SVM

```
from sklearn.svm import SVC
```

```
svc=SVC()
```

```
svc.fit(X_train, Y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
X_train_prediction1 = svc.predict(X_train)
training_data_accuracy1= accuracy_score(X_train_prediction1, Y_train)
```

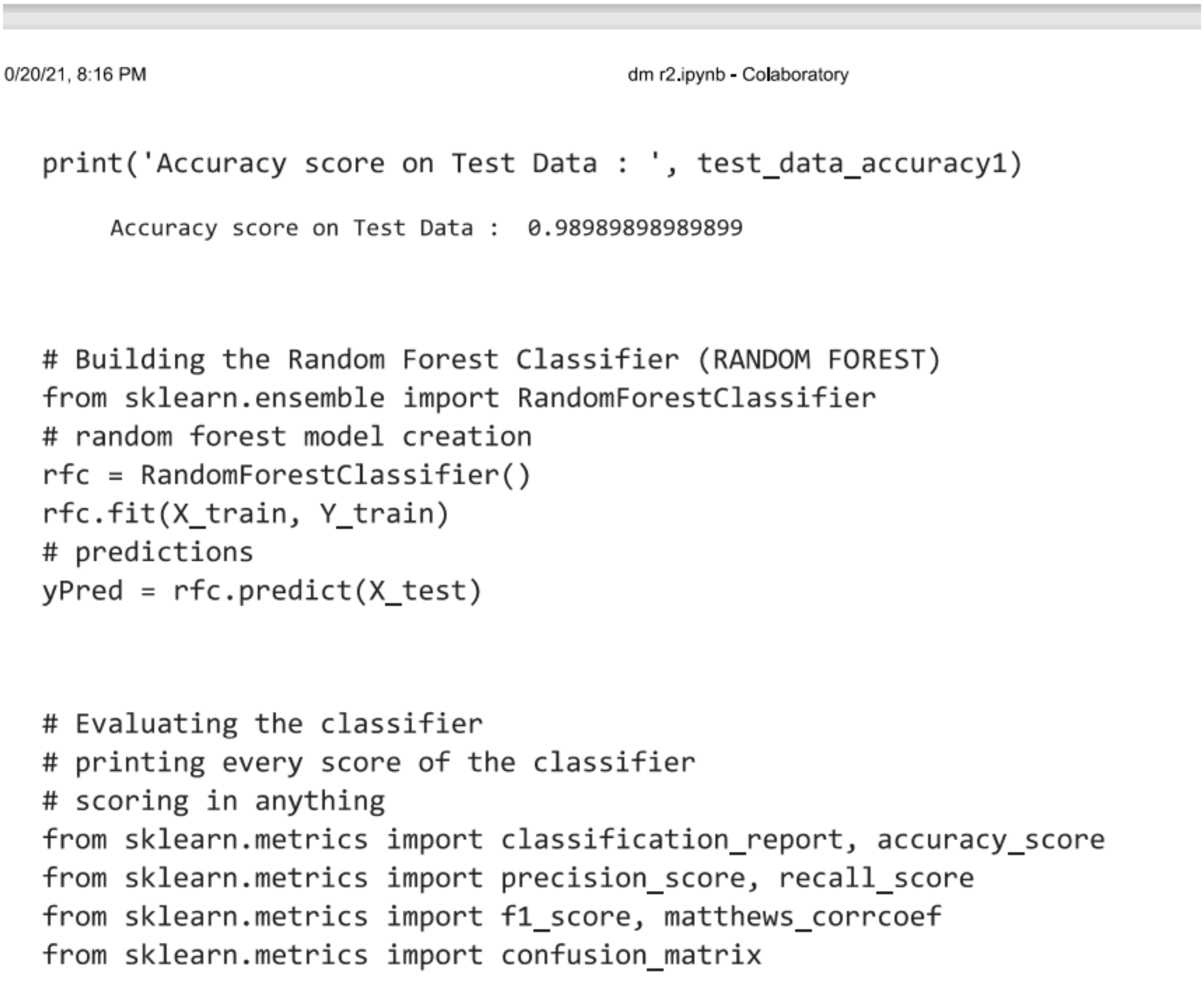
```
print('Accuracy on Training data : ', training_data_accuracy1)
```

```
Accuracy on Training data :  0.9949494949494949
```

```
# accuracy on test data
X_test_prediction1 = svc.predict(X_test)
test_data_accuracy1 = accuracy_score(X_test_prediction1, Y_test)
```

Random Forest Algorithm

ttps://colab.research.google.com/drive/13vsZrabAPsHlZRsImt7YsvdSuumY0Pkq?usp=sharing#printMode=true



0/20/21, 8:16 PM dm r2.ipynb - Colaboratory

```
print('Accuracy score on Test Data : ', test_data_accuracy1)

    Accuracy score on Test Data :  0.98989898989899
```

```
# Building the Random Forest Classifier (RANDOM FOREST)
from sklearn.ensemble import RandomForestClassifier
# random forest model creation
rfc = RandomForestClassifier()
rfc.fit(X_train, Y_train)
# predictions
yPred = rfc.predict(X_test)
```

```
# Evaluating the classifier
# printing every score of the classifier
# scoring in anything
from sklearn.metrics import classification_report, accuracy_score
from sklearn.metrics import precision_score, recall_score
from sklearn.metrics import f1_score, matthews_corrcoef
from sklearn.metrics import confusion_matrix
```

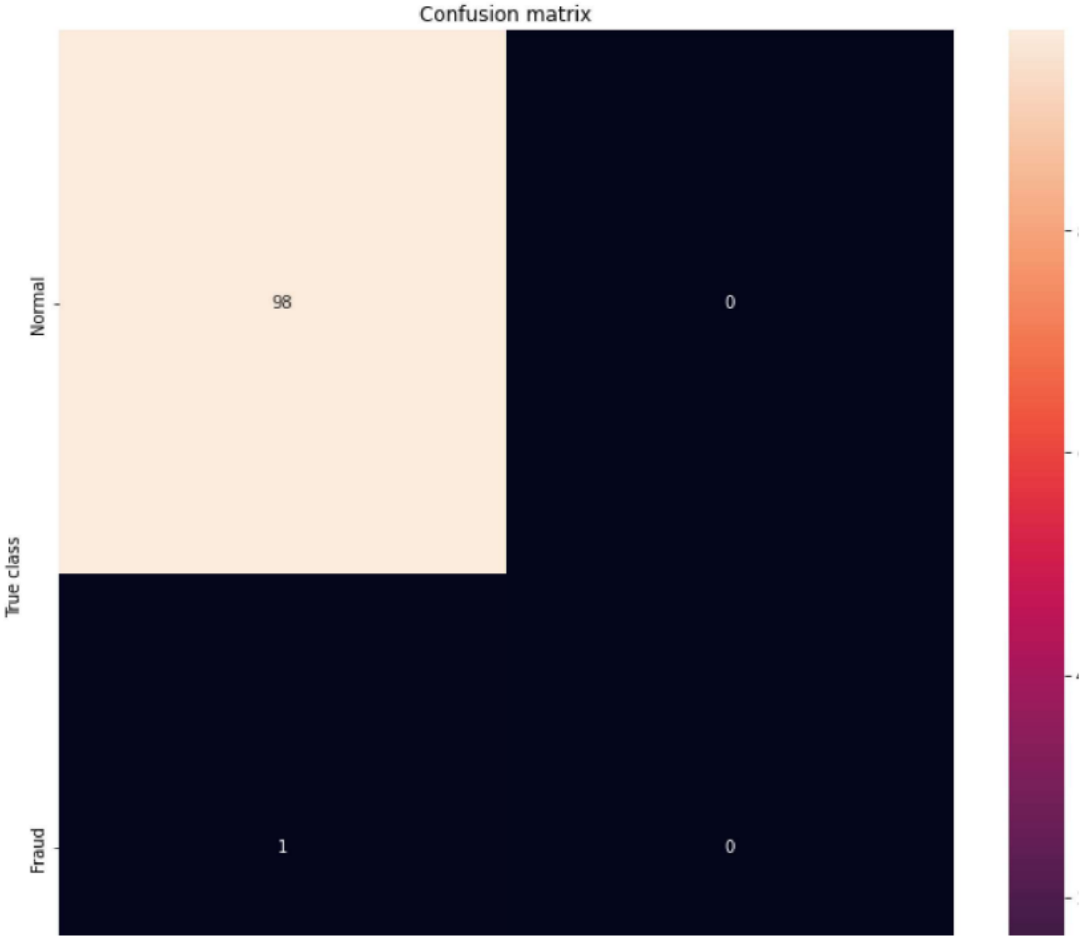
```
n_outliers = len(fraud)
n_errors = (yPred != Y_test).sum()
print("The model used is Random Forest classifier")

acc = accuracy_score(Y_test, yPred)
print("The accuracy is {}".format(acc))
```

```
The model used is Random Forest classifier
The accuracy is 0.98989898989899
```

```
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import gridspec
```

```
# printing the confusion matrix
LABELS = ['Normal', 'Fraud']
conf_matrix = confusion_matrix(Y_test, yPred)
plt.figure(figsize =(12, 12))
sns.heatmap(conf_matrix, xticklabels = LABELS,
            yticklabels = LABELS, annot = True, fmt ="d");
```

```
plt.title("Confusion matrix")
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show()
```


Confusion matrix
Normal
True class
Fraud