Customer Segmentation using Cluster Analysis and Discriminant Analysis

- Shreya Singireddy
-

The customer segmentation is the process of dividing customers into groups based on their characteristics which can help corporation and companies to understand and target their customers better with their products. For example, if we identify a segment of high spenders, the company may want to offer exclusive promotions or events to this group. On the other hand, if we identify a segment of price-sensitive customers who prefer to stick to their budget, the company may want to focus on offering discounts to this group.

In this project, we plan to distinguish and find similarities between different customer groups using Cluster Analysis and Discriminant Analysis for a hypothetical company based on a credit card dataset.

How will this project help the hypothetical company?

Major corporations and large size retail outlets often have difficulty keeping a track about the buying habits of each and every customer, but it is important that businesses keep a consistent base of customers and simultaneously try to expand the customer base, but most major corporations and large size retail outlets often have difficulty keeping a track about the buying habits of each and every customer. In order to achieve that, we need to study the 'best' customers i.e., the regular, high-spending customers as well as the high-risk of churning or 'dormant' customers. This prevents the company from spending its limited resources on customers that are likely to churn and prevents the valuable customers from churning by offering them personalized marketing campaigns.

We can use the customer credit card behaviour to understand their transactional patterns and hence, target them with campaigns accordingly.

We will use K-means clustering algorithm to identify the customer segments from the variables given in the dataset and Linear Discriminant Analysis in order to make a comparison of results from K-means clustering and see if the variables chosen to help in distinguishing different customer segments.

With these results, we will make marketing recommendations on the basis of the customer segments achieved. In detail we will discuss, what kind of customers we are encountering and how the retail company can market to them.

Understanding the Dataset:

The 'Credit Card' dataset we are using contains usage behaviour of 9000 credit card users for a 6-month period.

Data Attributes: (As per the resource)

8950 instances, 19 variables.

CUST_ID : Identification of Credit Card holder (Categorical)
BALANCE : Balance amount left in their account to make purchases
BALANCE_FREQUENCY : How frequently the Balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated)
PURCHASES : Amount of purchases made from account
ONEOFF_PURCHASES : Maximum purchase amount done in one-go
INSTALLMENTS_PURCHASES : Amount of purchase done in instalments
CASH_ADVANCE : Cash in advance given by the user
PURCHASES_FREQUENCY : How frequently the Purchases are being made, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)
ONEOFFPURCHASESFREQUENCY : How frequently Purchases are happening in one-go, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)
PURCHASESINSTALLMENTSFREQUENCY : How frequently purchases in instalments are being done, score between 0 and 1 (1 = frequently done, 0 = not frequently done)
CASHADVANCEFREQUENCY : How frequently the cash in advance being paid
CASHADVANCETRX : Number of Transactions made with "Cash in Advanced"
PURCHASES_TRX : Number of purchase transactions made
CREDIT_LIMIT : Limit of Credit Card for user
PAYMENTS : Amount of Payment done by user
MINIMUM_PAYMENTS : Minimum amount of payments made by user
PRCFULLPAYMENT : Percent of full payment paid by user
TENURE : Tenure of credit card service for user

Dataset Exploration:

Initially, let us try to understand our data through various techniques.

Basic Summary:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	\
count	8950.000000	8950.000000	8950.000000	8950.000000	
mean	1564.474828	0.877271	1003.204834	592.437371	
std	2081.531879	0.236904	2136.634782	1659.887917	
min	0.000000	0.000000	0.000000	0.000000	
25%	128.281915	0.888889	39.635000	0.000000	
50%	873.385231	1.000000	361.280000	38.000000	
75%	2054.140036	1.000000	1110.130000	577.405000	
max	19043.138560	1.000000	49039.570000	40761.250000	

	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	\
count	8950.000000	8950.000000	8950.000000	
mean	411.067645	978.871112	0.490351	
std	904.338115	2097.163877	0.401371	
min	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.083333	
50%	89.000000	0.000000	0.500000	
75%	468.637500	1113.821139	0.916667	
max	22500.000000	47137.211760	1.000000	

	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY
count	8950.000000	8950.000000
mean	0.202458	0.364437
std	0.298336	0.397448
min	0.000000	0.000000
25%	0.000000	0.000000
50%	0.083333	0.166667
75%	0.300000	0.750000
max	1.000000	1.000000

	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_TRX	CREDIT_LIMIT
count	8950.000000	8950.000000	8950.000000	8949.000000
mean	0.135144	3.248827	14.709832	4494.449450
std	0.200121	6.824647	24.857649	3638.815725
min	0.000000	0.000000	0.000000	50.000000
25%	0.000000	0.000000	1.000000	1600.000000
50%	0.000000	0.000000	7.000000	3000.000000
75%	0.222222	4.000000	17.000000	6500.000000
max	1.500000	123.000000	358.000000	30000.000000

	PAYMENTS	MINIMUM_PAYMENTS	PRC_FULL_PAYMENT	TENURE
count	8950.000000	8637.000000	8950.000000	8950.000000
mean	1733.143852	864.206542	0.153715	11.517318
std	2895.063757	2372.446607	0.292499	1.338331
min	0.000000	0.019163	0.000000	6.000000
25%	383.276166	169.123707	0.000000	12.000000
50%	856.901546	312.343947	0.000000	12.000000
75%	1901.134317	825.485459	0.142857	12.000000
max	50721.483360	76406.207520	1.000000	12.000000

Checking the missing values:

CUST_ID	0
BALANCE	0
BALANCE_FREQUENCY	0
PURCHASES	0
ONEOFF_PURCHASES	0
INSTALLMENTS_PURCHASES	0
CASH_ADVANCE	0
PURCHASES_FREQUENCY	0
ONEOFF_PURCHASES_FREQUENCY	0
PURCHASES_INSTALLMENTS_FREQUENCY	0
CASH_ADVANCE_FREQUENCY	0
CASH_ADVANCE_TRX	0
PURCHASES_TRX	0
CREDIT_LIMIT	1
PAYMENTS	0
MINIMUM_PAYMENTS	313
PRC_FULL_PAYMENT	0
TENURE	0
dtype: int64	

There are missing values in CREDIT_LIMIT (1 missing value) and MINIMUM_PAYMENTS (313 missing values).

Checking the percentage of outliers in each variable:

```
Outliers in "BALANCE": 7.77%
Outliers in "BALANCE_FREQUENCY": 16.68%
Outliers in "PURCHASES": 9.03%
Outliers in "ONEOFF_PURCHASES": 11.32%
Outliers in "INSTALLMENTS_PURCHASES": 9.69%
Outliers in "CASH_ADVANCE": 11.51%
Outliers in "PURCHASES_FREQUENCY": 0.0%
Outliers in "ONEOFF_PURCHASES_FREQUENCY": 8.74%
Outliers in "PURCHASES_INSTALLMENTS_FREQUENCY": 0.0%
Outliers in "CASH_ADVANCE_FREQUENCY": 5.87%
Outliers in "CASH_ADVANCE_TRX": 8.98%
Outliers in "PURCHASES_TRX": 8.56%
Outliers in "CREDIT_LIMIT": 2.77%
Outliers in "PAYMENTS": 9.03%
Outliers in "MINIMUM_PAYMENTS": 9.74%
Outliers in "PRC_FULL_PAYMENT": 16.47%
Outliers in "TENURE": 15.26%
```

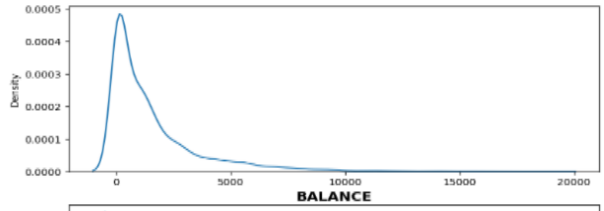
Most of the variables in the dataset have outliers in the dataset. The generic method of handling outliers in the dataset is through deleting the values, which is not recommended as that causes data loss.

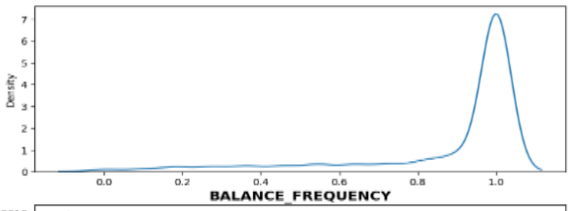
Hence, we will be imputing the data and then scaling the data before performing clustering for a fair result.

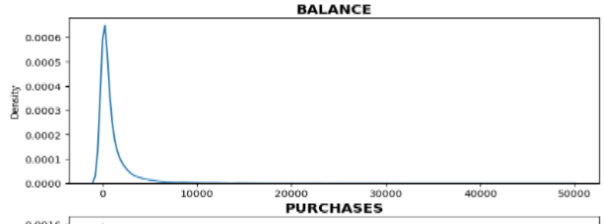
Understanding the Density Plot Distributions of the individual variables:

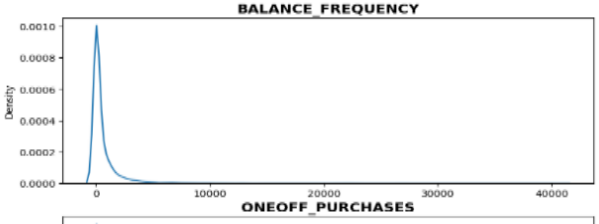
Based on the plots below:

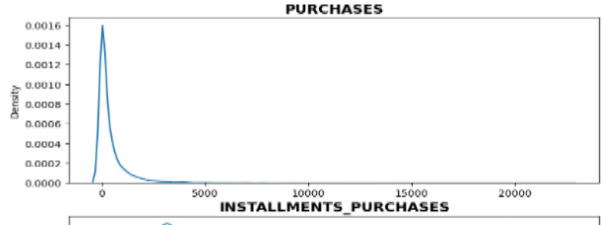
- Most columns have a massive number of 0 values as per the given distribution among the individual plots.
- Most columns are highly right-skewed, as seen from the density plot.
- When looking into the BALANCE and PURCHASE column plot , there are many credit cards with 0 balances. Based on the plots, it can be assumed that some users are intentionally keeping the balance low in their account in order to get high credit limit, which affects to increase credit utilization ratio and the credit score.
- As per the TENURE distribution plot given, when using a credit card, most credit card customers prefer 12 months tenure compared to other tenure options indicating that the customers would be more likely to repay credits in the long term with the consequence of a higher interest rate.
- In BALANCE_FREQUENCY column, most of credit card accounts have 1 score, which indicates that most customers use credit card frequently.
- In ONEOFF_PURCHASES and PURCHASES_INSTALLMENT_FREQUENCY, we notice that the majority of customers do not use credit cards for one-time transactions or payments in instalments.

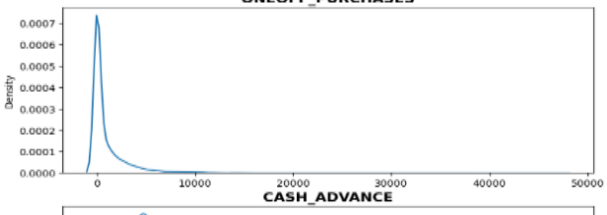

Density
BALANCE

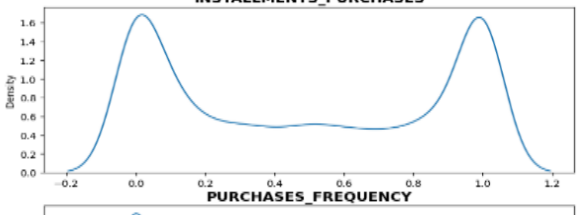

Density
BALANCE_FREQUENCY

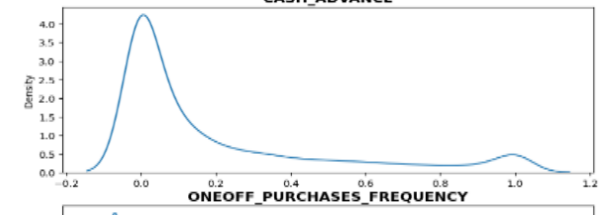

Density
PURCHASES

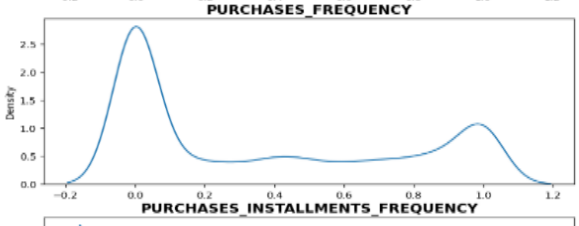

Density
ONEOFF_PURCHASES

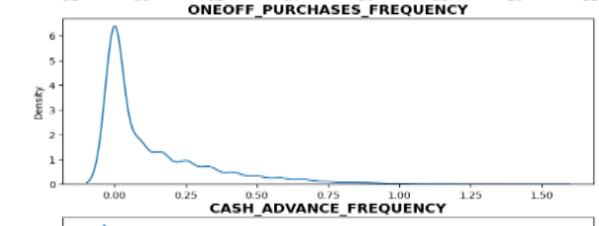

Density
INSTALLMENTS_PURCHASES

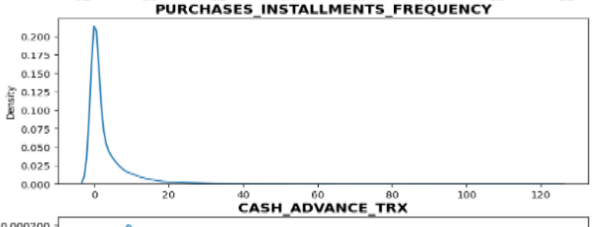

Density
CASH_ADVANCE

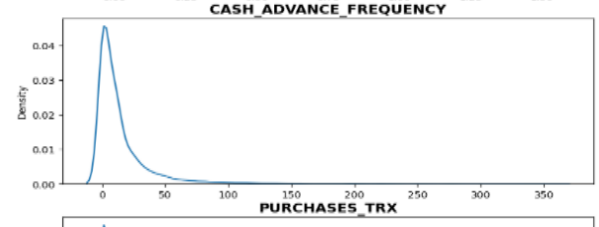

Density
PURCHASES_FREQUENCY

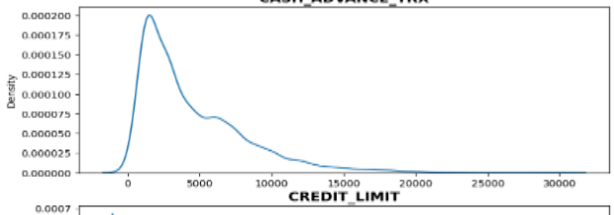

Density
ONEOFF_PURCHASES_FREQUENCY

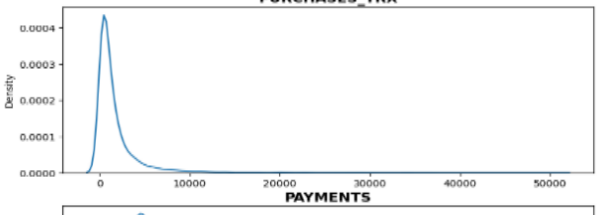

Density
PURCHASES_INSTALLMENTS_FREQUENCY

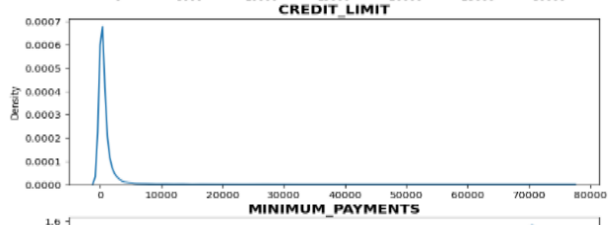

Density
CASH_ADVANCE_FREQUENCY

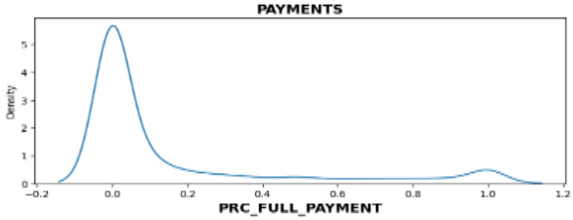

Density
CASH_ADVANCE_TRX

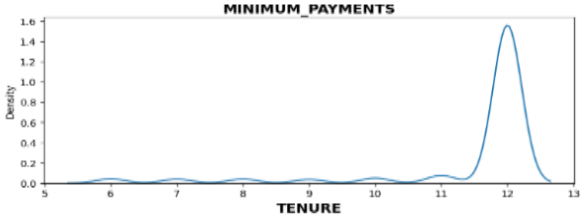

Density
PURCHASES_TRX


Density
CREDIT_LIMIT


Density
PAYMENTS


Density
MINIMUM_PAYMENTS


Density
PRC_FULL_PAYMENT


Density
TENURE

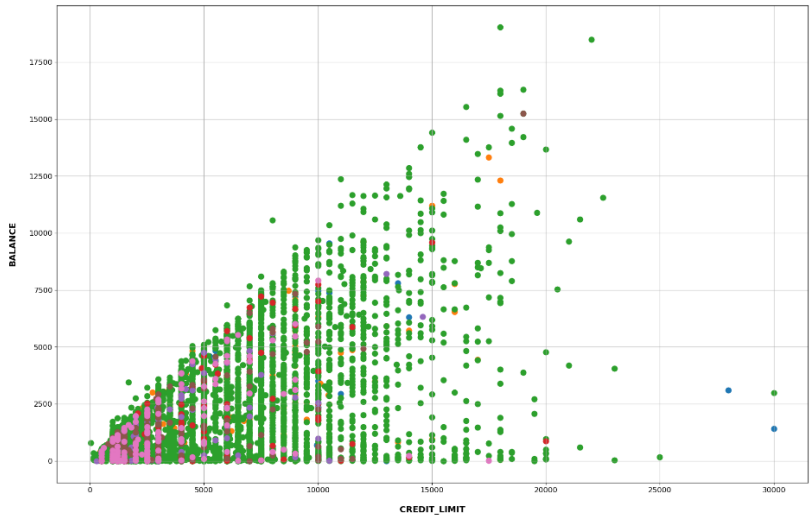
Looking into the Correlation Plot:

From above correlation plot, we can understand that:

- PURCHASES and ONEOFF_PURCHASES has a high correlation with a 0.92 correlation value. This suggests that if the customers made high number of purchases from account, there is a high chance that they made high end purchases.
- CASH_ADVANCE_TRX with CASH_ADVANCE_FREQUENCY has a high correlation with a 0.8 correlation value.
- PURCHASES and INSTALLMENT_PURCHASES are highly correlated.
- PURCHASE_FREQUENCY and CASH_ADVANCE_FREQUENCY are inversely correlated. i.e., if the frequency of purchases are high, the number of times cash is paid in advance is less and vice-versa.

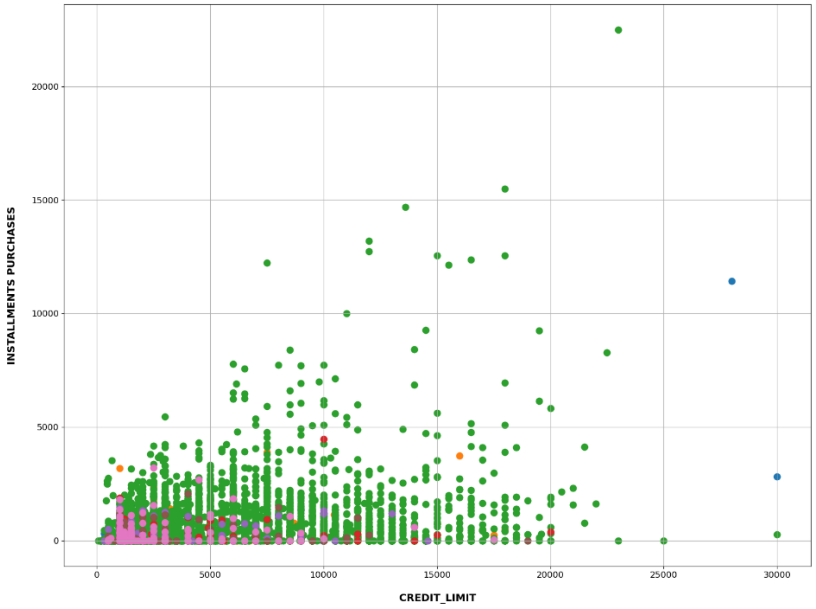
- The variables, CASH_ADVANCE, CASH_ADVANCE_FREQUENCY, and CREDIT_LIMIT, show a stronger correlation with BALANCE.
- The variables, PURCHASES and ONE_OFF_PURCHASES, show a stronger correlation with the PAYMENTS variable.
- The variables, CASH_ADVANCE and CASH_ADVANCE_FREQUENCY, show a negative correlation with the TENURE variable.

Bivariate Plot Comparison: Credit Limit v/s Balance based on Tenure.



The scatter plot shows that when the CREDIT_LIMIT value increase, the distribution/residual variance based on BALANCE and TENURE also increases. This aligns with the assumption made from the distribution plot that most credit card customers prefer 12 months tenure and aligns with the correlation plot done earlier, where it suggests that CREDIT_LIMIT and BALANCE have a positive correlation.

Bivariate Plot Comparison: Credit Limit v/s Instalment Purchases



One can assume that there would be a relationship between CREDIT_LIMIT and INSTALLMENT_PURCHASES as there might a specific impact of instalment-based purchases on available credit based on at least the percentage of the credit limit that a cardholder, but there is no correlation between both variables since the scatter plot shows a random pattern. This is reflective in the correlation plot as well where CREDIT_LIMIT and INSTALLMENT_PURCHASES have minimal positive correlation.

Data Pre-processing:

Removing Categorical Variables:

In the first stage is to remove variables that do not contribute to clustering; hence, we will be removing CUST_ID, as it is simply a unique ID for each row.

Imputation:

We noticed while performing basic analysis that there are missing values present, hence, we will impute that dataset using KNN-Imputation. KNN or K-Nearest Neighbour is a popular algorithm in machine learning used for both classification and regression. KNN can also be used as an imputation method. The method leverages that similar data points are likely to have similar values for the missing attribute.

Initially, the missing attributes are identified, the relevant features are selected by the model, the distance is calculated and then we determine the nearest neighbours. Once the nearest neighbours are identified, we take the weighted average of the attribute values of the nearest neighbours and assign it as the imputed value for the missing attribute of the data point. Then, we repeat the same for all the missing values. Please find the same of the code.

In [53]:
```
from sklearn.impute import KNNImputer

null_col = credit_card2.columns[credit_card2.isnull().any()].tolist()

imputer = KNNImputer()
credit_card2_imp = pd.DataFrame(imputer.fit_transform(credit_card2[null_col]), columns=null_col)
credit_card2 = credit_card2.fillna(credit_card2_imp)

credit_card2.head().style.background_gradient(vmin=6.7, vmax=21.6).hide_index()
```

Out[53]:

BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY
40.900749	0.818182	95.400000	0.000000	95.400000	0.000000	0.166667
3202.467416	0.909091	0.000000	0.000000	0.000000	6442.945483	0.000000
2495.148862	1.000000	773.170000	773.170000	0.000000	0.000000	1.000000
1666.670542	0.636364	1499.000000	1499.000000	0.000000	205.788017	0.083333
817.714335	1.000000	16.000000	16.000000	0.000000	0.000000	0.083333

Scaling the dataset:

Now that we are done with the imputation, we will now scale the dataset in order to manage the dataset variability and transform the data into a defined range using linear transformation to produce high-quality clusters by boosting the precision of our K-means model.

```
In [15]: from sklearn.preprocessing import StandardScaler
         scaler = StandardScaler()
         scaled_data = scaler.fit_transform(credit_card2)
         print("Standardized Data = ", scaled_data)
```

```
Standardized Data =  [[-0.73198937 -0.24943448 -0.42489974 ... -0.31172902 -0.52555097
   0.36067954]
 [ 0.78696085  0.13432467 -0.46955188 ...  0.08704715  0.2342269
   0.36067954]
 [ 0.44713513  0.51808382 -0.10766823 ... -0.10320983 -0.52555097
   0.36067954]
 ...
 [-0.7403981  -0.18547673 -0.40196519 ... -0.33613506  0.32919999
  -4.12276757]
 [-0.74517423 -0.18547673 -0.46955188 ... -0.34753313  0.32919999
  -4.12276757]
 [-0.57257511 -0.88903307  0.04214581 ... -0.33362544 -0.52555097
  -4.12276757]]
```

Measuring the Clustering Tendency using Hopkins Test:

The Hopkins Test is a measure used to assess the clustering tendency in a given dataset. It helps determine whether the data points in a dataset are randomly distributed, highly clustered, or uniformly distributed.

The test takes a random sample of points from the dataset, calculates the distances between each randomly selected point and its nearest neighbour among the remaining data points and later among all the points in the dataset. This process is repeated in several iterations till a stable estimate is achieved.

The statistic is supposed to be within 0 to 1. If the value generated is closer to 1, then the dataset is highly clustered, if the value generated is closer to 0.5, then the dataset is randomly distributed and if the value generated is closer to 0, then the dataset is uniformly distributed.

As per our test, we have achieved the score 0.9646.

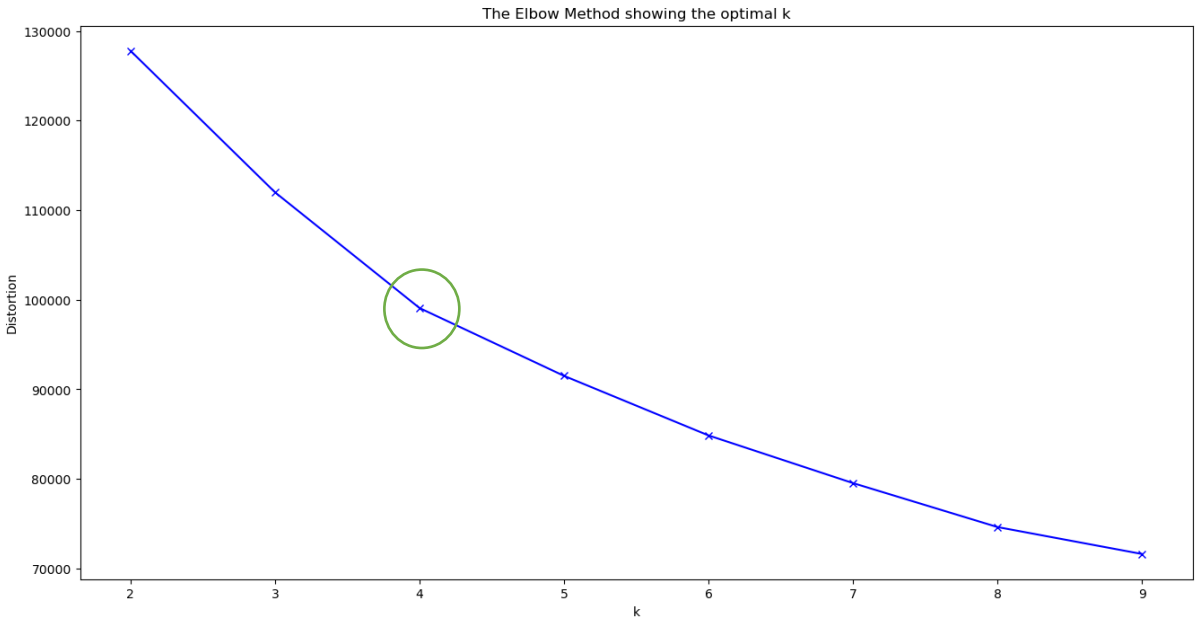
```
In [61]: l = [] #list to hold values for each call
         for i in range(20):
             H=hopkins_statistic(scaled_data)
             l.append(H)
         #print the estimate
         print('Estimated value from the Hopkins Test: \n',np.mean(l))
```

```
Estimated value from the Hopkins Test:
 0.964694566497234
```

As the above value is closer to 1, we can say that the data is high tendency to cluster.

Choosing the number of clusters using Elbow Plot:

Elbow Plot is used for finding the optimal number of clusters required to capture variation in the dataset. The plot visualizes the relationship between the number of clusters and the variability in the data.



The number of clusters is subjective. Generally, we choose the 'k' i.e., the number of clusters wherever the bend appears. Here, we are considering k as 4 due to the elbow appearing.

K-Means Model:

The K-means model is a popular model for clustering analysis where it partitions a dataset into k distinct clusters based on the mean (centroid) of the data points. The algorithm aims to minimize the within-cluster sum of squares between each point and its assigned centroid.

Here, we have already chosen k as 4.

```
In [23]: kmeansModel = KMeans(n_clusters=4)
         kmeansModel.fit(scaled_data)
         kmeansPredict= kmeansModel.predict(scaled_data)
```

Now, let us see how the dataset has divided into 4 clusters.

```
In [26]: print('Count in each cluster: \n', credit_card2['clust'].value_counts())
Count in each cluster:
 2    3976
1    3367
3    1198
0     409
Name: clust, dtype: int64
```

Evaluating Clustering Quality:

Davies-Bouldin Index: The Davies-Bouldin Index is a metric used for evaluating the quality of clustering results. It provides a measure of the average similarity between clusters and the dissimilarity between clusters. The lower the DBI value, the better the clustering performance.

```
In [27]: from sklearn.metrics import davies_bouldin_score
         db_index = davies_bouldin_score(scaled_data, kmeansModel.labels_)
         print('Davies-Bouldin Index:',db_index)

         Davies-Bouldin Index: 1.5756611165323287
```

As per the index value generated, it suggests that the clusters exhibit a reasonable level of separation and compactness.

Silhouette Score: Silhouette Score is another method of calculating the quality of clustering results. It considers both cohesion within the clusters and separation among the clusters and gives a measure of how well each data point fits in the assigned cluster.

```
In [28]: from sklearn.metrics import silhouette_score
         ss_score = silhouette_score(scaled_data, kmeansModel.labels_)
         print('Silhouette Score:',ss_score)

         Silhouette Score: 0.19749317075203252
```

As per the score generated, it suggests a moderate level of clustering quality. The score is positive, indicating that, on average, the data points are reasonably well-clustered and have a certain level of separation from neighbouring clusters.

Calinski Harabasz Index: Calinski Harabasz Index is also a metric used for evaluating cluster quality. It provides a measure of the separation between clusters and the compactness within clusters. The higher the value, the better the clustering performance.

```
In [71]: from sklearn.metrics import calinski_harabasz_score
         ch_index = calinski_harabasz_score (scaled_data, kmeansModel.labels_)
         print('Calinski Harabasz Index:',ch_index)

         Calinski Harabasz Index: 1597.3989595044886
```

As per the index generated, it suggests that there was a relatively clustering performance and there was a better separation between clusters and the compactness within clusters.

Linear Discriminant Analysis:

We will now use LDA to make a comparison of results from K-means clustering and see if the variables chosen are helpful in clustering the dataset.

```
X = credit_card2.iloc[:, :-1]
y = credit_card2.iloc[:, -1]
```

For the LDA, we will separate the clusters from the dataset (splitting into X and y) and try to use the values again to predict the clusters again and generate the accuracy.

```
In [35]: print(lda.coef_)

[[ 2.98568080e-04  1.36016940e+00 -1.37440532e-02  1.55945078e-02
   1.61004790e-02 -3.46306750e-04 -2.38026491e+00  4.01831558e+00
   5.70123000e+00 -1.92685063e+00  5.60995367e-04  1.65713407e-01
   1.62234192e-04  1.05118537e-04  4.10826014e-05  1.62687532e+00
  -2.56557186e-01]
 [-2.26872572e-04  2.17238157e+00  3.73122376e-03 -3.76286401e-03
  -3.76222462e-03 -1.12256236e-04  8.35074523e+00 -1.23308978e-01
   1.42781581e+00 -2.16690191e+00 -3.36539870e-02 -1.06767314e-02
   6.99532002e-06  1.48363693e-05 -3.00812991e-06  9.77675145e-01
   5.51133076e-03]
 [-1.17253881e-04 -1.47462415e+00 -4.82235455e-04  3.61725264e-04
   3.87775832e-04 -1.14095954e-04 -6.56771982e+00  9.58077324e-02
  -1.24772277e+00 -1.60322517e+00 -8.16270123e-03 -3.27304457e-03
  -8.08648016e-05  4.73231565e-06 -2.59496440e-05 -1.39269199e+00
  -5.09239305e-03]
 [ 9.24847278e-04 -1.67580336e+00 -4.19394365e-03  4.05107661e-03
   3.79008154e-03  8.12397097e-04 -8.59913882e-01 -1.34327319e+00
  -1.81829146e+00  1.20688346e+01  1.21484497e-01 -1.57050114e-02
   1.93331739e-04 -9.32915058e-05  8.05520652e-05  1.31896421e+00
   8.90004952e-02]]
```

After training the dataset, these are the coefficients generated.

```
In [37]: accuracy = np.mean(predictions == y)
         print("Accuracy: ", accuracy)

         Accuracy:  0.9427932960893854
```

Based on the prediction accuracy generated for the model, we can say there was a high level of classification performance, indicating there was a high proportion of correctly classified instances in the dataset.

Hence, we can say that the classification of values in the dataset was done accurately.

Analysis of Customer Segments based on the clusters generated:

Summarizing the Clusters:

Column Name	Metrics	0	1	2	3	Overall
BALANCE	mean	3551.153761	894.907458	1011.751528	4602.462714	1564.474828
BALANCE_FREQUENCY	mean	0.986879	0.934734	0.789871	0.968415	0.877271
PURCHASES	mean	7681.620098	1236.178934	269.973466	501.896219	1003.204834
ONEOFF_PURCHASES	mean	5095.878826	593.974874	209.853863	320.373681	592.437371
INSTALLMENTS_PURCHASES	mean	2587.208264	642.478274	60.386625	181.607404	411.067645
CASH_ADVANCE	mean	653.638891	210.570626	595.759339	4520.724309	978.871112
PURCHASES_FREQUENCY	mean	0.946418	0.885165	0.170146	0.287731	0.490351
ONEOFF_PURCHASES_FREQUENCY	mean	0.739031	0.297070	0.086281	0.138934	0.202458
PURCHASES_INSTALLMENTS_FREQUENCY	mean	0.788060	0.711842	0.080578	0.185516	0.364437
CASH_ADVANCE_FREQUENCY	mean	0.071290	0.042573	0.114833	0.484526	0.135144
CASH_ADVANCE_TRX	mean	2.085575	0.790021	2.125503	14.284641	3.248827
PURCHASES_TRX	mean	89.359413	22.091773	2.903421	7.661102	14.709832
CREDIT_LIMIT	mean	9696.943765	4213.207678	3277.352448	7546.957050	4494.293646
PAYMENTS	mean	7288.739497	1332.194205	974.505090	3481.145990	1733.143852
MINIMUM_PAYMENTS	mean	1975.664034	658.158925	587.722303	2015.160519	868.716633
PRC_FULL_PAYMENT	mean	0.286707	0.269258	0.078001	0.034859	0.153715
TENURE	mean	11.951100	11.594595	11.446429	11.387312	11.517318

CLUSTER 0 (HIGH END): In this cluster, we have high-end customers who frequently use their bank's credit card. Their purchase frequency, payments and balance have a pretty high mean strongly supporting our statement. A higher credit limit indicates that the customers here have higher purchasing power, and this is granted by the credit card company only for people with a good credit history.

For any retail company, these set of customers are gold standard and they can spend a good amount of advertising money on. The company can build tailored marketing campaigns, promote exclusive offers and rewards, give personalized recommendations, personalized customer service, partner with the credit card companies. They can factorize on convenience, quality, and luxury during these promotions.

CLUSTER 1 (INSTALMENT): This cluster consists of customers who use credit cards for instalments as we can note from high level of transactions using instalments. Additionally, they frequently engage in transactions involving substantial amounts, while cash advances are relatively infrequent. Payments and cash advances are rare in this cluster, with customers demonstrating a lower frequency and smaller amounts associated with such transactions.

For this hypothetical retail company to target these customers, they need to arrange loyalty programs, promote instalment plans where they emphasize the convenience and flexibility of paying in instalments, focus on showcasing products or services that align with the customers' preference for substantial transactions, create loyalty programs which rewards customers for utilizing instalment options.

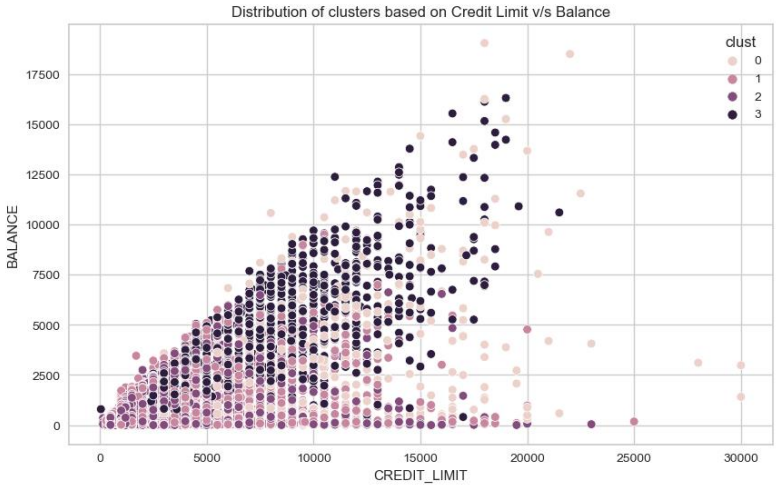
CLUSTER 2 (NEW USER): This cluster consists of credit card users who rarely use credit cards for transactions and instalments. A small balance with infrequent changes indicates a conservative usage pattern. A low credit limit suggests this even further. Cash advances are also rare among these customers. The low balance suggests that customers in this cluster may include new users who are gradually exploring credit card usage with this particular bank.

The retail company can target this cluster by creating any educational material and resources to understand the benefits and responsible usage of credit cards while shopping as well as provide information on how credit cards can be utilized for transactions and instalment options, along with tips for managing balances and making payments. They can give credit building offers and provide opportunities for them to establish a positive credit history, provide simplified payment processes for all transactions within the retail company branches.

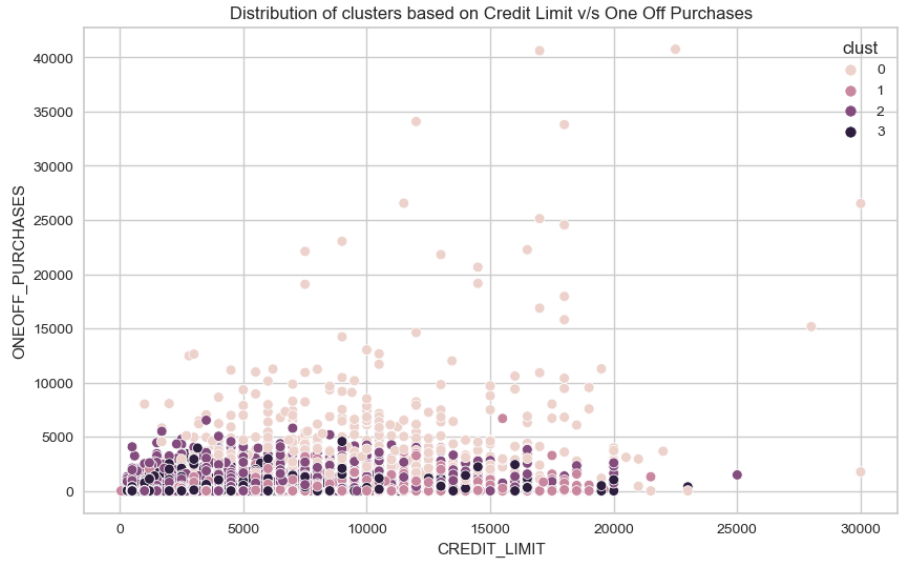
CLUSTER 3 (ADVANCES): In this cluster, Customers exhibit high balances with frequent changes, indicating active credit card usage. They have a high frequency of cash advances and the lowest interest rates compared to other clusters. Additionally, they possess the second-highest credit limit and make regular payments. However, customers in this cluster tend to avoid making instalment or one-off purchases, and their tenure ranks third among the four clusters. Therefore, it can be inferred that customers in this cluster primarily utilize credit cards for cash withdrawal or cash advance purposes.

The retail company can target the customers in this cluster, by giving cash advance promotions offer exclusive discounts, rewards, or additional benefits for using credit cards to withdraw cash for their purchases, ensure that cash advance services are easily accessible and convenient within the retail store and provide rewards, discounts, or cashback offers for their frequent cash advance transactions within the retail store.

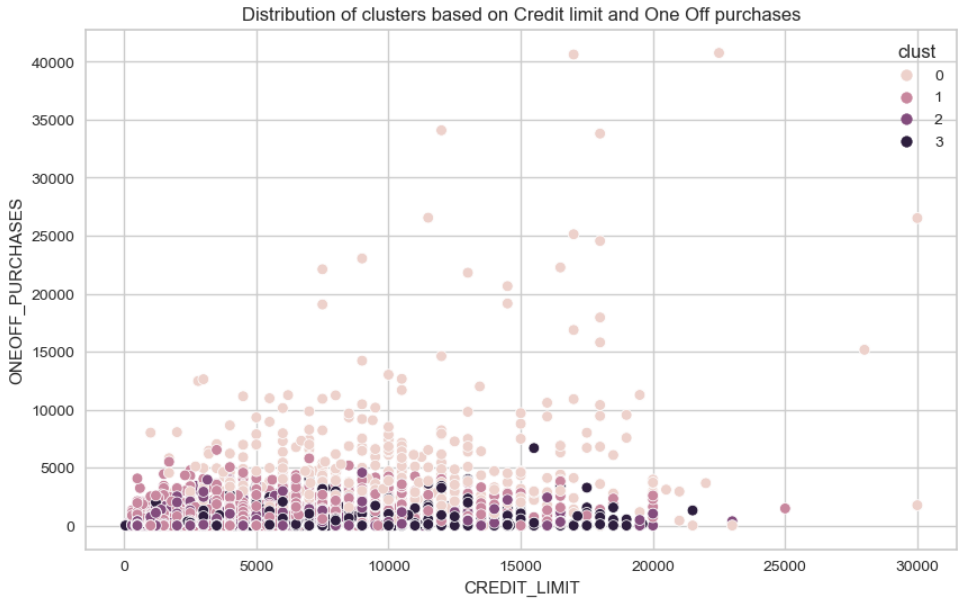
Understanding the clusters even further using visualizations:



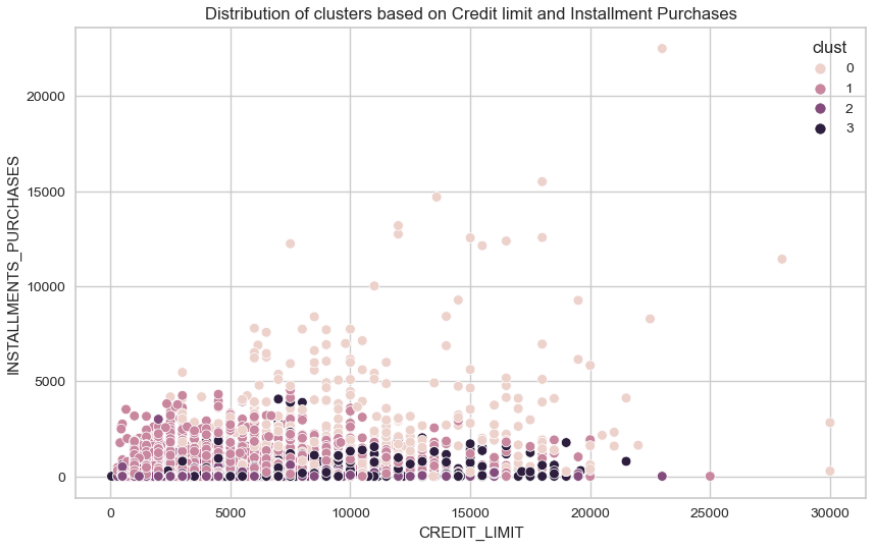
From the plot above we can infer that more the balance increases, the more credit limits the customer gets. From the figure above, it can be seen that clusters 0 and 3 have the highest balance and credit limit. Cluster 1 and 2 have the lowest balance card holders.



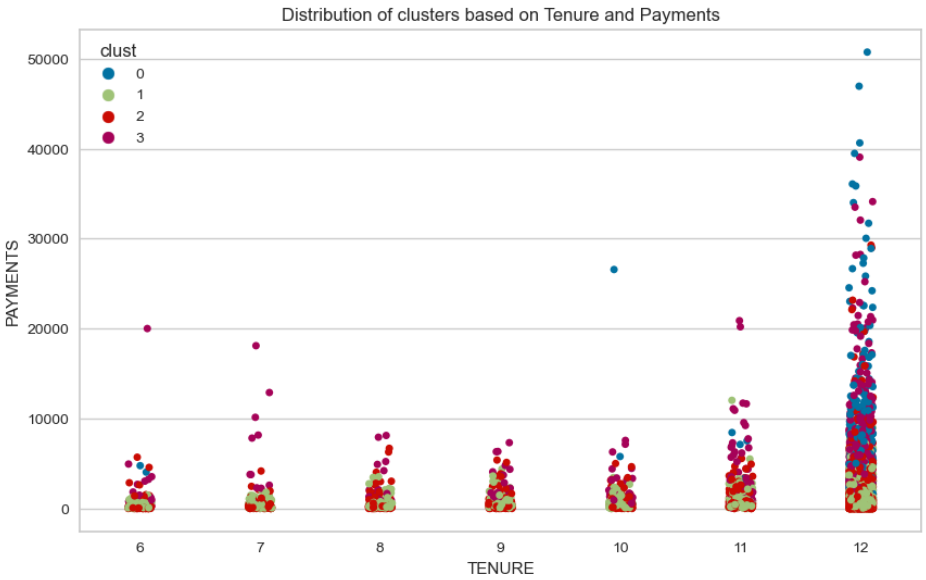
Based on the plot above, we can say cluster 0 has highest one-off purchase transactions. One-off purchase does not affect the additional credit limit obtained by the user.



Cluster 0 has more one-off purchases than the other clusters, but there is no correlation between Credit limit and one-off purchases.



The cluster 0 has the most instalment purchases. Large number of instalment purchases are not correlated with the credit limit increase.



Most customers in cluster 0 do not have 0 transactions and often prefer 12-month tenure. Cluster 1 and 2 have lesser payments, irrespective of tenure.

Conclusion:

Our report aimed to assist a hypothetical company in understanding and targeting its customers through customer segmentation. By utilizing cluster analysis and discriminant analysis, we divided the customers into distinct segments, enabling the company to customize its marketing strategies and offerings to cater to the unique needs of each segment. In order to achieve fairness, we implemented various tests to confident about our results.

Through basic analysis, data pre-processing, and the application of K-means clustering and LDA (Linear Discriminant Analysis), we successfully generated customer segments. These segments were then analyzed and summarized based on their specific characteristics and behaviour patterns. We provided marketing recommendations for each segment, outlining effective strategies to target and engage customers. By implementing these recommendations, this hypothetical company can optimize its resources, reduce customer churn, and ultimately enhance customer satisfaction and loyalty.