Assignment 1 - Fraud Detection Dataset

Exploratory Data Analysis

01 November 2020

Yihong Qiu

Cosimo Cambi

Craig Perkins

Noelani Roy

Northeastern University
College *of* Professional Studies

Where did the data come from?

Our dataset comes from Kaggle - https://www.kaggle.com/kartik2112/fraud-detection

Why did you choose this data?

We had an hour long zoom meeting to go through a couple contender datasets. The finalists were:

- Credit Card Transactions Fraud Detection Dataset - https://www.kaggle.com/kartik2112/fraud-detection

- Synthetic Financial Datasets For Fraud Detection - https://www.kaggle.com/ntnu-testimon/paysim1

- Top Personality Dataset - https://www.kaggle.com/arslanali4343/top-personality-dataset

- Suicide Rates Overview 1985 to 2016 - https://www.kaggle.com/russellyates88/suicide-rates-overview-1985-to-2016

- Credit Card Fraud Detection - https://www.kaggle.com/mlg-ulb/creditcardfraud

- Crimes in Boston - https://www.kaggle.com/AnalyzeBoston/crimes-in-boston

- National Renewable Energy Laboratory's (NREL) PV Rooftop Database - https://registry.opendata.aws/nrel-oedi-pv-rooftops/

We ultimately picked the first one on the list because it contains 23 columns and a large number of rows (1,296,675 rows) and we thought we would be able to utilize most or all of the methods presented in the class on the dataset. There is another popular Credit card fraud dataset on Kaggle (https://www.kaggle.com/mlg-ulb/creditcardfraud) that we considered, but the columns in the dataset contain only numerical inputs variables which are the result of a PCA transformation due to confidentiality issues. It would be interesting to try the models and thinking we deploy on the first dataset to the other credit card fraud dataset and see if we can port the model to another dataset successfully.

<table>
<tr><th>Columns in the dataset</th><th>Sample row</th></tr>
<tr><td>

column	dtype
`trans_date_trans_time`	object
`cc_num`	int64
`merchant`	object
`category`	object
`amt`	float64
`first`	object
`last`	object
`gender`	object
`street`	object
`city`	object
`state`	object
`zip`	int64
`lat`	float64
`long`	float64
`city_pop`	int64
`job`	object
`dob`	object
`trans_num`	object
`unix_time`	int64
`merch_lat`	float64
`merch_long`	float64
`is_fraud`	int64
`txn_datetime`	datetime64[ns]
`txn_date`	object
`date_of_birth`	datetime64[ns]

</td><td>

column	value
trans_date_trans_time	2019-01-01 00:00:18
cc_num	2703186189652095
merchant	fraud_Rippin, Kub and Mann
category	misc_net
amt	4.97
first	Jennifer
last	Banks
gender	F
street	561 Perry Cove
city	Moravian Falls
state	NC
zip	28654
lat	36.0788
long	-81.1781
city_pop	3495
job	Psychologist, counselling
dob	1988-03-09
trans_num	0b242abb623afc578575680df30655b9
unix_time	1325376018
merch_lat	36.011293
merch_long	-82.048315
is_fraud	0
txn_datetime	2019-01-01 00:00:18
date_of_birth	1988-03-09
year_of_birth	1988
txn_date	2019-01-01

</td></tr>
</table>

Not only can we try to predict fraudulent transactions with this dataset, we may also be able to build models around consumer behavior that could be of interest to advertisers. This dataset also contains categories separated from in person and on the net transactions so we may be able to analyze trends in consumer behavior. Along with consumer behavior, this dataset contains data until July this year so we may be able to gather insights into how the pandemic affects consumer behavior. This is simulated data, so I am not sure if the pandemic was incorporated into the synthesis of the data, but that will certainly be an exercise we perform in our exploratory data analysis.

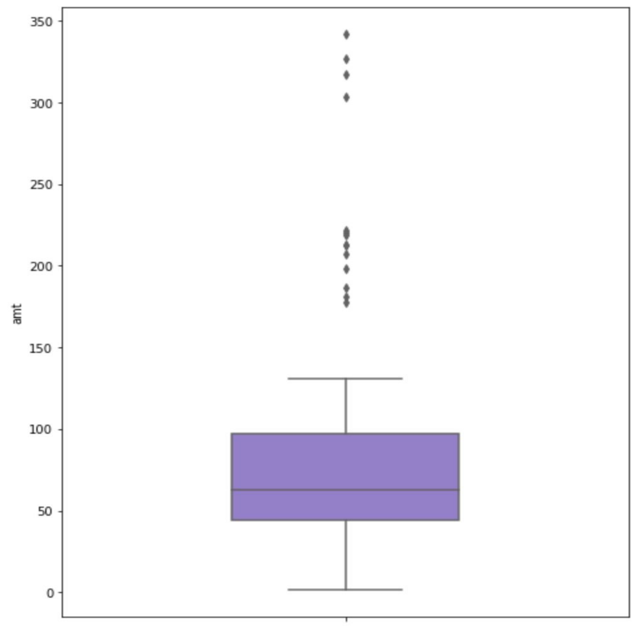
Ultimately, we think this dataset is rich enough for us to deploy the knowledge we learn in this class and utilize all techniques introduced. Furthermore, financial fraud detection is a great example of how Machine Learning is used in the real world and this dataset could be a good avenue into seeing what challenges financial institutions are presented with in keeping their customers safe.

What did you do with the data in the context of exploration?

In this exploratory phase we really just wanted to get acclimated with our dataset that we will be using for the next 6 weeks. In the first few analyses performed we analyzed things like:

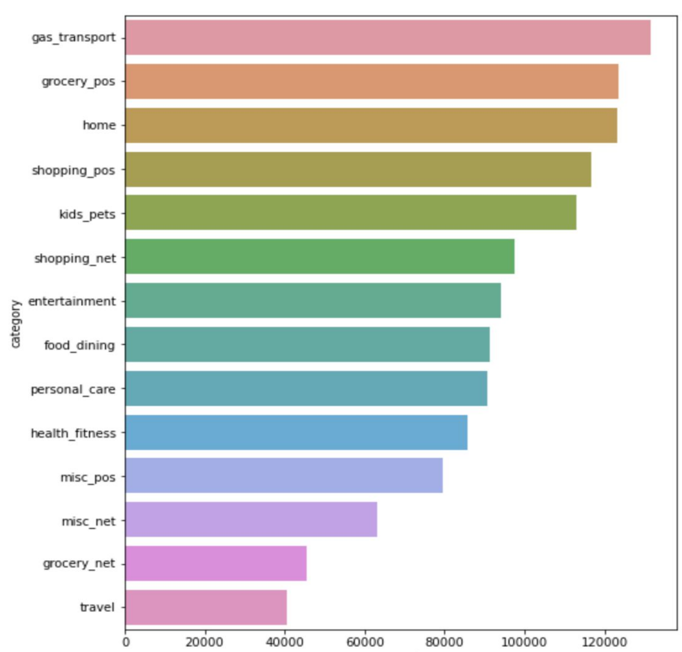
1) Distribution of the amount of transactions

```
plt.figure(figsize=(8,10))
sns.boxplot(y='amt', data=fraud_df.head(100), width = 0.4, color= 'mediumpurple')
```



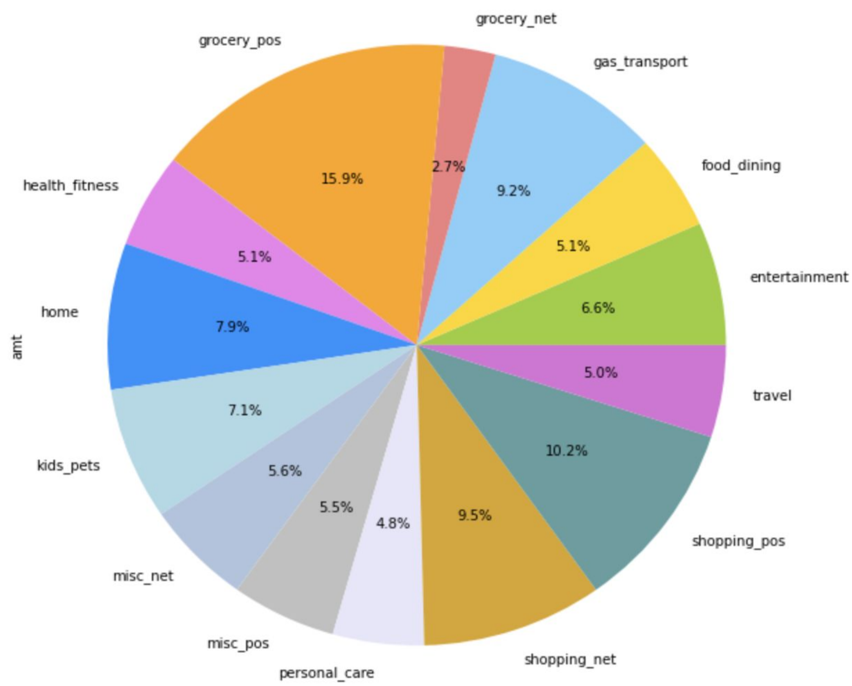
2) Distribution of the category of transactions

```
plt.figure(figsize=(8,10))
sns.countplot(y="category", data=fraud_df, order= fraud_df['category'].value_counts().index)
```



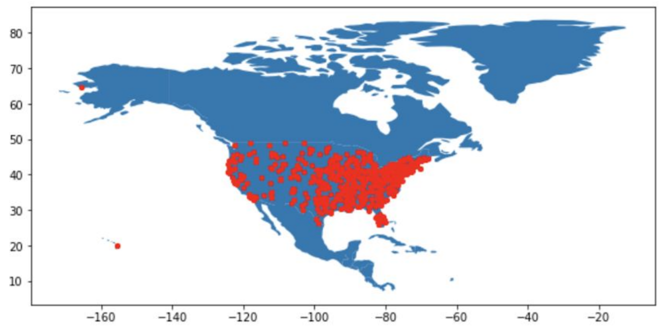
3) Distribution of the amount of transactions by category

```
cs = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral','orange','violet',
        'dodgerblue', 'lightblue','lightsteelblue','silver', 'lavender','goldenrod',
        'cadetblue','orchid']
category_amt = fraud_df.groupby("category")["amt"].sum()
category_amt.plot.pie(autopct="%.1f%%", colors=cs, figsize=(10, 12))
```



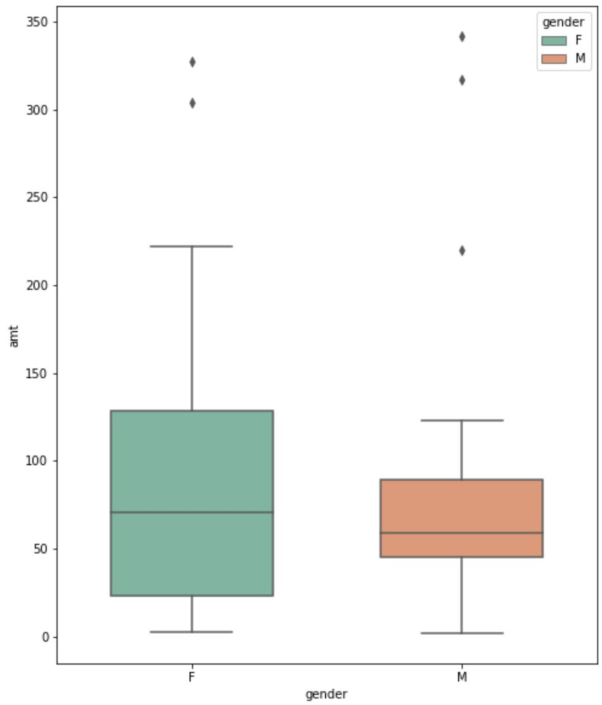
4) Seeing where the transactions took place and plotting on a map using GeoPandas

```
In [11]: from shapely.geometry import Point
         import geopandas as gpd
         from geopandas import GeoDataFrame

         geometry = [Point(xy) for xy in zip(fraud_df.head(1000)['long'], fraud_df.head(1000)['lat'])]
         gdf = GeoDataFrame(fraud_df.head(1000), geometry=geometry)

         #this is a simple map that goes with geopandas
         world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
         ax = world[world.continent == 'North America']
         gdf.plot(ax=ax.plot(figsize=(10, 6)), marker='o', color='red', markersize=15);
```



5) Distribution of the volume of transactions based on gender. It would also be interesting to see the total amount spent by gender.

```
gender_amt = pd.DataFrame(fraud_df.head(100), columns = ['amt', 'gender'])

plt.figure(figsize=(8,10))
sns.boxplot(y='amt', x='gender', data=gender_amt, hue='gender', dodge=False, width = 0.6, palette= 'Set2')
```



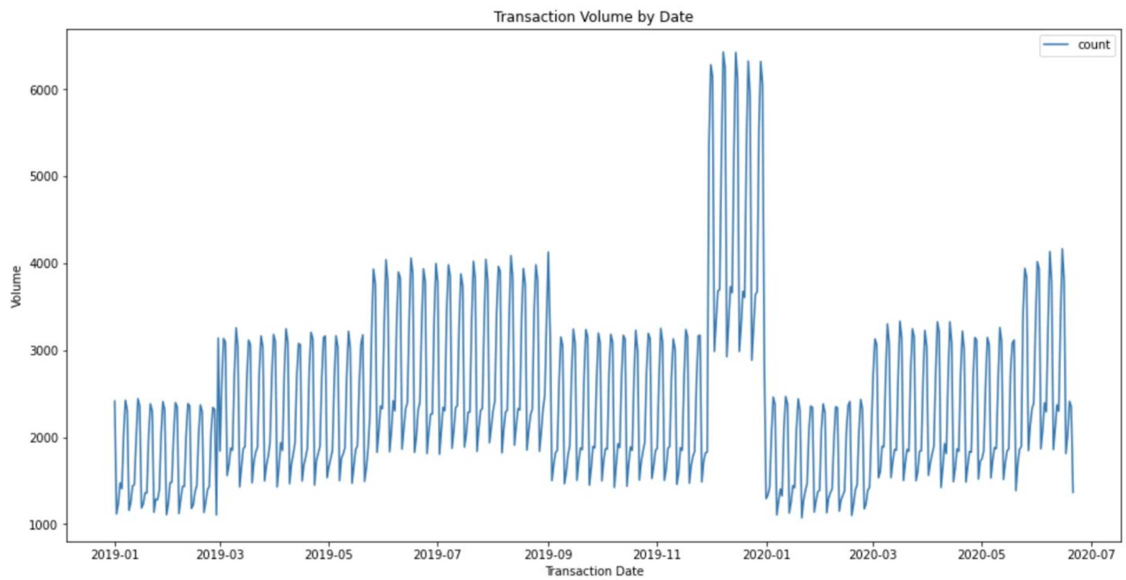
6) Occupation of the highest spenders

7) Volume of transactions by day. Similar to gender it would also be great to see this displayed as dollar amount along with the number of transactions.

In [12]:
```
txn_count_by_date = fraud_df['txn_date'].value_counts().sort_index().reset_index()
txn_count_by_date.columns = ['txn_date','count']

# qgrid.show_grid(txn_count_by_date.head(100), grid_options={'forceFitColumns': False, 'defaultColumnWidth': 100})
```

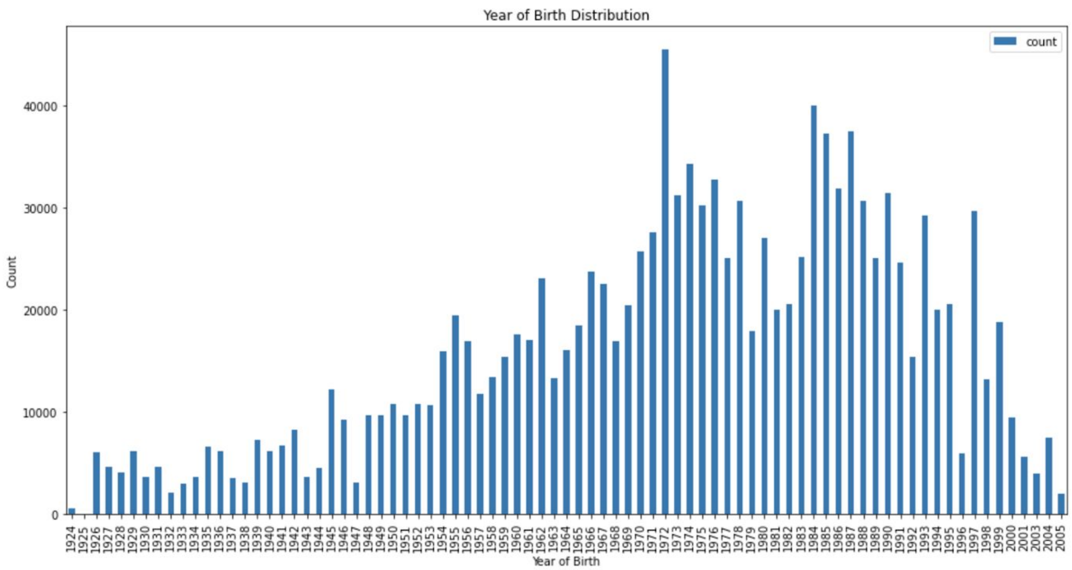
In [11]:
```
plt = txn_count_by_date.plot.line(x='txn_date', y='count', figsize=(16, 8), title="Transaction Volume by Date")
plt.set_xlabel('Transaction Date')
plt.set_ylabel('Volume')
```

Out[11]:
```
Text(0, 0.5, 'Volume')
```



8) Age distribution of the spenders in the dataset

```
In [13]: dob_counts = fraud_df['year_of_birth'].value_counts().sort_index().reset_index()
         dob_counts.columns = ['year_of_birth','count']

         plt = dob_counts.plot.bar(x='year_of_birth', y='count', figsize=(16, 8), title="Year of Birth Distribution")
         plt.set_xlabel('Year of Birth')
         plt.set_ylabel('Count')

Out[13]: Text(0, 0.5, 'Count')
```



9) Number of unique users and merchants in the dataset

```
In [11]: # Number of merchants in the dataset

         print(f"Number of merchants: {fraud_df['merchant'].nunique()}")

         # Number of cards in the dataset

         print(f"Number of cards: {fraud_df['cc_num'].nunique()}")

         # Number of cards in the dataset

         print(f"Number of unique users: {fraud_df.groupby(['first', 'last', 'gender', 'street', 'city']).ngroups}")

         Number of merchants: 693
         Number of cards: 983
         Number of unique users: 983
```

10) Merchants with the highest number of transactions and highest dollar amount of transactions

```
In [16]: fraud_df.groupby(['merchant'])['amt'].agg('sum').nlargest(10)

Out[16]: merchant
         fraud_Kilback LLC                     391078.15
         fraud_Bradtke PLC                     302481.25
         fraud_Doyle Ltd                       300971.37
         fraud_Hackett-Lueilwitz               300208.14
         fraud_Schumm, Bauch and Ondricka      299115.14
         fraud_Rau and Sons                    298354.77
         fraud_Goodwin-Nitzsche                298083.31
         fraud_Pacocha-O'Reilly                297584.38
         fraud_Murray-Smitham                  296982.73
         fraud_Bauch-Raynor                    295721.20
         Name: amt, dtype: float64
```

11) Spenders with the highest number of transactions and highest dollar amount of transactions

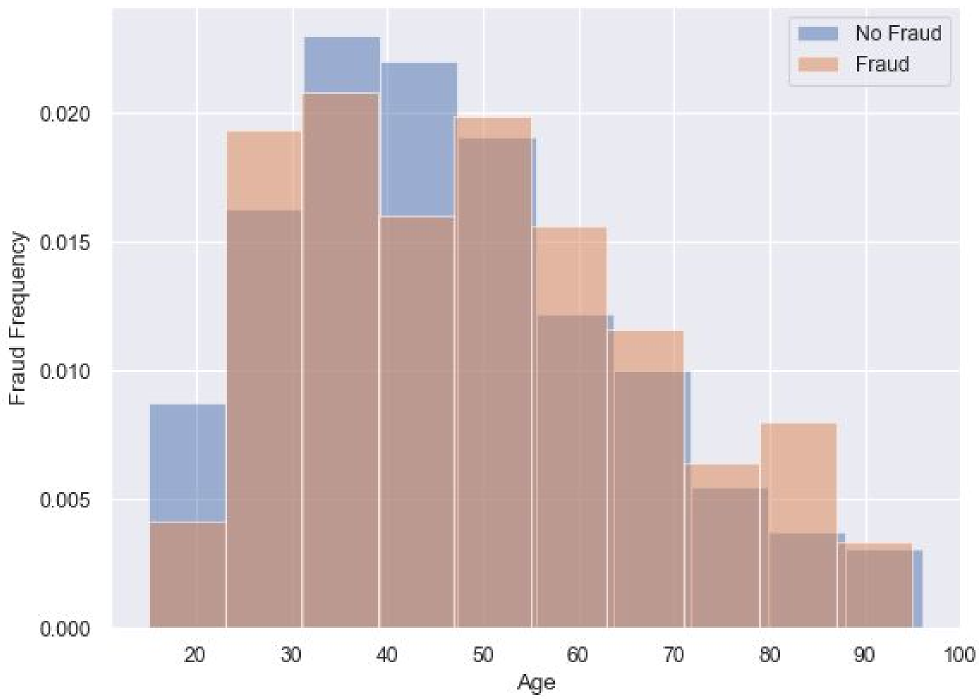
```
In [15]: fraud_df.groupby(['cc_num', 'first', 'last'])['amt'].agg('sum').nlargest(10)

Out[15]: cc_num             first    last
         6011367958204270   Tammy    Ayers       296436.73
         4908846471916297   Lauren   Torres      290478.49
         6011438889172900   Allison  Allen       284013.50
         36722699017270     Jessica  Perez       280008.05
         6011893664860915   Erin     Chavez      278325.97
         6011109736646996   Rebecca  Erickson    278139.27
         3583635130604947   Crystal  Gamble      278042.99
         2712209726293386   Jenna    Brooks      277085.65
         4836998673805450   Susan    Hardy       275930.63
         372509258176510    Kristen  Hanson      275889.68
         Name: amt, dtype: float64
```

12) In the training set we identified how many transactions were flagged as fraudulent and it would be good to analyze these cases more in depth for patterns

What did you find? Why does that matter?

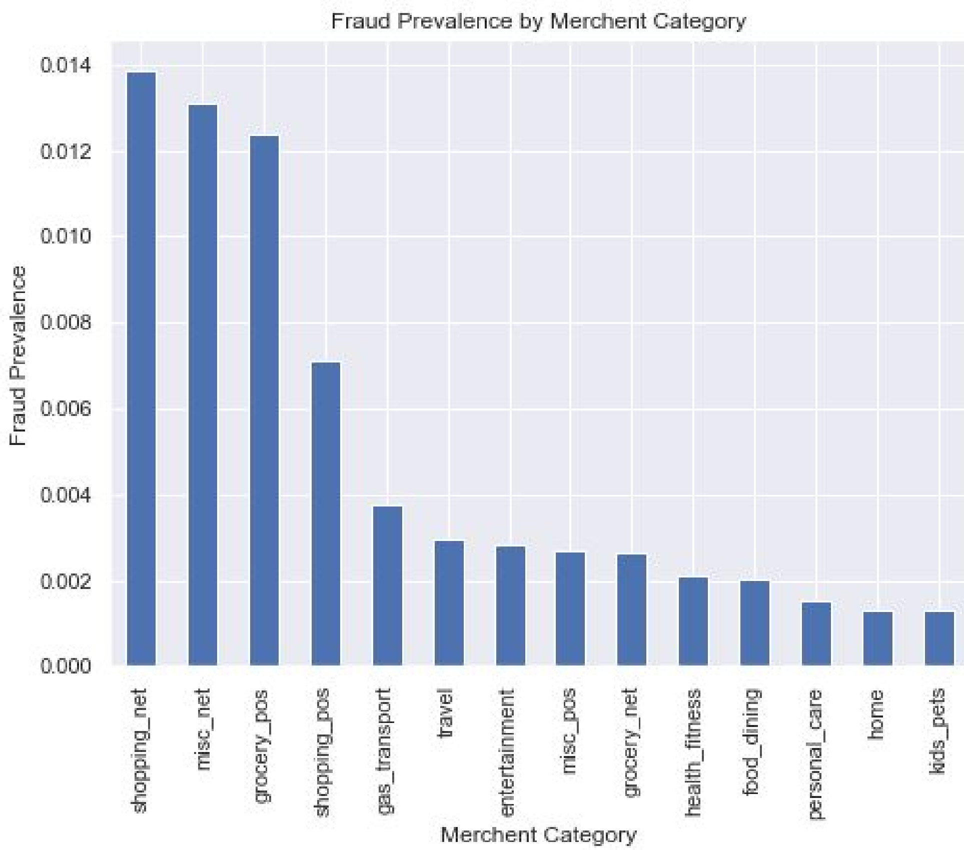
To find meaningful information from the data we wanted to explore the data in a manner that could possibly guide us into where to focus our predictive models later in the course. Initially we wanted to see how fraudulent credit card charges related to the features that we have available. The three areas that we looked at were how age was correlated with fraud, how merchant categories were corelated with fraud, and if there was an age range and merchant category combination where fraud was more prevalent. These insights could help us understand where to focus our efforts and how these features are related to each other. Due to the large data set, a sample size of 10% of the data was taken to complete this analysis.

First, we plotted two frequency distribution histograms to see how the age distribution differed between fraudulent and non-fraudulent credit card transactions.



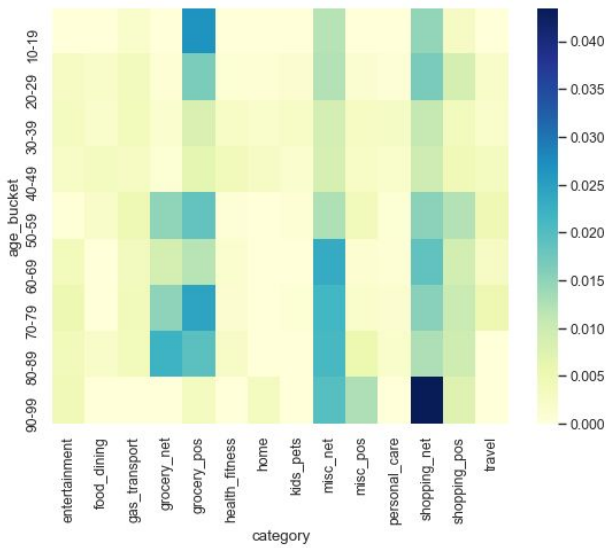
Here we can see that for the most part the age distribution of fraudulent charges vs not is similar. However, there is a key spike in fraudulent charges that starts in the 70-79 age range and then in the 80-89 age range.

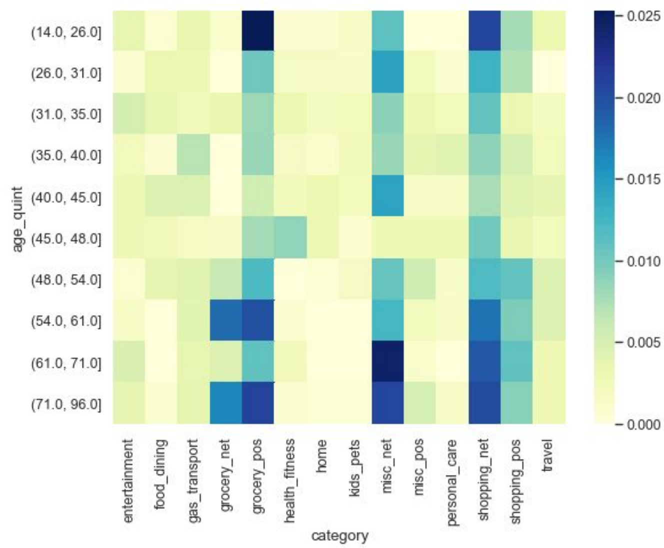
Next, we looked at the percentage of fraudulent charges across the various merchant categories.



Interestingly there are some categories that had a relatively higher percentage than the others. It is not surprising that online shopping was the most susceptible to fraudulent charges, however it was surprising to see that grocery stores were also high in that chart.

Lastly, we also reviewed the combination of age ranges and merchant categories. In this part we new broke the ages by both ten-year buckets and then into deciles. The 10-year buckets allow us to easily read where age and merchant categories may overlap, however creating deciles, allows for a more focused and even distribution of the data.





Reviewing the charts, it is interesting to see how the fraudulent charge percentage changes depending on how the ages are broken out and where the hot spots are.

What would your proposed next steps be?

Exploratory Data Analysis (EDA) has led us to understand the structure and some of the content within the dataset. The next steps that we will take will be to clean up and pre-process the data. We will be looking for Missing Values, Anomalies, Duplicates, and a Class Imbalance. Our EDA has shown that we have no missing values or duplicates (Figure 1.0).

<u>Figure 1.0</u>

```
duplicate_rows_df = df[df.duplicated()]
print("number of duplicate rows:",duplicate_rows_df.shape)
```

```
number of duplicate rows: (0, 18)
```

```
print(df.isnull().sum())
```

```
merchant      0
category      0
amt           0
first         0
last          0
gender        0
street        0
city          0
state         0
zip           0
lat           0
long          0
job           0
dob           0
unix_time     0
merch_lat     0
merch_long    0
is_fraud      0
dtype: int64
```

A Class Imbalance is where an item I am looking for, such as fraud, has an uneven distribution within the dataset. This can cause machine learning algorithms to have a low predictive accuracy. We are at risk of having a Class Imbalance in this dataset, due to the low percentage of identified fraud when compared to the total length of the dataset (shown in Figure 2.0).

<u>Figure 2.0</u>

```
In [52]:
isFraudTotal=df['is_fraud'].sum()
print("Count of Fraud:",isFraudTotal)
total_rows = df['merchant'].count()
print("Total Rows:",total_rows)
```

```
Count of Fraud: 2145
Total Rows: 555719
```

Our next steps to solve this will be to evaluate the following techniques to correct this imbalance: Over Sampling, Under Sampling, and SMOTE.

We have also identified some outliers within our dataset, which we will evaluate for either removal or normalization to ensure the accuracy of our eventual algorithm. Outliers are anything that does not fall within the minimum and maximum range as defined by the following equation.

Equation 1.0

minimum: ‘Quartile 1’ – 1.5 * (‘Quartile 3’ – ‘Quartile 1’)

maximum: ‘Quartile 3’ – 1.5 * (‘Quartile 3’ – ‘Quartile 1’)

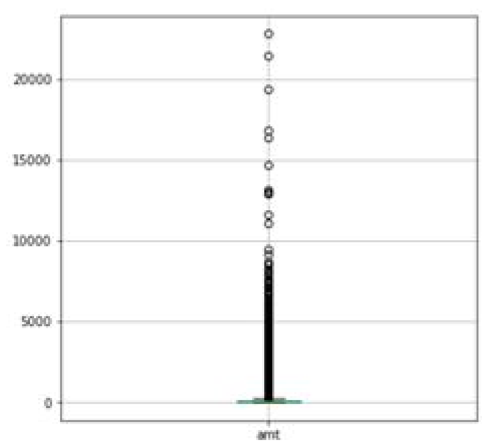
To understand our outliers, we must calculate them by column. In this case, I am going to evaluate outliers for both the ‘amt’ column, which shows the transaction amount and the ‘unix_time’ column which shows our date range. Using a boxplot, I can visualize my interquartile range and with a df.describe() function, I can see some of the relevant numbers to calculate my outliers.

Figure 3.0: Transaction Amount

```
stats = df['amt'].describe()
print("Transaction Amount:")
print(stats)
df.boxplot(column='amt', figsize=(6, 6))
```

```
Transaction Amount:
count    555719.000000
mean         69.392810
std         156.745941
min           1.000000
25%           9.630000
50%          47.290000
75%          83.010000
max       22768.110000
Name: amt, dtype: float64
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e5a05a56d0>
```



In the case of transaction amount, I am viewing a significant number of outliers. My next steps here would be to use this information to remove or normalize outliers from the amount column.

After my dataset is suitably clean, I would start laying out my framework for how I am going to develop my algorithm. We are interested in setting up an algorithm for Pattern Recognition, which would detect classes, clusters, and patterns of suspicious behavior. This could help us identify characteristics most often found in fraudulent transactions and patterns in consumer spending.

We could plot out a consumer's path to see patterns in a single user's spending habits. Based on a single user's spending habits it would be interesting to see if we could predict a big life change such as new occupation, birth of a child, etc.

What business problem are you intending to solve using ML with the data?

In 2018, fraudulent credit card transactions cost $24.26 billion dollars worldwide. Experts have projected that these numbers will continue to grow to over $35 billion dollars over the next three years (1). With the level of fraud on the rise, it becomes imperative for a credit card company to get better at detecting and preventing these transactions. Not only do fraudulent credit card transactions cost credit card companies a lot of money, but it also causes unhappiness with their clientele, making them less likely to open new cards with that institution or recommend them to friends. Dealing with a fraudulent transaction posted to your account takes time and energy to resolve out of their day.

1. Which features are highly correlated to credit card fraud?

Correlation matrix and regression models will be used to find out which features, such as gender, age and state, have strong correlations with the target variable is_fraud, which can provide banks more details about which group of people might be the target in credit card fraud.

2. To predict whether the user's credit card will be frauded or not.

Tree-based models, such as random forest, decision tree and gradient boosting models, and neural networks will be applied to predict the accuracy of target variable is_fraud. We will also compare and seek the best performance among these predictive models by using different Machine Learning Algorithms.

Citations

(1) Credit card fraud statistics: What are the odds?

Letić, Budanović, & Jovanović

https://dataprot.net/statistics/credit-card-fraud-statistics/