Dimensionality Reduction

Stacy

2022-04-03

```
# This section of the project entails reducing your dataset to a low dimensional dataset using the t-SN
```

```
# The metric for success:

# The context:1. reducing your dataset to a low dimensional dataset
#using the t-SNE algorithm or PCA
#

# Experimental design taken:
## Problem Definition
## Data Sourcing
## Check the Data
## Perform Data Cleaning
## Perform Exploratory Data Analysis(Univariate, Bivariate & Multivariate)
## Implement the Solution

# The appropriateness of the available data to answer the given question:The data provided information
```

```
library(Rtsne)
library(tsne)
library(plotly)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'plotly'
```

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

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

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
data1<- read.csv('http://bit.ly/CarreFourDataset')
head(data1)
```

```
##    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
```

```
colnames(data1)
```

```
##  [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"
```

```
summary(data1)
```

```
##   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
```

```
# Curating the database for analysis
Labels<-data1$Product.line
data1$label<-as.factor(data1$Product.line)
```

```
dim(data1)
```

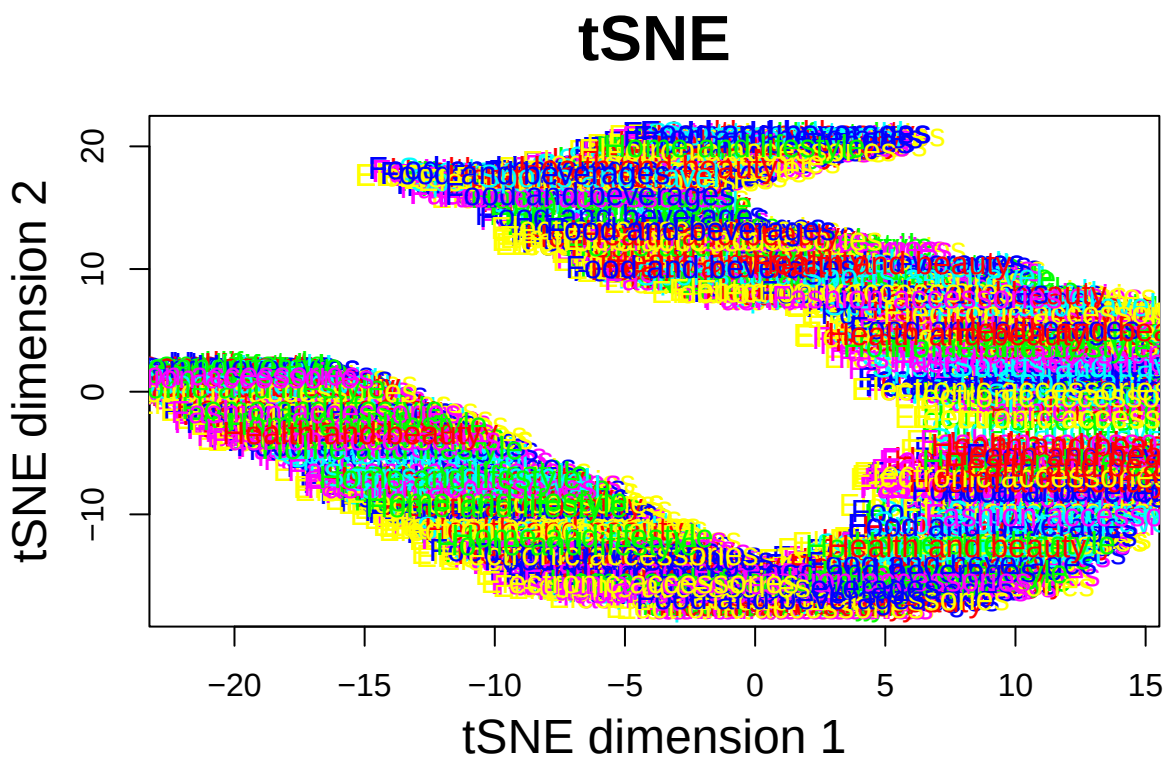
```
## [1] 1000   17
```

```
# shrinking the size for the time limit
numTrain <- 500
set.seed(1)
rows <- sample(1:nrow(data1), numTrain)
train <- data1[rows,]
```

```
# using tsne
set.seed(1) # for reproducibility
tsne <- Rtsne(train[,-1], dims = 2, perplexity=30, verbose=TRUE, max_iter = 500)
```

```
## Performing PCA
## Read the 500 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 0.13 seconds (sparsity = 0.205992)!
## Learning embedding...
## Iteration 50: error is 53.783496 (50 iterations in 0.08 seconds)
## Iteration 100: error is 48.390613 (50 iterations in 0.08 seconds)
## Iteration 150: error is 47.341933 (50 iterations in 0.07 seconds)
## Iteration 200: error is 47.045770 (50 iterations in 0.08 seconds)
## Iteration 250: error is 46.934092 (50 iterations in 0.08 seconds)
## Iteration 300: error is 0.313141 (50 iterations in 0.07 seconds)
## Iteration 350: error is 0.247808 (50 iterations in 0.07 seconds)
## Iteration 400: error is 0.232625 (50 iterations in 0.07 seconds)
## Iteration 450: error is 0.226848 (50 iterations in 0.07 seconds)
## Iteration 500: error is 0.222379 (50 iterations in 0.07 seconds)
## Fitting performed in 0.74 seconds.
```

```r
# visualizing
colors = rainbow(length(unique(data1$Product.line)))
names(colors) = unique(data1$Product.line)
par(mgp=c(2.5,1,0))
plot(tsne$Y, t='n', main="tSNE", xlab="tSNE dimension 1", ylab="tSNE dimension 2", "cex.main"=2, "cex.la
text(tsne$Y, labels=data1$Product.line, col=colors[data1$Product.line])
```




```r
# Conclusion
# tsne display of the clusters of the product.line .
```