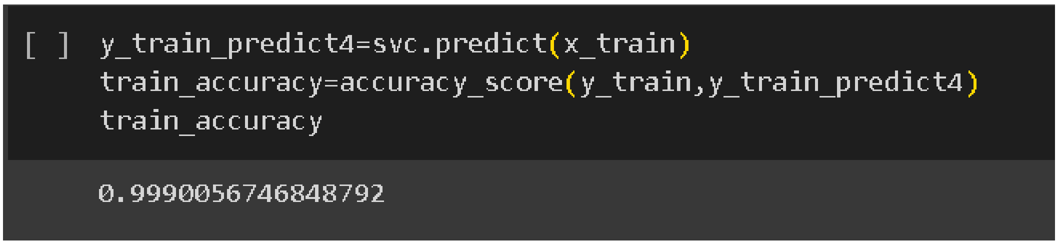
1. Test accuracy



Support vector machine classifier

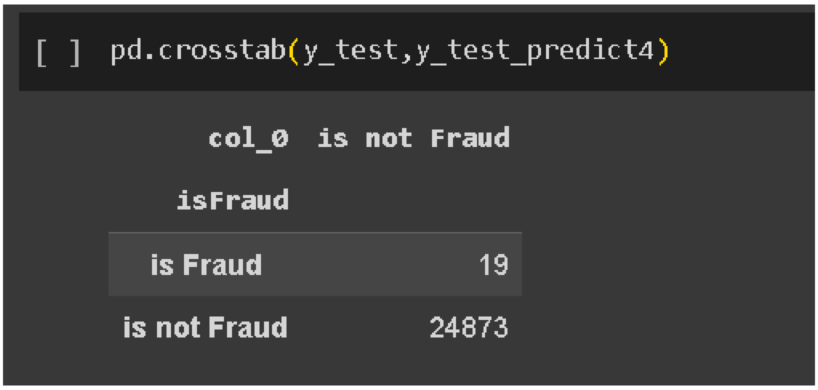
```
[ ]  from sklearn.svm import SVC
     from sklearn.metrics import accuracy_score
     svc=SVC()
     svc.fit(x_train,y_train)
     y_test_predict4=svc.predict(x_test)
     test_accuracy=accuracy_score(y_test,y_test_predict4)
     test_accuracy

     0.9992367025550377
```

2. Train accuracy



```
[ ]  y_train_predict4=svc.predict(x_train)
     train_accuracy=accuracy_score(y_train,y_train_predict4)
     train_accuracy

     0.9990056746848792
```

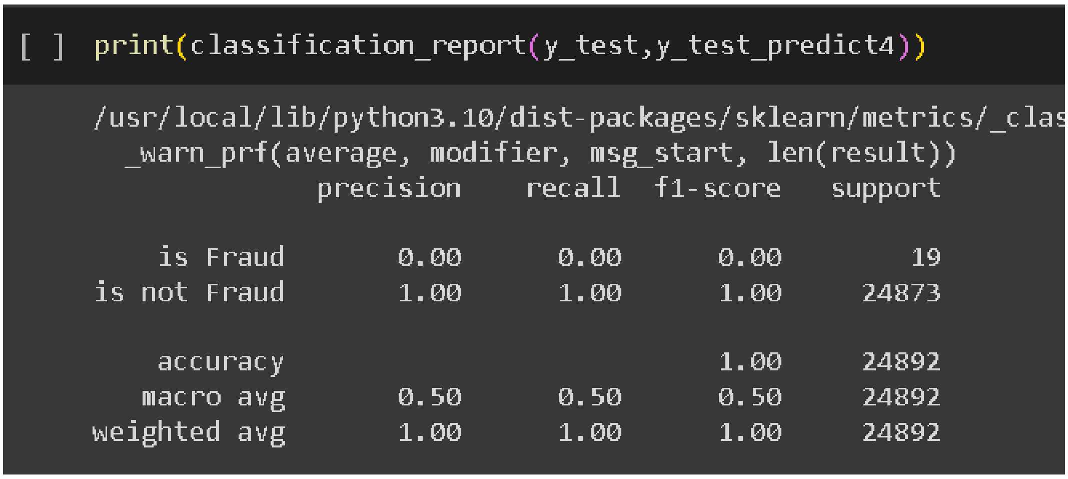
3. Confusion Matrix



```
[ ]  pd.crosstab(y_test,y_test_predict4)
```

```
              col_0   is not Fraud
            isFraud
           is Fraud             19
       is not Fraud          24873
```

4. Classification Report

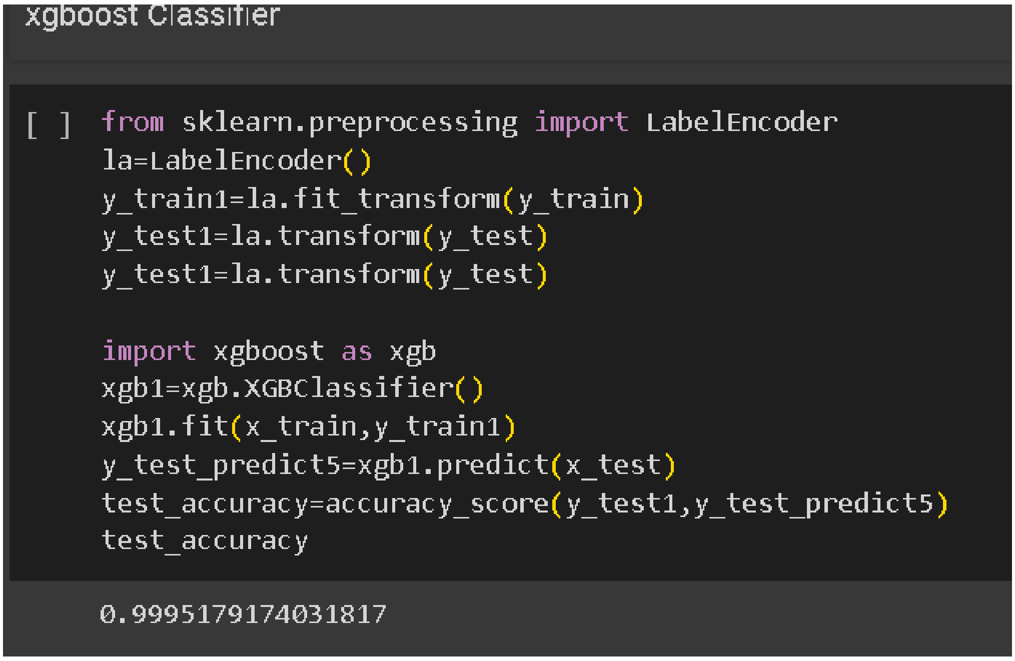


```
[ ]  print(classification_report(y_test,y_test_predict4))
```

```
     /usr/local/lib/python3.10/dist-packages/sklearn/metrics/_clas
       _warn_prf(average, modifier, msg_start, len(result))
                   precision    recall  f1-score   support

         is Fraud       0.00      0.00      0.00        19
     is not Fraud       1.00      1.00      1.00     24873

         accuracy                           1.00     24892
        macro avg       0.50      0.50      0.50     24892
     weighted avg       1.00      1.00      1.00     24892
```

xgBooost Classifier:-

1. Test accuracy

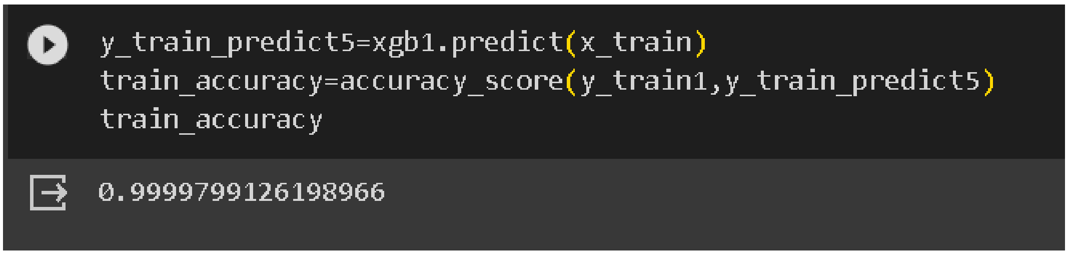


xgboost Classifier

```
[ ]  from sklearn.preprocessing import LabelEncoder
     la=LabelEncoder()
     y_train1=la.fit_transform(y_train)
     y_test1=la.transform(y_test)
     y_test1=la.transform(y_test)

     import xgboost as xgb
     xgb1=xgb.XGBClassifier()
     xgb1.fit(x_train,y_train1)
     y_test_predict5=xgb1.predict(x_test)
     test_accuracy=accuracy_score(y_test1,y_test_predict5)
     test_accuracy
```

```
     0.9995179174031817
```

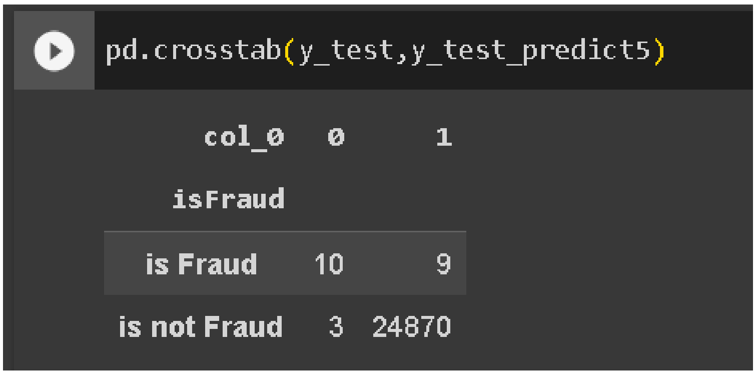
2. Train accuracy



```
y_train_predict5=xgb1.predict(x_train)
train_accuracy=accuracy_score(y_train1,y_train_predict5)
train_accuracy
```

```
0.9997799126198966
```

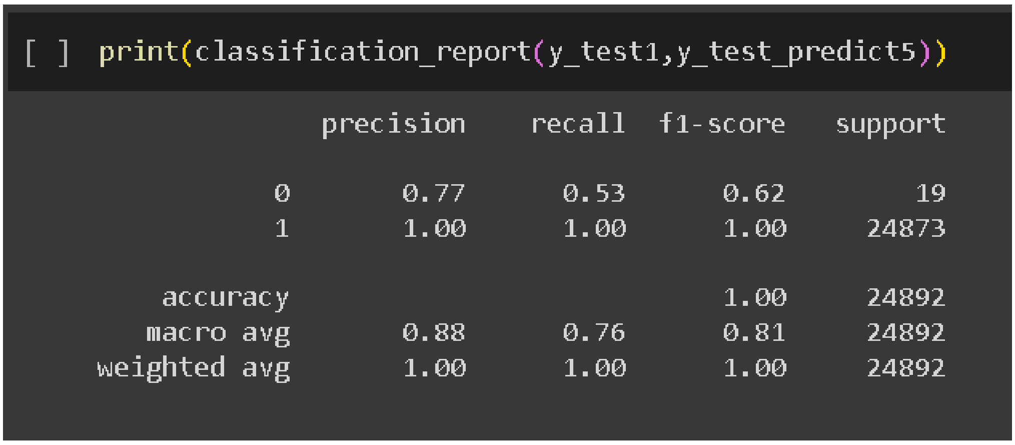
3. Confusion Matrix



```
pd.crosstab(y_test,y_test_predict5)
```

col_0	0	1
isFraud		
is Fraud	10	9
is not Fraud	3	24870

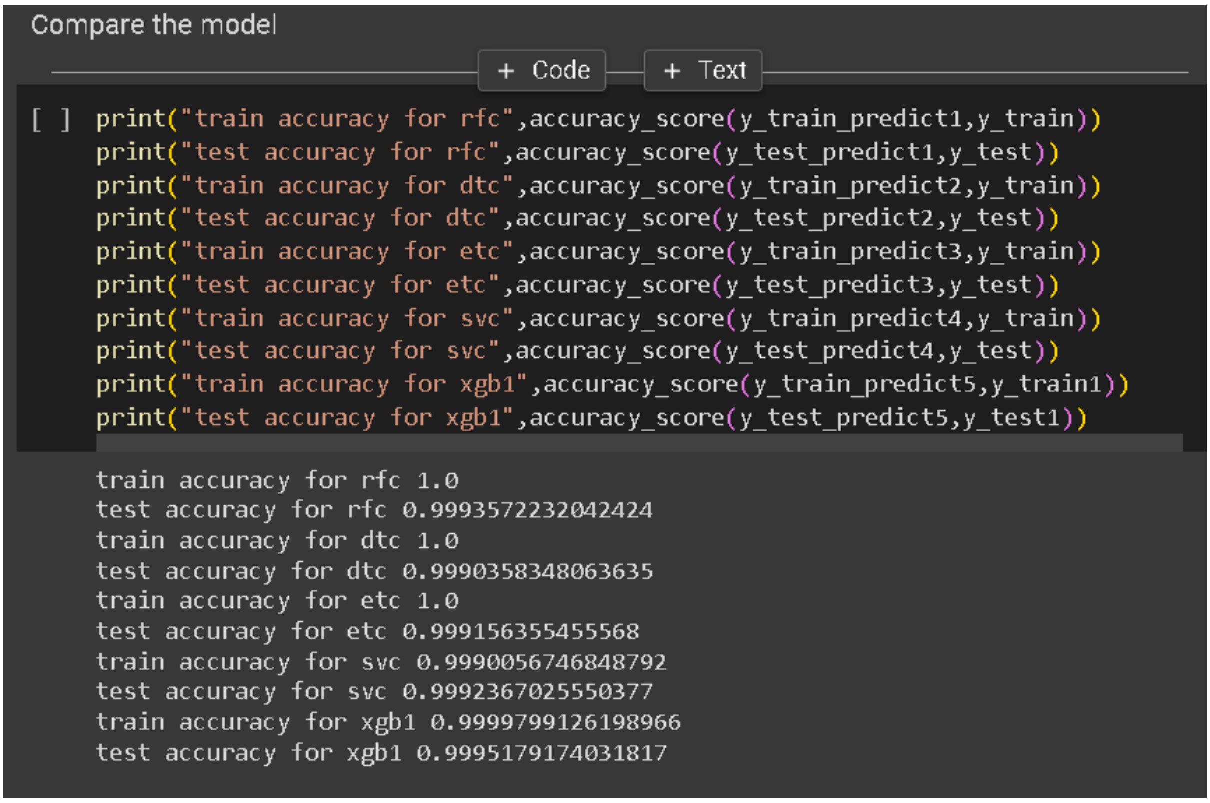
4. Classification Report



```
[ ] print(classification_report(y_test1,y_test_predict5))
```

```
              precision    recall  f1-score   support

           0       0.77      0.53      0.62        19
           1       1.00      1.00      1.00     24873

    accuracy                           1.00     24892
   macro avg       0.88      0.76      0.81     24892
weighted avg       1.00      1.00      1.00     24892
```

Final comparisons:-



Compare the model

```
print("train accuracy for rfc",accuracy_score(y_train_predict1,y_train))
print("test accuracy for rfc",accuracy_score(y_test_predict1,y_test))
print("train accuracy for dtc",accuracy_score(y_train_predict2,y_train))
print("test accuracy for dtc",accuracy_score(y_test_predict2,y_test))
print("train accuracy for etc",accuracy_score(y_train_predict3,y_train))
print("test accuracy for etc",accuracy_score(y_test_predict3,y_test))
print("train accuracy for svc",accuracy_score(y_train_predict4,y_train))
print("test accuracy for svc",accuracy_score(y_test_predict4,y_test))
print("train accuracy for xgb1",accuracy_score(y_train_predict5,y_train1))
print("test accuracy for xgb1",accuracy_score(y_test_predict5,y_test1))
```

```
train accuracy for rfc 1.0
test accuracy for rfc 0.9993572232042424
train accuracy for dtc 1.0
test accuracy for dtc 0.9990358348063635
train accuracy for etc 1.0
test accuracy for etc 0.999156355455568
train accuracy for svc 0.9990056746848792
test accuracy for svc 0.9992367025550377
train accuracy for xgb1 0.9999799126198966
test accuracy for xgb1 0.9995179174031817
```