ex10-online-fraud-detection

October 16, 2024

[1]:
```
import pandas as pd

# Load the creditcard.csv using pandas
datainput = pd.read_csv("datasets/creditcard.csv")
# https://www.Rkaggle.com/mlg-ulb/creditcardfraud
# Print the top 5 records
print(datainput[0:5])
# Print the complete shape of the dataset
print("Shape of Complete Data Set")
print(datainput.shape)
```

```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9  …       V21       V22       V23       V24       V25  \
0  0.098698  0.363787  … -0.018307  0.277838 -0.110474  0.066928  0.128539
1  0.085102 -0.255425  … -0.225775 -0.638672  0.101288 -0.339846  0.167170
2  0.247676 -1.514654  …  0.247998  0.771679  0.909412 -0.689281 -0.327642
3  0.377436 -1.387024  … -0.108300  0.005274 -0.190321 -1.175575  0.647376
4 -0.270533  0.817739  … -0.009431  0.798278 -0.137458  0.141267 -0.206010

        V26       V27       V28  Amount  Class
0 -0.189115  0.133558 -0.021053  149.62      0
1  0.125895 -0.008983  0.014724    2.69      0
2 -0.139097 -0.055353 -0.059752  378.66      0
3 -0.221929  0.062723  0.061458  123.50      0
4  0.502292  0.219422  0.215153   69.99      0

[5 rows x 31 columns]
Shape of Complete Data Set
(284807, 31)
```

[2]:
```
cls = datainput.get("Class")
false = datainput[cls == 1]
```

```
true = datainput[cls == 0]
n = len(false) / float(len(true))
print(n)
print("False Detection Cases: {}".format(len(datainput[cls == 1])))
print("True Detection Cases: {}".format(len(datainput[cls == 0])))
```

```
0.0017304750013189597
False Detection Cases: 492
True Detection Cases: 284315
```

[3]:
```
# False Detection Cases
print("False Detection Cases")
print("----------------------")
print(false.Amount.describe())
# True Detection Cases
print("True Detection Cases")
print("----------------------")
print(true.Amount.describe())
```

```
False Detection Cases
----------------------
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
True Detection Cases
----------------------
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

[4]:
```
# separating features(X) and Label(y)
# Select all columns except the last for all rows
X = datainput.iloc[:, :-1].values
# Select the last column of all rows
Y = datainput.iloc[:, -1].values
print(X.shape)
```

```
print(Y.shape)
```

```
(284807, 30)
(284807,)
```

[5]:
```
from sklearn.model_selection import train_test_split

# train_test_split method
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
```

[6]:
```
from sklearn import metrics

# DecisionTreeClassifier
from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier(max_depth=4)
classifier.fit(X_train, Y_train)
predicted = classifier.predict(X_test)
print("predicted values :", predicted)

# Accuracy
DT = metrics.accuracy_score(Y_test, predicted) * 100
print("The accuracy score using the DecisionTreeClassifier : ", DT)
```

```
predicted values : [0 0 0 … 0 0 0]
The accuracy score using the DecisionTreeClassifier :  99.9367999719111
```

[7]:
```
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score

# Precision
print("precision")
# Precision = TP / (TP + FP) (Where TP = True Positive, TN = True Negative, FP =
precision = precision_score(Y_test, predicted, pos_label=1)
print(precision_score(Y_test, predicted, pos_label=1))
# Recall
print("recall")
# Recall = TP / (TP + FN)
recall = recall_score(Y_test, predicted, pos_label=1)
print(recall_score(Y_test, predicted, pos_label=1))
# f1-score
print("f-Score")
# F - scores are a statistical method for determining accuracy accounting for␣
 ↪bot
fscore = f1_score(Y_test, predicted, pos_label=1)
print(f1_score(Y_test, predicted, pos_label=1))
```

```
precision
```

```
0.8160919540229885
recall
0.7802197802197802
f-Score
0.797752808988764
```