import pandas as pd

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#https://www.kaggle.com/mlg-ulb/creditcardfraud

# Print the top 5 records

print(datainput[0:5],"\n")

# Print the complete shape of the dataset

   print("Shape of Complete Data Set")

   print(datainput.shape,"\n")

import pandas as pd

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

false = datainput[datainput['Class'] == 1]

true = datainput[datainput['Class'] == 0]

n = len(false)/float(len(true))

print(n)

print('False Detection Cases: {}'.format(len(datainput[datainput['Class'] == 1])))

print('True Detection Cases: {}'.format(len(datainput[datainput['Class'] == 0])),"\n")

import pandas as pd

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#Check for imbalance in data

false = datainput[datainput['Class'] == 1]

true = datainput[datainput['Class'] == 0]

#False Detection Cases

print("False Detection Cases")

print("----------------------")

print(false.Amount.describe(),"\n")

#True Detection Cases

print("True Detection Cases")

print("----------------------")

print(true.Amount.describe(),"\n")

import pandas as pd

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#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)

import pandas as pd

from sklearn.model\_selection import train\_test\_split

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#separating features(X) and label(y)

X = datainput.iloc[:, :-1].values

# Select the last column of all rows

Y = datainput.iloc[:, -1].values

#train\_test\_split method

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2)

import pandas as pd

from sklearn import metrics

from sklearn.model\_selection import train\_test\_split

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#separating features(X) and label(y)

X = datainput.iloc[:, :-1].values

Y = datainput.iloc[:, -1].values

#train\_test\_split method

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2)

#DecisionTreeClassifier

from sklearn.tree import DecisionTreeClassifier

classifier=DecisionTreeClassifier(max\_depth=4)

classifier.fit(X\_train,Y\_train)

predicted=classifier.predict(X\_test)

print("\npredicted values :\n",predicted)

#Accuracy

DT = metrics.accuracy\_score(Y\_test, predicted) \* 100

print("\nThe accuracy score using the DecisionTreeClassifier : ",DT)

import pandas as pd

from sklearn import metrics

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

#Load the creditcard.csv using pandas

datainput = pd.read\_csv('E:\\creditcard.csv')

#separating features(X) and label(y)

X = datainput.iloc[:, :-1].values

Y = datainput.iloc[:, -1].values

#train\_test\_split method

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2)

#DecisionTreeClassifier

from sklearn.tree import DecisionTreeClassifier

classifier=DecisionTreeClassifier(max\_depth=4)

classifier.fit(X\_train,Y\_train)

predicted=classifier.predict(X\_test)

print("\npredicted values :\n",predicted)

#

# #Accuracy

DT = metrics.accuracy\_score(Y\_test, predicted) \* 100

print("\nThe accuracy score using the DecisionTreeClassifier : ",DT)

#

# #Precision

print('precision')

# Precision = TP / (TP + FP) (Where TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative).

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 both precision and recall.

fscore = f1\_score(Y\_test, predicted, pos\_label=1)

print(f1\_score(Y\_test, predicted, pos\_label=1))