Data Science CA2 – Daroogheh

Overview

In this project, we implement a real time data pipeline to process and store data as it gets to the server. We can divide this project into 4 phases. First, we need to get the historical data and store them properly. Then we need to perform some basic analysis on them to gain some general insight about our domain. Next, we need to get the real time data and perform various analysis on them including fraud detection. Lastly, we need to update our database with the new data and then analyze them.

For the first part, we will use spark batch sessions to get the data from the synthetic transaction generator. We then use MongoDB as our database and store the historical data. In the second part we use the spark batch sessions to perform various analysis on the data. In the third section, we use spark stream sessions to get the real time data from the synthetic transaction generator and mark the fraudulent transactions. We then update our database with the incoming data. In all of the stages, we use Kafka in order to mimic a real word scenario in which data comes to our server from different sources and we have to process it.

Store the Historical Data

In this section we Store the historical data. The historical data is generated by a synthetic transaction generator. It consists of 20,000 different transactions. We have to store these transactions in our database. For this project, we use MongoDB which is a NoSQL database which means the underlying structure of the database is not tables but collections instead.

We designed a database with 6 different collections. The first collection is transaction summary which stores all of the information about the transactions. Each datapoint here represents one transaction with the transaction id being the key. The second collection is the daily collection where we group the transactions by the date. This table could be used to do temporal analysis and find trends in the data. The third one is the merchant data. This table stores the information about each merchant with the merchant id being the key. We have a similar collection for the customers as well. we store basic data like the transaction count, transaction value, total commission value, successful transactions and failed ones and … in this collection. The last two collections store the information of the customer types and merchant categories. These two collections come really handy when performing future analysis.

To store the data in our database, we need to perform various aggregations so we use the spark batch sessions to make the processing faster. We first read the data from Kafka and save them in a pandas dataframe. Then we open the dataframe and convert it into a spark dataframe. After that, we start aggregating the data and store it into our collections. After this step our database is filled with the historical data.

Historical Data Analysis

Now that we have our historical data saved into the database, we can move on with analyzing this data. After some trials we understood that loading the data again from the database could be time-consuming and unnecessary as we already have the data we want as a spark dataframe ready to be analyzed. So, this part is somehow integrated with the previous part. Right after saving the data, in the same session, we start analyzing the data. We did many different and interesting analysis. However, as the synthetic transaction generator is not really sophisticated and generates the transaction information from a uniform random distribution the reports currently are nor really meaningful. We hope as we replace the synthetic generator with the real-world data our reports will be more meaningful, pinpointing interesting facts about the business.

1. **Commission Type**
   The first thing we analyzed was the commission strategies as they play a key role in our business. Finding the optimal strategy can significantly boost our income. We grouped the transactions by the commission type and then calculated the commission to transaction value ration. The higher this ratio the higher our income will be for the same amount of transaction value. We can see the output below. However, as stated before, this report is currently meaning less as the ratio for all different strategies are roughly the same.

commission_type	total_commissions	avg_commission_per_trx	total_trx_amount	commission_to_trx_ratio
flat	136829115	20678.421490101253	7457361035	0.01834819507300413
progressive	137414931	20394.02359750668	7489291468	0.018348188421714127
tiered	136072908	20489.822014756814	7416145409	0.018348198490669588

2. **Customer Summary**
   The next step is to identify the key customers. Knowing our customers' behavior enables us to make better business decisions. Therefore, knowing our top customers play a huge role in moving our business forward. Below we can see the basic information about each of our customers.

customer_id	total_trxs	total_trx_value	total_commissions	avg_commission	avg_trx_value	approved_trxs	declined_trxs
cust_464	37	42302822	776180	20977.837837837837	1143319.5135135136	36	1
cust_458	32	41126495	754600	23581.25	1285202.96875	32	0
cust_174	30	39863881	731432	24381.066666666666	1328796.0333333334	25	5
cust_129	31	38654540	709240	22878.709677419356	1246920.6451612904	29	2
cust_167	29	37694914	691634	23849.44827586207	1299824.6206896552	28	1
cust_139	29	37506653	688180	23730.344827586207	1293332.8620689656	28	1
cust_582	29	37073046	680222	23455.931034482757	1278380.896551724	28	1
cust_394	26	37046952	679749	26144.19230769231	1424882.7692307692	25	1
cust_23	30	36976438	678451	22615.033333333333	1232547.9333333333	29	1
cust_375	33	36710767	673578	20411.454545454544	1112447.4848484849	32	1
cust_320	30	35844052	657673	21922.433333333334	1194801.7333333334	29	1
cust_297	30	35792721	656731	21891.033333333333	1193090.7	28	2
cust_955	33	35639594	653923	19815.848484848484	1079987.696969697	33	0
cust_463	32	35604714	653284	20415.125	1112647.3125	31	1
cust_691	28	35465111	650724	23240.14285714286	1266611.107142857	27	1
cust_296	29	35246525	646712	22300.41379310345	1215397.4137931035	29	0
cust_883	25	35093808	643913	25756.52	1403752.32	24	1
cust_326	25	35072679	643523	25740.92	1402907.16	22	3
cust_323	25	34843498	639319	25572.76	1393739.92	25	0
cust_974	30	34469568	632453	21081.766666666666	1148985.6	29	1

only showing top 20 rows

3. Merchant Summary
   Just like the consumers, knowing the merchants is very important to us as we can tailor our business plan to best fit their needs. Below we can see the basic information about each of the merchants.

merchant_id	total_trxs	total_trx_value	total_commissions	avg_commission	avg_trx_value	approved_trxs	declined_trxs
merch_22	436	494802594	9078739	20822.795871559632	1134868.3348623854	418	18
merch_1	419	494604256	9075106	21658.964200477327	1180439.7517899761	397	22
merch_30	423	486932865	8934353	21121.401891252954	1151141.524822695	393	30
merch_36	415	475993766	8733626	21044.881927710845	1146972.9301204819	389	26
merch_37	411	475766068	8729455	21239.5498783455	1157581.6739659368	394	17
merch_46	400	473256825	8683422	21708.555	1183142.0625	382	18
merch_47	411	472480213	8669166	21092.861313868612	1149586.8929440388	392	19
merch_38	423	470860589	8639445	20424.21985815603	1113145.6004728132	409	14
merch_29	412	468129288	8589325	20847.876213592233	1136236.1359223302	391	21
merch_31	439	466525322	8559884	19498.59681093394	1062700.0501138952	410	29
merch_42	420	465086520	8533494	20317.842857142856	1107348.857142857	402	18
merch_28	403	464470744	8522197	21146.89081885856	1152532.8635235731	380	23
merch_40	407	462678900	8489317	20858.27272727272 8	1136803.1941031942	385	22
merch_33	430	460913610	8456932	19667.283720930234	1071892.1162790698	413	17
merch_41	415	459448316	8430047	20313.366265060242	1107104.3759036146	381	34
merch_15	400	457889033	8401445	21003.6125	1144722.5825	378	22
merch_16	396	457465313	8393665	21196.12373737374	1155215.436868687	378	18
merch_2	413	457397832	8392420	20320.629539951573	1107500.803874092	388	25
merch_9	392	456553461	8376933	21369.727040816328	1164677.1964285714	373	19
merch_14	387	456009635	8366962	21620.05684754522	1178319.4702842378	367	20

only showing top 20 rows

4. Hourly Summary
   Now, we can focus on some temporal analysis. First, we can check when our servers are the most crowded and when the load is lighter. This will enable us to better plan our militance phases when some sections of the server might not be available and therefore lose less money.

hour_of_day	num_transactions	total_transaction_amount
12	824	938592882
22	787	882232904
1	861	959213831
13	831	940604804
16	840	955273504
6	849	951920933
3	838	919101777
20	810	909056541
5	755	837626217
19	822	914006937
15	809	919284589
9	849	959477330
17	831	926130424
4	861	956127429
8	848	922631088
23	855	971740346
7	863	917146397
10	845	918961386
21	858	971689538
11	913	1030384215

only showing top 20 rows

5. Time of the Day Summary
   Another table which could be useful for less formal meetings and reports is the table in which we show the number of transactions in each time of the day. For example, knowing that there are more transactions near noon compared to early morning could be insightful.

time_of_day	num_transactions	total_transaction_amount
Evening	4958	5555525109
Morning	5167	5700521349
Afternoon	4941	5615177197
Night	4930	5491574257

6. Day of the Week Summary
   The final temporal analysis on transaction values which could further help us determine the best maintenance time is to understand which days of the week the value of transactions is lower.

day_of_week	num_transactions	total_transaction_amount
1	2905	3317245621
6	2852	3202789018
3	2800	3131368603
5	2865	3121534197
4	2910	3241533079
7	2853	3223485792
2	2811	3124841602

7. Risk assessment Summary
   The final report that could benefit us is related to risk management. The better we could identify transactions with high risk and high probability of being declined the better we can mange them. Understand when these types of transactions are more likely to happen will let us prepare for them.

hour	approved	declined	approval_rate
0	763	38	95.25593008739077
1	821	40	95.35423925667828
2	766	48	94.1031941031941
3	798	40	95.22673031026252
4	814	47	94.54123112659698
5	713	42	94.43708609271523
6	804	45	94.69964664310953
7	817	46	94.66975666280418
8	801	47	94.45754716981132
9	812	37	95.64193168433451
10	806	39	95.38461538461539
11	861	52	94.30449069003286
12	787	37	95.50970873786407
13	774	57	93.14079422382672
14	758	48	94.04466501240695
15	770	39	95.17923362175526
16	798	42	95.0
17	794	37	95.54753309265945
18	769	57	93.09927360774817
19	782	40	95.1338199513382

only showing top 20 rows

hour	avg_risk_level
0	2.0886392009987516
1	2.0662020905923346
2	1.9987714987714988
3	2.10381861575179
4	2.073170731707317
5	2.013245033112583
6	1.994110718492344
7	2.0938586326767092
8	2.1108490566037736
9	2.0836277974087163
10	2.0284023668639053
11	2.085432639649507
12	2.0934466019417477
13	1.9951865222623346
14	2.0719602977667493
15	2.042027194066749
16	1.980952380952381
17	2.02647412755716
18	2.0617433414043584
19	2.0547445255474455

only showing top 20 rows

Real-Time Data Processing and Fraud Detection

Now that we studied and store the historical data, we can proceed to get the real-time data and store it. We use spark stream sessions for this task which enables us to create minibatches from our data and process each batch on a regular basis. This is essential for processing real-time data effectively. After getting the data, we first summarize it and print some basic reports from it. Then filter the fraudulent transactions and output them to a separate Kafka topic. Lastly, we update the database with the new data.

I. Basic Reports

In this section we create very simple reports just to get a sense of the incoming data. Below we can see the results of some of these reports.

window	commission_type	total_commission
{2025-05-02 18:31...	flat	1675527
{2025-05-02 18:31...	tiered	130898
{2025-05-02 18:31...	progressive	120664

```
-------------------------------------------
Batch: 1
-------------------------------------------
+--------------------+-----------------+--------------------+
|              window|merchant_category|    commission_ratio|
+--------------------+-----------------+--------------------+
|{2025-05-02 18:31...|     food_service|0.018348217957761398|
|{2025-05-02 18:31...|    entertainment|0.018348600729456016|
|{2025-05-02 18:31...|           retail|0.018348500463693504|
|{2025-05-02 18:31...|   transportation| 0.01834821607157938|
|{2025-05-02 18:31...|       government|0.018347803593939074|
+--------------------+-----------------+--------------------+
```

```
-------------------------------------------
Batch: 2
-------------------------------------------
+--------------------+-----------+----------------+
|              window|merchant_id|total_commission|
+--------------------+-----------+----------------+
|{2025-05-02 18:32...|    merch_5|         1621023|
|{2025-05-02 18:32...|   merch_14|          847016|
|{2025-05-02 18:32...|   merch_31|          845007|
|{2025-05-02 18:32...|   merch_36|          844881|
|{2025-05-02 18:32...|   merch_29|          810409|
|{2025-05-02 18:32...|   merch_17|          809836|
|{2025-05-02 18:32...|   merch_33|          809811|
|{2025-05-02 18:32...|   merch_49|          803512|
|{2025-05-02 18:32...|   merch_39|          800000|
|{2025-05-02 18:32...|   merch_37|          800000|
+--------------------+-----------+----------------+
```

II. Filter Fraudulent transactions

The most important step in the real-time data analysis is finding and filtering fraudulent transactions. There are different methods to filter such transactions. One could even use machine learning methods to detect them. In this project, we are going to follow a rule-based method. We have three rules that classify one transaction as fraudulent.

1. **Velocity Check**
   If a customer makes more than 5 transactions per 2 minutes, we classify them as potentially fraudulent. To check this, we use a 2-minute window with a 2-minute watermark as well. If we detect such a case, we print the customer id and all the transaction ids that contributed to the velocity error. With the original synthetic generator, no velocity error occurred so just to check the program we limited the number of costumers to 50 while keeping the generation rate the same. Below is a sample output of this report.

customer_id	transaction_ids	window_start	window_end	transaction_count
cust_10	[23d080d4-647c-44...	2025-05-02 09:09:00	2025-05-02 09:11:00	21
cust_5	[9dc547a7-9988-49...	2025-05-02 09:09:00	2025-05-02 09:11:00	24
cust_1	[1d3271c9-bb97-4e...	2025-05-02 09:09:00	2025-05-02 09:11:00	17
cust_9	[5fd9660e-e7d5-4f...	2025-05-02 09:09:00	2025-05-02 09:11:00	24
cust_3	[e4e6e7b5-e69a-46...	2025-05-02 09:09:00	2025-05-02 09:11:00	13
cust_8	[f72af06f-e8ce-40...	2025-05-02 09:09:00	2025-05-02 09:11:00	17
cust_7	[a204b7e2-c204-49...	2025-05-02 09:09:00	2025-05-02 09:11:00	27
cust_6	[844e5d6a-0177-4b...	2025-05-02 09:09:00	2025-05-02 09:11:00	34
cust_2	[c0fb8968-79a3-4c...	2025-05-02 09:09:00	2025-05-02 09:11:00	22
cust_4	[3fd5e54f-cf62-4f...	2025-05-02 09:09:00	2025-05-02 09:11:00	29

2. **Amount Anomaly**

   Another rule we have for determining fraudulent activity is comparing the transaction amount with the average transaction of the customer. To get this information, we access our database and join the customer summary topic with the Realtime data. This way, we have access to the average transaction value of the customer initializing each transaction. Again, in this section the original synthetic generator did not produce any errors so we tweaked it manually to test the program. Below we can see a sample report.

Batch: 1

tx_id	fraud_type	customer_avg_trx_value	transaction_amount
9e5048a7-2fc5-4fe...	Amount anomaly	907472.1666666666	43600000
39c126bc-7aa6-4fd...	Amount anomaly	943418.5454545454	43600000

3. Geographical impossibility
   The last rule for filtering fraudulent transactions is the geographical impossibility. We know that a human cannot move more than 50Km in 5 minutes therefore getting two transactions in 5 minutes from the same customer with over 50Km distance is fishy. We determine the distance using the longitude and the latitude. In this part the synthetic generator didn’t produce any errors and we could not change it in a way that it does so.

III. Update the Database

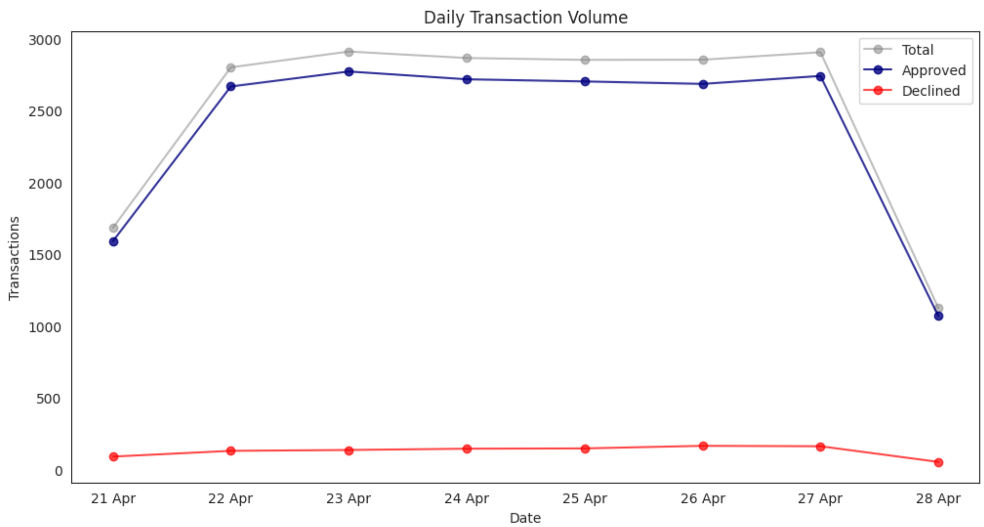
In this section we add all of the new transactions to the database. As we had different collections in our dataset, we should first aggerate the new data and then merge it with the current data. This process is not very straight forward as updating some values of the collections can be challenging. For example, in order to update the average transaction value for each customer we cannot simply take the average of the current and new value but instead we should first calculate the new total transaction value and the total transaction count and then divide them to get the final result. In order to make sure everything is working as expected we created a simple test program for the database in which we print the last data for each collection. By comparing the results of different executions, we can make sure that the database is updated correctly

Final Data Analysis and Visualizations

Now that we have our pipeline and the updated database ready, we can proceed to analyze the data and show our collected insight through visualizations. In this section, we first get the data we want from the updated database and then create various visualizations. The goal of these plots is to help us understand our business better. We focus on three aspects of our business. First, we focus on our customers and how we categorize them. Then we focus on our merchants. And lastly, we focus on abnormal and fraudulent behaviors.

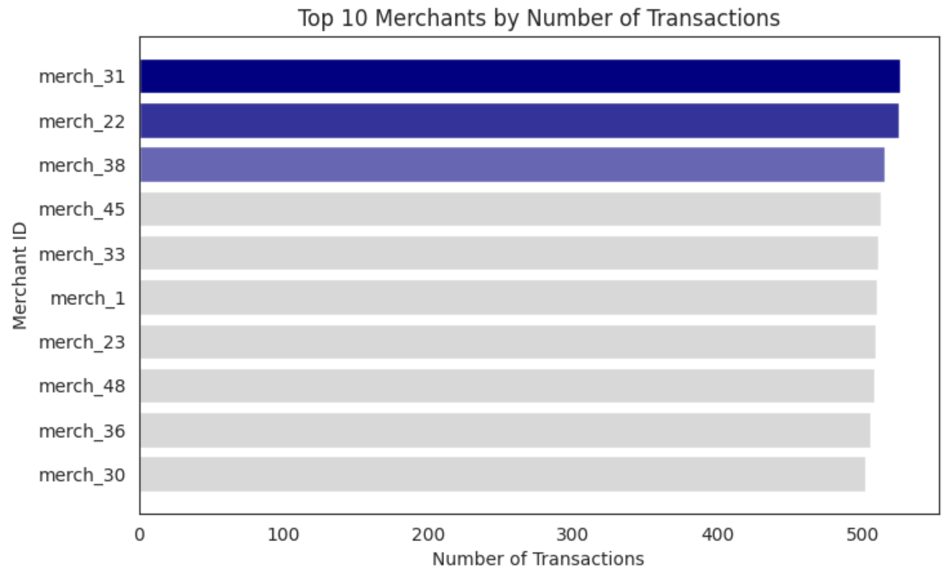
1. Daily Volume

We start our analysis with a big picture of the data. We plot the trend of total, approved and decline number of transactions. We can see that in this example data which was generated by the synthetic generator the transaction numbers remain roughly the same for each day. The only datapoint which is different is April 28th which is the current day and as it has not finished yet the number transactions so far is lower than the previous days.



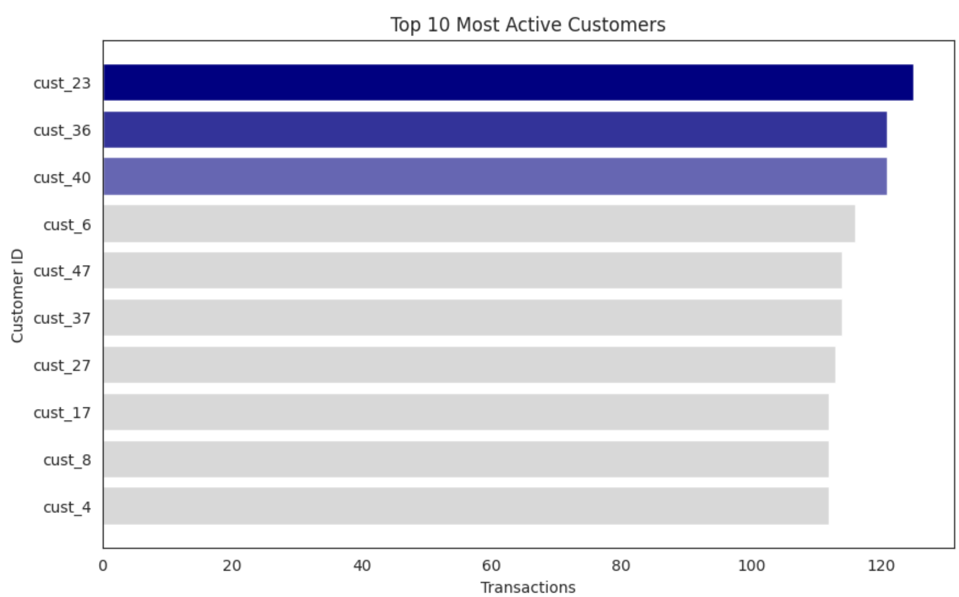
2. Top Merchants

Now, we shift our focus to the merchants. We want to find the top merchants by the number of transactions. These are the merchants who are really important to our business. Currently the difference between the merchants is negligible. However, we believe by replacing the synthetic data with real-word data this plot will be much more insightful.



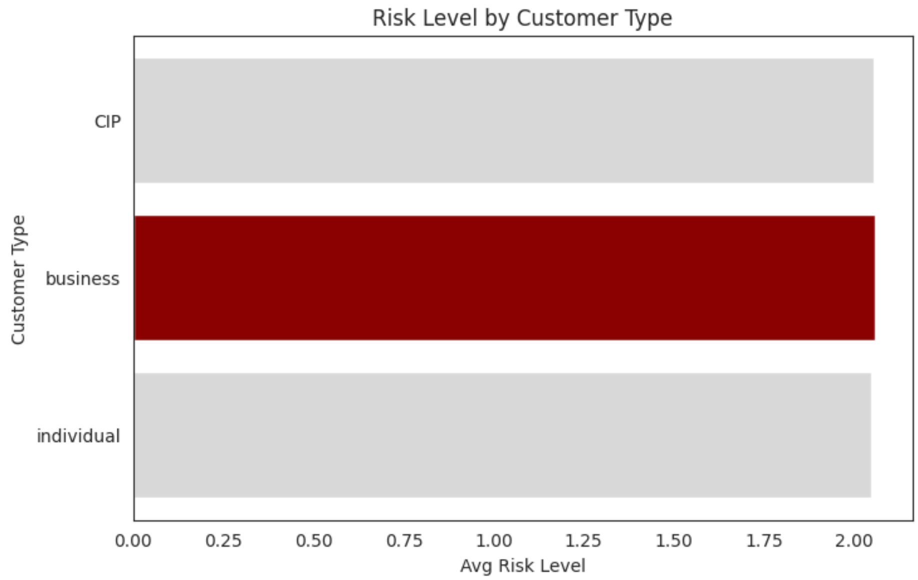
3. Top Customers

Just like what we did with the merchants, in this plot we rank our customers by the number of transactions. Again, we cannot really infer anything meaningful from this plot because of the synthetic generator. But in general, by identifying our most active customers, we can analyze their specific behavior and figure out why they use our business and how we can make the most profit from them.



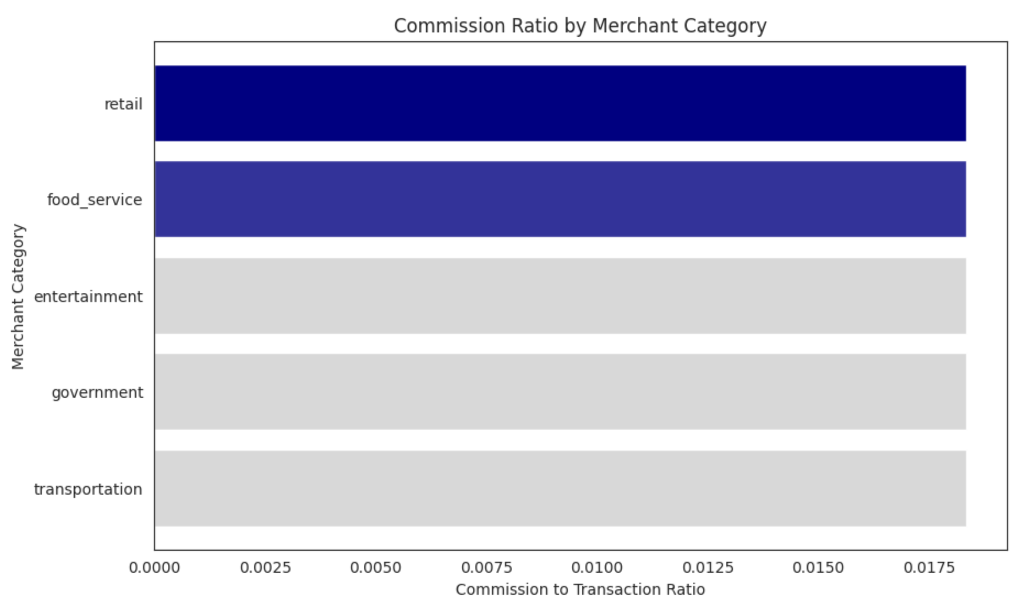
4. Most Risky Customer Category

Identifying our most risky costumers is for sure as important as finding our most active customers. These customers can be very harmful for our business. If we do not manage their behavior accordingly, they can leave us with huge losses. In this plot we plot the average risk level of each of our costumer categories. Currently we can see that all of them have the same average risk. However, in real world this plot would give us insight on which category is the riskiest one helping us to better plan to minimize our risk.



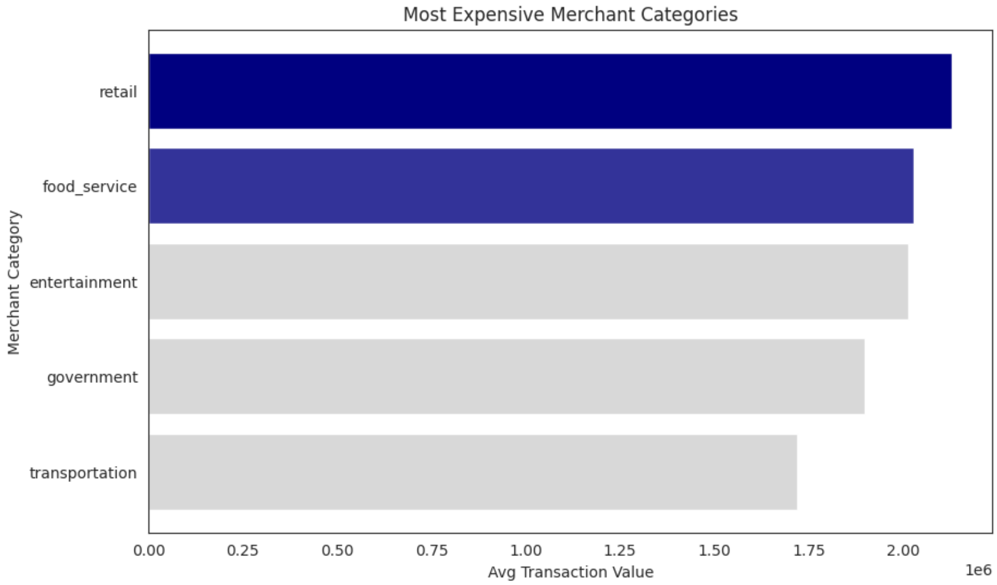
5. Merchants with Highest Commission to Transaction Ration

If we wanted to advertise what would be the best merchant category to focus for? This plot is answering this question. These merchants have the highest commission to transaction ration. So, if we increased the transaction volume of these category, they will have the highest increase in commissions. So, our goal with these merchants is to increase their transaction volume.



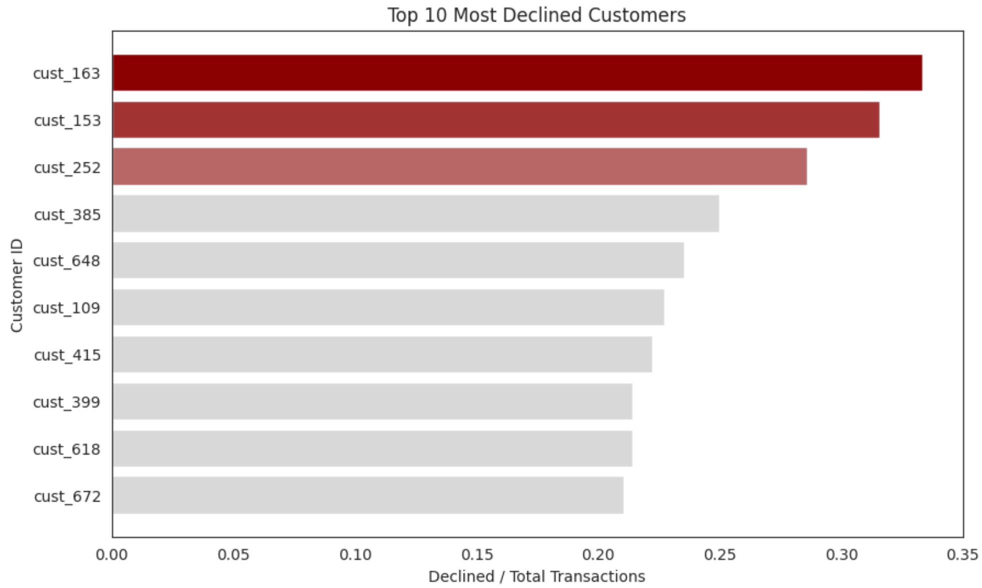
6. **Most Expensive Merchant Category**

In this section we have our merchants ranked by the average transaction value. Merchants with the highest average transaction value are the ones we should really tailor our commission strategy for. Getting the right strategy for calculating the commissions of these merchants can have significant impact on the profitability of our business as the transaction value itself is significant.



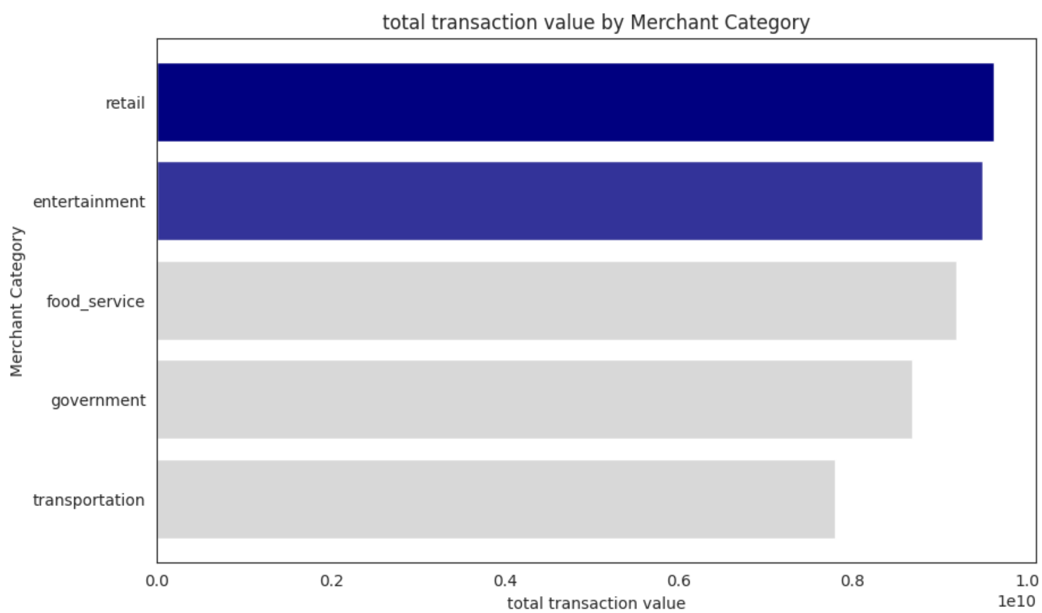
7. **Customers with Highest Declined transactions rate**

Another method to detect risky costumers is by soring them by the frequency of their declined transactions. A high ration of declined transactions can be due to many factors. We should analyze the details to see if something is wrong with the customer. This high decline frequency could be a consequence of fraudulent activity. Therefore, we need to further analyze each of these customers. So, this plot helps us find potential risky customers.



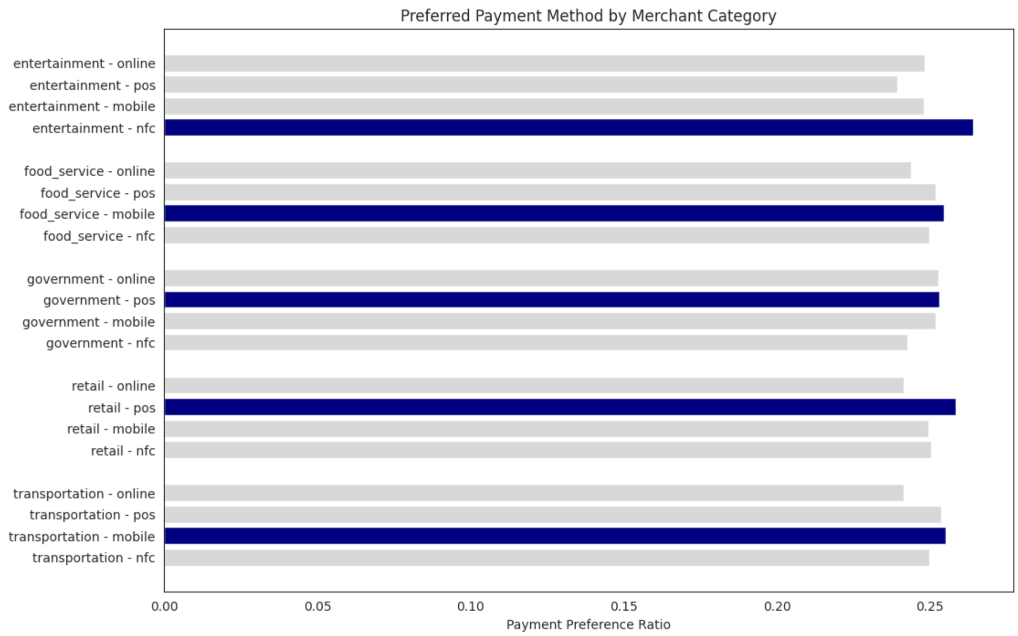
8. Merchants with Highest Transaction Value

This plot combines two of the plots we had giving us a more general view of the state of our merchant categories. Here we are not talking about the number of transactions or the average value of each transaction. But instead, we examine the total transaction value. Just like the total number of transactions and the average transaction value, this plot helps us identify our most important merchant category.



9. Merchant Categories' Preferred Payment Methods

We previously saw the merchant categories with the highest commission to transaction volume and we discussed why we should focus on them and increase their transaction volume. In this section, we can see their preferences when it comes to the payment method. We can use the insight from this plot to facilitate the use of each method for the main customers of each merchant category. This may lead to an increase in the total transaction volume and total commission paid.



AmirArsalan Shahbazi **81010151**

Farnoush Fallah **810101484**

Farjad Fallah **810101483**