Feature Selection

Stacy

2022-04-03

```
# This section requires you to perform feature selection through the use of the unsupervised learning m
#
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## 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::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(cluster)
library(factoextra)
```

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

```
library(ggplot2)
library(dendextend)
```

```
##
## ---------------------
## Welcome to dendextend version 1.15.2
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(tidyverse)
library(magrittr)
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(numDeriv)
library(e1071)
library (caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(moments)
```

```
##
## Attaching package: 'moments'

## The following objects are masked from 'package:e1071':
##
##     kurtosis, moment, skewness
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
dataset<- read.csv('http://bit.ly/CarreFourDataset')
head(dataset)
```

```
##    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
```

```
dim(dataset)
```

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

```
colnames(dataset)
```

```
##  [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"
```

```
#  sum of null values per column
colSums(is.na(dataset))
```

```
##              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
```

```
str(dataset)
```

```
## '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 ...
```

```
# Deal with Duplicated values
#unique_data <- unique(data)
dup<- dataset[duplicated(dataset),]
head(dup)
```

```
##  [1] Invoice.ID              Branch                  Customer.type
##  [4] Gender                  Product.line            Unit.price
##  [7] Quantity                Tax                     Date
## [10] Time                    Payment                 cogs
## [13] gross.margin.percentage gross.income            Rating
## [16] Total
## <0 rows> (or 0-length row.names)
```

```
summary(dataset)
```

```
##   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
```

```r
# Find the mean of numeric columns
colMeans(dataset[sapply(dataset, is.numeric)])
```

```
##              Unit.price                Quantity                     Tax
##               55.672130                5.510000               15.379369
##                    cogs gross.margin.percentage            gross.income
##              307.587380                4.761905               15.379369
##                  Rating                   Total
##                6.972700              322.966749
```

```r
# Standard Deviation
sapply(dataset, sd)
```

```
## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =
## na.rm): NAs introduced by coercion
```

```
##               Invoice.ID                    Branch             Customer.type
##                       NA                        NA                        NA
##                   Gender              Product.line                Unit.price
##                       NA                        NA                 26.494628
##                 Quantity                       Tax                      Date
##                 2.923431                  11.708825                        NA
##                     Time                   Payment                      cogs
##                       NA                        NA                234.176510
## gross.margin.percentage              gross.income                    Rating
##                 0.000000                  11.708825                 1.718580
##                    Total
##                245.885335
```

```
# Kurtosis
# Unit.price, Quantity, Tax, cogs, gross.margin.percentage,  gross.income
# Rating, Total
kurtosis(dataset$Unit.price, na.rm=FALSE)
```

```
## [1] 1.781499
```

```
kurtosis(dataset$Quantity, na.rm=FALSE)
```

```
## [1] 1.784528
```

```
kurtosis(dataset$Tax, na.rm=FALSE)
```

```
## [1] 2.91253
```

```
kurtosis(dataset$cogs, na.rm=FALSE)
```

```
## [1] 2.91253
```

```
kurtosis(dataset$gross.income, na.rm=FALSE)
```

```
## [1] 2.91253
```

```
kurtosis(dataset$Rating, na.rm=FALSE)
```

```
## [1] 1.848169
```

```
kurtosis(dataset$Total, na.rm=FALSE)
```

```
## [1] 2.91253
```

```
skewness(dataset$Unit.price, na.rm=FALSE)
```

```
## [1] 0.007066827
```

```r
skewness(dataset$Quantity, na.rm=FALSE)
```

```
## [1] 0.01292163
```

```r
skewness(dataset$Tax, na.rm=FALSE)
```

```
## [1] 0.8912304
```

```r
skewness(dataset$cogs, na.rm=FALSE)
```

```
## [1] 0.8912304
```

```r
skewness(dataset$gross.income, na.rm=FALSE)
```

```
## [1] 0.8912304
```

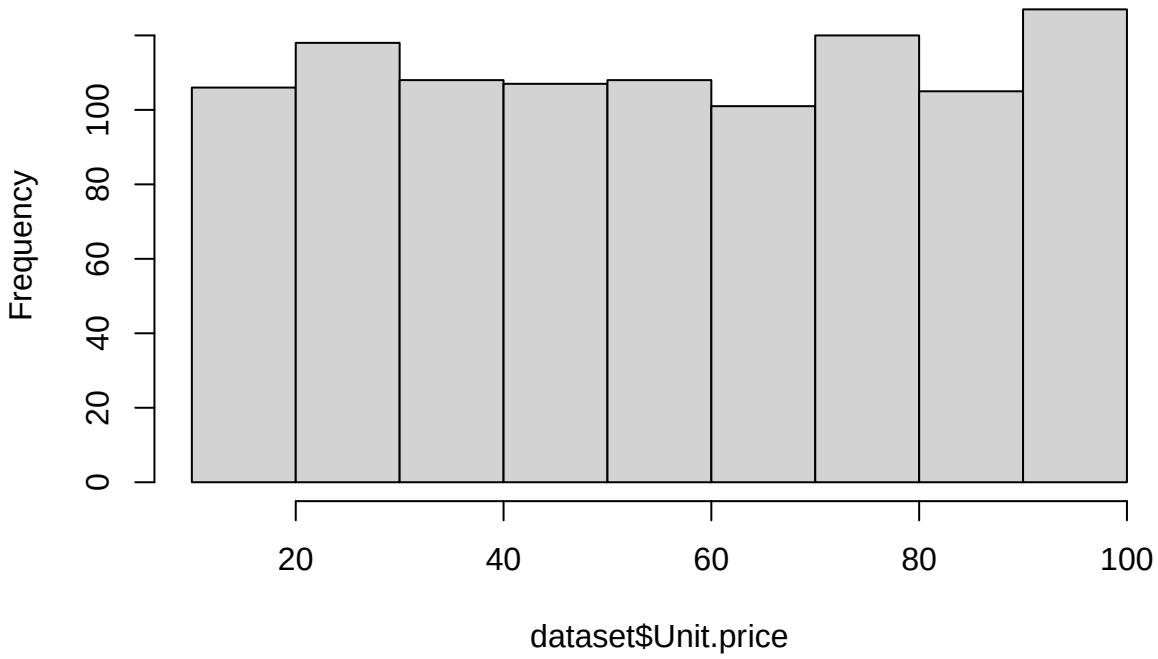
```r
skewness(dataset$Rating, na.rm=FALSE)
```

```
## [1] 0.008996129
```

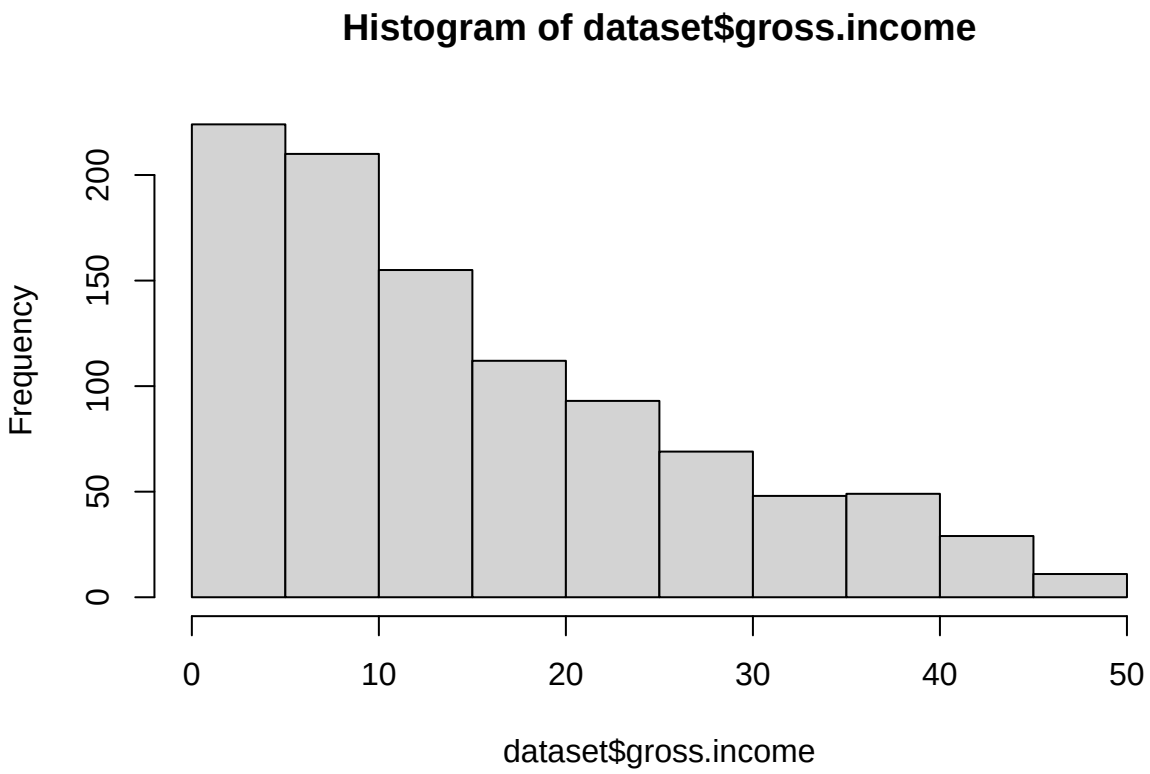
```r
skewness(dataset$Total, na.rm=FALSE)
```

```
## [1] 0.8912304
```

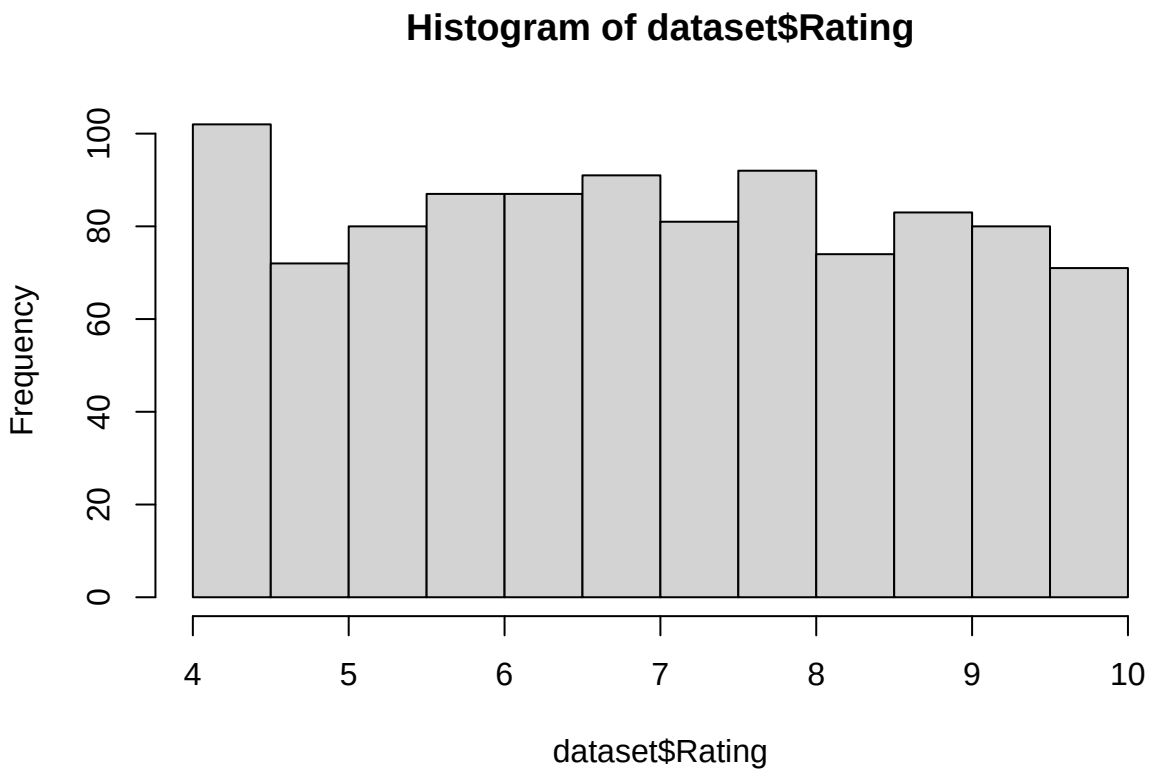
```r
# data vs Unit.price
hist(dataset$Unit.price)
```



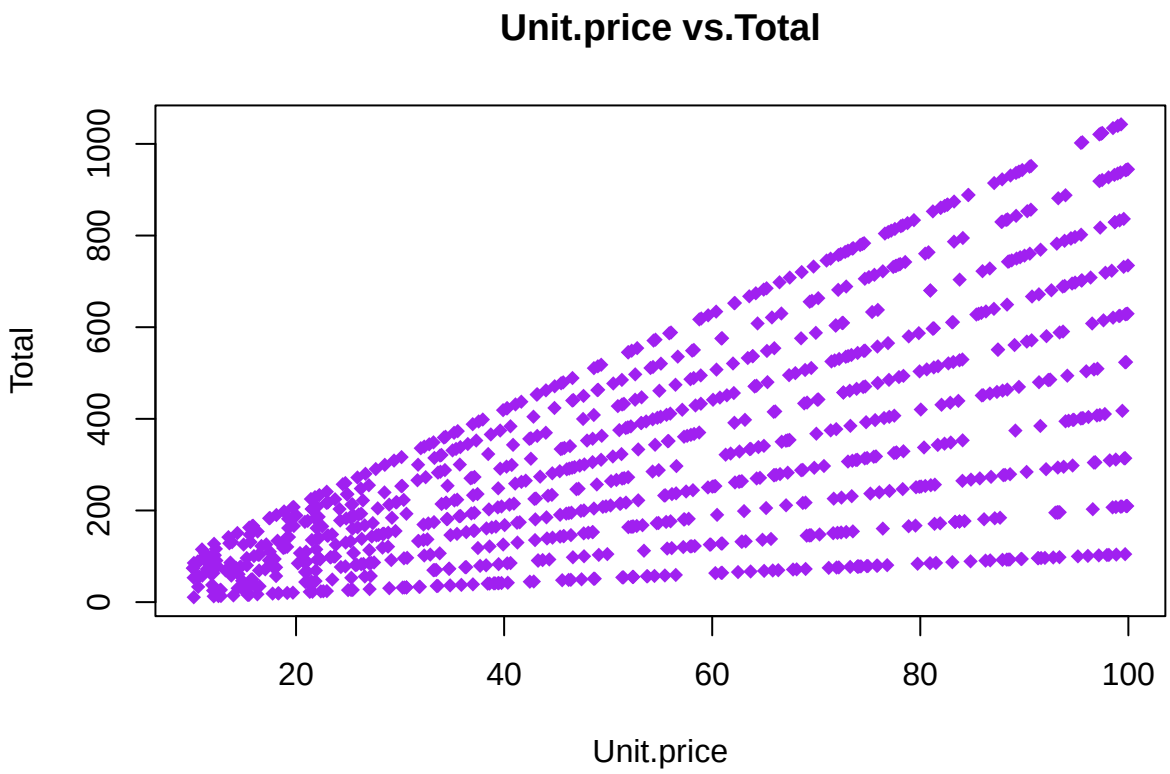

```
hist(dataset$gross.income)
```




```
hist(dataset$Rating)
```




```
# Scatter plot :dataset vs Unit.price
plot(dataset$Unit.price, dataset$Total, pch=18, col='purple',
     main='Unit.price vs.Total',
     xlab='Unit.price', ylab='Total')
```




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

```
## corrplot 0.92 loaded
```

```
library(mclust)
```

```
## Package 'mclust' version 5.4.9
## Type 'citation("mclust")' for citing this R package in publications.

##
## Attaching package: 'mclust'

## The following object is masked from 'package:purrr':
##
##     map
```

```
library(clustvarsel)
```

```
## Package 'clustvarsel' version 2.3.4

## Type 'citation("clustvarsel")' for citing this R package in publications.
```

```
library(mlbench)
```

```
# select numeric columns
my_data=dataset %>% select_if(is.numeric)
head(my_data)
```

```
##   Unit.price Quantity     Tax   cogs gross.margin.percentage gross.income
## 1      74.69        7 26.1415 522.83                4.761905      26.1415
## 2      15.28        5  3.8200  76.40                4.761905       3.8200
## 3      46.33        7 16.2155 324.31                4.761905      16.2155
## 4      58.22        8 23.2880 465.76                4.761905      23.2880
## 5      86.31        7 30.2085 604.17                4.761905      30.2085
## 6      85.39        7 29.8865 597.73                4.761905      29.8865
##   Rating    Total
## 1    9.1 548.9715
## 2    9.6  80.2200
## 3    7.4 340.5255
## 4    8.4 489.0480
## 5    5.3 634.3785
## 6    4.1 627.6165
```

```
# calculate correlation matrix
correlationMatrix <- cor(my_data)
```

```
## Warning in cor(my_data): the standard deviation is zero
```

```
# summarize the correlation matrix
print(correlationMatrix)
```

```
##                            Unit.price    Quantity        Tax       cogs
## Unit.price                1.000000000  0.01077756  0.6339621  0.6339621
## Quantity                  0.010777564  1.00000000  0.7055102  0.7055102
## Tax                       0.633962089  0.70551019  1.0000000  1.0000000
## cogs                      0.633962089  0.70551019  1.0000000  1.0000000
## gross.margin.percentage           NA          NA         NA         NA
## gross.income              0.633962089  0.70551019  1.0000000  1.0000000
## Rating                   -0.008777507 -0.01581490 -0.0364417 -0.0364417
## Total                     0.633962089  0.70551019  1.0000000  1.0000000
##                          gross.margin.percentage gross.income       Rating
## Unit.price                                    NA    0.6339621 -0.008777507
## Quantity                                      NA    0.7055102 -0.015814905
## Tax                                           NA    1.0000000 -0.036441705
## cogs                                          NA    1.0000000 -0.036441705
## gross.margin.percentage                        1           NA           NA
## gross.income                                  NA    1.0000000 -0.036441705
## Rating                                        NA   -0.0364417  1.000000000
## Total                                         NA    1.0000000 -0.036441705
##                               Total
## Unit.price                0.6339621
## Quantity                  0.7055102
## Tax                       1.0000000
```

```
## cogs                      1.0000000
## gross.margin.percentage         NA
## gross.income              1.0000000
## Rating                   -0.0364417
## Total                     1.0000000
```

```
# find attributes that are highly corrected (ideally >0.75)
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.5)
# print indexes of highly correlated attributes
print(highlyCorrelated)
```

```
## [1] 4 8 3 6
```

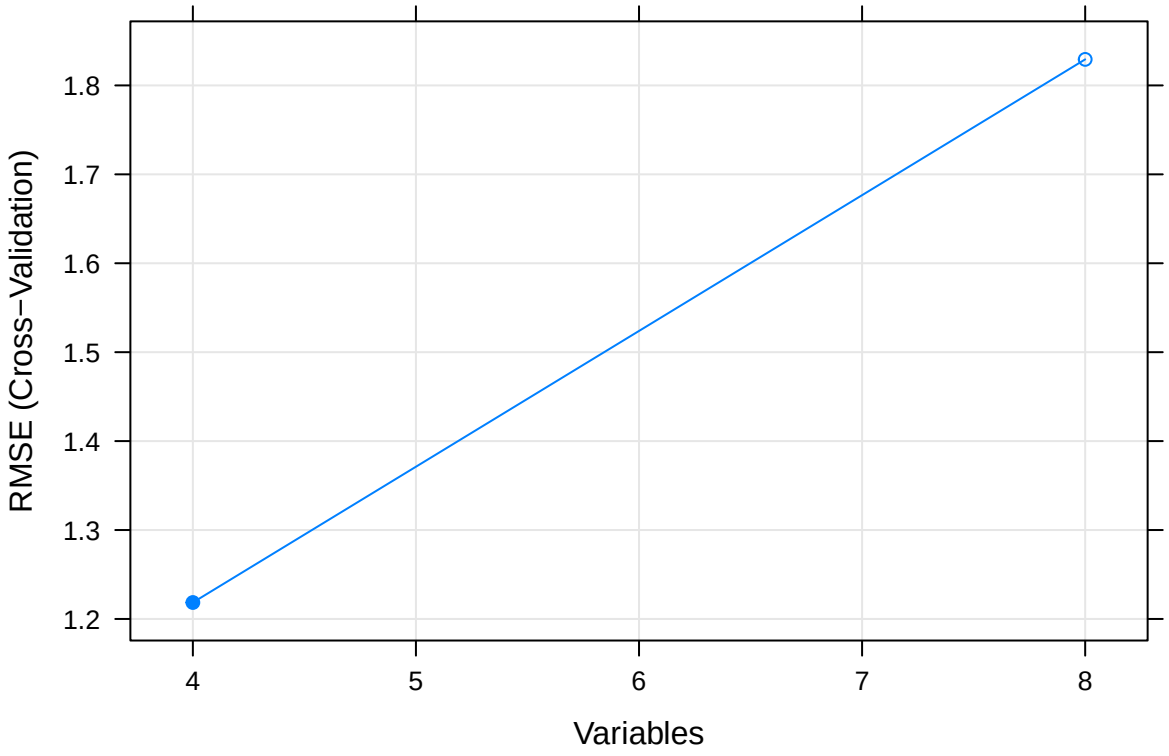
```
# define the control using a random forest selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
# run the RFE algorithm
results <- rfe(my_data, my_data[,8], rfeControl=control)
# summarize the results
print(results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables  RMSE Rsquared    MAE RMSESD RsquaredSD  MAESD Selected
##          4 1.218   1.0000 0.6353 0.3570  1.274e-05 0.1158        *
##          8 1.829   0.9999 1.0920 0.6431  2.669e-05 0.2297
##
## The top 4 variables (out of 4):
##    Tax, Total, cogs, gross.income
```

```
# list the chosen features
predictors(results)
```

```
## [1] "Tax"          "Total"        "cogs"         "gross.income"
```

```
# plot the results
plot(results, type=c("g", "o"))
```




```
library(rquery)
```

```
## Loading required package: wrapr
```

```
##
## Attaching package: 'wrapr'
```

```
## The following object is masked from 'package:dplyr':
##
##     coalesce
```

```
## The following objects are masked from 'package:tidyr':
##
##     pack, unpack
```

```
## The following object is masked from 'package:tibble':
##
##     view
```

```
##
## Attaching package: 'rquery'
```

```
## The following object is masked from 'package:tidyr':
##
##     expand_grid
```

```
## The following object is masked from 'package:ggplot2':
##
##     arrow
```

```
library(CORM)
```

```
## Loading required package: limma
```

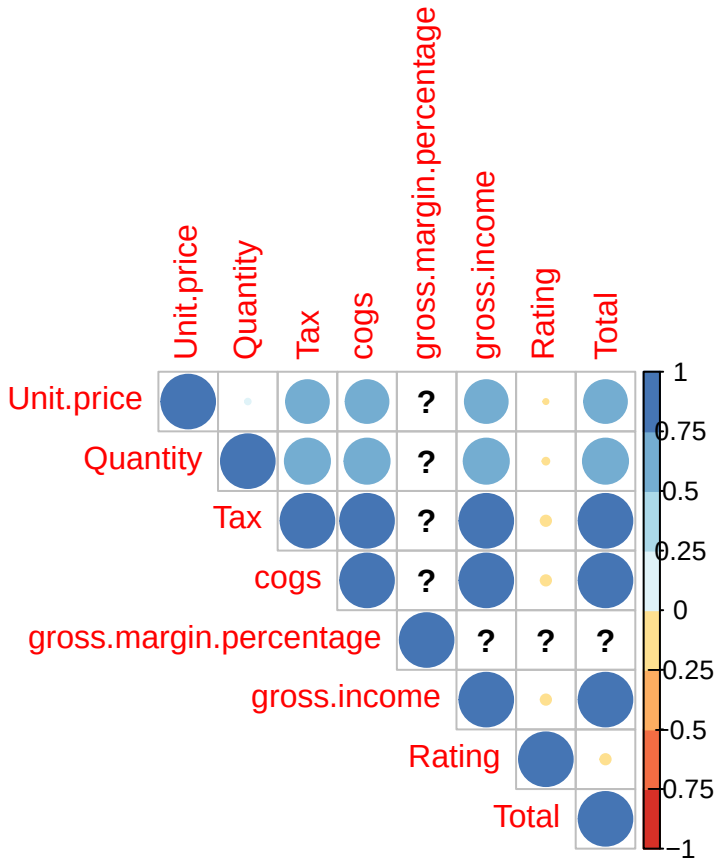
```
library(tidyverse)
library(corrplot)
library(RColorBrewer)
```

```
df2=unlist(my_data)
```

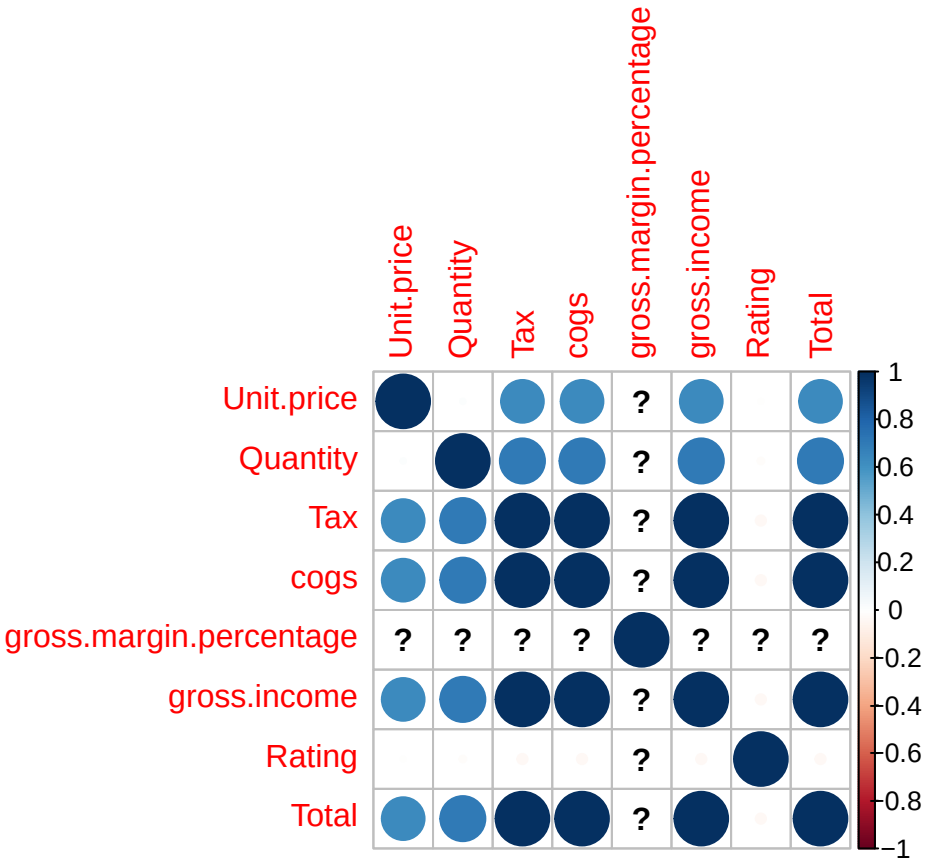
```
M <-cor(my_data)
```

```
## Warning in cor(my_data): the standard deviation is zero
```

```
corrplot(M, type="upper",
         col=brewer.pal(n=8, name="RdYlBu"))
```



```
corrplot(M, method="circle")
```




```
# Positive correlations are displayed in blue and negative correlations in red color. Color intensity a
```