Algorithmic Digital Marketing Assignment 2

Summary	In this codelab, we have analyzed Elo Merchant Category Recommendation dataset to gain some marketing insights
URL	https://www.kaggle.com/c/elo-merchant-category-recommendation/data
Category	Data analysis and Visualization
Author	Prathamesh Limaye and Saurabh Satra

About Dataset

There are 2 datasets that contain information about all transactions of these cards buying from different merchants:

- **Historical_transactions.csv**: up to 3 months' worth of historical transactions for each card_id
- **New_merchant_transactions.csv**: two months' worth of data for each card_id containing ALL purchases that card_id made at merchant_ids that were not visited in the historical data.

Lastly, there is 1 dataset that contains information about the merchants:

- **Merchants.csv**: additional information about all merchants / merchant_ids in the dataset.

Data Preprocessing

Data Preprocessing is a technique that is used to convert the raw data into a clean data set.
The data set which we have used has many columns with NULL Values and Missing Values.
Also, the dataset used has several inconsistencies.
We have used XSV, Pandas, and Trifacta for data preprocessing so that the data can be cleaned and is feasible for analysis or visualization.

Using XSV:

We have used XSV for checking the headers of the tables, the frequency of each column data and to sample the data using select and sample commands.

Headers:

```
C:\Users\prath>xsv headers C:\Users\prath\Downloads\elo-merchant-category-recommendation\historical_transactions.csv
1   authorized_flag
2   card_id
3   city_id
4   category_1
5   installments
6   category_3
7   merchant_category_id
8   merchant_id
9   month_lag
10  purchase_amount
11  purchase_date
12  category_2
13  state_id
14  subsector_id

C:\Users\prath>xsv headers C:\Users\prath\Downloads\elo-merchant-category-recommendation\merchants.csv
1   merchant_id
2   merchant_group_id
3   merchant_category_id
4   subsector_id
5   numerical_1
6   numerical_2
7   category_1
8   most_recent_sales_range
9   most_recent_purchases_range
10  avg_sales_lag3
11  avg_purchases_lag3
12  active_months_lag3
13  avg_sales_lag6
14  avg_purchases_lag6
15  active_months_lag6
16  avg_sales_lag12
17  avg_purchases_lag12
18  active_months_lag12
19  category_4
20  city_id
21  state_id
22  category_2
```

Sampling:

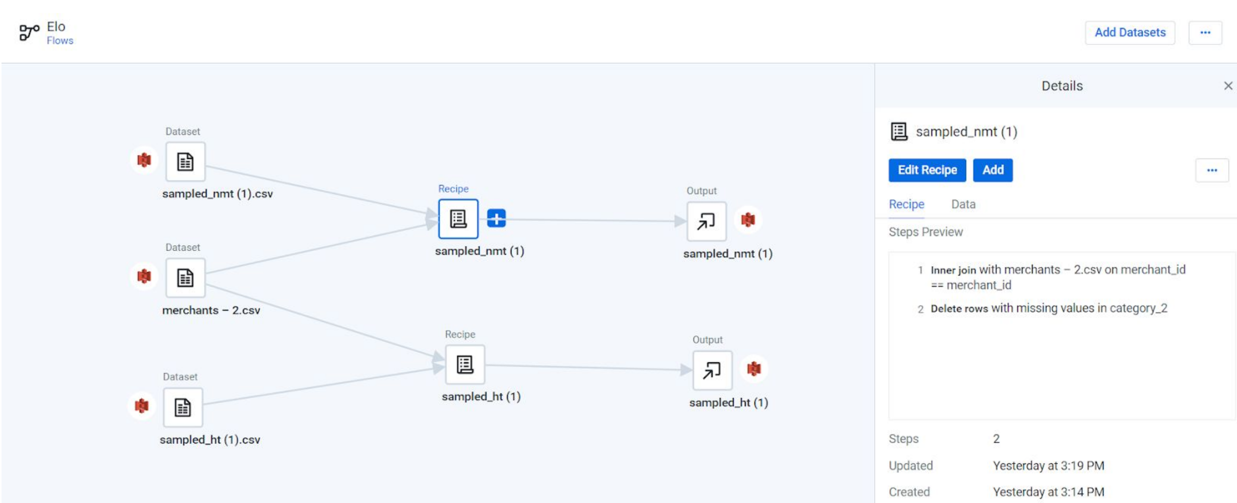
```
C:\Users\prath>xsv select authorized_flag,card_id,city_id,category_1,installments,category_3,merchant_category_id,merchant_id,month_lag,purchase_amount,pur
chase_date,category_2,state_id,subsector_id C:\Users\prath\Downloads\elo-merchant-category-recommendation\new_merchant_transactions.csv | xsv sample 106000
0 > C:\Users\prath\Downloads\elo-merchant-category-recommendation\sampled_nmt.csv

C:\Users\prath>xsv select authorized_flag,card_id,city_id,category_1,installments,category_3,merchant_category_id,merchant_id,month_lag,purchase_amount,pur
chase_date,category_2,state_id,subsector_id C:\Users\prath\Downloads\elo-merchant-category-recommendation\historical_transactions.csv | xsv sample 1060000
> C:\Users\prath\Downloads\elo-merchant-category-recommendation\sampled_ht.csv
```

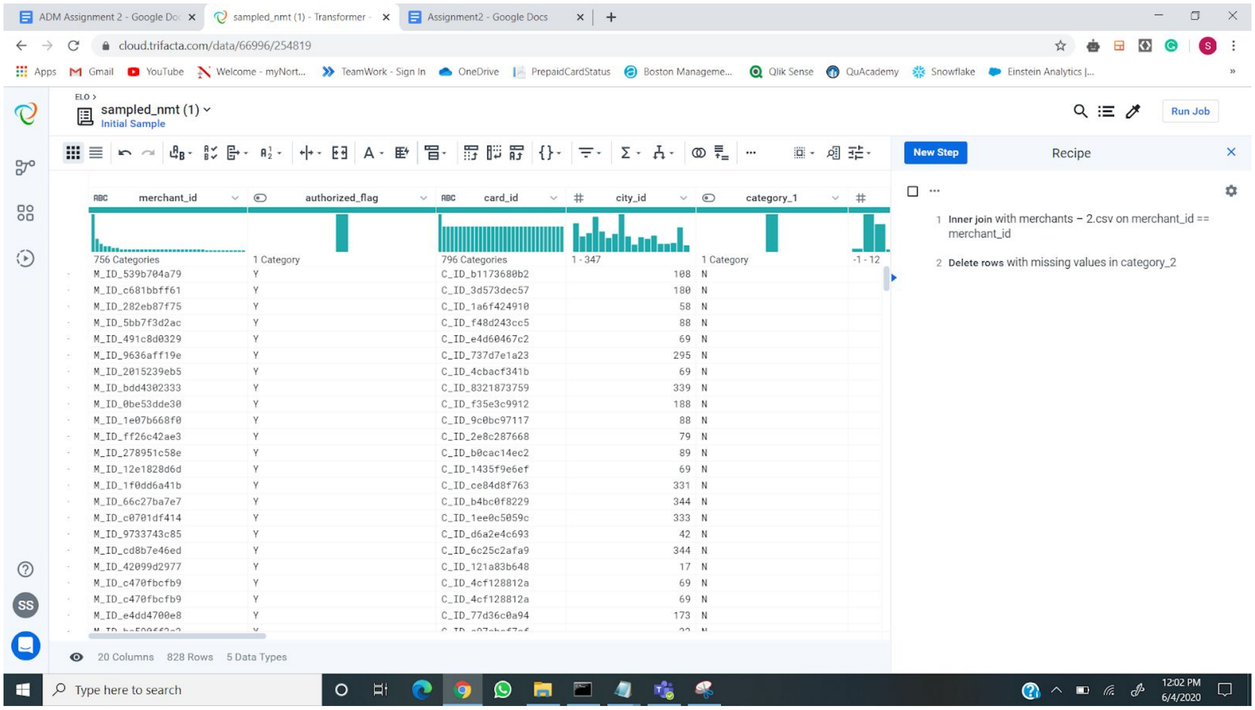
Frequency:

```
C:\Users\prath>xsv frequency C:\Users\prath\Downloads\elo-merchant-category-recommendation\new_merchant_transactions.csv | xsv table
field                 value             count
authorized_flag       Y                 1963031
card_id               C_ID_b7ebee6539   109
card_id               C_ID_c729288535   106
card_id               C_ID_0e4f6af077   104
card_id               C_ID_8946508722   104
card_id               C_ID_6cef1dba4b   100
card_id               C_ID_4e8e856f1a   99
card_id               C_ID_8e0c15d39b   95
card_id               C_ID_4a0143e1a7   92
card_id               C_ID_a9b2895f68   90
card_id               C_ID_5da6f7704e   88
city_id               69                328916
city_id               -1                99349
city_id               158               84962
city_id               19                70961
city_id               17                65300
city_id               143               53997
city_id               88                46301
city_id               137               45007
city_id               331               41429
city_id               87                33967
category_1            N                 1899935
category_1            Y                 63096
installments          0                 922244
installments          1                 836178
installments          -1                55922
installments          2                 54729
installments          3                 44750
installments          4                 14815
installments          6                 10389
installments          5                 9296
installments          10                8899
installments          12                2850
category_3            A                 922244
category_3            B                 836178
category_3            C                 148687
category_3            (NULL)            55922
merchant_category_id  307               191631
merchant_category_id  705               168852
merchant_category_id  278               168140
merchant_category_id  80                144667
merchant_category_id  367               116406
merchant_category_id  683               58176
merchant_category_id  560               57327
```

Using Trifacta:

We have used Trifacta for joining the tables that we had sampled through XSV. Also, we did some cleaning of the data like removing missing values and inconsistent data.





Using Pandas:

We used Pandas for the following analysis:

Reading the data:

```
[ ] data_ht = pd.read_csv('C:/Users/prath/Downloads/sampled_ht.csv')
```

```
[ ] nObs, nFea = data_ht.shape
    print(f'There are {nObs} Observations and {nFea} Features')
```

```
There are 250000 Observations and 14 Features
```

```
[ ] data_ht.columns
```

```
Index(['authorized_flag', 'card_id', 'city_id', 'category_1', 'installments',
       'category_3', 'merchant_category_id', 'merchant_id', 'month_lag',
       'purchase_amount', 'purchase_date', 'category_2', 'state_id',
       'subsector_id'],
      dtype='object')
```

```
[ ] print("First observations:")
    data_ht.head()
```

First observations:

	authorized_flag	card_id	city_id	category_1	installments	category_3	merchant_category_id	merchant_id	month_lag	purchase_amount
0	Y	C_ID_1235abf2e9	14	N	0	A	560	M_ID_d88c2ba479	-8	-0.740146
1	N	C_ID_a85430af23	33	N	1	B	307	M_ID_4ec1f91e4d	-10	-0.611669
2	Y	C_ID_f407be70a5	188	N	1	B	278	M_ID_de68864494	-7	-0.714120
3	Y	C_ID_afaafbc63a	-1	Y	1	B	217	M_ID_5a0a412718	-6	-0.521510
4	Y	C_ID_81496e38bf	20	N	0	A	683	M_ID_01b1eec377	-11	-0.689807

Checking for null values:

```
[ ]  print("Missing values by features:")
     data_ht.isnull().sum(axis=0)
```

```
Missing values by features:
authorized_flag              0
card_id                      0
city_id                      0
category_1                   0
installments                 0
category_3                1576
merchant_category_id         0
merchant_id               1213
month_lag                    0
purchase_amount              0
purchase_date                0
category_2               22917
state_id                     0
subsector_id                 0
dtype: int64
```

Converting the date part into individual columns as Year, Month, Day, Quarter, Week:

```
[ ]  data_nmt['Year'] = pd.DatetimeIndex(data_nmt['purchase_date']).year
     data_nmt['month'] = pd.DatetimeIndex(data_nmt['purchase_date']).month
     data_nmt['Day'] = pd.DatetimeIndex(data_nmt['purchase_date']).day
     data_nmt['Quarter'] = pd.DatetimeIndex(data_nmt['purchase_date']).quarter
     data_nmt['Week'] = pd.DatetimeIndex(data_nmt['purchase_date']).week
```

```
[ ]  data_nmt.columns
```

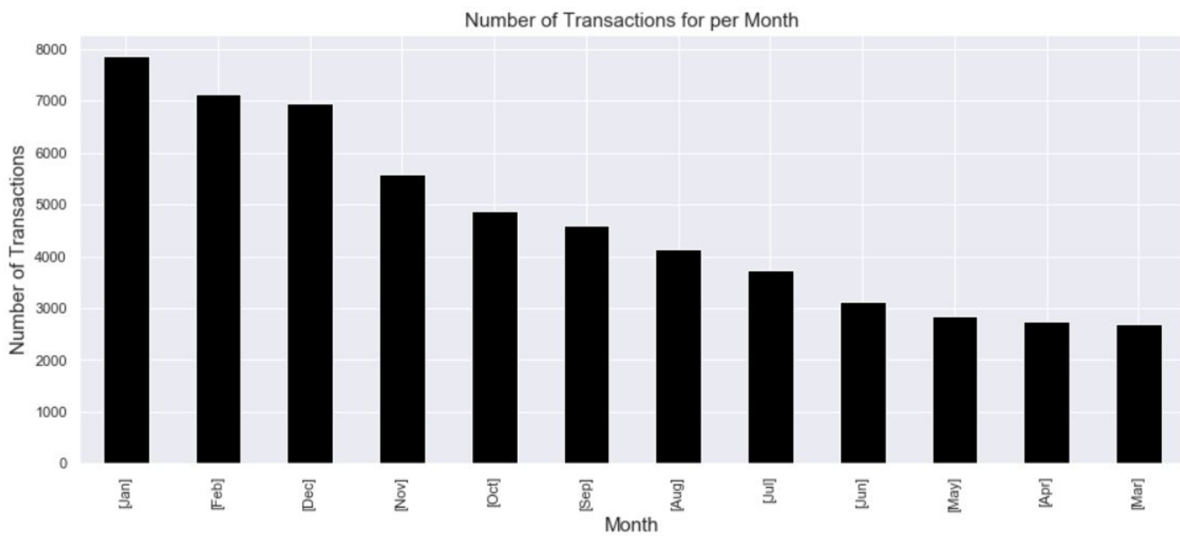
```
Index(['authorized_flag', 'card_id', 'city_id', 'category_1', 'installments',
       'category_3', 'merchant_category_id', 'merchant_id', 'month_lag',
       'purchase_amount', 'purchase_date', 'category_2', 'state_id',
       'subsector_id', 'Year', 'month', 'Day', 'Quarter', 'Week'],
      dtype='object')
```

```
[ ]  daily_data.head(10)
```

	card_id	city_id	merchant_id	purchase_date	state_id	subsector_id	Year	month	Day	Quarter	Week	Weekday	purchase_amount
0	C_ID_00007093c1	76	M_ID_edeafa75d9	2017-11-10 16:21:22	2	33	2017	11	10	4	45	4	-0.569219
1	C_ID_00007093c1	244	M_ID_9400cf2342	2017-08-28 19:21:16	2	19	2017	8	28	3	35	0	-0.683796
2	C_ID_0001506ef0	137	M_ID_b1fc88154d	2018-02-08 14:30:56	19	33	2018	2	8	1	6	3	1.493545
3	C_ID_0001793786	204	M_ID_f17a1b0efa	2017-10-27 13:51:16	24	41	2017	10	27	4	43	4	-0.679288

Creating visualizations:

```
Monthly_transactions = data.groupby('card_id')['month'].unique().value_counts().iloc[0:12].plot(kind ='bar',color='black'
Monthly_transactions_csv = data.groupby('card_id')['month'].unique().value_counts().iloc[0:12]
Monthly_transactions.set_xlabel('Month',fontsize=15)
Monthly_transactions.set_ylabel('Number of Transactions',fontsize=15)
Monthly_transactions.set_title('Number of Transactions for per Month',fontsize=15)
plt.xticks();
Monthly_transactions_csv.to_csv('C:/Users/prath/Downloads/CSV_files/Monthly_transactions.csv')
```



RFM

RFM segmentation is a powerful way to identify groups of customers for special treatment. RFM stands for recency, frequency, and monetary.

- **Recency** - This represents the age of the customer when they made their latest transactions. (Current_date - last_transaction_date)
- **Frequency** - This represents the total number of transactions/number of visits a customer has made. (Count of total transactions)
- **Monetary** - This represents the total purchase amount that a specified customer has made. (Sum of purchase_amt)

- **Time** - This represents the age of the customer. The time span between a customer's first and last transactions.

To perform an RFM analysis, each customer is assigned a score for recency, frequency, and monetary value, and then a final RFM score is calculated.

Recency score is calculated based on the date of their most recent purchase. The scores are generally categorized based on the values.

Similarly, the frequency score is calculated based on the number of times the customers purchased. Customers with higher frequency receive a higher score.

Finally, customers are assigned a score based on the amount they spent on their purchases. For calculating this score, you may consider the actual amount spent or the average spent per visit.

By combining these three scores, a final RFM score is calculated. The customers with the highest RFM score are considered to be the ones that are most likely to respond to their offers.

card_id	Frequency	Monitary	Time	Recency	AOV
C_ID_8eb97da9da	2	48090.089996	55.0	902.0	24045.044998
C_ID_dfada69aa2	4	42085.421926	316.0	893.0	10521.355482
C_ID_54707b0914	4	12031.074815	344.0	840.0	3007.768704
C_ID_edca884f4a	3	8436.988117	57.0	985.0	2812.329372
C_ID_f7621c5e17	5	9929.383120	112.0	840.0	1985.876624

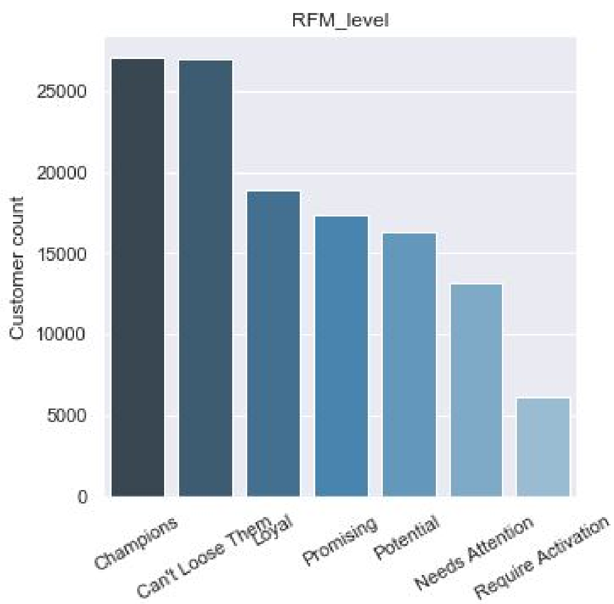
Customer Lifetime Value (CLV):- The lifetime value of a customer, or customer lifetime value (CLV), represents the total amount of money a customer is expected to spend in your business, or on your products, during their lifetime.

We calculated Customer Lifetime Values by calculating different parameters such as profit margin, purchase frequency, repeat rate, churn rate etc.

card_id	Frequency	Monitary	Time	Recency	AOV	profit_margin	CLV	cust_lifetime_value
C_ID_8eb97da9da	2	48090.089996	55.0	902.0	24045.044998	0.2	72678.310039	14535.662008
C_ID_dfada69aa2	4	42085.421926	316.0	893.0	10521.355482	0.4	31801.742762	12720.697105
C_ID_54707b0914	4	12031.074815	344.0	840.0	3007.768704	0.4	9091.251291	3636.500516
C_ID_f7621c5e17	5	9929.383120	112.0	840.0	1985.876624	0.5	6002.490617	3001.245308
C_ID_edca884f4a	3	8436.988117	57.0	985.0	2812.329372	0.3	8500.518342	2550.155503

We divided customers using RFM scores into several sections such as Can’t Loose Them (Highest RFM Score) or Require Activation (Lowest RFM score).

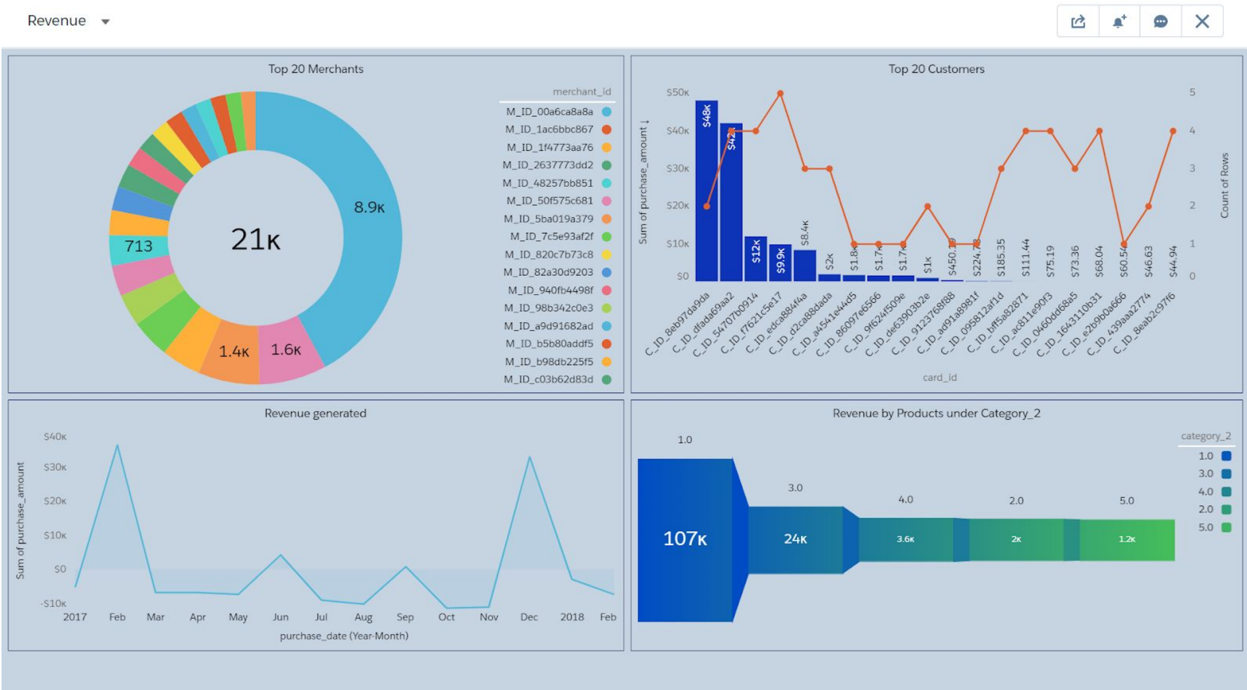
	Recency	Frequency	Monitary	
	mean	mean	mean	count
RFM_Level				
Can't Loose Them	977.0	3.6	13.3	26970
Champions	991.4	1.8	4.7	27075
Loyal	1028.7	1.3	3.2	18908
Needs Attention	887.0	1.0	2.3	13175
Potential	1014.6	1.0	2.4	16237
Promising	943.7	1.0	2.4	17332
Require Activation	855.1	1.0	2.3	6124



Dashboards

Insights based on Revenue:

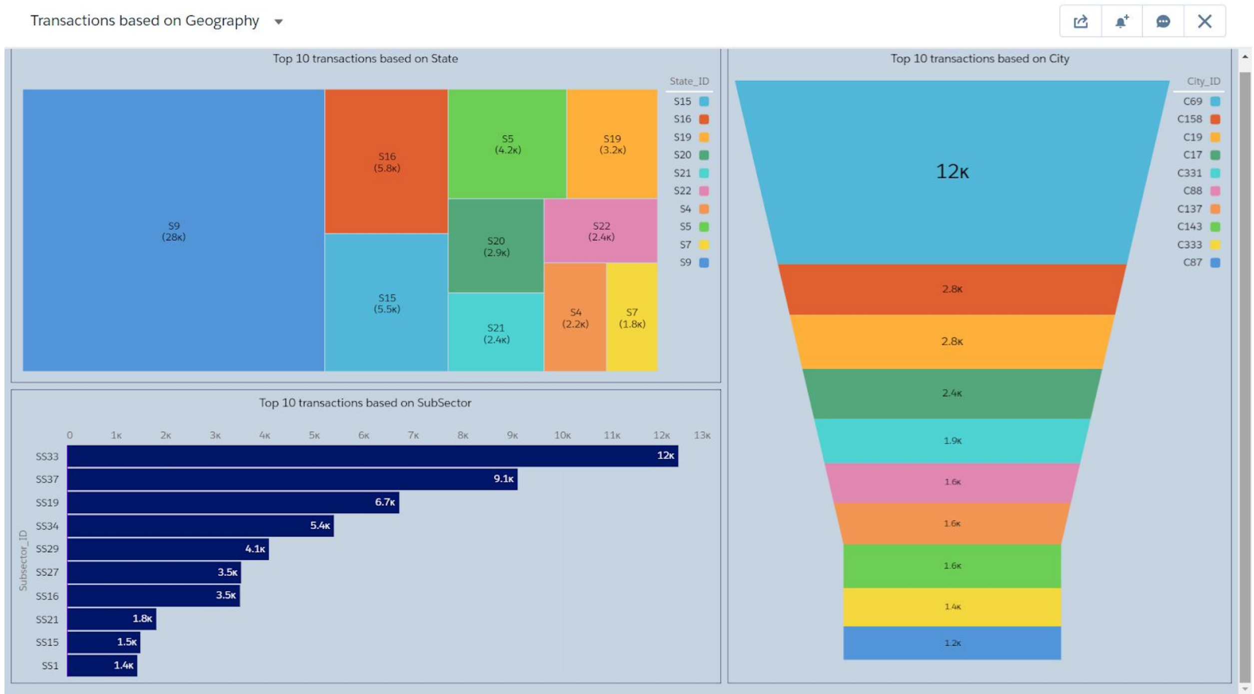
This Dashboard gives us the top merchants, top customers, top products based on revenue.



1. The merchant M_ID_00a6ca8a8a has the most transactions: 8858 transactions
2. Customer C_ID_8eb97da9da has spent the most with the purchase amount of $48,084.09 amongst all customers
3. The most popular product was ‘1.0’ under Category_2 with revenue of around $107k

Insights based on Geography:

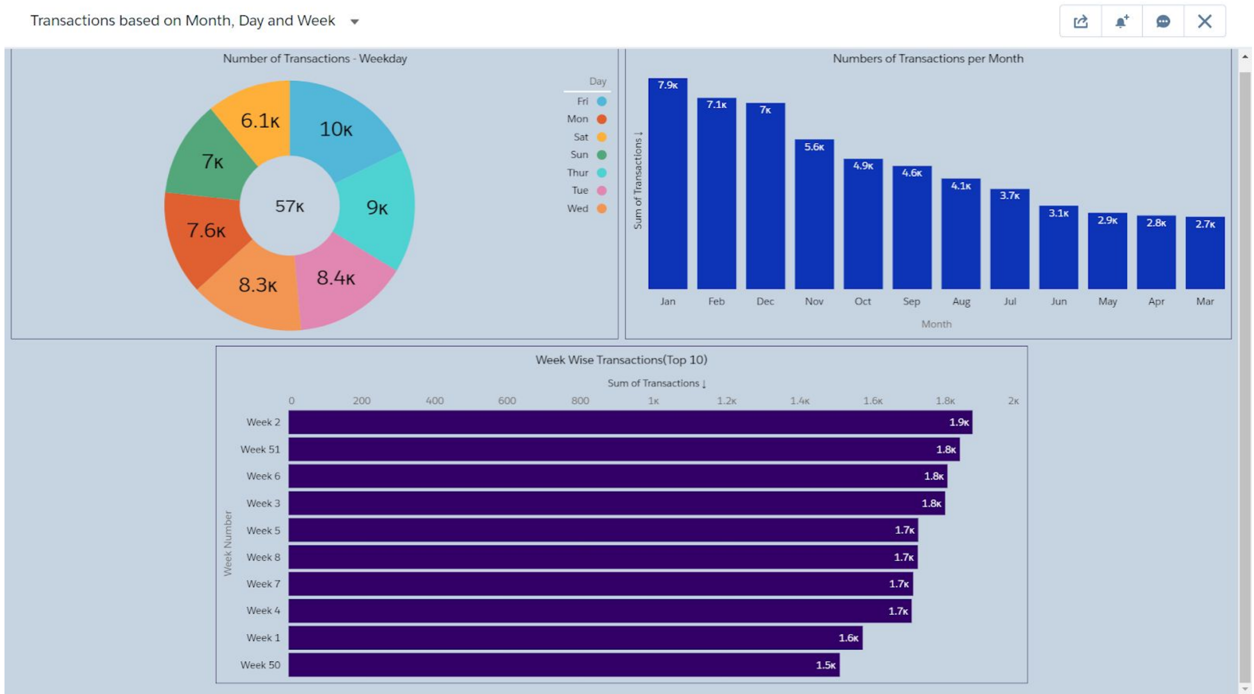
This Dashboard gives us insights about the most popular state, city, and subsector based on the number of transactions.



Category	Number of Transactions
State - S9	27,736
City - C69	11,868
Subsector - SS33	12,344

Insights based on Calendar:

The Dashboard shown below highlights the most favored weekday, month and week of the year based on the number of transactions



Most Desired	**Value**	**Number of Transactions**
Day	Monday	10,028
Month	Jan	7,882
Week	Week 2	1,834