# Carrefour Sales Data

library(readr)  
Supermarket\_Dataset\_1\_Sales\_Data <- read\_csv("C:/Users/mutho/Downloads/Supermarket\_Dataset\_1 - Sales Data.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## `Invoice ID` = col\_character(),  
## Branch = col\_character(),  
## `Customer type` = col\_character(),  
## Gender = col\_character(),  
## `Product line` = col\_character(),  
## `Unit price` = col\_double(),  
## Quantity = col\_double(),  
## Tax = col\_double(),  
## Date = col\_character(),  
## Time = col\_time(format = ""),  
## Payment = col\_character(),  
## cogs = col\_double(),  
## `gross margin percentage` = col\_double(),  
## `gross income` = col\_double(),  
## Rating = col\_double(),  
## Total = col\_double()  
## )

View(Supermarket\_Dataset\_1\_Sales\_Data)

Sales\_data <- Supermarket\_Dataset\_1\_Sales\_Data  
View(Sales\_data)

Dimensionality reduction

library(Rtsne)

## Warning: package 'Rtsne' was built under R version 4.0.5

# Curating the database for analysis   
#   
Labels<-Sales\_data$Total  
Sales\_data$Total<-as.factor(Sales\_data$Total)  
  
# For plotting  
#  
colors = rainbow(length(unique(Sales\_data$Total)))  
names(colors) = unique(Sales\_data$Total)

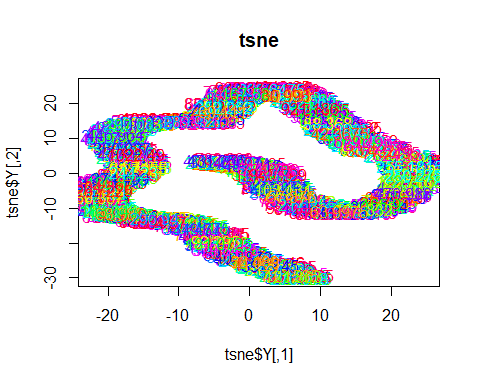
# Executing the algorithm on curated data  
#   
tsne <- Rtsne(Sales\_data[,-1], dims = 2, perplexity=30, verbose=TRUE, max\_iter = 500)

## 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 0.19 seconds (sparsity = 0.095300)!  
## Learning embedding...  
## Iteration 50: error is 57.705827 (50 iterations in 0.12 seconds)  
## Iteration 100: error is 49.741636 (50 iterations in 0.09 seconds)  
## Iteration 150: error is 47.924510 (50 iterations in 0.13 seconds)  
## Iteration 200: error is 47.066225 (50 iterations in 0.12 seconds)  
## Iteration 250: error is 46.642693 (50 iterations in 0.13 seconds)  
## Iteration 300: error is 0.491595 (50 iterations in 0.15 seconds)  
## Iteration 350: error is 0.320889 (50 iterations in 0.12 seconds)  
## Iteration 400: error is 0.288925 (50 iterations in 0.11 seconds)  
## Iteration 450: error is 0.272773 (50 iterations in 0.12 seconds)  
## Iteration 500: error is 0.260906 (50 iterations in 0.11 seconds)  
## Fitting performed in 1.20 seconds.

# Getting the duration of execution  
#   
exeTimeTsne <- system.time(Rtsne(Sales\_data[,-1], 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 0.24 seconds (sparsity = 0.095300)!  
## Learning embedding...  
## Iteration 50: error is 58.189772 (50 iterations in 0.15 seconds)  
## Iteration 100: error is 50.008855 (50 iterations in 0.15 seconds)  
## Iteration 150: error is 48.190996 (50 iterations in 0.14 seconds)  
## Iteration 200: error is 47.301533 (50 iterations in 0.14 seconds)  
## Iteration 250: error is 46.681319 (50 iterations in 0.13 seconds)  
## Iteration 300: error is 0.492081 (50 iterations in 0.13 seconds)  
## Iteration 350: error is 0.327401 (50 iterations in 0.12 seconds)  
## Iteration 400: error is 0.288068 (50 iterations in 0.12 seconds)  
## Iteration 450: error is 0.271042 (50 iterations in 0.12 seconds)  
## Iteration 500: error is 0.258739 (50 iterations in 0.11 seconds)  
## Fitting performed in 1.29 seconds.

# Plotting our graph and closely examining the graph  
#   
plot(tsne$Y, t='n', main="tsne")  
text(tsne$Y, labels=Sales\_data$Total, col=colors[Sales\_data$Total])

 Feature Selection

#   
suppressWarnings(  
 suppressMessages(if  
 (!require(caret, quietly=TRUE))  
 install.packages("caret")))  
library(caret)

suppressWarnings(  
 suppressMessages(if  
 (!require(corrplot, quietly=TRUE))  
 install.packages("corrplot")))  
library(corrplot)

# Getting the numerical columns   
Sales <- Sales\_data[,6:8,12:16]

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

## integer(0)

names(Sales[,highlyCorrelated])

## character(0)