Dimensionality Reduction and Feature Selection

RuthNguli

2022-03-31

Loading libraries

```
# call libraries
#
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(devtools)
```

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

```
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

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

## Loading required package: scales

##
## Attaching package: 'scales'

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

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

## Loading required package: grid
```

```
library(ggplot2)
library(Rtsne)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

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

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(clustvarsel)
```

```
## Loading required package: 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

## Package 'clustvarsel' version 2.3.4

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

```
library(mclust)
library(wskm)
```

```
## Loading required package: latticeExtra

##
## Attaching package: 'latticeExtra'

## The following object is masked from 'package:ggplot2':
##
##     layer

## Loading required package: fpc
```

```
library("cluster")
library(FSelector)
```

Loading and viewing data

```
# load and read data from url
#
data <- read.csv("http://bit.ly/CarreFourDataset")

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

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

```
##   gross.income Rating    Total
## 1      26.1415    9.1 548.9715
## 2       3.8200    9.6  80.2200
## 3      16.2155    7.4 340.5255
## 4      23.2880    8.4 489.0480
## 5      30.2085    5.3 634.3785
## 6      29.8865    4.1 627.6165
```

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

```
##       Invoice.ID Branch Customer.type Gender          Product.line Unit.price
## 995  652-49-6720      C        Member Female Electronic accessories      60.95
## 996  233-67-5758      C        Normal   Male      Health and beauty      40.35
## 997  303-96-2227      B        Normal Female     Home and lifestyle      97.38
## 998  727-02-1313      A        Member   Male     Food and beverages      31.84
## 999  347-56-2442      A        Normal   Male     Home and lifestyle      65.82
## 1000 849-09-3807      A        Member Female    Fashion accessories      88.34
##      Quantity     Tax      Date  Time Payment   cogs gross.margin.percentage
## 995         1  3.0475 2/18/2019 11:40 Ewallet  60.95                4.761905
## 996         1  2.0175 1/29/2019 13:46 Ewallet  40.35                4.761905
## 997        10 48.6900  3/2/2019 17:16 Ewallet 973.80                4.761905
## 998         1  1.5920  2/9/2019 13:22    Cash  31.84                4.761905
## 999         1  3.2910 2/22/2019 15:33    Cash  65.82                4.761905
## 1000        7 30.9190 2/18/2019 13:28    Cash 618.38                4.761905
##      gross.income Rating     Total
## 995        3.0475    5.9   63.9975
## 996        2.0175    6.2   42.3675
## 997       48.6900    4.4 1022.4900
## 998        1.5920    7.7   33.4320
## 999        3.2910    4.1   69.1110
## 1000      30.9190    6.6  649.2990
```

```
# Check the data structure
#
str(data)
```

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

Datatypes are well represented.

Data Cleaning

```
# Checking for missing values
#
anyNA(data)
```

```
## [1] FALSE
```

```
# checking if data is duplicated
#

anyDuplicated(data)
```

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

There are no duplicates and missing values in the dataset.

```
# Checking for summary
#
summary(data)
```

```
##   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
```

```
# converting columns to numeric
#
data$Quantity <- as.numeric(data$Quantity)
data$Branch <- as.numeric(as.factor(data$Branch))
data$Customer.type <- as.numeric(as.factor(data$Customer.type))
data$Product.line<- as.numeric(as.factor(data$Product.line))
data$Payment <- as.numeric(as.factor(data$Payment))

# converting gender column  to factor
#
data$Gender <- as.factor(data$Gender)
```

Dimensionality Reduction

1. PCA

```
#  selecting numerical columns
#
data_num <- data %>%
  select(-c(Invoice.ID,Gender,Date,Time))
#data_num <- data %>%
 # select(c(Unit.price,Quantity,Tax,cogs,gross.margin.percentage,gross.income,Rating,Total))
# Get non constant numerical columns
#
data_num <- data_num[,apply(data_num, 2, var, na.rm=TRUE) != 0]

# previewing numerical columns without the constant columns
#
head(data_num)
```

```
##   Branch Customer.type Product.line Unit.price Quantity     Tax Payment   cogs
## 1      1             1            4      74.69        7 26.1415       3 522.83
## 2      3             2            1      15.28        5  3.8200       1  76.40
## 3      1             2            5      46.33        7 16.2155       2 324.31
## 4      1             1            4      58.22        8 23.2880       3 465.76
## 5      1             2            6      86.31        7 30.2085       3 604.17
## 6      3             2            1      85.39        7 29.8865       3 597.73
##   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
```

```
# performing PCA
#
data_pca <- prcomp(data_num, center = TRUE, scale. = TRUE)

# looking at the summary of pca data
#
summary(data_pca)
```

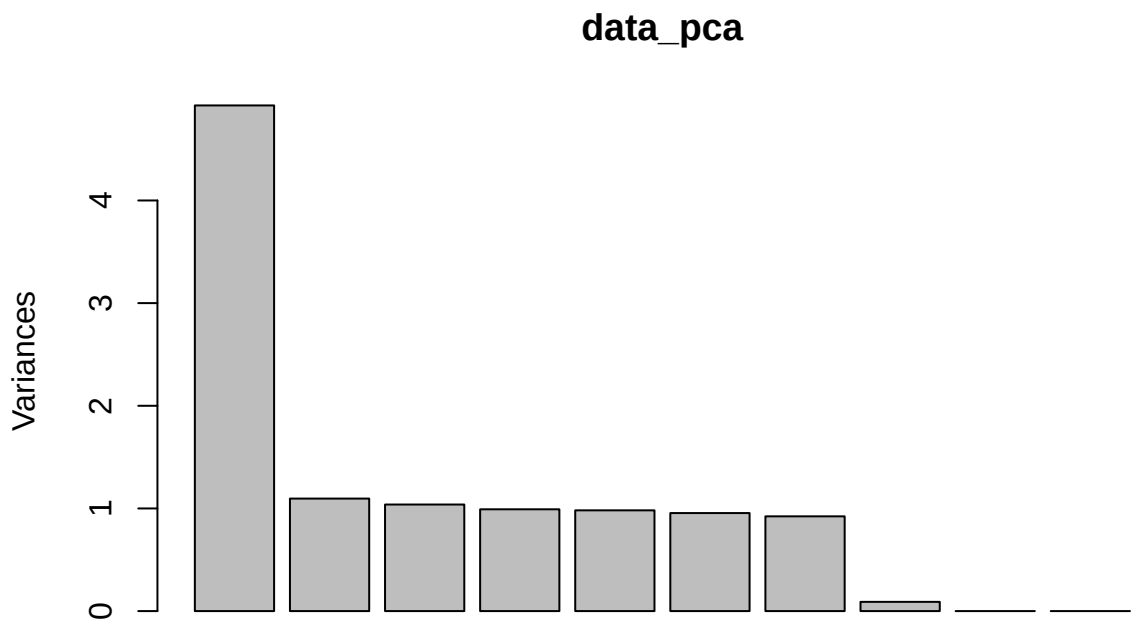
```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5    PC6     PC7
## Standard deviation     2.2194 1.04689 1.01886 0.99571 0.99060 0.9771 0.96064
## Proportion of Variance 0.4478 0.09963 0.09437 0.09013 0.08921 0.0868 0.08389
## Cumulative Proportion  0.4478 0.54743 0.64180 0.73193 0.82114 0.9079 0.99183
##                           PC8       PC9      PC10      PC11
## Standard deviation     0.29978 3.256e-16 1.642e-16 1.033e-16
## Proportion of Variance 0.00817 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00 1.000e+00 1.000e+00
```

We have 11 principal components.

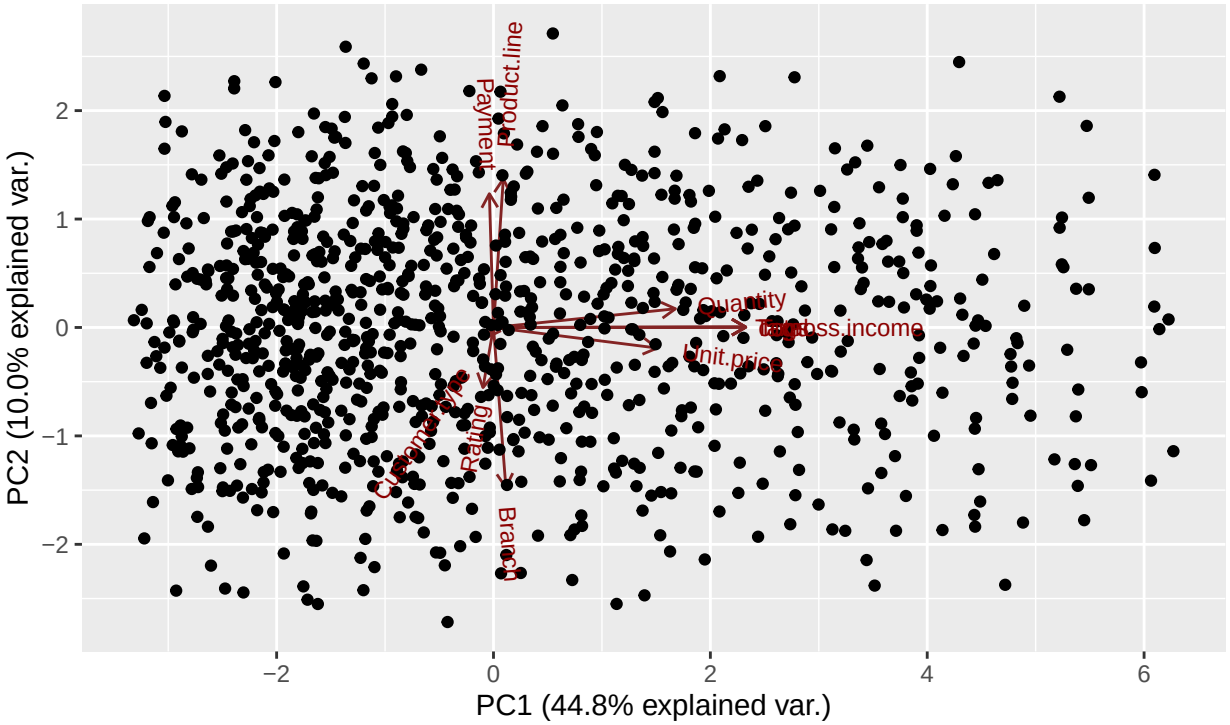
```
# looking at the pca structure
#
str(data_pca)
```

```
## List of 5
##  $ sdev    : num [1:11] 2.219 1.047 1.019 0.996 0.991 ...
##  $ rotation: num [1:11, 1:11] 0.0221 -0.0123 0.0174 0.2917 0.3243 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:11] "Branch" "Customer.type" "Product.line" "Unit.price" ...
##   .. ..$ : chr [1:11] "PC1" "PC2" "PC3" "PC4" ...
##  $ center  : Named num [1:11] 1.99 1.5 3.45 55.67 5.51 ...
##   ..- attr(*, "names")= chr [1:11] "Branch" "Customer.type" "Product.line" "Unit.price" ...
##  $ scale   : Named num [1:11] 0.818 0.5 1.715 26.495 2.923 ...
##   ..- attr(*, "names")= chr [1:11] "Branch" "Customer.type" "Product.line" "Unit.price" ...
##  $ x       : num [1:1000, 1:11] 1.987 -2.306 0.163 1.486 2.776 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:11] "PC1" "PC2" "PC3" "PC4" ...
##  - attr(*, "class")= chr "prcomp"
```

```
# visualizing the variance across each component
#
plot(data_pca)
```




```
# visualizing the Pca using ggbiplot
#
ggbiplot(data_pca, obs.scale = 1, var.scale = 1)
```

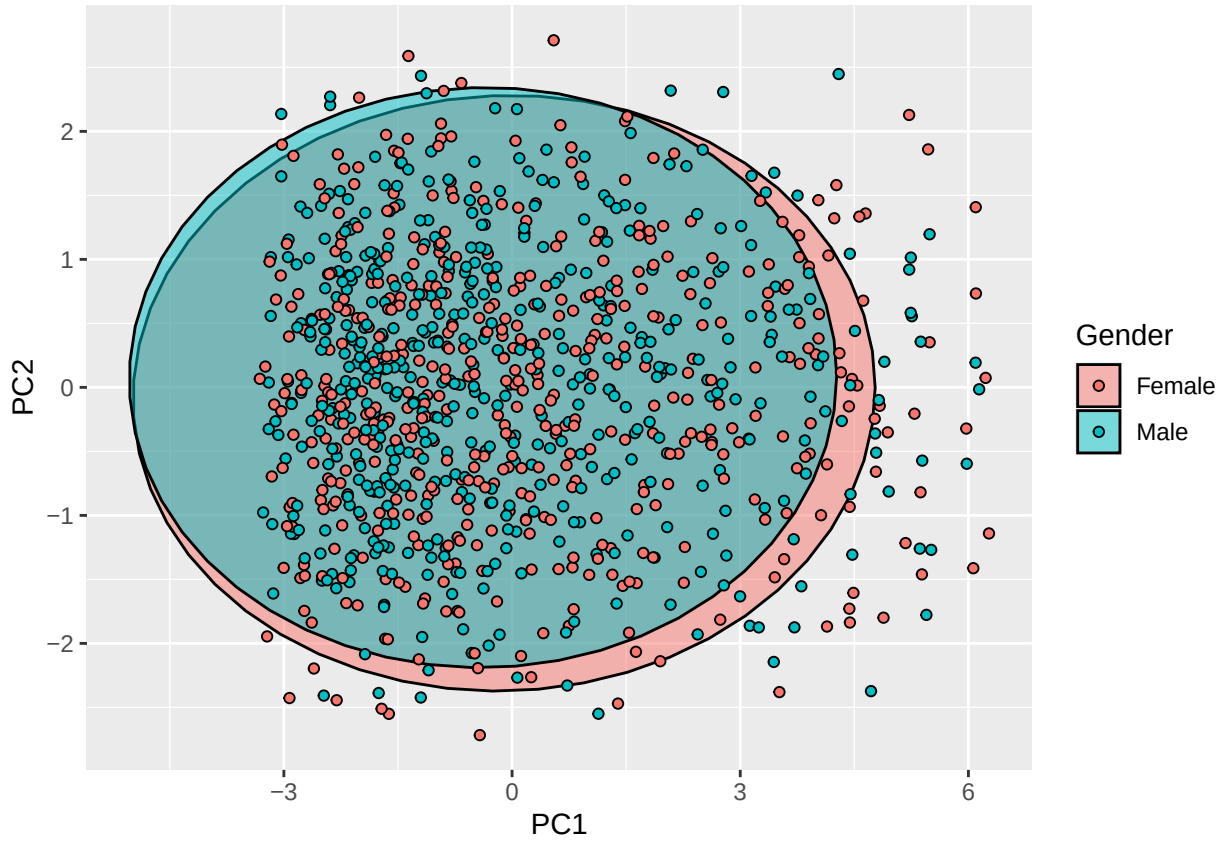



```
# exctracting pc1 & pc2 score and mapping them to dataset for visualizing
#
#data_pca$x

data2 <- cbind(data, data_pca$x[,1:2])
head(data2)
```

```
##    Invoice.ID Branch Customer.type Gender Product.line Unit.price Quantity
## 1 750-67-8428      1             1 Female            4      74.69        7
## 2 226-31-3081      3             2 Female            1      15.28        5
## 3 631-41-3108      1             2   Male            5      46.33        7
## 4 123-19-1176      1             1   Male            4      58.22        8
## 5 373-73-7910      1             2   Male            6      86.31        7
## 6 699-14-3026      3             2   Male            1      85.39        7
##       Tax      Date  Time Payment   cogs gross.margin.percentage gross.income
## 1 26.1415  1/5/2019 13:08       3 522.83                4.761905      26.1415
## 2  3.8200  3/8/2019 10:29       1  76.40                4.761905       3.8200
## 3 16.2155  3/3/2019 13:23       2 324.31                4.761905      16.2155
## 4 23.2880 1/27/2019 20:33       3 465.76                4.761905      23.2880
## 5 30.2085  2/8/2019 10:37       3 604.17                4.761905      30.2085
## 6 29.8865 3/25/2019 18:30       3 597.73                4.761905      29.8865
##   Rating    Total        PC1        PC2
## 1    9.1 548.9715  1.9865730  1.2600100
## 2    9.6  80.2200 -2.3061811 -2.4440077
## 3    7.4 340.5255  0.1629462  1.2055482
## 4    8.4 489.0480  1.4855788  1.4236300
```

```
## 5    5.3 634.3785  2.7762861  2.3075432
## 6    4.1 627.6165  2.7329510 -0.6465001
```

```
# Visualizing pca using ggplot2
#
ggplot(data2, aes(PC1,PC2, col = Gender, fill = Gender))+
  stat_ellipse(geom = "polygon", col = "black", alpha= 0.5) +
  geom_point(shape = 21, col = "black")
```



We observe that majority of the points of Female and Male are more similar each other.

```
# Correlation between variables and principle components
#
cor(data_num, data2[,17:18])
```

```
##                      PC1          PC2
## Branch         0.04901148 -0.632919376
## Customer.type -0.02729099 -0.040029833
## Product.line   0.03872125  0.587677627
## Unit.price     0.64732505 -0.081330080
## Quantity       0.71972102  0.073933258
## Tax            0.99775370  0.001353063
## Payment       -0.01638717  0.529051503
## cogs           0.99775370  0.001353063
## gross.income   0.99775370  0.001353063
## Rating        -0.04157809 -0.237564750
## Total          0.99775370  0.001353063
```

gross income,Cogs,Total & Tax is 0.99 correlated to PC1 which is a strong positive indicating that if pC1 increases gross income increases.

Branch -0.63 is negatively correlated to PC2 indicating that a decrease in PC2 will cause a decrease in Branch

2. TSNE

```
# Curating the data for analysis
# choosing label
Labels<-data$Gender

# preparing data For plotting
#
colors = rainbow(length(unique(data$Gender)))
names(colors) = unique(data$Gender)
```

```
# Executing the algorithm on curated data
#
tsne <- Rtsne(data[,-4], dims = 2, perplexity=30, verbose=TRUE, max_iter = 500)
```

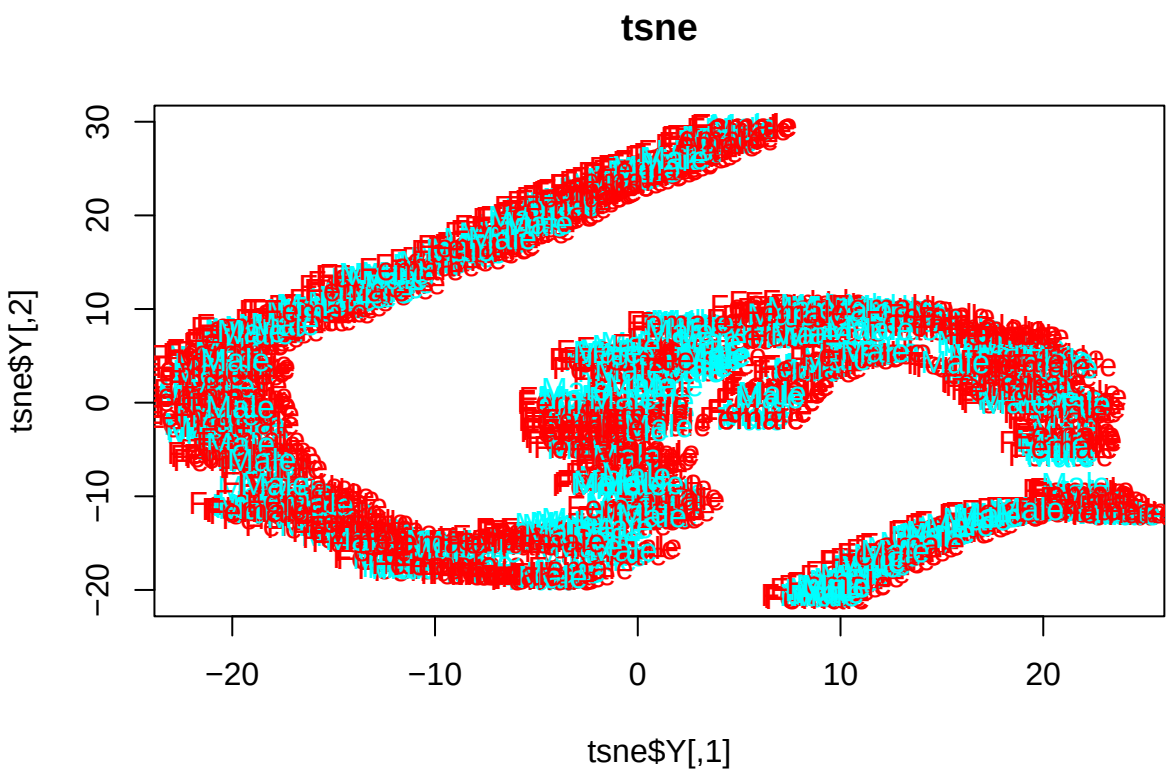
```
## Performing PCA
## Read the 1000 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.28 seconds (sparsity = 0.101272)!
## Learning embedding...
## Iteration 50: error is 59.433514 (50 iterations in 0.26 seconds)
## Iteration 100: error is 51.788388 (50 iterations in 0.24 seconds)
## Iteration 150: error is 50.640449 (50 iterations in 0.24 seconds)
## Iteration 200: error is 50.226310 (50 iterations in 0.25 seconds)
## Iteration 250: error is 49.975352 (50 iterations in 0.25 seconds)
## Iteration 300: error is 0.543736 (50 iterations in 0.24 seconds)
## Iteration 350: error is 0.400211 (50 iterations in 0.24 seconds)
## Iteration 400: error is 0.368138 (50 iterations in 0.25 seconds)
## Iteration 450: error is 0.349630 (50 iterations in 0.25 seconds)
## Iteration 500: error is 0.336503 (50 iterations in 0.25 seconds)
## Fitting performed in 2.47 seconds.
```

```
# Getting the duration of execution
#
exeTimeTsne <- system.time(Rtsne(data[,-4], dims = 2, perplexity=30, verbose=TRUE, max_iter = 500))
```

```
## Performing PCA
## Read the 1000 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.500000
## Computing input similarities...
```

```
## Building tree...
## Done in 0.27 seconds (sparsity = 0.101272)!
## Learning embedding...
## Iteration 50: error is 59.079407 (50 iterations in 0.31 seconds)
## Iteration 100: error is 51.982216 (50 iterations in 0.24 seconds)
## Iteration 150: error is 50.714632 (50 iterations in 0.25 seconds)
## Iteration 200: error is 50.213405 (50 iterations in 0.24 seconds)
## Iteration 250: error is 49.950876 (50 iterations in 0.24 seconds)
## Iteration 300: error is 0.541145 (50 iterations in 0.25 seconds)
## Iteration 350: error is 0.390112 (50 iterations in 0.24 seconds)
## Iteration 400: error is 0.351686 (50 iterations in 0.24 seconds)
## Iteration 450: error is 0.335945 (50 iterations in 0.24 seconds)
## Iteration 500: error is 0.327229 (50 iterations in 0.24 seconds)
## Fitting performed in 2.49 seconds.
```

```
# visualizing
#
plot(tsne$Y, t='n', main="tsne")
text(tsne$Y, labels=data$Gender, col=colors[data$Gender])
```



Feature Selection

```
#  selecting numerical columns
#
```

```
data_num <- data %>%
  select(-c(Invoice.ID,Gender,Date,Time))
```

1. Filter Method

```
# Calculating the correlation matrix
#
corr <- cor(data_num)
```

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

```
# Get non constant numerical columns
#
data_num <- data_num[,apply(data_num, 2, var, na.rm=TRUE) != 0]

# Finding highly correlated attributes
#

high_corr <- findCorrelation(corr, cutoff=0.75)

#showing the highly correlated attributes
#
high_corr
```

```
## [1]  6  8 10
```

```
names(data_num[,high_corr])
```

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

```
# Removing the highly correlated variables
#
Data_new <- data_num[-high_corr]

# previewing the new data
#
head(Data_new)
```

```
##   Branch Customer.type Product.line Unit.price Quantity Payment gross.income
## 1      1             1            4      74.69        7       3      26.1415
## 2      3             2            1      15.28        5       1       3.8200
## 3      1             2            5      46.33        7       2      16.2155
## 4      1             1            4      58.22        8       3      23.2880
## 5      1             2            6      86.31        7       3      30.2085
## 6      3             2            1      85.39        7       3      29.8865
##      Total
## 1 548.9715
## 2  80.2200
```

```
## 3 340.5255
## 4 489.0480
## 5 634.3785
## 6 627.6165
```

2. Wrapper Method

```
# Performing greed search
#
data_out <- clustvarsel(data_num)
data_out
```

```
## ----------------------------------------------------
## Variable selection for Gaussian model-based clustering
## Stepwise (forward/backward) greedy search
## ----------------------------------------------------
##
##  Variable proposed Type of step   BICclust Model G   BICdiff Decision
##           Quantity          Add  -4308.761     E 9  687.4466 Accepted
##      Customer.type          Add  -4793.471   VEV 6  980.6843 Accepted
##                Tax          Add -12661.307   VEV 5 -778.1974 Rejected
##      Customer.type       Remove  -4192.156     E 9  864.0795 Rejected
##
## Selected subset: Quantity, Customer.type
```

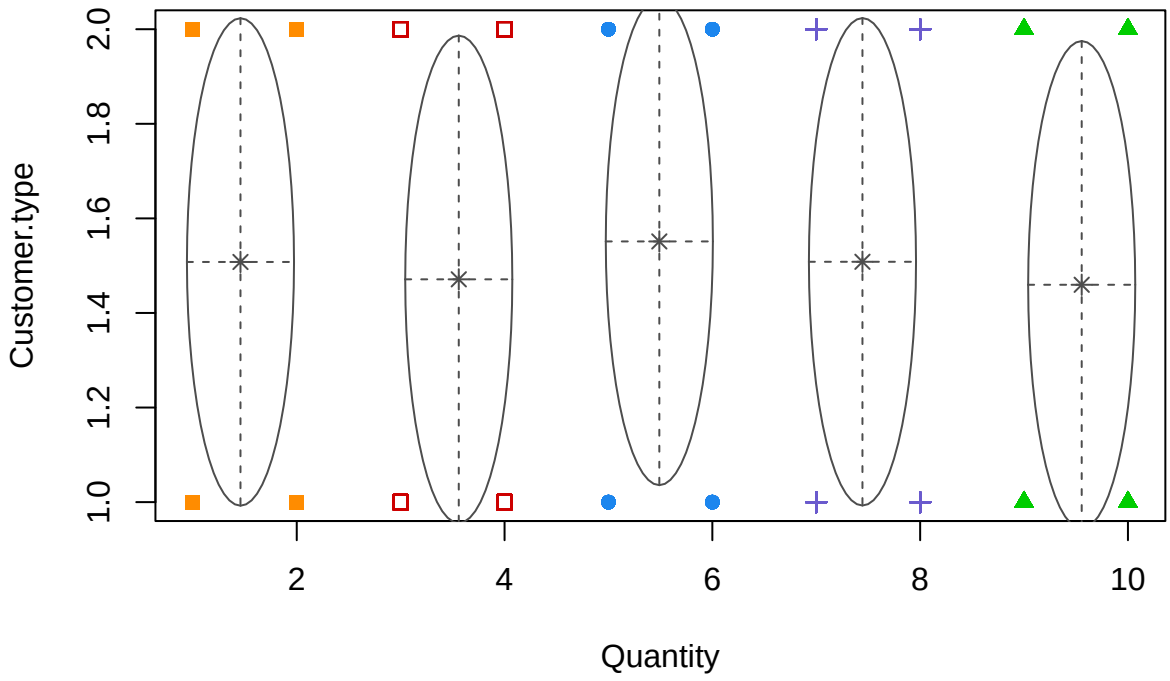
We find that subsets used for clustering consists of Quantity and customer type subsets

```
# Building a clustering model
#
subset <- data_num[,data_out$subset]

model <- Mclust(subset)

summary(model)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EII (spherical, equal volume) model with 5 components:
##
##  log-likelihood    n df       BIC       ICL
##       -3038.703 1000 15 -6181.022 -6215.724
##
## Clustering table:
##   1   2   3   4   5
## 200 199 211 187 203
```

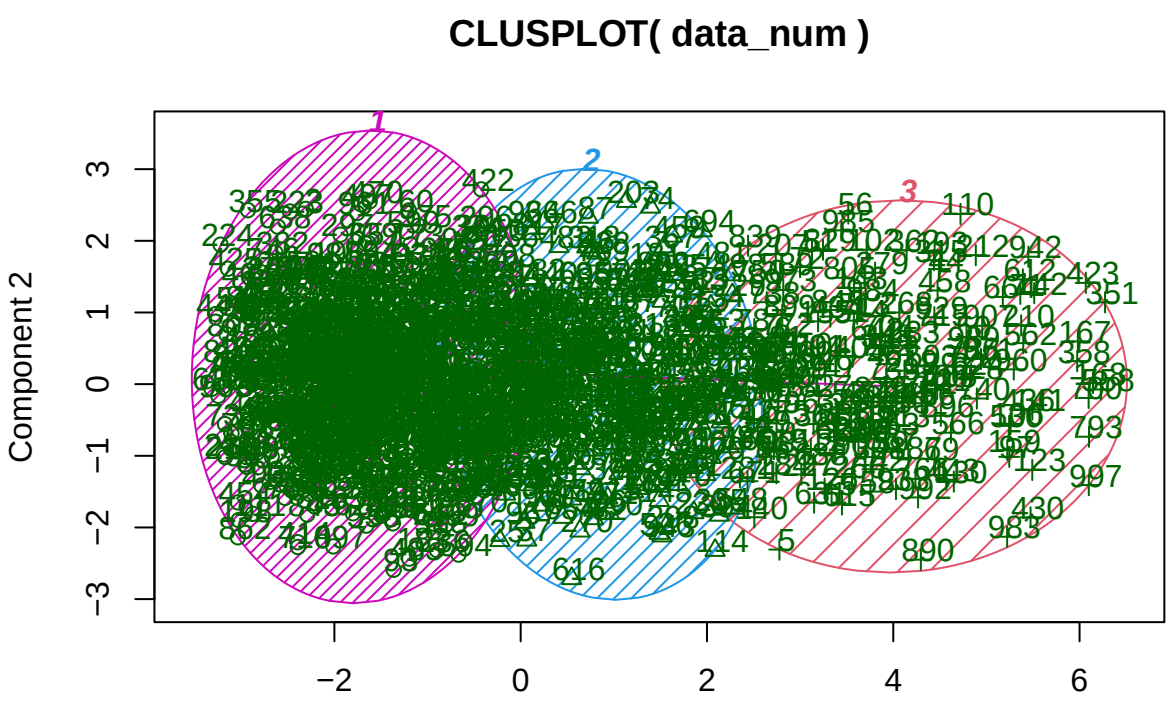
The clusters contain upto 5 clusters

```
# Plotting the clusters
#

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



3. Embedded Method

```
# Creating the model
#
set.seed(25)
model <- ewkm(data_num, 3, lambda=2, maxiter=1000)

# plotting the clusters
#
clusplot(data_num, model$cluster, color=TRUE, shade=TRUE,
         labels=2, lines=1)
```




```
# Calculating the weights of the 3 clusters and variables
#
weights <- round(model$weights*100,2)
weights
```

```
##   Branch Customer.type Product.line Unit.price Quantity Tax Payment cogs
## 1      0         99.99            0          0        0   0       0    0
## 2      0         99.99            0          0        0   0       0    0
## 3      0         99.99            0          0        0   0       0    0
##   gross.income Rating Total
## 1            0      0     0
## 2            0      0     0
## 3            0      0     0
```

4. Feature ranking

```
# previewing the data
#
head(data_num)
```

```
##   Branch Customer.type Product.line Unit.price Quantity     Tax Payment   cogs
## 1      1             1            4      74.69        7 26.1415       3 522.83
```

```
## 2      3             2           1      15.28        5  3.8200       1  76.40
## 3      1             2           5      46.33        7 16.2155       2 324.31
## 4      1             1           4      58.22        8 23.2880       3 465.76
## 5      1             2           6      86.31        7 30.2085       3 604.17
## 6      3             2           1      85.39        7 29.8865       3 597.73
##   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
```

```
# evaluating wih correlation coeffecient
#
coeff <- linear.correlation(Total~., data_num)
coeff
```

```
##               attr_importance
## Branch             0.04104666
## Customer.type      0.01967028
## Product.line       0.03162072
## Unit.price         0.63396209
## Quantity           0.70551019
## Tax                1.00000000
## Payment            0.01243364
## cogs               1.00000000
## gross.income       1.00000000
## Rating             0.03644170
```

```
# getting the top 5 performing variables
#

top <- cutoff.k(coeff, 5)
as.data.frame(top)
```

```
##            top
## 1          Tax
## 2         cogs
## 3 gross.income
## 4     Quantity
## 5   Unit.price
```