CREDIT CARD FRAUD DETECTION

May 5, 2024

1 CREDIT CARD FRAUD DETECTION

[1]:
```
import numpy as np import pandas as pd import sklearn
import matplotlib.pyplot as plt import seaborn as sns
import tkinter as tk from tkinter import filedialog
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import
classification_report,accuracy_score from
sklearn.preprocessing import StandardScaler from
sklearn.metrics import accuracy_score from
sklearn.preprocessing import StandardScaler
```

1.1 About Dataset

This is a simulated credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. It covers credit cards of 1000 customers doing transactions with a pool of 800 merchants. saction datasets.saction datasets.saction datasets.

[2]:
```
trd = pd.read_csv(r'C:/Users/gagan/Downloads/archive/fraudTrain.csv')
```

[3]:
```
trd.head(5)
```

[3]:
```
 Unnamed: 0 trans_date_trans_time        cc_num \
0          0 2019-01-01 00:00:18 2703186189652095
1          1 2019-01-01 00:00:44 630423337322 2 2 2019-
           01-01 00:00:51 38859492057661
3          3 2019-01-01 00:01:16 3534093764340240
4          4 2019-01-01 00:03:06   375534208663984


                         merchant      category    amt     first \
0         fraud_Rippin, Kub and Mann    misc_net    4.97  Jennifer
1         fraud_Heller, Gutmann and Zieme    grocery_pos 107.23
          Stephanie 2   fraud_Lind-Buckridge entertainment 220.11
            Edward
3              fraud_Kutch, Hermiston and Farrell gas_transport
                45.00 Jeremy
4              fraud_Keeling-Crist     misc_pos    41.96 Tyler


     last gender                       street …      lat      long \
```

```
0   Banks  F     561 Perry Cove … 36.0788 -81.1781
1   Gill   F 43039 Riley Greens Suite 393 … 48.8878 -118.2105 2
    Sanchez M    594 White Dale Suite 530 … 42.1808 -112.2620
3   White  M     9443 Cynthia Court Apt. 038 … 46.2306 -112.1138
4   Garcia M     408 Bradley Rest … 38.4207 -79.4629

   city_pop                          job       dob \
0       3495      Psychologist, counselling 1988-03-09
1       149 Special educational needs teacher 1978-06-21
2       4154      Nature conservation officer 1962-01-19
3       1939      Patent attorney 1967-01-12
4       99  Dance movement psychotherapist 1986-03-28

                       trans_num unix_time merch_lat merch_long \
0  0b242abb623afc578575680df30655b9 1325376018 36.011293 -82.048315
1  1f76529f8574734946361c461b024d99 1325376044 49.159047 -
   118.186462
2  a1a22d70485983eac12b5b88dad1cf95 1325376051 43.150704 -
   112.154481
3  6b849c168bdad6f867558c3793159a81 1325376076 47.034331 -
   112.561071
4  a41d7549acf90789359a9aa5346dcb46 1325376186 38.674999 -78.632459

   is_fraud
0         0
1         0
2         0
3         0
4         0

[5 rows x 23 columns]
```

1.2 Dataset Infromation

[4]:
```
trd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1296675 entries, 0 to
1296674 Data columns (total 23
columns):
 #   Column                Non-Null Count   Dtype
--- ------                --------------   -----
 0   Unnamed: 0   1296675 non-null int64
 1   trans_date_trans_time 1296675 non-null object
 2   cc_num                 1296675 non-null
                           int64
```

```
 3   merchant              1296675 non-null
                           object
 4   category              1296675 non-null
                           object
 5    amt                  1296675       non-null
                           float64
 6   first                 1296675 non-null
                           object
 7   last                  1296675 non-null
                           object
 8   gender                1296675 non-null
                           object
 9   street                1296675 non-null
                           object
 10 city                   1296675 non-null
                           object
 11 state                  1296675 non-null
                           object
 12 zip                    1296675 non-null
                           int64
 13 lat                    1296675       non-null
                           float64
 14 long                   1296675       non-null
                           float64
 15 city_pop               1296675 non-null
                           int64
 16 job                    1296675 non-null
                           object
 17 dob                    1296675 non-null
                           object
 18 trans_num              1296675 non-null
                           object
 19 unix_time              1296675 non-null
                           int64
 20 merch_lat              1296675       non-null
                           float64
 21 merch_long             1296675       non-null
                           float64
 22 is_fraud               1296675 non-null
                           int64
dtypes: float64(5), int64(6), object(12)
memory usage: 227.5+ MB
```

1.3 Checking for number of missing values in each column

[5]:
```
trd.isnull().sum()
```

```
[5]: Unnamed: 0 0
trans_date_trans_time 0
cc_num     0 merchant 0
category   0 amt 0 first
0 last     0 gender   0
street     0 city     0
state 0 zip 0 lat 0 long
0 city_pop 0 job 0 dob 0
trans_num  0 unix_time
0 merch_lat      0
merch_long 0 is_fraud 0
dtype: int64
```

2 Exploratory Data Analysis

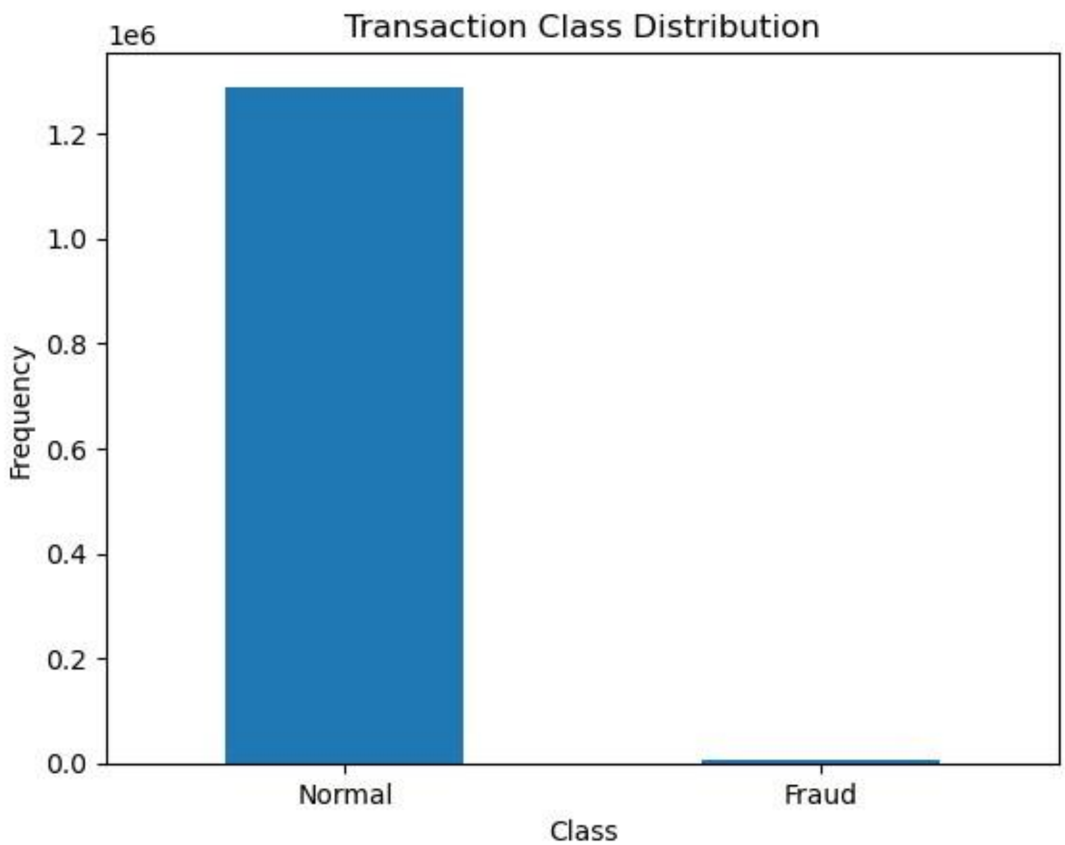
2.0.1 Distribution of Legit and Fraudlent transaction

```
[6]: trd['is_fraud'].value_counts()
```

```
[6]: is_fraud
0      1289169
1      7506
Name: count, dtype: int64
```

```
[7]: LABELS = ["Normal", "Fraud"] count_classes =
pd.Series(trd['is_fraud']).value_counts(sort=True)
count_classes.plot(kind = 'bar', rot=0)
plt.title("Transaction Class Distribution")
plt.xticks(range(2), LABELS) plt.xlabel("Class")
plt.ylabel("Frequency")
```

```
[7]: Text(0, 0.5, 'Frequency')
```



2.0.2 Seperating the data for Analysis

[8]:
```
legit = trd[trd.is_fraud==0]
fraud = trd[trd.is_fraud==1]
```

[9]:
```
legit.shape
```

[9]: (1289169, 23)

[10]:
```
fraud.shape
```

[10]: (7506, 23)

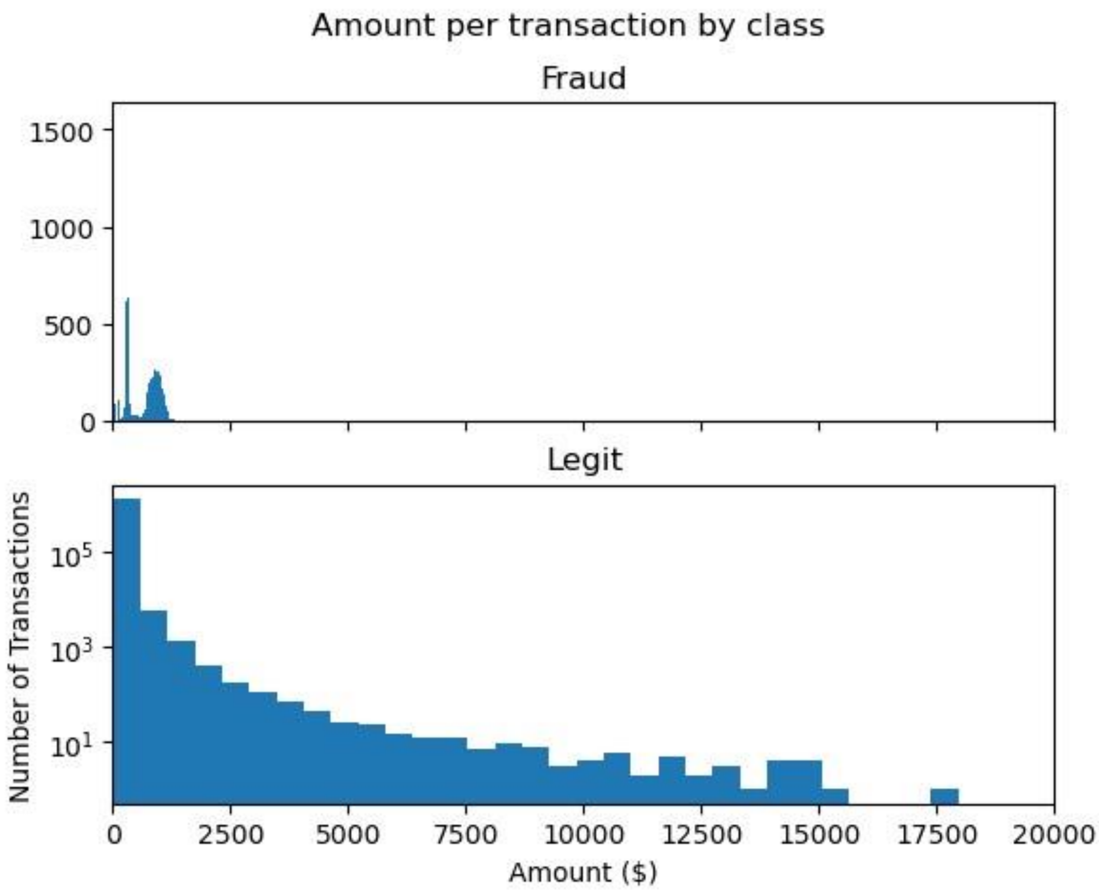
2.0.3 Statistical measures of the data

[11]:
```
legit.amt.describe()
fraud.amt.describe()
```

[11]:
```
count 7506.000000
mean     531.320092
std      390.560070
min        1.060000
25%      245.662500
50%      396.505000
```

```
75%      900.875000
max     1376.040000
Name: amt, dtype: float64
```

[12]:
```
f, (ax1, ax2) = plt.subplots(2, 1, sharex=True)
f.suptitle('Amount per transaction by class')
bins = 50
ax1.hist(fraud.amt, bins = bins)
ax1.set_title('Fraud')
ax2.hist(legit.amt, bins = bins)
ax2.set_title('Legit')
plt.xlabel('Amount ($)')
plt.ylabel('Number of Transactions')
plt.xlim((0, 20000))
plt.yscale('log')
plt.show()
```



2.0.4 Undersampling

[13]:
```
new_legit = legit.sample(n=492)
new_legit
new_df = pd.concat([new_legit,fraud], axis=0)
new_df.sample(6)
new_df['is_fraud'].value_counts()
```

[13]:
```
is_fraud
1    7506
0     492
Name: count, dtype: int64
```

[14]:
```
trd = new_df.drop(columns='is_fraud', axis=1)
ted = new_df['is_fraud']
print(trd)
print(ted)
```

```
         Unnamed: 0 trans_date_trans_time             cc_num  \
350954       350954 2019-06-13 23:27:53   6506116513503136
820313 820313 2019-12-08 20:07:56 36485887555770 949368
949368   2020-01-15   09:52:21   376012912828093   752917
752917 2019-11-18 02:12:56 4319584480204988982
536816       536816 2019-08-18 16:09:25     30238755902988
…              …                   …                 …
1295399    1295399 2020-06-21 01:00:08   3524574586339330
1295491    1295491 2020-06-21 01:53:35   3524574586339330
1295532    1295532 2020-06-21 02:16:56   4005676619255478
1295666    1295666 2020-06-21 03:26:20   3560725013359375
1295733    1295733 2020-06-21 03:59:46   4005676619255478

                                   merchant       category     amt  \
350954                  fraud_Pouros-Haag  shopping_pos    5.73
820313   fraud_Hyatt, Russel and Gleichner health_fitness  7.84
949368                  fraud_Bradtke PLC   grocery_pos   76.25
752917                    fraud_Brown PLC      misc_net    5.83
536816             fraud_Jakubowski Group   food_dining    4.03
…                                      …             …       …
1295399     fraud_Kassulke PLC    shopping_net     977.01 1295491
 fraud_Schumm PLC shopping_net 1210.91
1295532 fraud_Tillman, Dickinson and Labadie gas_transport 10.24
1295666 fraud_Corwin-Collins gas_transport 21.69
1295733                fraud_Koss and Sons gas_transport   10.20

             first    last gender                     street …    zip \
350954 Kimberly    Rice  F     63991 Destiny Rue Apt. 651 … 75703
820313 Michael     Gross M     230 Ryan Tunnel Apt. 025 … 43321
```

```
949368    Nathan   Mayer    M 478 Donovan Corners Apt. 803 … 60193
752917 Kathleen Nash F 010 Salazar Walk … 41810 536816 Danielle Yu
F 5395 Colon Burgs Suite 037 … 76578
…          …     …    …                        … …    …
1295399   Ashley Cabrera    F 94225 Smith Springs Apt. 617 … 32960
1295491   Ashley Cabrera    F 94225 Smith Springs Apt. 617 … 32960
1295532  William   Perry    M 458 Phillips Island Apt. 768 … 70726
1295666   Brooke   Smith     F   63542 Luna Brook Apt. 012 … 79759
1295733  William   Perry    M 458 Phillips Island Apt. 768 … 70726

            lat     long city_pop                       job        dob \
350954  32.2768 -95.3031 144160 Sports development officer 1984-05-04
820313 40.4971 -82.8342 267   Facilities manager 2005-01-29 949368
 42.0144 -88.0935 92294 Claims inspector/assessor 1969-05-01
752917  37.1788 -82.6950     502  Chief Financial Officer 1960-02-01
536816  30.5920 -97.2893    1766                Press sub 1976-01-02
…         …       …       …                         …        …
1295399 27.6330 -80.4031  105638        Librarian, public 1986-05-07
1295491 27.6330 -80.4031  105638        Librarian, public 1986-05-07
1295532 30.4590 -90.9027   71335                Herbalist 1994-05-31
 1295666 31.8599 -102.7413    23         Cytogeneticist 1969-09-15
 1295733 30.4590 -90.9027   71335               Herbalist 1994-05-31
                    trans_num unix_time merch_lat merch_long 350954
  fe97939179beb525d0cfcb99a34bae0b 1339630073 33.096473 -95.361257
8203130dfcdcdd00053889a38ef9339035bdb3 1354997276 41.459637 -82.502199
949368a98e009459e3745897744b68b95d76af 1358243541 41.051417 -89.086272
75291777bf83b9ec8476c6386e1d826d7d3d58 1353204776 36.938058 -82.502641
536816b39b7b92ea61f73344ea42785cb66ece 1345306165 30.431111 -97.577077
…                                   …         …        …        …
1295399 a83b093f0c1d9068fa0089f7c722615f 1371776408 26.888686 -
80.834389
1295491 f75b35bed13b9e692f170dba45a15b21 1371779615 28.216707 -
79.855648
1295532 a0ba2472cd3fc9731f2a18d3f308f5c3 1371781016 29.700456 -
91.361632
1295666 daa281350b1e16093c7b4bf97bf4d6ed 1371785180 32.675272 -
103.484949
1295733 0c1c20470fc0d16019b5c368cadf563a 1371787186 31.363252 -
89.932309

[7998 rows x 22 columns]
350954    0
820313    0
949368    0
```

```
    752917    0
    536816    0
..
    1295399   1
    1295491   1
    1295532   1
    1295666   1
    1295733   1
    Name: is_fraud, Length: 7998, dtype: int64
```

[15]:
```
numerical_columns = ['amt', 'lat', 'long', 'city_pop', 'unix_time',␣
 ↪'merch_lat', 'merch_long']
trd = trd[numerical_columns]
print(trd.head(5))
for i in trd:
    print(i)
```

```
           amt      lat      long  city_pop   unix_time  merch_lat  merch_long
350954   5.73  32.2768  -95.3031    144160  1339630073  33.096473  -95.361257
820313   7.84  40.4971  -82.8342       267  1354997276  41.459637  -82.502199
949368  76.25  42.0144  -88.0935     92294  1358243541  41.051417  -89.086272
752917   5.83  37.1788  -82.6950       502  1353204776  36.938058  -82.502641
536816   4.03  30.5920  -97.2893      1766  1345306165  30.431111  -97.577077
amt
lat
long
city_pop
unix_time
merch_lat
merch_long
```

[16]:
```
X_train, X_test, Y_train, Y_test = train_test_split(trd, ted, test_size=0.1,␣
 ↪random_state=2)
print(X_train.shape, X_test.shape)
```

```
(7198, 7) (800, 7)
```

2.1 Model Training Using LogisticRegression

[17]:
```
model = LogisticRegression()
model.fit(X_train.values, Y_train)
```

[17]:
```
LogisticRegression()
```

[18]:
```
X_train_prediction = model.predict(X_train.values)
training_data_accuracy = accuracy_score(X_train_prediction,
Y_train) print('Accuracy on Training data : ',
training_data_accuracy)
```

```
Accuracy on Training data : 0.9373437065851625
```

[19]:
```
X_test_prediction = model.predict(X_test.values)
test_data_accuracy = accuracy_score(X_test_prediction,
Y_test) print('Accuracy on Testing data:
',test_data_accuracy)
```

```
Accuracy on Testing data: 0.94875
```

[30]:
```
def predict_credit_card_scam(input_data): input_data =
    input_data[numerical_columns]  input_data_scaled  =
    sklearn.preprocessing.StandardScaler().
 ↪fit_transform(input_data) predictions =
    model.predict(input_data_scaled) return
    predictions
def get_file(): file_path = filedialog.askopenfilename() if
    file_path: selected_file_label.config(text="Selected
    file: " + file_path) global ted
        ted = pd.read_csv(file_path)
        print("File loaded into 'ted'
        variable.")
        ted2 = ted
numerical_columns = ['amt', 'lat', 'long', 'city_pop', 'unix_time',␣
 ↪'merch_lat', 'merch_long'] ted =
        ted[numerical_columns] ted.columns =
        numerical_columns predictions =
        predict_credit_card_scam(ted) for i
        in predictions:
```

```
            if i==1:
                print(f"Fraud {i}")
            else:
                print(f"Genuine {i}")
root = tk.Tk()
button = tk.Button(root, text="Select your test file", command=get_file)
button.pack(pady=100)
selected_file_label = tk.Label(root, text="Selected file: None")
selected_file_label.pack()
root.geometry("300x300")
root.mainloop()
```