Dimensionality Reduction

1. Problem Definition.

Carrefour Kenya is a part of the French multination Carrefour S.A. The company specialises in retail, operating in several shopping malls in the country including The Hub, Two Rivers and THika Road mall among others. The company also operates using several e-retail platforms mainly the company's website as well as through Jumia market place.

The retail seeks to roll out several marketing strategies to increase their sales numbers. The marketing team is seeking the help of the company's data scientist to assist in determining which are the best strategies to employ that will result in the highest number of sales.

2. Data Sourcing

The data used for the analysis was sourced fromhere The data contains a list of transaction from various customers, it includes the type of customer, whether a Normal customer or a Member, their gender, product line they purchased, quantities purchased, etc.

3. Loading and Checking the Data

```
carre =  read.csv("http://bit.ly/CarreFourDataset")

head(carre)
```

```
##    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
```

```
tail(carre)
```

```
##       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
```

```
dim(carre)
```

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

Our dataset is made up of 16 columns and 1000 rows

```
# checking the data type as we will need to change some when performing dimensionality reduction
sapply(carre, class)
```

```
##              Invoice.ID                  Branch           Customer.type
##                "factor"                "factor"                "factor"
##                  Gender            Product.line              Unit.price
##                "factor"                "factor"               "numeric"
##                Quantity                     Tax                    Date
##               "integer"               "numeric"                "factor"
##                    Time                 Payment                    cogs
##                "factor"                "factor"               "numeric"
## gross.margin.percentage            gross.income                  Rating
##               "numeric"               "numeric"               "numeric"
##                   Total
##               "numeric"
```

4. Data Cleaning

Checking for and Dealing with Null Values

```
colSums(is.na(carre))
```

```
##              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
```

Our dataset contains no null values.

Checking for and Dealing with Duplicates

```
duplicates = carre[duplicated(carre),]
dim(duplicates)
```

```
## [1]  0 16
```

Our dataset contains no duplicate values.

5. Dimensionality Reduction

PCA

Data Preparation

```
library(CatEncoders)
```

```
## Warning: package 'CatEncoders' was built under R version 3.6.3
```

```
##
## Attaching package: 'CatEncoders'
```

```
## The following object is masked from 'package:base':
##
##     transform
```

```
#Changing categorical data to numerical data
factors <- names(which(sapply(carre, is.factor)))
# Label Encoder
for (i in factors){
encode <- LabelEncoder.fit(carre[, i])
carre[, i] <- transform(encode, carre[, i])
}
```

```
#checking whether all columns have changed to the correct data types

sapply(carre, class)
```

```
##              Invoice.ID                Branch         Customer.type
##               "integer"             "integer"             "integer"
##                  Gender          Product.line            Unit.price
##               "integer"             "integer"             "numeric"
##                Quantity                   Tax                  Date
##               "integer"             "numeric"             "integer"
##                    Time               Payment                  cogs
##               "integer"             "integer"             "numeric"
## gross.margin.percentage        gross.income                Rating
##               "numeric"             "numeric"             "numeric"
##                   Total
##               "numeric"
```

```
#dropping the invoice ID column

carre$Invoice.ID <- NULL
```

```
#chaecking our new dataset

head(carre)
```

```
##   Branch Customer.type Gender Product.line Unit.price Quantity     Tax Date
## 1      1             1      1            4      74.69        7 26.1415   27
## 2      3             2      1            1      15.28        5  3.8200   88
## 3      1             2      2            5      46.33        7 16.2155   82
## 4      1             1      2            4      58.22        8 23.2880   20
## 5      1             2      2            6      86.31        7 30.2085   58
## 6      3             2      2            1      85.39        7 29.8865   77
##   Time Payment   cogs gross.margin.percentage gross.income Rating    Total
## 1  147       3 522.83                4.761905      26.1415    9.1 548.9715
## 2   24       1  76.40                4.761905       3.8200    9.6  80.2200
## 3  156       2 324.31                4.761905      16.2155    7.4 340.5255
## 4  486       3 465.76                4.761905      23.2880    8.4 489.0480
## 5   30       3 604.17                4.761905      30.2085    5.3 634.3785
## 6  394       3 597.73                4.761905      29.8865    4.1 627.6165
```

```
# carrying out our PCA to check for
carre.pca <- prcomp(carre, center = TRUE, scale. = FALSE)

# previewing our PCA summary
summary(carre.pca)
```

```
## Importance of components:
##                             PC1      PC2      PC3      PC4     PC5     PC6
## Standard deviation     340.384 147.0671 25.88456 20.51561 1.73430 1.69601
## Proportion of Variance   0.836   0.1561  0.00483  0.00304 0.00002 0.00002
## Cumulative Proportion    0.836   0.9921  0.99690  0.99993 0.99995 0.99998
##                             PC7     PC8    PC9   PC10  PC11      PC12      PC13
## Standard deviation     1.24606 0.84341 0.7999 0.5068 0.487 7.832e-14 1.527e-14
## Proportion of Variance 0.00001 0.00001 0.0000 0.0000 0.000 0.000e+00 0.000e+00
## Cumulative Proportion  0.99999 0.99999 1.0000 1.0000 1.000 1.000e+00 1.000e+00
##                            PC14      PC15
## Standard deviation     8.532e-16 6.292e-31
```

```
## Proportion of Variance 0.000e+00 0.000e+00
## Cumulative Proportion  1.000e+00 1.000e+00
```

```
str(carre.pca)
```

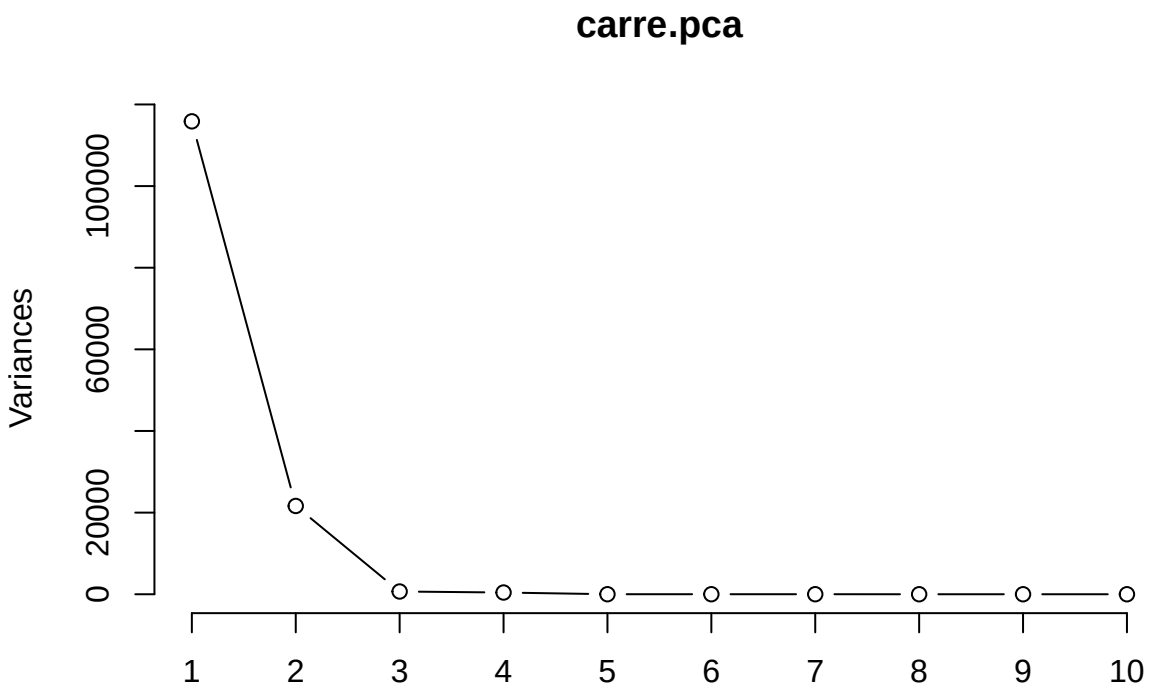
```
## List of 5
##  $ sdev    : num [1:15] 340.38 147.07 25.88 20.52 1.73 ...
##  $ rotation: num [1:15, 1:15] 9.85e-05 -2.89e-05 -7.26e-05 1.60e-04 4.95e-02 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:15] "Branch" "Customer.type" "Gender" "Product.line" ...
##   .. ..$ : chr [1:15] "PC1" "PC2" "PC3" "PC4" ...
##  $ center  : Named num [1:15] 1.99 1.5 1.5 3.45 55.67 ...
##   ..- attr(*, "names")= chr [1:15] "Branch" "Customer.type" "Gender" "Product.line" ...
##  $ scale   : logi FALSE
##  $ x       : num [1:1000, 1:15] 313 -337 24 229 432 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:15] "PC1" "PC2" "PC3" "PC4" ...
##  - attr(*, "class")= chr "prcomp"
```

```
# Installing our ggbiplot visualisation package
#
library(devtools)
```

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

```
#visualising our PCA


plot(carre.pca, type="l")
```



t-Distributed Stochastic Neighbor Embedding (t-SNE)

```
library(Rtsne)
```

```
## Warning: package 'Rtsne' was built under R version 3.6.3
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
library(lattice)
```

Trying t-SNE with Branch

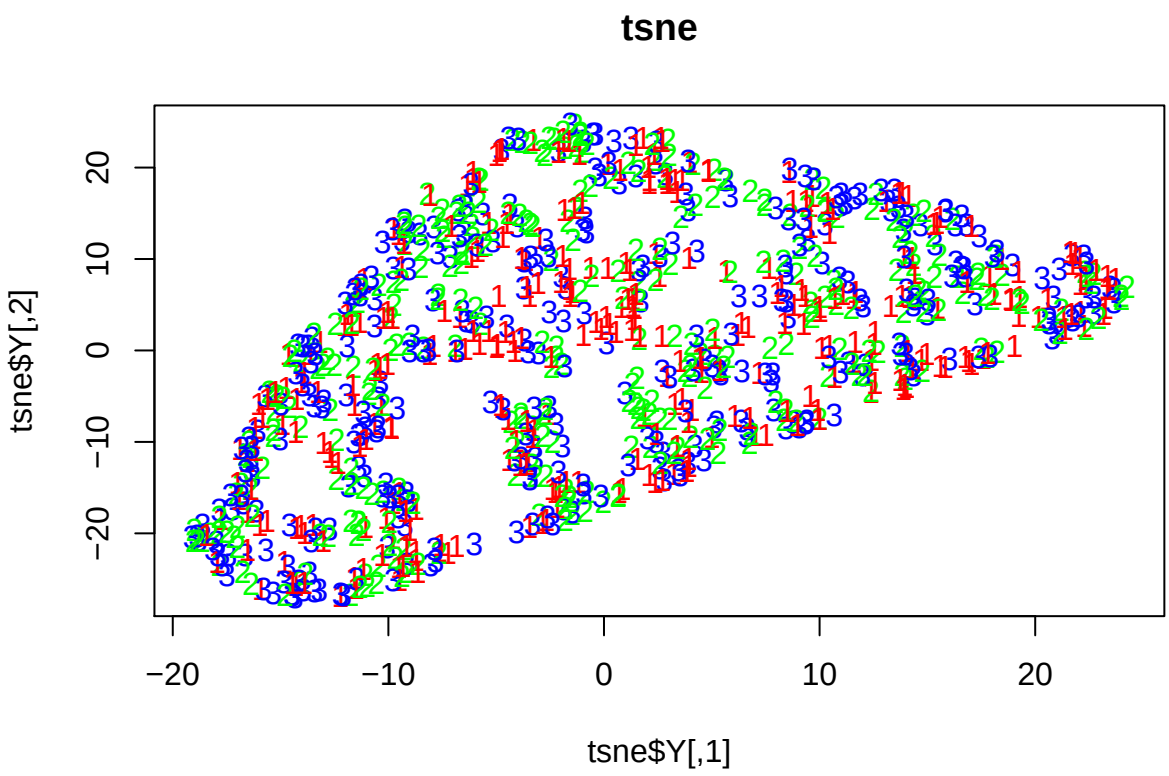
```
carre.branch <- carre
carre.branch$Branch <- as.factor(carre.branch$Branch)
```

```r
colors = rainbow(length(unique(carre.branch$Branch)))
names(colors) = unique(carre.branch$Branch)
```

```r
tsne <- Rtsne(carre[,-1], dims = 2, perplexity=30, verbose=TRUE, max_iter = 500)
```

```
## Performing PCA
## Read the 1000 x 14 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 0.19 seconds (sparsity = 0.103660)!
## Learning embedding...
## Iteration 50: error is 64.385102 (50 iterations in 0.21 seconds)
## Iteration 100: error is 57.876485 (50 iterations in 0.13 seconds)
## Iteration 150: error is 57.615991 (50 iterations in 0.13 seconds)
## Iteration 200: error is 57.590925 (50 iterations in 0.12 seconds)
## Iteration 250: error is 57.586959 (50 iterations in 0.12 seconds)
## Iteration 300: error is 0.752410 (50 iterations in 0.11 seconds)
## Iteration 350: error is 0.609841 (50 iterations in 0.12 seconds)
## Iteration 400: error is 0.579446 (50 iterations in 0.12 seconds)
## Iteration 450: error is 0.566818 (50 iterations in 0.11 seconds)
## Iteration 500: error is 0.558625 (50 iterations in 0.13 seconds)
## Fitting performed in 1.30 seconds.
```

```r
plot(tsne$Y, t='n', main="tsne")
text(tsne$Y, labels=carre$Branch, col=colors[carre$Branch])
```



Trying T-SNE with Product Line

```
carre.prod <- carre
carre.prod$Product.line <- as.factor(carre.prod$Product.line)

# for plotting
colors = rainbow(length(unique(carre.prod$Product.line)))
names(colors) = unique(carre.prod$Product.line)

tsne <- Rtsne(carre[, -4], dims = 2, perplexity = 30, verbose = TRUE,
              max_iter = 500)
```

```
## Performing PCA
## Read the 1000 x 14 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 0.17 seconds (sparsity = 0.103656)!
## Learning embedding...
## Iteration 50: error is 63.981024 (50 iterations in 0.17 seconds)
## Iteration 100: error is 57.979898 (50 iterations in 0.17 seconds)
## Iteration 150: error is 57.710602 (50 iterations in 0.16 seconds)
## Iteration 200: error is 57.681837 (50 iterations in 0.15 seconds)
## Iteration 250: error is 57.677603 (50 iterations in 0.17 seconds)
## Iteration 300: error is 0.753584 (50 iterations in 0.13 seconds)
## Iteration 350: error is 0.610855 (50 iterations in 0.12 seconds)
```

```
## Iteration 400: error is 0.575060 (50 iterations in 0.13 seconds)
## Iteration 450: error is 0.562386 (50 iterations in 0.13 seconds)
## Iteration 500: error is 0.555600 (50 iterations in 0.13 seconds)
## Fitting performed in 1.45 seconds.
```

```r
# plotting our graph
plot(tsne$Y, t = 'n', main = 'tsne')
text(tsne$Y, labels = carre.prod$Product.line,
col = colors[carre.prod$Product.line])
```