BANK CUSTOMER SEGMENTATION

CONTRIBUTORS: Akarsha Mandrekar, Dhruv Jain, Sathya Krishnan TS and Shreekant Kambli

GITHUB LINK: Bank-customer-segmentation

OVERVIEW:

Market segmentation is the process of dividing a market (mostly done by companies and organizations) into various segments to meet the needs of customers more efficiently and to increase the company's profits.

What is the need for segmentation? Well, most of the markets now have a varied customer base and a single product cannot satisfy them all. This is where segmentation plays a key role. Segmentation gives an organization the necessary insights to not only modify their current products but also gives them an idea about the future services/products they must produce.

PROBLEM STATEMENT:

Every city has at least a bank and each bank has a lot of customers. Banks have the most varied customer base. With the advent of transactions through mobile, the number of transactions has exploded, and each bank is now processing a minimum of million transactions per month. Our goal here is to perform segmentation on a bank customer dataset that consists of a million transactions and find out what is the criteria that defines each segment.

DATASET:

	TransactionID	CustomerID	CustomerDOB	CustGender	CustLocation	CustAccountBalance	TransactionDate	TransactionTime	TransactionAmount (INR)
0	T1	C5841053	10/1/94	F	JAMSHEDPUR	17819.05	2/8/16	143207	25.00
1	T2	C2142763	4/4/57	M	JHAJJAR	2270.69	2/8/16	141858	27999.00
2	T3	C4417068	26/11/96	F	MUMBAI	17874.44	2/8/16	142712	459.00
3	T4	C5342380	14/9/73	F	MUMBAI	866503.21	2/8/16	142714	2060.00
4	T5	C9031234	24/3/88	F	NAVI MUMBAI	6714.43	2/8/16	181156	1762.50
5	T6	C1536588	8/10/72	F	ITANAGAR	53609.20	2/8/16	173940	676.00
6	T7	C7126560	26/1/92	F	MUMBAI	973.46	2/8/16	173806	566.00
7	T8	C1220223	27/1/82	M	MUMBAI	95075.54	2/8/16	170537	148.00
8	T9	C8536061	19/4/88	F	GURGAON	14906.96	2/8/16	192825	833.00
9	T10	C6638934	22/6/84	M	MUMBAI	4279.22	2/8/16	192446	289.11
10	T11	C5430833	22/7/82	M	MOHALI	48429.49	2/8/16	204133	259.00
11	T12	C6939838	7/7/88	M	GUNTUR	14613.46	2/8/16	205108	202.00
12	T13	C6339347	13/6/78	M	AHMEDABAD	32274.78	2/8/16	203834	12300.00
13	T14	C8327851	5/1/92	F	THANE	59950.44	1/8/16	84706	50.00
14	T15	C7917151	24/3/78	M	PUNE	10100.84	1/8/16	82253	338.00
15	T16	C8334633	10/7/68	F	NEW DELHI	1283.12	1/8/16	125725	250.00
16	T17	C1376215	1/1/1800	M	MUMBAI	77495.15	1/8/16	124727	1423.11
17	T18	C8967349	16/7/89	M	MUMBAI	2177.85	1/8/16	124734	54.00
18	T19	C3732016	11/1/91	M	MUMBAI	32816.17	1/8/16	122135	315.00
19	T20	C8999019	24/6/85	M	PUNE	10643.50	1/8/16	152821	945.00

The dataset is from Kaggle. It consists of more than a million rows and nine columns. It has the details of a transaction. The transactions are from all over India.

Dataset Link: https://www.kaggle.com/shivamb/bank-customer-segmentation

Python libraries such as numpy, pandas, matplotlib, seaborn and scikit-learn will be used to perform segmentation.

DATA PREPARATION:

In this section, we prepare the dataset, so that segmentation can be done in an effective way. Here the outliers, discrepancies and some absurd values in the dataset have been spotted and removed from the dataset. Since the dataset contains a million rows, removing some of the rows is not going to affect the final result.

The 'CustomerDOB' column had 1800 as the birth year and it had to be removed.

```
Name: CustomerDOB, Length: 17233, dtype: int64
```

```
In [17]: df = df.loc[~(df['CustomerDOB']=='1/1/1800')]
```

The 'TransactionDate' and 'CustomerDOB' columns were of string type and they were converted to 'datetime' type.

```
df['TransactionDate'] = pd.to_datetime(df['TransactionDate'], format = '%d/%m/%y')
df['CustomerDOB'] = pd.to_datetime(df['CustomerDOB'])
```

```
df.head()
```

	CustomerID	CustomerDOB	CustGender	CustLocation	CustAccountBalance	TransactionDate	TransactionTime	TransactionAmount (INR)
0	C5841053	1994-10-01	F	JAMSHEDPUR	17819.05	2016-08-02	143207	25.0
1	C2142763	2057-04-04	M	JHAJJAR	2270.69	2016-08-02	141858	27999.0
2	C4417068	1996-11-26	F	MUMBAI	17874.44	2016-08-02	142712	459.0
3	C5342380	1973-09-14	F	MUMBAI	866503.21	2016-08-02	142714	2060.0
4	C9031234	1988-03-24	F	MUMBAI	6714.43	2016-08-02	181156	1762.5

Finally, a column called 'CustomerAge' is created which tells us about the age of the customer during the transaction.

```
df.loc[df['CustomerDOB'].dt.year >= 2021, ['CustomerDOB']] -= pd.DateOffset(years = 100)
df['CustomerAge'] = (pd.to_datetime('today') - df['CustomerDOB'])/np.timedelta64(1, 'Y')
```

DATA PREPROCESSING:

Data preprocessing is a very important step as it builds the base upon which ML models will do their work. In this stage, we will explore the dataset and find the secrets that our dataset holds. We also create some useful features from the already existing features. All these things will allow us to better segment the customer dataset.

The following processes have been performed on the dataset in this stage:

- One-hot encoding for gender category.
- Hour and minute of transaction.
- Dividing customer age into categories.
- Dividing the transaction amount into categories.
- AM or PM.
- Basic visualization.
- Scaling the appropriate features.

One-hot encoding is necessary if we are going to fit our dataset to a lot of models as some models have a problem with 'string' type. Performing one-hot encoding will also speed up the process.

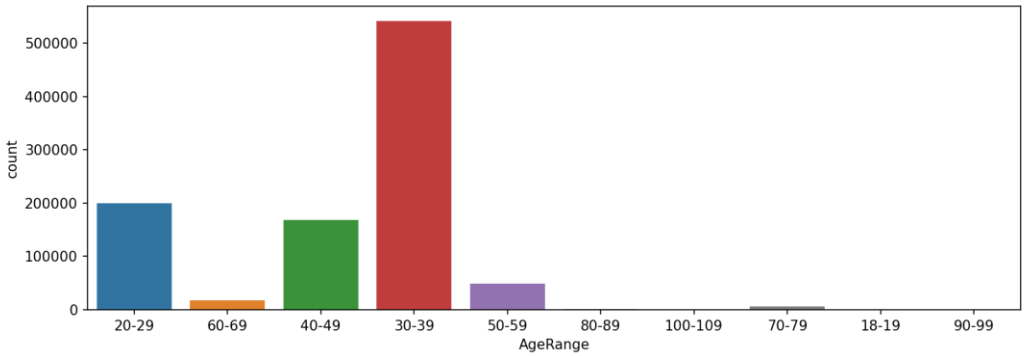
Hour of transaction is a very important feature and it can also be used as a segmentation criteria.

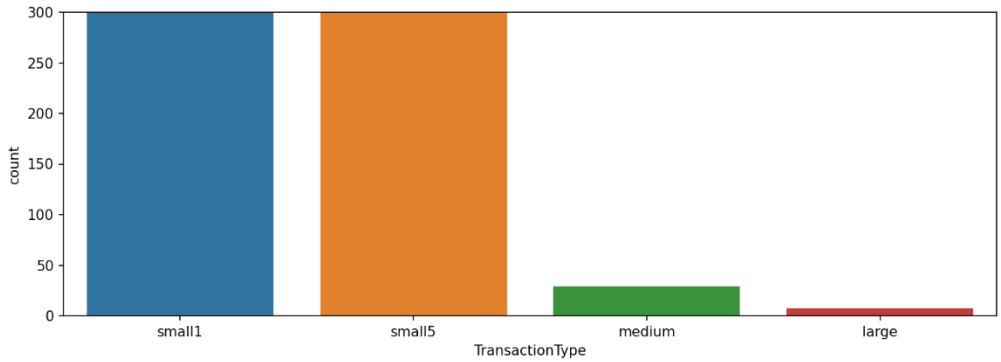
Since people of the same age category go for a particular kind of transaction, a separate feature column is created which specifies the category they belong to.

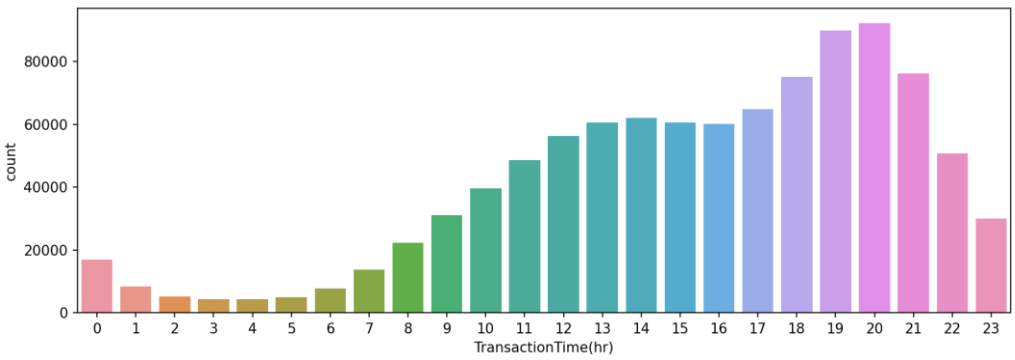
Forming an ordinal feature column from the continuous 'TransactionAmount (INR)' feature column is another important feature as it gives us an idea about the range of amount the bank is dealing with.

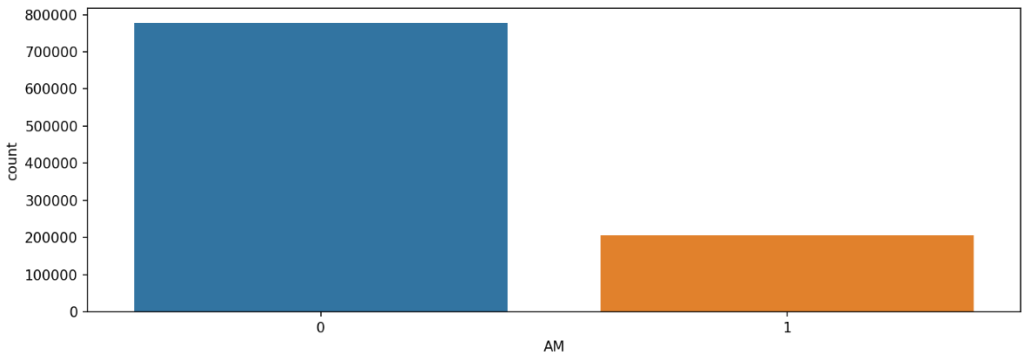
AM or PM is used to find out the part of the day in which most transactions take place.

Here, the features are created with the forethought of making the segmentation easier and clearer. Visualization has been performed at this stage to justify the creation of these features.









Scaling is done here for ‘TransactionAmount (INR)’ and ‘CustAccountBalance’ as a good amount of the clustering methods are distance based and we don’t want a particular feature to dominate other features.

SCALING THE DATASET

```
df.columns
```

```
Index(['CustomerID', 'CustomerDOB', 'CustLocation', 'CustAccountBalance',
       'TransactionDate', 'TransactionAmount (INR)', 'CustomerAge',
       'TransactionMonth', 'M', 'TransactionTime(hr)', 'TransactionTime(min)',
       'AgeRange', 'TransactionRange(L)', 'TransactionType', 'AM'],
      dtype='object')
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X = df[['TransactionAmount (INR)','CustAccountBalance']]
scaled_X = scaler.fit_transform(X)
scaled_df = df
scaled_df[['TransactionAmount (INR)','CustAccountBalance']] = scaled_X
```

```
scaled_df
```

	CustomerID	CustomerDOB	CustLocation	CustAccountBalance	TransactionDate	TransactionAmount (INR)	CustomerAge	TransactionMonth	M	Transac
0	C5841053	1994-10-01	JAMSHEDPUR	-0.107885	2016-08-02	-0.232423	27	8	0	
1	C2142763	1957-04-04	JHAJJAR	-0.126892	2016-08-02	4.323506	64	8	1	
2	C4417068	1996-11-26	MUMBAI	-0.107818	2016-08-02	-0.161741	25	8	0	
3	C5342380	1973-09-14	MUMBAI	0.929583	2016-08-02	0.099003	48	8	0	
4	C9031234	1988-03-24	MUMBAI	-0.121460	2016-08-02	0.050551	33	8	0	
...	...	...	...	...	...	...	...	...	...	
1048562	C8020229	1990-08-04	DELHI	-0.120335	2016-09-18	-0.106367	31	9	1	