Project Name – Credit Card Segmentation

Submitted By:

Prakhar Acharya

Content:

Problem Statement

In this case we are required to develop a customer segmentation to define marketing strategy.
The sample dataset summarizes the usage behavior of about 9000 active credit card holders during the last 6 months. The file is at a customer level with 18 behavioral variables.

The objective is segment the data to help us define a marketing strategy.

General Information of Data

CUST_ID: Credit card holder ID
BALANCE: Monthly average balance (based on daily balance averages)
BALANCE_FREQUENCY: Ratio of last 12 months with balance
PURCHASES: Total purchase amount spent during last 12 months
ONEOFF_PURCHASES: Total amount of one-off purchases
INSTALLMENTS_PURCHASES: Total amount of installment purchases
CASH_ADVANCE :Total cash-advance amount
PURCHASES_ FREQUENCY:Frequency of purchases (percentage of months with at least on purchase)
ONEOFF_PURCHASES_FREQUENCY: Frequency of one-off-purchases
PURCHASES_INSTALLMENTS_FREQUENCY: Frequency of installment purchases
CASH_ADVANCE_ FREQUENCY :Cash-Advance frequency
AVERAGE_PURCHASE_TRX :Average amount per purchase transaction
CASH_ADVANCE_TRX :Average amount per cash-advance transaction
PURCHASES_TRX :Average amount per purchase transaction
CREDIT_LIMIT :Credit limit
PAYMENTS:Total payments (due amount paid by the customer to decrease their statement balance) in the period
MINIMUM_PAYMENTS: Total minimum payments due in the period.
PRC_FULL_PAYMENT: Percentage of months with full payment of the due statement balance
TENURE :Number of months as a customer

Procedure

Firstly, we will import our data and explore the data. We will then investigate if there are any missing values and make a decision on how we move forward with these. Next, we will see what correlations we have between variables. Following this we will then split our data into a training and test set. We will then use the K-means algorithm to classify our data, using the Elbow method to find the optimal number of k clusters. Finally, we will test our results using the test set. In python we will also try some different methods like spectral as well to get to understand more about the data

Data Understanding

```
CUST_ID         BALANCE         BALANCE_FREQUENCY   PURCHASES         ON
 C10001 :   1   Min.    :    0.0   Min.   :0.0000    Min.    :    0.00
 C10002 :   1   1st Qu.:  128.3   1st Qu.:0.8889    1st Qu.:   39.63
 C10003 :   1   Median :  873.4   Median :1.0000    Median :  361.28
 C10004 :   1   Mean   : 1564.5   Mean   :0.8773    Mean   : 1003.20
 C10005 :   1   3rd Qu.: 2054.1   3rd Qu.:1.0000    3rd Qu.: 1110.13
 C10006 :   1   Max.   :19043.1   Max.   :1.0000    Max.   :49039.57
```

The above is a snippet of the data structure.
On analyzing the data, we understand that at least one customer has spent about 40k in one go. This relates to the fact that there may be outliers in our data, something we will have to manage at the coming stage of our process.

The summary function helps us to understand a bit more about the data, the average balance for example, how much the data varies, the spread and more.

```
CUST_ID    BALANCE BALANCE_FREQUENCY PURCHASES ONEOFF_PURCHASES INSTA
1  C10001   40.90075          0.818182     95.40             0.00
2  C10002 3202.46742          0.909091      0.00             0.00
3  C10003 2495.14886          1.000000    773.17           773.17
4  C10004 1666.67054          0.636364   1499.00          1499.00
5  C10005  817.71434          1.000000     16.00            16.00
  PURCHASES_FREQUENCY ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMEN
1            0.166667                  0.000000
2            0.000000                  0.000000
3            1.000000                  1.000000
4            0.083333                  0.083333
5            0.083333                  0.083333
```

The above is a peek into the data.

Missing Value:

Now let's see if there are any missing values in the given data.

```
MINIMUM_PAYMENTS      313
CREDIT_LIMIT            1
TENURE                  0
PURCHASES_FREQUENCY     0
BALANCE_FREQUENCY       0
```

Clearly there are some missing values, and in fact. Minimum_Payments has 313 NA's in total.

Similarly, the credit limit has one missing value.
For now I'm going to use the median to replace these values.

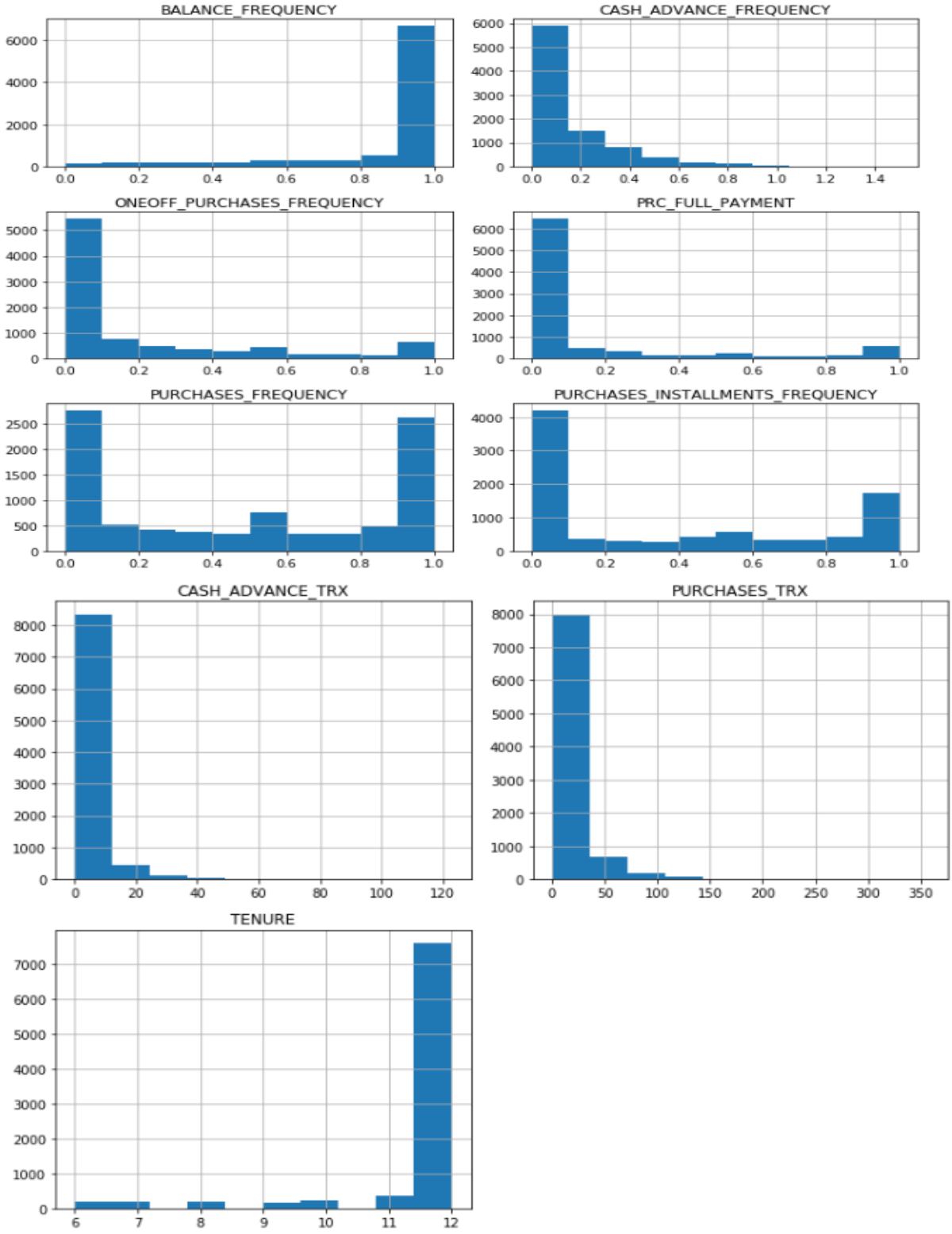
Correlation

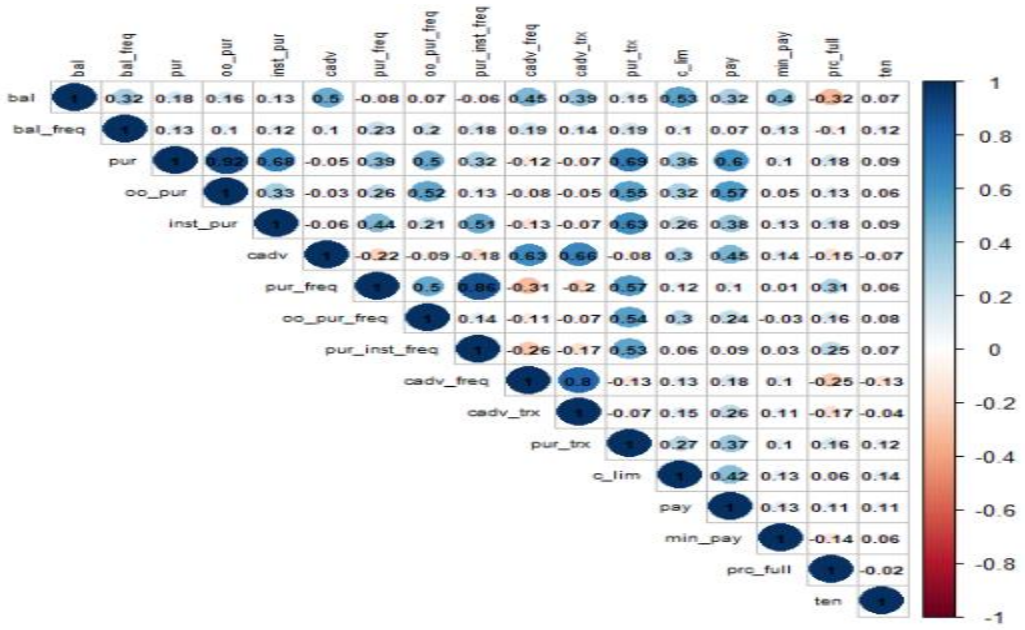
To perform correlation we need our variables to be numeric. First, let's see what the data types are of our variables.

```
$ CUST_ID                          : Factor w/ 8950 levels "C10001","
$ BALANCE                          : num  40.9 3202.5 2495.1 1666.7 8
$ BALANCE_FREQUENCY                : num  0.818 0.909 1 0.636 1 ...
$ PURCHASES                        : num  95.4 0 773.2 1499 16 ...
$ ONEOFF_PURCHASES                 : num  0 0 773 1499 16 ...
$ INSTALLMENTS_PURCHASES           : num  95.4 0 0 0 0 ...
$ CASH_ADVANCE                     : num  0 6443 0 206 0 ...
$ PURCHASES_FREQUENCY              : num  0.1667 0 1 0.0833 0.0833 ..
$ ONEOFF_PURCHASES_FREQUENCY       : num  0 0 1 0.0833 0.0833 ...
$ PURCHASES_INSTALLMENTS_FREQUENCY: num  0.0833 0 0 0 0 ...
$ CASH_ADVANCE_FREQUENCY           : num  0 0.25 0 0.0833 0 ...
$ CASH_ADVANCE_TRX                 : int  0 4 0 1 0 0 0 0 0 0 ...
$ PURCHASES_TRX                    : int  2 0 12 1 1 8 64 12 5 3 ...
$ CREDIT_LIMIT                     : num  1000 7000 7500 7500 1200 18
$ PAYMENTS                         : num  202 4103 622 0 678 ...
$ MINIMUM_PAYMENTS                 : num  140 1072 627 312 245 ...
$ PRC_FULL_PAYMENT                 : num  0 0.222 0 0 0 ...
$ TENURE                           : int  12 12 12 12 12 12 12 12 12
```

We can drop CUST_ID , as the variable is not very useful in our dataset.

In Python we have visualized the data by using histogram.





The correlogram or correlation matrix helps us to understand the correlation between variables in our data set.

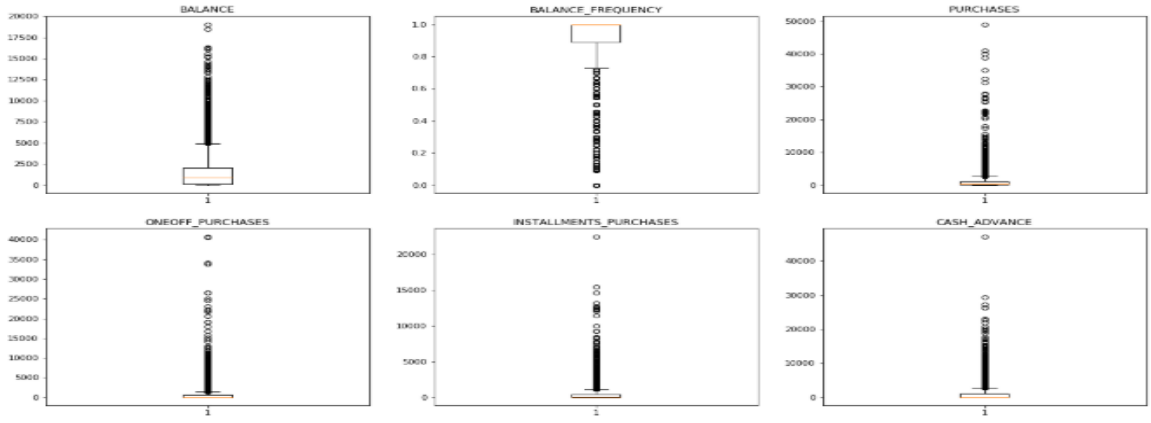
Key findings:

– The more purchases a customer makes, the more likely they will have had a larger one off purchase.

– Customers with higher credit balances are more likely to have a higher credit limit and also have more cash advances.

– Customers who make more purchases also make more payments.

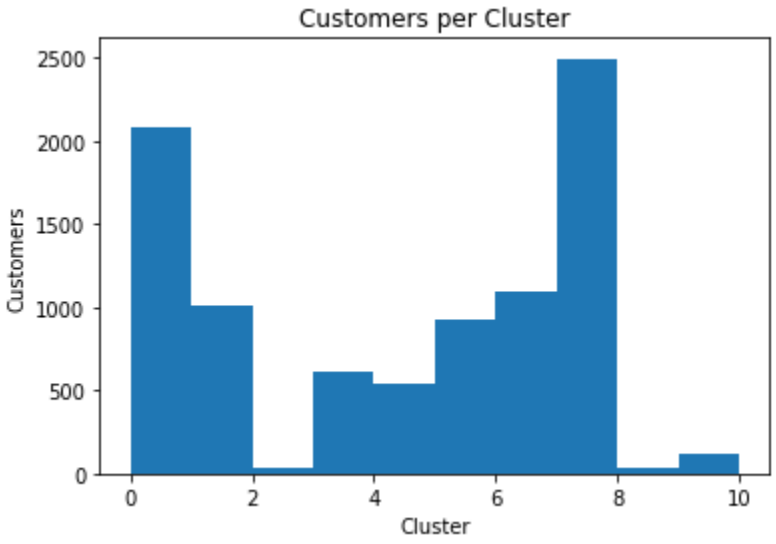
Overview of Outliers



The above is a snippet of the boxplot graph for overview of outliers.

Clustering:

It is the assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense. Clustering is a method of unsupervised learning, and a common technique for statistical data analysis used in many fields.



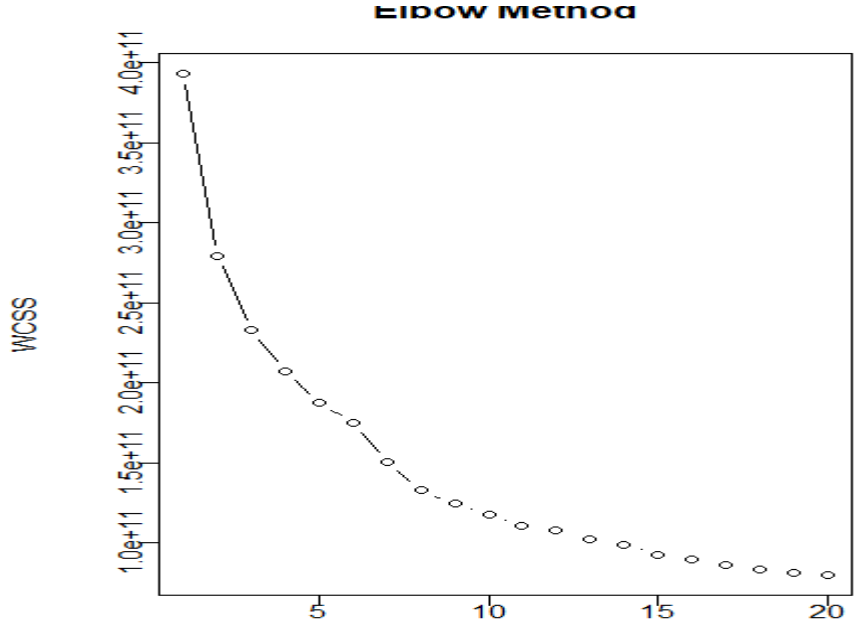
By the above graph, we can analyze the customers per Cluster and now we will proceed with the methods of clustering.

Finding optimal number of clusters, k

Here we use the elbow method, essentially, we calculate the within-cluster-sum-of-squares (WCSS), iterating k through 1 to 20.

In R:

```
set.seed(123)
WCSS <- vector()
for (i in 1:20) WCSS[i] <- sum(kmeans(ccdata, i)$withins)
plot(1:20, WCSS, type = "b", main = "Elbow Method", xlab = "Clusters", ylab = "WCSS")
```



In Python:

```
wcss = []
cluster_list = range(1, 11)
for i in cluster_list :
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_i
nit = 10, random_state = 40)
    kmeans.fit(df_std)
    wcss.append(kmeans.inertia_)
```

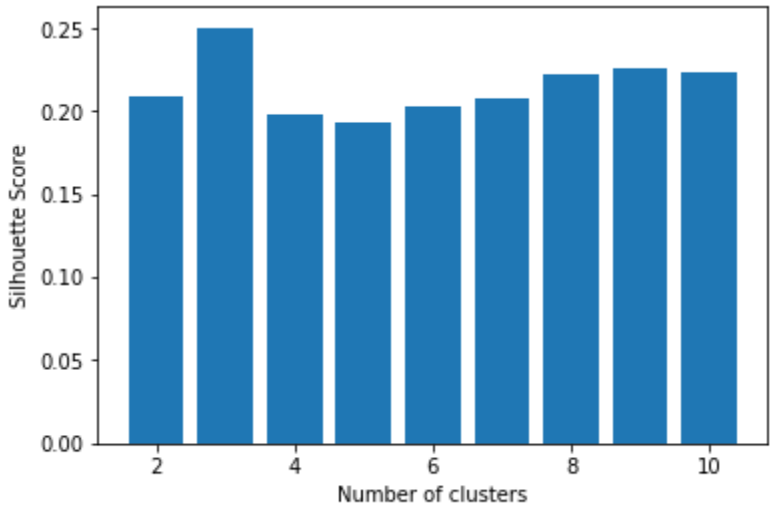
```
plt.plot(cluster_list, wcss)
plt.title('Elbow Method')
plt.xlabel('Clusters')
plt.ylabel('WCSS')
plt.show()
```

We will also determine number of Clusters with Silhouette Scores Method

```
silhouette_scores = []

for n_cluster in range(2, 11):
    silhouette_scores.append(
        silhouette_score(df_std, KMeans(n_clusters = n_cluster).fit_predict
(df_std)))

# Plotting a bar graph to compare the results
k = [2, 3, 4, 5, 6,7,8,9,10]
plt.bar(k, silhouette_scores)
plt.xlabel('Number of clusters', fontsize = 10)
plt.ylabel('Silhouette Score', fontsize = 10)
plt.show()
```



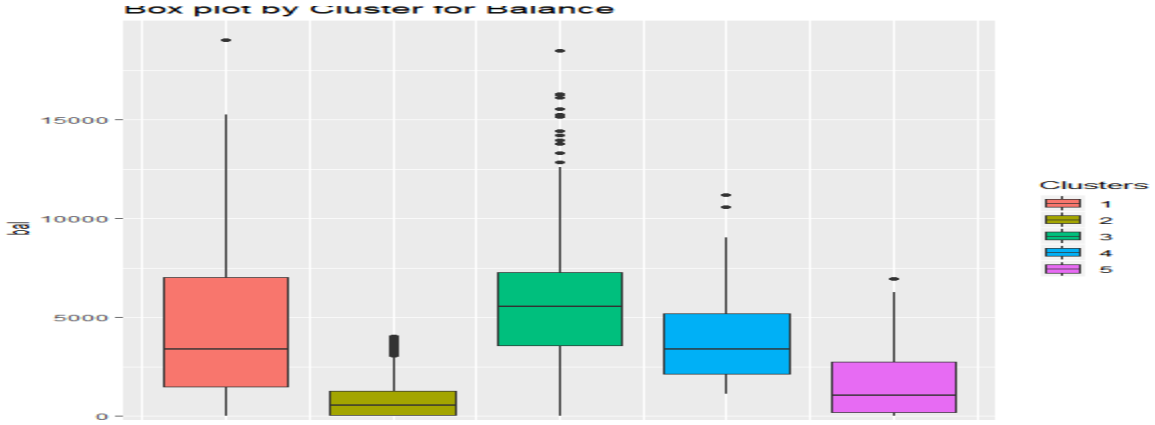
Kmeans Clustering:

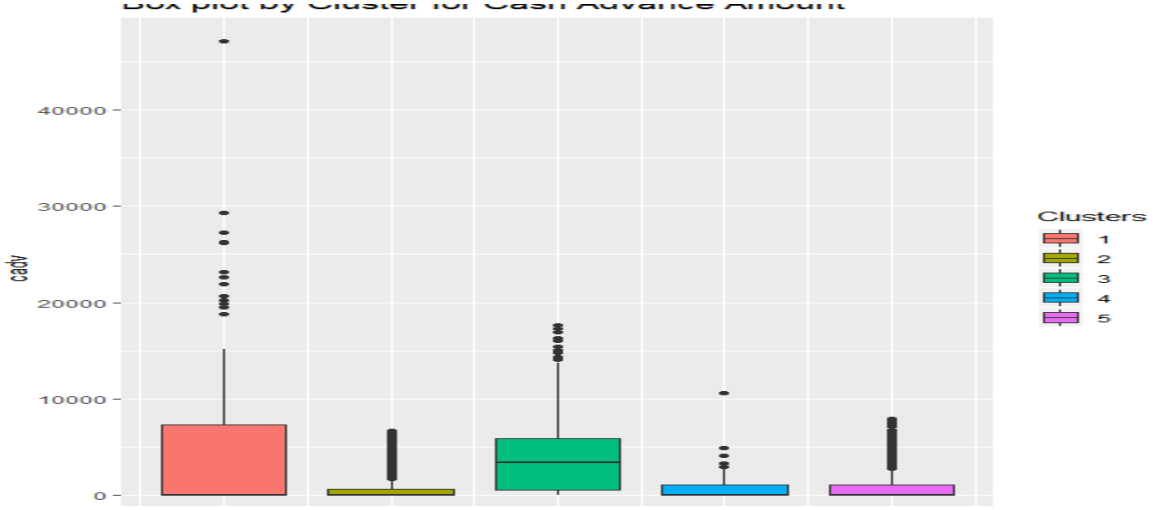
In R:

For the purpose of just exploring the K-means clustering algorithm, I'm going to choose 5 and then interpret the results. We can now save our clusters in our ccdata data set and can also count the number of records in each cluster. Some of our clusters have a very small number of customers in, for example, cluster 1 and cluster 4 make up less than 2% of our sample.

```
kmeans <- kmeans(ccdata, 5, iter.max = 300, nstart = 10)
ccdata <- data.frame(ccdata,kmeans$cluster)
ccdata %>% group_by(ccdata$kmeans.cluster) %>% summarise(n())
```

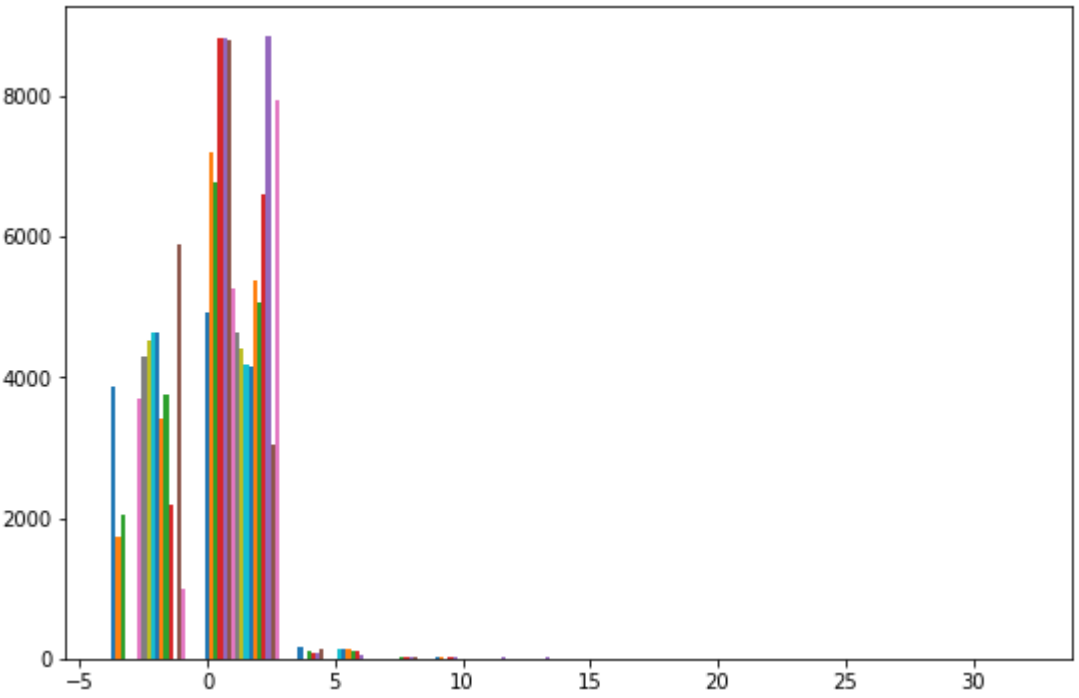
Using boxplots, we can interpret the results. Boxplots is going to tell us some interesting things, such as how spread the data is, the median and outliers in the data too.

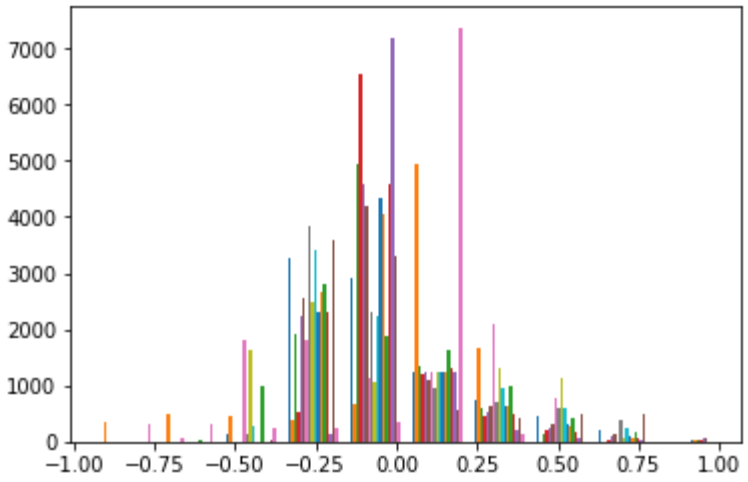




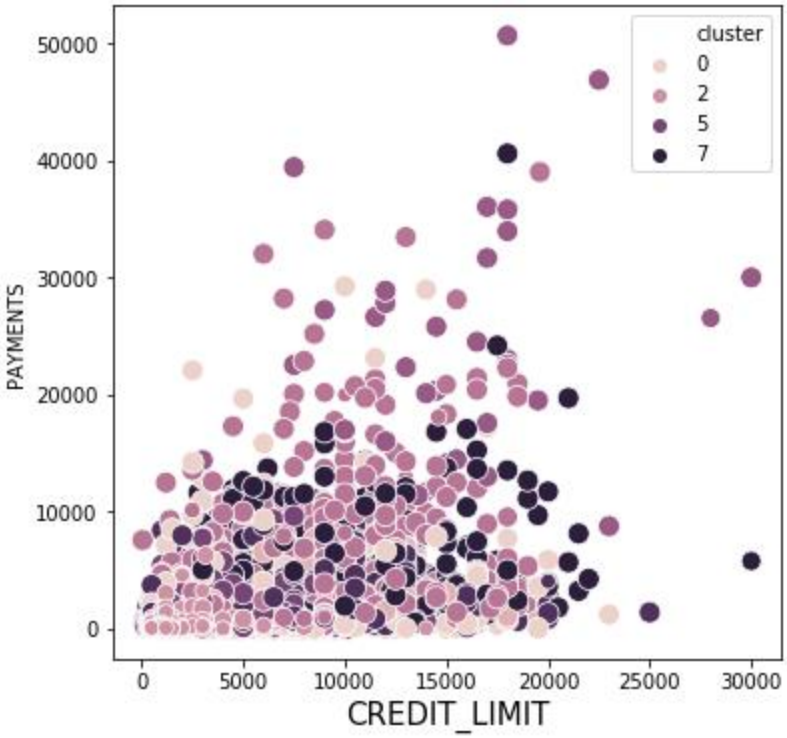
Box Plot by cluster for Cash Advance Amount

In Python I tried exploring Spectral clustering method.

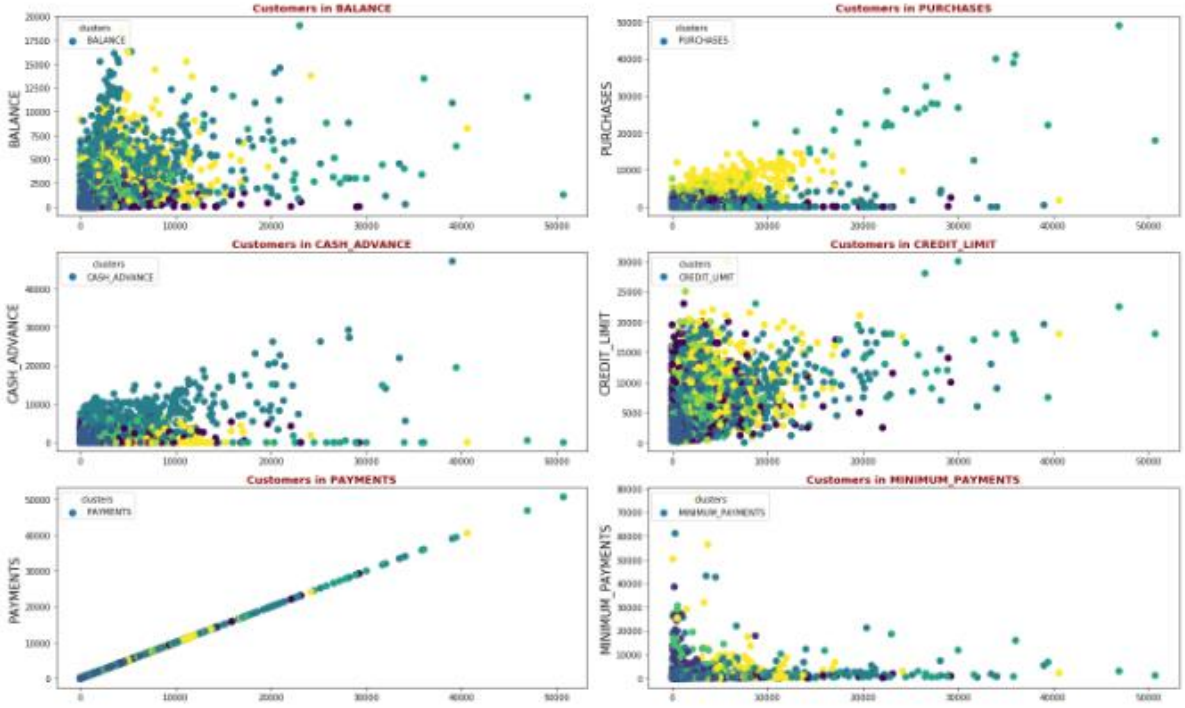




After analyzing silhouette score, we will now go for Kmeans method.



We can analyze the data now through scatter plot:



This study aims to understand the customer behaviours based on credit card users.

- Large Payments are done by a small group with expensive purchases and a credit limit that is between average and high.
- Small group of people have a higher amount of Cash Advance* especially after payments of 30.000. Large group of people have a lower cash limit contrats large payments.
- *Credit Limit is very low on a large group of customers with little purchases.

Conclusion:

I have tried implementing by some different methods in R and Python to analyze and learn more, I implemented spectral clustering in Python to get more depth understanding.

- Cluster 1: High user of credit card, likely has high income as balance isn't that high but spends quite a lot.
- Cluster 2: Mid user of credit card, but only buys low value goods.
- Cluster 3: Mid user of credit card, tendency to buy large value goods but not great at paying it back as still high balance.
- Cluster 4: Mid user of credit card, tends to buy low value goods but not great at paying back.
- Cluster 5: High user of credit card, however doesn't purchase goods that are high in value. Keeps balance low.