Credit Card Segmentation – Case Study

**This case requires trainees to develop a customer segmentation to define marketing strategy. The sample dataset summarizes the usage behaviour of about 9000 active credit card holders during the last 6 months. The file is at a customer level with 18 behavioural variables.**

**The following case study is completed in both R and Python. This document is explained in R.**

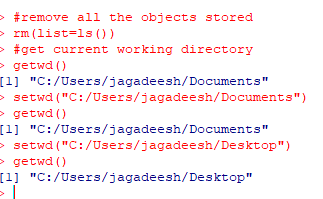
**Exploratory Data Analysis:**

Exploratory Data Analysis (EDA) is an approach to data analysis that postpones the usual assumptions about what kind of model the data follow with the more direct approach of allowing the data itself to reveal its underlying structure and model.

For analysis of data and to perform exploratory data analysis, we have to read the data. Finally, we will go through the input data to gain necessary insights from the data.

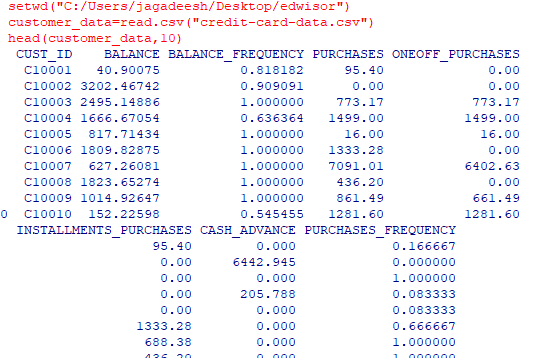
Data from CSV files can be read into R environment using the read.csv() function which is available in the base R package.

**CODE:**



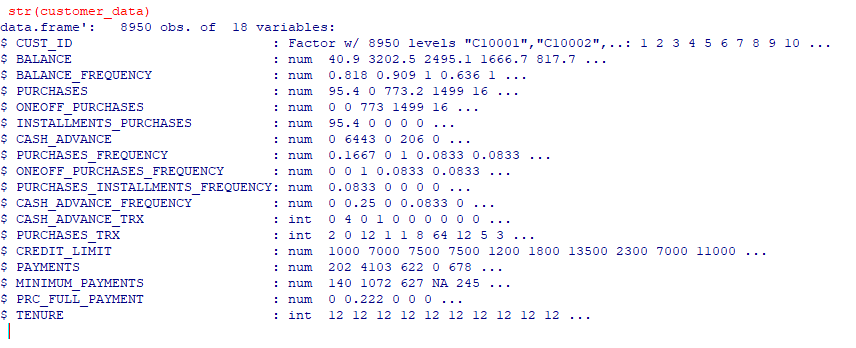
Now after setting my working directory, I stored my csv file on the working directory.

**CODE:**

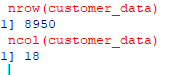


Here, the data present in the csv file has been stored in the variable **customer\_data.** The command **head(customer\_data,10)** gives us the top 10 rows data information for all variables.

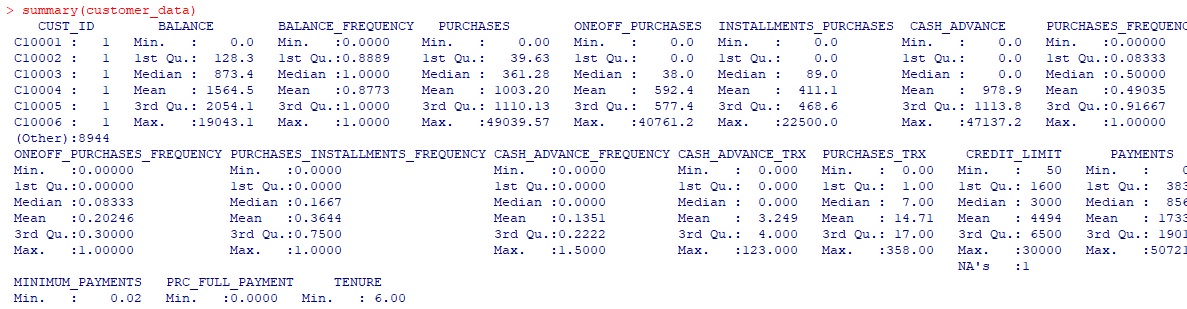
Now the structure of the data can be viewed as below:



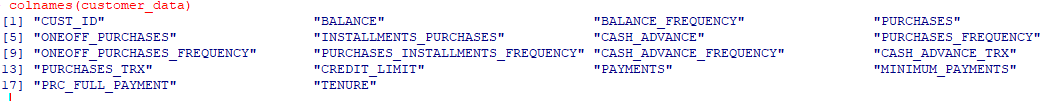
And the data contains 8950 rows and all the observations are distributed to 18 columns.



And to get summary of it, we use summary() function in R.



The list of columns available in R can be viewed as follows:



To build an enriched customer profile by deriving ‘intelligent KPI’s such as monthly average purchase and cash advance amount, the data present is not consistent. There are a lot more missing values for each of the columns and we need to perform data cleaning to all columns. This data cleaning performed is not same for all type of variables. Data present here is divided into nominal, ordinal and ratio, data cleaning methods is different for each type of variable.

**DATA CLEANING:**

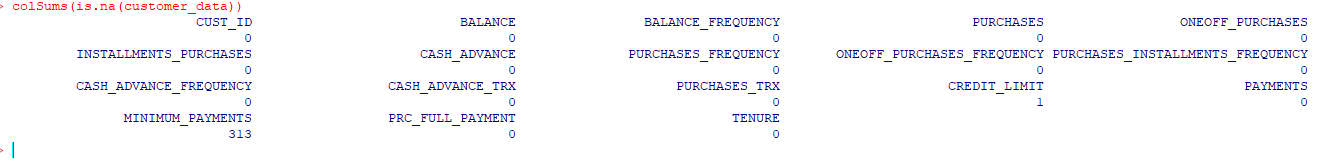
Data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a data set.

By performing Data cleaning, raw data is converted to consistent data which is ready for analysis.

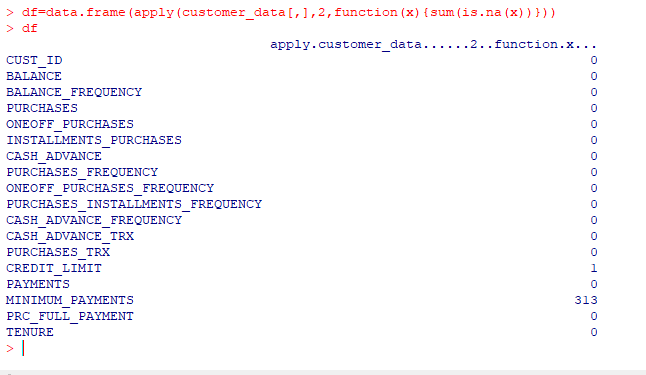
Raw data is processed and is made technically correct before being imported to R. This involves correcting data types, labels, headers etc.

Once the technically correct data is imported into R, it has to be cleaned in order to obtain consistent data. This involves checking, whether the data is correct with respect to the domain constraints. For Ex: Having a negative value in Age column or having an entry in the date of the marriage field for a person whose relationship status reads single. There can also be the presence of NA values, which has to deal with care.

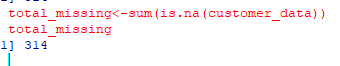
By using colSums(is.na(customer\_data)), I was able to find the sum of na values in each column



Data Cleaning in R can also be done in many ways and I tried the same using apply command to fetch the output

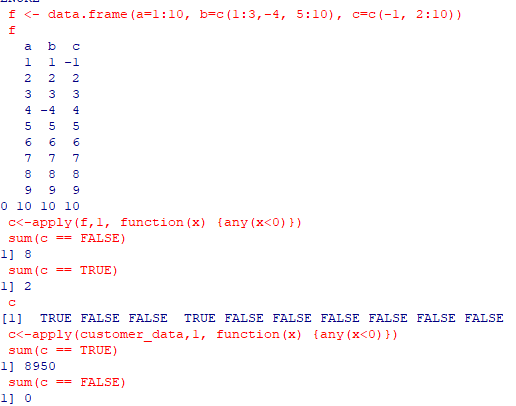


From the output screen shot, we can see that **minimum\_payments** has more null values followed by credit\_limit which has 1 null value. And now, let us calculate total missing values in the dataset customer\_data.

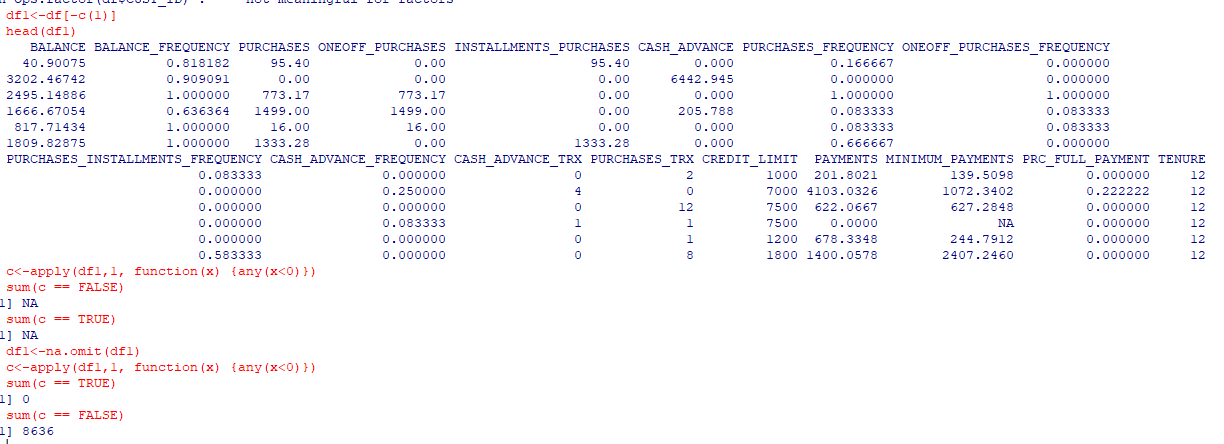


From the screen shot, we can clearly visualize that there are a total of 314 missing values in which 313 missing values are from **payments** columns and 1 missing value is from **credit\_limit**.

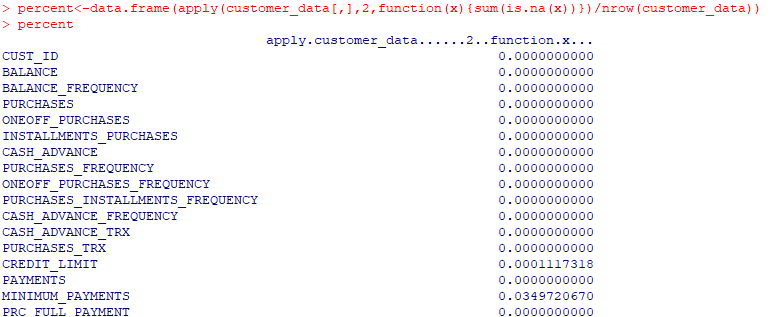
So, let us check if there are any negative values in the data.



From the output screenshot, we could find that there are no negative values in the data and all the 8950 rows contains positive values. Any(x<0) takes each row as a parameter and returns true if there is any negative value in particular row else it returns false. Here, Customer id is a factor variable and is to be excluded. So, I used na.omit() function to remove NA values as the output for this data frame will results to NA if not omitted. Now, I could clearly confirm that there is no negative values in the data. The below screenshot explains the output:

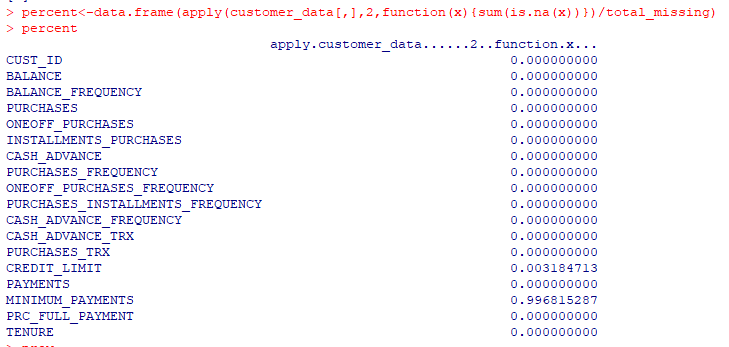


**CALCUATING TOTAL PERCENT OF MISSING VALUES IN DATA:**

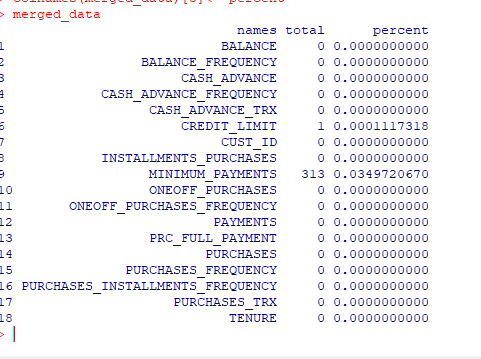


There are a total of 8950 rows in which 313 rows contains missing values from minimum payments and credit\_limit occupies a percent of 0.0001 among total rows.

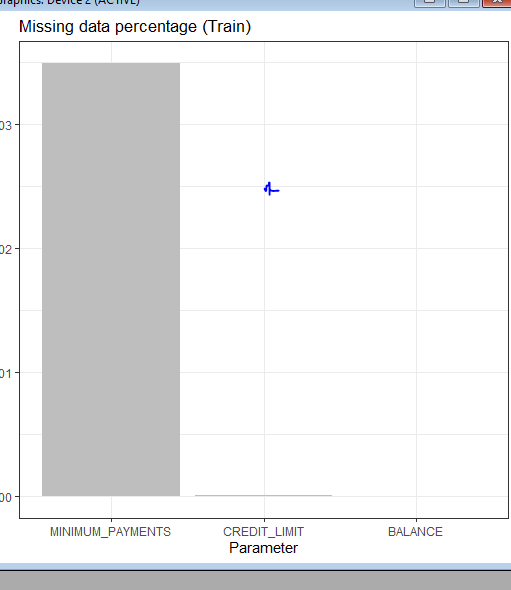
From the below output screenshot, we can clearly see that among the total missing values, 99.3% are from minimum payments.



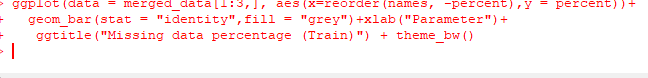
Now, let’s have a comparison of total number of missing values and percentage it is occupying among the total number of values.



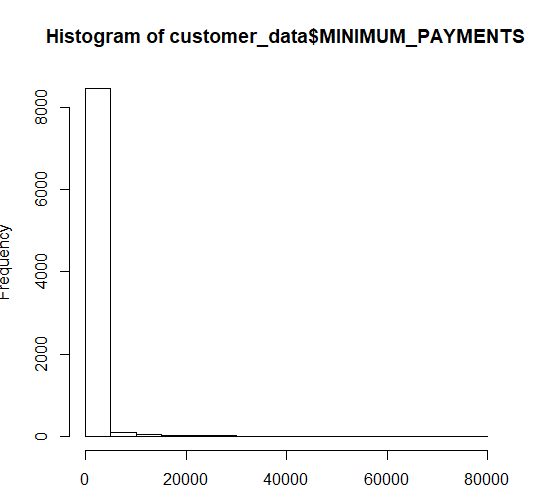
**VISULAIZATION OF DATA IN TERMS OF PERCENTAGE:**



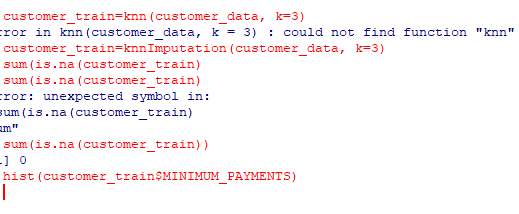
We used ggplot library to do so.



So, now missing values can be imputed if it occupies not less than 30% of total values. So, here let’s impute minimum payments column with mean or median. In order to do so, let us visualize the data column and see if it is normally distributed or not.

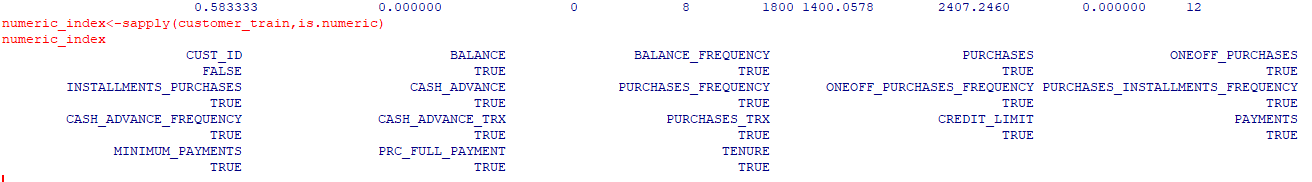


From this we can clearly state that data is not normally distributed and mean imputation is not the right method and using KNN imputation method to impute the missing values.

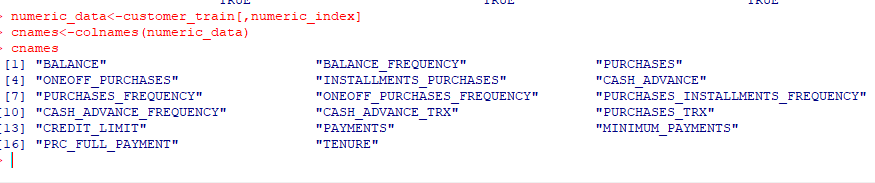


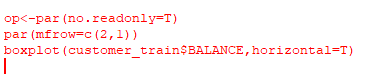
**OUTLIER ANALYSIS:**

In order to do outlier analysis, let us convert all the factors i.e categorical variables into labels so that it will be convenient to do outlier analysis For example, for a particular variable we have four levels namely high, low, medium and average. Let’s assign 1,2,3,4 as labels to them. In our data there are no outliers and hence proceeding to outlier analysis.

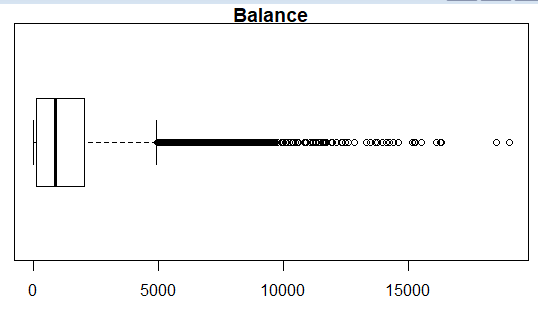


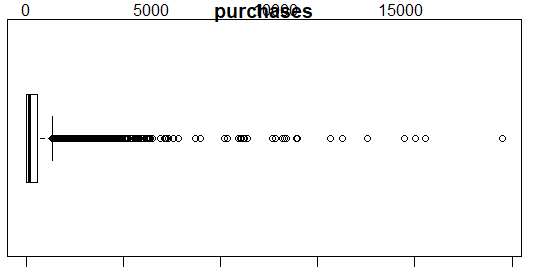
By applying sapply function and is.numeric, we can clearly visulaize that all the variables are numeric except custi\_id as there are no levels associated with them skipping this variable for outlier analysis.



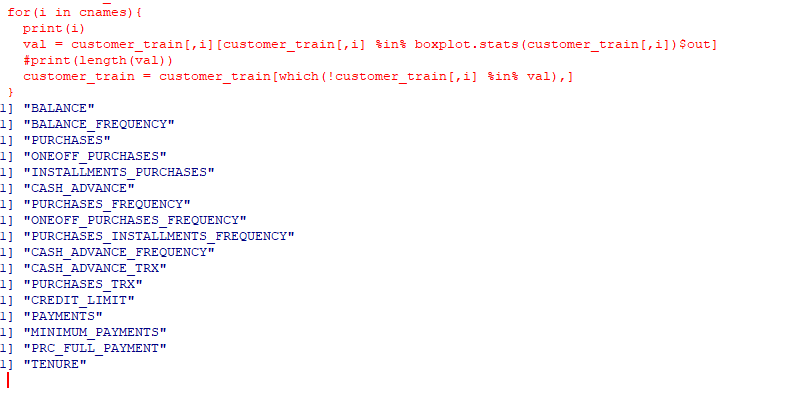
Here, I filtered all numeric variables into numeric\_data.   


This code helps us in plotting boxplot to visualize outliers and we can remove them from the data or impute them with mean or median. Outliers tells us that data quality is poor.



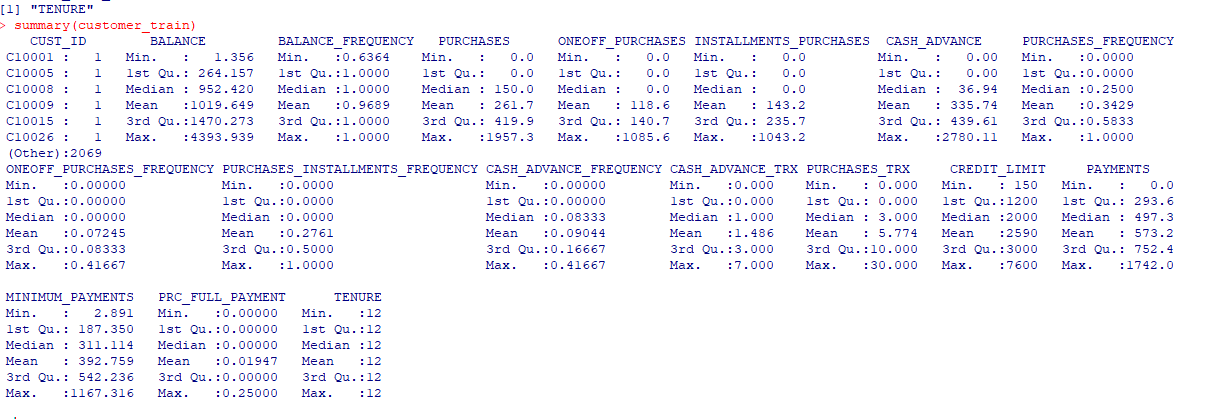


Now let us remove outliers using boxplot method



The following lines of code explains us that variable “var” stores outliers data from customer\_train variable. In 2nd line of code, the outliers are excluded and stored in customer\_train.

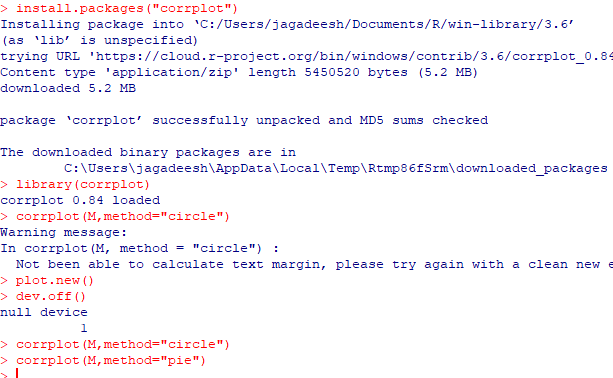
The summary of data is as follows after removing outliers:

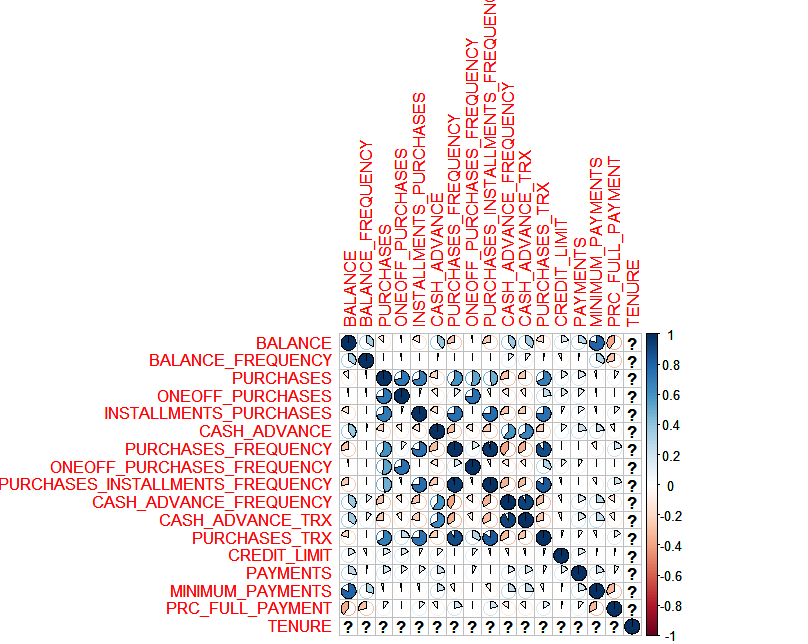


**CORRELATION GRAPH:**

Here, I used package “corrplot” to plot correlation plot and the parametre I should provide is a correlation matrix. Here M is a correlation matrix and I used cor() to find correlation matrix.

cor(customer\_train[,numeric\_index])->M





Now the data is stable and the dataset is not affected to outliers.

**KPI’s:**

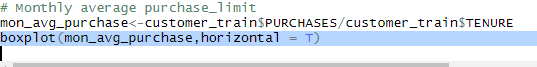
Advanced data preparation. Build an ‘enriched’ customer profile by deriving ‘intelligent’ KPI’s such as monthly average purchase and cash advance amount, purchases by type (one-off, instalments), average amount per purchase and cash advance transaction, limit usage (balance to credit limit ratio), payments to minimum payments ratio etc.

1. **Monthly Average Purchase:**

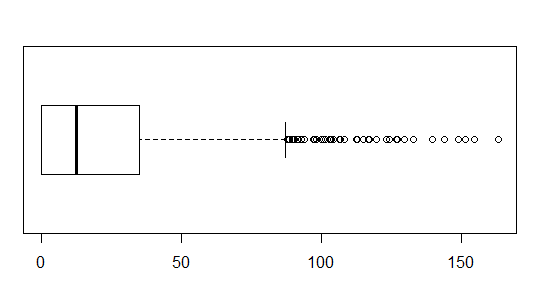
As Purchases column gives us that total purchase spent during last 12 months and tenure variable gives us that number of months as a customer.

Monthly Average Purchase = Purchases/ tenure

The R code for Monthly Average Purchase is



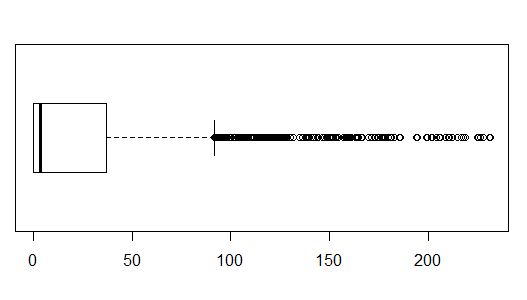
**BOX PLOT of** **Monthly Average Purchase:**





**MONTHLY\_CASH\_ADVANCE\_AMT:**

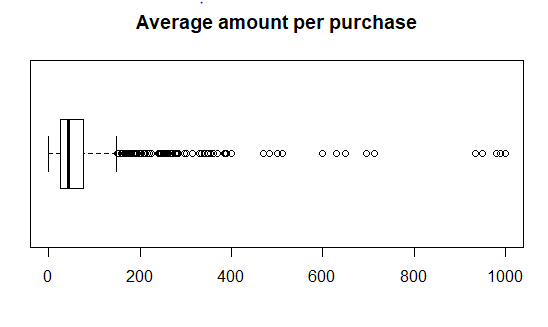
cash\_adv\_amount<-customer\_train$CASH\_ADVANCE/customer\_train$TENURE

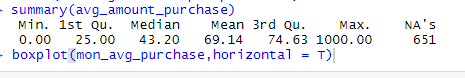




1. **Average amount per purchase:**

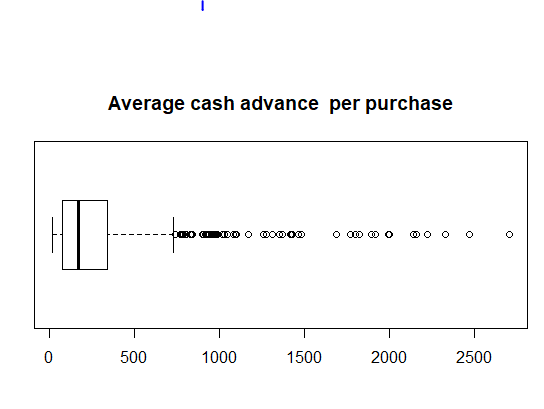
avg\_amount\_purchase<-customer\_train$PURCHASES/customer\_train$PURCHASES\_TRX

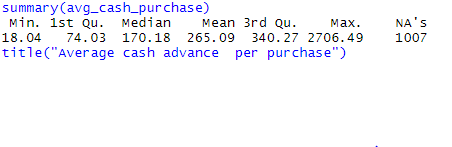




1. **Average Cash advance per purchase:**

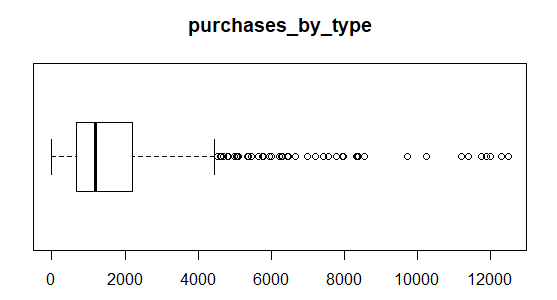
avg\_cash\_purchase<-customer\_train$CASH\_ADVANCE/customer\_train$CASH\_ADVANCE\_TRX





**PURCHASES\_BY\_TYPE:**

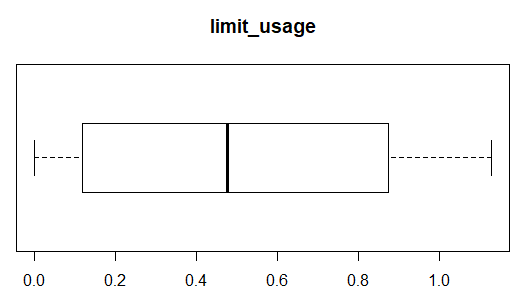
purchases\_by\_type<-customer\_train$ONEOFF\_PURCHASES/customer\_train$ONEOFF\_PURCHASES\_FREQUENCY





**LIMIT USAGE:**

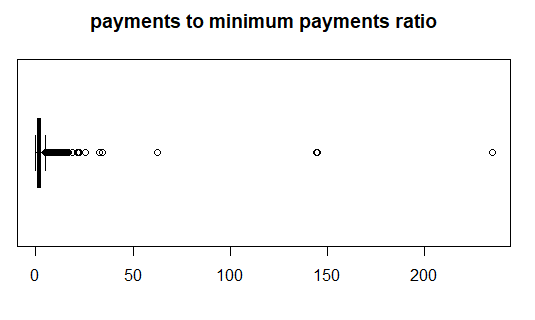
limit\_usage<-customer\_train$BALANCE/customer\_train$CREDIT\_LIMIT





**Payments to Minimum Payments Ratio:**

pay\_to\_min\_ratio<-customer\_train$PAYMENTS/customer\_train$MINIMUM\_PAYMENTS





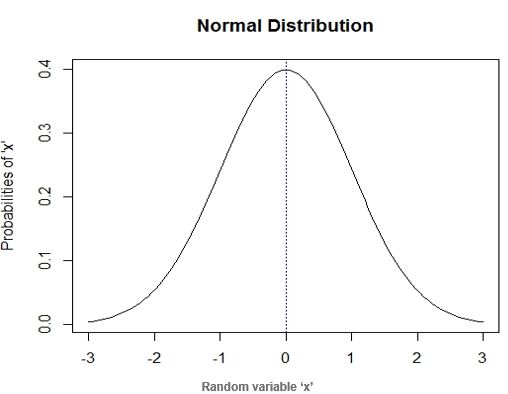
**CLUSTERING ANALYSIS:**

**NORMALIZATION:**

Standard Scaler: It transforms the data in such a manner that it has mean as 0 and standard deviation as 1. In short, it standardizes the data. Standardization is useful for data which has negative values. It arranges the data in normal distribution.

In probability theory, the Normal Distribution a.k.a. Gaussian Distribution is symmetrical with a single central peak (unimodal) at the mean.

When normally distributed random variables are plotted with random variable values on the horizontal axis (x-axis) and their respective probabilities on the vertical axis (y-axis), a bell-shaped curve known as a normal curve is created, falling evenly on either side of the mean.



**Following are properties of X (Random Variable) which is following the normal distribution:**

* The bell-shaped frequency distribution curve
* 1 is the total area under the normal curve
* 0 is the probability of X (normal random variable) being equal to particular value.
* Mean, median and mode of the distribution are at the centre.
* The curve is x-axis asymptote, i.e. two normal distribution tails never touch the x-axis and extend indefinitely

**Factor Analysis:**

Factor analysis is a useful tool for investigating variable relationships for complex concepts such as socioeconomic status, dietary patterns, or psychological scales. It allows researchers to investigate concepts that are not easily measured directly by collapsing a large number of variables into a few interpretable underlying factors. In every factor analysis, there are the same number of factors as there are variables. Each factor captures a certain amount of the overall variance in the observed variables, and the factors are always listed in order of how much variation they explain.

The eigenvalue is a measure of how much of the variance of the observed variables a factor explains. Any factor with an eigenvalue ≥1 explains more variance than a single observed variable.

So if the factor for socioeconomic status had an eigenvalue of 2.3 it would explain as much variance as 2.3 of the three variables. This factor, which captures most of the variance in those three variables, could then be used in other analyses.

The factors that explain the least amount of variance are generally discarded. The first chunk provides the “Uniquenesses”, which range from 0 to 1. This is the Ψ^ in our model above. What we’re looking for are high numbers. A high uniqueness for a variable usually means it doesn’t fit neatly into our factors

**CODE:**

#Standaradization of data frame to mean =0 and deviation =1

scaled\_df<-scale(customer\_train1)

scaled\_df[,-17]->scale\_df

**CODE TO DO FACTOR ANALYSIS:**

#FACTOR ANALYSIS

factorana<-factanal(scale\_df,factors = 2)

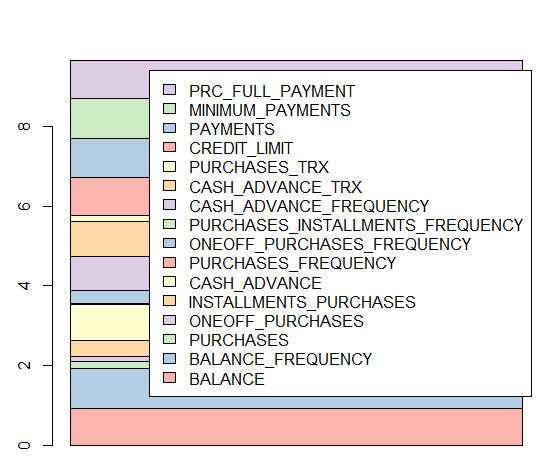
newfact1<-data.frame(factorana$uniquenesses)

head(newfact)

rownames(newfact)->row1

newfact$factorana.uniquenesses->row2

barplot(as.matrix(newfact1),legend=rownames(newfact1),col=brewer.pal(6,"Pastel1"))



newfact<-newfact[order(row2)]

CASH\_ADVANCE\_FREQUENCY -0.248568

CASH\_ADVANCE\_TRX -0.214977

CASH\_ADVANCE -0.199806

BALANCE -0.025506

TENURE -0.000000

BALANCE\_FREQUENCY 0.001609

MINIMUM\_PAYMENTS 0.013869

PRC\_FULL\_PAYMENT 0.079183

PAYMENTS 0.210395

CREDIT\_LIMIT 0.215486

PURCHASES\_INSTALLMENTS\_FREQUENCY 0.402919

PURCHASES\_FREQUENCY 0.569656

INSTALLMENTS\_PURCHASES 0.606326

ONEOFF\_PURCHASES\_FREQUENCY 0.649062

PURCHASES\_TRX 0.687207

ONEOFF\_PURCHASES 0.863917

PURCHASES 0.996624

From this data frame, based on features to be included in our clustering model will be selected based on components with Max Variance.

* PURCHASES
* ONEOFF\_PURCHASES
* PURCHASES\_TRX
* ONEOFF\_PURCHASES\_FREQUENCY
* INSTALLMENTS\_PURCHASES
* PURCHASES\_FREQUENCY
* PURCHASES\_INSTALLMENTS\_FREQUENCY
* CREDIT\_LIMIT
* PAYMENTS
* CASH\_ADVANCE\_FREQUENCY

**CLUSTERING ANALYSIS:**

Let us try to create the clusters for this data. As we can observe this data doesnot have a pre-defined class/output type defined and so it becomes necessary to know what will be an optimal number of clusters. Let us choose random value of cluster numbers for now and see how the clusters are created. Here we have taken a sample based on the one-off values taken from factor analysis.

sample1<-c("PURCHASES","ONEOFF\_PURCHASES","PURCHASES\_TRX","ONEOFF\_PURCHASES\_FREQUENCY",

"INSTALLMENTS\_PURCHASES","PURCHASES\_FREQUENCY","PURCHASES\_INSTALLMENTS\_FREQUENCY","CREDIT\_LIMIT","PAYMENTS",

"CASH\_ADVANCE\_FREQUENCY")

**CODE:**

subset(scale\_df,select=sample1)->cluster\_sample

head(cluster\_sample)

sample2

k\_means\_cluster<-kmeans(cluster\_sample,3)

k\_means\_cluster$cluster

k\_means\_cluster<-kmeans(cluster\_sample,3)

**OUTPUT:**

K-means clustering with 3 clusters of sizes 333, 555, 1187

Within cluster sum of squares by cluster:

[1] 3091.457 3215.147 4618.991

(between\_SS / total\_SS = 47.3 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss"

[6] "betweenss" "size" "iter" "ifault"

**Deciding Optimal K value:**

So the idea is to find such a value of k for which the model is not overfitting and at the same time clusters the data as per the actual distribution. Let us now approach how we will solve this problem of finding the best number of clusters. Here we use Elbow method to decide optimal value of K.

**Elbow Method:**

The elbow method looks at the percentage of variance explained as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn’t give much better modelling of the data. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the “elbow criterion”. This “elbow” cannot always be unambiguously identified.

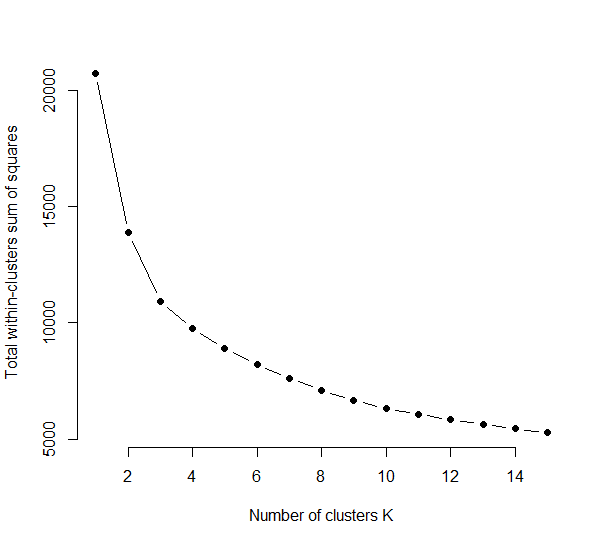
Plotting the value of K from the K value 2 to 15usingthe below code:

set.seed(123)

data<-cluster\_sample

max<-15

wss<-sapply(1:max,function(k){kmeans(data,k,nstart=50,iter.max=15)$tot.withinss})



wss

[1] 20740.000 13882.817 10925.595 9768.454 8912.478 8224.187 7611.480 7118.145

[9] 6665.038 6314.417 6068.820 5847.013 5642.991 5448.550 5285.317

From this graph, there is not much change from k=3. So, let us take k=3 as number of clusters.Here K=4 or K=5 will also be a potential choice.

For K value=5,

**Size of cluster**

k\_means\_cluster$size

[1] 448 522 167 294 644

**k-means cluster centres:**

> k\_means\_cluster$centers

PURCHASES ONEOFF\_PURCHASES PURCHASES\_TRX ONEOFF\_PURCHASES\_FREQUENCY

1 0.2628903 -0.4239638 0.9331710 -0.4499393

2 -0.7353506 -0.4478669 -0.7757268 -0.5060116

3 1.9538718 0.6496751 1.8535541 0.7094528

4 1.0458583 1.7796303 0.1904057 1.6583138

5 -0.5709640 -0.3229577 -0.5879721 -0.2178758

INSTALLMENTS\_PURCHASES PURCHASES\_FREQUENCY PURCHASES\_INSTALLMENTS\_FREQUENCY

1 0.8226000 1.30500339 1.3845690

2 -0.6134266 -0.85430984 -0.7105816

3 2.1826179 1.39287511 1.3770296

4 -0.3048664 0.05354709 -0.4144816

5 -0.5018362 -0.60100119 -0.5550771

CREDIT\_LIMIT PAYMENTS CASH\_ADVANCE\_FREQUENCY

1 -0.2275795 -0.2346572 -0.5601046

2 0.1071314 0.5716236 1.1178095

3 0.5026740 0.7341183 -0.1876793

4 0.3610959 0.2128712 -0.3055443

5 -0.2237201 -0.5876443 -0.3282566