Week 14 - Dimensionality Reduction

Wilkister Mbaka

2022-07-29

CarreFour Marketing - Dimensionality Reduction

Definining The Question

Specifying the Question

1. This section of the project entails reducing your dataset to a low dimensional dataset using the t-SNE algorithm or PCA.
2. You will be required to perform your analysis and provide insights gained from your analysis.

Metric of success

- Importing the data
- Cleaning the data
- performing a thorough EDA
- Performing Dimensionality Reduction

Data relevance

The data has been provided by the supermarket itself

Understanding the context

You are a Data analyst at Carrefour Kenya and are currently undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax). Your project has been divided into four parts where you'll explore a recent marketing dataset by performing various unsupervised learning techniques and later providing recommendations based on your insights.

Experimental design

The experimental design will involve the following steps:

- Dealing with missing values.
- Dropping variables of low variance.
- Use of decision trees to tackle missing values, outliers and identifying significant variables.
- Use of random forest to select a smaller subset of input features.
- Using the Pearson correlation matrix to identify and later drop variables with high correlation.
- Performing backward feature elimination.
- Performing factor analysis to group high correlated variables.
- Using Principal Component Analysis (PCA).

Reading The Data

```
# Importing Libraries
library (tidyr)
library(naniar)
library (ggplot2)
library (e1071)
library (corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(NbClust)
library(superml)
```

```
## Loading required package: R6
```

```
#installing packages
library(data.table)
#
#Loading the dataset
df <- fread("http://bit.ly/CarreFourDataset")
```

Checking The Data

```
# Preview the data
head(df)
```

```
##     Invoice ID Branch Customer type Gender           Product line Unit price
## 1: 750-67-8428      A        Member Female      Health and beauty      74.69
## 2: 226-31-3081      C        Normal Female Electronic accessories      15.28
## 3: 631-41-3108      A        Normal   Male     Home and lifestyle      46.33
## 4: 123-19-1176      A        Member   Male      Health and beauty      58.22
## 5: 373-73-7910      A        Normal   Male      Sports and travel      86.31
## 6: 699-14-3026      C        Normal   Male Electronic accessories      85.39
##    Quantity     Tax      Date  Time     Payment   cogs gross margin percentage
## 1:        7 26.1415  1/5/2019 13:08     Ewallet 522.83               4.761905
## 2:        5  3.8200  3/8/2019 10:29        Cash  76.40               4.761905
## 3:        7 16.2155  3/3/2019 13:23 Credit card 324.31               4.761905
## 4:        8 23.2880 1/27/2019 20:33     Ewallet 465.76               4.761905
## 5:        7 30.2085  2/8/2019 10:37     Ewallet 604.17               4.761905
## 6:        7 29.8865 3/25/2019 18:30     Ewallet 597.73               4.761905
##    gross income Rating    Total
## 1:      26.1415    9.1 548.9715
## 2:       3.8200    9.6  80.2200
```

```
## 3:      16.2155    7.4 340.5255
## 4:      23.2880    8.4 489.0480
## 5:      30.2085    5.3 634.3785
## 6:      29.8865    4.1 627.6165
```

```
# Preview the data
tail(df)
```

```
##     Invoice ID Branch Customer type Gender           Product line Unit price
## 1: 652-49-6720      C        Member Female Electronic accessories      60.95
## 2: 233-67-5758      C        Normal   Male      Health and beauty      40.35
## 3: 303-96-2227      B        Normal Female     Home and lifestyle      97.38
## 4: 727-02-1313      A        Member   Male     Food and beverages      31.84
## 5: 347-56-2442      A        Normal   Male     Home and lifestyle      65.82
## 6: 849-09-3807      A        Member Female    Fashion accessories      88.34
##    Quantity     Tax      Date  Time Payment   cogs gross margin percentage
## 1:        1  3.0475 2/18/2019 11:40 Ewallet  60.95                4.761905
## 2:        1  2.0175 1/29/2019 13:46 Ewallet  40.35                4.761905
## 3:       10 48.6900  3/2/2019 17:16 Ewallet 973.80                4.761905
## 4:        1  1.5920  2/9/2019 13:22    Cash  31.84                4.761905
## 5:        1  3.2910 2/22/2019 15:33    Cash  65.82                4.761905
## 6:        7 30.9190 2/18/2019 13:28    Cash 618.38                4.761905
##    gross income Rating     Total
## 1:       3.0475    5.9   63.9975
## 2:       2.0175    6.2   42.3675
## 3:      48.6900    4.4 1022.4900
## 4:       1.5920    7.7   33.4320
## 5:       3.2910    4.1   69.1110
## 6:      30.9190    6.6  649.2990
```

```
# Dimensionanity of the data
dim(df)
```

```
## [1] 1000   16
```

The dataframe has 1000 rows and 16 columns

Tidying The Dataset

```
# check the column names
colnames(df)
```

```
##  [1] "Invoice ID"             "Branch"
##  [3] "Customer type"          "Gender"
##  [5] "Product line"           "Unit price"
##  [7] "Quantity"               "Tax"
##  [9] "Date"                   "Time"
## [11] "Payment"                "cogs"
## [13] "gross margin percentage" "gross income"
## [15] "Rating"                 "Total"
```

```
# standardize column names with standard naming convention ie lowercase and replace spaces with '_'
# replace the spaces with underscores using gsub() function
names(df) <- gsub(" ","_", names(df))

# The column names have a mixture of uppercase and lowercase charachers we should correct that and
#make all the characters lowercase.
names(df) <- tolower(names(df))
# Confirmation
colnames(df)
```

```
##  [1] "invoice_id"             "branch"
##  [3] "customer_type"          "gender"
##  [5] "product_line"           "unit_price"
##  [7] "quantity"               "tax"
##  [9] "date"                   "time"
## [11] "payment"                "cogs"
## [13] "gross_margin_percentage" "gross_income"
## [15] "rating"                 "total"
```

```
# Let us find the datatypes of the data
str(df)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  16 variables:
##  $ invoice_id             : chr  "750-67-8428" "226-31-3081" "631-41-3108" "123-19-1176" ...
##  $ branch                 : chr  "A" "C" "A" "A" ...
##  $ customer_type          : chr  "Member" "Normal" "Normal" "Member" ...
##  $ gender                 : chr  "Female" "Female" "Male" "Male" ...
##  $ product_line           : chr  "Health and beauty" "Electronic accessories" "Home and lifestyle" "
##  $ unit_price             : num  74.7 15.3 46.3 58.2 86.3 ...
##  $ quantity               : int  7 5 7 8 7 7 6 10 2 3 ...
##  $ tax                    : num  26.14 3.82 16.22 23.29 30.21 ...
##  $ date                   : chr  "1/5/2019" "3/8/2019" "3/3/2019" "1/27/2019" ...
##  $ time                   : chr  "13:08" "10:29" "13:23" "20:33" ...
##  $ payment                : chr  "Ewallet" "Cash" "Credit card" "Ewallet" ...
##  $ cogs                   : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ gross_margin_percentage: num  4.76 4.76 4.76 4.76 4.76 ...
##  $ gross_income           : num  26.14 3.82 16.22 23.29 30.21 ...
##  $ rating                 : num  9.1 9.6 7.4 8.4 5.3 4.1 5.8 8 7.2 5.9 ...
##  $ total                  : num  549 80.2 340.5 489 634.4 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

The dataset has character, integer and numerical datatypes Time and date are in the incorrect format

```
# Change date to date format
df$date <- as.Date(df$date, "%m/%d/%Y")

# Change time to time format

df$time <- as.ITime(df$time)

head(df)
```

```
##     invoice_id branch customer_type gender           product_line unit_price
## 1: 750-67-8428      A        Member Female      Health and beauty      74.69
## 2: 226-31-3081      C        Normal Female Electronic accessories      15.28
## 3: 631-41-3108      A        Normal   Male     Home and lifestyle      46.33
## 4: 123-19-1176      A        Member   Male      Health and beauty      58.22
## 5: 373-73-7910      A        Normal   Male      Sports and travel      86.31
## 6: 699-14-3026      C        Normal   Male Electronic accessories      85.39
##    quantity     tax       date     time     payment   cogs
## 1:        7 26.1415 2019-01-05 13:08:00     Ewallet 522.83
## 2:        5  3.8200 2019-03-08 10:29:00        Cash  76.40
## 3:        7 16.2155 2019-03-03 13:23:00 Credit card 324.31
## 4:        8 23.2880 2019-01-27 20:33:00     Ewallet 465.76
## 5:        7 30.2085 2019-02-08 10:37:00     Ewallet 604.17
## 6:        7 29.8865 2019-03-25 18:30:00     Ewallet 597.73
##    gross_margin_percentage gross_income rating    total
## 1:                4.761905      26.1415    9.1 548.9715
## 2:                4.761905       3.8200    9.6  80.2200
## 3:                4.761905      16.2155    7.4 340.5255
## 4:                4.761905      23.2880    8.4 489.0480
## 5:                4.761905      30.2085    5.3 634.3785
## 6:                4.761905      29.8865    4.1 627.6165
```

```
#Finding the total number of missing values in each column
colSums(is.na(df))
```

```
##              invoice_id                  branch           customer_type
##                       0                       0                       0
##                  gender            product_line              unit_price
##                       0                       0                       0
##                quantity                     tax                    date
##                       0                       0                       0
##                    time                 payment                    cogs
##                       0                       0                       0
## gross_margin_percentage            gross_income                  rating
##                       0                       0                       0
##                   total
##                       0
```

There are no missing values in the dataset

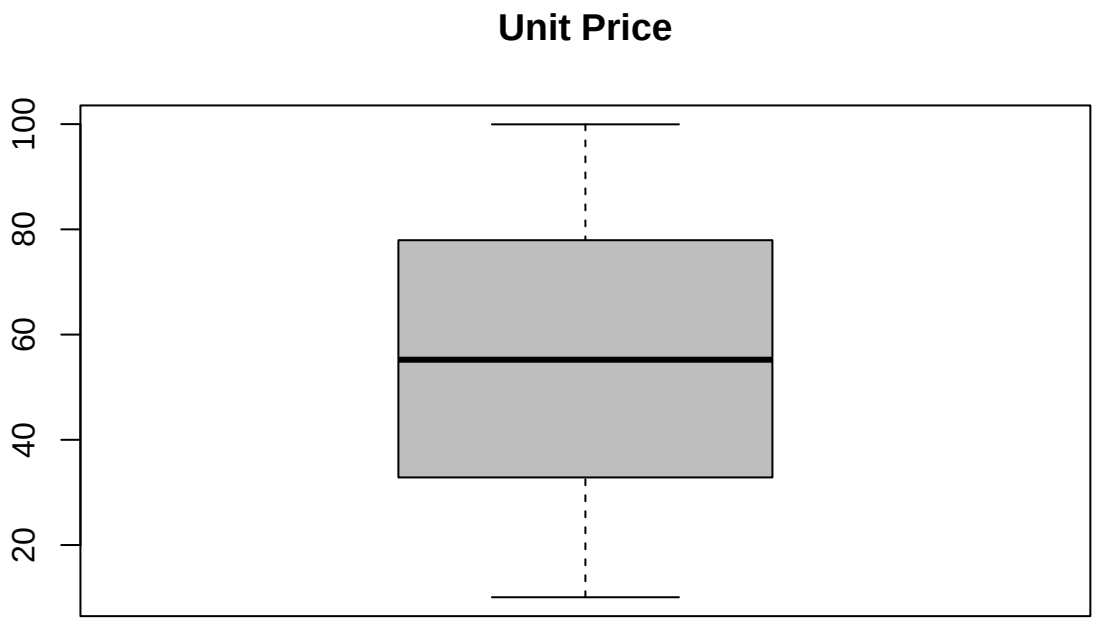
```
# Cheking for duplicates
df_dup <- df[duplicated(df),]
df_dup
```

```
## Empty data.table (0 rows and 16 cols): invoice_id,branch,customer_type,gender,product_line,unit_price
```

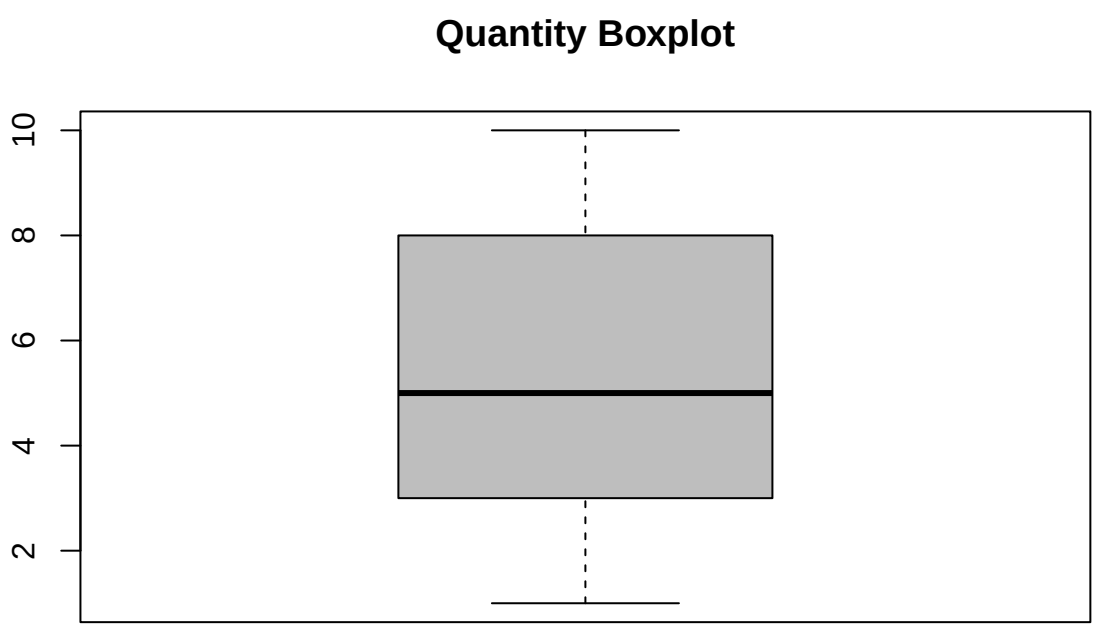
There is no duplicate data in this dataset

Checking for outliers

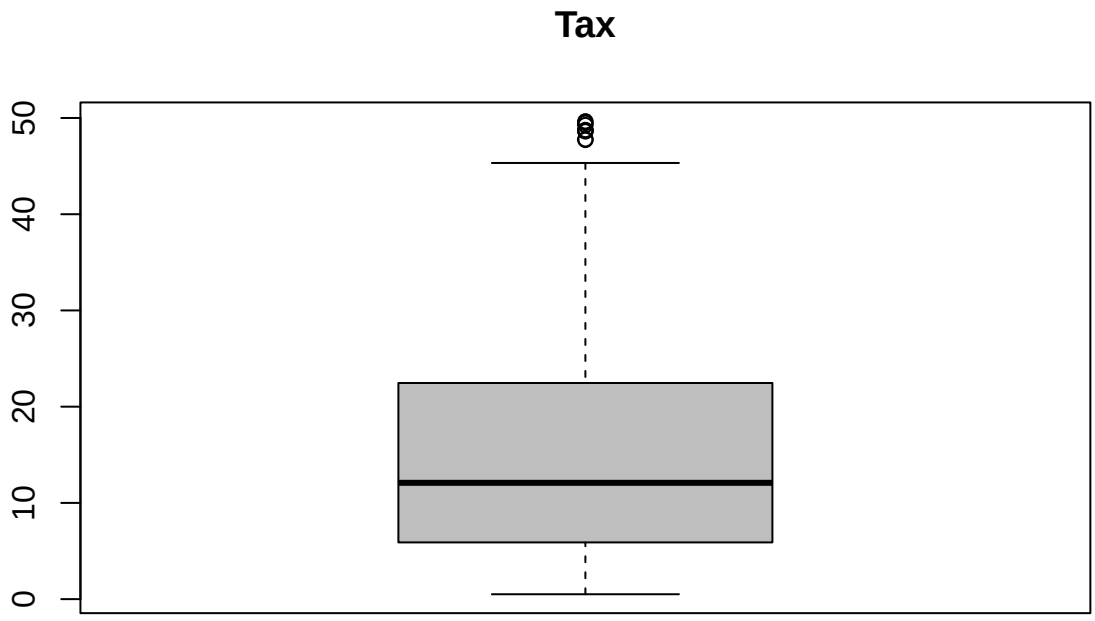
```
# Plotting boxplots to check for outliers
boxplot(df$unit_price,col='grey', main = 'Unit Price')
```



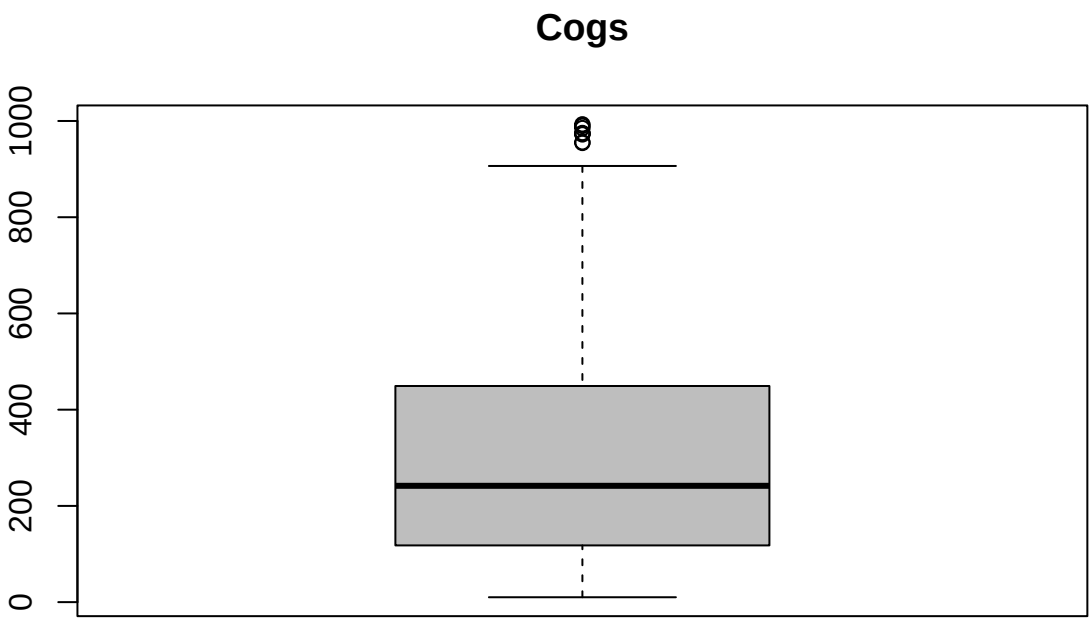

```
boxplot(df$quantity,col='grey', main = 'Quantity Boxplot')
```



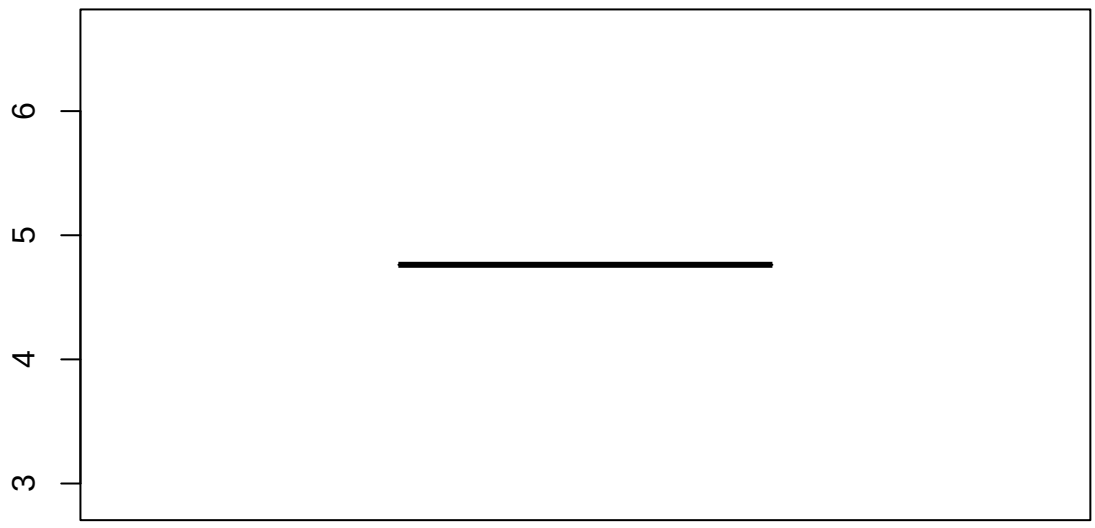

```
boxplot(df$tax,col='grey', main = 'Tax')
```



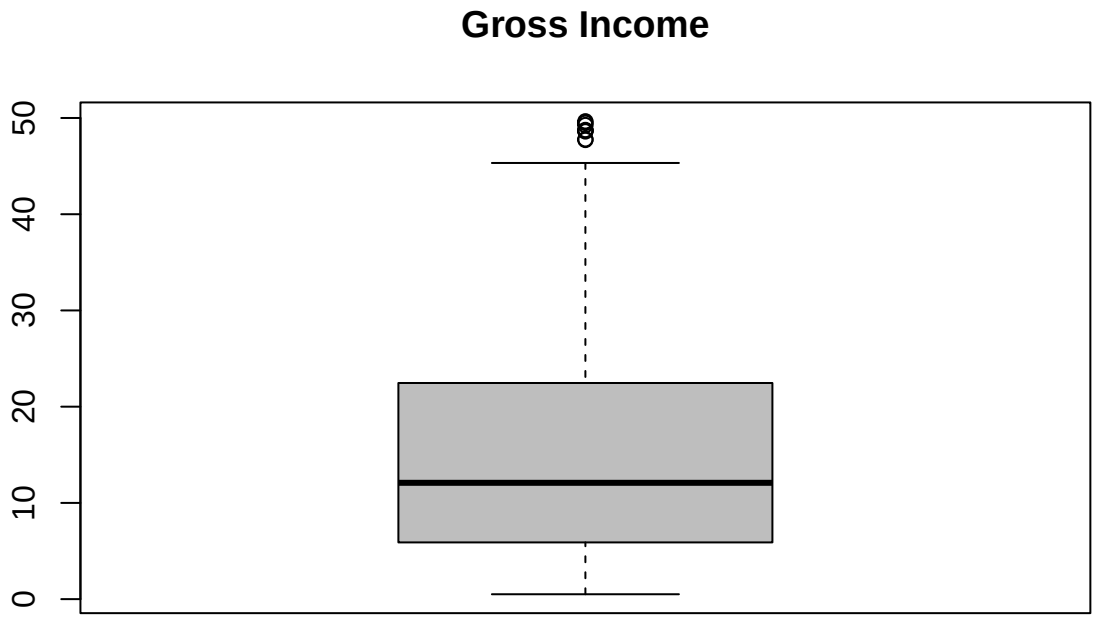

```
boxplot(df$cogs,col='grey', main = 'Cogs')
```



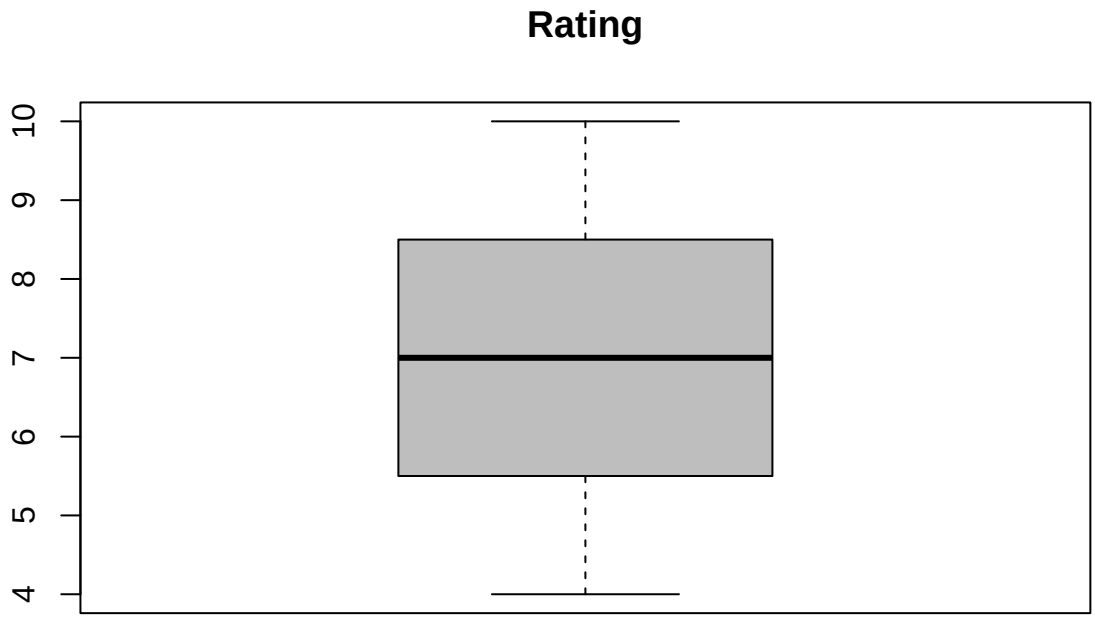

```
boxplot(df$gross_margin_percentage,col='grey', main = 'Gross Margin Percentage')
```



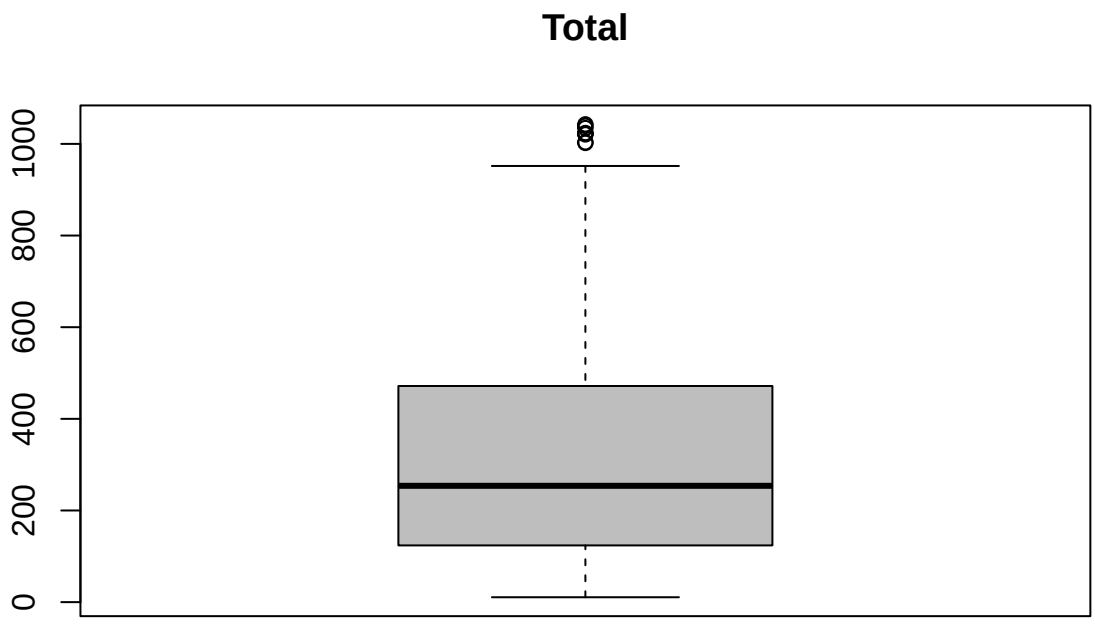

```
boxplot(df$gross_income,col='grey', main = 'Gross Income')
```




```
boxplot(df$rating,col='grey', main = 'Rating')
```




```
boxplot(df$total,col='grey', main = 'Total')
```

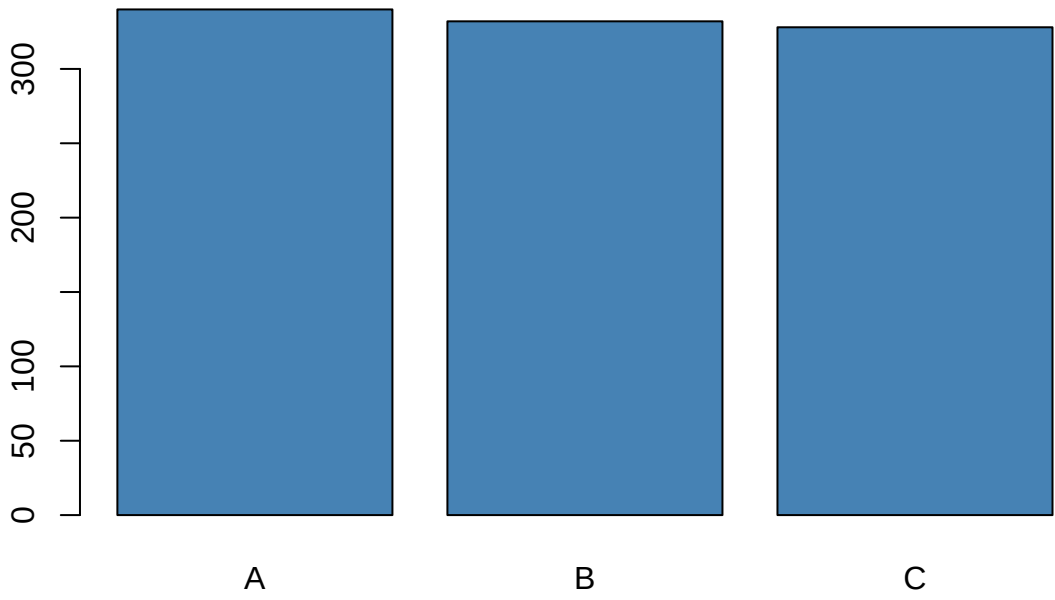


Tax, Cogs, Gross Income, Total has some outliers but we will leave them because they are actual representation of the data

Exploratory Data Analysis

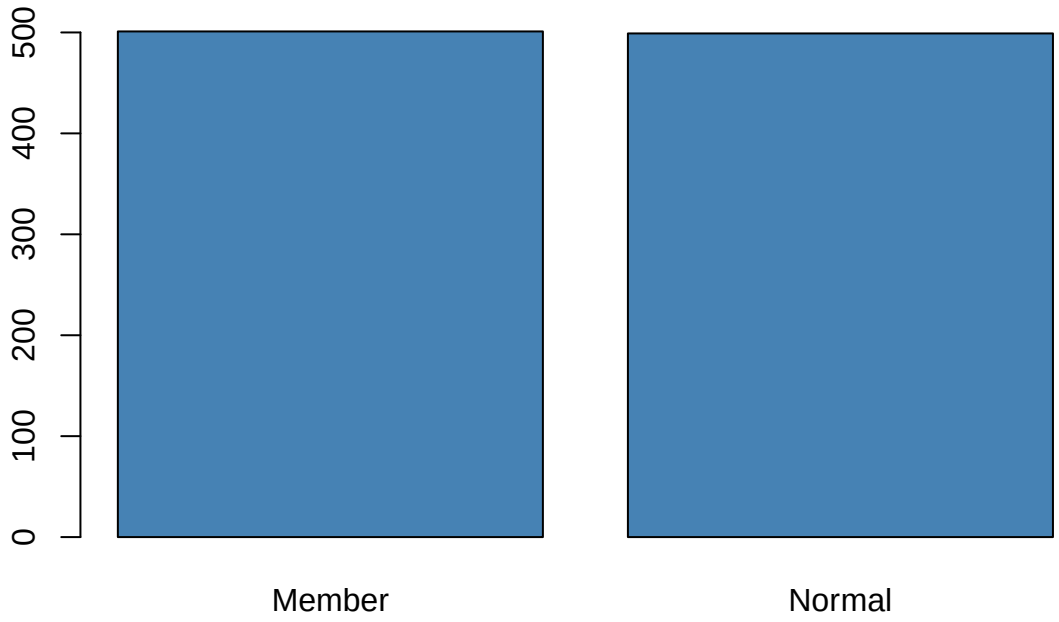
Univariate Analysis

```
# Frequency of categorical columns
#Branch , customer_type, Gender, productline , payment
branch <- table(df$branch)
barplot(branch, col = "steelblue")
```

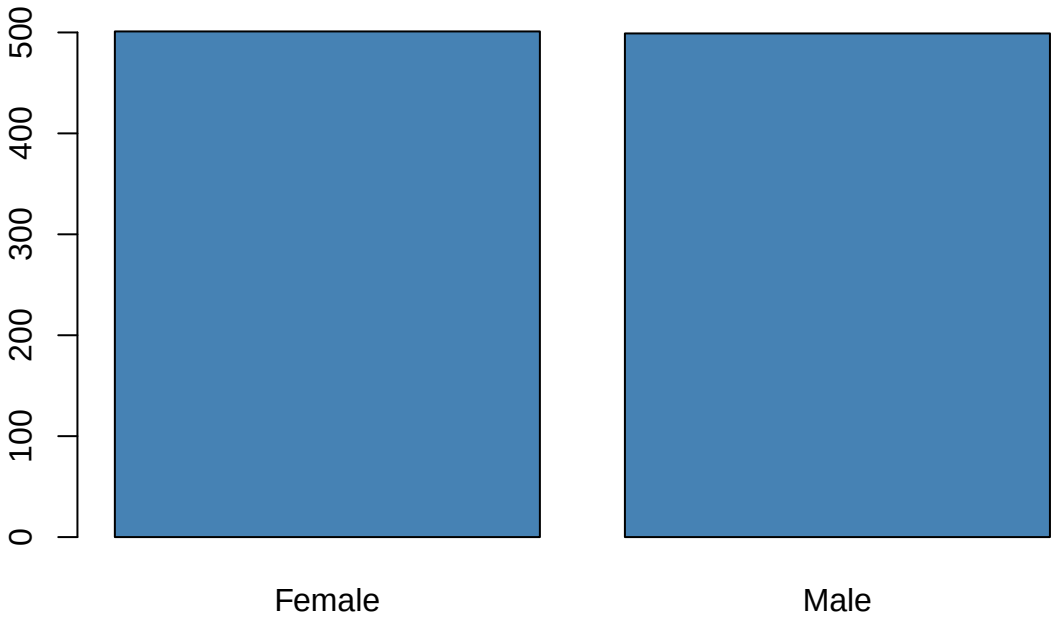


Categorical Variables

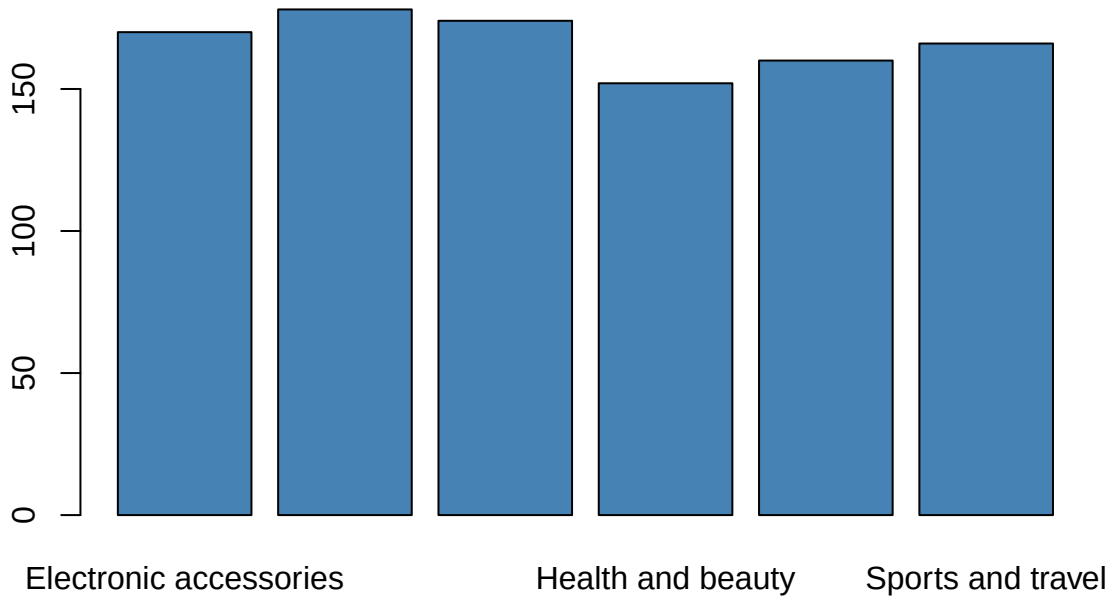
```
customer_type_freq <- table (df$customer_type)
barplot(customer_type_freq, col = "steelblue")
```



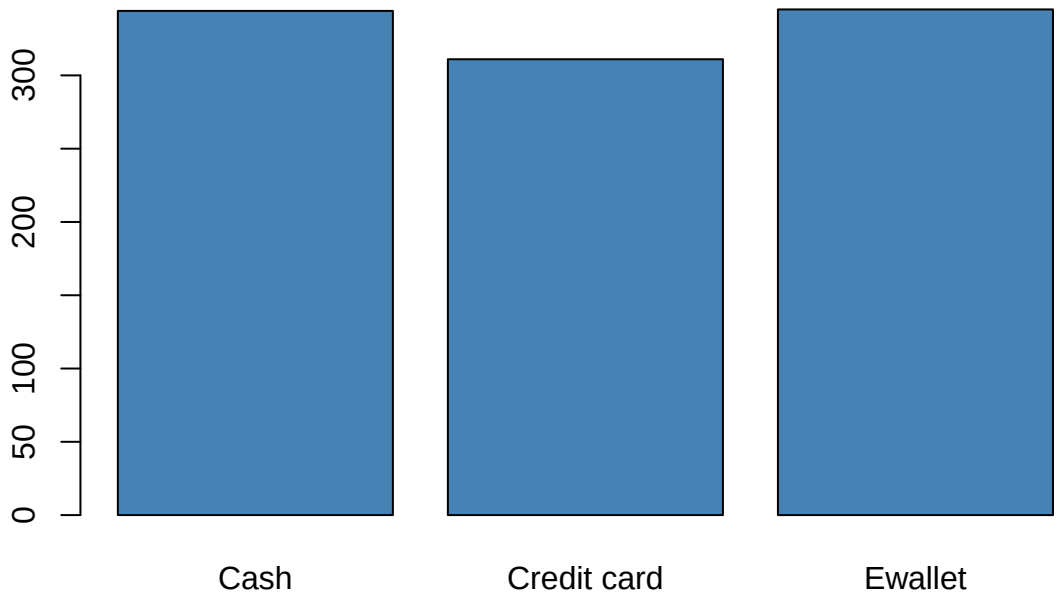

```
gender <- table(df$gender)
barplot(gender, col = "steelblue")
```




```
product_line <- table(df$product_line)
barplot(product_line, col = "steelblue")
```

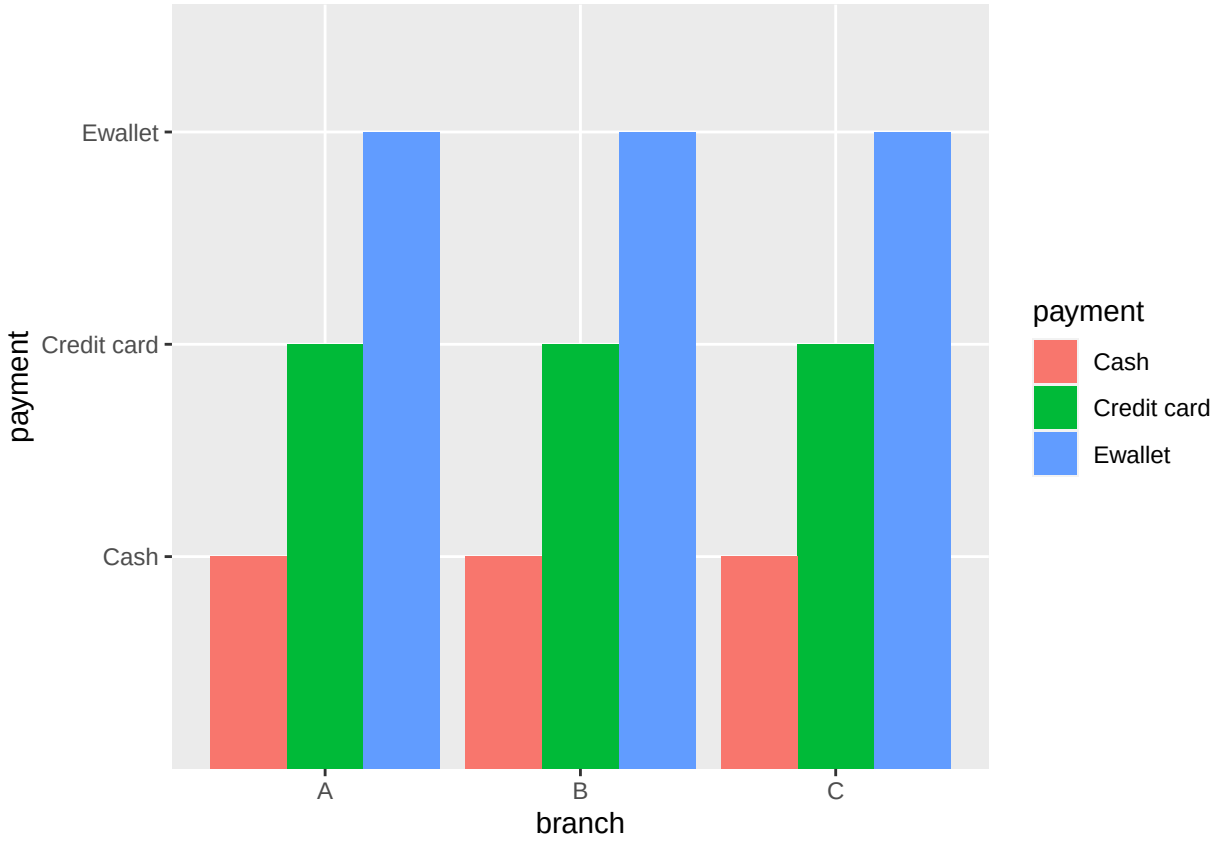



```
payment <- table(df$payment)
barplot(payment, col = "steelblue")
```



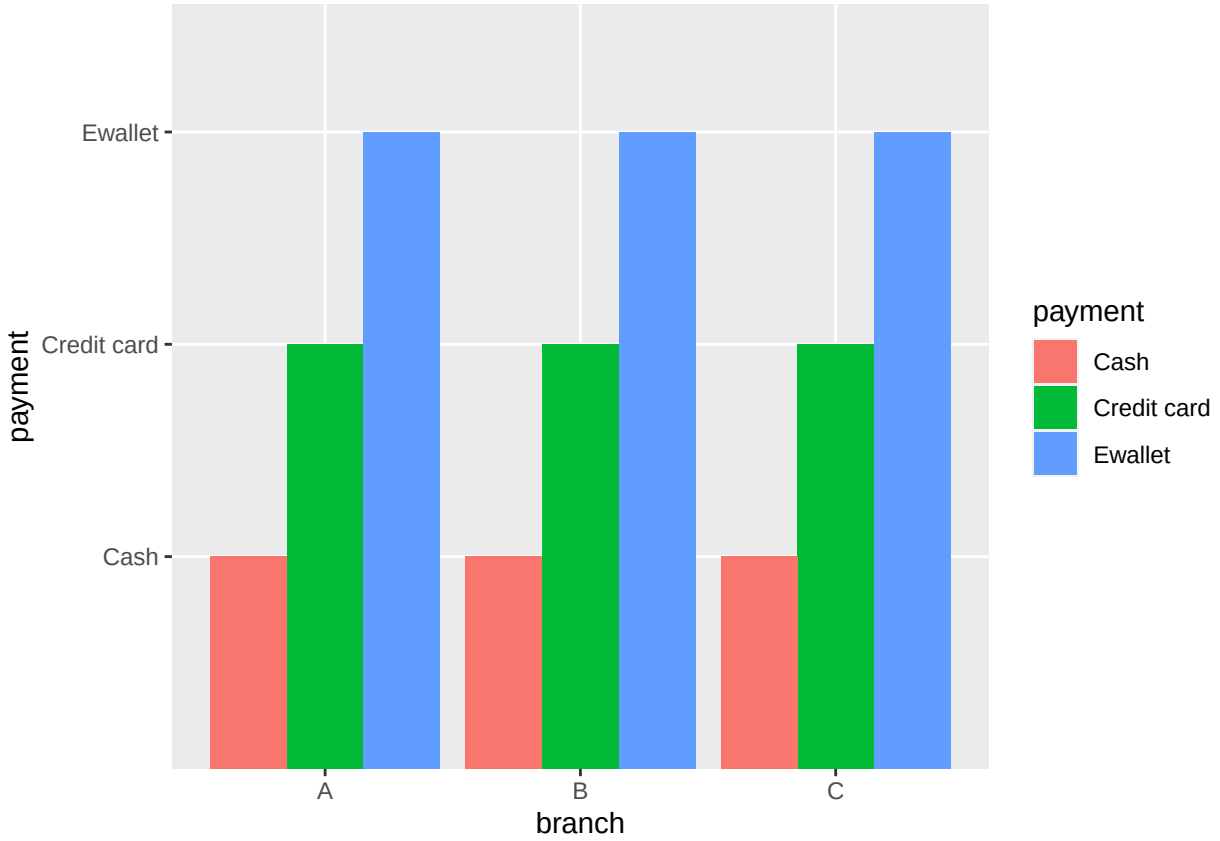
From the bar plots above we can conclude that: - The data is collected on Branches A, B and C equally. - The information collected was half from the members and half from the normal customers. - The gender was equally balances in the data. - Slightly More people paid their bills with E wallet and cash rather than Credit card

```
ggplot(df, aes(fill=payment, y= payment, x=branch)) +
    geom_bar(position="dodge", stat="identity")
```



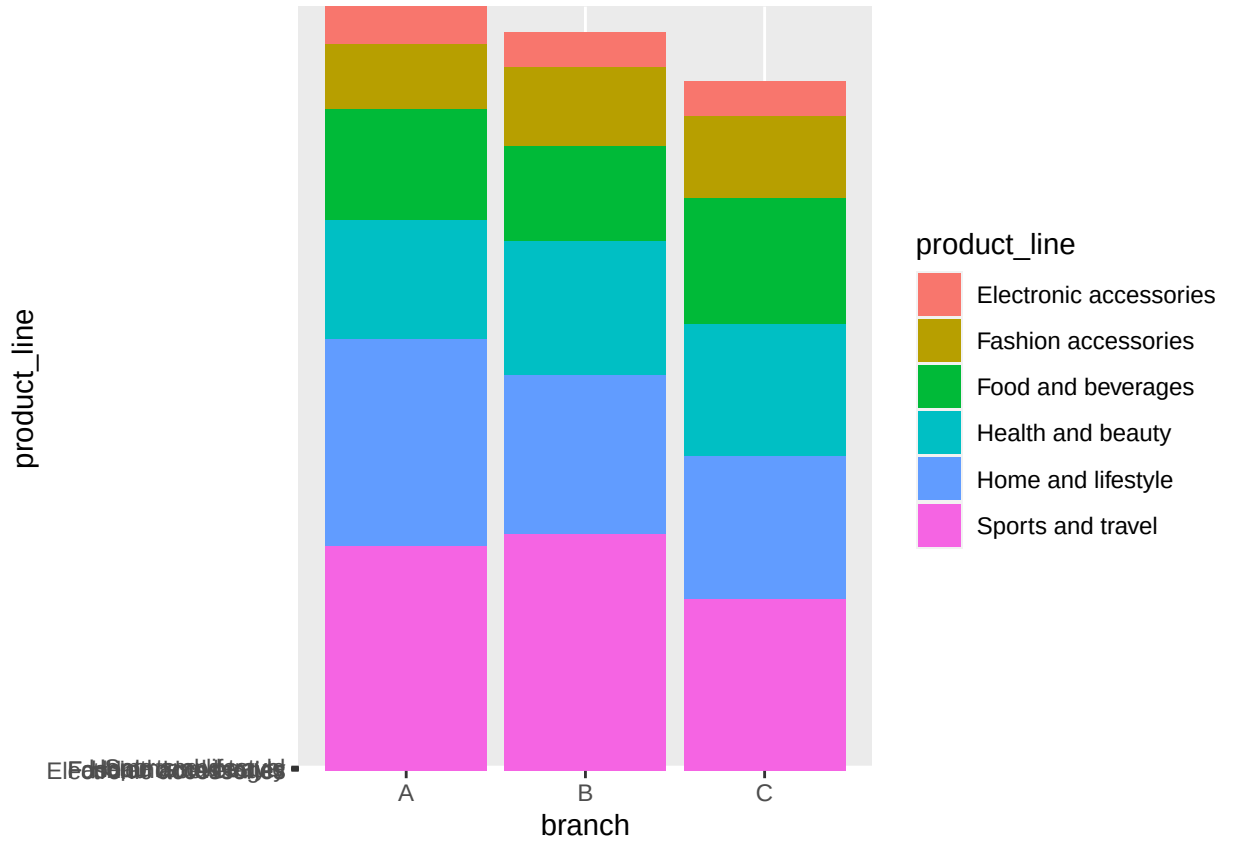
From the data, Ewallet payments are the most popular in all the three branches.

```
ggplot(df, aes(fill=payment, y= payment, x=branch)) +
    geom_bar(position="dodge", stat="identity")
```



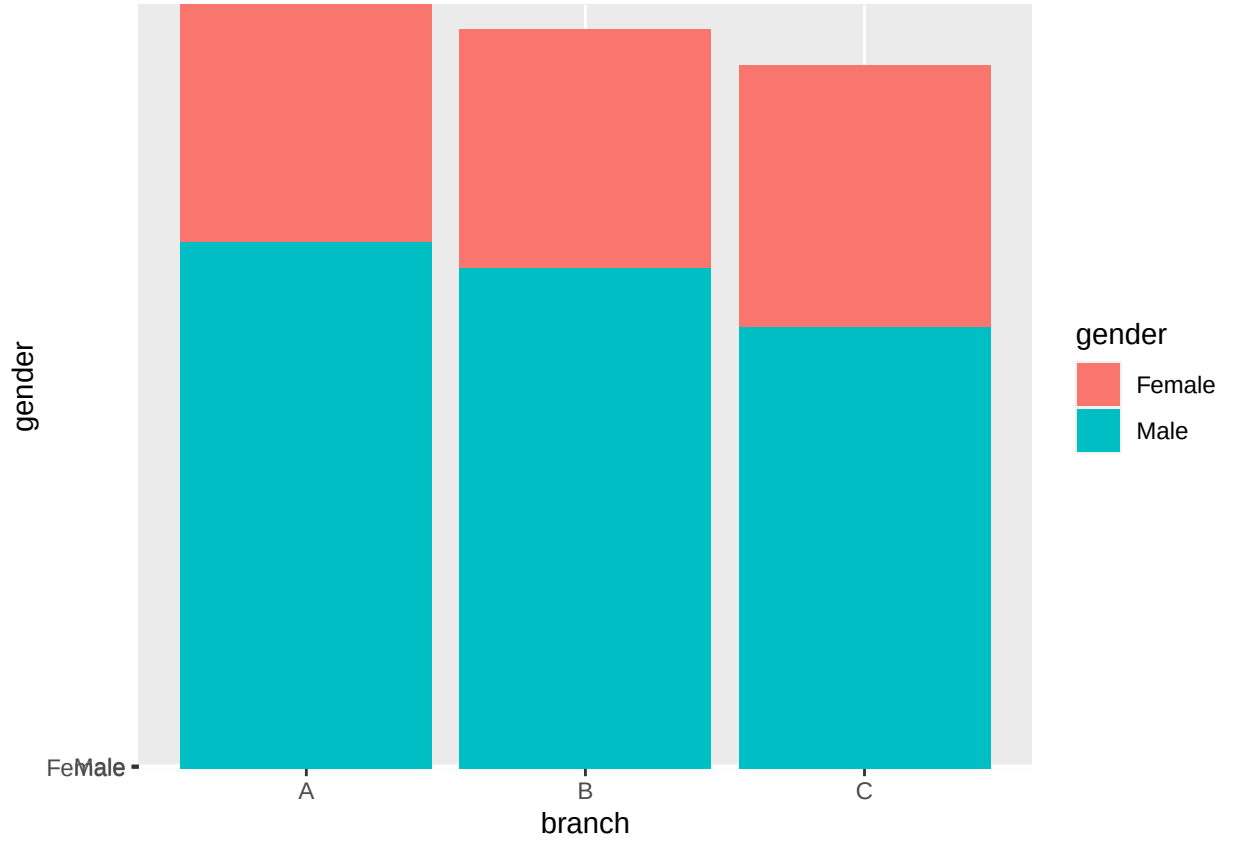
From the data, Ewallet payments are the most popular in all the three branches.

```
ggplot(df, aes(fill=product_line, y= product_line, x=branch)) +
    geom_bar(position="stack", stat="identity")
```



From the plot, Branch B sells more sports and travel goods than the other branches. Branch A sells more home and lifestyle goods than the other branches. Therefore, the marketing team should stack these branches with the product with which they sell more.

```
ggplot(df, aes(fill=gender, y= gender, x=branch)) +
    geom_bar(position="stack", stat="identity")
```



There are more males in the Carrefour branches than the females. This is not what many people assume as many people erroneously think that there are usually more females doing shopping.

Measures of central tendency for the numerical columns

```
# numerical columns.
num_col <- unlist(lapply(df, is.numeric))
df_num <- subset(df, select = num_col)
head (df_num)
```

```
##    unit_price quantity     tax     time   cogs gross_margin_percentage
## 1:      74.69        7 26.1415 13:08:00 522.83                4.761905
## 2:      15.28        5  3.8200 10:29:00  76.40                4.761905
## 3:      46.33        7 16.2155 13:23:00 324.31                4.761905
## 4:      58.22        8 23.2880 20:33:00 465.76                4.761905
## 5:      86.31        7 30.2085 10:37:00 604.17                4.761905
## 6:      85.39        7 29.8865 18:30:00 597.73                4.761905
##    gross_income rating    total
## 1:      26.1415    9.1 548.9715
## 2:       3.8200    9.6  80.2200
## 3:      16.2155    7.4 340.5255
## 4:      23.2880    8.4 489.0480
## 5:      30.2085    5.3 634.3785
## 6:      29.8865    4.1 627.6165
```

```r
#Getting the measures of dispersion in the numerical columns.
summary_stats <- data.frame(
  Mean = apply(df_num, 2, mean),
  Median = apply(df_num, 2, median),
  Min = apply(df_num, 2, min),
  Max = apply(df_num, 2, max))
summary_stats
```

```
##                               Mean       Median          Min          Max
## unit_price                55.672130    55.230000    10.080000    99.960000
## quantity                   5.510000     5.000000     1.000000    10.000000
## tax                       15.379369    12.088000     0.508500    49.650000
## time                   55481.880000 55140.000000 36000.000000 75540.000000
## cogs                     307.587380   241.760000    10.170000   993.000000
## gross_margin_percentage    4.761905     4.761905     4.761905     4.761905
## gross_income              15.379369    12.088000     0.508500    49.650000
## rating                     6.972700     7.000000     4.000000    10.000000
## total                    322.966749   253.848000    10.678500  1042.650000
```

```r
# Define the function
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
```

```r
# Mode
mode.unit_price <- getmode(df$unit_price)
mode.unit_price
```

```
## [1] 83.77
```

```r
mode.quantity <- getmode(df$quantity)
mode.quantity
```

```
## [1] 10
```

```r
mode.tax <- getmode(df$tax)
mode.tax
```

```
## [1] 39.48
```

```r
mode.cogs <- getmode(df$cogs)
mode.cogs
```

```
## [1] 789.6
```

```r
mode.gross_income <- getmode(df$gross_income)
mode.gross_income
```

```
## [1] 39.48
```

```
mode.rating <- getmode(df$rating)
mode.rating
```

```
## [1] 6
```

```
mode.total  <- getmode(df$total)
mode.total
```

```
## [1] 829.08
```

```
# Label Encoder
#Branch , customer_type, Gender, productline , payment
lbl <- LabelEncoder$new()
lbl$fit(df$branch)
df$branch <- lbl$fit_transform(df$branch)
lbl$fit(df$customer_type)
df$customer_type <- lbl$fit_transform(df$customer_type)
lbl$fit(df$gender)
df$gender <- lbl$fit_transform(df$gender)
lbl$fit(df$product_line)
df$product_line <- lbl$fit_transform(df$product_line)
lbl$fit(df$payment)
df$payment <- lbl$fit_transform(df$payment)
```

```
str(df)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  16 variables:
##  $ invoice_id             : chr  "750-67-8428" "226-31-3081" "631-41-3108" "123-19-1176" ...
##  $ branch                 : num  0 1 0 0 0 1 0 1 0 2 ...
##  $ customer_type          : num  0 1 1 0 1 1 0 1 0 0 ...
##  $ gender                 : num  0 0 1 1 1 1 0 0 0 0 ...
##  $ product_line           : num  0 1 2 0 3 1 1 2 0 4 ...
##  $ unit_price             : num  74.7 15.3 46.3 58.2 86.3 ...
##  $ quantity               : int  7 5 7 8 7 7 6 10 2 3 ...
##  $ tax                    : num  26.14 3.82 16.22 23.29 30.21 ...
##  $ date                   : Date, format: "2019-01-05" "2019-03-08" ...
##  $ time                   : 'ITime' int  13:08:00 10:29:00 13:23:00 20:33:00 10:37:00 18:30:00 14:36
##  $ payment                : num  0 1 2 0 0 0 0 0 2 2 ...
##  $ cogs                   : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ gross_margin_percentage: num  4.76 4.76 4.76 4.76 4.76 ...
##  $ gross_income           : num  26.14 3.82 16.22 23.29 30.21 ...
##  $ rating                 : num  9.1 9.6 7.4 8.4 5.3 4.1 5.8 8 7.2 5.9 ...
##  $ total                  : num  549 80.2 340.5 489 634.4 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Since the gross margin percentage has only one value we can drop the column.
table(df$gross_margin_percentage)
```

```
##
## 4.761904762
##        1000
```

```
df$gross_margin_percentage <- NULL
```

```
# Drop the categorcal columns
df$invoice_id <- NULL
df$date <- NULL
df$time <- NULL
```

```
# Separate the data
df.x <-  df[ , 1:11]
df.y <-  df[, 12]
```

```
head(df.x)
```

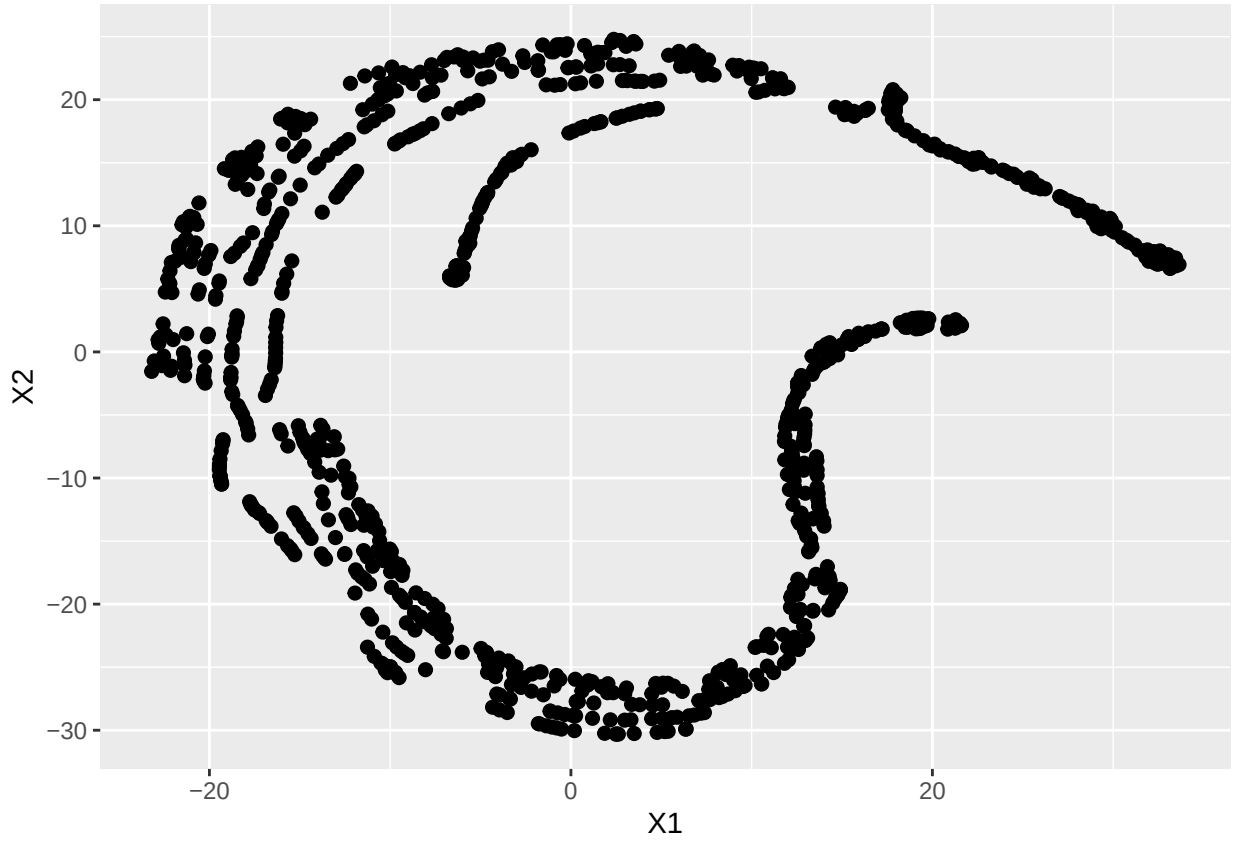
```
##    branch customer_type gender product_line unit_price quantity     tax payment
## 1:      0             0      0            0      74.69        7 26.1415       0
## 2:      1             1      0            1      15.28        5  3.8200       1
## 3:      0             1      1            2      46.33        7 16.2155       2
## 4:      0             0      1            0      58.22        8 23.2880       0
## 5:      0             1      1            3      86.31        7 30.2085       0
## 6:      1             1      1            1      85.39        7 29.8865       0
##      cogs gross_income rating
## 1: 522.83      26.1415    9.1
## 2:  76.40       3.8200    9.6
## 3: 324.31      16.2155    7.4
## 4: 465.76      23.2880    8.4
## 5: 604.17      30.2085    5.3
## 6: 597.73      29.8865    4.1
```

```
head(df.y)
```

```
##       total
## 1: 548.9715
## 2:  80.2200
## 3: 340.5255
## 4: 489.0480
## 5: 634.3785
## 6: 627.6165
```

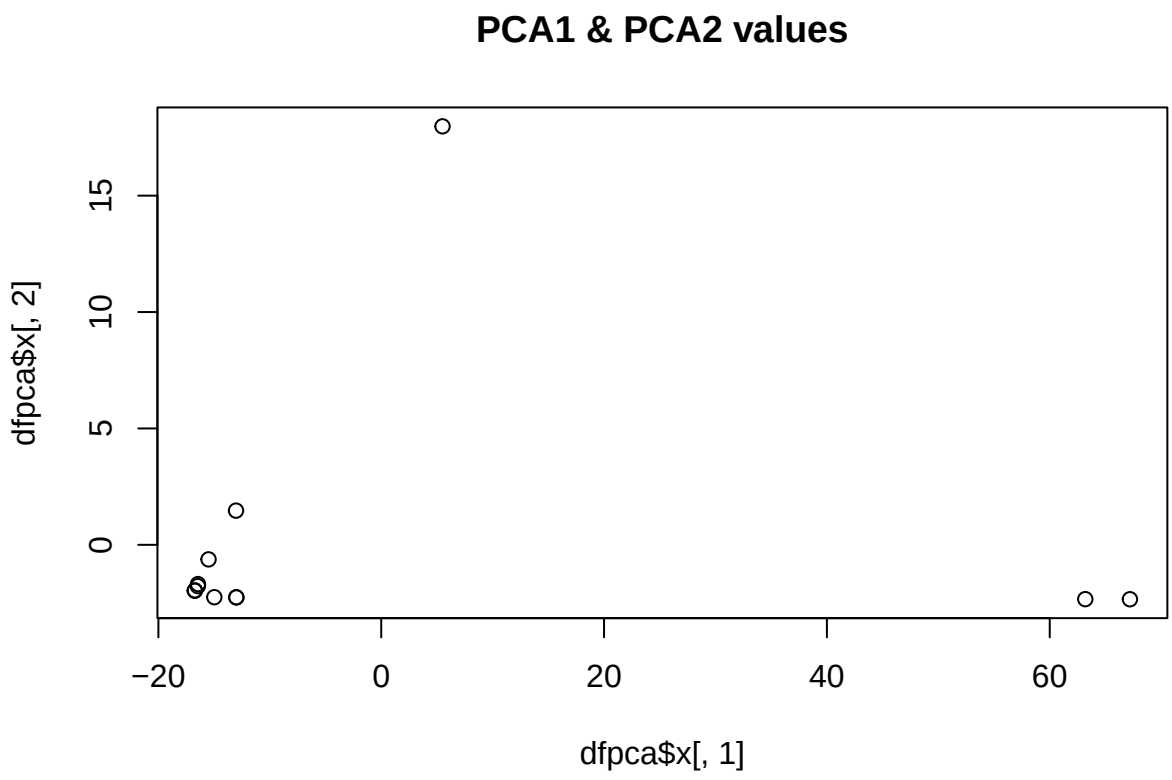
```
# perform tsne
library(Rtsne)
tsne = Rtsne(df.x, dims = 2,  perplexity = 30)
```

```
#visualize TSNE
df.tsne = data.frame(tsne$Y)
ggplot(df.tsne, aes(x=X1, y=X2)) + geom_point(size=2)
```



Performing the PCA

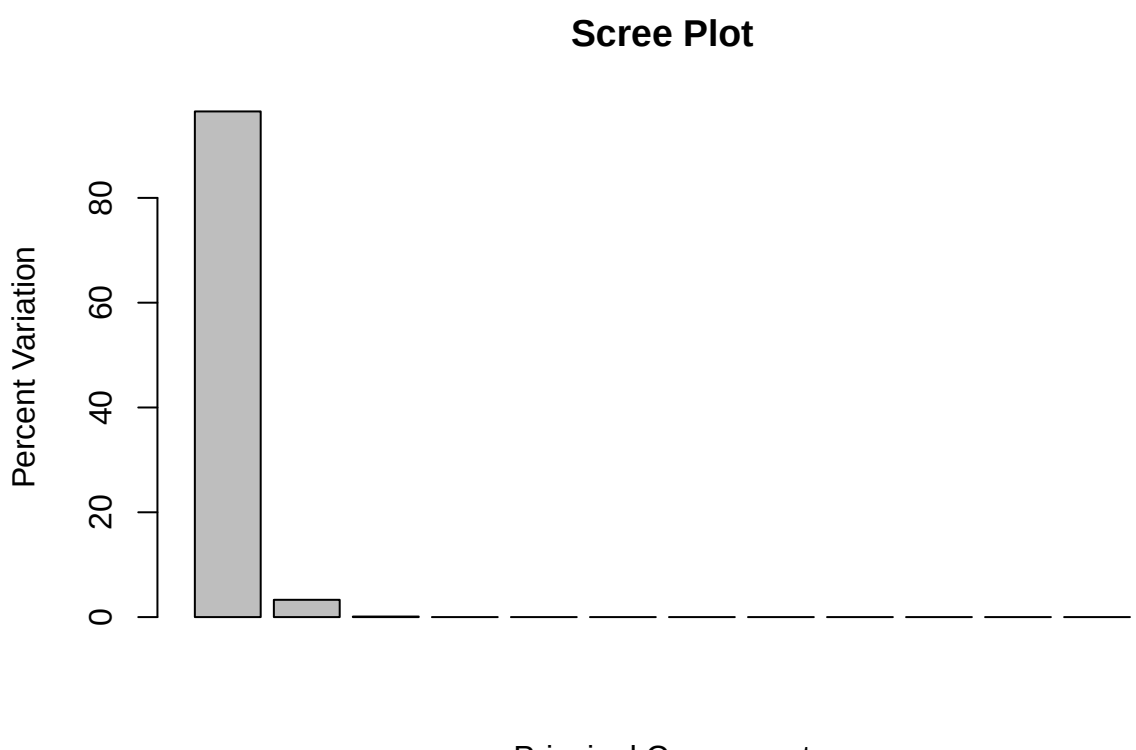
```
# Run the PCA on the df
dfpca <- prcomp(t(df),center = TRUE, scale=TRUE)
## plot pc1 and pc2
plot(dfpca$x[,1], dfpca$x[,2], main = "PCA1 & PCA2 values")
```




```
# Lets get a summary of the pca
summary (dfpca)
```

```
## Importance of components:
##                             PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     31.0616 5.76498 1.21319 0.50237 0.29831 0.23451 0.20497
## Proportion of Variance  0.9648 0.03323 0.00147 0.00025 0.00009 0.00005 0.00004
## Cumulative Proportion   0.9648 0.99806 0.99953 0.99978 0.99987 0.99993 0.99997
##                            PC8     PC9      PC10      PC11      PC12
## Standard deviation     0.14119 0.09579 2.638e-14 1.965e-15 6.211e-17
## Proportion of Variance 0.00002 0.00001 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  0.99999 1.00000 1.000e+00 1.000e+00 1.000e+00
```

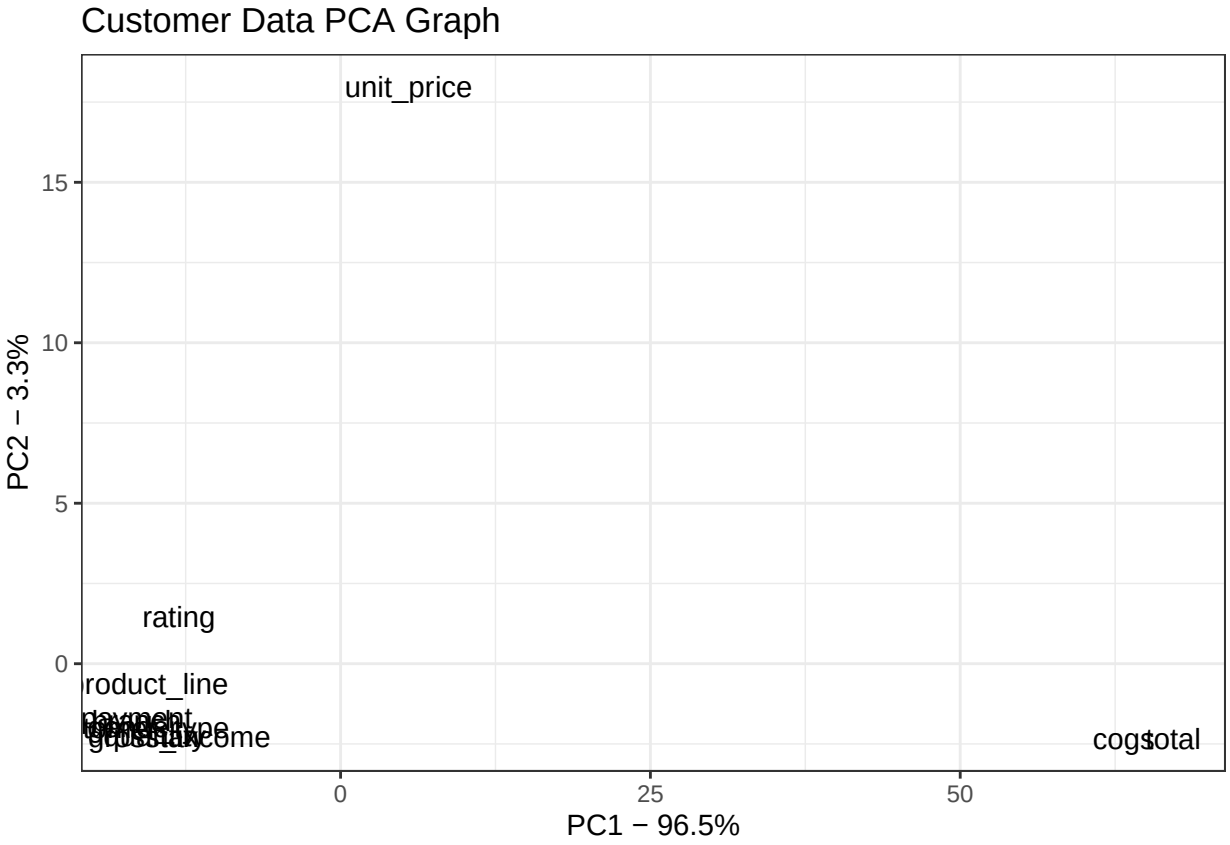
```
## make a scree plot
pca.var <- dfpca$sdev^2
pca.var.per <- round(pca.var/sum(pca.var)*100, 1)
barplot(pca.var.per, main="Scree Plot", xlab="Principal Component", ylab="Percent Variation")
```




```
## plot that shows the PCs and the variation:
pca.data <- data.frame(Sample=rownames(dfpca$x),
                       X=dfpca$x[,1],
                       Y=dfpca$x[,2])
pca.data
```

```
##                       Sample          X         Y
## branch                branch -16.460925 -1.774218
## customer_type customer_type -16.728657 -1.974943
## gender                gender -16.727800 -1.955175
## product_line   product_line -15.501089 -0.625429
## unit_price       unit_price   5.501295 17.977265
## quantity           quantity -14.979897 -2.249242
## tax                     tax -13.006234 -2.255524
## payment             payment -16.446861 -1.686001
## cogs                   cogs  63.189817 -2.333115
## gross_income   gross_income -13.006234 -2.255524
## rating               rating -13.033551  1.469104
## total                 total  67.200135 -2.337199
```

```
ggplot(data=pca.data, aes(x=X, y=Y, label=Sample)) +
  geom_text() +
  xlab(paste("PC1 - ", pca.var.per[1], "%", sep="")) +
  ylab(paste("PC2 - ", pca.var.per[2], "%", sep="")) +
  theme_bw() +
  ggtitle("Customer Data PCA Graph")
```



PC1 explains 96.5% of the total variance, which means that nearly 96% of the information in the dataset (11 variables) can be encapsulated by just that one Principal Component. PC2 explains 3.3% of the variance. etc

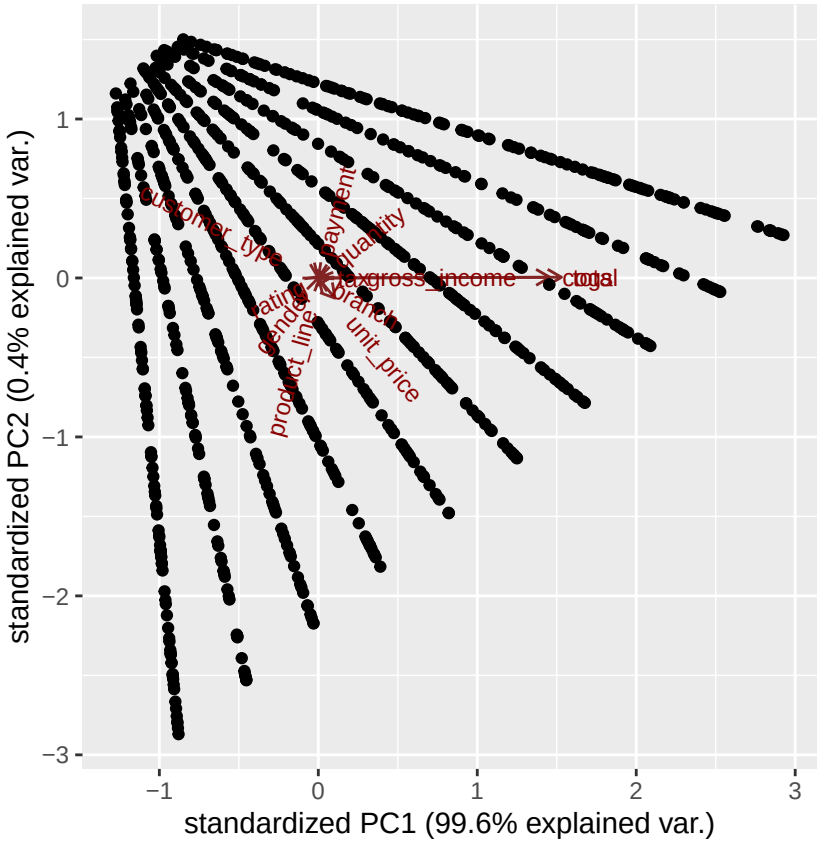
```
library(ggbiplot)
```

```
## Loading required package: plyr
```

```
## Loading required package: scales
```

```
## Loading required package: grid
```

```
ggbiplot (prcomp(df))
```


standardized PC2 (0.4% explained var.)
standardized PC1 (99.6% explained var.)
1
0
−1
−2
−3
−1
0
1
2
3
customer_type
payment
quantity
gross_income
total
rating
gender
product_line
branch
unit_price