1. Defining the question

1.1 Specifying the data analytic objective

Part 1: Dimensionality Reduction

This section of the project entails reducing your dataset to a low dimensional dataset using the t-SNE algorithm or PCA. You will be required to perform your analysis and provide insights gained from your analysis.

Part 2: Feature Selection

This section requires you to perform feature selection through the use of the unsupervised learning methods learned earlier this week. You will be required to perform your analysis and provide insights on the features that contribute the most information to the dataset.

1.2 Defining the metric of success

Dataset with reduced dimensions.

1.3 Understanding the context

You are a Data analyst at Carrefour Kenya and are currently undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax). Your project has been divided into three parts where you'll explore a recent marketing dataset by performing various unsupervised learning techniques and later providing recommendations based on your insights.

1.4 Recording the Experimental Design

1. Loading the data
2. Checking the data
3. Tidying the data
4. EDA
5. Dimensionality Reduction
6. Feature Selection

```
# Installing packages
#install.packages("dplyr")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

#install.packages("tidyverse")
library(tidyverse)

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

## -- Attaching packages --------------------------------------- tidyverse
1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   1.4.0

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

#install.packages("ggplot2")
library(ggplot2)
#install.packages("devtools",dependencies=TRUE)
library(devtools)

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

## Loading required package: usethis

#install_github("vqv/ggbiplot")
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 object is masked from 'package:purrr':
##
##     compact
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## 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

#install.packages("arules")
library(arules)

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

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

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

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following objects are masked from 'package:base':
##
##     abbreviate, write

#install.packages("arulesViz")
library(arulesViz)

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

#install.packages("Rtsne")
library(Rtsne)

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

```
#install.packages("anomalize")
library(anomalize)

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

## == Use anomalize to improve your Forecasts by 50%!
============================
## Business Science offers a 1-hour course - Lab #18: Time Series Anomaly
Detection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-
pro </>

#install.packages("caret")
library(caret)

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

## Loading required package: lattice

##
## Attaching package: 'caret'

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

#install.packages("corrplot")
library(corrplot)

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

## corrplot 0.84 loaded
```

3. Loading the data

```
# load the dataset
df <- read_csv("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()
## )

spaceless <- function(x) {colnames(x) <- gsub(" ", "_", colnames(x));x}
df <- spaceless(df)

#view the head

head(df)

## # A tibble: 6 x 16
##   Invoice_ID  Branch Customer_type Gender Product_line Unit_price Quantity
Tax
##   <chr>       <chr>  <chr>         <chr>  <chr>             <dbl>    <dbl>
<dbl>
## 1 750-67-8428 A      Member        Female Health and ~       74.7        7
26.1
## 2 226-31-3081 C      Normal        Female Electronic ~       15.3        5
3.82
## 3 631-41-3108 A      Normal        Male   Home and li~       46.3        7
16.2
## 4 123-19-1176 A      Member        Male   Health and ~       58.2        8
23.3
## 5 373-73-7910 A      Normal        Male   Sports and ~       86.3        7
30.2
## 6 699-14-3026 C      Normal        Male   Electronic ~       85.4        7
29.9
## # ... with 8 more variables: Date <chr>, Time <time>, Payment <chr>,
## #   cogs <dbl>, gross_margin_percentage <dbl>, gross_income <dbl>,
## #   Rating <dbl>, Total <dbl>

#structure of the dataset

str(df)

## spec_tbl_df[,16] [1,000 x 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Invoice_ID            : chr [1:1000] "750-67-8428" "226-31-3081" "631-
41-3108" "123-19-1176" ...
##  $ Branch                : chr [1:1000] "A" "C" "A" "A" ...
##  $ Customer_type         : chr [1:1000] "Member" "Normal" "Normal"
"Member" ...
##  $ Gender                : chr [1:1000] "Female" "Female" "Male" "Male"
...
##  $ Product_line          : chr [1:1000] "Health and beauty" "Electronic
accessories" "Home and lifestyle" "Health and beauty" ...
##  $ Unit_price            : num [1:1000] 74.7 15.3 46.3 58.2 86.3 ...
```

```
##  $ Quantity               : num [1:1000] 7 5 7 8 7 7 6 10 2 3 ...
##  $ Tax                    : num [1:1000] 26.14 3.82 16.22 23.29 30.21 ...
##  $ Date                   : chr [1:1000] "1/5/2019" "3/8/2019" "3/3/2019"
"1/27/2019" ...
##  $ Time                   : 'hms' num [1:1000] 13:08:00 10:29:00 13:23:00
20:33:00 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Payment                : chr [1:1000] "Ewallet" "Cash" "Credit card"
"Ewallet" ...
##  $ cogs                   : num [1:1000] 522.8 76.4 324.3 465.8 604.2 ...
##  $ gross_margin_percentage: num [1:1000] 4.76 4.76 4.76 4.76 4.76 ...
##  $ gross_income           : num [1:1000] 26.14 3.82 16.22 23.29 30.21 ...
##  $ Rating                 : num [1:1000] 9.1 9.6 7.4 8.4 5.3 4.1 5.8 8 7.2
5.9 ...
##  $ Total                  : num [1:1000] 549 80.2 340.5 489 634.4 ...
##  - attr(*, "spec")=
##   .. 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()
##   .. )
```

```
#check the dimensions of the dataset
dim(df)
```

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

```
#summary statistics
summary(df)
```

```
##   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   Class1:hms        Class :character   1st Qu.:118.50
##  Mode  :character   Class2:difftime   Mode  :character   Median :241.76
##                     Mode  :numeric                        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
```

```
# Concatenate the columns data and time

df$Date_time = paste(df$Date,df$Time)

# Converting the column into string
df$Date_time = as.character(df$Date_time)

# converting the Date_time column data type to date time

df$Date_time = strptime(df$Date_time, "%m/%d/%Y %M:%S")

# checking if the data type has been converted

str(df$Date_time)
```

```
##  POSIXlt[1:1000], format: "2019-01-05 00:13:08" "2019-03-08 00:10:29"
"2019-03-03 00:13:23" ...
```

4. Data Cleaning

```
# Checking for Missing Values

colSums(is.na(df))
```

```
##               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               Date_time
##                        0                         0
```

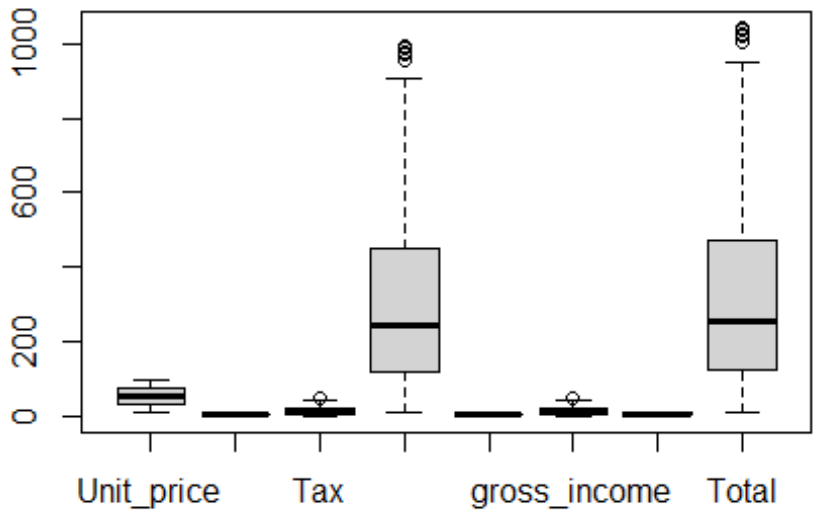
There are no missing values

```
# Checking for duplicated data

duplicates <- df[duplicated(df),]

#duplicates

anyDuplicated(df)

## [1] 0
```

There are no duplicated data

```
head(df)

## # A tibble: 6 x 17
##   Invoice_ID  Branch Customer_type Gender Product_line Unit_price Quantity
Tax
##   <chr>       <chr>  <chr>         <chr>  <chr>             <dbl>    <dbl>
<dbl>
## 1 750-67-8428 A      Member        Female Health and ~       74.7        7
26.1
## 2 226-31-3081 C      Normal        Female Electronic ~       15.3        5
3.82
## 3 631-41-3108 A      Normal        Male   Home and li~       46.3        7
16.2
## 4 123-19-1176 A      Member        Male   Health and ~       58.2        8
23.3
## 5 373-73-7910 A      Normal        Male   Sports and ~       86.3        7
30.2
## 6 699-14-3026 C      Normal        Male   Electronic ~       85.4        7
29.9
## # ... with 9 more variables: Date <chr>, Time <time>, Payment <chr>,
## #   cogs <dbl>, gross_margin_percentage <dbl>, gross_income <dbl>,
## #   Rating <dbl>, Total <dbl>, Date_time <dttm>
```

```r
# Plot a boxplot to help us visualise any existing outliers
numerical_col<-df[,  c(6, 7, 8, 12, 13, 14, 15, 16)]
boxplot(numerical_col)
```




```r
# Dropping unnecessary column

df <- df[,c(-1,-9,-10,-13)]
head(df)
```

```
## # A tibble: 6 x 13
##   Branch Customer_type Gender Product_line    Unit_price Quantity   Tax
Payment
##   <chr>  <chr>         <chr>  <chr>                <dbl>    <dbl> <dbl>
<chr>
## 1 A      Member        Female Health and bea~       74.7        7 26.1
Ewallet
## 2 C      Normal        Female Electronic acc~       15.3        5  3.82
Cash
## 3 A      Normal        Male   Home and lifes~       46.3        7 16.2
Credit ~
## 4 A      Member        Male   Health and bea~       58.2        8 23.3
Ewallet
## 5 A      Normal        Male   Sports and tra~       86.3        7 30.2
Ewallet
## 6 C      Normal        Male   Electronic acc~       85.4        7 29.9
Ewallet
```

```
## # ... with 5 more variables: cogs <dbl>, gross_income <dbl>, Rating <dbl>,
## #   Total <dbl>, Date_time <dttm>
```

5. EDA

Univariate Analysis

```
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.0.5
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: Formula
```

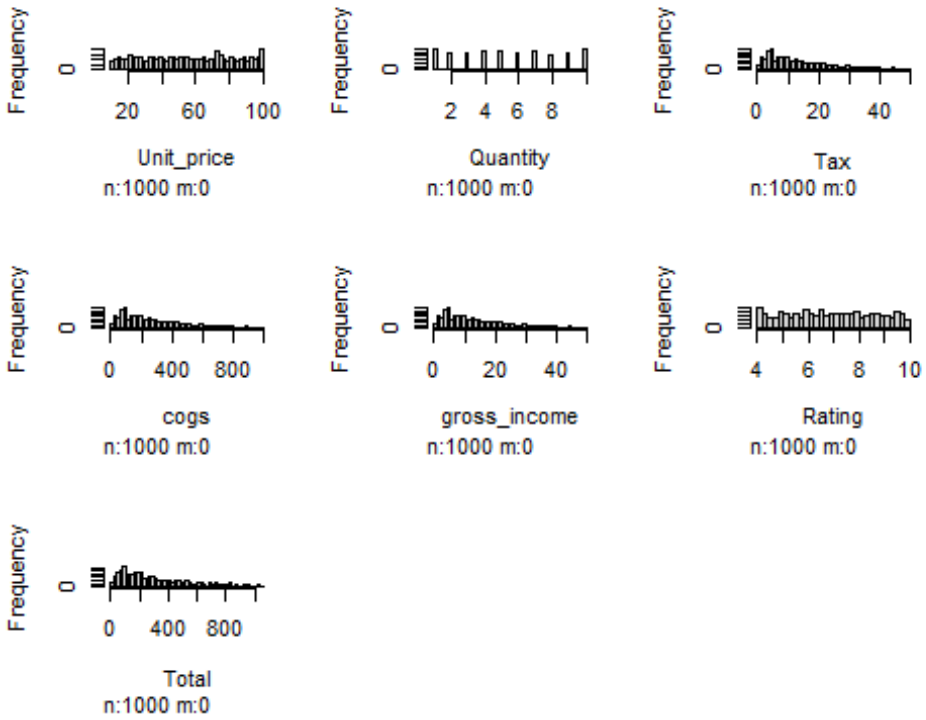
```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:plyr':
##
##     is.discrete, summarize
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
hist(numerical_col)
```



Part 1: Dimensionality Reduction

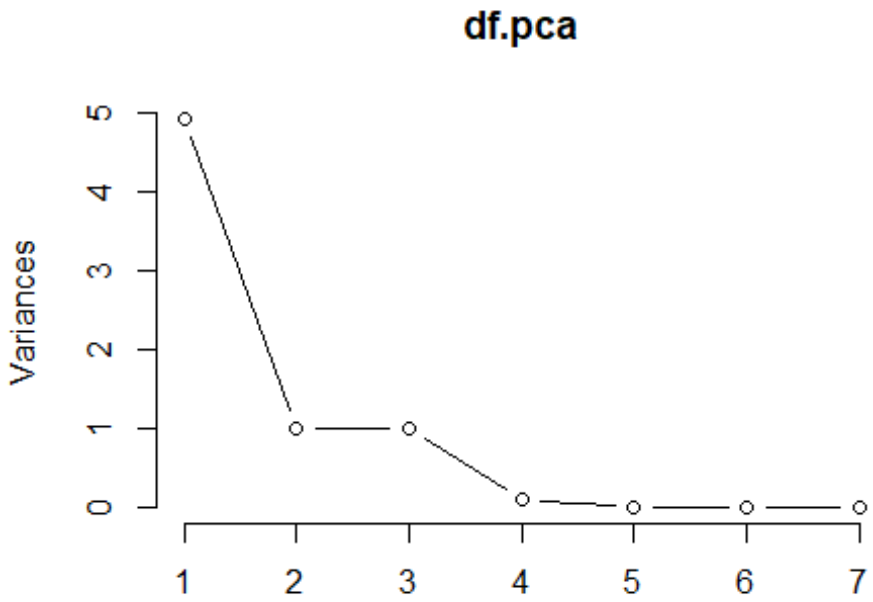
PCA

```
# Selecting the numerical data

numericals <- df[,c(5,6,7,9,10,11,12)]
head(numericals)
```

```
## # A tibble: 6 x 7
##   Unit_price Quantity   Tax  cogs gross_income Rating Total
##        <dbl>    <dbl> <dbl> <dbl>        <dbl>  <dbl> <dbl>
## 1       74.7        7 26.1  523.         26.1     9.1 549.
## 2       15.3        5  3.82  76.4         3.82    9.6  80.2
## 3       46.3        7 16.2  324.         16.2     7.4 341.
## 4       58.2        8 23.3  466.         23.3     8.4 489.
## 5       86.3        7 30.2  604.         30.2     5.3 634.
## 6       85.4        7 29.9  598.         29.9     4.1 628.
```

```
# We then pass df to the prcomp(). We also set two arguments, center and
scale,
# to be TRUE then preview our object with summary
# ---
#
df.pca <- prcomp(df[,c(5,6,7,9:12)], center = TRUE, scale. = TRUE)
summary(df.pca)
```

```
## Importance of components:
##                             PC1    PC2    PC3     PC4       PC5       PC6
## Standard deviation     2.2185 1.0002 0.9939 0.30001 2.981e-16 1.493e-16
## Proportion of Variance 0.7031 0.1429 0.1411 0.01286 0.000e+00 0.000e+00
## Cumulative Proportion  0.7031 0.8460 0.9871 1.00000 1.000e+00 1.000e+00
##                               PC7
## Standard deviation     9.831e-17
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00

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



Part 2: Feature Selection

```
# Calculating the correlation matrix
# ---
#
correlationMatrix <- cor(numericals)

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

# Highly correlated attributes
# ---
```

```
#
highlyCorrelated
```

```
## [1] 4 7 3
```

```
names(numericals[,highlyCorrelated])
```

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

```
# Removing Redundant Features
# ---
#
drop <-numericals[-highlyCorrelated]
```