Carrefour Supermarket Sales Dataset

Dimensionality reduction and Feature selection

Martha Irungu

9/17/2020

Ia) Specifying the question

The objective of this study is to explore a recent marketing dataset from Carrefour Kenya to provide insights and recommendations that will inform their marketing strategy with an aim increasing their turnover.

b)Defining the Metrics for success

To meet the objective of the study we will need to do the following:

i) Implement unsupervised learning techniques to unearth insights emerging from the dataset provided

ii) Make conclusions and recommendations that will inform the marketing strategy of Carrefour supermarket with an aim on increasing their turnover

c) Understanding the context

Being an new entrant in the Kenyan market, it is in the interest of Carrefour to sharpen their marketing strategy hence increase sales of their products. As a Data analyst at Carrefour Kenya they 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).

The retail business has had alot of challenges in the recent past, with the likes of Nakumatt supermarket collapsing. This therefor calls the supermarkets taht are in the space to be subtle in their business model to ensure they keep customers and remain in business into the future. This analysis will support carrefour with insights that they can tap into to implement a working marketing strategy.

d) Recording the experimental design

The following steps were implemented

1.) Business Understanding.

2.) Reading the data.

3.) Data Exploration and cleaning to prepare the data for analysis

4.) Perform dimesionality reduction using PCA

5.)Implement feature selection methodologies

6.) Conclusion of the findings and recommendation.

e)Data Relevance

The data provided for this study consists of details of products, branches,customer type,unit price, quantity among other varibles that can help one understand the products the supermarket sells, the customers targeted and the prices at which the products retail at. This dataset is relevant for the study.

2)Previewing and reading the data

```
library("data.table")
sales<-fread("/Users/marthairungu/desktop/supermarket_dataset.csv")
head(sales)

##     Invoice ID Branch Customer type Gender           Product line Unit pri
ce
## 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 percen
tage
## 1:        7 26.1415  1/5/2019 13:08     Ewallet 522.83                4.76
1905
## 2:        5  3.8200  3/8/2019 10:29        Cash  76.40                4.76
1905
## 3:        7 16.2155  3/3/2019 13:23 Credit card 324.31                4.76
1905
## 4:        8 23.2880 1/27/2019 20:33     Ewallet 465.76                4.76
1905
## 5:        7 30.2085  2/8/2019 10:37     Ewallet 604.17                4.76
1905
## 6:        7 29.8865 3/25/2019 18:30     Ewallet 597.73                4.76
1905
```

```
##    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
```

#Checking the dimension of the dataset

```
dim(sales)

## [1] 1000   16
```

#The dataset has 1,000 observations and 16 variables

#Checking the structure of the dataset

```
str(sales)

## 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 accessori
es" "Home and lifestyle" "Health and beauty" ...
##  $ 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" "1/27/20
19" ...
##  $ 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 variables have datatypes in character and interger datatypes. We will convert the varibles as appropropriate as we analyse the data.

#Checking the summary of the dataset

```
summary(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.   :  10.6
8
##  1st Qu.:4.762           1st Qu.: 5.9249   1st Qu.: 5.500   1st Qu.: 124.4
2
##  Median :4.762           Median :12.0880   Median : 7.000   Median : 253.8
5
##  Mean   :4.762           Mean   :15.3794   Mean   : 6.973   Mean   : 322.9
7
##  3rd Qu.:4.762           3rd Qu.:22.4453   3rd Qu.: 8.500   3rd Qu.: 471.3
5
##  Max.   :4.762           Max.   :49.6500   Max.   :10.000   Max.   :1042.6
5
```

#Summary for the numerica variables is as tabulated

3)Data Cleaning

#Getting column names

```
colnames(sales)
```

```
##  [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"
```

#For ease of working with the data, we will change column names and convert to lower case

```r
names(sales)[1]<- 'invoice_id'
names(sales)[2]<- 'branch'
names(sales)[3]<-'customer'
names(sales)[4]<-'gender'
names(sales)[5]<-'product'
names(sales)[6]<-'unit_price'
names(sales)[7]<-'quantity'
names(sales)[8]<-'tax'
names(sales)[9]<-'date'
names(sales)[10]<-'time'
names(sales)[11]<-'payment'
names(sales)[12]<-'cogs'
names(sales)[13]<-'margin_percent'
names(sales)[14]<-'gross_income'
names(sales)[15]<-'rating'
names(sales)[16]<-'total'


#Confirming the variable names have been changed
colnames(sales)
```

```
##  [1] "invoice_id"     "branch"         "customer"       "gender"
##  [5] "product"        "unit_price"     "quantity"       "tax"
##  [9] "date"           "time"           "payment"        "cogs"
## [13] "margin_percent" "gross_income"   "rating"         "total"
```

#Description of the variables

#Invoice ID-Invoice identification number.

#Branch-We have 3 branches A,B and C.

#Customer type-We have 2 types of customer Member and Normal.

#Gender-We have Male and female.

#Product line-We have 6 levels of product line

#Unit price-price per unit #Quantity-quantity sold

#Tax-tax charged #Date- Date of transaction

#Time-Time of transaction #Payment-Amount pais for the product

#cogs
#gross margin percentage-gross margin in percentage

#gross income-gross income

#Rating-rating of the product

#Total -total amount

#Checking for missing values

```
colSums(is.na(sales))

##     invoice_id         branch       customer         gender        product
##              0              0              0              0              0
##     unit_price       quantity            tax           date           time
##              0              0              0              0              0
##        payment           cogs margin_percent   gross_income         rating
##              0              0              0              0              0
##          total
##              0
```

#We note that our dataset has no missing values.

#Checking for duplicates

```
duplicated_rows <- sales[duplicated(sales),]
duplicated_rows

## Empty data.table (0 rows and 16 cols): invoice_id,branch,customer,gender,p
roduct,unit_price...
```

#We note that our dataset has no duplicates

#splitting date to day, month and year and time to hours and minute

```
sales$day <- format(as.POSIXct(sales$date,format="%m/%d/%Y"),"%d")
sales$month <-format(as.POSIXct(sales$date,format="%m/%d/%Y"),"%m")
sales$year <- format(as.POSIXct(sales$date, format="%m/%d/%Y"), "%Y")
sales$hour <- format(as.POSIXct(sales$time, format="%H:%M"), "%H")
sales$minute <-format(as.POSIXct(sales$time, format="%H:%M"), "%M")
str(sales)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  21 variables:
##  $ invoice_id    : chr  "750-67-8428" "226-31-3081" "631-41-3108" "123-19-1176" ...
##  $ branch        : chr  "A" "C" "A" "A" ...
##  $ customer      : chr  "Member" "Normal" "Normal" "Member" ...
##  $ gender        : chr  "Female" "Female" "Male" "Male" ...
##  $ product       : chr  "Health and beauty" "Electronic accessories" "Home and lifestyle" "Health and beauty" ...
##  $ 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 ...
##  $ margin_percent: 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 ...
##  $ day           : chr  "05" "08" "03" "27" ...
##  $ month         : chr  "01" "03" "03" "01" ...
##  $ year          : chr  "2019" "2019" "2019" "2019" ...
##  $ hour          : chr  "13" "10" "13" "20" ...
##  $ minute        : chr  "08" "29" "23" "33" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#changing the data types of columns to appropriate datatypes and dropping rendadant columns

```r
sales$invoice_id<-NULL #dropping the invoice id as we will note need it
sales$date<-NULL
sales$time<-NULL
sales$branch<-as.factor(sales$branch)
sales$customer<-as.factor(sales$customer)
sales$gender<-as.factor(sales$gender)
sales$product<-as.factor(sales$product)
sales$payment<-as.factor(sales$payment)
sales$year<-as.factor(sales$year)
sales$month<-as.factor(sales$month)
sales$day<-as.factor(sales$day)
sales$hour<-as.factor(sales$hour)
sales$minute<-as.factor(sales$minute)

str(sales)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  18 variables:
##  $ branch        : Factor w/ 3 levels "A","B","C": 1 3 1 1 1 3 1 3 1 2 ...
##  $ customer      : Factor w/ 2 levels "Member","Normal": 1 2 2 1 2 2 1 2 1
```

```
1 ...
##  $ gender        : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 1 1 1
...
##  $ product       : Factor w/ 6 levels "Electronic accessories",..: 4 1 5 4
6 1 1 5 4 3 ...
##  $ 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 ...
##  $ payment       : Factor w/ 3 levels "Cash","Credit card",..: 3 1 2 3 3 3
3 3 2 2 ...
##  $ cogs          : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ margin_percent: 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 ...
##  $ day           : Factor w/ 31 levels "01","02","03",..: 5 8 3 27 8 25 25
24 10 20 ...
##  $ month         : Factor w/ 3 levels "01","02","03": 1 3 3 1 2 3 2 2 1 2
...
##  $ year          : Factor w/ 1 level "2019": 1 1 1 1 1 1 1 1 1 1 ...
##  $ hour          : Factor w/ 11 levels "10","11","12",..: 4 1 4 11 1 9 5 2
8 4 ...
##  $ minute        : Factor w/ 60 levels "00","01","02",..: 9 30 24 34 38 31
37 39 16 28 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#We note branch has 3levels(3branches),customer 2levels(member and normal),gender has 2(male and female),product has 6 levels,payment has 3 levels(cash,credit card and),day has 31 days, month has 3, hour has 11levels and minute 60.

#For us to be able to apply PCA we need to Change all the variables to numeric

```
sales$branch<-as.numeric(sales$branch)
sales$customer<-as.numeric(sales$customer)
sales$gender<-as.numeric(sales$gender)
sales$product<-as.numeric(sales$product)
sales$payment<-as.numeric(sales$payment)
sales$year<-as.numeric(sales$year)
sales$month<-as.numeric(sales$month)
sales$day<-as.numeric(sales$day)
sales$hour<-as.numeric(sales$hour)
sales$minute<-as.numeric(sales$minute)
str(sales)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  18 variables:
##  $ branch        : num  1 3 1 1 1 3 1 3 1 2 ...
##  $ customer      : num  1 2 2 1 2 2 1 2 1 1 ...
##  $ gender        : num  1 1 2 2 2 2 1 1 1 1 ...
##  $ product       : num  4 1 5 4 6 1 1 5 4 3 ...
```

```
##  $ 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 ...
##  $ payment       : num  3 1 2 3 3 3 3 3 2 2 ...
##  $ cogs          : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ margin_percent: 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 ...
##  $ day           : num  5 8 3 27 8 25 25 24 10 20 ...
##  $ month         : num  1 3 3 1 2 3 2 2 1 2 ...
##  $ year          : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ hour          : num  4 1 4 11 1 9 5 2 8 4 ...
##  $ minute        : num  9 30 24 34 38 31 37 39 16 28 ...
##  - attr(*, ".internal.selfref")=<externalptr>

sales$year <-NULL
sales$margin_percent<-NULL#since this is a percentage and we have gross incom
e column
```

#Checking for missing values

```
colSums(is.na(sales))

##       branch      customer        gender       product    unit_price      quant
ity
##            0             0             0             0             0             
0
##          tax       payment          cogs  gross_income        rating          to
tal
##            0             0             0             0             0             
0
##          day         month          hour        minute
##            0             0             0             0
```

#We have no missing values

#Checking for duplicates

```
duplicated_rows <- sales[duplicated(sales),]
duplicated_rows

## Empty data.table (0 rows and 16 cols): branch,customer,gender,product,unit
_price,quantity...
```

#We have no duplicates

4)Implementing PCA

#We then pass the sales dataset to the prcomp(). We also set two arguments, center and scale to be TRUE then preview our object with summary

```
sales.pca <- prcomp(sales, center = TRUE, scale. = TRUE)
summary(sales.pca)

## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     P
C7
## Standard deviation     2.2205 1.0874 1.08282 1.05002 1.02123 1.01763 0.990
88
## Proportion of Variance 0.3081 0.0739 0.07328 0.06891 0.06518 0.06472 0.061
36
## Cumulative Proportion  0.3081 0.3821 0.45533 0.52424 0.58942 0.65414 0.715
51
##                           PC8    PC9    PC10    PC11    PC12   PC13      P
C14
## Standard deviation     0.9757 0.9641 0.95863 0.92025 0.90270 0.2994 3.027e
-16
## Proportion of Variance 0.0595 0.0581 0.05744 0.05293 0.05093 0.0056 0.000e
+00
## Cumulative Proportion  0.7750 0.8331 0.89054 0.94347 0.99440 1.0000 1.000e
+00
##                            PC15      PC16
## Standard deviation     1.404e-16 7.688e-17
## Proportion of Variance 0.000e+00 0.000e+00
## Cumulative Proportion  1.000e+00 1.000e+00
```

#As a result we obtain 16 principal components, # each which explain a percentate of the total variation of the dataset # PC1 explains 30.8% of the total variance, which means one-thirds # of the information in the dataset (16 variables) can be encapsulated # by just that one Principal Component. PC2 explains 7.4% of the variance. Etc

```
# Calling str() to have a look at your PCA object
# ---
#
str(sales.pca)

## List of 5
##  $ sdev    : num [1:16] 2.22 1.09 1.08 1.05 1.02 ...
##  $ rotation: num [1:16, 1:16] 0.0224 -0.0125 -0.0283 0.0174 0.2911 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:16] "branch" "customer" "gender" "product" ...
##   .. ..$ : chr [1:16] "PC1" "PC2" "PC3" "PC4" ...
##  $ center  : Named num [1:16] 1.99 1.5 1.5 3.45 55.67 ...
##   ..- attr(*, "names")= chr [1:16] "branch" "customer" "gender" "product"
...
```

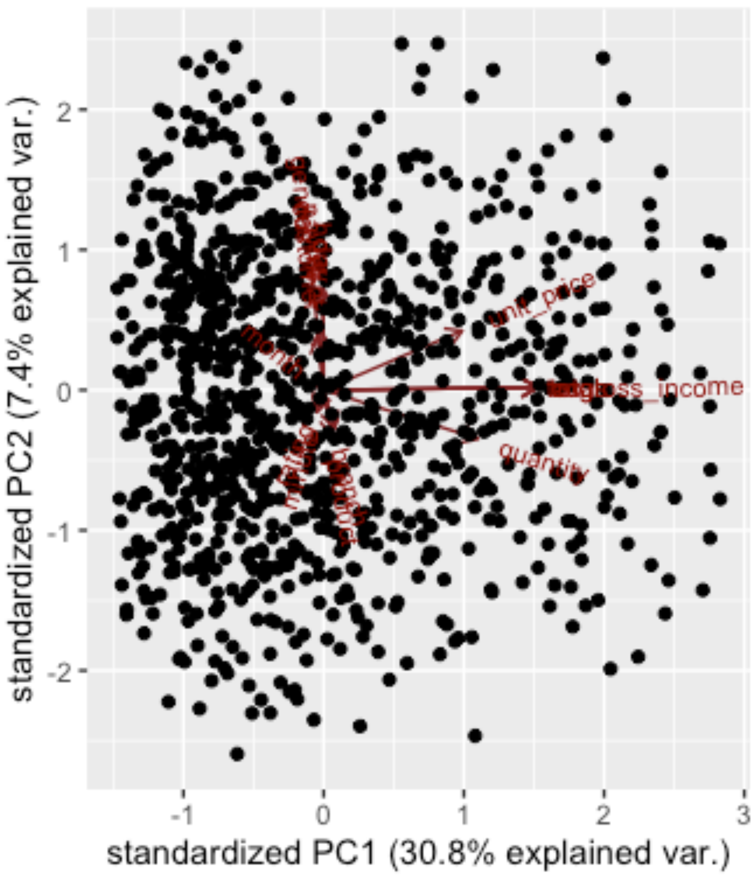
```
##  $ scale   : Named num [1:16] 0.818 0.5 0.5 1.715 26.495 ...
##   ..- attr(*, "names")= chr [1:16] "branch" "customer" "gender" "product"
...
##  $ x       : num [1:1000, 1:16] 2.05 -2.287 0.126 1.466 2.743 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:16] "PC1" "PC2" "PC3" "PC4" ...
##  - attr(*, "class")= chr "prcomp"
```

#Here we note that our pca object: The center point ($center$), $scaling$(scale), #standard deviation(sdev) of each principal component. #The relationship (correlation or anticorrelation, etc) #between the initial variables and the principal components ($rotation). #The values of each sample in terms of the principal components ($x)

#We will now plot our pca. This will provide us with some very useful insights i.e. #which variables determine customers purchase

#Installing our ggbiplot visualisation package

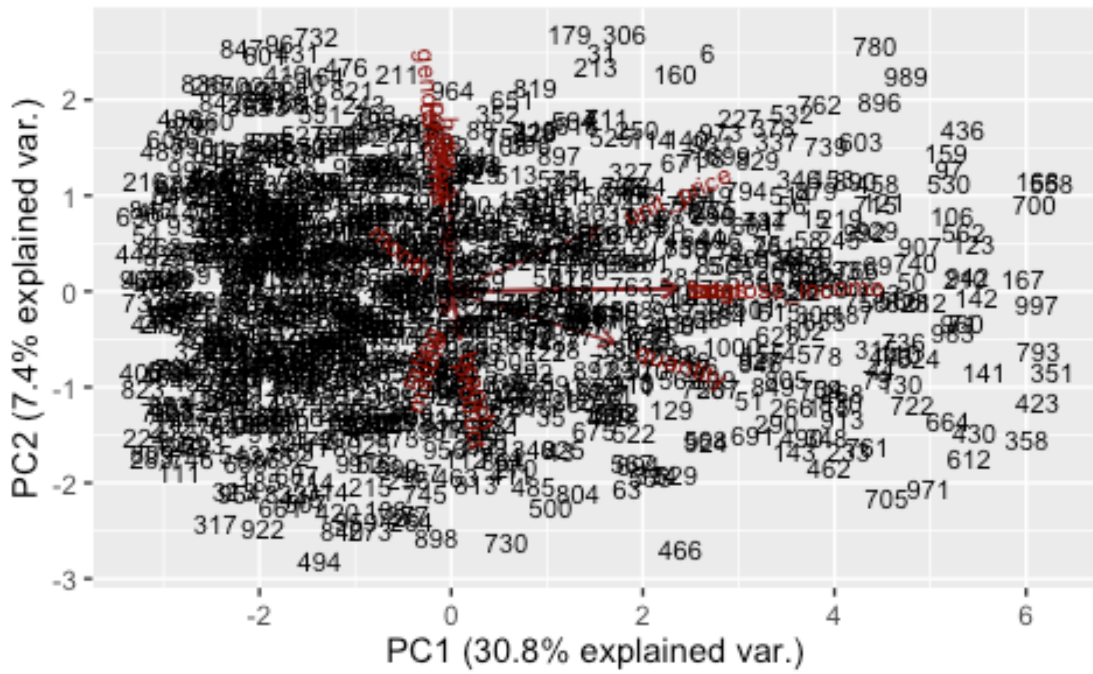
```
library(devtools)

## Loading required package: usethis

# Then Loading our ggbiplot library
#
library(ggbiplot)

## Loading required package: ggplot2

## Loading required package: plyr

## Loading required package: scales

## Loading required package: grid

ggbiplot(sales.pca)
```



#From the graph we will see that the variables quantity, gross income and unit price contribute to PC1, #with higher values in those variables moving the samples to the right on the plot

#Adding more detail to the plot, we provide arguments rownames as labels

```
ggbiplot(sales.pca, labels=rownames(sales), obs.scale = 1, var.scale = 1)
```



5)Implementing Feature Selection in Unsupervised Learning

#1.Filter Method

#We will use the findCorrelation function included in the caret package to create a subset of variables. #This function would allows us to remove redundancy by correlation using the dataset. #It would search through a correlation matrix and return a vector of integers corresponding to the columns, #hence allowing us to remove or reduce/filter pair-wise correlations.

#Checking the structure of our dataset

```
str(sales)

## Classes 'data.table' and 'data.frame':   1000 obs. of  16 variables:
##  $ branch      : num  1 3 1 1 1 3 1 3 1 2 ...
##  $ customer    : num  1 2 2 1 2 2 1 2 1 1 ...
##  $ gender      : num  1 1 2 2 2 2 1 1 1 1 ...
##  $ product     : num  4 1 5 4 6 1 1 5 4 3 ...
##  $ 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 ...
```

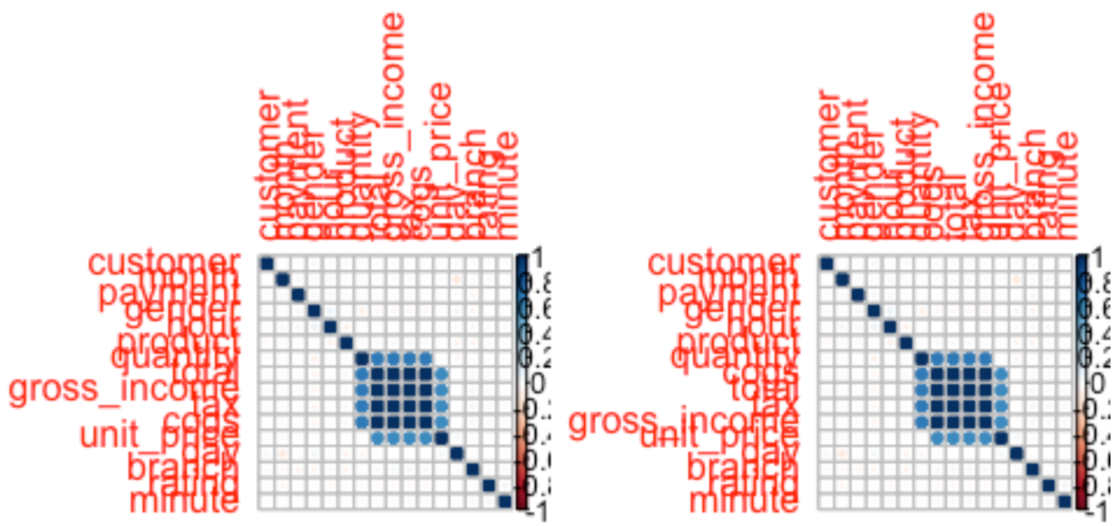
```
##  $ payment     : num  3 1 2 3 3 3 3 3 2 2 ...
##  $ cogs        : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ gross_income: num  26.14 3.82 16.22 23.29 30.21 ...
##  $ rating      : num  9.1 9.6 7.4 8.4 5.3 4.1 5.8 8 7.2 5.9 ...
##  $ total       : num  549 80.2 340.5 489 634.4 ...
##  $ day         : num  5 8 3 27 8 25 25 24 10 20 ...
##  $ month       : num  1 3 3 1 2 3 2 2 1 2 ...
##  $ hour        : num  4 1 4 11 1 9 5 2 8 4 ...
##  $ minute      : num  9 30 24 34 38 31 37 39 16 28 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

#All are numeric which id the format that we need

```
# Installing and loading our caret package

suppressWarnings(
        suppressMessages(if
                         (!require(caret, quietly=TRUE))
                install.packages("caret")))
library(caret)

# Installing and loading the corrplot package for plotting

suppressWarnings(
        suppressMessages(if
                         (!require(corrplot, quietly=TRUE))
                install.packages("corrplot")))
library(corrplot)

# Calculating the correlation matrix
# ---
#
correlationMatrix <- cor(sales)

# Find attributes that are highly correlated
# ---
#
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.75)

# Highly correlated attributes
# ---
#
highlyCorrelated

## [1]  9 12  7
```

#The columns that are highly correlated are column 7-this is tax,9-cogs and 12-total. So we will remove them.

```r
# We will remove the variables with a higher correlation
# and comparing the results graphically

# Removing Redundant Features
sales2<-sales[-highlyCorrelated]

# Performing our graphical comparison

par(mfrow = c(1,2))
corrplot(correlationMatrix, order = "hclust")
corrplot(cor(sales2), order = "hclust")
```



#2.Wrapper Methods

#We use the clustvarsel package that contains an implementation of wrapper methods. #The clustvarsel function will implement variable section methodology #for model-based clustering to find the optimal subset of variables in a dataset

```r
str(sales)

## Classes 'data.table' and 'data.frame':   1000 obs. of  16 variables:
##  $ branch      : num  1 3 1 1 1 3 1 3 1 2 ...
##  $ customer    : num  1 2 2 1 2 2 1 2 1 1 ...
```

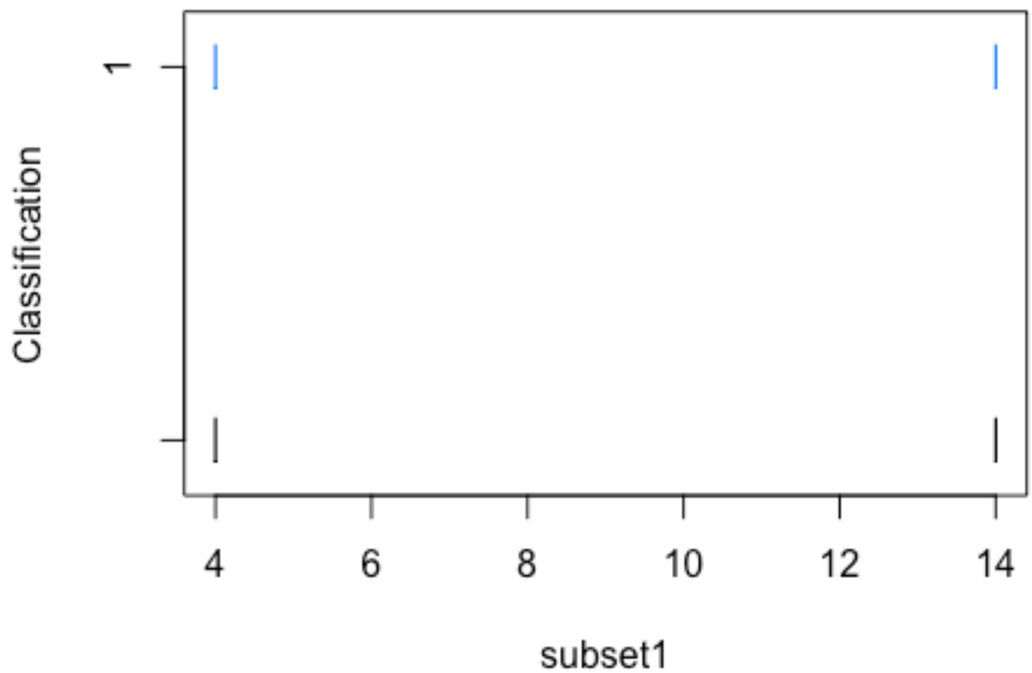
```
##  $ gender      : num  1 1 2 2 2 2 1 1 1 1 ...
##  $ product     : num  4 1 5 4 6 1 1 5 4 3 ...
##  $ unit_price  : num  74.7 15.3 46.3 58.2 86.3 ...
##  $ quantity    : int  7 5 7 8 7 7 6 10 2 3 ...
##  $ tax         : num  26.14 3.82 16.22 23.29 30.21 ...
##  $ payment     : num  3 1 2 3 3 3 3 3 2 2 ...
##  $ cogs        : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ 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 ...
##  $ day         : num  5 8 3 27 8 25 25 24 10 20 ...
##  $ month       : num  1 3 3 1 2 3 2 2 1 2 ...
##  $ hour        : num  4 1 4 11 1 9 5 2 8 4 ...
##  $ minute      : num  9 30 24 34 38 31 37 39 16 28 ...
##  - attr(*, ".internal.selfref")=<externalptr>

# Installing and loading our clustvarsel package
# ---
#
suppressWarnings(
        suppressMessages(if
                         (!require(clustvarsel, quietly=TRUE))
                install.packages("clustvarsel")))

library(clustvarsel)

# Installing and loading our mclust package
# ---
#
suppressWarnings(
        suppressMessages(if
                         (!require(mclust, quietly=TRUE))
                install.packages("mclust")))
library(mclust)

# Sequential forward greedy search (default)
# ---
#
out = clustvarsel(sales, 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
##            product          Add -3498.098     E 5  431.9005 Accepted
##              month          Add -5459.333   VEI 3  529.4172 Accepted
##            payment          Add -8092.699   VEV 4 -154.1886 Rejected
##              month       Remove -3498.098     E 5  529.4172 Rejected
```

```
##
## Selected subset: product, month
```

#From the above, we can observe that product and month have been accepted as variables and payment have been rejected.

#Building the model using the selected variables

```
# The selection algorithm would indicate that the subset
# we use for the clustering model is composed of variables X1 and X2
# and that other variables should be rejected.
# Having identified the variables that we use, we proceed to build the cluste
ring model:
# ---
#

subset1 = sales[,out$subset]
mod = Mclust(subset1, G = 1:7)
summary(mod)

## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust X (univariate normal) model with 1 component:
##
##  log-likelihood n df       BIC       ICL
##       -6.056753 2  2 -13.4998 -13.4998
##
## Clustering table:
## 1
## 2

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

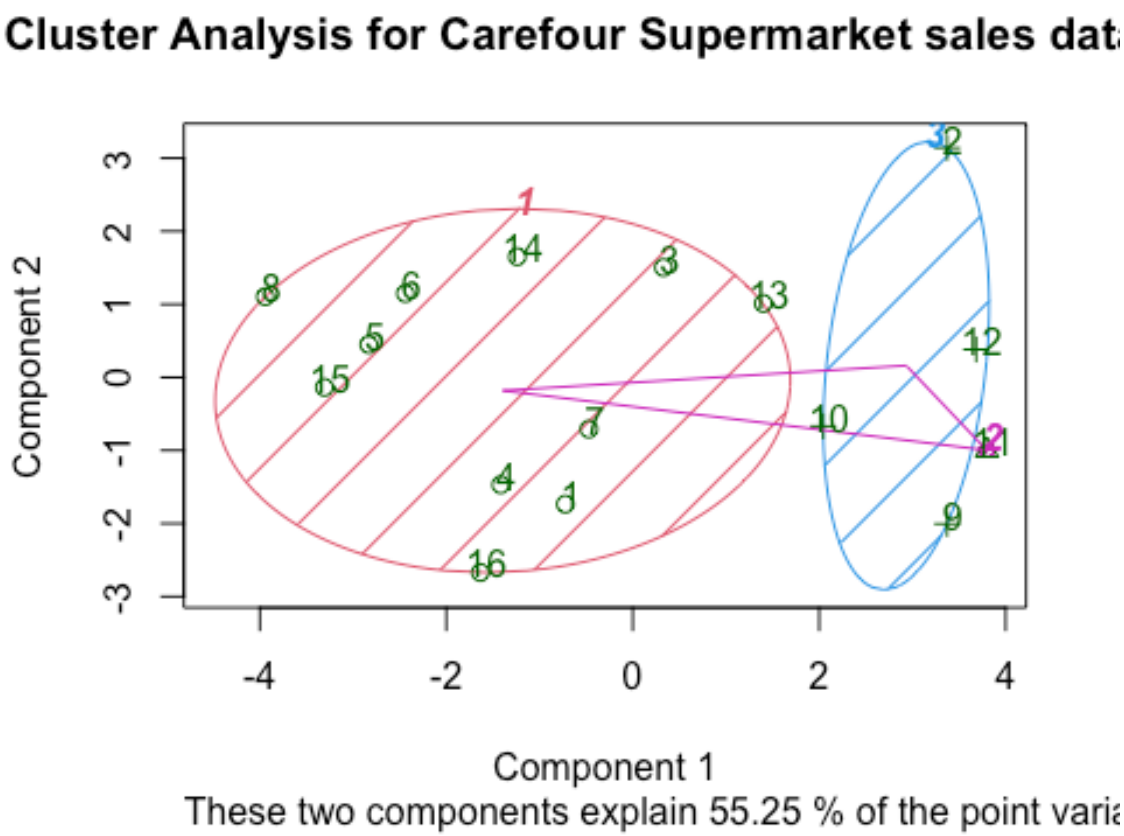


#3.Embedded Methods

#We will use the ewkm function from the wskm package. #This is a weighted subspace clustering algorithm that is well suited to very high dimensional data.

```
# We install and load our wskm package
suppressWarnings(
        suppressMessages(if
                         (!require(wskm, quietly=TRUE))
                install.packages("wskm")))
library(wskm)

set.seed(2)
model <- ewkm(sales[1:16], 3, lambda=2, maxiter=1000)

# Loading and installing our cluster package
# ---
#
suppressWarnings(
        suppressMessages(if
                         (!require(cluster, quietly=TRUE))
                install.packages("cluster")))
library("cluster")
```

```r
# Cluster Plot against 1st 2 principal components
# ---
#
clusplot(sales[1:16], model$cluster, color=TRUE, shade=TRUE,
         labels=2, lines=1,main='Cluster Analysis for Carefour Supermarket sa
les dataset')
```




```r
# Weights are calculated for each variable and cluster.
# They are a measure of the relative importance of each variable
# with regards to the membership of the observations to that cluster.
# The weights are incorporated into the distance function,
# typically reducing the distance for more important variables.
# Weights remain stored in the model and we can check them as follows:
#
round(model$weights*100,2)
```

```
##   branch customer gender product unit_price quantity  tax payment cogs
## 1   5.29    42.77  39.05    0.00       0.00     0.00 0.00    5.29 0.00
## 2   6.25     6.25   6.25    6.25       6.25     6.25 6.25    6.25 6.25
## 3  13.41    25.05  25.05    1.25       0.00     2.99 0.04   22.10 0.00
##   gross_income rating total  day month hour minute
## 1         0.00   0.00  0.00 0.00  7.60 0.00   0.00
```

```
## 2          6.25   6.25  6.25 6.25  6.25 6.25   6.25
## 3          0.04   0.86  0.00 0.00  9.21 0.00   0.00
```

#4.Feature Ranking

#We will use the FSelector Package. This is a package containing functions for selecting attributes from a given dataset

```
# We install and load the required packages

suppressWarnings(
        suppressMessages(if
                         (!require(FSelector, quietly=TRUE))
                install.packages("FSelector")))
library(FSelector)
```

#Structure of the dataset

```
str(sales)

## Classes 'data.table' and 'data.frame':   1000 obs. of  16 variables:
##  $ branch      : num  1 3 1 1 1 3 1 3 1 2 ...
##  $ customer    : num  1 2 2 1 2 2 1 2 1 1 ...
##  $ gender      : num  1 1 2 2 2 2 1 1 1 1 ...
##  $ product     : num  4 1 5 4 6 1 1 5 4 3 ...
##  $ 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 ...
##  $ payment     : num  3 1 2 3 3 3 3 3 2 2 ...
##  $ cogs        : num  522.8 76.4 324.3 465.8 604.2 ...
##  $ 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 ...
##  $ day         : num  5 8 3 27 8 25 25 24 10 20 ...
##  $ month       : num  1 3 3 1 2 3 2 2 1 2 ...
##  $ hour        : num  4 1 4 11 1 9 5 2 8 4 ...
##  $ minute      : num  9 30 24 34 38 31 37 39 16 28 ...
##  - attr(*, ".internal.selfref")=<externalptr>

# From the FSelector package, we use the correlation coefficient as a unit of
valuation.
# This would be one of the several algorithms contained in the FSelector pack
age that can
#be used rank the variables.

scores <- linear.correlation(sales)
scores
```

```
##                attr_importance
## customer           0.01960787
## gender             0.05631756
## product            0.05393756
## unit_price         0.02820244
## quantity           0.01596379
## tax                0.04104666
## payment            0.05010429
## cogs               0.04104666
## gross_income       0.04104666
## rating             0.01023848
## total              0.04104666
## day                0.01308653
## month              0.03530092
## hour               0.03300711
## minute             0.03837833
```

```
# From the output above, we observe a list containing
# rows of variables on the left and score on the right.
# In order to make a decision, we define a cutoff
# We check top 5 representative variables,
# through the use of the cutoff.k function included in the FSelector package.
#
# cutoff.k: The algorithms select a subset from a ranked attributes.
# ---
#
subset <- cutoff.k(scores, 5)
as.data.frame(subset)
```

```
##    subset
## 1  gender
## 2 product
## 3 payment
## 4     tax
## 5    cogs
```

#Gender, product and payment are the top 3 varibles selected

```
# We could also set cutoff as a percentage which would indicate
# that we would want to work with the percentage of the best variables.

subset2 <-cutoff.k.percent(scores, 0.4)
as.data.frame(subset2)
```

```
##        subset2
## 1       gender
## 2      product
## 3      payment
## 4          tax
```

```
## 5        cogs
## 6 gross_income
```

6)conclusions and recommendations

Based on the above analysis, we conclude the following:

1. When we perfomed PCA, we established that one variable represent 30% of the rest of the 16 variables.Out of this, quantity,gross income and unit price were 3 key variables that contributed. This therefore need to be taken with weight as they can already provide 30% of the information we are looking for.

2. In regard to the features, we established that product and month are 2 key variables that are were accepted.

3. Variable rankings identified gender, payment and products as key variables

Based on this Carrefour should be keen with specific products targeting customers based on gender, their unit price, quantities they are selling and gross income for each.