STRIPE

Data Scientist Intern – Written Project

Table of Contents

PROBLEM 1 ..... ***2***

Project Overview **2**
- Problem Statement 2
- Problem Breakdown 2
- Problem Translating To Data Science 2
- High Level Project Flow 2
- Assumptions 2

Solution **2**
- Data Preparation 2
- Data Wrangling 3
- Data Analysis 3
- RFM Segmentation 4
- Train Clustering Models: K-Means, HDBSCAN 8

Project Overview **13**
- Problem Statement 13
- Definition for churn 13
- Problem Translating To Data Science 13
- High Level Project Flow 13

Solution **14**
- Identify Churn 14
- Resampling the data set 14
- Assumptions of Logistic Regression 15
- Train model: Logistic Regression 16
- Model evaluation 16
- Improve the model 16
- Predict which active user will churn in the future 17

PROBLEM 1

Project Overview

Problem Statement

- Stripe wants to categorize merchants using its platform and gain deep understanding in their payment activity. However, Stripe has limited data on merchants and their transactions. The only available data contains merchant id, time, and transaction amount.

- The goal of this project is to provide insights into users' payment activity and to categorize users based on those insights.

Problem Breakdown

- There are different kinds of business classification: i.e based on industry, size, contract terms, customer segmentation, etc
- The merchant column in the data contains only merchant id code, both integer and string and does not specify the name of the merchant or follow a specific pattern. Therefore, we can't use this variable to infer the type of the business
- The other two columns are time and transaction amount, which are useful to categorize merchants based on platform usage

Problem Translating To Data Science

The problem comes down to customer segmentation. Without a training set, this is accomplished via unsupervised learning of transaction data. Therefore, we can use clustering models to address it.

Solution: Using RFM Segmentation & Clustering Method

High Level Project Flow

1. Data Preparation
2. Data Wrangling
3. Data Analysis - RFM Segmentation (Recency, Frequency, Monetary)
4. Assign merchants with RFM score
5. Train Models: K-Means & HDBSCAN Clustering
6. Match clusters with segments made by RFM score

Assumptions

- Current time is 01/01/2035
- There is no fraud transaction
- All transactions have been processed successfully (no declined or refunded)
- All transactions have been processed only through Stripe (no imported by other payment systems/platforms)

Solution

Data Preparation: Load raw data

	Unnamed: 0	merchant	time	amount_usd_in_cents
0	1	faa029c6b0	2034-06-17 23:34:14	6349
1	2	ed7a7d91aa	2034-12-27 00:40:38	3854
2	3	5608f200cf	2034-04-30 01:29:42	789
3	4	15b1a0d61e	2034-09-16 01:06:23	4452
4	5	4770051790	2034-07-22 16:21:42	20203
...	...	...	...	...
1513714	1513715	72d37bedbf	2034-06-21 13:47:51	5274
1513715	1513716	5608f200cf	2034-04-20 02:23:59	754
1513716	1513717	fcbd1dae68	2033-09-19 14:02:33	13203
1513717	1513718	9843e52410	2034-12-28 20:07:59	4845

Data issues:

- Unnecessary column ‘Unnamed:0’
- 218 duplicate values
- 0 null value
- Lack month and date columns

Data Wrangling

- Drop the unnecessary column
- Clean 218 duplicate value
- Add month and date columns

	merchant	time	amount_usd_in_cents	month	date
0	faa029c6b0	2034-06-17 23:34:14	6349	2034-06	2034-06-17
1	ed7a7d91aa	2034-12-27 00:40:38	3854	2034-12	2034-12-27
2	5608f200cf	2034-04-30 01:29:42	789	2034-04	2034-04-30
3	15b1a0d61e	2034-09-16 01:06:23	4452	2034-09	2034-09-16
4	4770051790	2034-07-22 16:21:42	20203	2034-07	2034-07-22
...	...	...	...	...	...
1513714	72d37bedbf	2034-06-21 13:47:51	5274	2034-06	2034-06-21
1513715	5608f200cf	2034-04-20 02:23:59	754	2034-04	2034-04-20
1513716	fcbd1dae68	2033-09-19 14:02:33	13203	2033-09	2033-09-19
1513717	9843e52410	2034-12-28 20:07:59	4845	2034-12	2034-12-28
1513718	32acddd6cc	2034-08-23 09:07:07	3862	2034-08	2034-08-23

Data Analysis

As the data we have only contains merchant id, transaction amount and date transacted, RFM analysis perfectly capitalizes on all those data points to segment customers based on when their last purchase was, how often they’ve purchased in the past, and how much they have spent.

Benefits of RFM Analysis

Conducting an RFM analysis on customer base and sending personalized campaigns to high value targets results in favorable benefits
Personalization: By creating effective customer segments, Stripe can create relevant, personalized offers.
Improve Conversion Rates: Personalized offers will yield higher conversion rates because customers are engaging with products they care about.
Improve unit economics
Increase revenue and profits

RFM Segmentation

RFM analysis is a data driven customer behavior segmentation technique.
RFM stands for recency, frequency, and monetary value.

- Recency: days since last transaction
- Frequency: counts of all transactions
- Monetary: total money transacted

These metrics are indicators that we can utilize to segment merchants. For example:

- High recency, high frequency, high monetary: the best value customer
- Low recency, low frequency: churn or leavers

RFM Segmentation step-by-step

Step 1: Create recency, frequency, monetary columns based on the above method.
The final table is below

	merchant	frequency	monetary	month	last_transaction_date	recency
0	5608f200cf	25507	20907471	2034-04	2034-04-20	256 days
1	53b3fbeae2	12178	60733740	2034-07	2034-07-20	165 days
2	1ddaea9838	12042	41443545	2034-06	2034-06-05	210 days
3	89e2d29885	11969	18054500	2034-11	2034-11-17	45 days
4	654930c922	11191	150931806	2033-01	2033-01-12	719 days
...	...	...	...	...	...	...
14346	3839c43c25	1	93980	2033-01	2033-01-12	719 days
14347	5cdc7cd9f5	1	23906	2034-03	2034-03-15	292 days
14348	1718d01b43	1	48197	2033-01	2033-01-08	723 days
14349	4cf644502f	1	17536	2033-02	2033-02-15	685 days
14350	314ea3d710	1	2068	2034-08	2034-08-29	125 days

Step 2: Determine the optimal number of splits/scores for recency, frequency and monetary

This method is similar to the elbow plot in K-Means Clustering. We can spot that 4 is optimal.
So here I will use quartile to split the data. Now we will score each customer according to their position.

There are many ways to do this but here we will use quartiles (divide the data by 4 as the optimal split) as it is the easiest and most understandable way

To create RFM scores, quartile analysis can help assign scores based on relative performance.

Step 3: Assign number from 1 to 4 for recency, frequency, monetary score (r_score, f_score, m_score)
The method is taking the total number of customers divided by four. Each quartile will give a score, 1 through 4. Next, give each quartile a score reflective of the position.

- First Quartile: 1 (0.75-1). (Highest)
- Second Quartile: 2 (0.5-0.75) (Second Highest)
- Third Quartile: 3 (0.25-0.5) (Third Highest)
- Fourth Quartile: 4 (0 – 0.25) (Lowest)

r_percentile	r_score	f_percentile	f_score	m_percentile	m_score
0.39	3	1.00	1	0.99	1
0.54	2	1.00	1	1.00	1
0.47	3	1.00	1	1.00	1
0.82	1	1.00	1	0.98	1
0.00	4	1.00	1	1.00	1
...	...	...	...	...	...
0.00	4	0.06	4	0.41	3
0.34	3	0.06	4	0.18	4
0.00	4	0.06	4	0.30	3
0.01	4	0.06	4	0.14	4
0.62	2	0.06	4	0.01	4

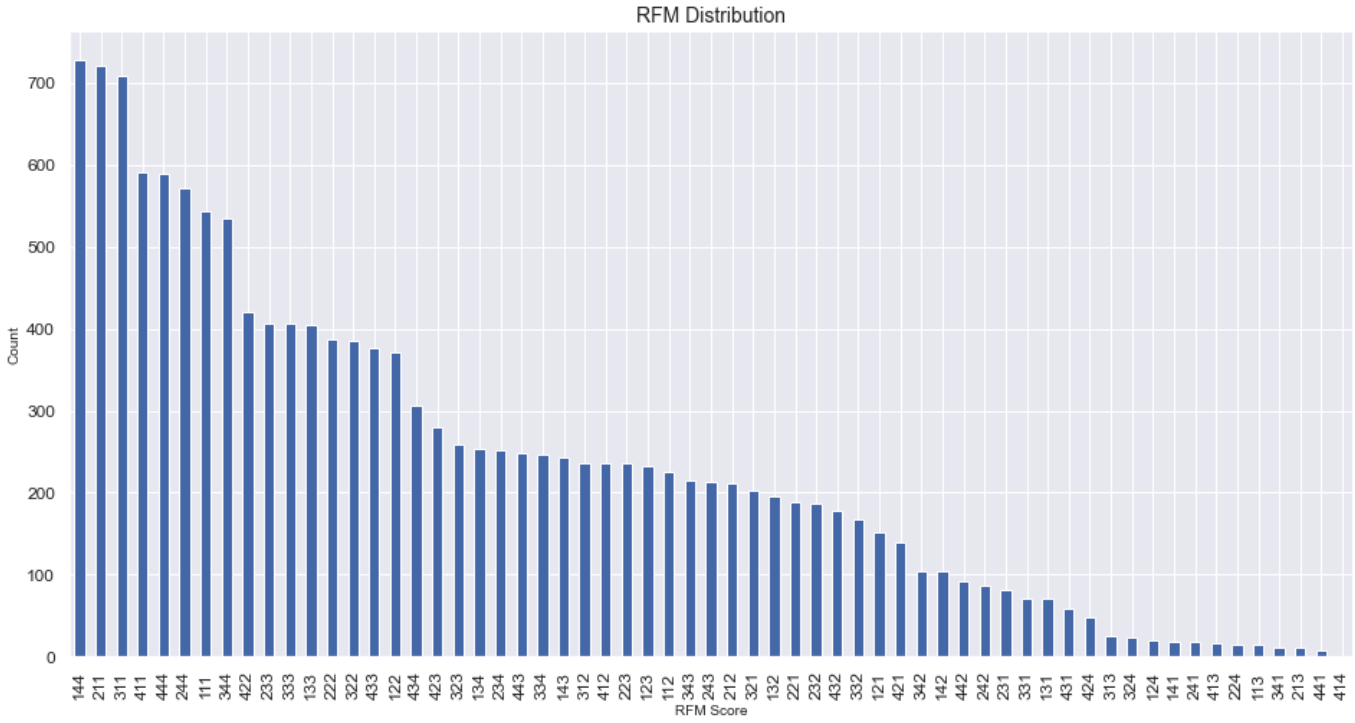
Step 4: Assign each merchant with RFM score

Last step, we will assign each merchant with accordant RFM score. RFM score is formed by concating recency, frequency and monetary score.

Now each merchant has been categorized by the rfm_score as the image below

r_percentile	r_score	f_percentile	f_score	m_percentile	m_score	rfm_score
0.39	3	1.00	1	0.99	1	311
0.54	2	1.00	1	1.00	1	211
0.47	3	1.00	1	1.00	1	311
0.82	1	1.00	1	0.98	1	111
0.00	4	1.00	1	1.00	1	411

Step 5: Visualize the segment distribution



Categories Interpretation

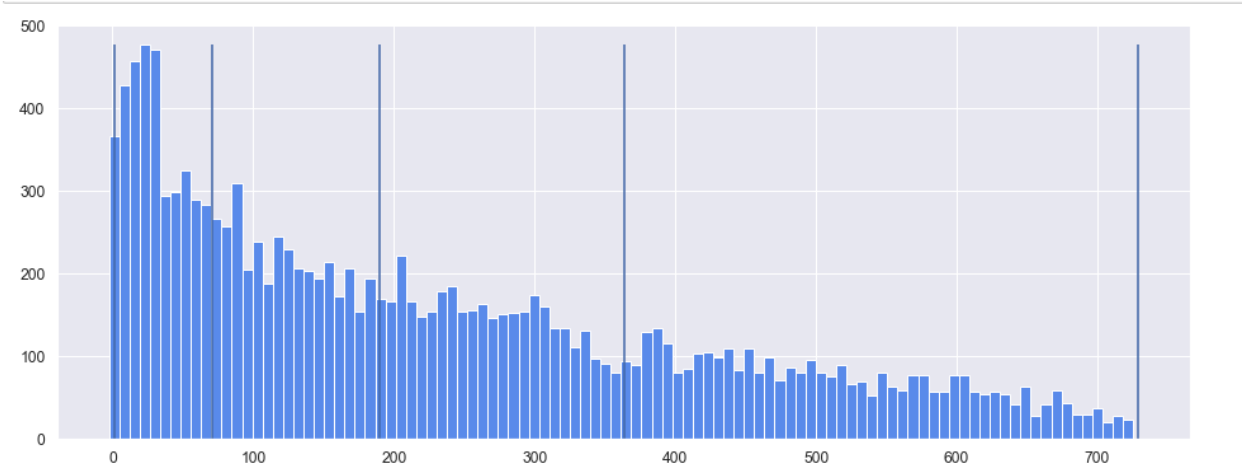
144 – recency:1, frequency:4, monetary: 4: merchants that made the last purchase most recent out 4 sectors, has lowest frequency and monetary level

Comments

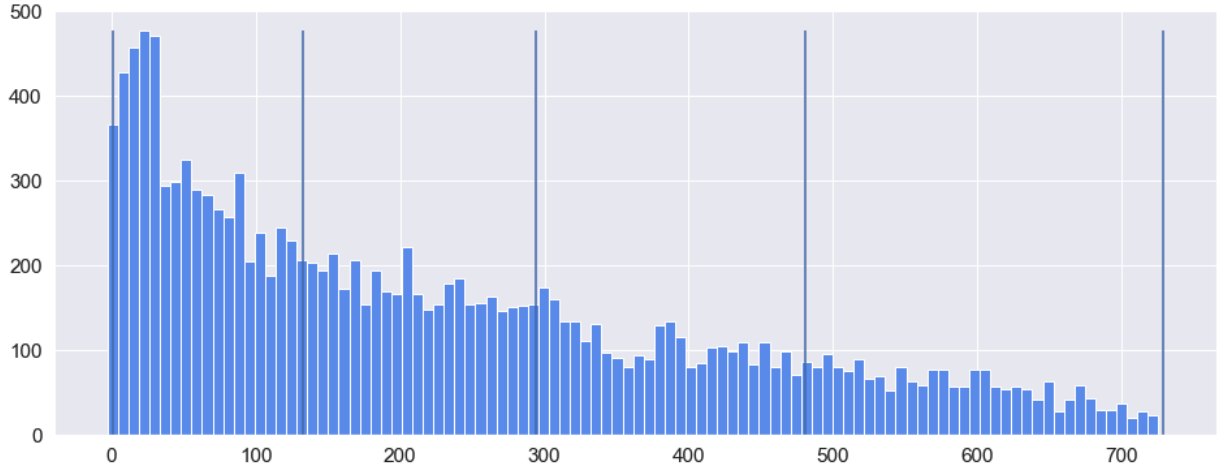
- The largest segment is one that has recently transacted within the last 3 months but their transaction amount and frequency are low. These could be new customers that are not familiar with the platform

- Our second, third and fourth largest segments are made up of our best valuable customers in terms of frequency and monetary. However, their last transaction was within 6 months to 2 years ago. This indicate that many high value customers are churn and might churn quickly. Therefore, management should have proper strategies to get them back.

Visualize the density of the segments by plotting a histogram with our partitions



The partitions split each part not equally.
We can try using Jenks Natural Breaks to partition our data instead of quartiles.

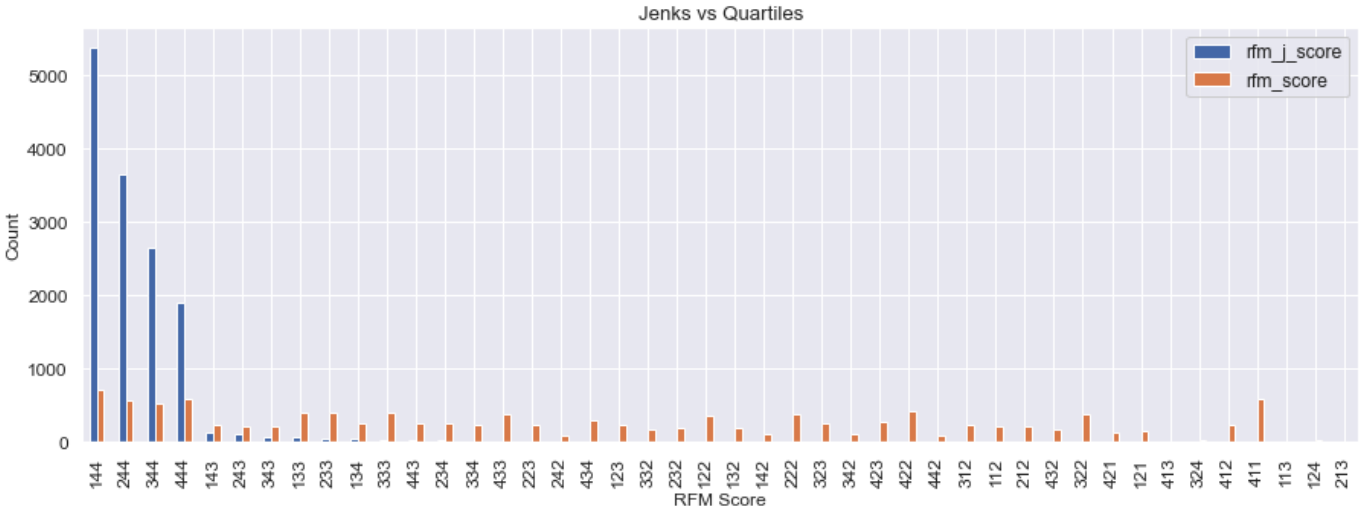


Compare partition tables side by side

	Q_Recency	J_Recency	Q_Frequency	J_Frequency	Q_Monetary	J_Monetary
min	1.00	1.00	1.00	1.00	201.00	201.00
0.25	71.00	133.00	3.00	715.00	36263.00	8019293.00
0.50	190.00	294.00	11.00	2859.00	160262.00	35923369.00
0.75	364.00	481.00	45.00	7172.00	823099.50	110302095.00
max	729.00	729.00	25507.00	25507.00	233045139.00	233045139.00

Find the rfm score based on Jenks Natural Breaks and see how the two scores compare

Compare distribution of Jenks vs Quartiles methods



- The result of Jenks rfm score is as same as Quartile as the largest segment that is made up of recent users but are not frequent and not spend much.
- The rest are very different. Jenks categorizes the next three largest segments are all customers with low frequency and monetary
- Even though the partitions of Jenks Natural Break method looks better than Quartile, it can't detect the segment with high monetary and frequency well. Therefore, we should move forward with just quartile RFM score

Train Clustering Models: K-Means, HDBSCAN

Inspect Data Quality Concerns

	frequency	monetary	recency
count	14351.00	14351.00	14351.00
mean	105.46	1632885.22	234.83
std	527.57	6419706.45	189.35
min	1.00	201.00	1.00
25%	3.00	36263.00	71.00
50%	11.00	160262.00	190.00
75%	45.00	823099.50	364.00
max	25507.00	233045139.00	729.00

We notice that features 'recency','frequency' and 'monetary' have different scales and have large variability, which is clearly shown by the mean and std of each feature.

Before training machine learning models, standardization is an important step of data preprocessing. It controls the variability of the dataset, it converts data into specific range using a linear transformation which generate good quality clusters and improve the accuracy

We standardize the features so that they are centered around 0 with a standard deviation of 1

In K-means clustering analyses, standardization is especially crucial in order to compare similarities between features based on certain distance measures.

Standardizing Features (Feature Scaling)

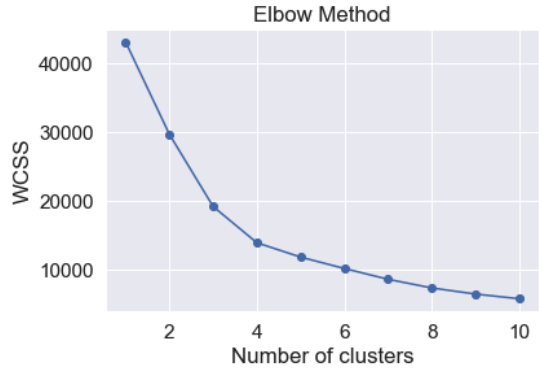
Use ‘preprocessing’ library and StandarScaler() to standardize features

	recency	frequency	monetary
0	0.11	48.15	3.00
1	-0.37	22.88	9.21
2	-0.13	22.63	6.20
3	-1.00	22.49	2.56
4	2.56	21.01	23.26
...	...	...	...
14346	2.56	-0.20	-0.24
14347	0.30	-0.20	-0.25
14348	2.58	-0.20	-0.25
14349	2.38	-0.20	-0.25

Train clustering models

K-Means Clustering

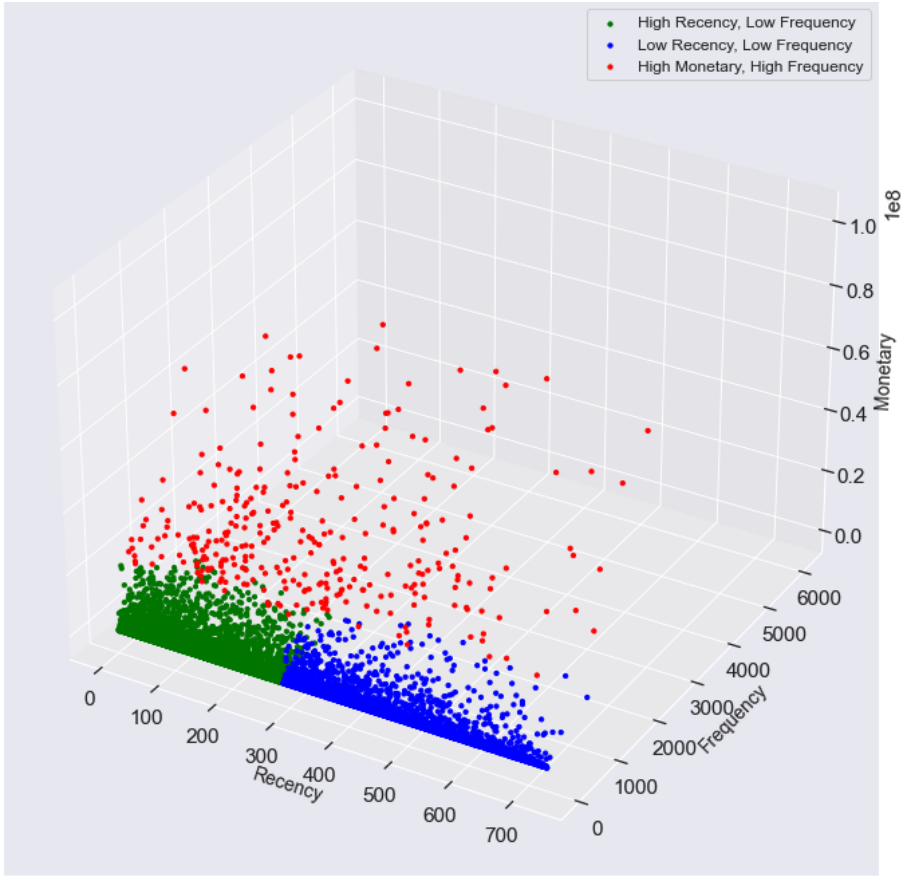
Step 1: Identify the optimal number of clusters using Elbow plot. The optimal number is 4.



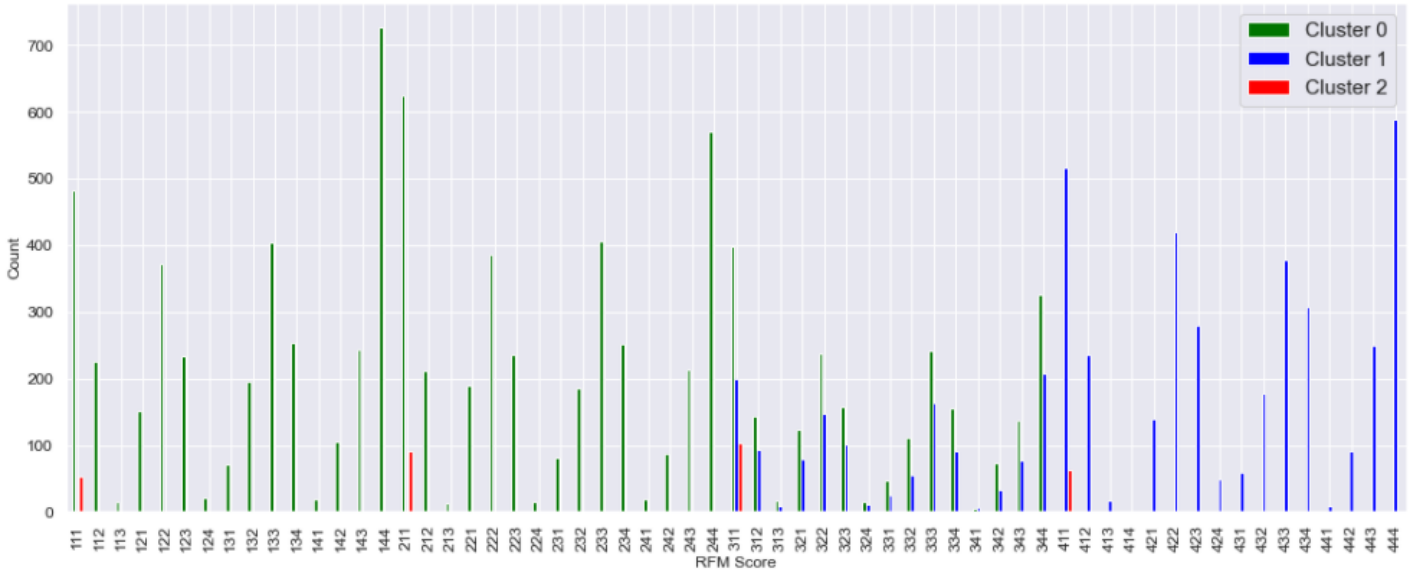
Step 2: Train the model using KMeans() function with n_cluster=4

f_score	m_percentile	m_score	rfm_score	K_Cluster
1	0.99	1	311	3
1	1.00	1	211	3
1	1.00	1	311	3
1	0.98	1	111	3

Step 3: Visualize the k-means clusters. The -1 cluster was excluded because it is just noise.



Step 4: Match clusters with RFM segments

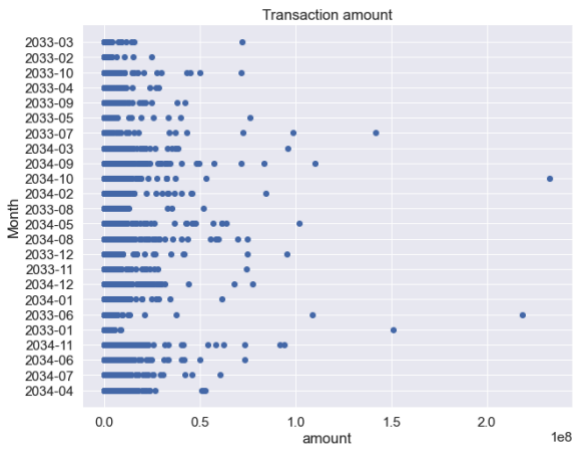


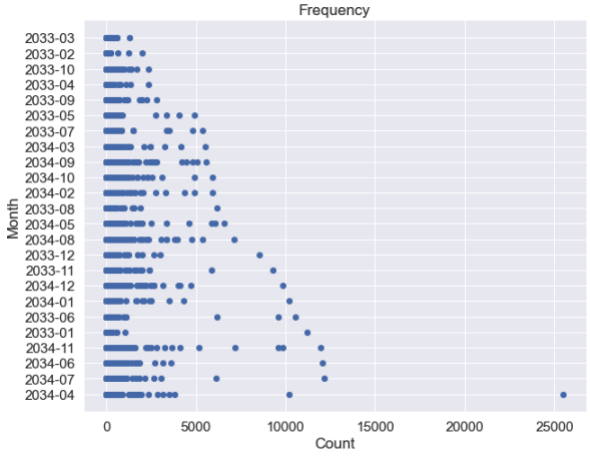
Comments

- The clusters are consistent with RFM segments
- Cons: The model can cluster recency, and frequency but does not capture the difference in monetary and frequency.
- Therefore, I have decided to try another clustering model that fits the data set better. Our data set has

Train Clustering Model: HDBSCAN

Step 1: Identify data properties and issues
Our merchant data contains many outliers (images below):





The data has large magnitude in variance among variables and may follow arbitrary shape.

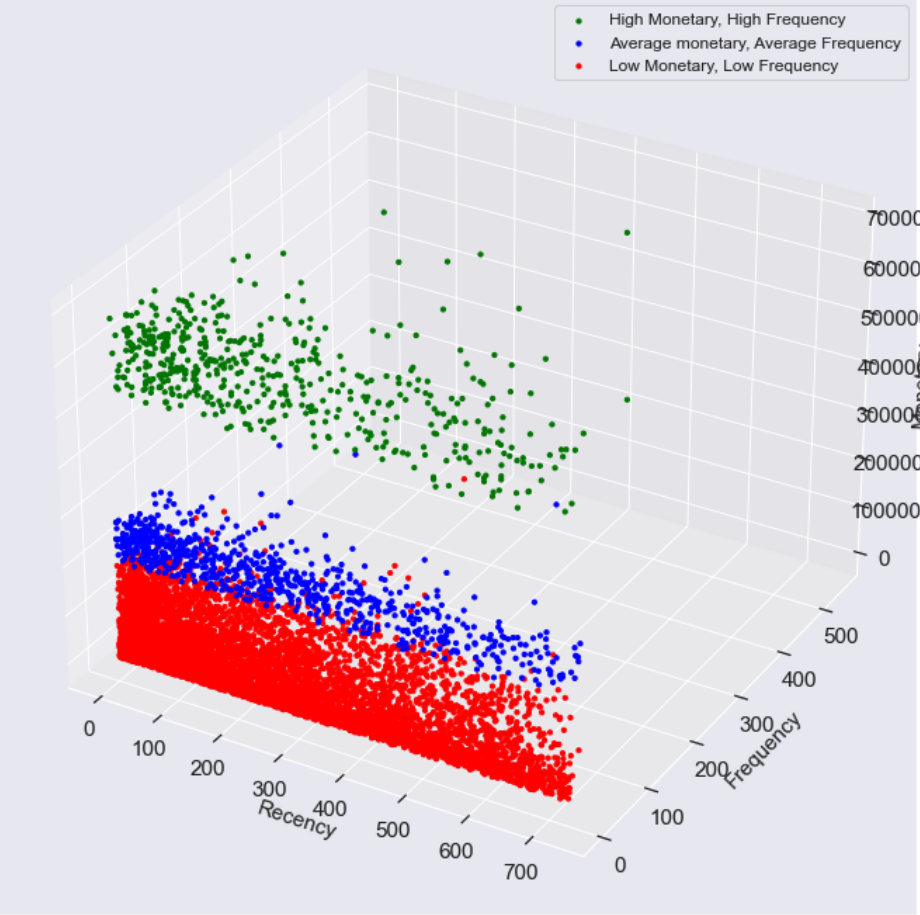
Step 2: Solution – using DBSCAN Clustering
The reasons we should use DBSCAN (density-based spatial clustering of applications with noise) for our data:

- DBSCAN clustering is also robust to outliers.

- K-Means and Hierarchical Clustering both fail in creating clusters of arbitrary shapes. They are not able to form clusters based on varying densities. DBSCAN clustering can!
- It groups ‘densely grouped’ data points into a single cluster. It can identify clusters in large spatial datasets by looking at the local density of the data points.

- HDBSCAN extends DBSCAN by converting it into a hierarchical clustering algorithm, and then using a technique to extract a flat clustering based in the stability of clusters.

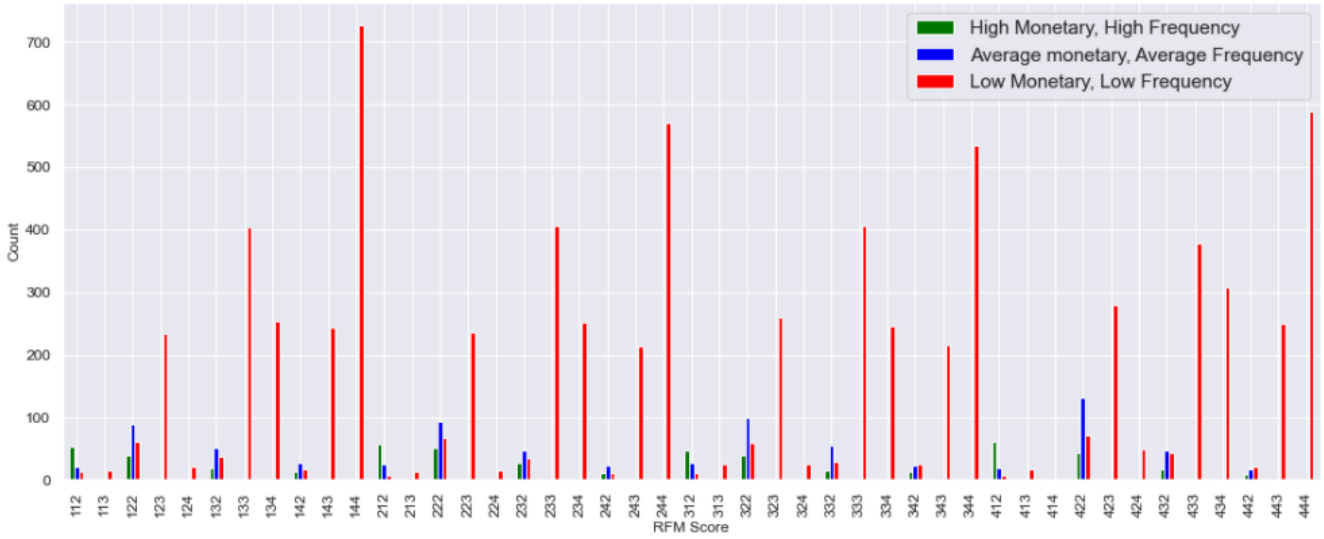
Step 3: Visualize the result of HDBSCAN Clustering model



Comments
Based on the graph, the clusters are spread over all recency levels so it can only classify different levels of monetary and frequency value.

- Cluster green: High monetary and frequency
- Cluster blue: Average monetary and frequency
- Cluster red: Low monetary and frequency

Step 4: Match the clusters above with RFM categories. The clusters and segments are consistent with each other. We also can notice that HDBSCAN clustering provides clearer results compared to the K-means we did earlier.



PROBLEM 2

Project Overview

Problem Statement

- The churn rate measures a company's loss in subscribers for a given period of time**.** Churn prevents growth of the company. Therefore, it is essential to identify and predict churn to reduce it in advance.

- The goals of this project:
  - Identify merchants that have already churned in the transaction data set of Stripe
  - Build a model to predict which active merchants are most likely to churn in the near future.

Definition for churn

- Churns are merchants that have recency more than 30 days[1]
- Churn is dummy variable, 1 is churn, 0 is active users

Problem Translating To Data Science

Churn prediction is typically treated it as a classification problem, classifying a customer as yes/no for churning. Therefore, we can use classification models in supervised learning to tackle this question.

High Level Project Flow

1. Identify churn
2. Resampling data

[1] https://support.stripe.com/questions/calculating-subscriber-churn-rate-in-billing

3. Train model: Logistic Regression
4. Improve the model using hyperparameter tuning
5. Predict churn for active user data

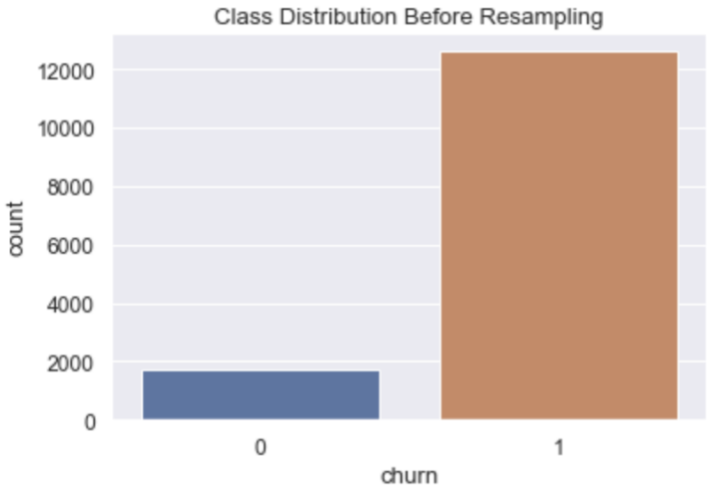
Solution

Identify Churn

Create churn column based on the definition mentioned above

merchant	frequency	monetary	month	last_transaction_date	recency	r_score	f_score	m_score	rfm_score	K_Cluster	HDB_Cluster	churn
637c3901b9	1	59068	2033-10	2033-10-24	434	4	4	3	443	1	2	1
e6ef86289b	1	901486	2034-12	2034-12-26	6	1	4	1	141	0	-1	0
86f216bd1a	1	109966	2033-07	2033-07-19	531	4	4	3	443	1	2	1
9162c065c3	1	513096	2034-11	2034-11-22	40	1	4	2	142	0	-1	1
dcb392770a	1	3398	2034-12	2034-12-14	18	1	4	4	144	0	2	0
3839c43c25	1	93980	2033-01	2033-01-12	719	4	4	3	443	1	2	1
5cdc7cd9f5	1	23906	2034-03	2034-03-15	292	3	4	4	344	1	2	1
1718d01b43	1	48197	2033-01	2033-01-08	723	4	4	3	443	1	2	1
4cf644502f	1	17536	2033-02	2033-02-15	685	4	4	4	444	1	2	1
314ea3d710	1	2068	2034-08	2034-08-29	125	2	4	4	244	0	2	1

Check the class distribution of churn

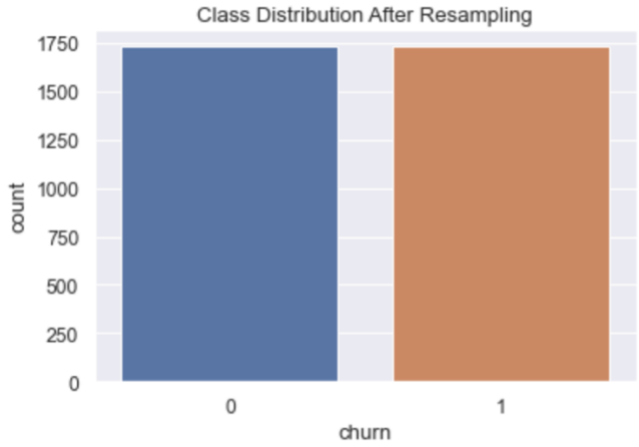


Resampling the data set

The class is significantly imbalanced. This is not ideal for machine learning modeling. Therefore, we would need to resample to rebalance the class.

Use up-sampling which means increasing the number of samples of the class with less samples by randomly selecting rows from it.

Combine positive and negative class and checking class distribution



Step 3 Compare the mean of RFM of the sample and the original data

	recency	frequency	monetary	recency	frequency	monetary
churn						
0	16.28	73.67	1135810.93	16.28	73.67	1135810.93
1	268.18	102.81	1740998.94	264.79	109.82	1701020.75

The mean of recency, frequency, monetary of these datasets are very similar Now we are ready to train the model

Model selection
There are many classification models that can be used to make churn prediction such as: Logistic Regression, Naïve Bayes, Random Forest.

Assumptions of Logistic Regression

- Binary logistic regression requires the dependent variable to be binary and ordinal logistic regression requires the dependent variable to be ordinal.
- The observations are independent of each other
- Logistic regression requires there to be little or no multicollinearity among the independent variables.
- Logistic regression assumes linearity of independent variables and log odds
- Requires a large sample size[2]

Logistic Regression is a simple starting point. It is easy to explain and implement. Behind many large companies' Machine Learning is just simple logistic regression.

Data Concerns
Logistic Regression is not sensitive to the magnitude of variables. Thus, standardization is not needed before fitting this kind of models.

The performance of logistic regression did not improve with data scaling. If there predictor variables with large ranges that do not affect the target variable, a regression algorithm will make the corresponding coefficients ai small so that they do not effect predictions so much.

[2] https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/assumptions-of-logistic-regression/

Therefore, the original merchant data is appropriate to train the model

Train model: Logistic Regression

- Assign x, y values
- Target variable y is ‘churn’
- All other variables except for ‘merchant’ (string) and ‘rfm_score’ are features x
- Split the data to train and test split, test size 80/20, random state 42
- Using scikit-learn package to import LogisticRegression function to fit the model
- Make prediction on the test set

Model evaluation

Using accuracy, precision and recall

When choosing from the many models out there to use to predict enterprise churn simple accuracy won’t show the whole picture. Enterprise churn should be under 25%, so simple accuracy isn’t the whole picture. A model that says “no one will churn” will be 75% accurate. So we need to focus on precision and recall.

- Precision is the % of all churn that the model correctly identifies
- Recall is the % of identified churn that actually ends up churning
- Accuracy is the % of all predicted values that the model correctly identifies

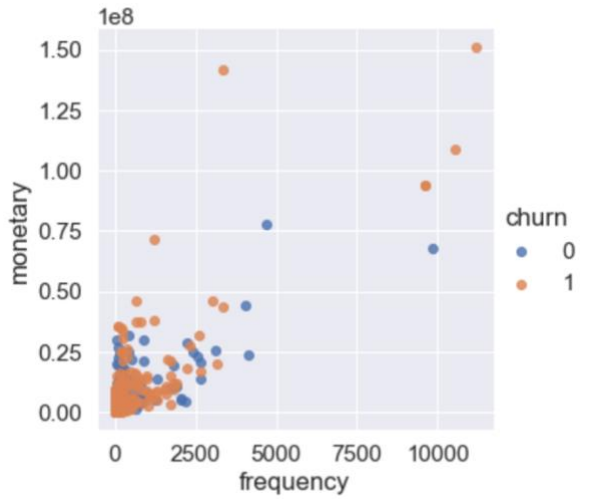
These evaluate false positives, which lead to unnecessary and costly churn prevention efforts, and false negatives, which lead to companies churning without being identified.

The Logistic Regression model provides the following results:

```
{'accuracy': 0.5158959537572254,
 'precision': 0.4915514592933948,
 'recall': 0.9876543209876543}
```

The accuracy and precision of the model are relatively low. We need to improve the model.
This could be conducted by many ways such as feature selection or hyperparameter tuning.

Improve the model



The graph shows that monetary and frequency are not ideal predictors of churn.
The churn variable is significantly biased because it was categorized using recency.

Since the churn variable is created solely based on recency, feature selection will choose recency as the best predictor. However, as the goal of the project is to predict which active user (recency <30) will likely churn in the future, this model will be unable to do the job.
Therefore, hyperparameter tuning would be more preferable.

Hyperparameter tuning
Using GridSearchCV to find the best parameters and the best accuracy score
Apply the best parameters to the new logistic regression model
Calculate model valuation metrics

```
Accuracy: 0.9754335260115607
Precision: 0.9967637540453075
Recall: 0.9506172839506173
```

As the metrics are optimized, prediction power is increased.

Predict which active user will churn in the future

- Active user set contains merchant that has churn = 0
- After making prediction on the active user set, the following result was obtained
- There are 8 active merchants listed below are predicted to churn in the future by our logistic regression model
- The ‘churn_pred’ column is the prediction result

merchant	frequency	monetary	month	last_transaction_date	recency	r_score	f_score	m_score	rfm_score	K_Cluster	HDB_Cluster	churn	churn_pred
f1dac61590	167	890080	2034-12	2034-12-02	30	1	1	1	111	0	-1	0	1
aedcc213a5	141	1782945	2034-12	2034-12-02	30	1	1	1	111	0	-1	0	1
487bb338df	128	1897774	2034-12	2034-12-02	30	1	1	1	111	0	-1	0	1
0c43e8ca9c	121	1134116	2034-12	2034-12-02	30	1	1	1	111	0	-1	0	1
8b15da3ecd	88	1774166	2034-12	2034-12-02	30	1	1	1	111	0	-1	0	1
b1220d7962	75	879892	2034-12	2034-12-04	28	1	1	1	111	0	-1	0	1
61b5030d9b	50	260651	2034-12	2034-12-02	30	1	1	2	112	0	1	0	1
3c781e3803	47	1179895	2034-12	2034-12-03	29	1	1	1	111	0	-1	0	1