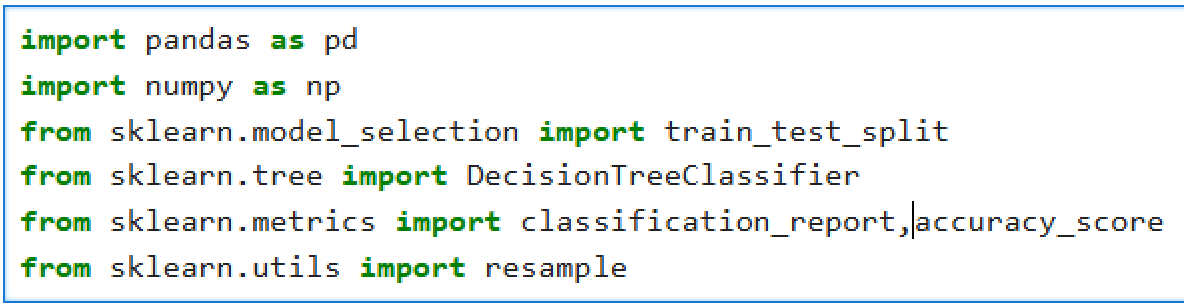
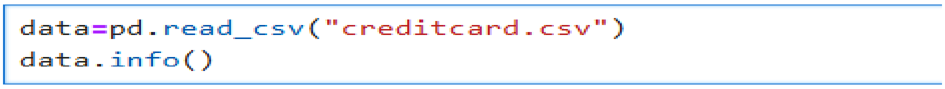
DMML LAB 2 – CREDIT CARD FRAUD DETECTION USING DECISION TREE CLASSIFIER

1. Import required libraries.

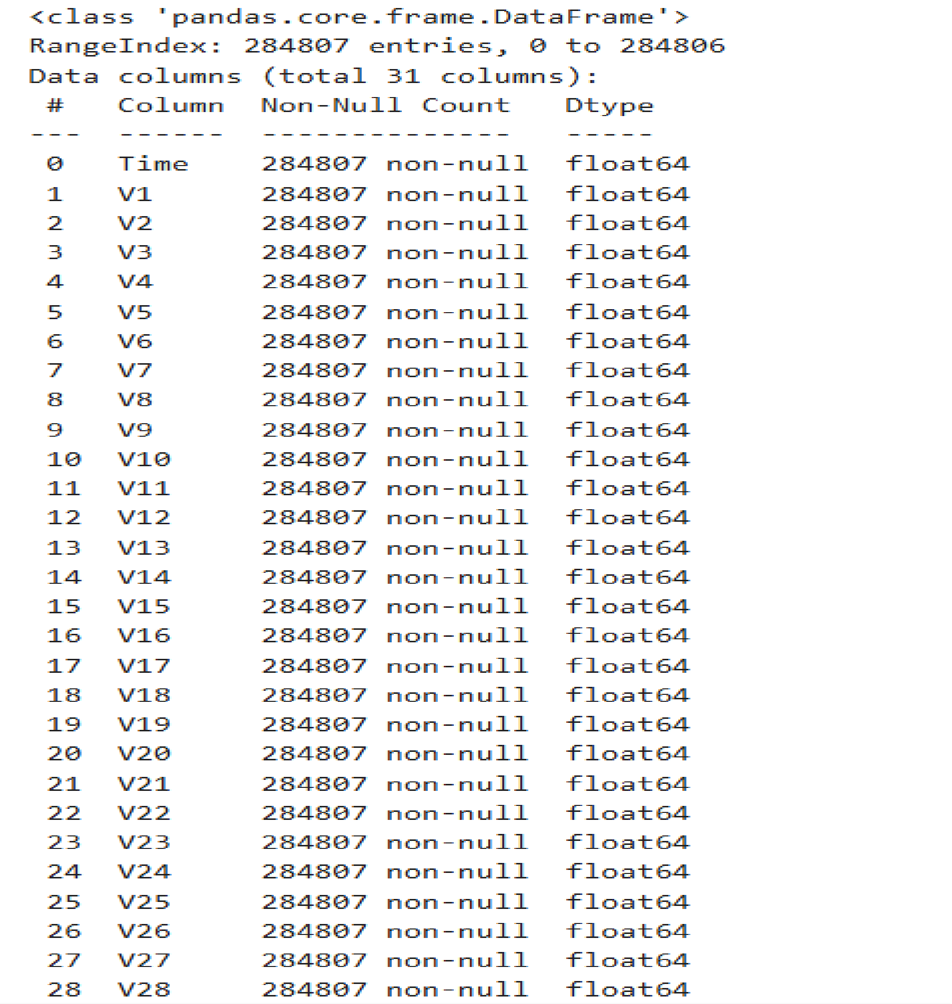


```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report,accuracy_score
from sklearn.utils import resample
```

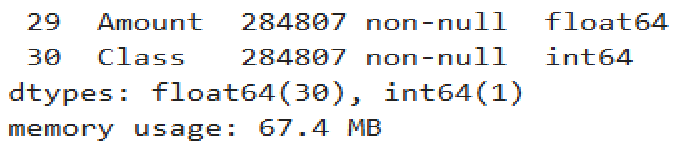
2. Load data to a data frame and analyze the contents.



```
data=pd.read_csv("creditcard.csv")
data.info()
```

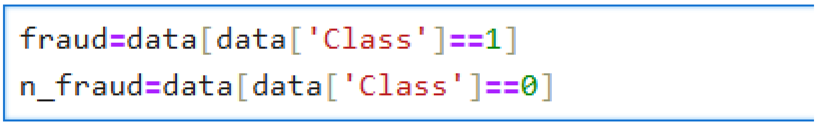


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
```



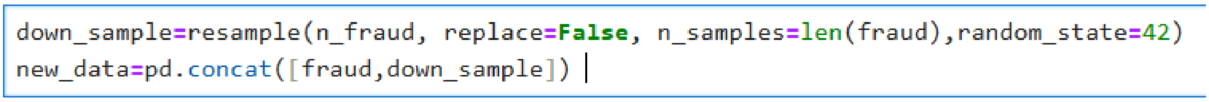
```
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

3. Divide the data into two subsets – fraudulent transactions and non fraudulent transactions.



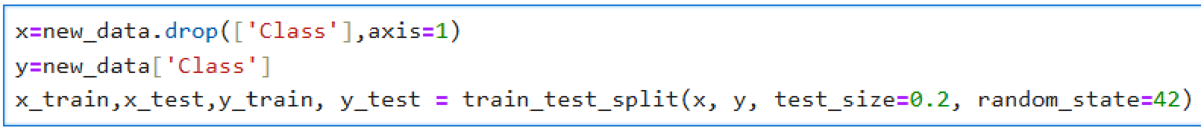
```
fraud=data[data['Class']==1]
n_fraud=data[data['Class']==0]
```

4. To make the data set balanced, perform downsampling on non fraudulent transactions. And create a balanced data.



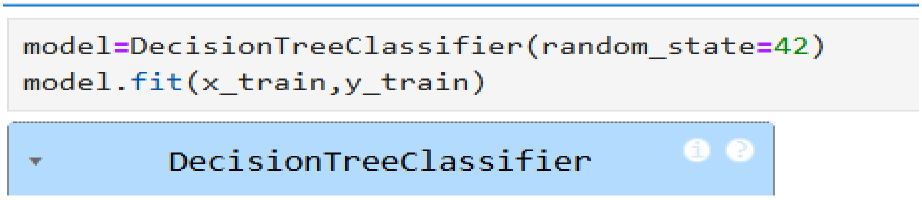
```
down_sample=resample(n_fraud, replace=False, n_samples=len(fraud),random_state=42)
new_data=pd.concat([fraud,down_sample])
```

5. Split the data in to features (x) and targets (y) and assign them to training data and test data.



```
x=new_data.drop(['Class'],axis=1)
y=new_data['Class']
x_train,x_test,y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

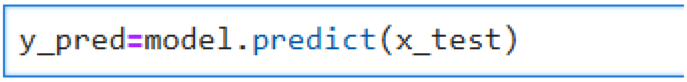
6. Train the model using decision tree classifier.



```
model=DecisionTreeClassifier(random_state=42)
model.fit(x_train,y_train)
```

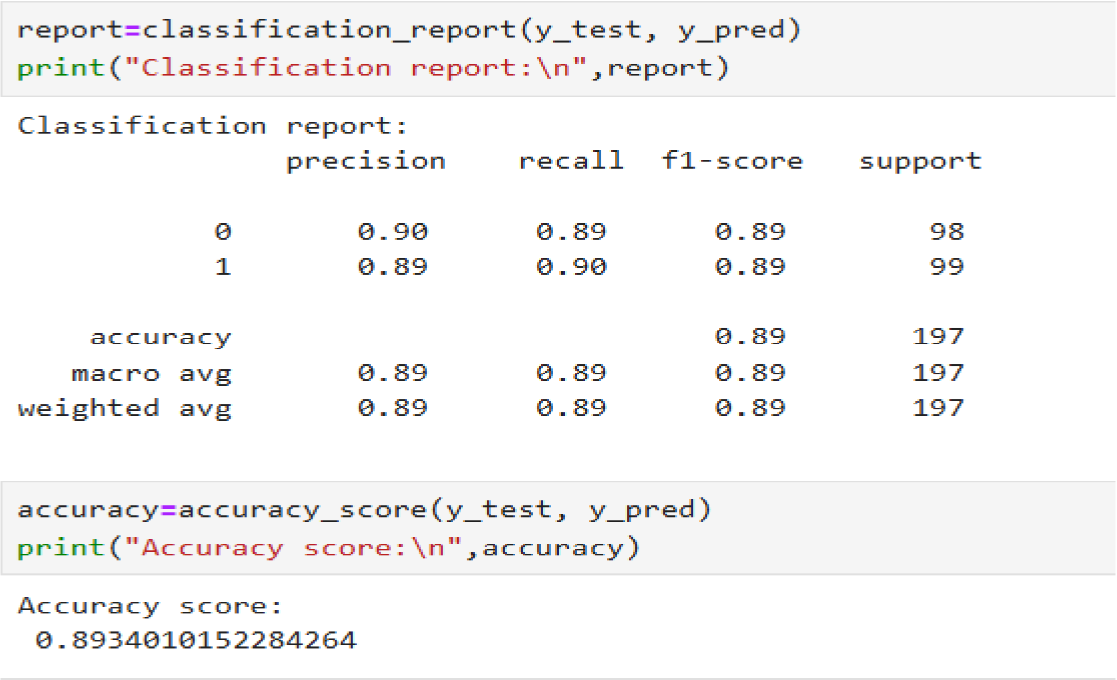
```
DecisionTreeClassifier
```

7. Predict the value for testing test.



```
y_pred=model.predict(x_test)
```

8. Evaluate the model.



```
report=classification_report(y_test, y_pred)
print("Classification report:\n",report)
```

```
Classification report:
               precision    recall  f1-score   support

           0       0.90      0.89      0.89        98
           1       0.89      0.90      0.89        99

    accuracy                           0.89       197
   macro avg       0.89      0.89      0.89       197
weighted avg       0.89      0.89      0.89       197
```

```
accuracy=accuracy_score(y_test, y_pred)
print("Accuracy score:\n",accuracy)
```

```
Accuracy score:
 0.8934010152284264
```