Dimensionality reduction

Whiterose

2022-06-13

Exploratory data analysis

Define the question

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).

defining the metric for success

explaining the context

Carre-four is a French multinational retail corporation headquartered in Massy, France. The eighth-largest retailer in the world by revenue, it operates a chain of hypermarkets, groceries stores and convenience stores, which as of January 2021, comprises its 12,225 stores in over 30 countries. Kenya been one of them a statistical analysis is needed to improve sales in this country.

experimental design

1.Problem Definition 2.Data Sourcing 3.Check the Data 4.Perform Data Cleaning 5.Perform Exploratory Data Analysis (Univariate, Bivariate & Multivariate) 6.Implement the Solution 7.Challenge the Solution 8.Follow up Questions

data source validation

loading packages

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stats)
library(readr)
library(rmarkdown)
library(tidyr)
library(tibble)
library(caret)
```

```
## Loading required package: lattice
```

```
library(solitude)
```

loading dataset

```
df <- read.table("Supermarket_Dataset_1 - Sales Data.csv",header=TRUE, sep=",", row.names=NULL)
```

viewing dataset

```
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" "
##  $ 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 ...
```

let us take a look at our data-set.

```
glimpse(df)
```

```
## Rows: 1,000
## Columns: 16
```

```
## $ Invoice.ID              <chr> "750-67-8428", "226-31-3081", "631-41-3108", "~
## $ Branch                  <chr> "A", "C", "A", "A", "A", "C", "A", "C", "A", "~
## $ Customer.type           <chr> "Member", "Normal", "Normal", "Member", "Norma~
## $ Gender                  <chr> "Female", "Female", "Male", "Male", "Male", "M~
## $ Product.line            <chr> "Health and beauty", "Electronic accessories",~
## $ Unit.price              <dbl> 74.69, 15.28, 46.33, 58.22, 86.31, 85.39, 68.8~
## $ Quantity                <int> 7, 5, 7, 8, 7, 7, 6, 10, 2, 3, 4, 4, 5, 10, 10~
## $ Tax                     <dbl> 26.1415, 3.8200, 16.2155, 23.2880, 30.2085, 29~
## $ Date                    <chr> "1/5/2019", "3/8/2019", "3/3/2019", "1/27/2019~
## $ Time                    <chr> "13:08", "10:29", "13:23", "20:33", "10:37", "~
## $ Payment                 <chr> "Ewallet", "Cash", "Credit card", "Ewallet", "~
## $ cogs                    <dbl> 522.83, 76.40, 324.31, 465.76, 604.17, 597.73,~
## $ gross.margin.percentage <dbl> 4.761905, 4.761905, 4.761905, 4.761905, 4.7619~
## $ gross.income            <dbl> 26.1415, 3.8200, 16.2155, 23.2880, 30.2085, 29~
## $ Rating                  <dbl> 9.1, 9.6, 7.4, 8.4, 5.3, 4.1, 5.8, 8.0, 7.2, 5~
## $ Total                   <dbl> 548.9715, 80.2200, 340.5255, 489.0480, 634.378~
```

Data cleaning

missing values

```
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
```

```
head(df[complete.cases(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
```

Seems like there are no missing values.

renaming columns

```
df <- df %>%
  rename(invoice_id = "Invoice.ID", branch = Branch, customer_type = `Customer.type`, gender = Gender, p

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
```

dropping invalid columns

```
df <- subset(df, select = -c(invoice_id))
head(df)
```

```
##   branch customer_type gender           product_line unit_price quantity
## 1      A        Member Female      Health and beauty      74.69        7
## 2      C        Normal Female Electronic accessories      15.28        5
```

```
## 3      A        Normal   Male     Home and lifestyle      46.33        7
## 4      A        Member   Male      Health and beauty      58.22        8
## 5      A        Normal   Male      Sports and travel      86.31        7
## 6      C        Normal   Male Electronic accessories      85.39        7
##       tax      date  time     payment   cogs gross_margin_percentage
## 1 26.1415  1/5/2019 13:08     Ewallet 522.83                4.761905
## 2  3.8200  3/8/2019 10:29        Cash  76.40                4.761905
## 3 16.2155  3/3/2019 13:23 Credit card 324.31                4.761905
## 4 23.2880 1/27/2019 20:33     Ewallet 465.76                4.761905
## 5 30.2085  2/8/2019 10:37     Ewallet 604.17                4.761905
## 6 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
```

duplicates

```
#distinct(df, .keep_all= TRUE)
#unique(df)

#df <- df[!(duplicated(df) | duplicated(df, fromLast = TRUE)), ]

#df[!duplicated(df[c('date')]), ]

#df %>%
 # distinct(invoice_id, .keep_all = TRUE)
df.un <- df[!duplicated(df), ]

#df <- df[-c(90:1000), ]
```

numeric columns

```
'df <- subset(df, select = -c("invoice_id", "branch", "customer_type", "gender", "product_line", "date"
```

```
## [1] "df <- subset(df, select = -c(\"invoice_id\", \"branch\", \"customer_type\", \"gender\", \"produ
```

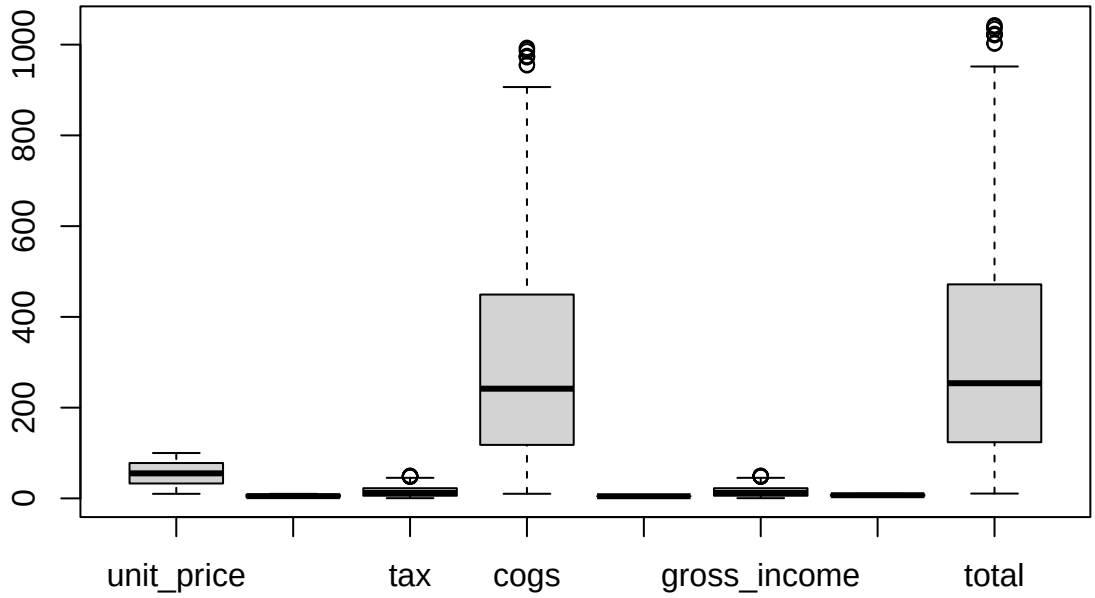
```
df1 <- sapply(df, is.numeric)
df1
```

```
##                   branch            customer_type                   gender
##                    FALSE                    FALSE                    FALSE
##             product_line               unit_price                 quantity
##                    FALSE                     TRUE                     TRUE
##                      tax                     date                     time
##                     TRUE                    FALSE                    FALSE
```

```
##                  payment                     cogs gross_margin_percentage
##                    FALSE                     TRUE                     TRUE
##             gross_income                   rating                    total
##                     TRUE                     TRUE                     TRUE
```

checking for outliers

```
df_out <- df %>%
  select(unit_price, quantity, tax, cogs, gross_margin_percentage, gross_income, rating, total)
boxplot(df_out)
```



There seems to be a few outliers in the numeric variables.

```
#remove the dependent and identifier variables
df <- subset(df, select = -c(tax, total))
```

lets check the available variables

```
colnames(df)
```

```
##  [1] "branch"                "customer_type"
##  [3] "gender"                "product_line"
##  [5] "unit_price"            "quantity"
##  [7] "date"                  "time"
```

```
##  [9] "payment"                 "cogs"
## [11] "gross_margin_percentage" "gross_income"
## [13] "rating"
```

```
length(df)
```

```
## [1] 13
```

```
rownames<-(value = rownames(df))
```

coverting categorical data to numeric

```
#load library
#install.packages("dummies", repos = "https://rdocumentation.org/packages/dummies/versions/1.5.6")
library(dummies)
```

```
## dummies-1.5.6 provided by Decision Patterns
```

```
#create a dummy data frame
df2 <- dummy.data.frame(df,names=c("branch","customer_type","gender","product_line","unit_price","quant
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
#str(new_df)
str(head(df2))
```

```
## 'data.frame':    6 obs. of  3605 variables:
##  $ branchA                           : int  1 0 1 1 1 0
##  $ branchB                           : int  0 0 0 0 0 0
##  $ branchC                           : int  0 1 0 0 0 1
##  $ customer_typeMember               : int  1 0 0 1 0 0
##  $ customer_typeNormal               : int  0 1 1 0 1 1
##  $ genderFemale                      : int  1 1 0 0 0 0
##  $ genderMale                        : int  0 0 1 1 1 1
##  $ product_lineElectronic accessories: int  0 1 0 0 0 1
##  $ product_lineFashion accessories   : int  0 0 0 0 0 0
##  $ product_lineFood and beverages    : int  0 0 0 0 0 0
##  $ product_lineHealth and beauty     : int  1 0 0 1 0 0
##  $ product_lineHome and lifestyle    : int  0 0 1 0 0 0
##  $ product_lineSports and travel     : int  0 0 0 0 1 0
##  $ unit_price10.08                   : int  0 0 0 0 0 0
##  $ unit_price10.13                   : int  0 0 0 0 0 0
##  $ unit_price10.16                   : int  0 0 0 0 0 0
##  $ unit_price10.17                   : int  0 0 0 0 0 0
##  $ unit_price10.18                   : int  0 0 0 0 0 0
##  $ unit_price10.53                   : int  0 0 0 0 0 0
##  $ unit_price10.56                   : int  0 0 0 0 0 0
##  $ unit_price10.59                   : int  0 0 0 0 0 0
##  $ unit_price10.69                   : int  0 0 0 0 0 0
##  $ unit_price10.75                   : int  0 0 0 0 0 0
##  $ unit_price10.96                   : int  0 0 0 0 0 0
##  $ unit_price10.99                   : int  0 0 0 0 0 0
##  $ unit_price11.28                   : int  0 0 0 0 0 0
##  $ unit_price11.43                   : int  0 0 0 0 0 0
##  $ unit_price11.53                   : int  0 0 0 0 0 0
##  $ unit_price11.81                   : int  0 0 0 0 0 0
##  $ unit_price11.85                   : int  0 0 0 0 0 0
##  $ unit_price11.94                   : int  0 0 0 0 0 0
##  $ unit_price12.03                   : int  0 0 0 0 0 0
##  $ unit_price12.05                   : int  0 0 0 0 0 0
##  $ unit_price12.09                   : int  0 0 0 0 0 0
##  $ unit_price12.1                    : int  0 0 0 0 0 0
##  $ unit_price12.12                   : int  0 0 0 0 0 0
##  $ unit_price12.19                   : int  0 0 0 0 0 0
##  $ unit_price12.29                   : int  0 0 0 0 0 0
##  $ unit_price12.34                   : int  0 0 0 0 0 0
##  $ unit_price12.45                   : int  0 0 0 0 0 0
##  $ unit_price12.54                   : int  0 0 0 0 0 0
##  $ unit_price12.73                   : int  0 0 0 0 0 0
##  $ unit_price12.76                   : int  0 0 0 0 0 0
```

```
##  $ unit_price12.78                : int  0 0 0 0 0 0
##  $ unit_price13.22                : int  0 0 0 0 0 0
##  $ unit_price13.5                 : int  0 0 0 0 0 0
##  $ unit_price13.59                : int  0 0 0 0 0 0
##  $ unit_price13.69                : int  0 0 0 0 0 0
##  $ unit_price13.78                : int  0 0 0 0 0 0
##  $ unit_price13.79                : int  0 0 0 0 0 0
##  $ unit_price13.85                : int  0 0 0 0 0 0
##  $ unit_price13.98                : int  0 0 0 0 0 0
##  $ unit_price14.23                : int  0 0 0 0 0 0
##  $ unit_price14.36                : int  0 0 0 0 0 0
##  $ unit_price14.39                : int  0 0 0 0 0 0
##  $ unit_price14.48                : int  0 0 0 0 0 0
##  $ unit_price14.62                : int  0 0 0 0 0 0
##  $ unit_price14.7                 : int  0 0 0 0 0 0
##  $ unit_price14.76                : int  0 0 0 0 0 0
##  $ unit_price14.82                : int  0 0 0 0 0 0
##  $ unit_price14.87                : int  0 0 0 0 0 0
##  $ unit_price14.96                : int  0 0 0 0 0 0
##  $ unit_price15.26                : int  0 0 0 0 0 0
##  $ unit_price15.28                : int  0 1 0 0 0 0
##  $ unit_price15.34                : int  0 0 0 0 0 0
##  $ unit_price15.37                : int  0 0 0 0 0 0
##  $ unit_price15.43                : int  0 0 0 0 0 0
##  $ unit_price15.49                : int  0 0 0 0 0 0
##  $ unit_price15.5                 : int  0 0 0 0 0 0
##  $ unit_price15.55                : int  0 0 0 0 0 0
##  $ unit_price15.62                : int  0 0 0 0 0 0
##  $ unit_price15.69                : int  0 0 0 0 0 0
##  $ unit_price15.8                 : int  0 0 0 0 0 0
##  $ unit_price15.81                : int  0 0 0 0 0 0
##  $ unit_price15.87                : int  0 0 0 0 0 0
##  $ unit_price15.95                : int  0 0 0 0 0 0
##  $ unit_price16.16                : int  0 0 0 0 0 0
##  $ unit_price16.28                : int  0 0 0 0 0 0
##  $ unit_price16.31                : int  0 0 0 0 0 0
##  $ unit_price16.37                : int  0 0 0 0 0 0
##  $ unit_price16.45                : int  0 0 0 0 0 0
##  $ unit_price16.48                : int  0 0 0 0 0 0
##  $ unit_price16.49                : int  0 0 0 0 0 0
##  $ unit_price16.67                : int  0 0 0 0 0 0
##  $ unit_price17.04                : int  0 0 0 0 0 0
##  $ unit_price17.14                : int  0 0 0 0 0 0
##  $ unit_price17.41                : int  0 0 0 0 0 0
##  $ unit_price17.42                : int  0 0 0 0 0 0
##  $ unit_price17.44                : int  0 0 0 0 0 0
##  $ unit_price17.48                : int  0 0 0 0 0 0
##  $ unit_price17.49                : int  0 0 0 0 0 0
##  $ unit_price17.63                : int  0 0 0 0 0 0
##  $ unit_price17.75                : int  0 0 0 0 0 0
##  $ unit_price17.77                : int  0 0 0 0 0 0
##  $ unit_price17.87                : int  0 0 0 0 0 0
##  $ unit_price17.94                : int  0 0 0 0 0 0
##  $ unit_price17.97                : int  0 0 0 0 0 0
```

```
##  $ unit_price18.08                  : int  0 0 0 0 0 0
##  $ unit_price18.11                  : int  0 0 0 0 0 0
##   [list output truncated]
##  - attr(*, "dummies")=List of 12
##   ..$ branch       : int [1:3] 1 2 3
##   ..$ customer_type: int [1:2] 4 5
##   ..$ gender       : int [1:2] 6 7
##   ..$ product_line : int [1:6] 8 9 10 11 12 13
##   ..$ unit_price   : int [1:943] 14 15 16 17 18 19 20 21 22 23 ...
##   ..$ quantity     : int [1:10] 957 958 959 960 961 962 963 964 965 966
##   ..$ date         : int [1:89] 967 968 969 970 971 972 973 974 975 976 ...
##   ..$ time         : int [1:506] 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 ...
##   ..$ payment      : int [1:3] 1562 1563 1564
##   ..$ cogs         : int [1:990] 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574 ...
##   ..$ gross_income : int [1:990] 2555 2556 2557 2558 2559 2560 2561 2562 2563 2564 ...
##   ..$ rating       : int [1:61] 3545 3546 3547 3548 3549 3550 3551 3552 3553 3554 ...
```

PCA

```
# Supply names of columns that have 0 variance
names(df[, sapply(df2, function(v) var(v, na.rm=TRUE)==0)])
```

```
## character(0)
```

```
pca_fit <- df2 %>%
  select(where(is.numeric)) %>% # retain only numeric columns
  prcomp(scale = TRUE) # do PCA on scaled data

df.pca <- prcomp(df2, center = TRUE, scale = TRUE, retx = T)
summary(head(df.pca))
```

```
##          Length  Class  Mode
## sdev        1000 -none- numeric
## rotation 3605000 -none- numeric
## center      3605 -none- numeric
## scale       3605 -none- numeric
## x        1000000 -none- numeric
```

let us look at the pca object

```
str(df.pca)
```

```
## List of 5
##  $ sdev    : num [1:1000] 2.44 2.4 2.36 2.36 2.34 ...
##  $ rotation: num [1:3605, 1:1000] -0.0212 -0.0357 0.0572 0.2427 -0.2427 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3605] "branchA" "branchB" "branchC" "customer_typeMember" ...
##   .. ..$ : chr [1:1000] "PC1" "PC2" "PC3" "PC4" ...
##  $ center  : Named num [1:3605] 0.34 0.332 0.328 0.501 0.499 0.501 0.499 0.17 0.178 0.174 ...
```

```
##   ..- attr(*, "names")= chr [1:3605] "branchA" "branchB" "branchC" "customer_typeMember" ...
##  $ scale   : Named num [1:3605] 0.474 0.471 0.47 0.5 0.5 ...
##   ..- attr(*, "names")= chr [1:3605] "branchA" "branchB" "branchC" "customer_typeMember" ...
##  $ x       : num [1:1000, 1:1000] 4.197 0.259 -2.867 -0.754 -3.153 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:1000] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:1000] "PC1" "PC2" "PC3" "PC4" ...
##  - attr(*, "class")= chr "prcomp"
```

```
names(df.pca)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

The prcomp() function results in 5 useful measures

```
#outputs the mean of variables
df.pca$center[1:20]
```

```
##                             branchA                             branchB
##                               0.340                               0.332
##                             branchC                 customer_typeMember
##                               0.328                               0.501
##                 customer_typeNormal                        genderFemale
##                               0.499                               0.501
##                          genderMale product_lineElectronic accessories
##                               0.499                               0.170
##    product_lineFashion accessories     product_lineFood and beverages
##                               0.178                               0.174
##      product_lineHealth and beauty     product_lineHome and lifestyle
##                               0.152                               0.160
##      product_lineSports and travel                    unit_price10.08
##                               0.166                               0.001
##                     unit_price10.13                    unit_price10.16
##                               0.001                               0.001
##                     unit_price10.17                    unit_price10.18
##                               0.001                               0.001
##                     unit_price10.53                    unit_price10.56
##                               0.001                               0.001
```

```
#outputs the standard deviation of variables
df.pca$scale[1:20]
```

```
##                             branchA                             branchB
##                          0.47394580                          0.47116664
##                             branchC                 customer_typeMember
##                          0.46971974                          0.50024919
##                 customer_typeNormal                        genderFemale
##                          0.50024919                          0.50024919
##                          genderMale product_lineElectronic accessories
##                          0.50024919                          0.37582076
##    product_lineFashion accessories     product_lineFood and beverages
##                          0.38270414                          0.37929918
```

```
##      product_lineHealth and beauty     product_lineHome and lifestyle
##                          0.35920054                          0.36678950
##      product_lineSports and travel                   unit_price10.08
##                          0.37226682                          0.03162278
##                    unit_price10.13                   unit_price10.16
##                          0.03162278                          0.03162278
##                    unit_price10.17                   unit_price10.18
##                          0.03162278                          0.03162278
##                    unit_price10.53                   unit_price10.56
##                          0.03162278                          0.03162278
```

The most important measure is the rotation

Let's look at first 4 principal components and first 5 rows.

```
df.pca$rotation[1:5,1:4]
```

```
##                              PC1         PC2         PC3         PC4
## branchA              -0.02124282  0.02717789  0.08250054  0.24856205
## branchB              -0.03568357 -0.08901340 -0.15440155 -0.21897264
## branchC               0.05722742  0.06186518  0.07163437 -0.03115121
## customer_typeMember   0.24272272 -0.27416284  0.08712535  0.02599957
## customer_typeNormal  -0.24272272  0.27416284 -0.08712535 -0.02599957
```

dimension of matrix

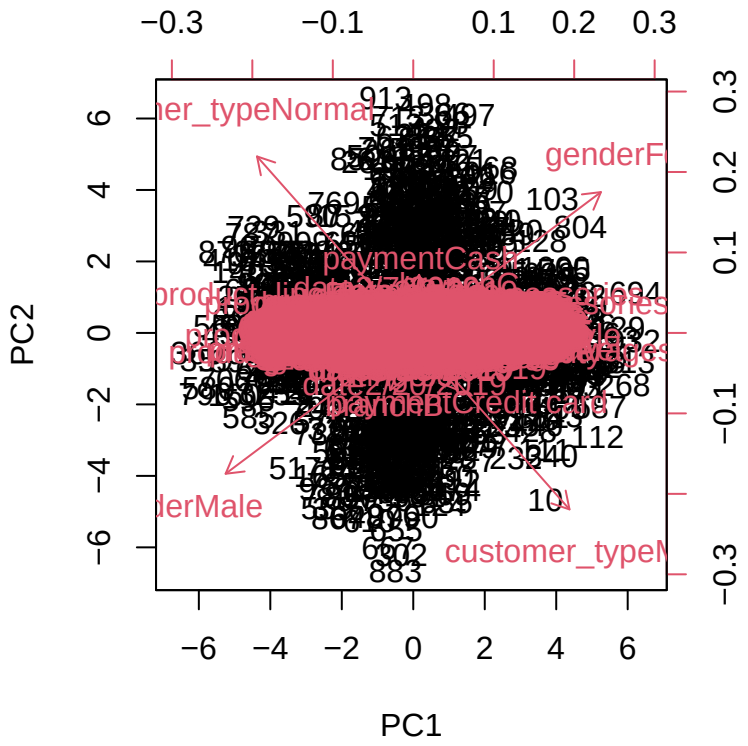
```
dim(df.pca$x)
```

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

plotting pca

resultant principal components

```
biplot(df.pca, scale = 0)
```



To make inference from image above, let us focus on the extreme ends (top, bottom, left, right) of this graph. The customer type, gender, product line stand out.

```
#compute standard deviation of each principal component
std_dev <- df.pca$sdev

#compute variance
pr_var <- std_dev^2

#check variance of first 10 components
pr_var[1:10]
```

```
##  [1] 5.959519 5.743858 5.577648 5.546084 5.457803 5.444021 5.382550 5.336938
##  [9] 5.313586 5.280407
```

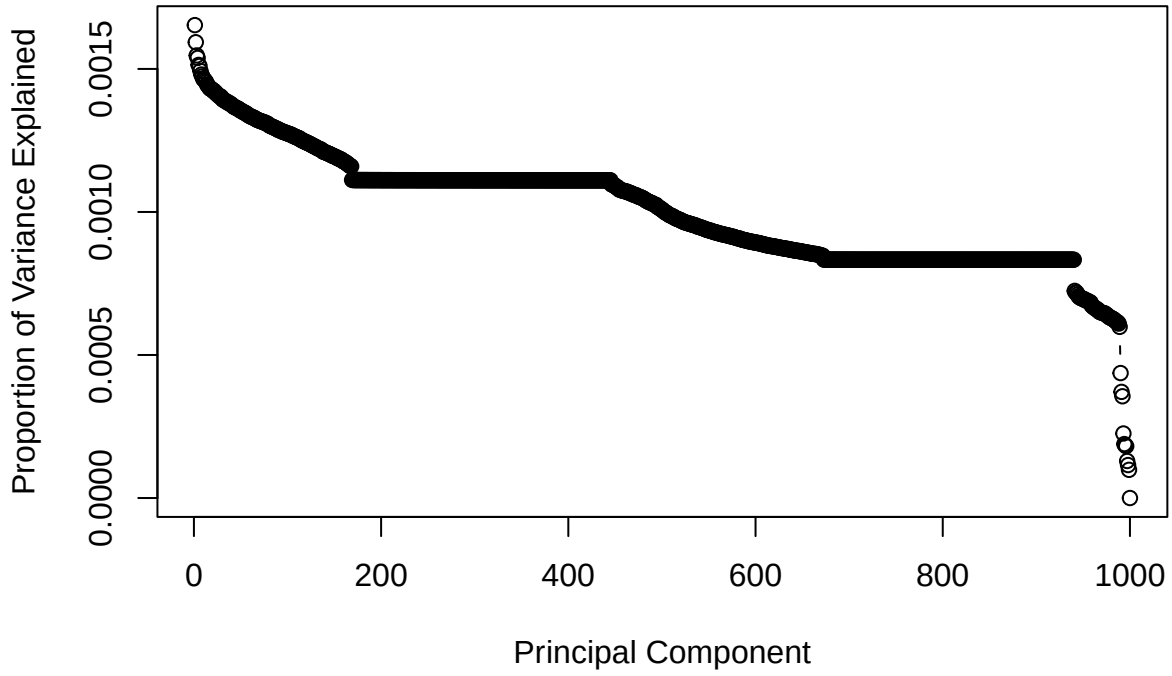
Let us divide the variance by sum of total variance so as to compute the proportion of variance explained by each component.

```
#proportion of variance explained
prop_varex <- pr_var/sum(pr_var)
prop_varex[1:20]
```

```
##  [1] 0.001653126 0.001593303 0.001547198 0.001538442 0.001513954 0.001510131
##  [7] 0.001493079 0.001480427 0.001473949 0.001464745 0.001462306 0.001458008
## [13] 0.001454387 0.001445873 0.001441582 0.001437063 0.001432045 0.001430009
## [19] 0.001429419 0.001425703
```

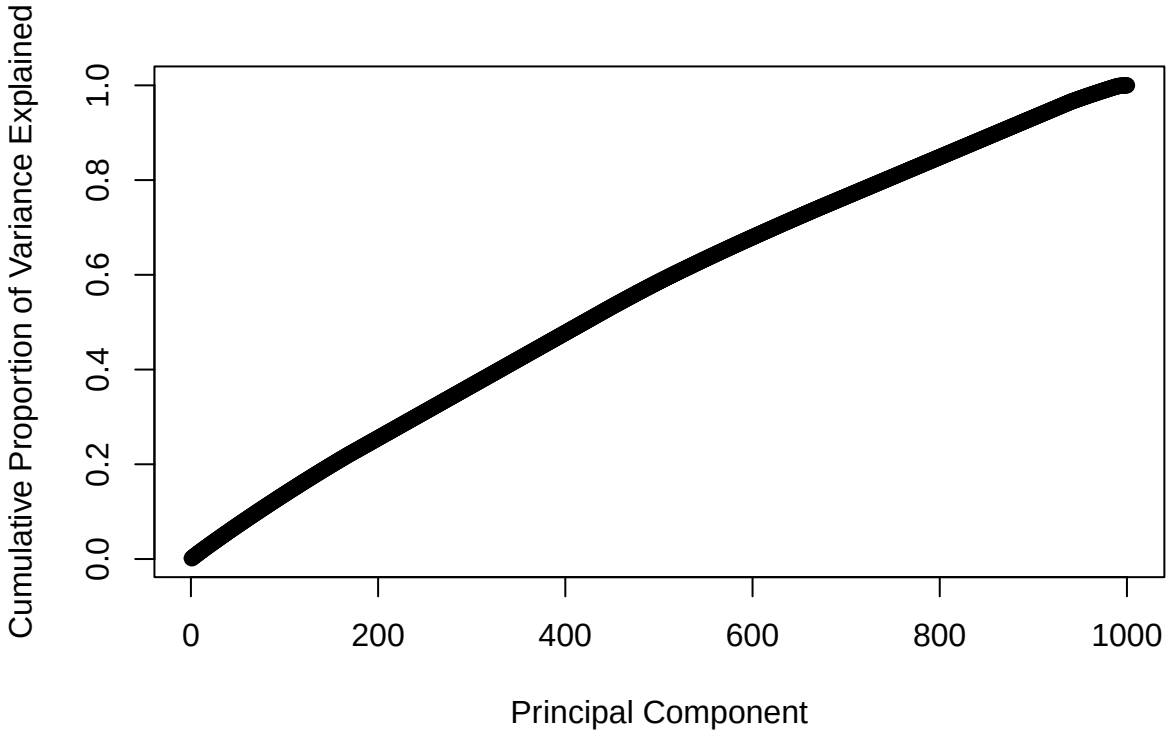
This shows that first principal component explains 0.1511% variance. Second component explains 0.1464% variance. Third component explains 0.1429% variance etc let us use a screeplot to determine components to select for modeling

```
#scree plot
plot(prop_varex, xlab = "Principal Component",
             ylab = "Proportion of Variance Explained",
             type = "b")
```



Let's do a check, by plotting a cumulative variance plot. This will give us a clear picture of number of components.

```
#cumulative scree plot
plot(cumsum(prop_varex), xlab = "Principal Component",
              ylab = "Cumulative Proportion of Variance Explained",
              type = "b")
```




```
library(devtools)
```

```
## Loading required package: usethis
```

```
#install_github("vqv/ggbiplot")

library(ggbiplot)
```

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

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

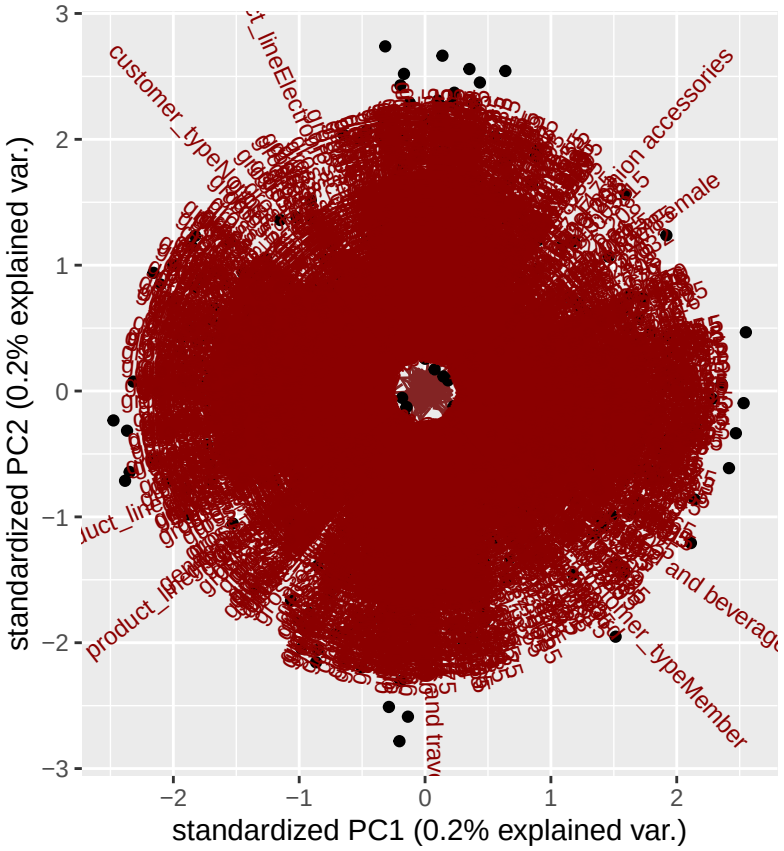
```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

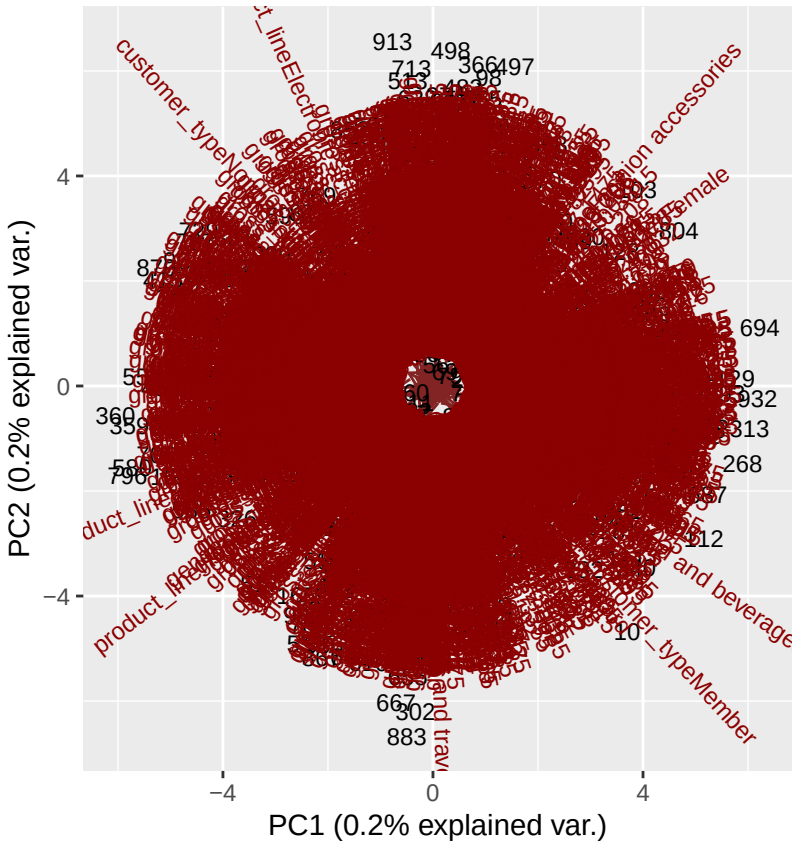
```
## Loading required package: scales

##
## Attaching package: 'scales'

## The following object is masked from 'package:readr':
##
##     col_factor

## Loading required package: grid
```

```
ggbiplot(df.pca)
```



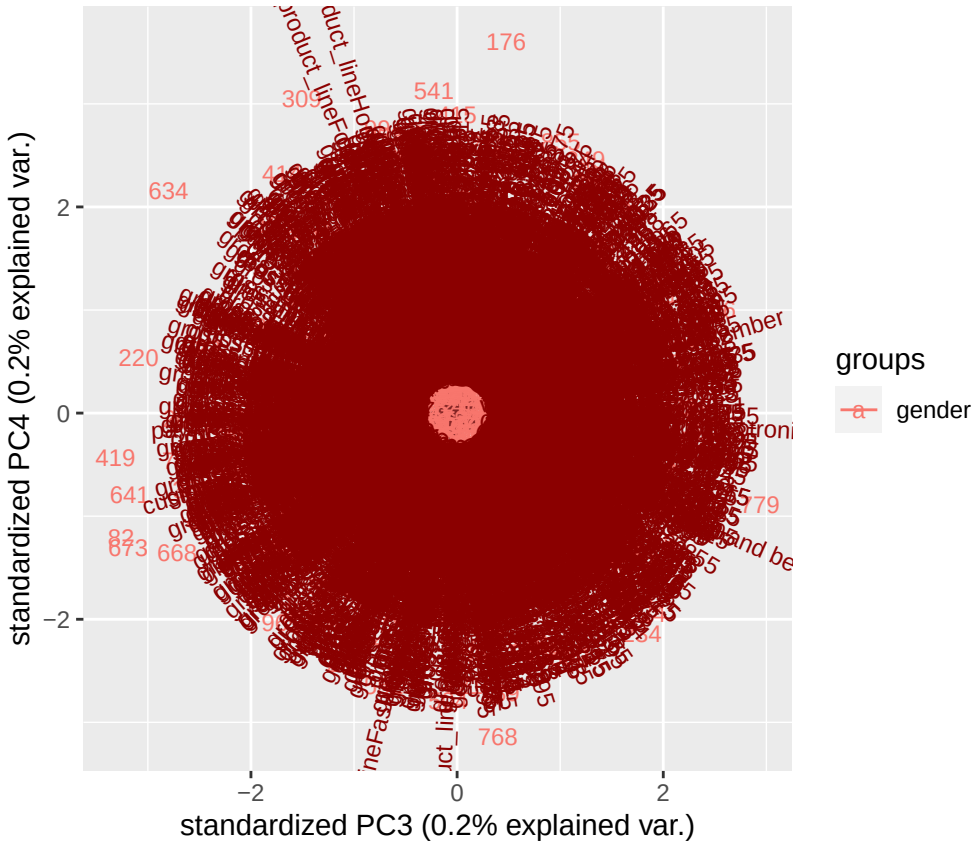
Adding more detail to the plot, we provide arguments rownames as labels

```
ggbiplot(df.pca, labels=rownames(df2), obs.scale = 1, var.scale = 1)
```



We now plot PC3 and PC4

```
ggbiplot(df.pca,ellipse=TRUE,choices=c(3,4),   labels=rownames(df2), groups='gender')
```



Feature engineering

Filter methods

```
library(caret)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

Calculating the correlation matrix

```
# Find attributes that are highly correlated
correlationMatrix <- cor(df2)
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.75)
```

Highly correlated attributes

```
highlyCorrelated
```

```
##    [1]    4    6  726  577   53  862  403  848  822  588  376  923  942  416
##   [15]  590  470   80  332  313  610  675   46   65   49  838  739   39  833
##   [29]  793  947  639  686  667   77  772  506  574  527  618  381  807  444
##   [43]  129  310   88  740  516  400  758  486  131  464  614  508  664  697
##   [57]  690  477  955  573  445  119  276  788   21  663   48  450  775  637
##   [71]   44  417  628  214  251  524  768  805  216  802  399  841  824  496
##   [85]  627  265  953  172  259  716  801  242  785  422  688  389  263  205
##   [99]  255  497  529  355  605  557  849  727  629  680  718  318  237  916
##  [113]  934  201  854  434  306  858  720  603  767  356  883  344  689  223
##  [127]  363  825  258  383  425  502  538  278  919   28  440  483  565  174
##  [141]  868  469   56  600  304  112  699  273  798  864  875  141  185  118
##  [155]  114  881  476  615  762  552  870  632  305  430  921  804   66  277
##  [169]  653  566  902  339  930  301  382  834  385  289   85  701  515  855
##  [183]  631  183  749  151  622  405  681  212   17   34   41 1250   52 1110
##  [197]   67 1307   78   93  101  102  113  127  146  148 1553   32 1295  175
##  [211]  180   42   43  200   55  219   59   61  229  232  234 1507   68 1236
##  [225]  240  241 1142   83  256  257  272   31  285  293 1363  312 1105  316
##  [239]  319  328  122 1384  347 1164  136   60 1160 1524  159   72  398  410
##  [253]  413  176   16  437  438  451  452   19  195   22  465  466 1357  472
##  [267]   25 1111 1415 1326 1433   29   33  550  121  125  563 1062  137   81
##  [281]  584   45 1093  250  254 1324  602  604 1530   27   50  612  270  271
##  [295]   14   15 1063 1548   95   97  641  645  286  165   57  661   58  671
##  [309]  672  674 1107  677   40  687 1385  302 1208 1080   64 1441  705  184
##  [323] 1465 1349  193  732 1408  117  743  198  331   18  754  202 1243 1322
##  [337]  124   20 1305   23  795 1117 1336 1381  799   87   89  358  220  810
##  [351]   92  815  820 1073  364   94  145 1118   96 1523  231  846  847   63
##  [365]  852  853  235  860  156  391  157  869  160   30  888 1065   76  897
##  [379]   35  908   37  920  922  418 1102  249  932   82  424 1297   26 1466
##  [393]  181  182  123   90  463  199   24  138   38  487  494  300  103  210
##  [407]  105 1396 1276  509  512  518   84  162 1150   62  528  323   86   36
##  [421]  540   47  338  168   51   71  554  556 1419  559  238  239  100  349
##  [435] 1188 1149  246  582  191  362  143  106  367 1108  110  203 1083  377
##  [449]  152  613  155  207 1473  617 1159  386 1453  209   70 1259  158  211
##  [463] 1401 1560  213  401  635   54  642  406  217  648  409   79  666 1386
##  [477]  295  179 1487  139  109  311  187 1306  448  314 1245  147 1341   74
##  [491]  456  115   75  730  194  462  324 1214  752  247  757  475  204 1215
##  [505]  337 1518  252  774 1247  485  120  777  492  782 1494  164  503  350
##  [519]  505  794  268  108 1200  169 1171  111  803  134   91  221 1161  361
##  [533]  173  142  281 1549  284  116   99  535  188  379  290 1522  296  387
##  [547]  861  863 1456  865 1420  871  104 1338  397  402  308  901  163  130
##  [561]  132 1173  208  578 1443  315  925  931 1467 1309 1461  939  941  423
##  [575]  951  954 1395  596  177  329  218  178 1320  432  334  433 1061  149
##  [589]  267  442  611  154  225 1233  192 1362  230  279  128  348  636  351
##  [603]  649  135 1203 1390  482  484  294  144  365  170  171  495  299  691
##  [617]  498 1260 1538  150 1147  710  514  394  395  189  317  525  731  734
##  [631] 1334  737  531  226 1172  228 1344  197  746  753  414  166  335  549
##  [645]  341  551  553  342  236  429  786  570  789 1527 1414 1222  291  186
##  [659] 1227  245  800  357  586  812  813  589  823  592  196  366  597 1393
##  [673]  837  261  468  370  606  478 1287  380 1358  206 1394  388 1555  866
##  [687]  393  874  325  499 1342  396 1383 1104  215 1350  282  404 1516  904
##  [701]  905  222 1089  668  670  419 1519  682  945 1127 1316  253  227  693
##  [715]  427  696  260  262  541 1361  307  543  713 1348 1481 1423  274 1407
##  [729]  562  372 1510  564 1059 1303  378  738  283  572  384  576  755  333
##  [743]  473  248  288  761 1379 1261  594  488  489  595 1557  298  780  598
```

```
##  [757] 1437  343  408  264  266  504  353  507  421  511 1166  517  624  275
##  [771]  359  520  280  827  633  431  640  441  443  447  330  850  662  292
##  [785]  459  460  547 1131 1520 1193  879  695  884  558  885  303 1431 1278
##  [799]  889  895  703 1472  898  709  480  346  714  906  411  911 1181  918
##  [813]  491  585 1078 1090  322  501  736  326 1410  368 1497  519  760  336
##  [827]  439  523  609  766 1198  340  776 1318  392  530  625  454 1286  536
##  [841]  461 1175  791  542 1122  352  354 1455  545 1511  654 1505  660  479
##  [855]  811  360  415 1257  817  669  679  685  567  369  569 1285  428  373
##  [869] 1444  375 1340  581  856  857  435  708 1365  510  390 1541 1367  878
##  [883]  522  599  455 1378  735  896 1106  534  903  741  907  467  619  750
##  [897] 1095 1092  756  420  481  544  765  950  771  634 1248 1094  646  560
##  [911]  656  500 1417 1512  659 1240  673 1163 1346  580  449  513  684 1174
##  [925] 1272  806  587  457  826  706 1115  471  844  474 1399 1337 1154  728
##  [939] 1542  490  733  623 1484  493  873  747 1551  751  638 1130  900  571
##  [953]  650  651  652 1267  770  917  521 1263  927  583  935  937  526  787
##  [967]  948 1289 1234 1220  593  790  692  533  796  700 1302  816  546  548
##  [981]  616  620  626  845  561 1072  568  744  745  859  643  575  579  872
##  [995]  876  882  887  591  892  778  683  601  899 1213  694 1406  607 1330
## [1009]  792  924  797  712  943  621 1299  719  721  725  819  821  630 1392
## [1023] 1486  830  832  836  644  647 1116 1279  742  655  851 1206 1133  676
## [1037]  678 1064 1559 1146 1457  880 1470  781  886  891  704  707  711  715
## [1051]  912  722 1060  929  808 1540  936  729  814  818  944  829  748  839
## [1065]  842 1179  759 1479  763  764  773  867  783  784 1277  910  915 1501
## [1079] 1281  928  933  828  938  949  835  840  843  890  894  909  913  914
## [1093] 1351  926 1353  940 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574
## [1107] 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 1588
## [1121] 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602
## [1135] 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616
## [1149] 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630
## [1163] 1631 1632 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644
## [1177] 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658
## [1191] 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672
## [1205] 1673 1674 1675 1676 1677 1678 1679 1680 1681 1682 1683 1684 1685 1686
## [1219] 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696 1697 1698 1699 1700
## [1233] 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712 1713 1714
## [1247] 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728
## [1261] 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742
## [1275] 1743 1744 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756
## [1289] 1757 1758 1759 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770
## [1303] 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784
## [1317] 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794 1795 1796 1797 1798
## [1331] 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812
## [1345] 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824 1825 1826
## [1359] 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840
## [1373] 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854
## [1387] 1855 1856 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868
## [1401] 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882
## [1415] 1883 1884 1885 1886 1887 1888 1889 1890 1891 1892 1893 1894 1895 1896
## [1429] 1897 1898 1899 1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910
## [1443] 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924
## [1457] 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938
## [1471] 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952
## [1485] 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966
## [1499] 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980
```

```
## [1513] 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994
## [1527] 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008
## [1541] 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022
## [1555] 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036
## [1569] 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050
## [1583] 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064
## [1597] 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078
## [1611] 2079 2080 2081 2082 2083 2084 2085 2086 2087 2088 2089 2090 2091 2092
## [1625] 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102 2103 2104 2105 2106
## [1639] 2107 2108 2109 2110 2111 2112 2113 2114 2115 2116 2117 2118 2119 2120
## [1653] 2121 2122 2123 2124 2125 2126 2127 2128 2129 2130 2131 2132 2133 2134
## [1667] 2135 2136 2137 2138 2139 2140 2141 2142 2143 2144 2145 2146 2147 2148
## [1681] 2149 2150 2151 2152 2153 2154 2155 2156 2157 2158 2159 2160 2161 2162
## [1695] 2163 2164 2165 2166 2167 2168 2169 2170 2171 2172 2173 2174 2175 2176
## [1709] 2177 2178 2179 2180 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190
## [1723] 2191 2192 2193 2194 2195 2196 2197 2198 2199 2200 2201 2202 2203 2204
## [1737] 2205 2206 2207 2208 2209 2210 2211 2212 2213 2214 2215 2216 2217 2218
## [1751] 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232
## [1765] 2233 2234 2235 2236 2237 2238 2239 2240 2241 2242 2243 2244 2245 2246
## [1779] 2247 2248 2249 2250 2251 2252 2253 2254 2255 2256 2257 2258 2259 2260
## [1793] 2261 2262 2263 2264 2265 2266 2267 2268 2269 2270 2271 2272 2273 2274
## [1807] 2275 2276 2277 2278 2279 2280 2281 2282 2283 2284 2285 2286 2287 2288
## [1821] 2289 2290 2291 2292 2293 2294 2295 2296 2297 2298 2299 2300 2301 2302
## [1835] 2303 2304 2305 2306 2307 2308 2309 2310 2311 2312 2313 2314 2315 2316
## [1849] 2317 2318 2319 2320 2321 2322 2323 2324 2325 2326 2327 2328 2329 2330
## [1863] 2331 2332 2333 2334 2335 2336 2337 2338 2339 2340 2341 2342 2343 2344
## [1877] 2345 2346 2347 2348 2349 2350 2351 2352 2353 2354 2355 2356 2357 2358
## [1891] 2359 2360 2361 2362 2363 2364 2365 2366 2367 2368 2369 2370 2371 2372
## [1905] 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2383 2384 2385 2386
## [1919] 2387 2388 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2399 2400
## [1933] 2401 2402 2403 2404 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414
## [1947] 2415 2416 2417 2418 2419 2420 2421 2422 2423 2424 2425 2426 2427 2428
## [1961] 2429 2430 2431 2432 2433 2434 2435 2436 2437 2438 2439 2440 2441 2442
## [1975] 2443 2444 2445 2446 2447 2448 2449 2450 2451 2452 2453 2454 2455 2456
## [1989] 2457 2458 2459 2460 2461 2462 2463 2464 2465 2466 2467 2468 2469 2470
## [2003] 2471 2472 2473 2474 2475 2476 2477 2478 2479 2480 2481 2482 2483 2484
## [2017] 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496 2497 2498
## [2031] 2499 2500 2501 2502 2503 2504 2505 2506 2507 2508 2509 2510 2511 2512
## [2045] 2513 2514 2515 2516 2517 2518 2519 2520 2521 2522 2523 2524 2525 2526
## [2059] 2527 2528 2529 2530 2531 2532 2533 2534 2535 2536 2537 2538 2539 2540
## [2073] 2541 2542 2543 2544 2545 2546 2547 2548 2549 2550 2551 2552 2553 2554
```

```
head(names(df2[,highlyCorrelated]))
```

```
## [1] "customer_typeMember" "genderFemale"        "unit_price78.77"
## [4] "unit_price64.99"     "unit_price14.23"     "unit_price93.12"
```

Removing Redundant Features

```
df3 <- df2[-highlyCorrelated]
```

Performing our graphical comparison

```
#par(mfrow = c(1, 2))
#corrplot(correlationMatrix, order = "hclust")
#corrplot(cor(df3), order = "hclust")
```

conclusion