Carrefour part1 and 2

2022-08-01

Specifying the question

To determine the most relevant sales strategies for the marketing team at Carrefour.

Defining metric for succcess

To be able to create recommendations for the sales team, at Carrefour

Understanding the context

THis is a business that is in the field of retail and would therefore need an analyst to review their sales and identify trends that would lead them to increase their sales.

```
library(data.table) # High-performance data frame package
library(tidyverse) # A Data exploration & visualization Package
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::transpose() masks data.table::transpose()
```

```
library(corrr) # A Correlation package
library(dplyr) # A Data Manipulation package
library(caret) # regression and correlation
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(corrplot)# visual exploration
```

```
## corrplot 0.92 loaded
```

```r
library(clustvarsel) # variable selection
```

```
## Loading required package: mclust
## Package 'mclust' version 5.4.10
## Type 'citation("mclust")' for citing this R package in publications.
##
## Attaching package: 'mclust'
##
## The following object is masked from 'package:purrr':
##
##     map
##
## Package 'clustvarsel' version 2.3.4
## Type 'citation("clustvarsel")' for citing this R package in publications.
```

```r
df<- read.csv("http://bit.ly/CarreFourDataset")
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
```

```r
# shape of our data set
dim(df)
```

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

```r
# information about our data set
str(df)
```

```
## '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" "I
##  $ 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 ...
```

```r
# convert the data into a tibble
df_sales<-as_tibble(df)
df_sales
```

```
## # A tibble: 1,000 x 16
##    Invoice.ID  Branch Custome~1 Gender Produ~2 Unit.~3 Quant~4   Tax Date  Time
##    <chr>       <chr>  <chr>     <chr>  <chr>     <dbl>   <int> <dbl> <chr> <chr>
##  1 750-67-8428 A      Member    Female Health~    74.7       7 26.1  1/5/~ 13:08
##  2 226-31-3081 C      Normal    Female Electr~    15.3       5  3.82 3/8/~ 10:29
##  3 631-41-3108 A      Normal    Male   Home a~    46.3       7 16.2  3/3/~ 13:23
##  4 123-19-1176 A      Member    Male   Health~    58.2       8 23.3  1/27~ 20:33
##  5 373-73-7910 A      Normal    Male   Sports~    86.3       7 30.2  2/8/~ 10:37
##  6 699-14-3026 C      Normal    Male   Electr~    85.4       7 29.9  3/25~ 18:30
##  7 355-53-5943 A      Member    Female Electr~    68.8       6 20.7  2/25~ 14:36
##  8 315-22-5665 C      Normal    Female Home a~    73.6      10 36.8  2/24~ 11:38
##  9 665-32-9167 A      Member    Female Health~    36.3       2  3.63 1/10~ 17:15
## 10 692-92-5582 B      Member    Female Food a~    54.8       3  8.23 2/20~ 13:27
## # ... with 990 more rows, 6 more variables: Payment <chr>, cogs <dbl>,
## #   gross.margin.percentage <dbl>, gross.income <dbl>, Rating <dbl>,
## #   Total <dbl>, and abbreviated variable names 1: Customer.type,
## #   2: Product.line, 3: Unit.price, 4: Quantity
## # i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

```r
# dataset summary
summary(df_sales)
```

```
##   Invoice.ID           Branch          Customer.type         Gender
##  Length:1000        Length:1000        Length:1000        Length:1000
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
```

```
##
##
##  Product.line         Unit.price       Quantity         Tax
##  Length:1000        Min.   :10.08   Min.   : 1.00   Min.   : 0.5085
##  Class :character   1st Qu.:32.88   1st Qu.: 3.00   1st Qu.: 5.9249
##  Mode  :character   Median :55.23   Median : 5.00   Median :12.0880
##                     Mean   :55.67   Mean   : 5.51   Mean   :15.3794
##                     3rd Qu.:77.94   3rd Qu.: 8.00   3rd Qu.:22.4453
##                     Max.   :99.96   Max.   :10.00   Max.   :49.6500
##      Date               Time             Payment              cogs
##  Length:1000        Length:1000        Length:1000        Min.   : 10.17
##  Class :character   Class :character   Class :character   1st Qu.:118.50
##  Mode  :character   Mode  :character   Mode  :character   Median :241.76
##                                                           Mean   :307.59
##                                                           3rd Qu.:448.90
##                                                           Max.   :993.00
##  gross.margin.percentage  gross.income         Rating           Total
##  Min.   :4.762           Min.   : 0.5085   Min.   : 4.000   Min.   :  10.68
##  1st Qu.:4.762           1st Qu.: 5.9249   1st Qu.: 5.500   1st Qu.: 124.42
##  Median :4.762           Median :12.0880   Median : 7.000   Median : 253.85
##  Mean   :4.762           Mean   :15.3794   Mean   : 6.973   Mean   : 322.97
##  3rd Qu.:4.762           3rd Qu.:22.4453   3rd Qu.: 8.500   3rd Qu.: 471.35
##  Max.   :4.762           Max.   :49.6500   Max.   :10.000   Max.   :1042.65
```

```
# check for missing values
#
colSums(is.na(df_sales))
```

```
##              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
```

```
df_sales[!complete.cases(df_sales),]
```

```
## # A tibble: 0 x 16
## # ... with 16 variables: Invoice.ID <chr>, Branch <chr>, Customer.type <chr>,
## #   Gender <chr>, Product.line <chr>, Unit.price <dbl>, Quantity <int>,
## #   Tax <dbl>, Date <chr>, Time <chr>, Payment <chr>, cogs <dbl>,
## #   gross.margin.percentage <dbl>, gross.income <dbl>, Rating <dbl>,
## #   Total <dbl>
## # i Use `colnames()` to see all variable names
```

```r
# Identifying Duplicated Data
# No duplicate values found

anyDuplicated(df_sales)
```

```
## [1] 0
```

```r
# Checking for numeric data types
#
Numeric<- df_sales %>% select_if(is.numeric)
Numeric
```

```
## # A tibble: 1,000 x 8
##    Unit.price Quantity   Tax  cogs gross.margin.percentage gross.~1 Rating Total
##         <dbl>    <int> <dbl> <dbl>                   <dbl>    <dbl>  <dbl> <dbl>
##  1       74.7        7 26.1  523.                     4.76    26.1     9.1 549.
##  2       15.3        5  3.82  76.4                    4.76     3.82    9.6  80.2
##  3       46.3        7 16.2  324.                     4.76    16.2     7.4 341.
##  4       58.2        8 23.3  466.                     4.76    23.3     8.4 489.
##  5       86.3        7 30.2  604.                     4.76    30.2     5.3 634.
##  6       85.4        7 29.9  598.                     4.76    29.9     4.1 628.
##  7       68.8        6 20.7  413.                     4.76    20.7     5.8 434.
##  8       73.6       10 36.8  736.                     4.76    36.8     8   772.
##  9       36.3        2  3.63  72.5                    4.76     3.63    7.2  76.1
## 10       54.8        3  8.23 165.                     4.76     8.23    5.9 173.
## # ... with 990 more rows, and abbreviated variable name 1: gross.income
## # i Use `print(n = ...)` to see more rows
```

Part 1: Dimensional Reduction

This section we will be reducing our data set to a low dimensional data set using the t-SNE algorithm or PCA.

Principal Component Analysis (PCA)

```r
# Selecting the numerical data
# PCA only works with numerical values, thus the selection of numerical variables from the dataset
numeric <- df_sales[,c(6:8,12:16)]
head(numeric)
```

```
## # A tibble: 6 x 8
##   Unit.price Quantity   Tax  cogs gross.margin.percentage gross.i~1 Rating Total
##        <dbl>    <int> <dbl> <dbl>                   <dbl>     <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.                     4.76     26.1     9.1 549.
## 2       15.3        5  3.82  76.4                    4.76      3.82    9.6  80.2
## 3       46.3        7 16.2  324.                     4.76     16.2     7.4 341.
## 4       58.2        8 23.3  466.                     4.76     23.3     8.4 489.
## 5       86.3        7 30.2  604.                     4.76     30.2     5.3 634.
## 6       85.4        7 29.9  598.                     4.76     29.9     4.1 628.
## # ... with abbreviated variable name 1: gross.income
```

```r
# Checking the variance
non_zero_var <- numeric[ , which(apply(numeric, 2, var) != 0)]
head(non_zero_var)
```

```
## # A tibble: 6 x 7
##   Unit.price Quantity   Tax  cogs gross.income Rating Total
##        <dbl>    <int> <dbl> <dbl>        <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.         26.1     9.1 549.
## 2       15.3        5  3.82  76.4         3.82    9.6  80.2
## 3       46.3        7 16.2  324.         16.2     7.4 341.
## 4       58.2        8 23.3  466.         23.3     8.4 489.
## 5       86.3        7 30.2  604.         30.2     5.3 634.
## 6       85.4        7 29.9  598.         29.9     4.1 628.
```

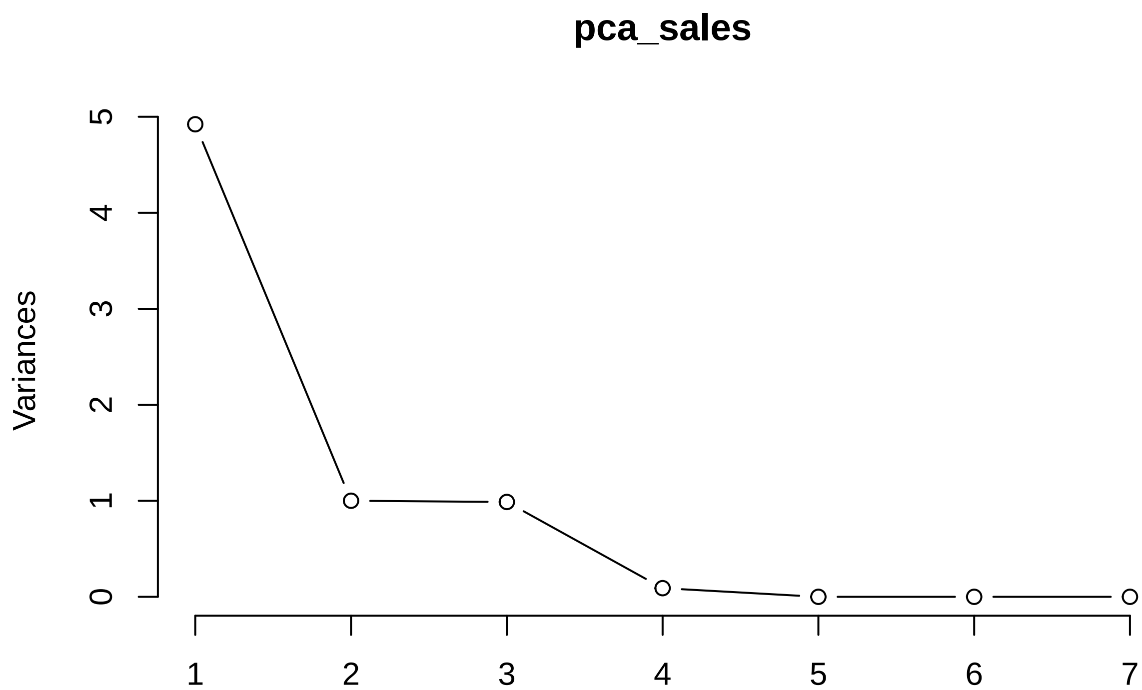
apply function is used instead of center to ensure there is no zero variance It also ensuring no column has zero mean. Therefore gross.margin.percentage column was removed.

```r
# now carrying out PCA with scale set to true
pca_sales <- prcomp(non_zero_var,scale=TRUE)
# previewing our PCA summary
summary(pca_sales)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4       PC5       PC6
## Standard deviation     2.2185 1.0002 0.9939 0.30001 3.132e-16 1.457e-16
## Proportion of Variance 0.7031 0.1429 0.1411 0.01286 0.000e+00 0.000e+00
## Cumulative Proportion  0.7031 0.8460 0.9871 1.00000 1.000e+00 1.000e+00
##                              PC7
## Standard deviation     3.219e-17
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```r
# PC1 describes 70.31% of the total variation of the dataset, PC2 describes 14.29%, and so on. PC1 has
```

```r
#plotting of pca
plot(pca_sales,type = 'l')
```




```
#Most of the variability in our data are in the first and second component of our PCA, Variance explain
```

Part 2: Feature Selection We will perform feature selection using unsupervised learniing

Filter Method

```
# dataset to be used for feature selection

feature<- df_sales
head(feature)
```

```
## # A tibble: 6 x 16
##   Invoice.ID  Branch Customer~1 Gender Produ~2 Unit.~3 Quant~4   Tax Date  Time
##   <chr>       <chr>  <chr>      <chr>  <chr>     <dbl>   <int> <dbl> <chr> <chr>
## 1 750-67-8428 A      Member     Female Health~    74.7       7 26.1  1/5/~ 13:08
## 2 226-31-3081 C      Normal     Female Electr~    15.3       5  3.82 3/8/~ 10:29
## 3 631-41-3108 A      Normal     Male   Home a~    46.3       7 16.2  3/3/~ 13:23
## 4 123-19-1176 A      Member     Male   Health~    58.2       8 23.3  1/27~ 20:33
## 5 373-73-7910 A      Normal     Male   Sports~    86.3       7 30.2  2/8/~ 10:37
## 6 699-14-3026 C      Normal     Male   Electr~    85.4       7 29.9  3/25~ 18:30
## # ... with 6 more variables: Payment <chr>, cogs <dbl>,
## #   gross.margin.percentage <dbl>, gross.income <dbl>, Rating <dbl>,
## #   Total <dbl>, and abbreviated variable names 1: Customer.type,
## #   2: Product.line, 3: Unit.price, 4: Quantity
## # i Use ‘colnames()‘ to see all variable names
```

```r
#Checking the columns with numeric datatypes
feature_num<- feature %>% select_if(is.numeric)
feature_num
```

```
## # A tibble: 1,000 x 8
##    Unit.price Quantity   Tax  cogs gross.margin.percentage gross.~1 Rating Total
##         <dbl>    <int> <dbl> <dbl>                   <dbl>    <dbl>  <dbl> <dbl>
##  1       74.7        7 26.1  523.                     4.76    26.1     9.1 549.
##  2       15.3        5  3.82  76.4                    4.76     3.82    9.6  80.2
##  3       46.3        7 16.2  324.                     4.76    16.2     7.4 341.
##  4       58.2        8 23.3  466.                     4.76    23.3     8.4 489.
##  5       86.3        7 30.2  604.                     4.76    30.2     5.3 634.
##  6       85.4        7 29.9  598.                     4.76    29.9     4.1 628.
##  7       68.8        6 20.7  413.                     4.76    20.7     5.8 434.
##  8       73.6       10 36.8  736.                     4.76    36.8     8   772.
##  9       36.3        2  3.63  72.5                    4.76     3.63    7.2  76.1
## 10       54.8        3  8.23 165.                     4.76     8.23    5.9 173.
## # ... with 990 more rows, and abbreviated variable name 1: gross.income
## # i Use `print(n = ...)` to see more rows
```

```r
# just like in pca, gross.margin.percentage column is dropped because it has a constant value
to_drop <- c("gross.margin.percentage")
```

```r
#dropping the highly correlated colunms
feature_num <- feature_num[, !names(feature_num) %in% to_drop]
head(feature_num)
```

```
## # A tibble: 6 x 7
##   Unit.price Quantity   Tax  cogs gross.income Rating Total
##        <dbl>    <int> <dbl> <dbl>        <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.         26.1     9.1 549.
## 2       15.3        5  3.82  76.4         3.82    9.6  80.2
## 3       46.3        7 16.2  324.         16.2     7.4 341.
## 4       58.2        8 23.3  466.         23.3     8.4 489.
## 5       86.3        7 30.2  604.         30.2     5.3 634.
## 6       85.4        7 29.9  598.         29.9     4.1 628.
```

```r
# Calculating the correlation matrix
cor_Matrix <- cor(feature_num)
cor_Matrix
```

```
##                 Unit.price    Quantity        Tax       cogs gross.income
## Unit.price    1.000000000  0.01077756  0.6339621  0.6339621    0.6339621
## Quantity      0.010777564  1.00000000  0.7055102  0.7055102    0.7055102
## Tax           0.633962089  0.70551019  1.0000000  1.0000000    1.0000000
## cogs          0.633962089  0.70551019  1.0000000  1.0000000    1.0000000
## gross.income  0.633962089  0.70551019  1.0000000  1.0000000    1.0000000
## Rating       -0.008777507 -0.01581490 -0.0364417 -0.0364417   -0.0364417
## Total         0.633962089  0.70551019  1.0000000  1.0000000    1.0000000
##                     Rating      Total
## Unit.price   -0.008777507  0.6339621
## Quantity     -0.015814905  0.7055102
```

```
## Tax          -0.036441705  1.0000000
## cogs         -0.036441705  1.0000000
## gross.income -0.036441705  1.0000000
## Rating        1.000000000 -0.0364417
## Total        -0.036441705  1.0000000
```

```
# Find attributes that are highly correlated
high_cor <- findCorrelation(cor_Matrix, cutoff=0.75)
high_cor
```

```
## [1] 4 7 3
```

```
# Finding highly correlated columns
names(feature_num[,high_cor])
```

```
## [1] "cogs"  "Total" "Tax"
```

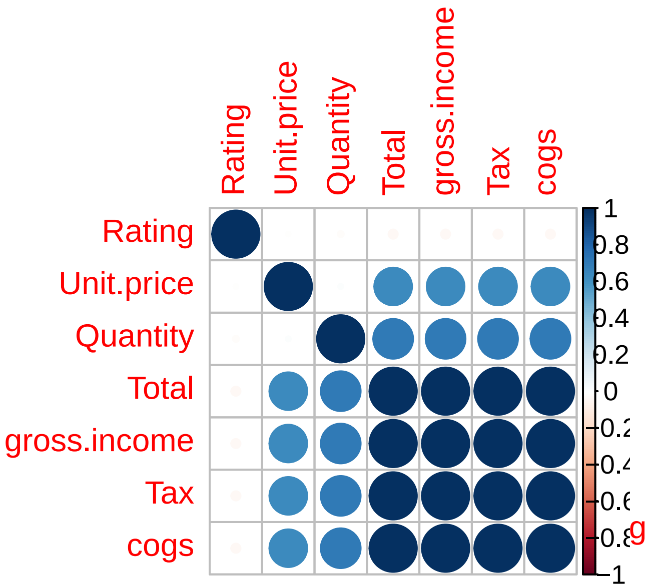
```
# Columns cogs, total and tax are highly correlated
```

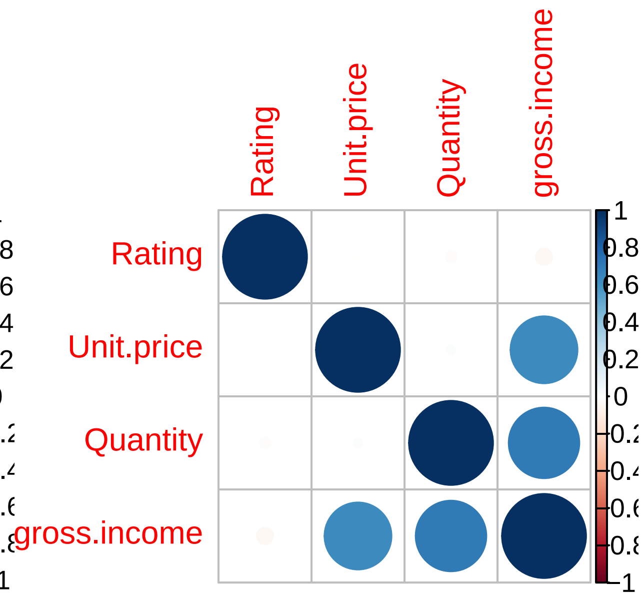
```
# We can remove the variables with a higher correlation
high_cor2<-feature_num[-high_cor]
high_cor2
```

```
## # A tibble: 1,000 x 4
##    Unit.price Quantity gross.income Rating
##         <dbl>    <int>        <dbl>  <dbl>
##  1       74.7        7        26.1     9.1
##  2       15.3        5         3.82    9.6
##  3       46.3        7        16.2     7.4
##  4       58.2        8        23.3     8.4
##  5       86.3        7        30.2     5.3
##  6       85.4        7        29.9     4.1
##  7       68.8        6        20.7     5.8
##  8       73.6       10        36.8     8
##  9       36.3        2         3.63    7.2
## 10       54.8        3         8.23    5.9
## # ... with 990 more rows
## # i Use `print(n = ...)` to see more rows
```

```
# Four columns remained after removing the highly correlated ones
```

```
# Performing our graphical comparison
par(mfrow = c(1, 2))
corrplot(cor_Matrix, order = "hclust")
corrplot(cor(high_cor2), order = "hclust")
```






```
#The final features that contribute the most information to the dataset are Unit.price, Quantity, and g
```

Wrapper Method

```
#getting the dataset
wrapper <- df_sales
#
#checking the dateset
#
head(wrapper)
```

```
## # A tibble: 6 x 16
##   Invoice.ID  Branch Customer~1 Gender Produ~2 Unit.~3 Quant~4   Tax Date  Time
##   <chr>       <chr>  <chr>      <chr>  <chr>     <dbl>   <int> <dbl> <chr> <chr>
## 1 750-67-8428 A      Member     Female Health~    74.7       7 26.1  1/5/~ 13:08
## 2 226-31-3081 C      Normal     Female Electr~    15.3       5  3.82 3/8/~ 10:29
## 3 631-41-3108 A      Normal     Male   Home a~    46.3       7 16.2  3/3/~ 13:23
## 4 123-19-1176 A      Member     Male   Health~    58.2       8 23.3  1/27~ 20:33
## 5 373-73-7910 A      Normal     Male   Sports~    86.3       7 30.2  2/8/~ 10:37
## 6 699-14-3026 C      Normal     Male   Electr~    85.4       7 29.9  3/25~ 18:30
## # ... with 6 more variables: Payment <chr>, cogs <dbl>,
## #   gross.margin.percentage <dbl>, gross.income <dbl>, Rating <dbl>,
## #   Total <dbl>, and abbreviated variable names 1: Customer.type,
## #   2: Product.line, 3: Unit.price, 4: Quantity
## # i Use 'colnames()' to see all variable names
```

```
# Selecting the numerical data
wrapper_num <- wrapper[,c(6:8,12:16)]
head(wrapper_num)
```

```
## # A tibble: 6 x 8
##   Unit.price Quantity   Tax  cogs gross.margin.percentage gross.i~1 Rating Total
##        <dbl>    <int> <dbl> <dbl>                   <dbl>     <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.                     4.76     26.1     9.1 549.
## 2       15.3        5  3.82  76.4                    4.76      3.82    9.6  80.2
## 3       46.3        7 16.2  324.                     4.76     16.2     7.4 341.
## 4       58.2        8 23.3  466.                     4.76     23.3     8.4 489.
## 5       86.3        7 30.2  604.                     4.76     30.2     5.3 634.
## 6       85.4        7 29.9  598.                     4.76     29.9     4.1 628.
## # ... with abbreviated variable name 1: gross.income
```

```
# selecting highly correlated columns to be dropped
to_drop <- c("gross.margin.percentage")
```

```
#dropping the colunm
wrapper_num <- wrapper_num[, !names(wrapper_num) %in% to_drop]
head(wrapper_num)
```

```
## # A tibble: 6 x 7
##   Unit.price Quantity   Tax  cogs gross.income Rating Total
##        <dbl>    <int> <dbl> <dbl>        <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.         26.1     9.1 549.
## 2       15.3        5  3.82  76.4         3.82    9.6  80.2
## 3       46.3        7 16.2  324.         16.2     7.4 341.
## 4       58.2        8 23.3  466.         23.3     8.4 489.
## 5       86.3        7 30.2  604.         30.2     5.3 634.
## 6       85.4        7 29.9  598.         29.9     4.1 628.
```

```
# normalizing our dataset by use of scale function.
# Previewing the scaled dataset
wrapper_norm <- as.data.frame(scale(wrapper_num))
head(wrapper_norm)
```

```
##    Unit.price   Quantity         Tax        cogs gross.income     Rating
## 1  0.71780097  0.5096752  0.91914693  0.91914693   0.91914693  1.2378240
## 2 -1.52454035 -0.1744526 -0.98723557 -0.98723557  -0.98723557  1.5287619
## 3 -0.35260468  0.5096752  0.07141032  0.07141032   0.07141032  0.2486355
## 4  0.09616553  0.8517391  0.67544187  0.67544187   0.67544187  0.8305111
## 5  1.15638044  0.5096752  1.26649176  1.26649176   1.26649176 -0.9733034
## 6  1.12165642  0.5096752  1.23899114  1.23899114   1.23899114 -1.6715541
##         Total
## 1  0.91914693
## 2 -0.98723557
## 3  0.07141032
## 4  0.67544187
## 5  1.26649176
## 6  1.23899114
```

```r
#Selecting the best features
out = clustvarsel(wrapper_norm, G = 1:7)
out
```

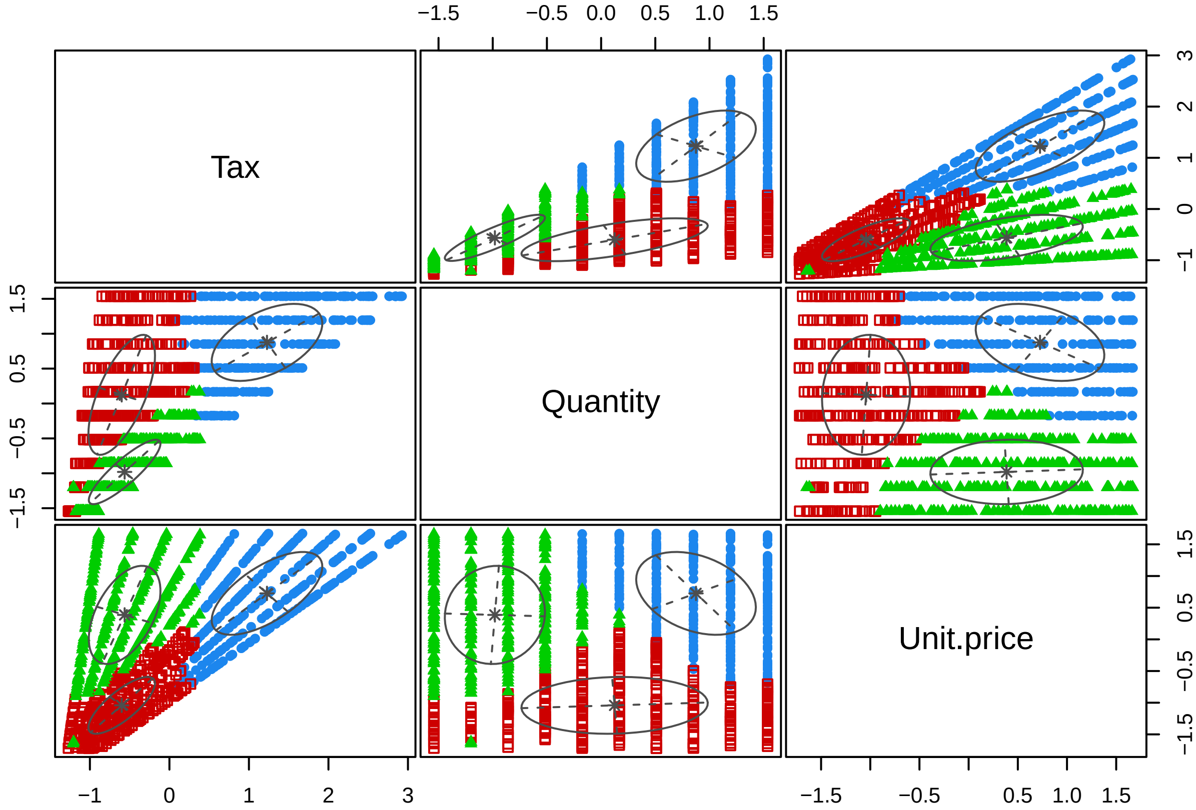
```
## ------------------------------------------------------
## Variable selection for Gaussian model-based clustering
## Stepwise (forward/backward) greedy search
## ------------------------------------------------------
##
##  Variable proposed Type of step  BICclust Model G   BICdiff Decision
##                Tax          Add -2460.877     V 4  389.8147 Accepted
##           Quantity          Add -3640.069   VEV 7  989.7613 Accepted
##         Unit.price          Add -1510.703   EVV 7 3474.0832 Accepted
##         Unit.price       Remove -3640.069   VEV 7 3474.0832 Rejected
##             Rating          Add -4599.859   EVV 7 -238.4641 Rejected
##         Unit.price       Remove -3640.069   VEV 7 3474.0832 Rejected
##
## Selected subset: Tax, Quantity, Unit.price
```

```r
# The variable selected include Tax, Quantity, and unit price.
```

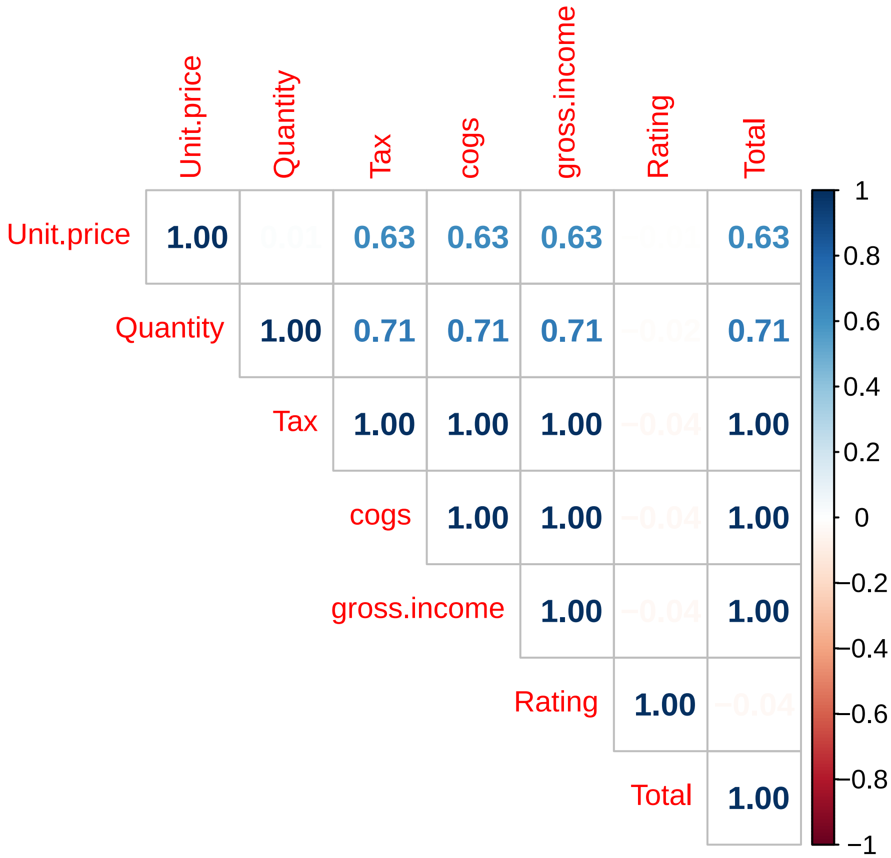
```r
# building the clustering model:
Subset1 = wrapper_norm[out$subset]
mod = Mclust(Subset1, G = 1:3)
summary(mod)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EVV (ellipsoidal, equal volume) model with 3 components:
##
##  log-likelihood    n df       BIC      ICL
##       -1846.246 1000 27 -3879.002 -3986.31
##
## Clustering table:
##   1   2   3
## 319 357 324
```

```r
plot(mod,c("classification"))
```




```r
# A corrplot to show the features
corrplot(cor(wrapper_norm), type = 'upper', method = 'number', tl.cex = 0.9)
```



Unit price determines how an item is popular by 63% while Quantity by 71% This can be uselful in identifying the goods to stock up by the supermarket. These two features have also ranked top during our feature selection therefore great determinants.