Mini Project – Market Segmentation in Context of Product Service Management

Submission Date: Oct 04, 2019

Author: Ayush Jain
Mentor: Deepak Gupta

Table of Content

Table of Contents

1 Project Objective 4

2 Assumptions 4

3 Exploratory Data Analysis – Step by step approach 5

3.1 Environment Set up and Data Import 5

3.1.1 Install necessary Packages and Invoke Libraries 5

3.1.2 Set up working Directory 5

3.1.3 Import and Read the Dataset 6

3.2 Variable Identification 6

4 Univariate Analysis 7

5 Bi-Variate Analysis 10

6 Missing Value Identification 16

7 Conclusion 16

7.1 Checking for Multicollinearity 16

7.1.1 Plotting the Collinearity 17

7.2 Checking the Sphericity of the data. 19

7.3 Sampling Adequacy Test 19

7.4 Eigen Value Test 20

7.5 Performing Principal Component Analysis without Rotation 20

7.5.1 Building Regression Model on PCA without Rotation 21

7.5.2 Validating the Model using Test Data. 23

7.6 Performing Principal Component Analysis with Rotation 24

7.6.1 Building Regression Model on PCA with Rotation 24

7.6.2 Validating the Model using Test Data. 26

7.7 Performing Factor Analysis without Rotation 26

7.7.1 Building Regression Model on Factor Analysis without Rotation 28

7.7.2 Validating the Model using Test Data. 29

7.8 Performing Factor Analysis with Rotation 30

7.8.1 Building Regression Model on Factor Analysis without Rotation 31

7.8.2 Validating the Model using Test Data. 32

7.9 Summary 33
8 Appendix A – Source Code 33

1 Project Objective

The objective of this report is to explore the Market Segmentation in Context of Product service management. This will be performed in R using the csv file “Factor-Hair-Revised”.

The data file consist of 12 variables and 100 observations and is represented as below:

Variable	Expansion
ProdQual	Product Quality
Ecom	E-Commerce
TechSup	Technical Support
CompRes	Complaint Resolution
Advertising	Advertising
ProdLine	Product Line
SalesFImage	Salesforce Image
ComPricing	Competitive Pricing
WartyClaim	Warranty & Claims
OrdBilling	Order & Billing
DelSpeed	Delivery Speed
Satisfaction	Customer Satisfaction

In this, we know that the Customer Satisfaction is the function of remaining 11 variables i.e. Satisfaction is dependent on the rest of the variables.

We are supposed to build an optimum regression model to predict satisfaction covering the below points.

- Performing Exploratory data analysis on the dataset to visualize and understand the data and identify the outliers and missing values
- To check the Multicollinearity between the variables in the dataset and display the results.
- Build the Simple Linear regression model for the dependent variable with every independent variable and provide the feeds.
- Perform PCA/Factor analysis by extracting 4 factors. Interpret the output and name the Factors.
- Build the Multiple linear regression model with customer satisfaction as dependent variables and the four factors as independent variables. Include the observations in the report on the Model output and validity.

2 Assumptions

3 Exploratory Data Analysis – Step by step approach

Exploratory Data Analysis is one of the important phases in the data Analysis and understanding the significance of data. It usually consists of setting up the environment to work in R, loading the data and checking the validity of data loaded.

A Typical Data exploration activity consists of the following steps:

- Environment Set up and Data Import.
- Variable Identification.

We shall follow these steps in exploring the provided dataset.

3.1 Environment Set up and Data Import

3.1.1 Install necessary Packages and Invoke Libraries

In this section, we will install and invoke the necessary Packages and Libraries that are going to be the part of our work throughout the project. Having all the packages at the same places increases code readability and Understandability.

```
#Installing the required Packages.

library(psych)

library(corrplot)

library(caTools)

install.packages("DataExplorer")

library(DataExplorer)

install.packages("devtools")

library(devtools)

install_github("vqv/ggbiplot")

library(ggbiplot)
```

3.1.2 Set up working Directory

Setting a working directory on starting of the R session makes importing and exporting data files and code files easier. Basically, working directory is the location/ folder on the PC where you have the data, codes etc. related to the project. This helps maintain the code readability and avoid unwanted errors.

```
# Setting up Working Directory.

setwd("D:/Great Learning/Project 2")
```

Please refer Appendix A for Source Code.

3.1.3 Import and Read the Dataset

The given dataset is in .csv format. Hence, the command 'read.csv' is used for importing the file.

```
#Reading the file to R.

dataset <- read.csv("Factor-Hair-Revised.csv")
str(dataset)
head(dataset)
dataset <- dataset[,-1]
```

```
> dataset <- read.csv("Factor-Hair-Revised.csv")
> str(dataset)
'data.frame':	100 obs. of  13 variables:
 $ ID          : int  1 2 3 4 5 6 7 8 9 10 ...
 $ ProdQual    : num  8.5 8.2 9.2 6.4 9 6.5 6.9 6.2 5.8 6.4 ...
 $ Ecom        : num  3.9 2.7 3.4 3.3 3.4 2.8 3.7 3.3 3.6 4.5 ...
 $ TechSup     : num  2.5 5.1 5.6 7 5.2 3.1 5 3.9 5.1 5.1 ...
 $ CompRes     : num  5.9 7.2 5.6 3.7 4.6 4.1 2.6 4.8 6.7 6.1 ...
 $ Advertising : num  4.8 3.4 5.4 4.7 2.2 4 2.1 4.6 3.7 4.7 ...
 $ ProdLine    : num  4.9 7.9 7.4 4.7 6 4.3 2.3 3.6 5.9 5.7 ...
 $ SalesFImage : num  6 3.1 5.8 4.5 4.5 3.7 5.4 5.1 5.8 5.7 ...
 $ ComPricing  : num  6.8 5.3 4.5 8.8 6.8 8.5 8.9 6.9 9.3 8.4 ...
 $ WartyClaim  : num  4.7 5.5 6.2 7 6.1 5.1 4.8 5.4 5.9 5.4 ...
 $ OrdBilling  : num  5 3.9 5.4 4.3 4.5 3.6 2.1 4.3 4.4 4.1 ...
 $ DelSpeed    : num  3.7 4.9 4.5 3 3.5 3.3 2 3.7 4.6 4.4 ...
 $ Satisfaction: num  8.2 5.7 8.9 4.8 7.1 4.7 5.7 6.3 7 5.5 ...
> head(dataset)
  ID ProdQual Ecom TechSup CompRes Advertising ProdLine SalesFImage ComPricing WartyClaim OrdBilling DelSpeed Satisfaction
1  1      8.5  3.9     2.5     5.9         4.8      4.9         6.0        6.8        4.7        5.0      3.7          8.2
2  2      8.2  2.7     5.1     7.2         3.4      7.9         3.1        5.3        5.5        3.9      4.9          5.7
3  3      9.2  3.4     5.6     5.6         5.4      7.4         5.8        4.5        6.2        5.4      4.5          8.9
4  4      6.4  3.3     7.0     3.7         4.7      4.7         4.5        8.8        7.0        4.3      3.0          4.8
5  5      9.0  3.4     5.2     4.6         2.2      6.0         4.5        6.8        6.1        4.5      3.5          7.1
6  6      6.5  2.8     3.1     4.1         4.0      4.3         3.7        8.5        5.1        3.6      3.3          4.7
> dataset <- dataset[,-1]
```

After importing the dataset we found that there is an additional variable called Id which can be removed from the dataset as this will not add any value to the model and it is just the sequence.

Please refer Appendix A for Source Code.

3.2 Variable Identification

This section holds the Variables/ Methods that are used during the Analysis of the problem. Below are the Functions that we have used for the Analysis.

- setwd(): setwd(dir) is used to set the working directory to dir.
- read.csv(): Reads a file in table format and creates a data frame from it.
- head(): Returns the first parts of a vector, matrix, table, data frame or function.
- str(): Compactly display the internal Structure of an R object.
- summary(): summary is a generic function used to produce result summaries of the results of various model fitting functions.
- Is.null(): NULL is often returned by expressions and functions whose value is undefined. is.null returns TRUE if its argument's value is NULL and FALSE otherwise.

- Boxplot(): It is plotting technique, which is used to identify if there any outliners are present in the data.
- Plot(): It is data visualization technique.
- lm(): lm is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance
- cor(): cor compute the variance of x and the covariance or correlation of x and y if these are vectors. If x and y are matrices then the covariances (or correlations) between the columns of x and the columns of y are computed.
- Corrplot(): This is used to plot the correlation matrix for better visualization and presentation.
- Plot_correlation():
  This is the another way of plotting the correlation and identify the multicollinearity of the variables in the dataset.
- Cortest.bartlett():
  This is a test performed to check the shape of the data. The p-value < 0.05 means that the data is in the proper shape and we can proceed further with the data.
- KMO(): It is a sample Adequacy test, which shows if the sample provided is sufficient enough to proceed further. Overall MSA > 0.5 means that the data is enough to work with.
- eigen(): Computes eigenvalues and eigenvectors of numeric (double, integer, logical) or complex matrices.
- principal(): It is principal components analysis (PCA) for n principal components of either a correlation or covariance matrix.
- fa(): Similar to principal, it is a data reduction method that is used to remove the multicollinearity and reduce the number of factors. In this variable is a linear combination of different factors.
- fa.diagram(): This is used to visualize the factor analysis and identify the association of the variables between the factors.
- cbind(): This method is used to join variables on the basis of the columns.
- Colnames(): This method is used to rename the column names.
- set.seeds(): set.seed is the recommended way to specify seeds.
- Sample.split(): Split data from vector Y into two sets in predefined ratio while preserving relative ratios of different labels in Y. Used to split the data used during classification into train and test subsets.
- Subset(): This method is used to subset the data.

4 Univariate Analysis

Univariate analysis is perhaps the simplest form of statistical analysis. Like other forms of statistics, it can be inferential or descriptive. The key fact is that only one variable is involved.

For Numeric variables, default plot is histogram and boxplot while for Categorical variables it is Bar plot.

Histogram: A histogram is an accurate representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable.

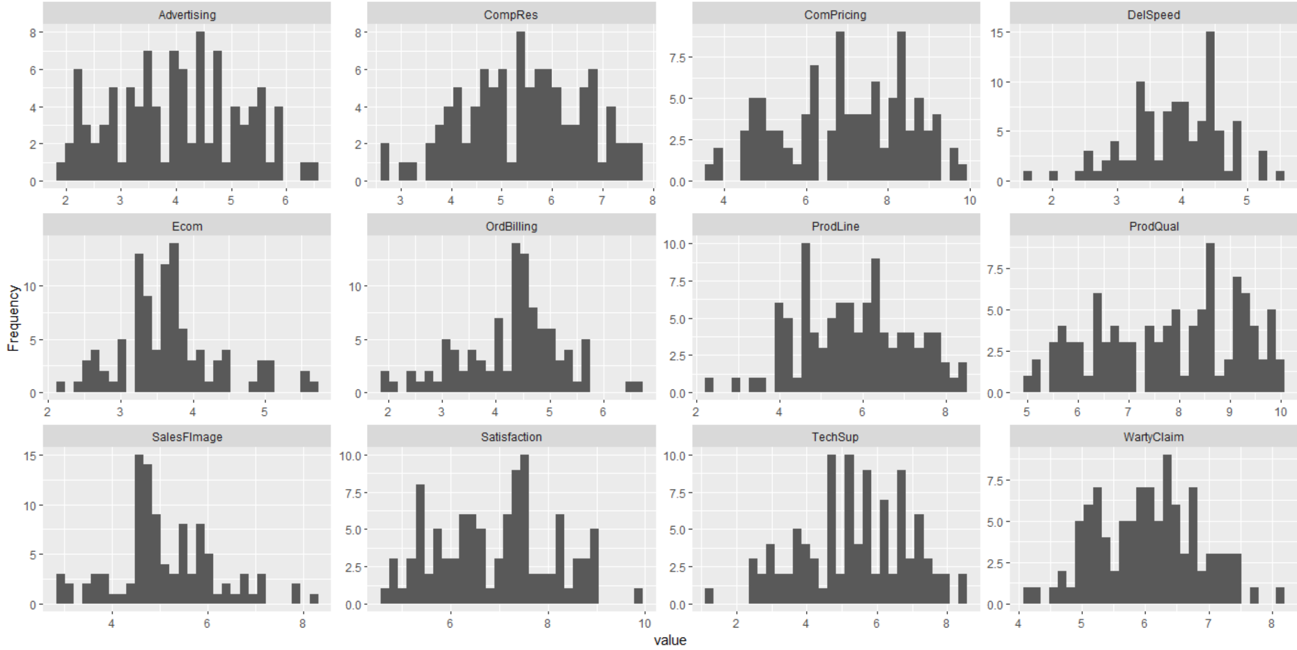
Boxplot: A box plot or boxplot is a method for graphically depicting groups of numerical data through their quartiles. Outliers may be plotted as individual points.

In the problem given, we will be using the above two plotting functions to perform the Univariate analysis on the dataset and identify any outliners present in the data.

Plotting the histogram for all the numeric variables in the dataset.

To analyze each variables, we plot the histogram for the variables.

```
###Performing Univariate Analysis on the dataset.

plot_histogram(dataset)
```



Plotting the Boxplot to identify the Outliers in the data.

We use Boxplot to check if there are any Outliers available in the data, boxplot identify the outliers basis the below formulation.

IQR = Q3 -Q1

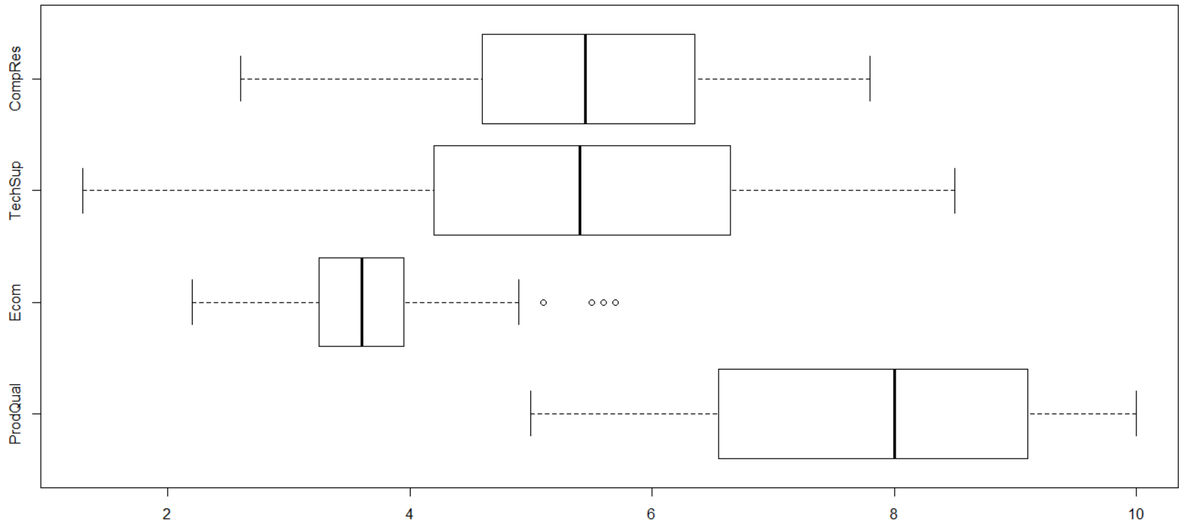
Lower Limit = Q1 – 1.5(IQR)

Upper Limit = Q3 + 1.5(IQR)

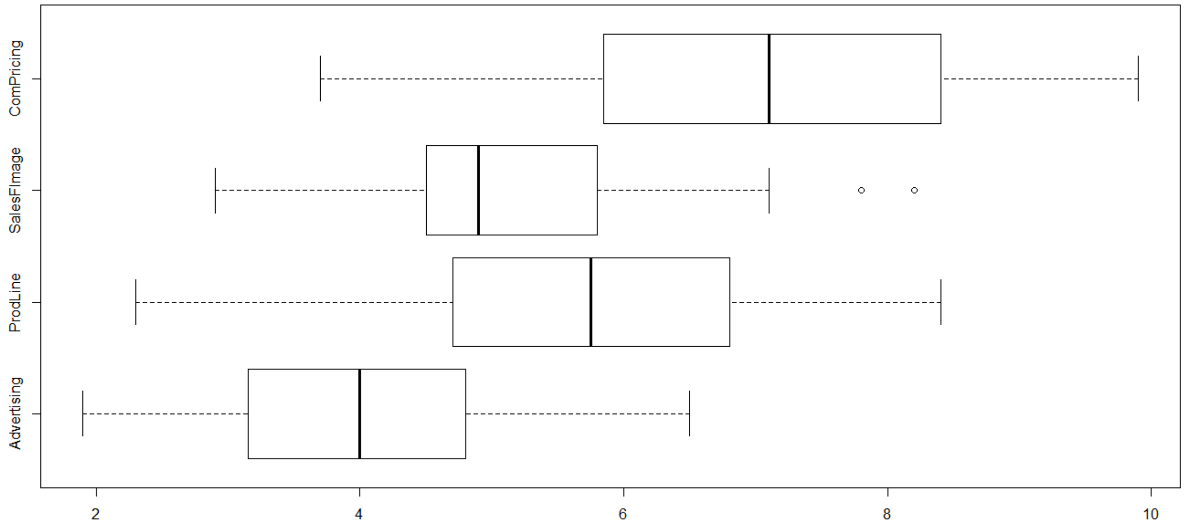
Points outside the upper and Lower limits are Outliers.

```
#Plotting the dataset to identify the outliers.
#Plot2
boxplot(dataset[,1:4],horizontal = TRUE)

#Plot2
boxplot(dataset[,5:8],horizontal = TRUE)

#Plot3
boxplot(dataset[,9:12],horizontal = TRUE)
```

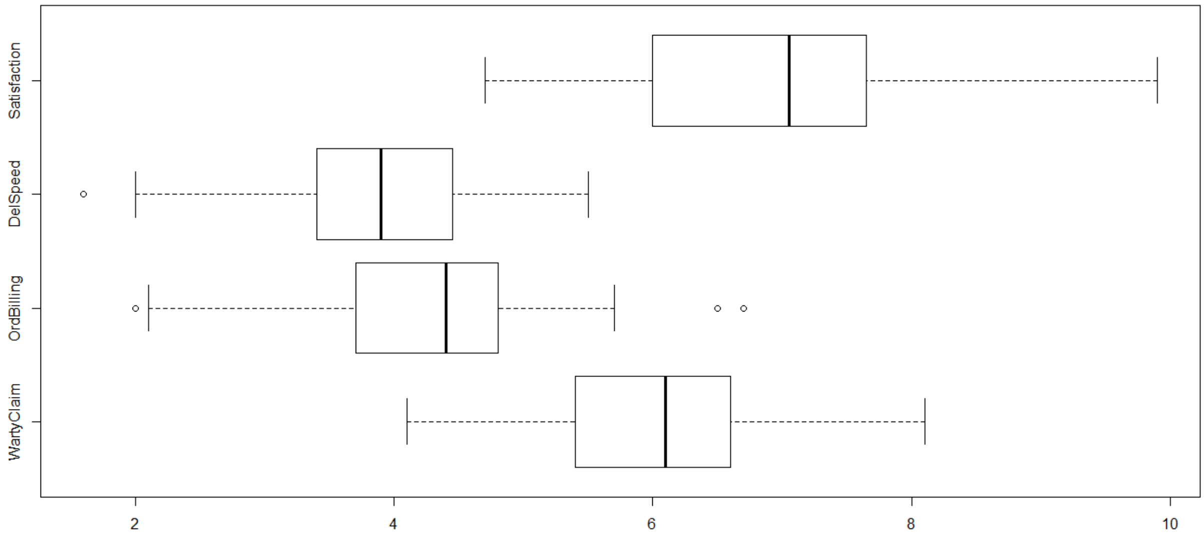
Plot 1:



Plot 2:



Plot 3:



By using Boxplot, we identify that the below variables in the dataset have the outliers present.

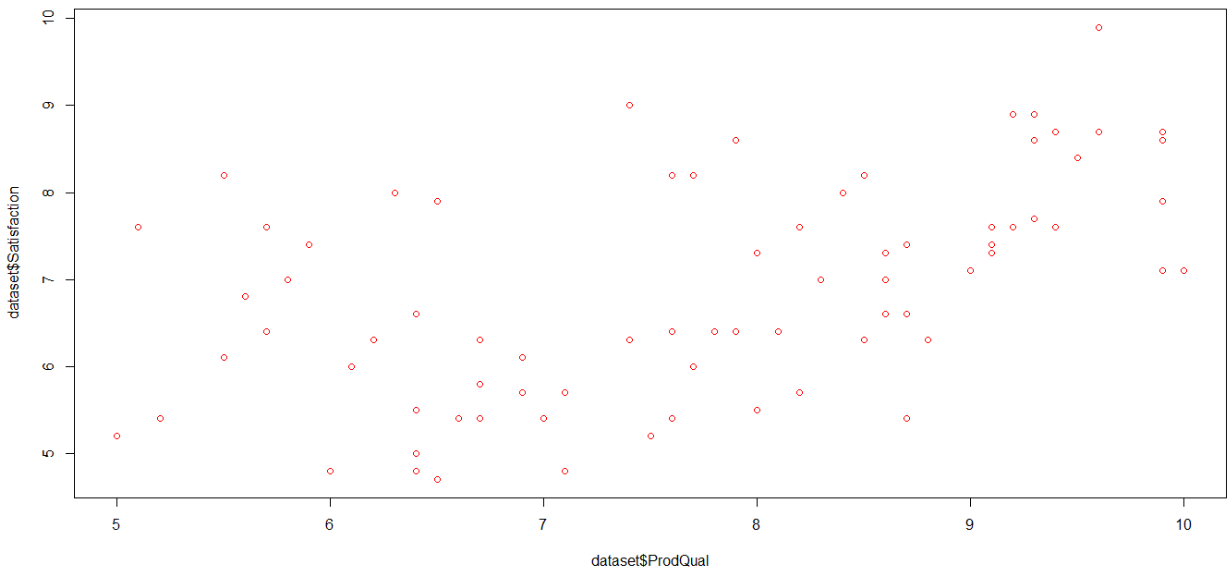
- ECOM
- SalesFImage
- DelSpeed
- OrdBilling

5 Bi-Variate Analysis

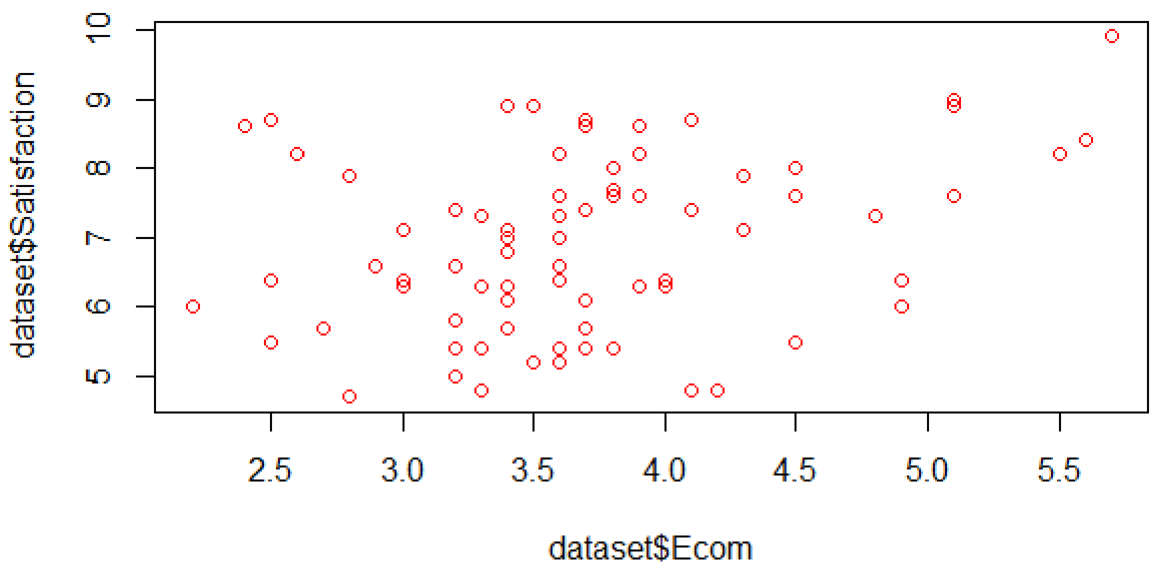
Multivariate analysis is a set of techniques used for analysis of data sets that contain more than one variable, and the techniques are especially valuable when working with correlated variables. The techniques provide an empirical method for information extraction, regression, or classification.

For Multivariate analysis, the default plot is the Scatter Plot. We will be plotting the correlation between the different variables with Customer Satisfaction to understand the relation between the dependent variable Satisfaction with the Independent variables.

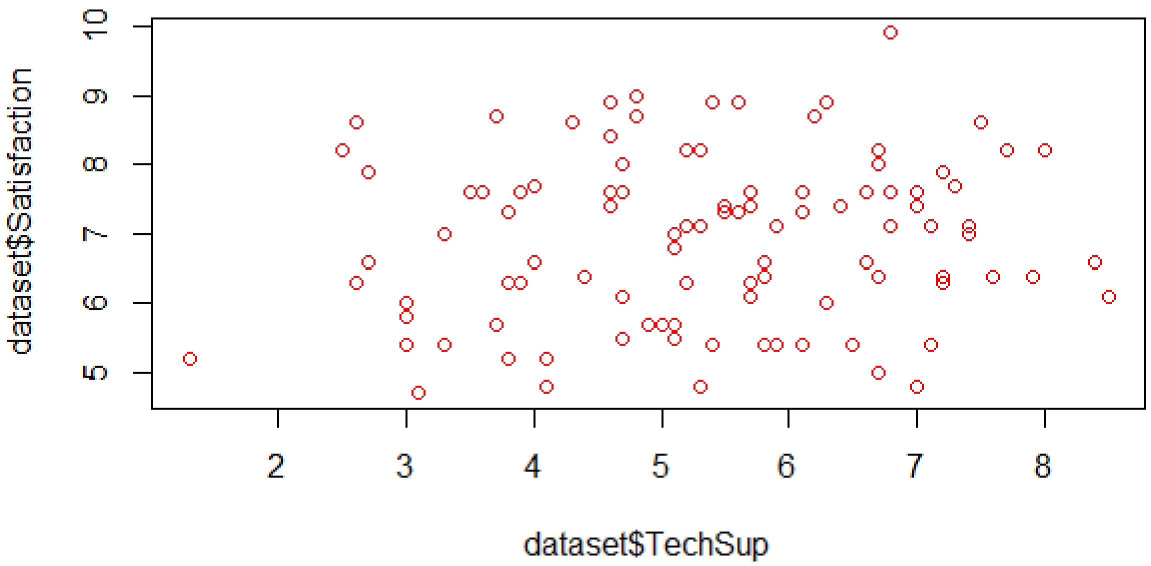
```
#Plot: ProdQual vs Satisfaction.
plot(dataset$ProdQual,dataset$Satisfaction,col = "Red" )
```




```
#Plot: Ecom vs Satisfaction.
plot(dataset$Ecom,dataset$Satisfaction,col = "Red")
```

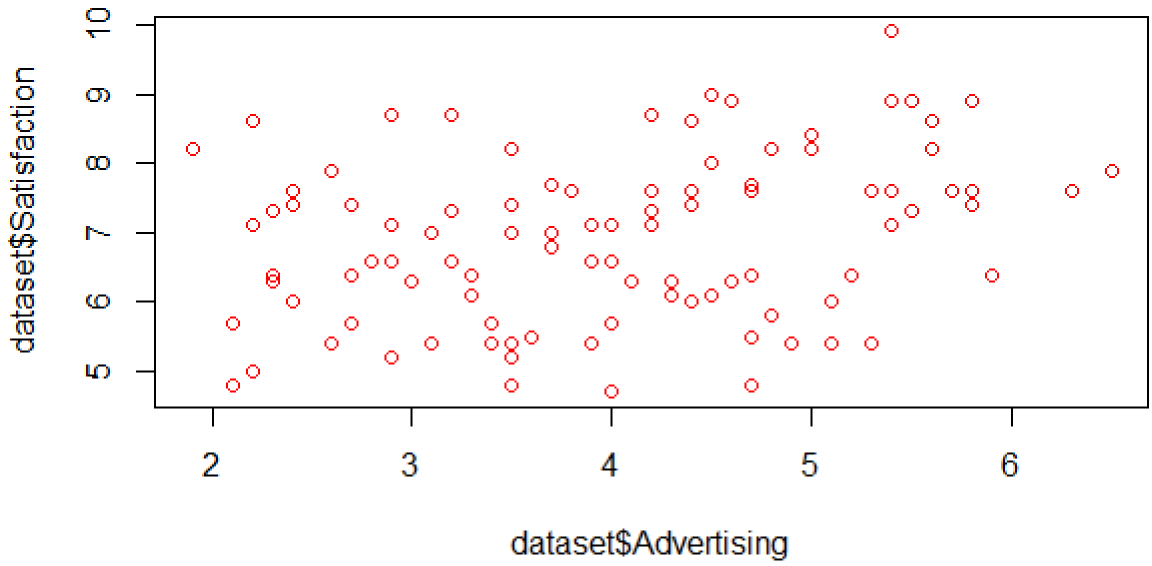



```
#Plot: TechSup vs Satisfaction.
plot(dataset$TechSup,dataset$Satisfaction,col = "Red")
```

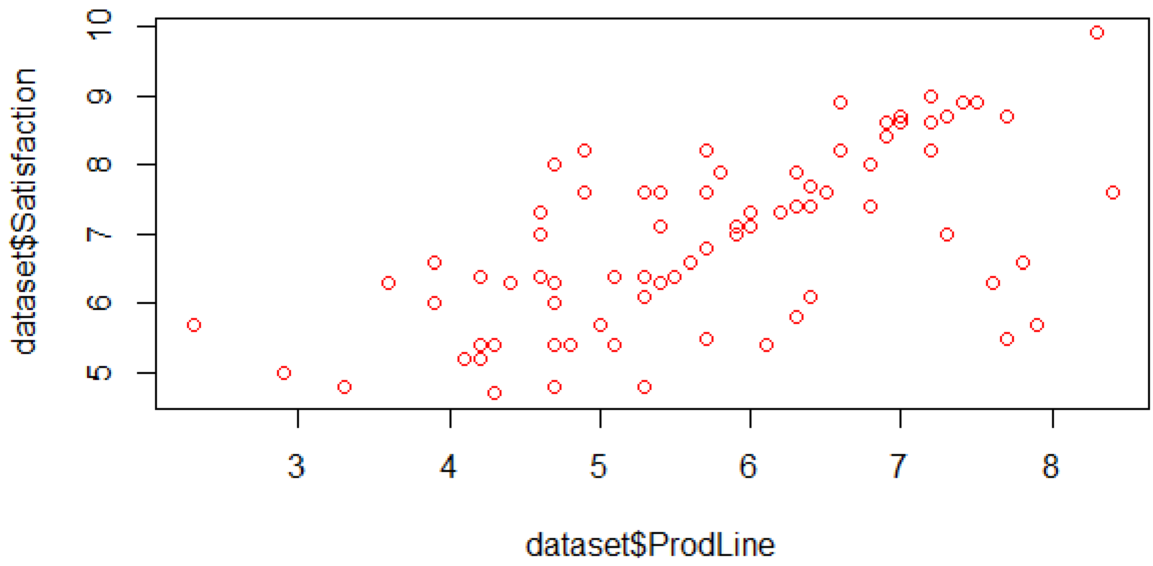



```
#Plot: CompRes vs Satisfaction.
plot(dataset$CompRes,dataset$Satisfaction,col = "Red")
```


```
#Plot: Advertising vs Satisfaction.
plot(dataset$Advertising,dataset$Satisfaction,col = "Red")
```

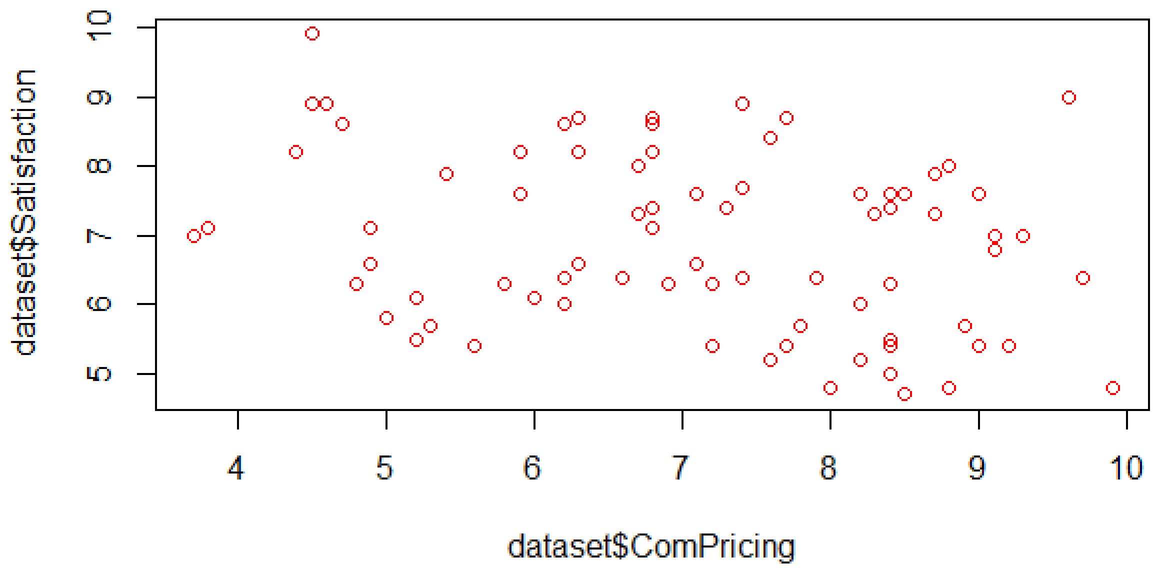



```
#Plot: ProdLine vs Satisfaction.
plot(dataset$ProdLine,dataset$Satisfaction,col = "Red")
```

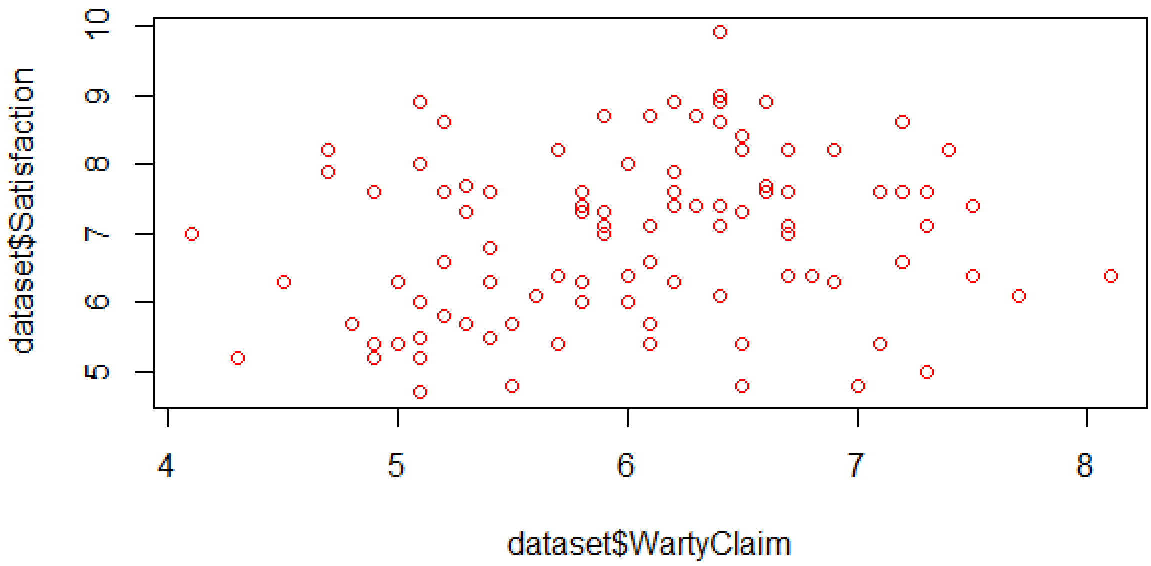



```
#Plot: SalesFImage vs Satisfaction.
plot(dataset$SalesFImage,dataset$Satisfaction,col = "Red")
```


```
#Plot: ComPricing vs Satisfaction.
plot(dataset$ComPricing,dataset$Satisfaction,col = "Red")
```

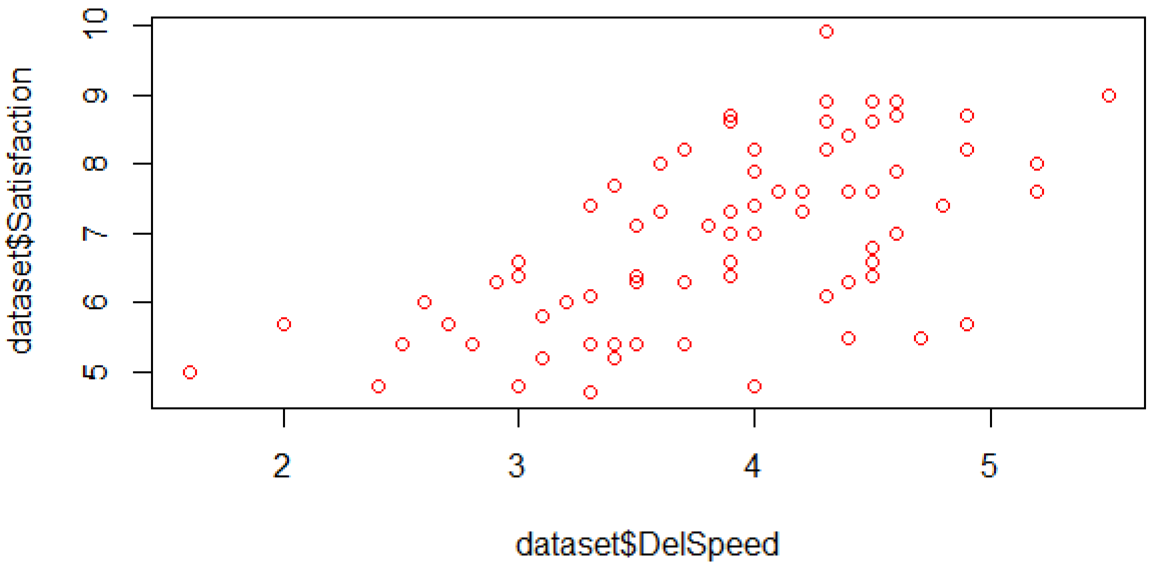



```
#Plot: WartyClaim vs Satisfaction.
plot(dataset$WartyClaim,dataset$Satisfaction,col = "Red")
```




```
#Plot: OrdBilling vs Satisfaction.
plot(dataset$OrdBilling,dataset$Satisfaction,col = "Red")
```


```
#Plot: DelSpeed vs Satisfaction.
plot(dataset$DelSpeed,dataset$Satisfaction,col = "Red")
```



By looking at the above plots, we can say that Dependent Variable (Satisfaction) is not showing a strong linear relationship with any of the Independent Variable. There is a sign of non-Significant correlation between the variables which needs to be removed. Hence to remove this we would proceed with the Principal Component Analysis/ Factor Analysis.

6 Missing Value Identification

In the above problem, we check is there are any null values available in the data.

```
> # Looking for the Null values in the Dataset
> is.null(dataset)
[1] FALSE
```

Hence, no missing values present in the dataset we can proceed further with the analysis.

7 Conclusion

Going further we will be checking for the presence of multicollinearity between the independent variables in the dataset.

Multicollinearity refers to a situation in which two or more explanatory variables in a multiple regression model are highly linearly related. It leads to the Imprecise estimate of the effect of the independent variable on the Dependent variable.

Hence Collinearity, must be treated and reduced before performing the Regression Model.

7.1 Checking for Multicollinearity

Before proceeding with the multicollinearity check we have a dependent variable available in the dataset that we need to exclude. To achieve this we create a copy of our dataset and subset the data.

```
#Identifying the Correlation between the Independent variables.
# Removing the Dependent Variable to check the multicolinearity
# between the Independent Variables

dataset1 <- dataset

dataset<- dataset[,-12]
```

To check for the existence of Multicollinearity, we use the method cor() from the package "psych", that will create a correlation matrix between the independent variables.

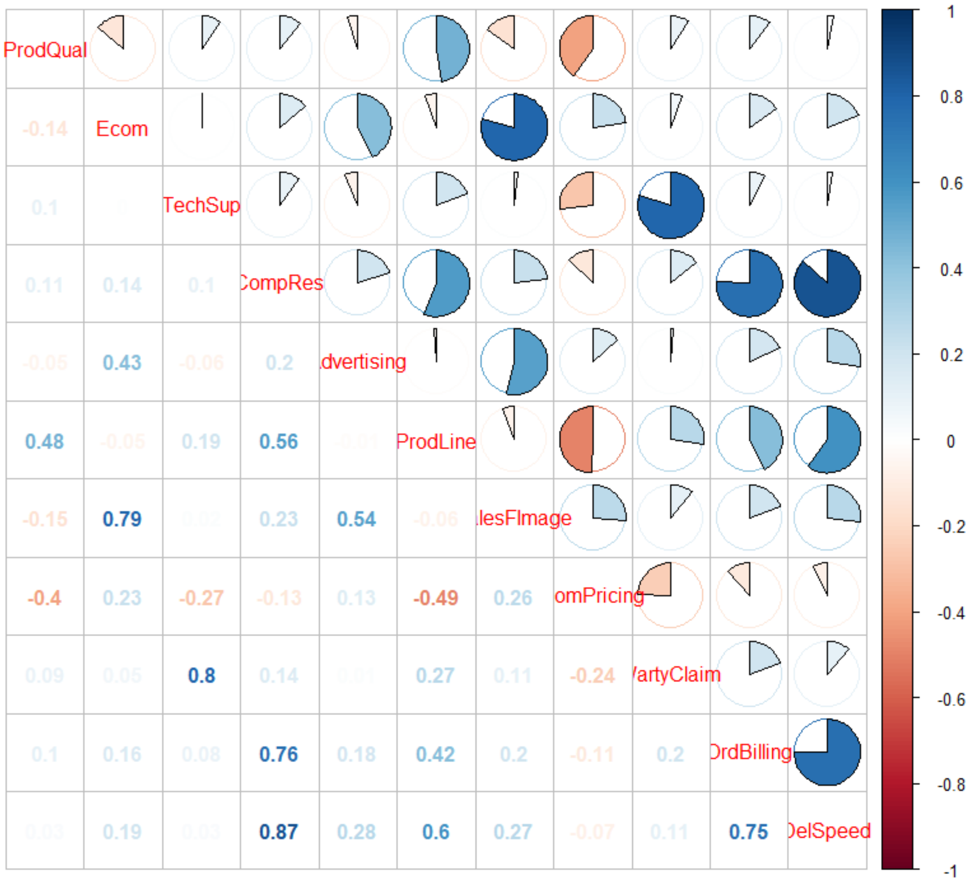
```
#Creating the correlation Matrix
mat <- cor(dataset)
mat
```

```
> mat
               ProdQual          Ecom       TechSup    CompRes Advertising    ProdLine SalesFImage  ComPricing  WartyClaim  OrdBilling
ProdQual     1.00000000 -0.1371632174  0.0956004542  0.1063700 -0.05347313  0.47749341 -0.15181287 -0.40128188  0.08831231  0.10430307
Ecom        -0.13716322  1.0000000000  0.0008667887  0.1401793  0.42989071 -0.05268784  0.79154371  0.22946240  0.05189819  0.15614733
TechSup      0.09560045  0.0008667887  1.0000000000  0.0966566 -0.06287007  0.19262546  0.01699054 -0.27078668  0.79716793  0.08010182
CompRes      0.10637000  0.1401792611  0.0966565978  1.0000000  0.19691685  0.56141695  0.22975176 -0.12795425  0.14040830  0.75686859
Advertising -0.05347313  0.4298907110 -0.0628700668  0.1969168  1.00000000 -0.01155082  0.54220366  0.13421689  0.01079207  0.18423559
ProdLine     0.47749341 -0.0526878383  0.1926254565  0.5614170 -0.01155082  1.00000000 -0.06131553 -0.49494840  0.27307753  0.42440825
SalesFImage -0.15181287  0.7915437115  0.0169905395  0.2297518  0.54220366 -0.06131553  1.00000000  0.26459655  0.10745534  0.19512741
ComPricing  -0.40128188  0.2294624014 -0.2707866821 -0.1279543  0.13421689 -0.49494840  0.26459655  1.00000000 -0.24498605 -0.11456703
WartyClaim   0.08831231  0.0518981915  0.7971679258  0.1404083  0.01079207  0.27307753  0.10745534 -0.24498605  1.00000000  0.19706512
OrdBilling   0.10430307  0.1561473316  0.0801018246  0.7568686  0.18423559  0.42440825  0.19512741 -0.11456703  0.19706512  1.00000000
DelSpeed     0.02771800  0.1916360683  0.0254406935  0.8650917  0.27586308  0.60185021  0.27155126 -0.07287173  0.10939460  0.75100307
               DelSpeed
ProdQual     0.02771800
Ecom         0.19163607
TechSup      0.02544069
CompRes      0.86509170
Advertising  0.27586308
ProdLine     0.60185021
SalesFImage  0.27155126
ComPricing  -0.07287173
WartyClaim   0.10939460
OrdBilling   0.75100307
DelSpeed     1.00000000
```

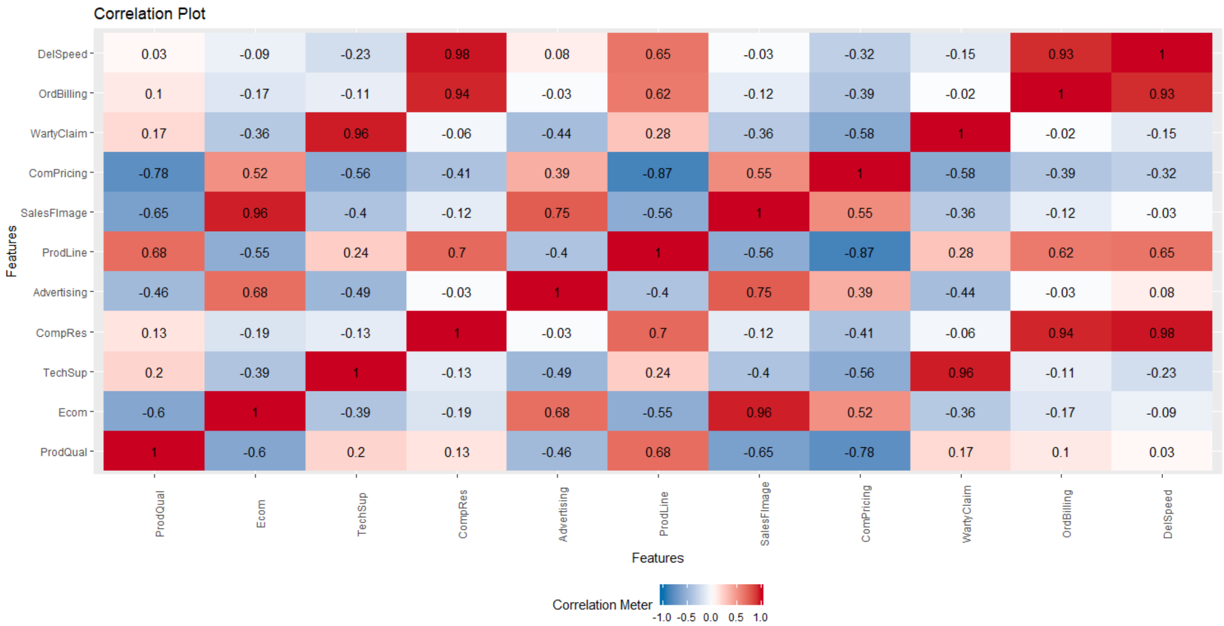
7.1.1 Plotting the Collinearity

```
###Plotting the Correlation Matrix

# Plot 1
corrplot.mixed(mat, lower = "number", upper = "pie")

# Plot 2
plot_correlation(mat, title = "Correlation Plot")
```

Plot 1:



Plot 2:



By looking at the above two plots we can conclude that the variables are highly correlated. In the Plot 2, collinearity can easily be interpreted on the scale of -1.0 to +1.0, where +1.0 indicates a strong relationship whereas -1.0 indicates the strong negative strong relationship and 0 indicates no collinearity.

7.2 Checking the Sphericity of the data.

Bartlett's test of sphericity tests the hypothesis that your correlation matrix is an identity matrix, which would indicate that your variables are unrelated and therefore unsuitable for structure detection. Small values (less than 0.05) of the significance level indicate that a factor analysis may be useful with your data.

```
#Performing Bartlett test and KMO Test to check the validity of the Correlation
#Bartlett Test: If p-Value < 0.05, correlation is valid

cortest.bartlett(mat,n = 100)
#$p.value = 1.79337e-96
```

```
> cortest.bartlett(mat,n = 100)
$chisq
[1] 619.2726

$p.value
[1] 1.79337e-96

$df
[1] 55
```

By the Bartlett's test, we achieved the p value as 1.79337 e-96 which is significantly smaller then 0.05, hence we can confirm that the data is non spherical and Factor Analysis will be useful.

7.3 Sampling Adequacy Test

The **Kaiser-Meyer-Olkin (KMO) Measure of Sampling Adequacy** is a statistic that indicates the proportion of variance in your variables that might be caused by underlying factors. High values (close to 1.0) generally indicate that a factor analysis may be useful with your data. If the value is less than 0.50, the results of the factor analysis probably won't be very useful.

```
#KMO Test: If the MSA > 0.5, correlation is valid
KMO(mat)
#Overall MSA =  0.65
```

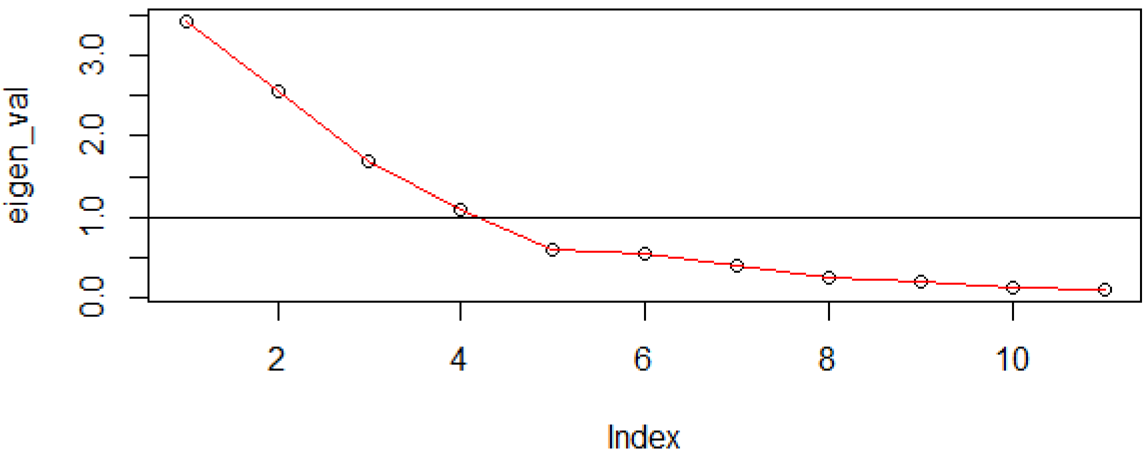
```
> #KMO Test: If the MSA > 0.5, correlation is valid
> KMO(mat)
Kaiser-Meyer-Olkin factor adequacy
Call: KMO(r = mat)
Overall MSA =  0.65
MSA for each item = 
   ProdQual        Ecom     TechSup     CompRes Advertising 
       0.51        0.63        0.52        0.79        0.78 
   ProdLine SalesFImage  ComPricing  WartyClaim  OrdBilling 
       0.62        0.62        0.75        0.51        0.76 
   DelSpeed 
       0.67 
```

In the above situation the overall MSA is 0.65, which is greater then 0.5 hence the samples are sufficient enough to proceed with the factor Analysis.

7.4 Eigen Value Test

This test holds true to identify the number of factors that can be achieved from the total number of factors using the Principal Component Analysis/ Factor Analysis.

```
### Performing Eigen Test

a = eigen(mat)
eigen_val <- a$values

#Plotting Eigen Values
plot(eigen_val)
lines(eigen_val, col = "red")
abline(h = 1)
```



Using Keiser Rule, we consider the values that are > 1, hence we take 4 factors to proceed with the Principal Component Analysis /Factor Analysis

7.5 Performing Principal Component Analysis without Rotation

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

```
### Performing PCA.

#Using the Kizer Rule, we take the number of factors for which the eigen values are greater then 1.
#In this case, we have 4 values which are greater the 1, hence we take 4 factors for PCA
#With Rotate = "None"

pca_no_rtt <- principal(dataset, nfactors = 4, rotate = "none")
pca_no_rtt

pca_score_no_rtt <- pca_no_rtt$scores
pca_score_no_rtt
```

```
> pca_no_rtt
Principal Components Analysis
Call: principal(r = dataset, nfactors = 4, rotate = "none")
Standardized loadings (pattern matrix) based upon correlation matrix
              PC1   PC2   PC3   PC4   h2    u2 com
ProdQual     0.25 -0.50 -0.08  0.67 0.77 0.232 2.2
Ecom         0.31  0.71  0.31  0.28 0.78 0.223 2.1
TechSup      0.29 -0.37  0.79 -0.20 0.89 0.107 1.9
CompRes      0.87  0.03 -0.27 -0.22 0.88 0.119 1.3
Advertising  0.34  0.58  0.11  0.33 0.58 0.424 2.4
ProdLine     0.72 -0.45 -0.15  0.21 0.79 0.213 2.0
SalesFImage  0.38  0.75  0.31  0.23 0.86 0.141 2.1
ComPricing  -0.28  0.66 -0.07 -0.35 0.64 0.359 1.9
WartyClaim   0.39 -0.31  0.78 -0.19 0.89 0.108 2.0
OrdBilling   0.81  0.04 -0.22 -0.25 0.77 0.234 1.3
DelSpeed     0.88  0.12 -0.30 -0.21 0.91 0.086 1.4

                       PC1  PC2  PC3  PC4
SS loadings           3.43 2.55 1.69 1.09
Proportion Var        0.31 0.23 0.15 0.10
Cumulative Var        0.31 0.54 0.70 0.80
Proportion Explained  0.39 0.29 0.19 0.12
Cumulative Proportion 0.39 0.68 0.88 1.00

Mean item complexity =  1.9
Test of the hypothesis that 4 components are sufficient.

The root mean square of the residuals (RMSR) is  0.06
 with the empirical chi square  39.02  with prob <  0.0018

Fit based upon off diagonal values = 0.97
```

After performing Principal Factor Analysis, we observed that the cumulative Variance in the data is 80% and the Communality(h2) for every individual variable is high. Let us proceed on the Model Building.

7.5.1 Building Regression Model on PCA without Rotation

Once we are all set with PCA and got the PCA Scores ready, we built a Regression Model on top of it. To build the model we need to have the Dependent variable or the Y variable in the data set. Now we combine the replicated dataset with the score Matrix to achieve the Satisfaction column.

Once we have the dataset ready, we split the data into Train and Test data with a ratio of 70, 30 respectively and build the model on Train data. Test data is used to check the validity of the Model.

```
#Creating the Score Matrix.
pca_score_no_rtt <- pca_no_rtt$scores
head(pca_score_no_rtt)

#Joining the Score Matrix with the Original dataset to get the Satisfaction variable
pca_reg_no_rtt <- cbind(dataset1[,12],pca_score_no_rtt)

#Renaming the Factored dataset.
colnames(pca_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
head(pca_reg_no_rtt)

pca_reg_no_rtt<- as.data.frame(pca_reg_no_rtt)

set.seed(42)

#Dividing the dataset into Test and Train.
index_pca_no_rtt <- sample.split(pca_reg_no_rtt$Satisfaction,SplitRatio = .70)
train_pca_no_rtt <- subset(pca_reg_no_rtt,index_pca_no_rtt ==TRUE)
test_pca_no_rtt <- subset(pca_reg_no_rtt, index_pca_no_rtt == FALSE)

#Building the Regression Model on the Train Data.
model_pca_no_rtt<- lm(Satisfaction~.,data = train_pca_no_rtt)
summary(model_pca_no_rtt)

#Validating the Model on Test Data.
pred_pca_no_rtt <- predict(model_pca_no_rtt,newdata = test_pca_no_rtt)
summary(pred_pca_no_rtt)
```

```
> #Creating the Score Matrix.
> pca_score_no_rtt <- pca_no_rtt$scores
> head(pca_score_no_rtt)
            PC1        PC2        PC3        PC4
[1,]  0.2426689  0.9840535 -1.4862465  1.2369663
[2,]  0.2452695 -1.5475268 -1.5186060 -0.9357584
[3,]  1.3499046 -0.4740518 -0.1435113  1.1415100
[4,] -1.1069113  0.1098079  1.4261590 -1.2131900
[5,] -0.3187131 -0.8668559 -0.2027249  0.3380636
[6,] -1.7049121  0.2294714 -1.1679476 -0.7306963
> #Joining the Score Matrix with the Original dataset to get the Satisfaction variable
> pca_reg_no_rtt <- cbind(dataset1[,12],pca_score_no_rtt)
> #Renaming the Factored dataset.
> colnames(pca_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
> head(pca_reg_no_rtt)
     Satisfaction    Factor1    Factor2    Factor3    Factor4
[1,]          8.2  0.2426689  0.9840535 -1.4862465  1.2369663
[2,]          5.7  0.2452695 -1.5475268 -1.5186060 -0.9357584
[3,]          8.9  1.3499046 -0.4740518 -0.1435113  1.1415100
[4,]          4.8 -1.1069113  0.1098079  1.4261590 -1.2131900
[5,]          7.1 -0.3187131 -0.8668559 -0.2027249  0.3380636
[6,]          4.7 -1.7049121  0.2294714 -1.1679476 -0.7306963
> pca_reg_no_rtt<- as.data.frame(pca_reg_no_rtt)
```

```
> pca_reg_no_rtt<- as.data.frame(pca_reg_no_rtt)
> set.seed(42)
> #Dividing the dataset into Test and Train.
> index_pca_no_rtt <- sample.split(pca_reg_no_rtt$Satisfaction,SplitRatio = .70)
> train_pca_no_rtt <- subset(pca_reg_no_rtt,index_pca_no_rtt ==TRUE)
> test_pca_no_rtt <- subset(pca_reg_no_rtt, index_pca_no_rtt == FALSE)
> #Building the Regression Model on the Train Data.
> model_pca_no_rtt<- lm(Satisfaction~.,data = train_pca_no_rtt)
> summary(model_pca_no_rtt)

Call:
lm(formula = Satisfaction ~ ., data = train_pca_no_rtt)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.1270 -0.3417  0.1013  0.3106  1.1879 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.92268    0.06081 113.842  < 2e-16 ***
Factor1      0.97570    0.05847  16.687  < 2e-16 ***
Factor2      0.05020    0.06200   0.810    0.421    
Factor3     -0.04506    0.06354  -0.709    0.481    
Factor4      0.41263    0.05923   6.967 1.86e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5109 on 66 degrees of freedom
Multiple R-squared:  0.8349,	Adjusted R-squared:  0.8249 
F-statistic: 83.46 on 4 and 66 DF,  p-value: < 2.2e-16
```

We found; the Value of Multiple R Square is 0.7014.

7.5.2 Validating the Model using Test Data.

Once the Model is built, we will Validate the Model using the Test data. This will help us understand if the Model is accurate, Overfitted or Underfitted.

```
#Validating the Model on Test Data.
pred_pca_no_rtt <- predict(model_pca_no_rtt,newdata = test_pca_no_rtt)
summary(pred_pca_no_rtt)

SST_no_rtt <- sum((test_pca_no_rtt$Satisfaction - mean(test_pca_no_rtt$Satisfaction))^2)
SSR_no_rtt <- sum((pred_pca_no_rtt - mean(test_pca_no_rtt$Satisfaction))^2)
SSE_no_rtt <- sum((test_pca_no_rtt$Satisfaction - pred_pca_no_rtt)^2)

calculated_Rsq_no_rtt <- 1-(SSE_no_rtt/SST_no_rtt)
calculated_Rsq_no_rtt
```

```
> #Validating the Model on Test Data.
> pred_pca_no_rtt <- predict(model_pca_no_rtt,newdata = test_pca_no_rtt)
> summary(pred_pca_no_rtt)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  3.977   6.241   6.903   6.783   7.400   9.445 
> SST_no_rtt <- sum((test_pca_no_rtt$Satisfaction - mean(test_pca_no_rtt$Satisfaction))^2)
> SSR_no_rtt <- sum((pred_pca_no_rtt - mean(test_pca_no_rtt$Satisfaction))^2)
> SSE_no_rtt <- sum((test_pca_no_rtt$Satisfaction - pred_pca_no_rtt)^2)
> calculated_Rsq_no_rtt <- 1-(SSE_no_rtt/SST_no_rtt)
> calculated_Rsq_no_rtt
[1] 0.4199801
```

Calculated R square value on Test is 0.4199.

Here, from the above validation, the Calculated value of R square is 0.4199, while the Derived value from the Model is 0.7014. Hence the model is Underfit. We will now perform Principal Component Analysis with Rotation as Varimax.

7.6 Performing Principal Component Analysis with Rotation

```
#Using Rotate = "Varimax".

pca_wt_rtt <- principal(dataset, nfactors = 4, rotate = "varimax")
pca_wt_rtt
```

```
> pca_wt_rtt <- principal(dataset, nfactors = 4, rotate = "varimax")
> pca_wt_rtt
Principal Components Analysis
Call: principal(r = dataset, nfactors = 4, rotate = "varimax")
Standardized loadings (pattern matrix) based upon correlation matrix
              RC1   RC2   RC3   RC4   h2    u2 com
ProdQual     0.00 -0.01 -0.03  0.88 0.77 0.232 1.0
Ecom         0.06  0.87  0.05 -0.12 0.78 0.223 1.1
TechSup      0.02 -0.02  0.94  0.10 0.89 0.107 1.0
CompRes      0.93  0.12  0.05  0.09 0.88 0.119 1.1
Advertising  0.14  0.74 -0.08  0.01 0.58 0.424 1.1
ProdLine     0.59 -0.06  0.15  0.64 0.79 0.213 2.1
SalesFImage  0.13  0.90  0.08 -0.16 0.86 0.141 1.1
ComPricing  -0.09  0.23 -0.25 -0.72 0.64 0.359 1.5
WartyClaim   0.11  0.05  0.93  0.10 0.89 0.108 1.1
OrdBilling   0.86  0.11  0.08  0.04 0.77 0.234 1.1
DelSpeed     0.94  0.18  0.00  0.05 0.91 0.086 1.1

                       RC1  RC2  RC3  RC4
SS loadings           2.89 2.23 1.86 1.77
Proportion Var        0.26 0.20 0.17 0.16
Cumulative Var        0.26 0.47 0.63 0.80
Proportion Explained  0.33 0.26 0.21 0.20
Cumulative Proportion 0.33 0.59 0.80 1.00

Mean item complexity =  1.2
Test of the hypothesis that 4 components are sufficient.

The root mean square of the residuals (RMSR) is  0.06
 with the empirical chi square  39.02  with prob <  0.0018

Fit based upon off diagonal values = 0.97
```

After performing Principal Factor Analysis, we observed that the cumulative Variance in the data is 80% which is the same as PCA without Rotation and the Communality(h2) for every individual variable have varied nominally. Let us proceed on the Model Building.

7.6.1 **Building Regression Model on PCA with Rotation**

```r
#Creating the Score Matrix.
pca_score_wt_rtt <- pca_wt_rtt$scores
head(pca_score_wt_rtt)

#Joining the Score Matrix with Original Dataset to get the Dependent Dataset
pca_reg_wt_rtt <- cbind(dataset1[,12],pca_score_wt_rtt)

#Renaming the Factored Dataset
colnames(pca_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
head(pca_reg_wt_rtt)

pca_reg_wt_rtt<- as.data.frame(pca_reg_wt_rtt)

set.seed(42)
#Dividing the dataset into Test and Train.
index_pca_wt_rtt <- sample.split(pca_reg_wt_rtt$Satisfaction,SplitRatio = .70)
train_pca_wt_rtt <- subset(pca_reg_wt_rtt,index_pca_wt_rtt ==TRUE)
test_pca_wt_rtt <- subset(pca_reg_wt_rtt, index_pca_wt_rtt == FALSE)

#Building the Model on Train Dataset
model_pca_wt_rtt<- lm(Satisfaction~.,data = train_pca_wt_rtt)
summary(model_pca_wt_rtt)
```

```
> #Creating the Score Matrix.
> pca_score_wt_rtt <- pca_wt_rtt$scores
> head(pca_score_wt_rtt)
            RC1        RC2        RC4         RC3
[1,]  0.1410926  0.8644269  0.5900387 -1.91075306
[2,]  1.1966316 -1.9588988  0.3387271 -0.50284798
[3,]  0.6381887  0.5559538  1.6290101  0.02376762
[4,] -0.8558873 -0.3604021 -1.4855540  1.29374124
[5,] -0.3258975 -0.7203627  0.6171453 -0.04900746
[6,] -0.6450724 -1.1379683 -1.2289660 -1.27904399
> #Joining the Score Matrix with Original Dataset to get the Dependent Dataset
> pca_reg_wt_rtt <- cbind(dataset1[,12],pca_score_wt_rtt)
> #Renaming the Factored Dataset
> colnames(pca_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
> head(pca_reg_wt_rtt)
     Satisfaction    Factor1    Factor2    Factor3     Factor4
[1,]          8.2  0.1410926  0.8644269  0.5900387 -1.91075306
[2,]          5.7  1.1966316 -1.9588988  0.3387271 -0.50284798
[3,]          8.9  0.6381887  0.5559538  1.6290101  0.02376762
[4,]          4.8 -0.8558873 -0.3604021 -1.4855540  1.29374124
[5,]          7.1 -0.3258975 -0.7203627  0.6171453 -0.04900746
[6,]          4.7 -0.6450724 -1.1379683 -1.2289660 -1.27904399

> pca_reg_wt_rtt<- as.data.frame(pca_reg_wt_rtt)
> set.seed(42)
> #Dividing the dataset into Test and Train.
> index_pca_wt_rtt <- sample.split(pca_reg_wt_rtt$Satisfaction,SplitRatio = .70)
> train_pca_wt_rtt <- subset(pca_reg_wt_rtt,index_pca_wt_rtt ==TRUE)
> test_pca_wt_rtt <- subset(pca_reg_wt_rtt, index_pca_wt_rtt == FALSE)
> #Building the Model on Train Dataset
> model_pca_wt_rtt<- lm(Satisfaction~.,data = train_pca_wt_rtt)
> summary(model_pca_wt_rtt)

Call:
lm(formula = Satisfaction ~ ., data = train_pca_wt_rtt)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.1270 -0.3417  0.1013  0.3106  1.1879 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.92268    0.06081 113.842  < 2e-16 ***
Factor1      0.62160    0.05864  10.600 6.89e-16 ***
Factor2      0.55967    0.05888   9.505 5.56e-14 ***
Factor3      0.65180    0.06140  10.615 6.51e-16 ***
Factor4      0.04842    0.06428   0.753    0.454    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5109 on 66 degrees of freedom
Multiple R-squared:  0.8349,	Adjusted R-squared:  0.8249 
F-statistic: 83.46 on 4 and 66 DF,  p-value: < 2.2e-16
```

From the Model, the Derived Value of R Square is 0.7014.

Now Validating the Model Built.

7.6.2 Validating the Model using Test Data.

```
#Validating the Model on Test Data.
pred_pca_wt_rtt <- predict(model_pca_wt_rtt,newdata = test_pca_wt_rtt)

summary(pred_pca_wt_rtt)
SST_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - mean(test_pca_wt_rtt$Satisfaction))^2)
SSR_wt_rtt <- sum((pred_pca_wt_rtt - mean(test_pca_wt_rtt$Satisfaction))^2)
SSE_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - pred_pca_wt_rtt)^2)

calculated_Rsq_wt_rtt <- 1-(SSE_wt_rtt/SST_wt_rtt)
calculated_Rsq_wt_rtt
```

```
> #Validating the Model on Test Data.
> pred_pca_wt_rtt <- predict(model_pca_wt_rtt,newdata = test_pca_wt_rtt)
> summary(pred_pca_wt_rtt)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  3.977   6.241   6.903   6.783   7.400   9.445 
> SST_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - mean(test_pca_wt_rtt$Satisfaction))^2)
> SSR_wt_rtt <- sum((pred_pca_wt_rtt - mean(test_pca_wt_rtt$Satisfaction))^2)
> SSE_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - pred_pca_wt_rtt)^2)
> calculated_Rsq_wt_rtt <- 1-(SSE_wt_rtt/SST_wt_rtt)
> calculated_Rsq_wt_rtt
[1] 0.4199801
```

Calculated Value of R Square is 0.4199

Here, from the above validation, the Calculated value of R square is 0.4199, while the Derived value from the Model is 0.7014. Hence the model is Underfit. Which is still the same as PCA without Rotation. We will now perform Factor Analysis without Rotation to check for the Significant Model.

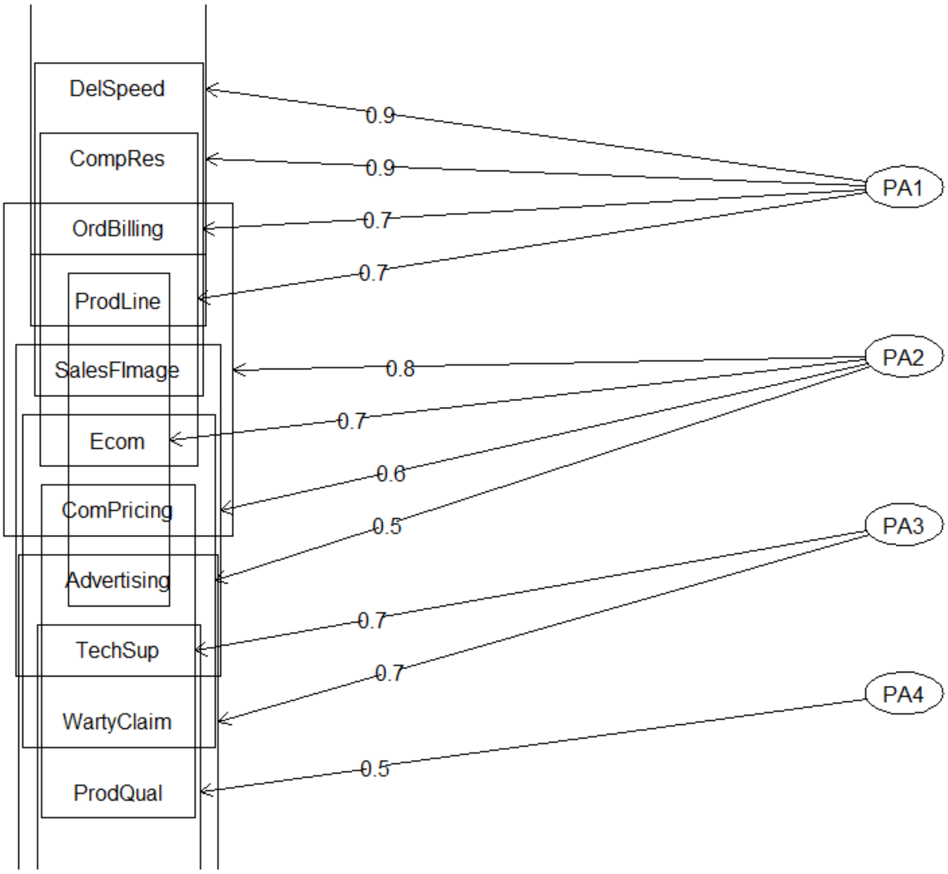
7.7 Performing Factor Analysis without Rotation

```
### Performing Factor Analysis.

#Using the Kizer Rule, we take the number of factors for which the eigen values are greater then 1.
#In this case, we have 4 values which are greater the 1, hence we take 4 factors for PCA.
#With Rotate = None

fa_no_rtt <- fa(dataset,nfactors = 4, rotate = "none", fm="pa")
fa_no_rtt

#Plotting the Factor Analysis
fa.diagram(fa_no_rtt)
```

```
> fa_no_rtt <- fa(dataset,nfactors = 4, rotate = "none", fm="pa")
> fa_no_rtt
Factor Analysis using method =  pa
Call: fa(r = dataset, nfactors = 4, rotate = "none", fm = "pa")
Standardized loadings (pattern matrix) based upon correlation matrix
              PA1   PA2   PA3   PA4   h2    u2 com
ProdQual     0.20 -0.41 -0.06  0.46 0.42 0.576 2.4
Ecom         0.29  0.66  0.27  0.22 0.64 0.362 2.0
TechSup      0.28 -0.38  0.74 -0.17 0.79 0.205 1.9
CompRes      0.86  0.01 -0.26 -0.18 0.84 0.157 1.3
Advertising  0.29  0.46  0.08  0.13 0.31 0.686 1.9
ProdLine     0.69 -0.45 -0.14  0.31 0.80 0.200 2.3
SalesFImage  0.39  0.80  0.35  0.25 0.98 0.021 2.1
ComPricing  -0.23  0.55 -0.04 -0.29 0.44 0.557 1.9
WartyClaim   0.38 -0.32  0.74 -0.15 0.81 0.186 2.0
OrdBilling   0.75  0.02 -0.18 -0.18 0.62 0.378 1.2
DelSpeed     0.90  0.10 -0.30 -0.20 0.94 0.058 1.4

                       PA1  PA2  PA3  PA4
SS loadings           3.21 2.22 1.50 0.68
Proportion Var        0.29 0.20 0.14 0.06
Cumulative Var        0.29 0.49 0.63 0.69
Proportion Explained  0.42 0.29 0.20 0.09
Cumulative Proportion 0.42 0.71 0.91 1.00

Mean item complexity =  1.9
Test of the hypothesis that 4 factors are sufficient.

The degrees of freedom for the null model are  55  and the objective function was  6.55 with Chi Square of  61
9.27
The degrees of freedom for the model are 17  and the objective function was  0.33

The root mean square of the residuals (RMSR) is  0.02
The df corrected root mean square of the residuals is  0.03

The harmonic number of observations is  100 with the empirical chi square  3.19  with prob <  1
The total number of observations was  100  with Likelihood Chi Square =  30.27  with prob <  0.024

Tucker Lewis Index of factoring reliability =  0.921
RMSEA index =  0.096  and the 90 % confidence intervals are  0.032 0.139
BIC =  -48.01
Fit based upon off diagonal values = 1
Measures of factor score adequacy
                                                  PA1  PA2  PA3  PA4
Correlation of (regression) scores with factors  0.98 0.97 0.95 0.88
Multiple R square of scores with factors         0.96 0.95 0.91 0.78
Minimum correlation of possible factor scores    0.92 0.90 0.82 0.56
```

After performing Factor Analysis, we observed that the cumulative Variance in the data is 69% and the Communality(h2) for every individual variable is high. Let us proceed on the Model Building.

Factor Analysis Plot:



7.7.1 Building Regression Model on Factor Analysis without Rotation

```
#Creating the Score Matrix
fa_score_no_rtt <- fa_no_rtt$scores
head(fa_score_no_rtt)

#Joining the Score Matrix with the Original dataset to get the Satisfaction variable
fa_reg_no_rtt <- cbind(dataset1[,12],fa_score_no_rtt)

#Renaming the Factored dataset
colnames(fa_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
head(fa_reg_no_rtt)

fa_reg_no_rtt<- as.data.frame(fa_reg_no_rtt)

set.seed(42)

#Dividing the Dataset to Test and Train Data
index <- sample.split(fa_reg_no_rtt$Satisfaction,SplitRatio = .70)
train_fa_no_rtt <- subset(fa_reg_no_rtt,index ==TRUE)
test_fa_no_rtt <- subset(fa_reg_no_rtt, index == FALSE)

#Building the Regression Model on Train Data
model_fa_no_rtt<- lm(Satisfaction~.,data = train_fa_no_rtt)
summary(model_fa_no_rtt)
```

```
> #Creating the Score Matrix
> fa_score_no_rtt <- fa_no_rtt$scores
> head(fa_score_no_rtt)
            PA1         PA2         PA3        PA4
[1,] -0.2367381  1.23998570 -1.12891346  0.9783118
[2,]  0.7704292 -1.70153639 -1.84032527 -0.7780463
[3,]  1.0117487 -0.09673524  0.06374278  1.2986265
[4,] -1.0930182 -0.49877402  1.38672297 -0.6291745
[5,] -0.4156728 -0.57647133 -0.01325216  0.3718166
[6,] -1.5462133 -0.08944095 -1.17590551 -0.7689481
> #Joining the Score Matrix with the Original dataset to get the Satisfaction variable
> fa_reg_no_rtt <- cbind(dataset1[,12],fa_score_no_rtt)
> #Renaming the Factored dataset
> colnames(fa_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
> head(fa_reg_no_rtt)
     Satisfaction    Factor1     Factor2     Factor3    Factor4
[1,]          8.2 -0.2367381  1.23998570 -1.12891346  0.9783118
[2,]          5.7  0.7704292 -1.70153639 -1.84032527 -0.7780463
[3,]          8.9  1.0117487 -0.09673524  0.06374278  1.2986265
[4,]          4.8 -1.0930182 -0.49877402  1.38672297 -0.6291745
[5,]          7.1 -0.4156728 -0.57647133 -0.01325216  0.3718166
[6,]          4.7 -1.5462133 -0.08944095 -1.17590551 -0.7689481
```

```
> fa_reg_no_rtt<- as.data.frame(fa_reg_no_rtt)
> set.seed(42)
> #Dividing the Dataset to Test and Train Data
> index <- sample.split(fa_reg_no_rtt$Satisfaction,SplitRatio = .70)
> train_fa_no_rtt <- subset(fa_reg_no_rtt,index ==TRUE)
> test_fa_no_rtt <- subset(fa_reg_no_rtt, index == FALSE)
> #Building the Regression Model on Train Data
> model_fa_no_rtt<- lm(Satisfaction~.,data = train_fa_no_rtt)
> summary(model_fa_no_rtt)

Call:
lm(formula = Satisfaction ~ ., data = train_fa_no_rtt)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.7245 -0.4309  0.1096  0.4250  1.1092 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.90291    0.07570  91.182  < 2e-16 ***
Factor1      0.90488    0.07467  12.119  < 2e-16 ***
Factor2      0.15522    0.08073   1.923   0.0588 .  
Factor3      0.03930    0.08277   0.475   0.6365    
Factor4      0.51205    0.08462   6.051 7.55e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6378 on 66 degrees of freedom
Multiple R-squared:  0.7427,	Adjusted R-squared:  0.7271 
F-statistic: 47.63 on 4 and 66 DF,  p-value: < 2.2e-16
```

Derived R Square Value from the Model built on Factor Analysis without Rotation is .7427

7.7.2 Validating the Model using Test Data.

```
#Validating the Model on Test Data
pred_fa_no_rtt <- predict(model_fa_no_rtt,newdata = test_fa_no_rtt)
summary(pred_fa_no_rtt)

SST_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - mean(test_pca_no_rtt$Satisfaction))^2)
SSR_fa_no_rtt <- sum((pred_fa_no_rtt - mean(test_fa_no_rtt$Satisfaction))^2)
SSE_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - pred_fa_no_rtt)^2)

calculated_Rsq_fa_no_rtt <- 1-(SSE_fa_no_rtt/SST_fa_no_rtt)
calculated_Rsq_fa_no_rtt
```

```
> #Validating the Model on Test Data
> pred_fa_no_rtt <- predict(model_fa_no_rtt,newdata = test_fa_no_rtt)
> summary(pred_fa_no_rtt)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  4.689   6.488   6.879   6.879   7.372   8.661 
> SST_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - mean(test_pca_no_rtt$Satisfaction))^2)
> SSR_fa_no_rtt <- sum((pred_fa_no_rtt - mean(test_fa_no_rtt$Satisfaction))^2)
> SSE_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - pred_fa_no_rtt)^2)
> calculated_Rsq_fa_no_rtt <- 1-(SSE_fa_no_rtt/SST_fa_no_rtt)
> calculated_Rsq_fa_no_rtt
[1] 0.5601423
```

Calculated R Square value from the Test Data is .5601.

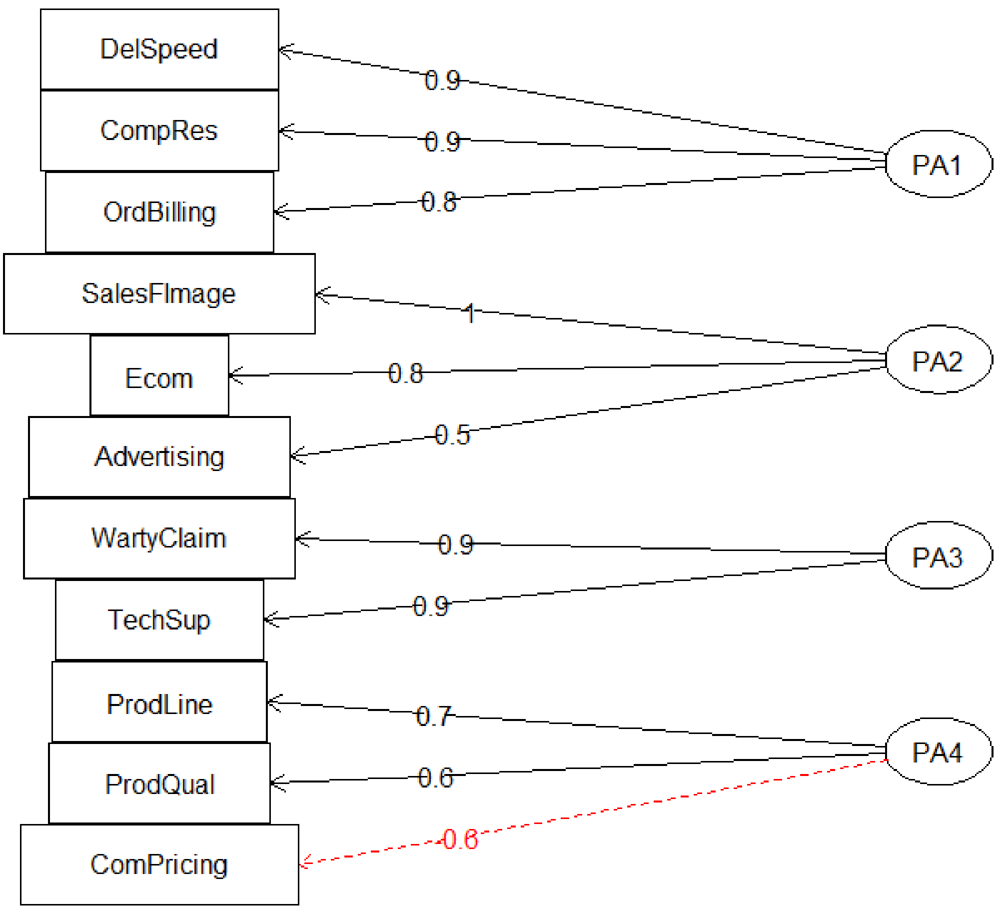
The Model created on Factor Analysis without Rotation is still Underfit as the Calculated Value is 0.5601 whereas the Derived value is 0.7427. Now we will perform Factor Analysis with Rotation as Varimax to check for the variation.

7.8 Performing Factor Analysis with Rotation

```
#Using the Keiser Law.
#In this case, we have 4 values which are greater the 1, hence we take 4 factors for FA.
#With rotate = Varimax
fa_wt_rtt <- fa(dataset,nfactors =4, rotate = "varimax", fm="pa")
fa_wt_rtt

fa.diagram(fa_wt_rtt)
```

```
> #Using the Keiser Law.
> #In this case, we have 4 values which are greater the 1, hence we take 4 factors for FA.
> #With rotate = Varimax
> fa_wt_rtt <- fa(dataset,nfactors =4, rotate = "varimax", fm="pa")
> fa_wt_rtt
Factor Analysis using method =  pa
Call: fa(r = dataset, nfactors = 4, rotate = "varimax", fm = "pa")
Standardized loadings (pattern matrix) based upon correlation matrix
              PA1   PA2   PA3   PA4   h2    u2 com
ProdQual     0.02 -0.07  0.02  0.65 0.42 0.576 1.0
Ecom         0.07  0.79  0.03 -0.11 0.64 0.362 1.1
TechSup      0.02 -0.03  0.88  0.12 0.79 0.205 1.0
CompRes      0.90  0.13  0.05  0.13 0.84 0.157 1.1
Advertising  0.17  0.53 -0.04 -0.06 0.31 0.686 1.2
ProdLine     0.53 -0.04  0.13  0.71 0.80 0.200 1.9
SalesFImage  0.12  0.97  0.06 -0.13 0.98 0.021 1.1
ComPricing  -0.08  0.21 -0.21 -0.59 0.44 0.557 1.6
WartyClaim   0.10  0.06  0.89  0.13 0.81 0.186 1.1
OrdBilling   0.77  0.13  0.09  0.09 0.62 0.378 1.1
DelSpeed     0.95  0.19  0.00  0.09 0.94 0.058 1.1

                       PA1  PA2  PA3  PA4
SS loadings           2.63 1.97 1.64 1.37
Proportion Var        0.24 0.18 0.15 0.12
Cumulative Var        0.24 0.42 0.57 0.69
Proportion Explained  0.35 0.26 0.22 0.18
Cumulative Proportion 0.35 0.60 0.82 1.00

Mean item complexity =  1.2
Test of the hypothesis that 4 factors are sufficient.

The degrees of freedom for the null model are  55  and the objective function was  6.55 with Chi Square of  619.27
The degrees of freedom for the model are 17  and the objective function was  0.33 

The root mean square of the residuals (RMSR) is  0.02 
The df corrected root mean square of the residuals is  0.03 

The harmonic number of observations is  100 with the empirical chi square  3.19  with prob <  1 
The total number of observations was  100  with Likelihood Chi Square =  30.27  with prob <  0.024 

Tucker Lewis Index of factoring reliability =  0.921
RMSEA index =  0.096  and the 90 % confidence intervals are  0.032 0.139
BIC =  -48.01
Fit based upon off diagonal values = 1
Measures of factor score adequacy             
                                                 PA1  PA2  PA3  PA4
```

After performing Factor Analysis, we observed that the cumulative Variance in the data is 69% and the Communality(h2) for every individual variable is high. Let us proceed on the Model Building.



7.8.1 Building Regression Model on Factor Analysis without Rotation

```
#Creating the Score Matrix
fa_score_wt_rtt <- fa_wt_rtt$scores
head(fa_score_wt_rtt)

#Joining the Score Matrix with the Original dataset to get the Satisfaction variable
fa_reg_wt_rtt <- cbind(dataset1[,12],fa_score_wt_rtt)

#Renaming the Factored Dataset
colnames(fa_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
head(fa_reg_wt_rtt)

fa_reg_wt_rtt<- as.data.frame(fa_reg_wt_rtt)

#Splitting the data into Test and Train.
set.seed(42)
index <- sample.split(fa_reg_wt_rtt$Satisfaction,SplitRatio = .70)
train_fa_wt_rtt <- subset(fa_reg_wt_rtt,index ==TRUE)
test_fa_wt_rtt <- subset(fa_reg_wt_rtt, index == FALSE)

#Building the Model
model_fa_wt_rtt<- lm(Satisfaction~.,data = train_fa_wt_rtt)
summary(model_fa_wt_rtt)
```

```
> #Creating the Score Matrix
> fa_score_wt_rtt <- fa_wt_rtt$scores
> head(fa_score_wt_rtt)
            PA1        PA2          PA3         PA4
[1,] -0.1338871  0.9175166 -1.719604873  0.09135411
[2,]  1.6297604 -2.0090053 -0.596361722  0.65808192
[3,]  0.3637658  0.8361736  0.002979966  1.37548765
[4,] -1.2225230 -0.5491336  1.245473305 -0.64421384
[5,] -0.4854209 -0.4276223 -0.026980304  0.47360747
[6,] -0.5950924 -1.3035333 -1.183019401 -0.95913571
> #Joining the Score Matrix with the Original dataset to get the Satisfaction variable
> fa_reg_wt_rtt <- cbind(dataset1[,12],fa_score_wt_rtt)
> #Renaming the Factored Dataset
> colnames(fa_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3", "Factor4")
> head(fa_reg_wt_rtt)
     Satisfaction    Factor1    Factor2      Factor3     Factor4
[1,]          8.2 -0.1338871  0.9175166 -1.719604873  0.09135411
[2,]          5.7  1.6297604 -2.0090053 -0.596361722  0.65808192
[3,]          8.9  0.3637658  0.8361736  0.002979966  1.37548765
[4,]          4.8 -1.2225230 -0.5491336  1.245473305 -0.64421384
[5,]          7.1 -0.4854209 -0.4276223 -0.026980304  0.47360747
[6,]          4.7 -0.5950924 -1.3035333 -1.183019401 -0.95913571
```

```
> fa_reg_wt_rtt<- as.data.frame(fa_reg_wt_rtt)
> #Splitting the data into Test and Train.
> set.seed(42)
> index <- sample.split(fa_reg_wt_rtt$Satisfaction,SplitRatio = .70)
> train_fa_wt_rtt <- subset(fa_reg_wt_rtt,index ==TRUE)
> test_fa_wt_rtt <- subset(fa_reg_wt_rtt, index == FALSE)
> #Building the Model
> model_fa_wt_rtt<- lm(Satisfaction~.,data = train_fa_wt_rtt)
> summary(model_fa_wt_rtt)

Call:
lm(formula = Satisfaction ~ ., data = train_fa_wt_rtt)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.7245 -0.4309  0.1096  0.4250  1.1092 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.90291    0.07570  91.182  < 2e-16 ***
Factor1      0.56931    0.07447   7.645 1.14e-10 ***
Factor2      0.65081    0.07723   8.427 4.57e-12 ***
Factor3      0.08159    0.08479   0.962    0.339    
Factor4      0.59355    0.08605   6.898 2.46e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6378 on 66 degrees of freedom
Multiple R-squared:  0.7427,	Adjusted R-squared:  0.7271 
F-statistic: 47.63 on 4 and 66 DF,  p-value: < 2.2e-16
```

7.8.2 Validating the Model using Test Data.

```
#Validating the Model on the Test Data.
pred_fa_wt_rtt <- predict(model_fa_wt_rtt,newdata = test_fa_wt_rtt)
summary(pred_fa_wt_rtt)
SST_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - mean(test_pca_wt_rtt$Satisfaction))^2)
SSR_fa_wt_rtt <- sum((pred_fa_wt_rtt - mean(test_fa_wt_rtt$Satisfaction))^2)
SSE_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - pred_fa_wt_rtt)^2)

calculated_Rsq_fa_wt_rtt <- 1-(SSE_fa_wt_rtt/SST_fa_wt_rtt)
calculated_Rsq_fa_no_rtt
```

```
> #Validating the Model on the Test Data.
> pred_fa_wt_rtt <- predict(model_fa_wt_rtt,newdata = test_fa_wt_rtt)
> summary(pred_fa_wt_rtt)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  4.689   6.488   6.879   6.879   7.372   8.661
> SST_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - mean(test_fa_wt_rtt$Satisfaction))^2)
> SSR_fa_wt_rtt <- sum((pred_fa_wt_rtt - mean(test_fa_wt_rtt$Satisfaction))^2)
> SSE_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - pred_fa_wt_rtt)^2)
> calculated_Rsq_fa_wt_rtt <- 1-(SSE_fa_wt_rtt/SST_fa_wt_rtt)
> calculated_Rsq_fa_no_rtt
[1] 0.5601423
```

Calculated R Square value from the Model based on FA with Rotation is .5601

The Model is the same as Factor Analysis without Rotation as the Calculated R Square is 0.5601 and Derived R Square is 0.7427.

7.9 Summary

We created 4 models on the basis of the below criteria.

- Principal Component Analysis without Rotation.
- Principal Component Analysis with Rotation as Varimax.
- Factor Analysis without Rotation.
- Factor Analysis with Rotation as varimax.

After Analyzing the 4 models build on the above criteria, we can conclude that the Model Built on FA with Rotation is the best fit Model with Derived Multiple R Square value of 0.7427 and Predicted value of Multiple R square is 0.5601, which states that with a change of 1 unit in data there will a variation of 69.30%. This Model can be interpreted as an underfitted Model.as the Derived R Square value is greater than Calculated R Square Value.

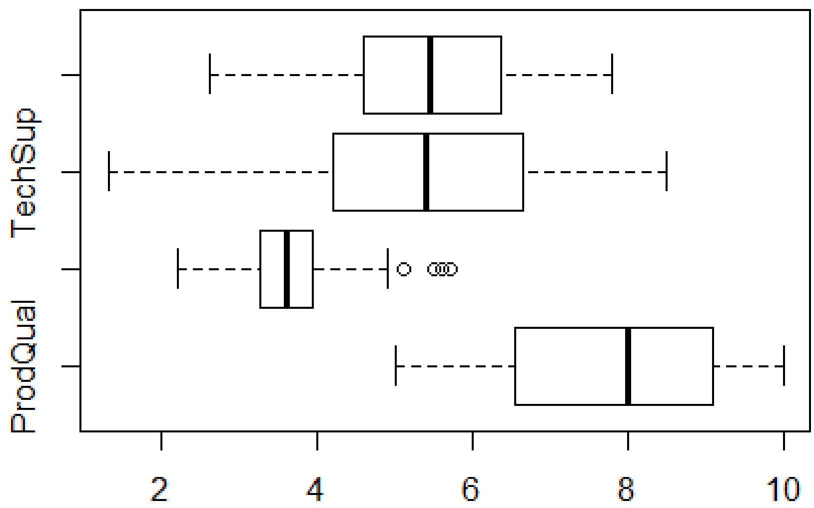
8 Appendix A – Source Code

```
############ Mini Project 2 for Regression Model Factor Analysis

#Installing the required Packages.

library(psych)

library(corrplot)

## corrplot 0.84 loaded

library(caTools)

library(DataExplorer)

library(devtools)

## Loading required package: usethis

library(ggbiplot)

## Loading required package: ggplot2
```

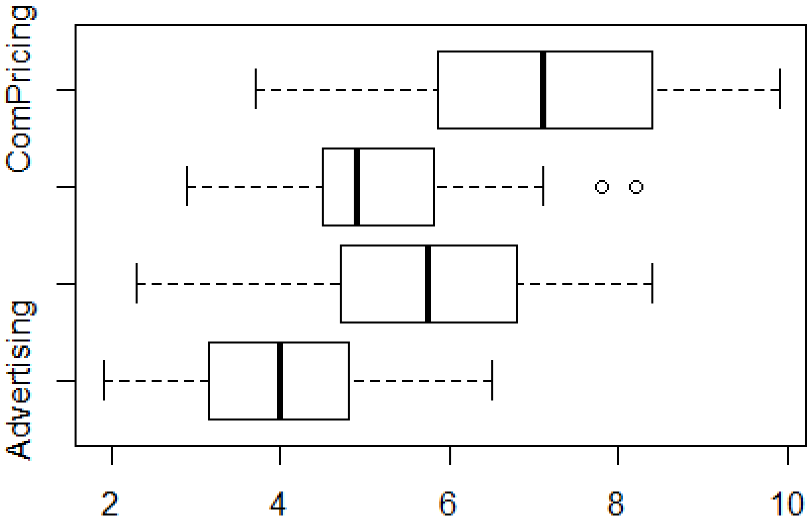
```
##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha

## Loading required package: plyr

## Loading required package: scales

##
## Attaching package: 'scales'

## The following objects are masked from 'package:psych':
##
##     alpha, rescale

## Loading required package: grid

# Setting up Working Directory.

setwd("D:/Great Learning/Project 2")

#Reading the file to R.

dataset <- read.csv("Factor-Hair-Revised.csv")
str(dataset)

## 'data.frame':    100 obs. of  13 variables:
##  $ ID          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ProdQual    : num  8.5 8.2 9.2 6.4 9 6.5 6.9 6.2 5.8 6.4 ...
##  $ Ecom        : num  3.9 2.7 3.4 3.3 3.4 2.8 3.7 3.3 3.6 4.5 ...
##  $ TechSup     : num  2.5 5.1 5.6 7 5.2 3.1 5 3.9 5.1 5.1 ...
##  $ CompRes     : num  5.9 7.2 5.6 3.7 4.6 4.1 2.6 4.8 6.7 6.1 ...
##  $ Advertising : num  4.8 3.4 5.4 4.7 2.2 4 2.1 4.6 3.7 4.7 ...
##  $ ProdLine    : num  4.9 7.9 7.4 4.7 6 4.3 2.3 3.6 5.9 5.7 ...
##  $ SalesFImage : num  6 3.1 5.8 4.5 4.5 3.7 5.4 5.1 5.8 5.7 ...
##  $ ComPricing  : num  6.8 5.3 4.5 8.8 6.8 8.5 8.9 6.9 9.3 8.4 ...
##  $ WartyClaim  : num  4.7 5.5 6.2 7 6.1 5.1 4.8 5.4 5.9 5.4 ...
##  $ OrdBilling  : num  5 3.9 5.4 4.3 4.5 3.6 2.1 4.3 4.4 4.1 ...
##  $ DelSpeed    : num  3.7 4.9 4.5 3 3.5 3.3 2 3.7 4.6 4.4 ...
##  $ Satisfaction: num  8.2 5.7 8.9 4.8 7.1 4.7 5.7 6.3 7 5.5 ...

head(dataset)

##   ID ProdQual Ecom TechSup CompRes Advertising ProdLine SalesFImage
## 1  1      8.5  3.9     2.5     5.9         4.8      4.9         6.0
## 2  2      8.2  2.7     5.1     7.2         3.4      7.9         3.1
## 3  3      9.2  3.4     5.6     5.6         5.4      7.4         5.8
## 4  4      6.4  3.3     7.0     3.7         4.7      4.7         4.5
## 5  5      9.0  3.4     5.2     4.6         2.2      6.0         4.5
## 6  6      6.5  2.8     3.1     4.1         4.0      4.3         3.7
##   ComPricing WartyClaim OrdBilling DelSpeed Satisfaction
## 1        6.8        4.7        5.0      3.7          8.2
## 2        5.3        5.5        3.9      4.9          5.7
## 3        4.5        6.2        5.4      4.5          8.9
## 4        8.8        7.0        4.3      3.0          4.8
## 5        6.8        6.1        4.5      3.5          7.1
## 6        8.5        5.1        3.6      3.3          4.7
```

```r
dataset <- dataset[,-1]

#Creating a replica od dataset for future use
dataset1 <- dataset

### Performing Exploratory Data Analysis

#Checking for the data load efficiency
head(dataset1, n= 10)

##    ProdQual Ecom TechSup CompRes Advertising ProdLine SalesFImage
## 1       8.5  3.9     2.5     5.9         4.8      4.9         6.0
## 2       8.2  2.7     5.1     7.2         3.4      7.9         3.1
## 3       9.2  3.4     5.6     5.6         5.4      7.4         5.8
## 4       6.4  3.3     7.0     3.7         4.7      4.7         4.5
## 5       9.0  3.4     5.2     4.6         2.2      6.0         4.5
## 6       6.5  2.8     3.1     4.1         4.0      4.3         3.7
## 7       6.9  3.7     5.0     2.6         2.1      2.3         5.4
## 8       6.2  3.3     3.9     4.8         4.6      3.6         5.1
## 9       5.8  3.6     5.1     6.7         3.7      5.9         5.8
## 10      6.4  4.5     5.1     6.1         4.7      5.7         5.7
##    ComPricing WartyClaim OrdBilling DelSpeed Satisfaction
## 1         6.8        4.7        5.0      3.7          8.2
## 2         5.3        5.5        3.9      4.9          5.7
## 3         4.5        6.2        5.4      4.5          8.9
## 4         8.8        7.0        4.3      3.0          4.8
## 5         6.8        6.1        4.5      3.5          7.1
## 6         8.5        5.1        3.6      3.3          4.7
## 7         8.9        4.8        2.1      2.0          5.7
## 8         6.9        5.4        4.3      3.7          6.3
## 9         9.3        5.9        4.4      4.6          7.0
## 10        8.4        5.4        4.1      4.4          5.5

# Checking for the Structure of the Dataset
str(dataset)

## 'data.frame':    100 obs. of  12 variables:
##  $ ProdQual    : num  8.5 8.2 9.2 6.4 9 6.5 6.9 6.2 5.8 6.4 ...
##  $ Ecom        : num  3.9 2.7 3.4 3.3 3.4 2.8 3.7 3.3 3.6 4.5 ...
##  $ TechSup     : num  2.5 5.1 5.6 7 5.2 3.1 5 3.9 5.1 5.1 ...
##  $ CompRes     : num  5.9 7.2 5.6 3.7 4.6 4.1 2.6 4.8 6.7 6.1 ...
##  $ Advertising : num  4.8 3.4 5.4 4.7 2.2 4 2.1 4.6 3.7 4.7 ...
##  $ ProdLine    : num  4.9 7.9 7.4 4.7 6 4.3 2.3 3.6 5.9 5.7 ...
##  $ SalesFImage : num  6 3.1 5.8 4.5 4.5 3.7 5.4 5.1 5.8 5.7 ...
##  $ ComPricing  : num  6.8 5.3 4.5 8.8 6.8 8.5 8.9 6.9 9.3 8.4 ...
##  $ WartyClaim  : num  4.7 5.5 6.2 7 6.1 5.1 4.8 5.4 5.9 5.4 ...
##  $ OrdBilling  : num  5 3.9 5.4 4.3 4.5 3.6 2.1 4.3 4.4 4.1 ...
##  $ DelSpeed    : num  3.7 4.9 4.5 3 3.5 3.3 2 3.7 4.6 4.4 ...
##  $ Satisfaction: num  8.2 5.7 8.9 4.8 7.1 4.7 5.7 6.3 7 5.5 ...

# Checking for the summary of the Dataset
summary(dataset)

##     ProdQual           Ecom          TechSup         CompRes     
##  Min.   : 5.000   Min.   :2.200   Min.   :1.300   Min.   :2.600  
##  1st Qu.: 6.575   1st Qu.:3.275   1st Qu.:4.250   1st Qu.:4.600  
##  Median : 8.000   Median :3.600   Median :5.400   Median :5.450  
##  Mean   : 7.810   Mean   :3.672   Mean   :5.365   Mean   :5.442  
##  3rd Qu.: 9.100   3rd Qu.:3.925   3rd Qu.:6.625   3rd Qu.:6.325  
##  Max.   :10.000   Max.   :5.700   Max.   :8.500   Max.   :7.800  
```

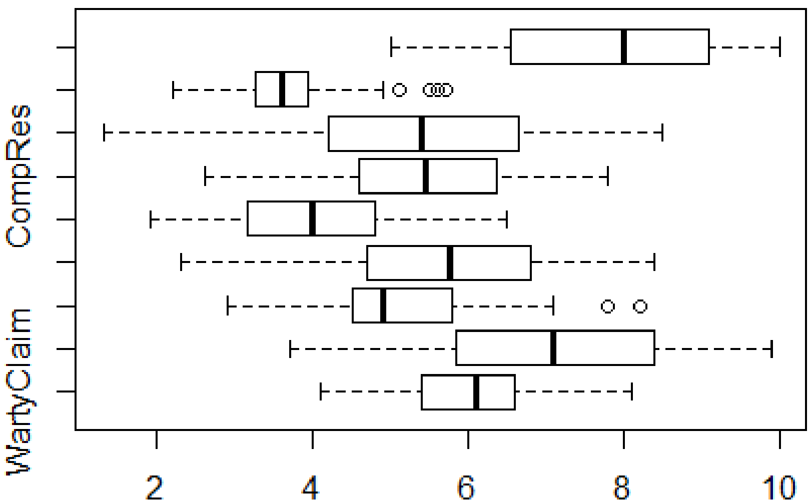
```
##   Advertising       ProdLine       SalesFImage      ComPricing
##  Min.   :1.900   Min.   :2.300   Min.   :2.900   Min.   :3.700
##  1st Qu.:3.175   1st Qu.:4.700   1st Qu.:4.500   1st Qu.:5.875
##  Median :4.000   Median :5.750   Median :4.900   Median :7.100
##  Mean   :4.010   Mean   :5.805   Mean   :5.123   Mean   :6.974
##  3rd Qu.:4.800   3rd Qu.:6.800   3rd Qu.:5.800   3rd Qu.:8.400
##  Max.   :6.500   Max.   :8.400   Max.   :8.200   Max.   :9.900
##    WartyClaim      OrdBilling       DelSpeed      Satisfaction
##  Min.   :4.100   Min.   :2.000   Min.   :1.600   Min.   :4.700
##  1st Qu.:5.400   1st Qu.:3.700   1st Qu.:3.400   1st Qu.:6.000
##  Median :6.100   Median :4.400   Median :3.900   Median :7.050
##  Mean   :6.043   Mean   :4.278   Mean   :3.886   Mean   :6.918
##  3rd Qu.:6.600   3rd Qu.:4.800   3rd Qu.:4.425   3rd Qu.:7.625
##  Max.   :8.100   Max.   :6.700   Max.   :5.500   Max.   :9.900

# Looking for the Null values in the Dataset
is.null(dataset)

## [1] FALSE

### Plotting the dataset

#Plotting the dataset to identify the Outliers.
#Plot2
boxplot(dataset[,1:4],horizontal = TRUE)
```



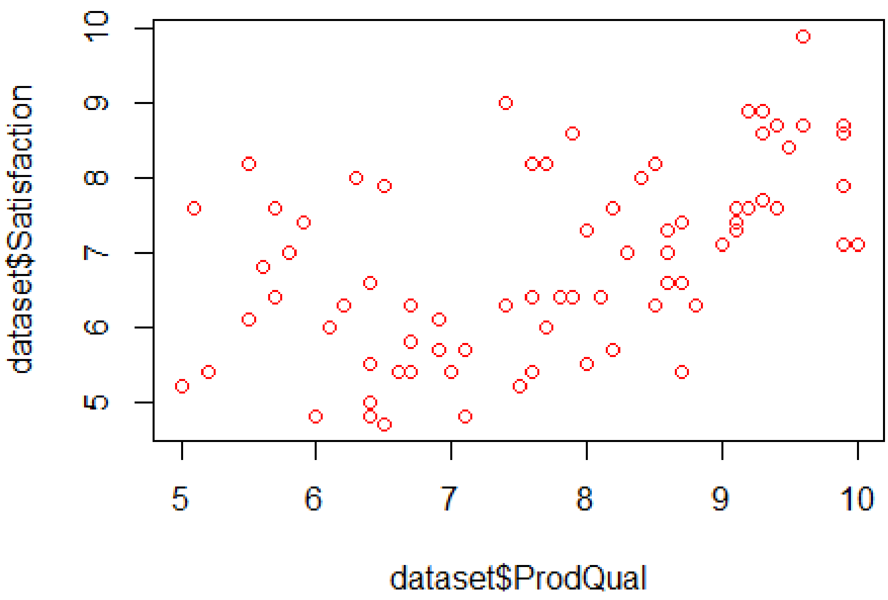

```
#Plot2
boxplot(dataset[,5:8],horizontal = TRUE)
```



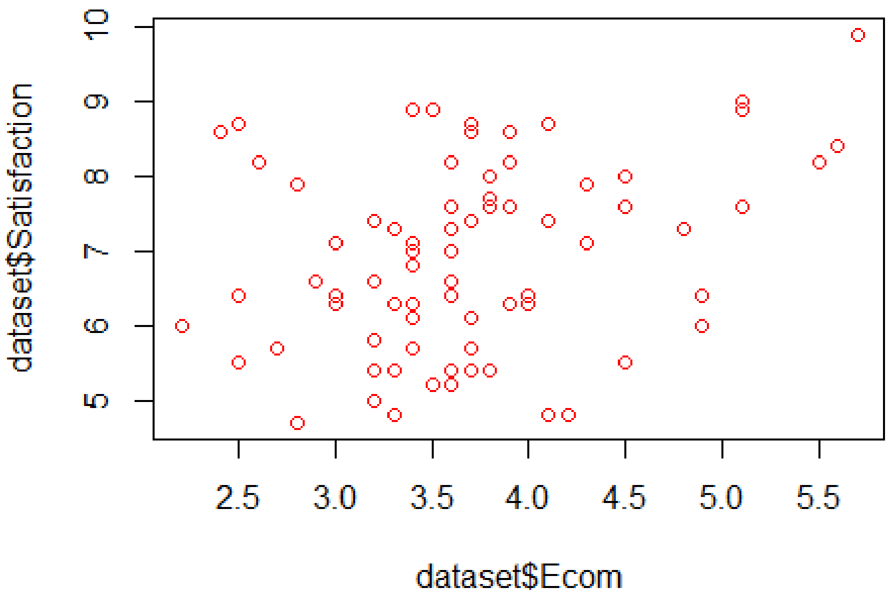

```
#Plot3
boxplot(dataset[,9:1],horizontal = TRUE)
```



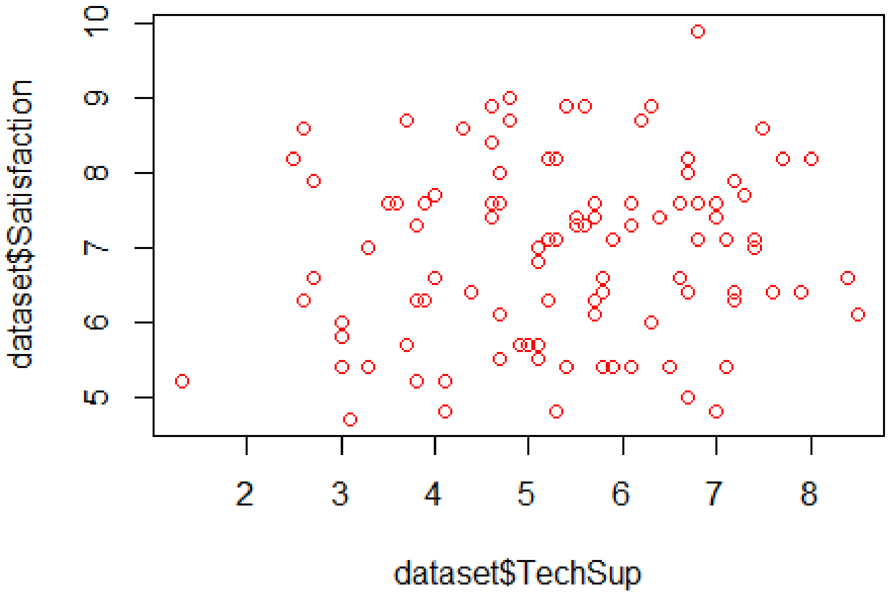

```
###Multivariate Analysis with the Independent variables on
###X axis and DependentVariable on Y axis.

#Plot: ProdQual vs Satisfaction.
plot(dataset$ProdQual,dataset$Satisfaction,col = "Red" )
```



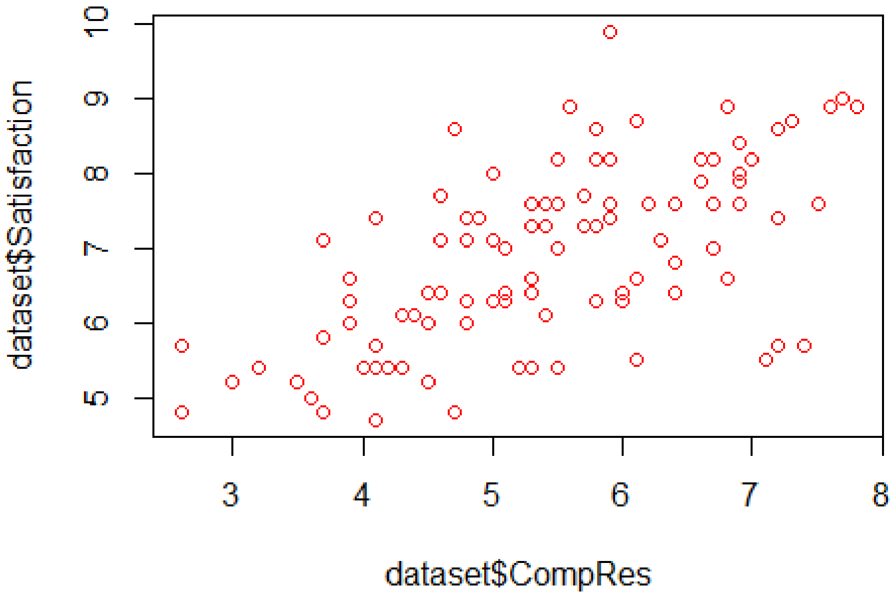

```
#Plot: Ecom vs Satisfaction.
plot(dataset$Ecom,dataset$Satisfaction,col = "Red")
```



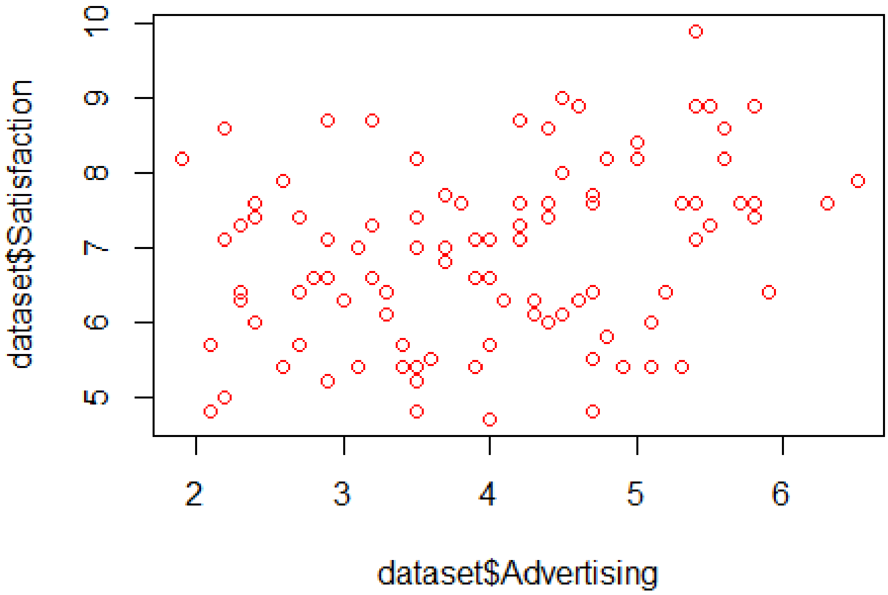

```
#Plot: TechSup vs Satisfaction.
plot(dataset$TechSup,dataset$Satisfaction,col = "Red")
```



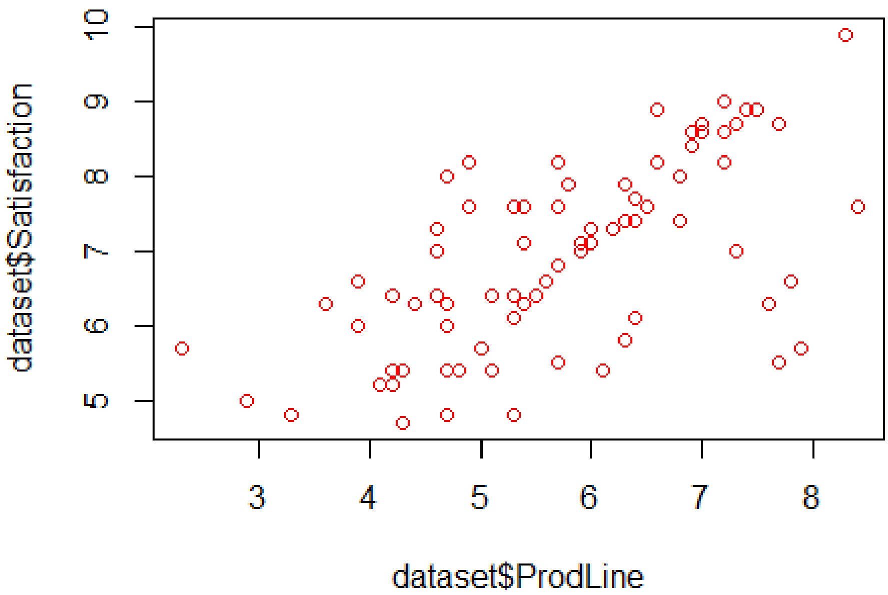

```
#Plot: CompRes vs Satisfaction.
plot(dataset$CompRes,dataset$Satisfaction,col = "Red")
```




```
#Plot: Advertising vs Satisfaction.
plot(dataset$Advertising,dataset$Satisfaction,col = "Red")
```

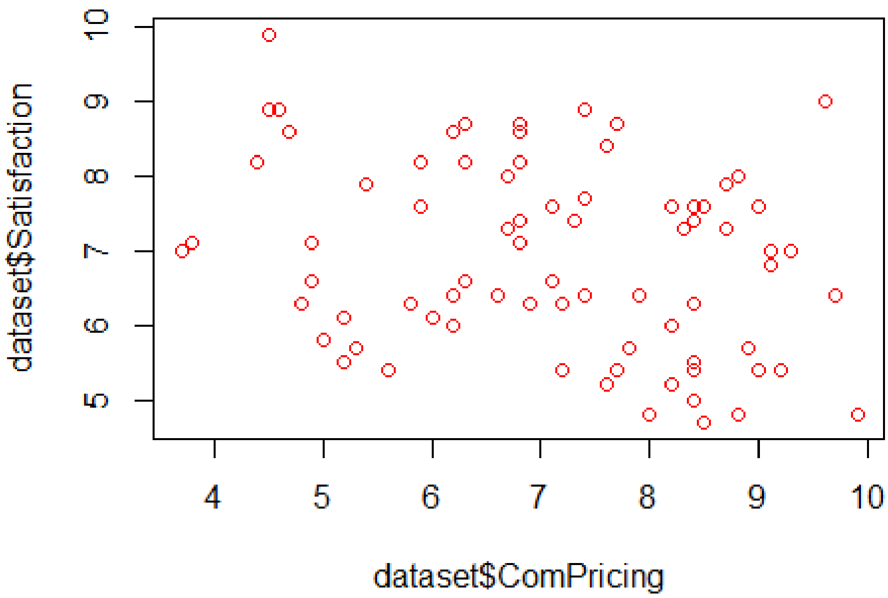



```r
#Plot: ProdLine vs Satisfaction.
plot(dataset$ProdLine,dataset$Satisfaction,col = "Red")
```

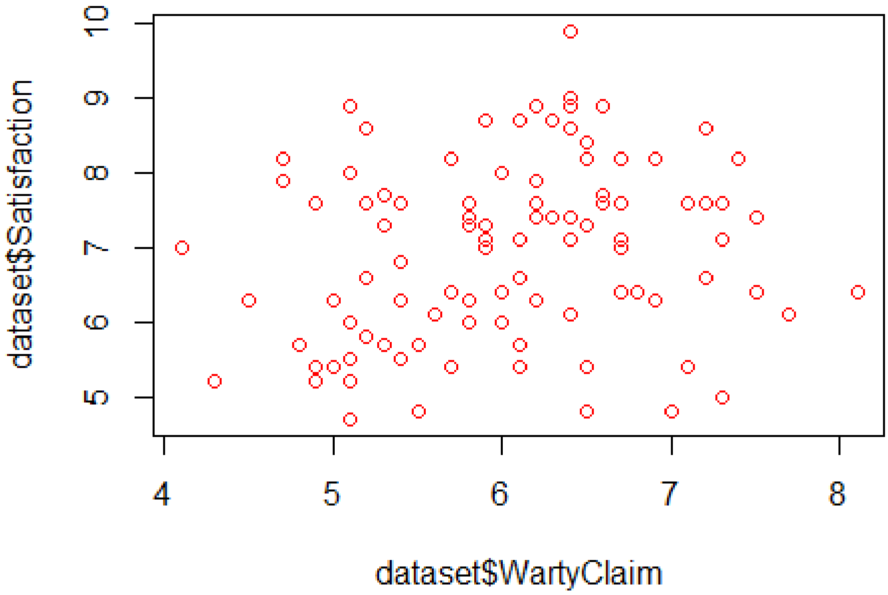



```r
#Plot: SalesFImage vs Satisfaction.
plot(dataset$SalesFImage,dataset$Satisfaction,col = "Red")
```

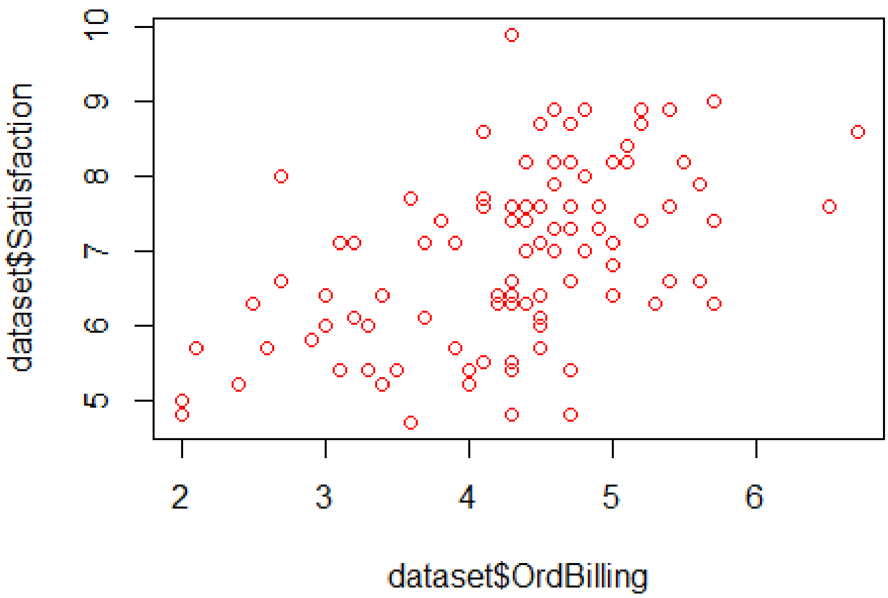

```
#Plot: ComPricing vs Satisfaction.
plot(dataset$ComPricing,dataset$Satisfaction,col = "Red")
```



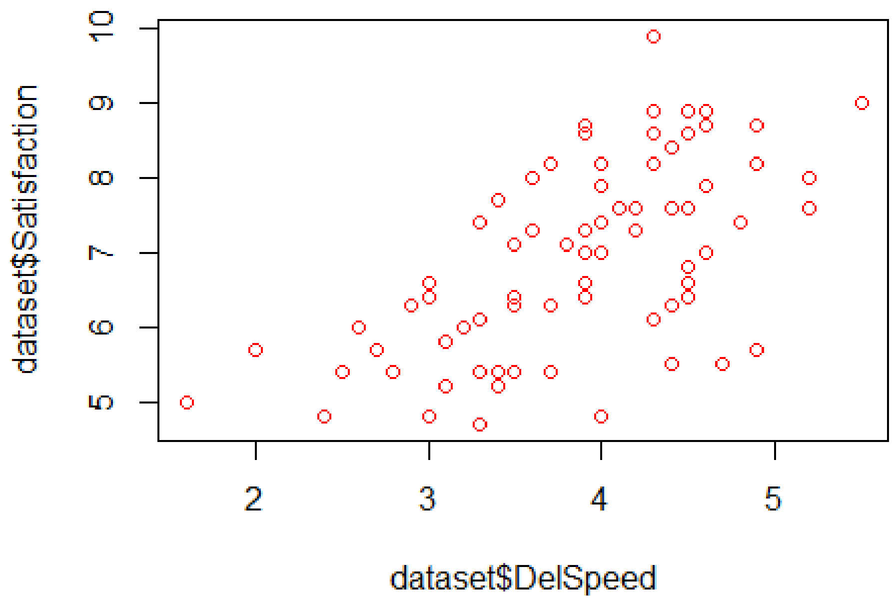

```
#Plot: WartyClaim vs Satisfaction.
plot(dataset$WartyClaim,dataset$Satisfaction,col = "Red")
```



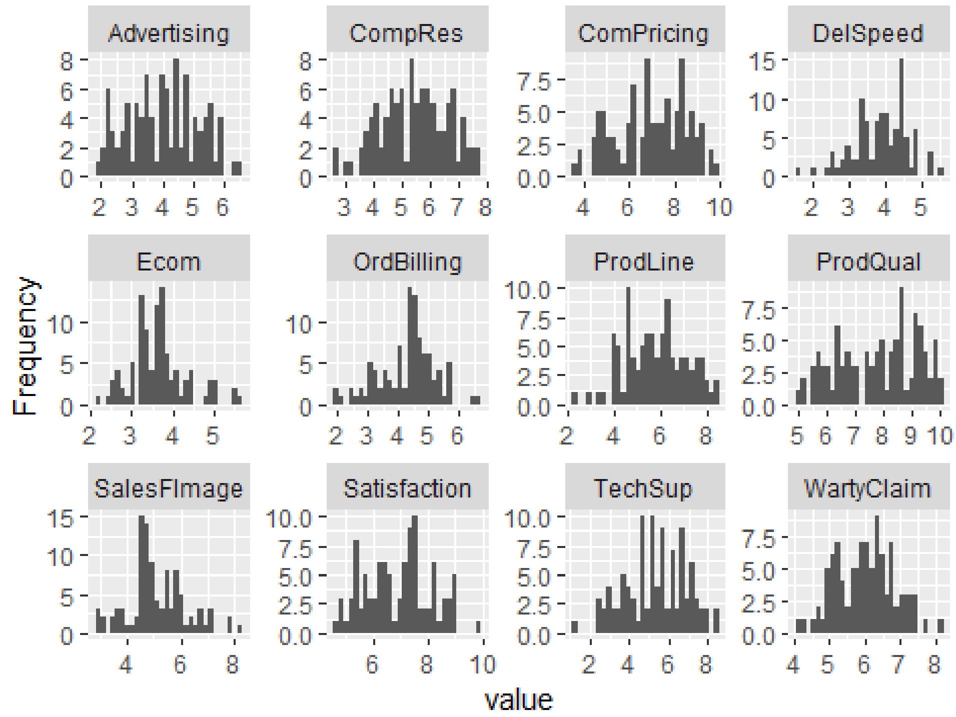

```
#Plot: OrdBilling vs Satisfaction.
plot(dataset$OrdBilling,dataset$Satisfaction,col = "Red")
```




```
#Plot: DelSpeed vs Satisfaction.
plot(dataset$DelSpeed,dataset$Satisfaction,col = "Red")
```




```
###Performing Univariate Analysis on the dataset.

plot_histogram(dataset)
```




```
### Performing Simple Linear Regression on Individual Variables

# Simple Linear Regression Model on Satisfaction and ProdQual
slr_prodqual <- lm(Satisfaction~ProdQual, data = dataset1)
summary(slr_prodqual)

##
## Call:
```

```
## lm(formula = Satisfaction ~ ProdQual, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.88746 -0.72711 -0.01577  0.85641  2.25220
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.67593    0.59765   6.151 1.68e-08 ***
## ProdQual     0.41512    0.07534   5.510 2.90e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.047 on 98 degrees of freedom
## Multiple R-squared:  0.2365, Adjusted R-squared:  0.2287
## F-statistic: 30.36 on 1 and 98 DF,  p-value: 2.901e-07
```

```
# Simple Linear Regression Model on Satisfaction and ECOM
slr_ecom <- lm(Satisfaction~Ecom, data = dataset1)
summary(slr_ecom)
```

```
##
## Call:
## lm(formula = Satisfaction ~ Ecom, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.37200 -0.78971  0.04959  0.68085  2.34580
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.1516     0.6161   8.361 4.28e-13 ***
## Ecom          0.4811     0.1649   2.918  0.00437 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.149 on 98 degrees of freedom
## Multiple R-squared:  0.07994,    Adjusted R-squared:  0.07056
## F-statistic: 8.515 on 1 and 98 DF,  p-value: 0.004368
```

```
# Simple Linear Regression Model on Satisfaction and TechSup
slr_Techsup <- lm(Satisfaction~TechSup, data = dataset1)
summary(slr_Techsup)
```

```
##
## Call:
## lm(formula = Satisfaction ~ TechSup, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.26136 -0.93297  0.04302  0.82501  2.85617
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.44757    0.43592  14.791   <2e-16 ***
## TechSup      0.08768    0.07817   1.122    0.265
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.19 on 98 degrees of freedom
## Multiple R-squared:  0.01268,    Adjusted R-squared:  0.002603
## F-statistic: 1.258 on 1 and 98 DF,  p-value: 0.2647
```

```
# Simple Linear Regression Model on Satisfaction and CompRes
slr_compres <- lm(Satisfaction~CompRes, data = dataset1)
summary(slr_compres)
```

```
##
## Call:
## lm(formula = Satisfaction ~ CompRes, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.40450 -0.66164  0.04499  0.63037  2.70949
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.68005    0.44285   8.310 5.51e-13 ***
## CompRes      0.59499    0.07946   7.488 3.09e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9554 on 98 degrees of freedom
## Multiple R-squared:  0.3639, Adjusted R-squared:  0.3574
## F-statistic: 56.07 on 1 and 98 DF,  p-value: 3.085e-11
```

```
# Simple Linear Regression Model on Satisfaction and Advertising
slr_advertising <- lm(Satisfaction~Advertising, data = dataset1)
summary(slr_advertising)
```

```
##
## Call:
## lm(formula = Satisfaction ~ Advertising, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.34033 -0.92755  0.05577  0.79773  2.53412
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.6259     0.4237  13.279  < 2e-16 ***
## Advertising   0.3222     0.1018   3.167  0.00206 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.141 on 98 degrees of freedom
## Multiple R-squared:  0.09282,    Adjusted R-squared:  0.08357
## F-statistic: 10.03 on 1 and 98 DF,  p-value: 0.002056
```

```
# Simple Linear Regression Model on Satisfaction and Prodline
slr_prodline <- lm(Satisfaction~ProdLine, data = dataset1)
summary(slr_prodline)
```

```
##
## Call:
## lm(formula = Satisfaction ~ ProdLine, data = dataset1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -2.3634 -0.7795  0.1097  0.7604  1.7373
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.02203    0.45471   8.845 3.87e-14 ***
## ProdLine     0.49887    0.07641   6.529 2.95e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1 on 98 degrees of freedom
## Multiple R-squared:  0.3031, Adjusted R-squared:  0.296
## F-statistic: 42.62 on 1 and 98 DF,  p-value: 2.953e-09
```

```
# Simple Linear Regression Model on Satisfaction and SalesFImage
slr_salesfimage <- lm(Satisfaction~SalesFImage, data = dataset1)
summary(slr_salesfimage)
```

```
##
## Call:
## lm(formula = Satisfaction ~ SalesFImage, data = dataset1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2164 -0.5884  0.1838  0.6922  2.0728
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.06983    0.50874   8.000 2.54e-12 ***
## SalesFImage  0.55596    0.09722   5.719 1.16e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.037 on 98 degrees of freedom
## Multiple R-squared:  0.2502, Adjusted R-squared:  0.2426
## F-statistic:  32.7 on 1 and 98 DF,  p-value: 1.164e-07
```

```
# Simple Linear Regression Model on Satisfaction and Compricing
slr_compricing <- lm(Satisfaction~ComPricing, data = dataset1)
summary(slr_compricing)
```

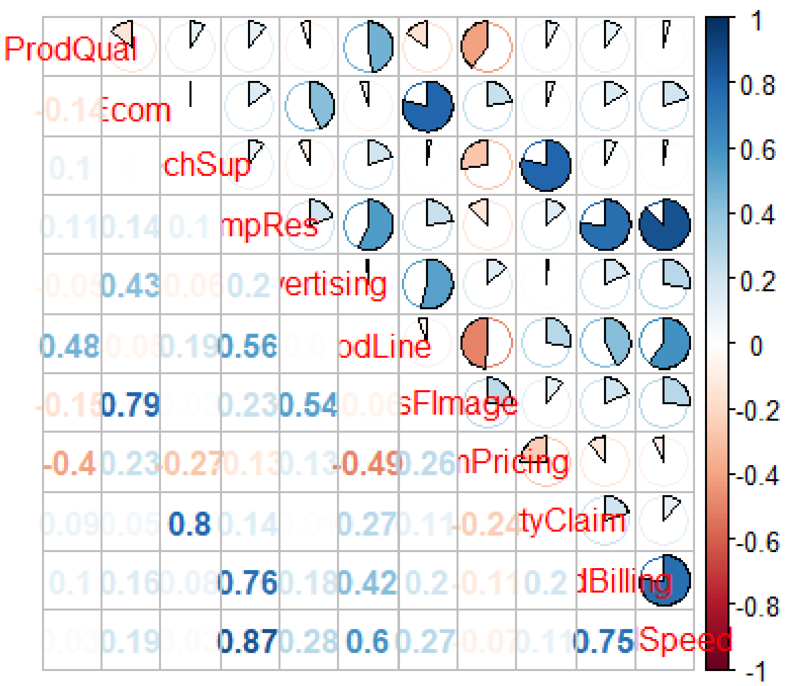
```
##
## Call:
## lm(formula = Satisfaction ~ ComPricing, data = dataset1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9728 -0.9915 -0.1156  0.9111  2.5845
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.03856    0.54427  14.769   <2e-16 ***
## ComPricing  -0.16068    0.07621  -2.108   0.0376 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.172 on 98 degrees of freedom
## Multiple R-squared:  0.04339,    Adjusted R-squared:  0.03363
## F-statistic: 4.445 on 1 and 98 DF,  p-value: 0.03756
```

```r
# Simple Linear Regression Model on Satisfaction and WartyClaims
slr_wartyclaim <- lm(Satisfaction~WartyClaim, data = dataset1)
summary(slr_wartyclaim)

##
## Call:
## lm(formula = Satisfaction ~ WartyClaim, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36504 -0.90202  0.03019  0.90763  2.88985
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.3581     0.8813   6.079 2.32e-08 ***
## WartyClaim    0.2581     0.1445   1.786   0.0772 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.179 on 98 degrees of freedom
## Multiple R-squared:  0.03152,    Adjusted R-squared:  0.02164
## F-statistic:  3.19 on 1 and 98 DF,  p-value: 0.0772

# Simple Linear Regression Model on Satisfaction and OrdBilling
slr_ordbilling <- lm(Satisfaction~OrdBilling, data = dataset1)
summary(slr_ordbilling)

##
## Call:
## lm(formula = Satisfaction ~ OrdBilling, data = dataset1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4005 -0.7071 -0.0344  0.7340  2.9673
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.0541     0.4840   8.377 3.96e-13 ***
## OrdBilling    0.6695     0.1106   6.054 2.60e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.022 on 98 degrees of freedom
## Multiple R-squared:  0.2722, Adjusted R-squared:  0.2648
## F-statistic: 36.65 on 1 and 98 DF,  p-value: 2.602e-08

# Simple Linear Regression Model on Satisfaction and DelSpeed
slr_delspeed <- lm(Satisfaction~DelSpeed, data = dataset1)
summary(slr_delspeed)

##
## Call:
## lm(formula = Satisfaction ~ DelSpeed, data = dataset1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.22475 -0.54846  0.08796  0.54462  2.59432
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

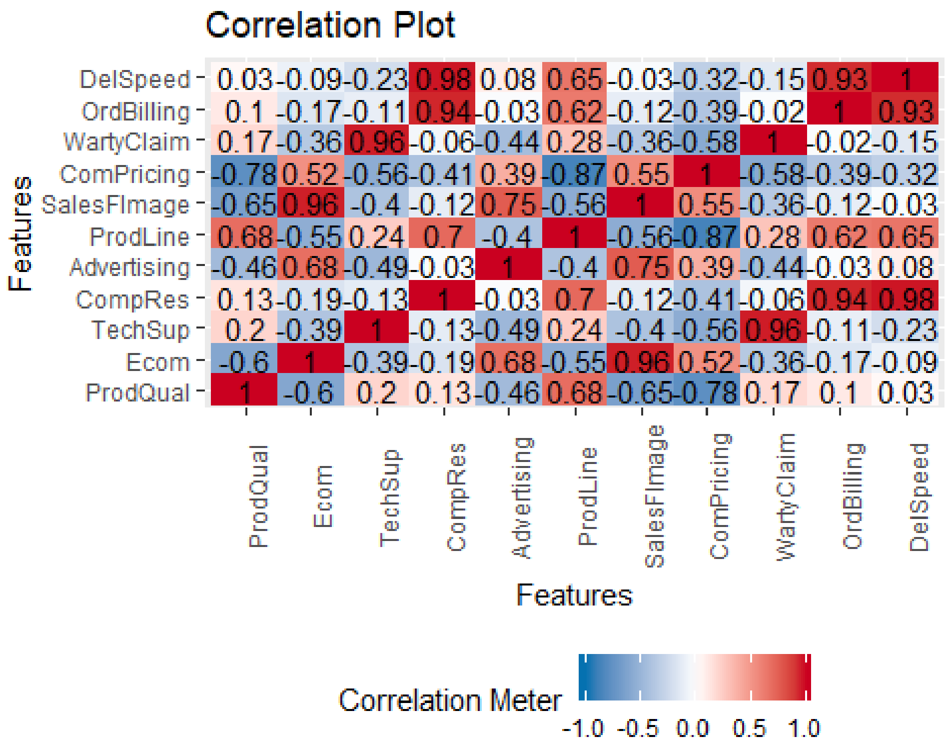
```
## (Intercept)   3.2791     0.5294   6.194 1.38e-08 ***
## DelSpeed      0.9364     0.1339   6.994 3.30e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9783 on 98 degrees of freedom
## Multiple R-squared:  0.333,  Adjusted R-squared:  0.3262
## F-statistic: 48.92 on 1 and 98 DF,  p-value: 3.3e-10

#Identifying the Correlation between the Independent variables.
# Removing the Dependent Variable to check the multicolinearity
# between the Independent Variables


dataset<- dataset[,-12]
str(dataset)

## 'data.frame':    100 obs. of  11 variables:
##  $ ProdQual   : num  8.5 8.2 9.2 6.4 9 6.5 6.9 6.2 5.8 6.4 ...
##  $ Ecom       : num  3.9 2.7 3.4 3.3 3.4 2.8 3.7 3.3 3.6 4.5 ...
##  $ TechSup    : num  2.5 5.1 5.6 7 5.2 3.1 5 3.9 5.1 5.1 ...
##  $ CompRes    : num  5.9 7.2 5.6 3.7 4.6 4.1 2.6 4.8 6.7 6.1 ...
##  $ Advertising: num  4.8 3.4 5.4 4.7 2.2 4 2.1 4.6 3.7 4.7 ...
##  $ ProdLine   : num  4.9 7.9 7.4 4.7 6 4.3 2.3 3.6 5.9 5.7 ...
##  $ SalesFImage: num  6 3.1 5.8 4.5 4.5 3.7 5.4 5.1 5.8 5.7 ...
##  $ ComPricing : num  6.8 5.3 4.5 8.8 6.8 8.5 8.9 6.9 9.3 8.4 ...
##  $ WartyClaim : num  4.7 5.5 6.2 7 6.1 5.1 4.8 5.4 5.9 5.4 ...
##  $ OrdBilling : num  5 3.9 5.4 4.3 4.5 3.6 2.1 4.3 4.4 4.1 ...
##  $ DelSpeed   : num  3.7 4.9 4.5 3 3.5 3.3 2 3.7 4.6 4.4 ...

#Creating the correlation Matrix
mat <- cor(dataset)
mat

##                ProdQual          Ecom       TechSup    CompRes Advertisin
g
## ProdQual     1.00000000 -0.1371632174  0.0956004542  0.1063700 -0.0534731
3
## Ecom        -0.13716322  1.0000000000  0.0008667887  0.1401793  0.4298907
1
## TechSup      0.09560045  0.0008667887  1.0000000000  0.0966566 -0.0628700
7
## CompRes      0.10637000  0.1401792611  0.0966565978  1.0000000  0.1969168
5
## Advertising -0.05347313  0.4298907110 -0.0628700668  0.1969168  1.0000000
0
## ProdLine     0.47749341 -0.0526878383  0.1926254565  0.5614170 -0.0115508
2
## SalesFImage -0.15181287  0.7915437115  0.0169905395  0.2297518  0.5422036
6
## ComPricing  -0.40128188  0.2294624014 -0.2707866821 -0.1279543  0.1342168
9
## WartyClaim   0.08831231  0.0518981915  0.7971679258  0.1404083  0.0107920
7
## OrdBilling   0.10430307  0.1561473316  0.0801018246  0.7568686  0.1842355
9
## DelSpeed     0.02771800  0.1916360683  0.0254406935  0.8650917  0.2758630
8
##                ProdLine SalesFImage  ComPricing  WartyClaim  OrdBilling
```

```
## ProdQual     0.47749341 -0.15181287 -0.40128188  0.08831231  0.10430307
## Ecom        -0.05268784  0.79154371  0.22946240  0.05189819  0.15614733
## TechSup      0.19262546  0.01699054 -0.27078668  0.79716793  0.08010182
## CompRes      0.56141695  0.22975176 -0.12795425  0.14040830  0.75686859
## Advertising -0.01155082  0.54220366  0.13421689  0.01079207  0.18423559
## ProdLine     1.00000000 -0.06131553 -0.49494840  0.27307753  0.42440825
## SalesFImage -0.06131553  1.00000000  0.26459655  0.10745534  0.19512741
## ComPricing  -0.49494840  0.26459655  1.00000000 -0.24498605 -0.11456703
## WartyClaim   0.27307753  0.10745534 -0.24498605  1.00000000  0.19706512
## OrdBilling   0.42440825  0.19512741 -0.11456703  0.19706512  1.00000000
## DelSpeed     0.60185021  0.27155126 -0.07287173  0.10939460  0.75100307
##                DelSpeed
## ProdQual     0.02771800
## Ecom         0.19163607
## TechSup      0.02544069
## CompRes      0.86509170
## Advertising  0.27586308
## ProdLine     0.60185021
## SalesFImage  0.27155126
## ComPricing  -0.07287173
## WartyClaim   0.10939460
## OrdBilling   0.75100307
## DelSpeed     1.00000000
```

```r
###Plotting the Correlation Matrix

# Plot 1
corrplot.mixed(mat, lower = "number", upper = "pie")
```

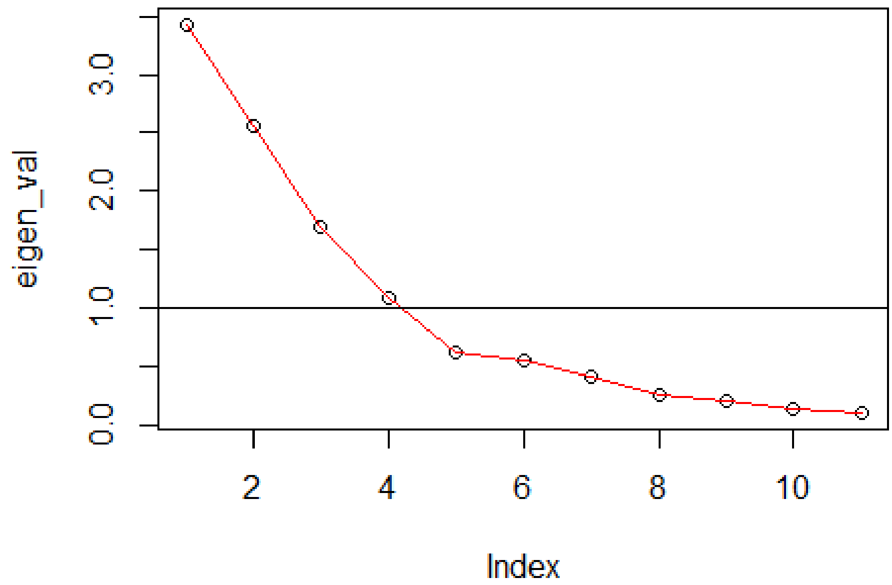



```r
# Plot 2
plot_correlation(mat, title = "Correlation Plot")
```




```
#Performing Bartlett test and KMO Test to check the validity of the Correlat
ion
#Bartlett Test: If p-Value < 0.05, correlation is valid

cortest.bartlett(mat,n = 100)

## $chisq
## [1] 619.2726
##
## $p.value
## [1] 1.79337e-96
##
## $df
## [1] 55

#$p.value = 1.79337e-96

#KMO Test: If the MSA > 0.5, correlation is valid
KMO(mat)

## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = mat)
## Overall MSA =  0.65
## MSA for each item =
##     ProdQual         Ecom      TechSup      CompRes Advertising     ProdLine
##         0.51         0.63         0.52         0.79        0.78         0.62
## SalesFImage  ComPricing  WartyClaim  OrdBilling     DelSpeed
##         0.62         0.75         0.51         0.76        0.67

#Overall MSA =  0.65

### Performing Eigen Test

a = eigen(mat)
eigen_val <- a$values
```

```r
#Plotting Eigen Values
plot(eigen_val)
lines(eigen_val, col = "red")
abline(h = 1)
```




```r
### Performing PCA.

#Using the Kizer Rule, We take the number of factors for which the eigen val
ues are greater then 1.
#In this case, we have 4 values which are greater the 1, hence we take 4 fac
tors for PCA
#With Rotate = "None"

pca_no_rtt <- principal(dataset, nfactors = 4, rotate = "none")
pca_no_rtt

## Principal Components Analysis
## Call: principal(r = dataset, nfactors = 4, rotate = "none")
## Standardized loadings (pattern matrix) based upon correlation matrix
##               PC1   PC2   PC3   PC4   h2    u2 com
## ProdQual     0.25 -0.50 -0.08  0.67 0.77 0.232 2.2
## Ecom         0.31  0.71  0.31  0.28 0.78 0.223 2.1
## TechSup      0.29 -0.37  0.79 -0.20 0.89 0.107 1.9
## CompRes      0.87  0.03 -0.27 -0.22 0.88 0.119 1.3
## Advertising  0.34  0.58  0.11  0.33 0.58 0.424 2.4
## ProdLine     0.72 -0.45 -0.15  0.21 0.79 0.213 2.0
## SalesFImage  0.38  0.75  0.31  0.23 0.86 0.141 2.1
## ComPricing  -0.28  0.66 -0.07 -0.35 0.64 0.359 1.9
## WartyClaim   0.39 -0.31  0.78 -0.19 0.89 0.108 2.0
## OrdBilling   0.81  0.04 -0.22 -0.25 0.77 0.234 1.3
## DelSpeed     0.88  0.12 -0.30 -0.21 0.91 0.086 1.4
##
##                        PC1  PC2  PC3  PC4
## SS loadings           3.43 2.55 1.69 1.09
## Proportion Var        0.31 0.23 0.15 0.10
## Cumulative Var        0.31 0.54 0.70 0.80
```

```
## Proportion Explained  0.39 0.29 0.19 0.12
## Cumulative Proportion 0.39 0.68 0.88 1.00
##
## Mean item complexity =  1.9
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.06
##  with the empirical chi square  39.02  with prob <  0.0018
##
## Fit based upon off diagonal values = 0.97

#Creating the Score Matrix.
pca_score_no_rtt <- pca_no_rtt$scores
head(pca_score_no_rtt)

##              PC1        PC2        PC3         PC4
## [1,] -0.04275697  0.9613752 -1.4500022  1.11500948
## [2,]  0.59174746 -1.5077872 -1.5651385 -0.40766575
## [3,]  1.18087946 -0.4531778 -0.1276279  1.25073815
## [4,] -0.84004527  0.1067572  1.3980732 -1.13803413
## [5,] -0.41255378 -0.8896788 -0.1793184  0.06636444
## [6,] -1.56332622  0.1927404 -1.1727562 -0.71266053

#Joining the Score Matrix with the Original dataset to get the Satisfaction
variable
pca_reg_no_rtt <- cbind(dataset1[,12],pca_score_no_rtt)

#Renaming the Factored dataset.
colnames(pca_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3
", "Factor4")
head(pca_reg_no_rtt)

##      Satisfaction     Factor1    Factor2    Factor3     Factor4
## [1,]          8.2 -0.04275697  0.9613752 -1.4500022  1.11500948
## [2,]          5.7  0.59174746 -1.5077872 -1.5651385 -0.40766575
## [3,]          8.9  1.18087946 -0.4531778 -0.1276279  1.25073815
## [4,]          4.8 -0.84004527  0.1067572  1.3980732 -1.13803413
## [5,]          7.1 -0.41255378 -0.8896788 -0.1793184  0.06636444
## [6,]          4.7 -1.56332622  0.1927404 -1.1727562 -0.71266053

pca_reg_no_rtt<- as.data.frame(pca_reg_no_rtt)

set.seed(42)

#Dividing the dataset into Test and Train.
index_pca_no_rtt <- sample.split(pca_reg_no_rtt$Satisfaction,SplitRatio = .7
0)
train_pca_no_rtt <- subset(pca_reg_no_rtt,index_pca_no_rtt ==TRUE)
test_pca_no_rtt <- subset(pca_reg_no_rtt, index_pca_no_rtt == FALSE)

#Building the Regression Model on the Train Data.
model_pca_no_rtt<- lm(Satisfaction~.,data = train_pca_no_rtt)
summary(model_pca_no_rtt)

##
## Call:
## lm(formula = Satisfaction ~ ., data = train_pca_no_rtt)
##
## Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -1.5949 -0.4991  0.1406  0.4696  1.5186
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.91854    0.08417  82.201  < 2e-16 ***
## Factor1      0.86814    0.08109  10.706 4.55e-16 ***
## Factor2      0.07492    0.08566   0.875    0.385
## Factor3     -0.02622    0.08768  -0.299    0.766
## Factor4      0.39469    0.08158   4.838 8.22e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7069 on 66 degrees of freedom
## Multiple R-squared:  0.684,  Adjusted R-squared:  0.6648
## F-statistic: 35.71 on 4 and 66 DF,  p-value: 7.314e-16
```

```
#Validating the Model on Test Data.
pred_pca_no_rtt <- predict(model_pca_no_rtt,newdata = test_pca_no_rtt)
summary(pred_pca_no_rtt)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.679   6.610   7.078   6.933   7.432   8.735
```

```
SST_no_rtt <- sum((test_pca_no_rtt$Satisfaction - mean(test_pca_no_rtt$Satis
faction))^2)
SSR_no_rtt <- sum((pred_pca_no_rtt - mean(test_pca_no_rtt$Satisfaction))^2)
SSE_no_rtt <- sum((test_pca_no_rtt$Satisfaction - pred_pca_no_rtt)^2)

calculated_Rsq_no_rtt <- 1-(SSE_no_rtt/SST_no_rtt)
calculated_Rsq_no_rtt
```

```
## [1] 0.591059
```

```
#Using Rotate = "Varimax".

pca_wt_rtt <- principal(dataset, nfactors = 4, rotate = "varimax")
pca_wt_rtt
```

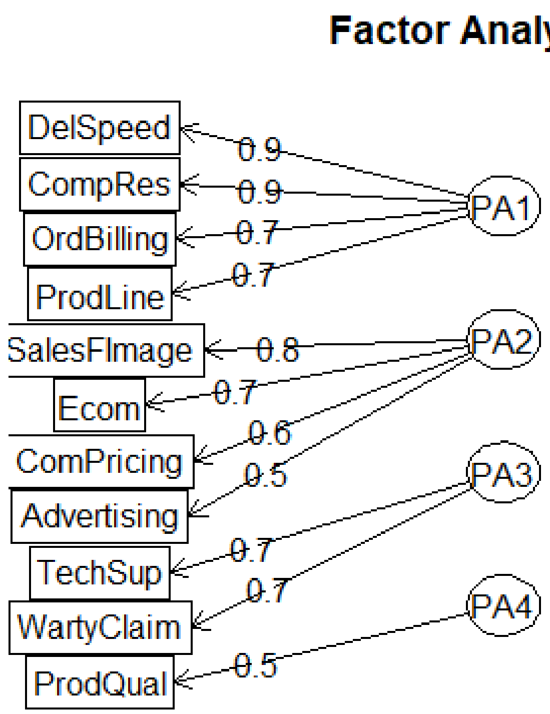
```
## Principal Components Analysis
## Call: principal(r = dataset, nfactors = 4, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##               RC1   RC2   RC3   RC4   h2    u2 com
## ProdQual     0.00 -0.01 -0.03  0.88 0.77 0.232 1.0
## Ecom         0.06  0.87  0.05 -0.12 0.78 0.223 1.1
## TechSup      0.02 -0.02  0.94  0.10 0.89 0.107 1.0
## CompRes      0.93  0.12  0.05  0.09 0.88 0.119 1.1
## Advertising  0.14  0.74 -0.08  0.01 0.58 0.424 1.1
## ProdLine     0.59 -0.06  0.15  0.64 0.79 0.213 2.1
## SalesFImage  0.13  0.90  0.08 -0.16 0.86 0.141 1.1
## ComPricing  -0.09  0.23 -0.25 -0.72 0.64 0.359 1.5
## WartyClaim   0.11  0.05  0.93  0.10 0.89 0.108 1.1
## OrdBilling   0.86  0.11  0.08  0.04 0.77 0.234 1.1
## DelSpeed     0.94  0.18  0.00  0.05 0.91 0.086 1.1
##
##                        RC1  RC2  RC3  RC4
## SS loadings           2.89 2.23 1.86 1.77
## Proportion Var        0.26 0.20 0.17 0.16
## Cumulative Var        0.26 0.47 0.63 0.80
## Proportion Explained  0.33 0.26 0.21 0.20
## Cumulative Proportion 0.33 0.59 0.80 1.00
```

```
##
## Mean item complexity =  1.2
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.06
##  with the empirical chi square  39.02  with prob <  0.0018
##
## Fit based upon off diagonal values = 0.97

#Creating the Score Matrix.
pca_score_wt_rtt <- pca_wt_rtt$scores
head(pca_score_wt_rtt)

##             RC1        RC2          RC3        RC4
## [1,]  0.1274910  0.7698686 -1.878446273  0.3664848
## [2,]  1.2216666 -1.6458617 -0.614030010  0.8130648
## [3,]  0.6158214  0.5800037  0.003689252  1.5699769
## [4,] -0.8446267 -0.2719218  1.267493254 -1.2541645
## [5,] -0.3197943 -0.8340650 -0.008096627  0.4475377
## [6,] -0.6470292 -1.0672683 -1.303198892 -1.0527792

#Joining the Score Matrix with Original Dataset to get the Dependent Dataset
pca_reg_wt_rtt <- cbind(dataset1[,12],pca_score_wt_rtt)

#Renaming the Factored Dataset
colnames(pca_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3
", "Factor4")
head(pca_reg_wt_rtt)

##      Satisfaction    Factor1    Factor2      Factor3    Factor4
## [1,]          8.2  0.1274910  0.7698686 -1.878446273  0.3664848
## [2,]          5.7  1.2216666 -1.6458617 -0.614030010  0.8130648
## [3,]          8.9  0.6158214  0.5800037  0.003689252  1.5699769
## [4,]          4.8 -0.8446267 -0.2719218  1.267493254 -1.2541645
## [5,]          7.1 -0.3197943 -0.8340650 -0.008096627  0.4475377
## [6,]          4.7 -0.6470292 -1.0672683 -1.303198892 -1.0527792

pca_reg_wt_rtt<- as.data.frame(pca_reg_wt_rtt)

set.seed(42)
#Dividing the dataset into Test and Train.
index_pca_wt_rtt <- sample.split(pca_reg_wt_rtt$Satisfaction,SplitRatio = .7
0)
train_pca_wt_rtt <- subset(pca_reg_wt_rtt,index_pca_wt_rtt ==TRUE)
test_pca_wt_rtt <- subset(pca_reg_wt_rtt, index_pca_wt_rtt == FALSE)

#Building the Model on Train Dataset
model_pca_wt_rtt<- lm(Satisfaction~.,data = train_pca_wt_rtt)
summary(model_pca_wt_rtt)

##
## Call:
## lm(formula = Satisfaction ~ ., data = train_pca_wt_rtt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5949 -0.4991  0.1406  0.4696  1.5186
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  6.91854    0.08417  82.201  < 2e-16 ***
## Factor1      0.61637    0.08110   7.600 1.37e-10 ***
## Factor2      0.51241    0.08114   6.315 2.62e-08 ***
## Factor3      0.07954    0.08878   0.896    0.374
## Factor4      0.51666    0.08492   6.084 6.63e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7069 on 66 degrees of freedom
## Multiple R-squared:  0.684,  Adjusted R-squared:  0.6648
## F-statistic: 35.71 on 4 and 66 DF,  p-value: 7.314e-16

#Validating the Model on Test Data.
pred_pca_wt_rtt <- predict(model_pca_wt_rtt,newdata = test_pca_wt_rtt)

summary(pred_pca_wt_rtt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.679   6.610   7.078   6.933   7.432   8.735

SST_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - mean(test_pca_wt_rtt$Satis
faction))^2)
SSR_wt_rtt <- sum((pred_pca_wt_rtt - mean(test_pca_wt_rtt$Satisfaction))^2)
SSE_wt_rtt <- sum((test_pca_wt_rtt$Satisfaction - pred_pca_wt_rtt)^2)

calculated_Rsq_wt_rtt <- 1-(SSE_wt_rtt/SST_wt_rtt)
calculated_Rsq_wt_rtt

## [1] 0.591059

### Performing Factor Analysis.

#Using the Kizer Rule, We take the number of factors for which the eigen val
ues are greater then 1.
#In this case, we have 4 values which are greater the 1, hence we take 4 fac
tors for PCA.
#With Rotate = None

fa_no_rtt <- fa(dataset,nfactors = 4, rotate = "none", fm="pa")
fa_no_rtt

## Factor Analysis using method =  pa
## Call: fa(r = dataset, nfactors = 4, rotate = "none", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PA1   PA2   PA3   PA4   h2    u2 com
## ProdQual      0.20 -0.41 -0.06  0.46 0.42 0.576 2.4
## Ecom          0.29  0.66  0.27  0.22 0.64 0.362 2.0
## TechSup       0.28 -0.38  0.74 -0.17 0.79 0.205 1.9
## CompRes       0.86  0.01 -0.26 -0.18 0.84 0.157 1.3
## Advertising   0.29  0.46  0.08  0.13 0.31 0.686 1.9
## ProdLine      0.69 -0.45 -0.14  0.31 0.80 0.200 2.3
## SalesFImage   0.39  0.80  0.35  0.25 0.98 0.021 2.1
## ComPricing   -0.23  0.55 -0.04 -0.29 0.44 0.557 1.9
## WartyClaim    0.38 -0.32  0.74 -0.15 0.81 0.186 2.0
## OrdBilling    0.75  0.02 -0.18 -0.18 0.62 0.378 1.2
## DelSpeed      0.90  0.10 -0.30 -0.20 0.94 0.058 1.4
##
##                        PA1  PA2  PA3  PA4
## SS loadings           3.21 2.22 1.50 0.68
## Proportion Var        0.29 0.20 0.14 0.06
```

```
## Cumulative Var         0.29 0.49 0.63 0.69
## Proportion Explained  0.42 0.29 0.20 0.09
## Cumulative Proportion 0.42 0.71 0.91 1.00
##
## Mean item complexity =  1.9
## Test of the hypothesis that 4 factors are sufficient.
##
## The degrees of freedom for the null model are  55  and the objective func
tion was  6.55 with Chi Square of  619.27
## The degrees of freedom for the model are 17  and the objective function w
as  0.33
##
## The root mean square of the residuals (RMSR) is  0.02
## The df corrected root mean square of the residuals is  0.03
##
## The harmonic number of observations is  100 with the empirical chi square
  3.19  with prob <  1
## The total number of observations was  100  with Likelihood Chi Square =
30.27  with prob <  0.024
##
## Tucker Lewis Index of factoring reliability =  0.921
## RMSEA index =  0.096  and the 90 % confidence intervals are  0.032 0.139
## BIC =  -48.01
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                  PA1  PA2  PA3  PA4
## Correlation of (regression) scores with factors   0.98 0.97 0.95 0.88
## Multiple R square of scores with factors          0.96 0.95 0.91 0.78
## Minimum correlation of possible factor scores     0.92 0.90 0.82 0.56
```

```r
#Plotting the Factor Analysis
fa.diagram(fa_no_rtt)
```




```r
#Creating the Score Matrix
fa_score_no_rtt <- fa_no_rtt$scores
head(fa_score_no_rtt)
```

```
##              PA1         PA2         PA3        PA4
## [1,] -0.2367381  1.23998570 -1.12891346  0.9783118
## [2,]  0.7704292 -1.70153639 -1.84032527 -0.7780463
## [3,]  1.0117487 -0.09673524  0.06374278  1.2986265
## [4,] -1.0930182 -0.49877402  1.38672297 -0.6291745
## [5,] -0.4156728 -0.57647133 -0.01325216  0.3718166
## [6,] -1.5462133 -0.08944095 -1.17590551 -0.7689481
```

```
#Joining the Score Matrix with the Original dataset to get the Satisfaction
variable
fa_reg_no_rtt <- cbind(dataset1[,12],fa_score_no_rtt)

#Renaming the Factored dataset
colnames(fa_reg_no_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3"
, "Factor4")
head(fa_reg_no_rtt)
```

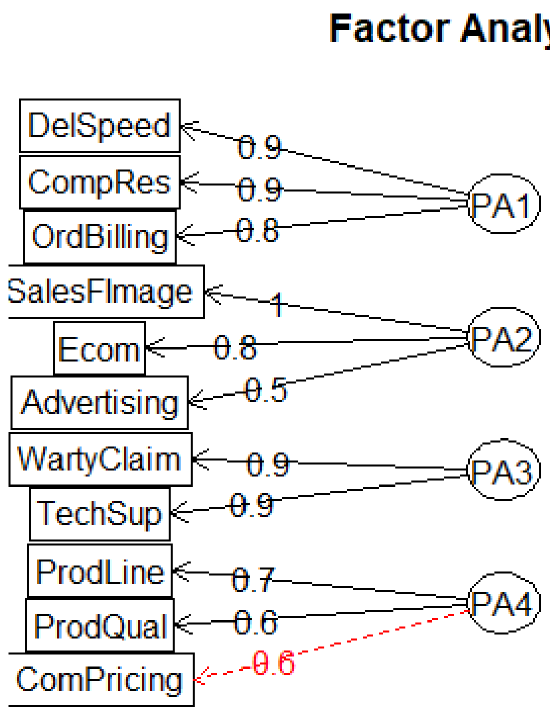
```
##      Satisfaction    Factor1     Factor2     Factor3    Factor4
## [1,]          8.2 -0.2367381  1.23998570 -1.12891346  0.9783118
## [2,]          5.7  0.7704292 -1.70153639 -1.84032527 -0.7780463
## [3,]          8.9  1.0117487 -0.09673524  0.06374278  1.2986265
## [4,]          4.8 -1.0930182 -0.49877402  1.38672297 -0.6291745
## [5,]          7.1 -0.4156728 -0.57647133 -0.01325216  0.3718166
## [6,]          4.7 -1.5462133 -0.08944095 -1.17590551 -0.7689481
```

```
fa_reg_no_rtt<- as.data.frame(fa_reg_no_rtt)

set.seed(42)

#Dividing the Dataset to Test and Train Data
index <- sample.split(fa_reg_no_rtt$Satisfaction,SplitRatio = .70)
train_fa_no_rtt <- subset(fa_reg_no_rtt,index ==TRUE)
test_fa_no_rtt <- subset(fa_reg_no_rtt, index == FALSE)

#Building the Regression Model on Train Data
model_fa_no_rtt<- lm(Satisfaction~.,data = train_fa_no_rtt)
summary(model_fa_no_rtt)
```

```
##
## Call:
## lm(formula = Satisfaction ~ ., data = train_fa_no_rtt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7245 -0.4309  0.1096  0.4250  1.1092
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.90291    0.07570  91.182  < 2e-16 ***
## Factor1      0.90488    0.07467  12.119  < 2e-16 ***
## Factor2      0.15522    0.08073   1.923   0.0588 .
## Factor3      0.03930    0.08277   0.475   0.6365
## Factor4      0.51205    0.08462   6.051 7.55e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6378 on 66 degrees of freedom
## Multiple R-squared:  0.7427, Adjusted R-squared:  0.7271
## F-statistic: 47.63 on 4 and 66 DF,  p-value: < 2.2e-16
```

```
#Validating the Model on Test Data
pred_fa_no_rtt <- predict(model_fa_no_rtt,newdata = test_fa_no_rtt)
summary(pred_fa_no_rtt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.689   6.488   6.879   6.879   7.372   8.661

SST_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - mean(test_pca_no_rtt$Sat
isfaction))^2)
SSR_fa_no_rtt <- sum((pred_fa_no_rtt - mean(test_fa_no_rtt$Satisfaction))^2)
SSE_fa_no_rtt <- sum((test_fa_no_rtt$Satisfaction - pred_fa_no_rtt)^2)

calculated_Rsq_fa_no_rtt <- 1-(SSE_fa_no_rtt/SST_fa_no_rtt)
calculated_Rsq_fa_no_rtt

## [1] 0.5601423

#Using the Keiser Law.
#In this case, we have 4 values which are greater the 1, hence we take 4 fac
tors for FA.
#With rotate = Varimax
fa_wt_rtt <- fa(dataset,nfactors =4, rotate = "varimax", fm="pa")
fa_wt_rtt

## Factor Analysis using method =  pa
## Call: fa(r = dataset, nfactors = 4, rotate = "varimax", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##               PA1   PA2   PA3   PA4   h2    u2 com
## ProdQual     0.02 -0.07  0.02  0.65 0.42 0.576 1.0
## Ecom         0.07  0.79  0.03 -0.11 0.64 0.362 1.1
## TechSup      0.02 -0.03  0.88  0.12 0.79 0.205 1.0
## CompRes      0.90  0.13  0.05  0.13 0.84 0.157 1.1
## Advertising  0.17  0.53 -0.04 -0.06 0.31 0.686 1.2
## ProdLine     0.53 -0.04  0.13  0.71 0.80 0.200 1.9
## SalesFImage  0.12  0.97  0.06 -0.13 0.98 0.021 1.1
## ComPricing  -0.08  0.21 -0.21 -0.59 0.44 0.557 1.6
## WartyClaim   0.10  0.06  0.89  0.13 0.81 0.186 1.1
## OrdBilling   0.77  0.13  0.09  0.09 0.62 0.378 1.1
## DelSpeed     0.95  0.19  0.00  0.09 0.94 0.058 1.1
##
##                        PA1  PA2  PA3  PA4
## SS loadings           2.63 1.97 1.64 1.37
## Proportion Var        0.24 0.18 0.15 0.12
## Cumulative Var        0.24 0.42 0.57 0.69
## Proportion Explained  0.35 0.26 0.22 0.18
## Cumulative Proportion 0.35 0.60 0.82 1.00
##
## Mean item complexity =  1.2
## Test of the hypothesis that 4 factors are sufficient.
##
## The degrees of freedom for the null model are  55  and the objective func
tion was  6.55 with Chi Square of  619.27
## The degrees of freedom for the model are 17  and the objective function w
as  0.33
##
## The root mean square of the residuals (RMSR) is  0.02
## The df corrected root mean square of the residuals is  0.03
##
## The harmonic number of observations is  100 with the empirical chi square
  3.19  with prob <  1
```

```
## The total number of observations was  100  with Likelihood Chi Square = 
30.27  with prob <  0.024 
## 
## Tucker Lewis Index of factoring reliability =  0.921
## RMSEA index =  0.096  and the 90 % confidence intervals are  0.032 0.139
## BIC =  -48.01
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy             
##                                                    PA1  PA2  PA3  PA4
## Correlation of (regression) scores with factors   0.98 0.99 0.94 0.88
## Multiple R square of scores with factors          0.96 0.97 0.88 0.78
## Minimum correlation of possible factor scores     0.93 0.94 0.77 0.55

fa.diagram(fa_wt_rtt)
```




```
#Creating the Score Matrix
fa_score_wt_rtt <- fa_wt_rtt$scores
head(fa_score_wt_rtt)

##             PA1        PA2          PA3         PA4
## [1,] -0.1338871  0.9175166 -1.719604873  0.09135411
## [2,]  1.6297604 -2.0090053 -0.596361722  0.65808192
## [3,]  0.3637658  0.8361736  0.002979966  1.37548765
## [4,] -1.2225230 -0.5491336  1.245473305 -0.64421384
## [5,] -0.4854209 -0.4276223 -0.026980304  0.47360747
## [6,] -0.5950924 -1.3035333 -1.183019401 -0.95913571

#Joining the Score Matrix with the Original dataset to get the Satisfaction 
variable
fa_reg_wt_rtt <- cbind(dataset1[,12],fa_score_wt_rtt)

#Renaming the Factored Dataset
colnames(fa_reg_wt_rtt) <- c("Satisfaction", "Factor1", "Factor2", "Factor3"
, "Factor4")
head(fa_reg_wt_rtt)
```

```
##      Satisfaction    Factor1    Factor2      Factor3     Factor4
## [1,]          8.2 -0.1338871  0.9175166 -1.719604873  0.09135411
## [2,]          5.7  1.6297604 -2.0090053 -0.596361722  0.65808192
## [3,]          8.9  0.3637658  0.8361736  0.002979966  1.37548765
## [4,]          4.8 -1.2225230 -0.5491336  1.245473305 -0.64421384
## [5,]          7.1 -0.4854209 -0.4276223 -0.026980304  0.47360747
## [6,]          4.7 -0.5950924 -1.3035333 -1.183019401 -0.95913571

fa_reg_wt_rtt<- as.data.frame(fa_reg_wt_rtt)

#Splitting the data into Test and Train.
set.seed(42)
index <- sample.split(fa_reg_wt_rtt$Satisfaction,SplitRatio = .70)
train_fa_wt_rtt <- subset(fa_reg_wt_rtt,index ==TRUE)
test_fa_wt_rtt <- subset(fa_reg_wt_rtt, index == FALSE)

#Building the Model
model_fa_wt_rtt<- lm(Satisfaction~.,data = train_fa_wt_rtt)
summary(model_fa_wt_rtt)

##
## Call:
## lm(formula = Satisfaction ~ ., data = train_fa_wt_rtt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7245 -0.4309  0.1096  0.4250  1.1092
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.90291    0.07570  91.182  < 2e-16 ***
## Factor1      0.56931    0.07447   7.645 1.14e-10 ***
## Factor2      0.65081    0.07723   8.427 4.57e-12 ***
## Factor3      0.08159    0.08479   0.962    0.339
## Factor4      0.59355    0.08605   6.898 2.46e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6378 on 66 degrees of freedom
## Multiple R-squared:  0.7427, Adjusted R-squared:  0.7271
## F-statistic: 47.63 on 4 and 66 DF,  p-value: < 2.2e-16

#Validating the Model on the Test Data.
pred_fa_wt_rtt <- predict(model_fa_wt_rtt,newdata = test_fa_wt_rtt)
summary(pred_fa_wt_rtt)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.689   6.488   6.879   6.879   7.372   8.661

SST_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - mean(test_fa_wt_rtt$Sati
sfaction))^2)
SSR_fa_wt_rtt <- sum((pred_fa_wt_rtt - mean(test_fa_wt_rtt$Satisfaction))^2)
SSE_fa_wt_rtt <- sum((test_fa_wt_rtt$Satisfaction - pred_fa_wt_rtt)^2)

calculated_Rsq_fa_wt_rtt <- 1-(SSE_fa_wt_rtt/SST_fa_wt_rtt)
calculated_Rsq_fa_no_rtt

## [1] 0.5601423
```