# Project: RIFT

Real-Time Identification of Fraudulent Transactions
Visa Capstone Group - Fraud Detection
Sayesha Aravapalli, Truett Bloxsom, Qingzi Zeng, Matthew Zlotnik

## Executive Summary

The issue that our project focuses on is Visa's cross-border cashout detection algorithms. Currently, Visa is not able to properly identify the scope of fraud within their system as it occurs, and as a result, they ultimately freeze more accounts/regions than necessary, costing large amounts of revenue by way of foreign and atm transaction fees. The question at hand is how can we use Visa's past data to create a model that looks at the most recent hour (Visa's time interval of choice) of data and identify which accounts and regions are at highest risk of fraud. The expected business value to Visa is not only proper identification to reduce frozen accounts in fraudulent events, but also to have an ever more accurate way of reducing fraud loss from such events. The data made available to our team was 18 months of cross border ATM transactions including transaction amount, masked account data, transaction time and location, and fraud label. To extract value from this data, our team took a two-pronged approach. Firstly, we used clustering algorithms such as Kmeans and DBSCAN to profile cashout events and identify the scale of cashout events within an hour of the first fraudulent transaction. Secondly, we used graphical analysis metrics such as number of connections against a baseline level to identify specific nodes at the highest risk of fraudulent transaction, then employed NetworkX with colorized hierarchical nodes to create a visualization of the sector(s) of Visa's network which were at particularly high risk. Ultimately, both of these analyses were able to deliver on their intended outcomes, with the clustering algorithms projected to save Visa $5.7 million and the graphical analysis algorithm projected to save Visa $20-25 million with a majority of that being reducing false positives i.e. allowing millions of dollars of non-fraudulent transactions to be transacted instead of being blocked by the supervised model that is currently in place. Our recommendations for Visa are to run our algorithms hourly on their network to detect pockets of fraudulent transactions, then deliver those results to the business operations team to implement proper network freezes. We believe that if Visa implements our suggestions, they will be able to provide a more reliable and more secure banking network to all of their clients conducting cross-border transactions.

# Articulate Business Outcomes

Whenever a transaction is completed with a Visa-backed card, a signal is sent to the acquirer bank (whichever bank processes transactions for the merchant), which then forwards the details of the transaction to Visa's credit card network, which checks the transaction for suspicious behavior, and if the transaction passes fraud detection, then Visa sends a signal to the bank that issued the consumer's card to authorize the movement of funds from the consumer to the merchant. Any given ATM transaction can be traced to an issuer bank, a bin of accounts within that bank, a specific account, an ATM, a city, and a country. Throughout this report, we will refer to any specific instance of one of these entities as a "node" and any of the types listed as "levels". When Visa detects a fraud in their system, they have the ability to freeze all transactions at any level. Currently, they shut down transactions at the largest levels, the issuer bank (also referred to as bid), and the large grouping of accounts (bin). They do this because their algorithms focus primarily on identification of fraud on a transaction-to-transaction basis and thus they cannot properly identify the scale and interconnectedness of the fraud.

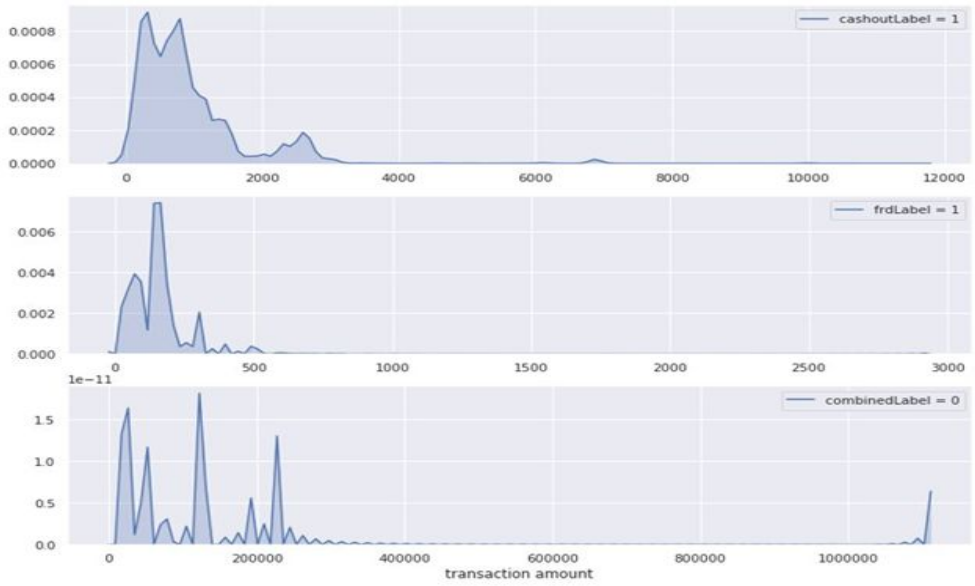
The primary stakeholders impacted by this solution are non-targeted Visa customers, who will not have to experience service disruptions from fraud events which do not involve them. Also, targeted visa customers as well as fraudsters will be impacted, as having targeted anti-fraud tactics will reduce the time taken to thwart these cashout attacks. Additionally, Visa's shareholders will be affected as less costly cashout events and fewer restricted-but-not-risky transactions could increase the Earnings Per Share (EPS). Our expected benefits to these stakeholders are that, by identifying fraudulent clusters rather than simply transactions, we could help Visa to freeze all transactions in only those necessary nodes while still ensuring that all nodes at risk of fraud are frozen. In doing this, we could save Visa money both on lost cash in fraud and in lost ATM/international fees by not shutting down nodes that are at low risk of fraud. To calculate the exact value of our project, we utilized Visa's spreadsheets which contain specific data about the handling of past cashout events. This data includes number of blocked transactions, method of blocking, and total dollars blocked, allowed, and lost in fraud. By comparing the recommendations of our models against the actions taken by Visa, we can project the difference in lost/gained revenues as a result of our model.

To take advantage of our models, Visa will integrate the code into their present systems, so that, as transactions come into the Visa network, they are clustered and analyzed to produce risk scores, cluster analytics, and network diagrams. These outputs are then sent to the Visa business team to decide which nodes or which levels of the Visa network need to be frozen in order to minimize fraudulent loss.

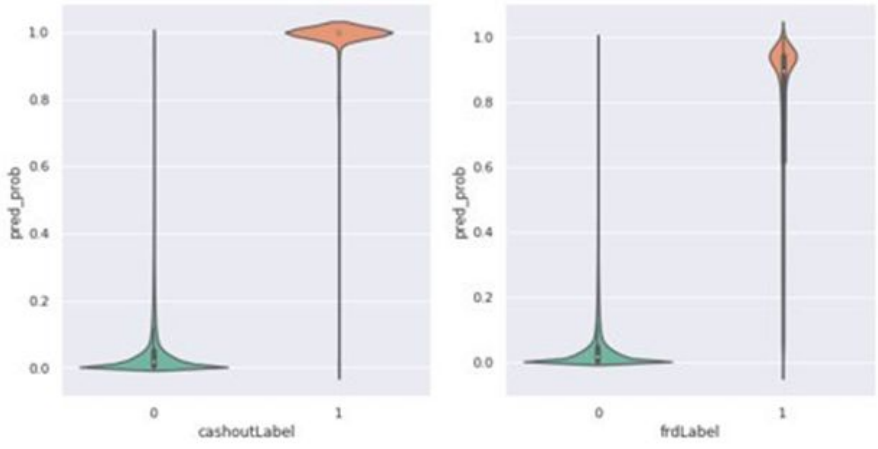
# Understanding Source Data

The data set we worked with was cleaned well and contains 47 million rows and 27 elements split into a train, test, and validation set. The rows of our data are individual transactions between a physical card and an ATM conducted in a country other than the one in which the card was issued. Our response variables are a binary indicator of fraud and a more specific binary indicator of whether or not the transaction was one of many in a cashout event, which is constituted by a large string of coordinated fraudulent charges. Our independent variables are the time, location, issuer bank, bin, account, and transaction amount. For any account or bank information, we were given masked versions of the data to protect customer privacy. Additional columns in the data included AA_score, which is a Visa-calculated metric which measures the riskiness of an individual transaction based on all available data. To process our data, we used a remote Jupyter notebook server which required specific Visa VPN access. As mentioned above, our data is expertly cleaned, and as such we have no missing values in the entirety of the data. We did, however, run into issues with our geographic variables. There was no standardized way of writing city names, and as such, we had several instances of one city name being listed many different ways. Additionally, as most of our data were categorical, we did not believe that collinearity of predictor variables would pose much of an issue. Regardless, we inspected our non-categorical variables and thankfully found no distinct instances of correlation between predictors

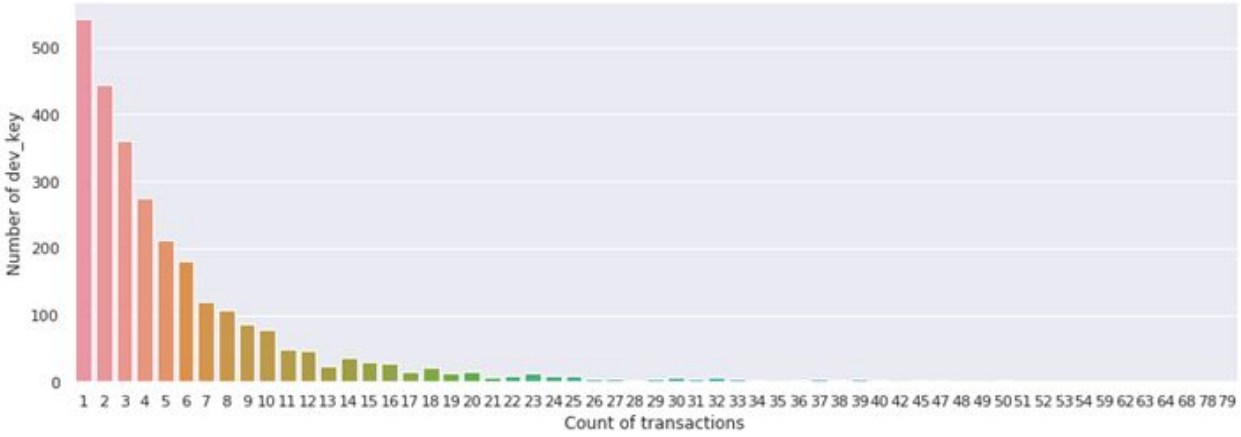
The biggest issue we ran into with our data was that our fraudulent data class was extremely rare, with cashouts constituting less than .1% of total transactions. In our analyses section we will discuss the many different methods we used to try and combat this problem. Additionally, some attributes of transactions, such as the account number or the unique atm, were very sparse, with nearly as many unique account IDs and ATMs as transactions in the dataset. Because of this, our models were not particularly effective at checking for collaborative fraud at the ATM or account level. Also, because 24 out of 27 features in the existing data set are categorical, we needed to create numerical features from raw data in order to build clustering models like K-means. To do this, we created new fields based on time. For example, the seconds it takes for a given account before it performs the next transaction (time till next transaction), count of transactions for a given account or ATM in the last hour (count features), and sum of transaction amount for a given account or ATM in the last 3 hour (sum features). For time till next fraud features, we needed to take the inverse of the time to smooth the distribution. We were able to use these engineered features as continuous variables in our analyses, which allowed our clustering methods to work as intended.

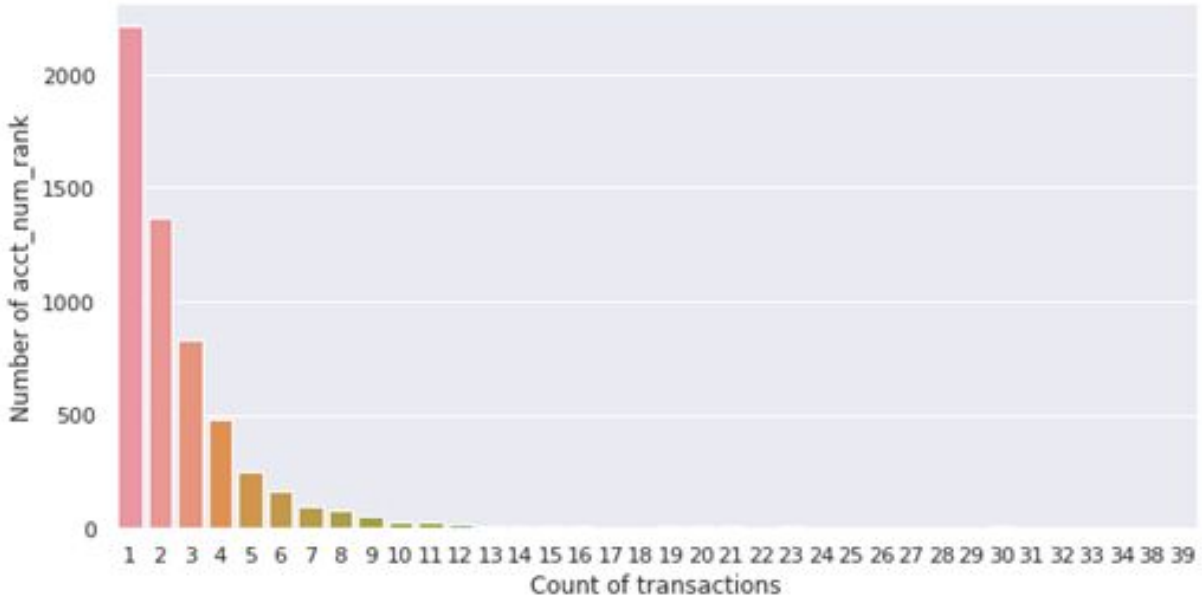



To discover exactly what a fraudulent transaction would look like, and how fraudulent, non-fraudulent, and cashout transactions all differed in profile, we graphed the distributions of our predictor variables contingent on our fraud label being equal to one. As you can see from the first graph, fraudulent transactions tend to have smaller transaction amounts, with a large fraction of transactions less than $500, while non-fraudulent transactions are more discretized, likely along the recommended withdrawal amounts listed at that ATM. Another interesting note from the transaction amount graph is that cashout transactions also typically involve smaller total transaction amounts, with most transactions under $3000, while the range of typical transaction amounts for non-fraudulent transactions exceeds $200,000. Additionally, as shown in the graph below, we can trust that Visa's predicted fraud probability metric is extremely effective at capturing individual fraudulent and cashout transactions. Normal transactions have an average predicted probability of fraud of .056, while fraudulent transactions have an average predicted probability of fraud of .752 and cashout transactions have a value of .977.



In our analyses, we also learned that cashout events take place on thousands of ATMs using thousands of accounts to spread the fraudulent withdrawal of funds. As you can see in the graphs below, typically an ATM, and especially an account, will only be used once or twice during a singular cashout event. If our data were all atm withdrawals, this would make distinguishing fraudulent from non-fraudulent transactions especially difficult. Our data, however, were only from cross-border ATM withdrawals, in which it is exceedingly rare, especially for an account, to withdraw money multiple times in the span of a few hours. With the combined knowledge of transaction count, and the engineered count, time, and sum features we decided that our predictor variables would likely be sufficient for our chosen methods of clustering and network analyses.




One of the key risks we identified in a meeting with Dr. Mitchell was that graphical analyses and clustering methods may not have been the best way to identify the fraud patterns in

our data, as simple supervised learning may be more effective at detecting fraud. To proceed with this risk in mind, we verified that our project goal was specifically to identify clusters of fraudulent transactions and not just fraud on a single-transaction basis. Additionally, the implication of creating a system to detect clusters of fraud real-time meant that we likely would not have access to the true fraud labels of data before having to identify the cluster in which each transaction belonged. One key ethical dilemma we encountered was that, when suggesting ways for a bank to block fewer accounts during a fraud/cashout event, we run the risk of possibly exposing more customers to these fraud attacks. If we do find certain patterns of fraud and create an alert system to stop said fraud, we have to ensure that we give more weight to the value of preventing a fraud against the value of a lost transaction fee.

## Analyses Section

To conduct our analyses, we first had to deconstruct our problem. To find out exactly what the possible indicators of collaborative fraud were, we first had to learn all we could about collaborative fraud motives and methods. After several meetings, including one long in-person meeting, we finally understood how collaborative frauds occurred, and could proceed with our project. Next, we had to plan how best to use the massive datasets we were given. After doing our EDA and realizing that Visa could accurately identify singular fraudulent transactions with their predicted probability metric, we came up with an idea. Because cashout and fraudulent transactions were extremely rare in our larger dataset, we could select only transactions with a value of predicted probability higher than a designated threshold. Because our data were time series, and our engineered features relied on having the transactions immediately prior to and after each transaction, we had to complete feature engineering before applying this filter. While this cost a significant amount of time and processing power, we believed it to be worthwhile because it only had to be conducted once, and it allowed us to properly subset our data without sacrificing predictive power.

Next, we had to decide which modeling techniques would be most appropriate in attacking our problem. We knew that the models would have to be effective at detecting patterns and groups of fraudulent transactions in an ocean of non-fraudulent transactions. Firstly, we knew that clustering algorithms would be a natural fit, as we hypothesized that clusters of fraudulent transactions would be detectable even amidst the massive scale of data. For pattern recognition, we used Kmeans and DBSCAN clustering algorithms to try and identify cashout transactions on a level-by-level basis. Kmeans allowed us to employ high-dimensional data, and ultimately was the inspiration behind engineering our count, time, and sum features, as Kmeans typically performs much better with continuous numerical features. We knew that Kmeans would be the most flexible model, so we chose DBSCAN as our other clustering model. Density Based Spatial Clustering with Added Noise is a unique algorithm which detects regions of high-density data and distinguishes them from lower density regions. As sparsity becomes

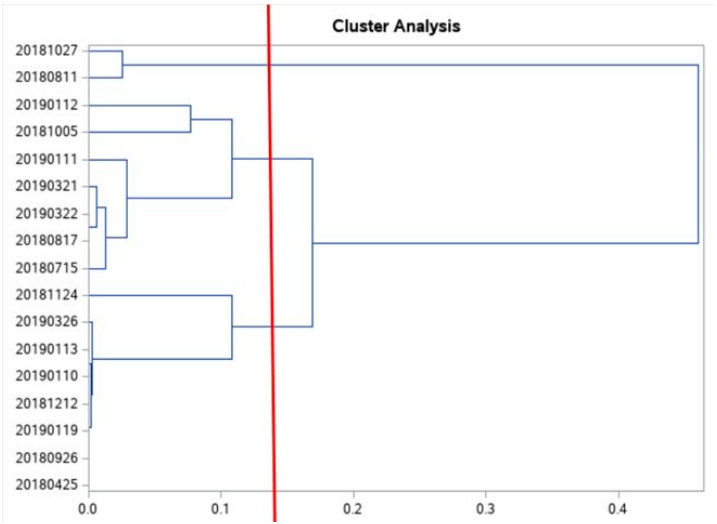
exponentially more apparent as dimensionality of data increases, we decided that DBSCAN would perform well only utilizing a few of our engineered continuous features to detect unique patterns.

To complement our clustering algorithms, we decided to employ the service of NetworkX. This package allowed us to analyze aspects of the Visa network which would have otherwise gone unnoticed, such as the number of unique connections to a node in an hour, and total amount of dollars flowing through any series of nodes in a given amount of time. With these new metrics, we believed that NetworkX would allow us to recognize any unusual activity in the Visa network as it was developing.

The ultimate tangible goal for each of the three modeling techniques we decided to use was to create a tangible alerting system which, when analyzed by members of the Visa team, could recommend Bins, Cities, Countries, or combinations thereof in which Visa could freeze all transactions and create the highest true positive rate and also lowest false negative rate. As each of these models are incredibly unique, we initially struggled at finding a proper metric which could compare and contrast the advantages of each model. Ultimately, after requesting more data from our sponsors, we discovered that we could create an average transaction amount, and then by attaching a value to each of true and false positives and negatives, we could calculate the average expected value created for each of our models for any given day in our dataset. Ultimately, our hypotheses for each of these modeling techniques were that DBSCAN would allow us to locate low-dimensional regions of high density in the data and identify regions at higher risk of fraud, K-means would allow us to divide our data into high-dimensional predefined numbers of clusters based on similarity in many different predictors and generate special cashout clusters, and NetworkX would allow us to analyze differences in Visa's network over time which could allow for the immediate detection of cashout networks.

## Types of Cashout Events

To obtain a general idea of cashouts, we calculated the basic statistics like risk, dollar amount, and scope of cashout events in our dataset. We used hierarchical clustering to allocate cashout events into 3 groups. The camouflaged cashouts are cashouts that look like normal transactions, and they have the lowest risk and highest duration compared to other cashouts.



Disastrous cashouts happen in a few hours and result in huge losses. Small cashouts happen most frequently, and usually within a small scope and lead to small loss.

| | Camouflaged Cashouts | Disastrous Cashouts | Small Cashouts |
|---|---|---|---|
| Loss | Moderate | Huge | Small |
| Risk | Low <0.3 | High >0.8 | High >0.6 |
| Duration | 5-9h | 4-5h | 1-2h |
| Countries | 2~10 | 20+ | 1~2 |
| Cities | <100 | 200+ | <50 |
| ATMs | <1100 | >1100 | <500 |

## DBSCAN

DBSCAN is a density-based clustering non-parametric algorithm: given a set of points in some space, it groups together points that are closely packed together and marking as outliers points that lie alone in low-density regions. DBSCAN has its own set of advantages:

1. It does not require one to specify the number of clusters in the data a priori, unlike with k-means.
2. It can find arbitrarily shaped clusters.
3. It has a notion of noise, and is robust to outliers.

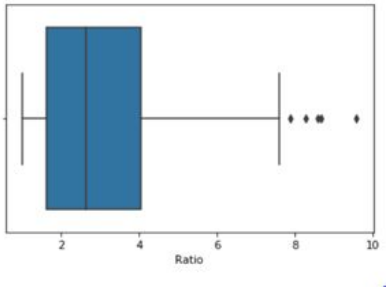
The intuition behind using DBSCAN is that cashout transactions would be pretty similar to each other and hence form clusters, whereas normal transactions would be pretty different and hence would be classified as outliers. As it is a distance based algorithm, numerical features work better for DBSCAN. The 'count' features were used i.e the features were the number of counts in the past one hour for each entity. To train our model based on the 3 types of cashout events, the train set we chose had an equal number of transactions of camouflaged, disastrous and small cashout events.The total number of transactions used to train were 4041. On modeling we expect some clusters to have mostly cashout transactions, i.e we expect pure clusters.This will help us in decreasing the False Positives (FP) i.e classifying normal transactions wrongly as cashout transactions.

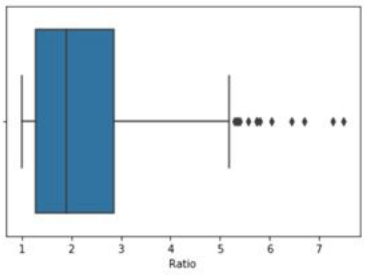
$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

DBSCAN requires two parameters: ε (eps) and the minimum number of points required to form a dense region (minPts). It starts with an arbitrary starting point that has not been visited. This point's ε-neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as noise. Note that this point might later be found in a sufficiently sized ε-environment of a different point and hence be made part of a cluster. The metric used to choose between different parameter combinations was the overall gini index. Gini index is as follows.

The overall gini index was calculated as the weighted average of the gini index of each cluster weighted by its size. Here, Gini index was used because it give a sense of separability in the classes.The lower the gini index the better.

The objective is to detect fraud in real time and in real time one would not have the ground truth labels to identify cashouts clusters. So, clustering would be done on 1 hour data. To identify the cashout clusters we came up with a 2 metrics BIN ratio and BID ratio. Based on the parameters, hour by hour dbscan modeling was done to identify patterns.Bid Ratio is the number of data points over the number of unique bids in the cluster. Similarly for bin ratio. To find an appropriate threshold, we looked at these ratios for normal hours. We used 7 days of data, clustered them hour by hour and calculated the ratios. The boxplot of the ratios is as follows :




a) Bid Ratio b) Bin ratio

The maximum value for Bid ratio and bin ratio are 7.5 and 9.5 respectively, hence these were the first set of thresholds to identify if the cluster had cashout transactions.

**BID ratio**

Bid ratio < 7.5 - Cluster with mainly normal transactions

8 < Bid ratio < 20 - Cluster with both normal and cashout transactions

Bid ratio > 20 - Cluster with mostly cashout transactions with few bins

**BIN ratio**

Bin ratio < 10 - Cluster with mainly normal transactions

10 < Bin ratio < 30 - Cluster with both normal and cashout transactions

Bin ratio > 30 - Cluster with mostly cashout transactions with few bins

The same set of parameters (epsilon and minpoints) were used for the cashout dates in the train dataset. The data was clustered hourly and a second set of thresholds was chosen. These are 20 and 30 for bid and bin respectively. So any cluster with a ratio above these are likely to be 95% cashout transactions (cashout cluster). The cashout clusters are profiled to see the possible bins and bids. Once the cashout bids and bins are identified, the countries related to these specific transactions can look into to identify the countries.

On testing the above, it is observed that the model did a good job in terms of giving 95% cashout clusters. But it failed to identify cashout transactions when there were very few cashout transactions happening in that particular hour irrespective of the overall scale/type of the cashout event.

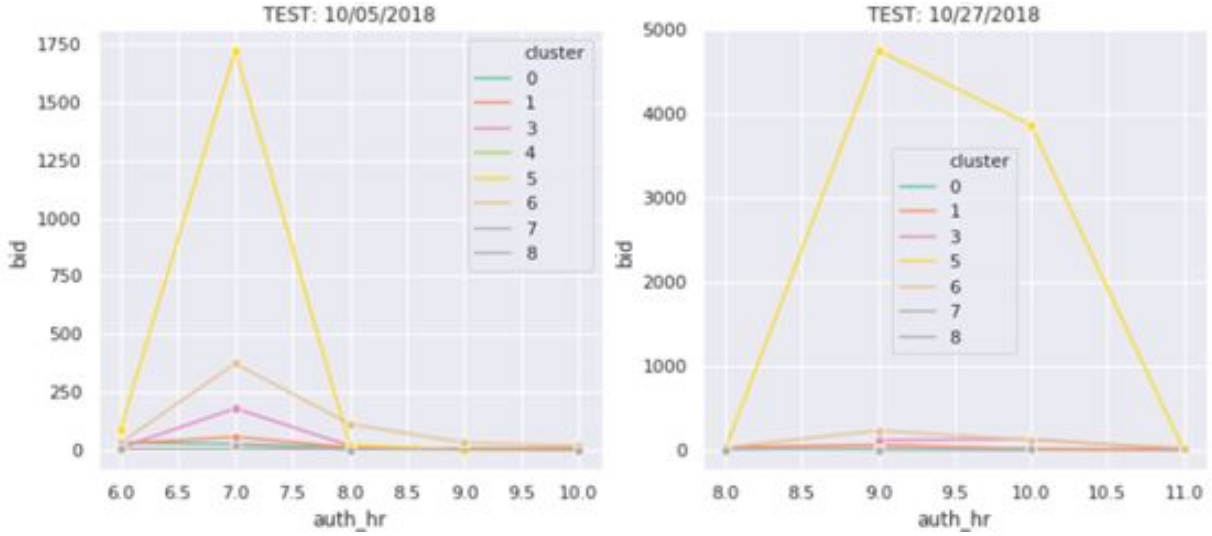
## K-Means

For Kmeans, we realized that over 90% of our features were categorical, and since Kmeans deals exclusively with numerical data, we knew that some intense preprocessing was required. As mentioned before, we calculated new fields based on time since the last transaction and count of transactions in the last hour for each entity in our data (an entity being a bid, bin, pan, atm, city, or country). In the K-means model, we have 2 raw features and 30 derived features.

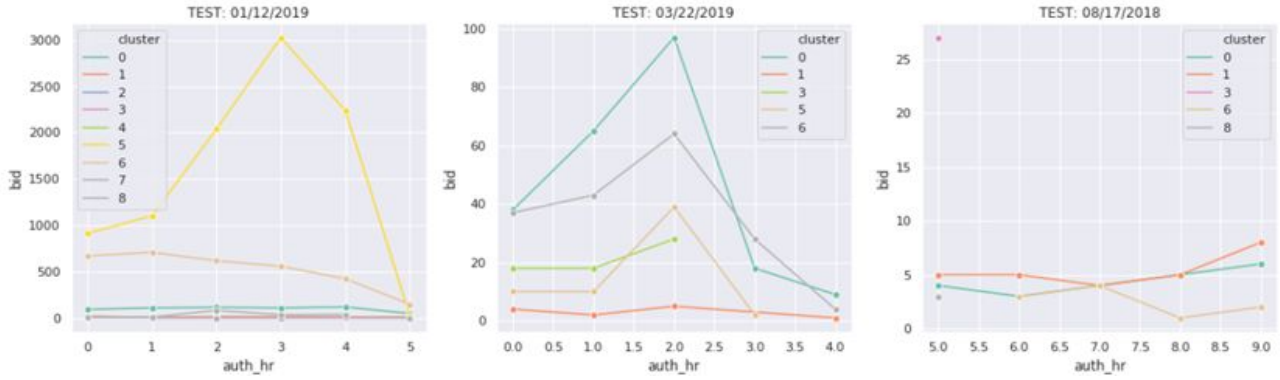
To construct the training set, we tried value-weighted and equal-weighted models. Value-weighted models weigh camouflaged, disastrous and small cashout events based on the number of transactions within each group; equal-weighted models weigh 3 types of cashout events equally. Then, using a combination of provided and calculated fields, we ran a kmeans clustering algorithm which only clustered entities with a risk value above our designated threshold. To decide on this threshold, we cross validated using Gini Scores and selected the threshold and number of clusters which combined provided the highest overall weighted Gini Index. We use 8 cashout events to train the K-means model. The best model we found using K-means is an equal-weighted model with 4.76% small cashouts, 4.76% disastrous cashouts and 4.76% camouflaged cashouts. The model gives us 9 clusters, and 3 clusters are cashout clusters.

| Cluster | % Fraud | % Cashout | # Transactions |
|---|---|---|---|
| 0 | 4.75% | 4.52% | 2147 |
| 1 | 1.28% | 0.87% | 2978 |
| 2 | 0.00% | 0.00% | 14 |
| 3 | 81.23% | 74.73% | 554 |
| 4 | 1.31% | 0.44% | 916 |
| 5 | 89.40% | 89.24% | 604 |
| 6 | 14.88% | 13.82% | 1889 |
| 7 | 0.00% | 0.00% | 115 |
| 8 | 3.77% | 2.83% | 212 |

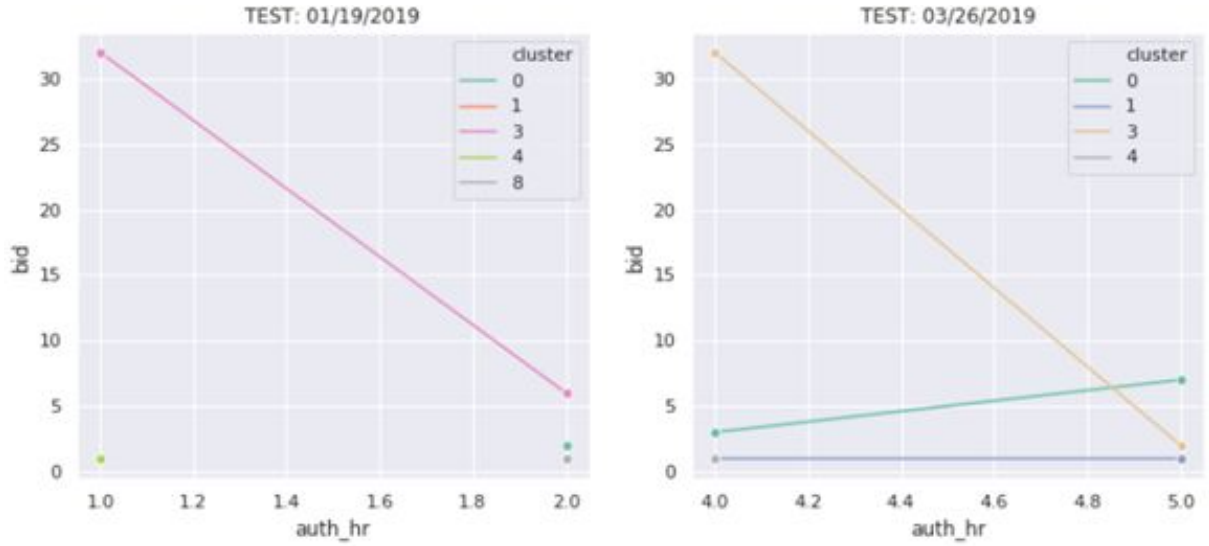
Our testing results are split up by type of cashout event. First, for disastrous cashouts, the graph below shows the number of cashout transactions captured by each cluster during the entire time span of a cashout event. Disastrous cashouts can be captured by cluster 5.




In the next section of images, we see that camouflaged cashouts are spread across all cashout clusters. For the cashout event in the middle, cashout clusters 3, 5&6 as a whole capture most of the transactions. However, the best individual cluster is 0, which is a non-cashout cluster. This is reasonable given the nature of camouflaged cashouts: they are difficult to detect because they behave like normal transactions.




Finally, we can see below that small cashout clusters are found in cluster 3 of the analysis, as this cluster shows elevated ratios on both the January 19th data and the March 26th data.




Next, we will discuss the recommended action to take as a result of the Kmeans analysis. Again, our project goal is to detect the bins in a bank that are hacked and the cashout locations, so that we can block the bins where the cashout happens without influencing other normal transactions. In order to do this, we need to look into cashout clusters and find the problematic BINs, cities and countries. We notice that the cashout clusters are not 100% pure - there are normal transactions mixed in them. To differentiate cashouts from normal transactions, in each cashout cluster, we calculate the number of transactions associated with each feature(BIN, CITY, COUNTRY) and divide the number by its one-hour baseline to get a ratio. We would run the k-means model on an hourly basis and block BINs in cities or countries with high ratios for the rest of the day once they are detected.

The table below shows a summary of different strategies and the true positives(cashouts that are captured), true negatives(cashouts that are not captured) and false positives(false alarm). The net gain or loss is calculated by TP-TN-0.1*FP. The action recommended by K-means is to block BINs in cluster 5, which will produce a net gain in $5.7m+ as a result of money not lost in fraud as well as transactions allowed over what Visa's baseline action allowed.

| Action | TP | TN | FP | Net Gain(Loss) |
|---|---|---|---|---|
| Block BINs in Cluster 3 | 6,887,405.29 | 10,764,441.33 | 144,269.43 | (3,891,462.983) |
| Block BINs in Cluster 5 | 11,684,031.40 | 5,970,844.90 | 60,274.00 | 5,707,159.100 |
| Block BINs in Cluster 6 | 11,698,117.76 | 5,965,131.22 | 380,552.26 | 5,694,931.314 |
| Block BINs in Cities in Cluster 5 | 6,218,697.02 | 11,435,629.96 | 27,570.60 | (5,219,690.000) |
| Block BINs in Countries in Cluster 5 | 9,230,621.32 | 8,424,091.81 | 41,025.43 | 802,426.967 |

## NetworkX

For semi-supervised modeling, we took the first month of data, and used it to create baselines for each entity to decide what amount of transactions and dollars moved per hour was normal at each level. To create this baseline, we took one month of data and excluded all transactions of response variable = 1 so that our baseline would contain as little noise as possible. Then we aggregated, averaged, and normalized the number of transactions, transaction amount, average riskiness score, and average predicted probability of response variable = 1 (both calculated by Visa). Subsequently, we calculated our own risk value using average normalized transaction amount, average normalized number of transactions, average normalized Visa-calculated risk score, and average normalized number of edges connecting to the entity (for example, number of ATMs which an account has interacted with or vice versa). Using this baseline, we would be able to determine if an entity in our system was exhibiting unusual behaviors.

The baseline calculation process took a long period of time to calculate since we had to go through a whole month of data i.e. millions of transactions. So to save time, we only calculated baselines for Bin, Bid, City, and Country. We did not calculate baselines for ATM or account since most ATM's and the majority of accounts are not used on a regular basis in cross border transactions, so the majority of ATM's and accounts we would have calculated baselines for, would have shown up rarely in other months in the data. This decision also had the outcome of reducing the time to merge the baseline data frame with the many hour data frames that would be in our train, validation, and test data sets.

To begin examining the unusual behaviors, we first selected the same training set of data as used in the Kmeans algorithm above, that is, a specific day which had a large cashout event which would allow our model to have many examples of our response variable being positive class. Our first step in modeling was to calculate all of the required fields for our specific training day. Next, we compared each node to its baseline and calculated the normalized risk value as a ratio of the current hour against the baseline. Thereafter, we looked at the distribution of normalized risk values and created a grid search algorithm that would try to classify hours of entities as cashout or not. The algorithm would test 10 different normalized risk value cutoffs, with a different 10 values for each distinct entity level in our data (Bid, Bin, Pan, ATM, City, Country), and prioritize F1 scores so that we could ensure the false positive rate would not balloon up as we prioritized weeding out instances of cashout = 1.

The gridsearch validation strategy was necessary because different entity levels in our data had vastly different numbers of transactions and number of edges connected and hence different scales of risk values. For example, a bin is connected to a fixed number of accounts and one bid, so there will be almost not variability in connected edges, but a bin is an aggregate of

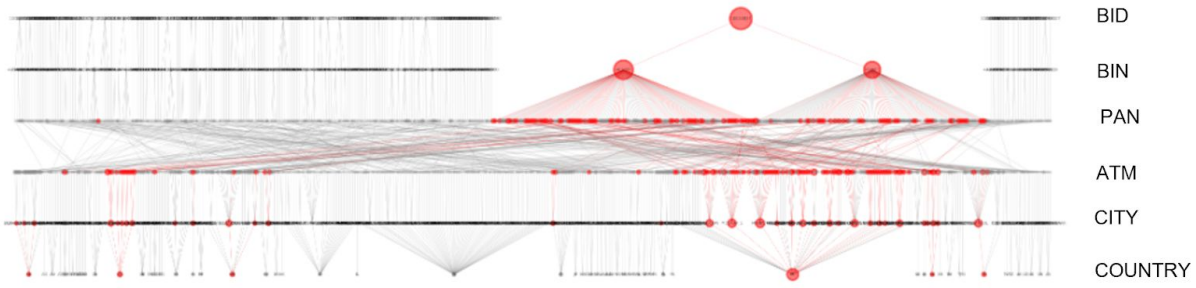
the transactions of many different accounts, so for a cashout event within that bin, the number of transactions and total amount of money flowing through may increase greatly.

After obtaining the most accurate risk value thresholds for each level based on f1 score, we moved on to validating our thresholds on another cashout event. Since we did not use any cross validation in grid searching for our thresholds, we believed that we might have overfit the thresholds to our training data. Because of this theory, we decided to use thresholds for all levels that were slightly above what the ideal thresholds were for the train set in the validation set. The results from the validation set showed that we had actually underfit the thresholds. After some thought, this result made sense. Our equation to calculate risk value is quite simple with only four metrics and baselines which makes our model hard to overfit but easy to underfit.

After tweaking the thresholds again to account for underfitting, we then used the last remaining major cashout event in our data to test our thresholds. Unlike the previous two major cashout events we used for train and validation sets, the test cashout event was much more spread out over time. The results show that our model was very accurate in predicting the cashout event at the Bid and Bin level with f1 scores of 97% and 99% respectively. Our model was less accurate with predicting at the City and Country level and had almost no predictive power at the ATM or account level.

In the training and validating process, we used camouflaged events. During our testing we used all three types of attacks and got similar results at each level. The only difference was that in small events, the model was not able to pick up the BID being attacked. We also tested blocking the at risk BINs in at risk cities with similar results. We were able to increase the recall slightly compared to the supervised model that was already inplace and we were able to greatly increase the precision.

Finally, we intend to use this analysis in conjunction with NetworkX to create visualizations of networks within the Visa system that have been highlighted as possibly being involved in a cashout event. Below is an example of the 1 hour of the Visa cross border network during a cashout event. The red nodes indicate that the risk value of the node is above its threshold and is likely being used in a cashout event. The business team can use this to block certain levels to stop the attack.



## Conclusions, Operational Execution, and Change Management

Ultimately we believe that our models will be able to provide value to Visa in a few different ways. Our models fit into the Visa system in two different ways. Firstly, our DBSCAN model can be fit on hourly engineered data as it enters the Visa system. Then, the DBSCAN system will produce clusters and a level-node profiling system which will alert Visa to possible large scale fraud occurring in their network. Next, our Kmeans model can have its centroids be pre-trained, allowing each transaction to be assigned to a cluster as it enters the system. This will allow kmeans to be run very quickly and efficiently as live data enters the system and means that, given Kmeans' effectiveness at identifying all types of cashout events, Visa can rely on our Kmeans model to provide accurate cashout detection in real-time. The two clustering methods will both provide profiles of at-risk Bins, Bids, Cities, etc. and allow the Visa Business team to take appropriate action in freezing transactions to specific combinations. Our Network model can also run in real-time. The NetworkX model creates baselines for each node in the Visa system, and thus can compare any new transaction to the baseline for that transaction's nodes. As the NetworkX model works, it will detect nodes which have elevated levels of activity and will send an alert. Additionally, the NetworkX model will map the region of the Visa network around this elevated-risk node and produce a visualization which can further help the Visa Business team to take appropriate actions in stopping the fraud before it gets too large. Our code was all written in Jupyter Notebooks on Visa's servers, so they will easily be able to find our notebooks and plug them into their existing real-time fraud detection systems. We recommend that Visa alter our code to make it run automatically as data flows in, and we believe by doing this, Visa can save millions of dollars during future cashout attacks. In conclusion, we believe that our unsupervised and semi-supervised learning methods, when applied to Visa's inflow of transaction data, will allow Visa to increase revenue by limiting false positives in fraud detection while also decreasing losses and increasing customer trust by maximizing the true positive rate, which will minimize monetary loss due to large scale cashout events.