Part 1: Dimensionality Reduction

VICTOR

6/5/2021

Dimensionality Reduction

a) Data Analytic Question

The aim of this project is to reduce the dataset to a low dimensional dataset via the t-SNE algorithm or PCA.

b) Success Metrics

- Successful Loading the data.
- Successful Handling missing data.
- Successful Outliers detection.
- Successful Outlier Visualization.
- Successful Handling outliers.
- Successful Univariate analysis.
- Successful Bivariate analysis.

c) Context

undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest number of sales.

d) Data Understanding

Variables

- The dataset consists of 8 numerical and 8 categorical attributes.
- Invoice.ID
- Branch
- Customer.type
- Gender
- Product.line
- Unit.price
- Quantity
- Tax
- Date
- Time
- Payment
- cogs
- gross.margin.percentage
- gross.income
- Rating
- Total

e) Experimental Design

- Formulation of the research question.
- Data Sourcing
- Check the Data
- Perform Data Cleaning
- Perform Exploratory Data Analysis (Univariate, Bivariate & Multivariate)
- Implement the Solution
- Challenging the Solution
- Follow up Questions

Data Importation

```
dataset1<- read.csv("http://bit.ly/CarreFourDataset",header =T)
```

converting data.frame data into data.table

```
dataset1<-as.data.table(dataset1)
class(dataset1) #checking class
```

```
## [1] "data.table" "data.frame"
```

Data Columns

```
kable(colnames(dataset1))
```

x
Invoice.ID
Branch
Customer.type
Gender
Product.line
Unit.price
Quantity
Tax
Date
Time
Payment
cogs
gross.margin.percentage
gross.income
Rating
Total

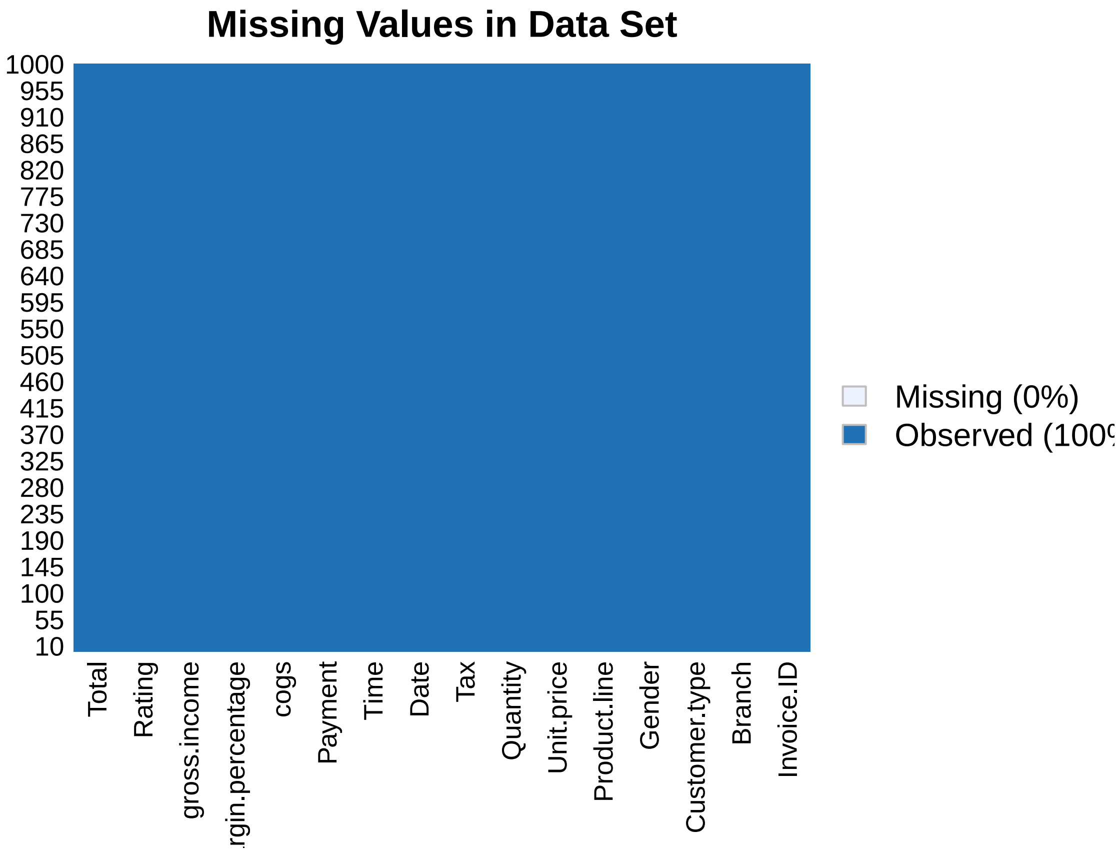
Check for missing values

```
library(Amelia)
```

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.8.0, built: 2021-05-26)
## ## Copyright (C) 2005-2021 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
missmap(dataset1,main="Missing Values in Data Set")
```



```
#colSums(is.na(dataset1))
```

any NAs in data set?

```
colSums(is.na(dataset1))
```

```
##              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
```

Now lets find the duplicated rows in the dataset df and assign to a variable duplicated_rows below.

```
duplicated_rows <- dataset1[duplicated(dataset1),]
#Lets print out the variable duplicated_rows and see these duplicated rows
#kable(duplicated_rows)
```

Removing these duplicated rows in the data set or showing these unique items and assigning to a variable unique_items below

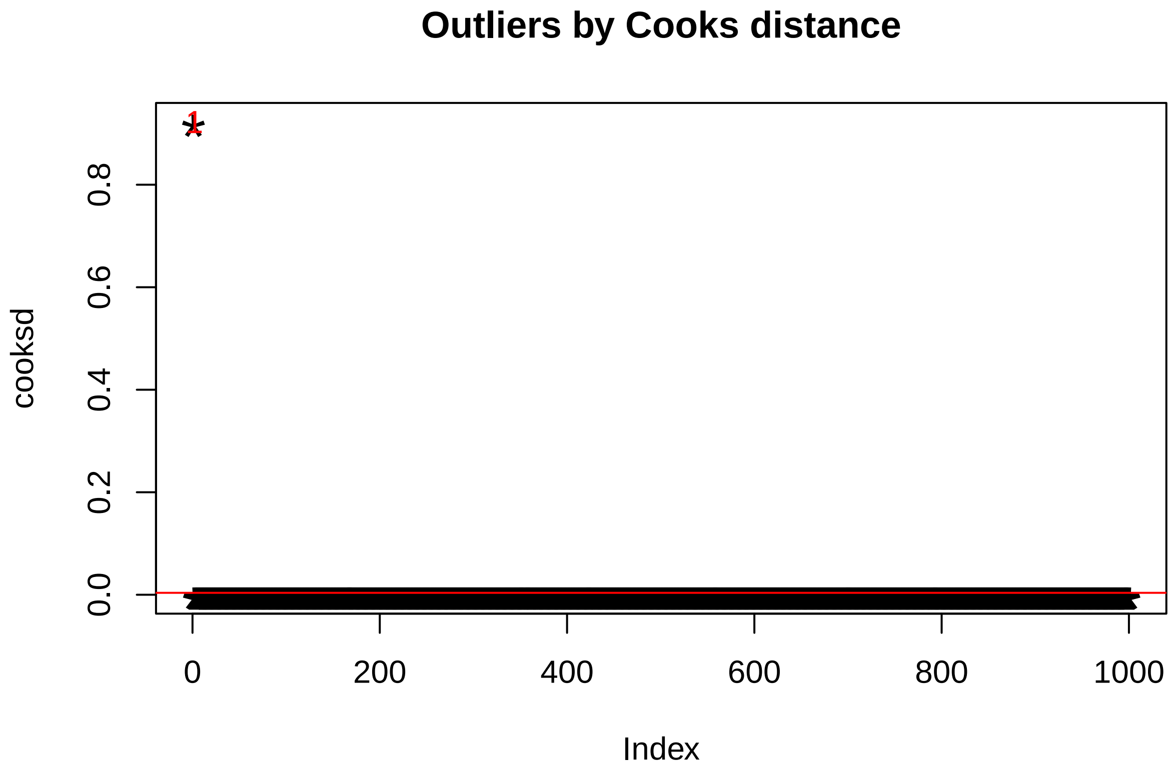
```
unique_items <- dataset1[!duplicated(dataset1), ]
```

Drop unnecessary column

```
dataset1 <- subset( dataset1, select = -Invoice.ID )
```

Outlier Treatment

```
mod <- lm( gross.margin.percentage~gross.income, data=dataset1)
cooksd <- cooks.distance(mod)

#Influence measures
#In general use, those observations that have a cook’s distance greater than 4 times
#the mean may be classified as Outlier


plot(cooksd, pch="*", cex=2, main="Outliers by Cooks distance")  # plot cook's distance
abline(h = 4*mean(cooksd, na.rm=T), col="red")  # add cutoff line
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4*mean(cooksd, na.rm=T),names(cooksd),""), co
```



Tibbles

A tibble is a special kind of data.frame used by dplyr and other packages of the tidyverse. Tidyverse is a set of packages for data science that work in harmony because they share common data representations and API design. When a data.frame is turned into a tibble its class will change.

```
class(dataset1)
```

```
## [1] "data.table" "data.frame"
```

```
dataset1<- tbl_df(dataset1)
```

```
## Warning: `tbl_df()` was deprecated in dplyr 1.0.0.
## Please use `tibble::as_tibble()` instead.
```

```
class(dataset1)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Data Overview

```
## Rows: 1,000
## Columns: 15
## $ Branch                  <chr> "A", "C", "A", "A", "A", "C", "A", "C", "A", "~
## $ Customer.type           <chr> "Member", "Normal", "Normal", "Member", "Norma~
## $ Gender                  <chr> "Female", "Female", "Male", "Male", "Male", "M~
## $ Product.line            <chr> "Health and beauty", "Electronic accessories",~
## $ Unit.price              <dbl> 74.69, 15.28, 46.33, 58.22, 86.31, 85.39, 68.8~
## $ Quantity                <int> 7, 5, 7, 8, 7, 7, 6, 10, 2, 3, 4, 4, 5, 10, 10~
## $ Tax                     <dbl> 26.1415, 3.8200, 16.2155, 23.2880, 30.2085, 29~
## $ Date                    <chr> "1/5/2019", "3/8/2019", "3/3/2019", "1/27/2019~
## $ Time                    <chr> "13:08", "10:29", "13:23", "20:33", "10:37", "~
## $ Payment                 <chr> "Ewallet", "Cash", "Credit card", "Ewallet", "~
## $ cogs                    <dbl> 522.83, 76.40, 324.31, 465.76, 604.17, 597.73,~
## $ gross.margin.percentage <dbl> 4.761904762, 4.761904762, 4.761904762, 4.76190~
## $ gross.income            <dbl> 26.1415, 3.8200, 16.2155, 23.2880, 30.2085, 29~
## $ Rating                  <dbl> 9.1, 9.6, 7.4, 8.4, 5.3, 4.1, 5.8, 8.0, 7.2, 5~
## $ Total                   <dbl> 548.9715, 80.2200, 340.5255, 489.0480, 634.378~
```

Number of columns

```
## [1] 15
```

Dimesion

```
## [1] 1000   15
```

Columnames

```
##  [1] "Branch"                  "Customer.type"          
##  [3] "Gender"                  "Product.line"           
##  [5] "Unit.price"              "Quantity"               
##  [7] "Tax"                     "Date"                   
##  [9] "Time"                    "Payment"                
## [11] "cogs"                    "gross.margin.percentage"
## [13] "gross.income"            "Rating"                 
## [15] "Total"
```

Encoding Categorical Variables

```
## 
## Attaching package: 'encode'

## The following object is masked from 'package:forcats':
## 
##     as_factor
```

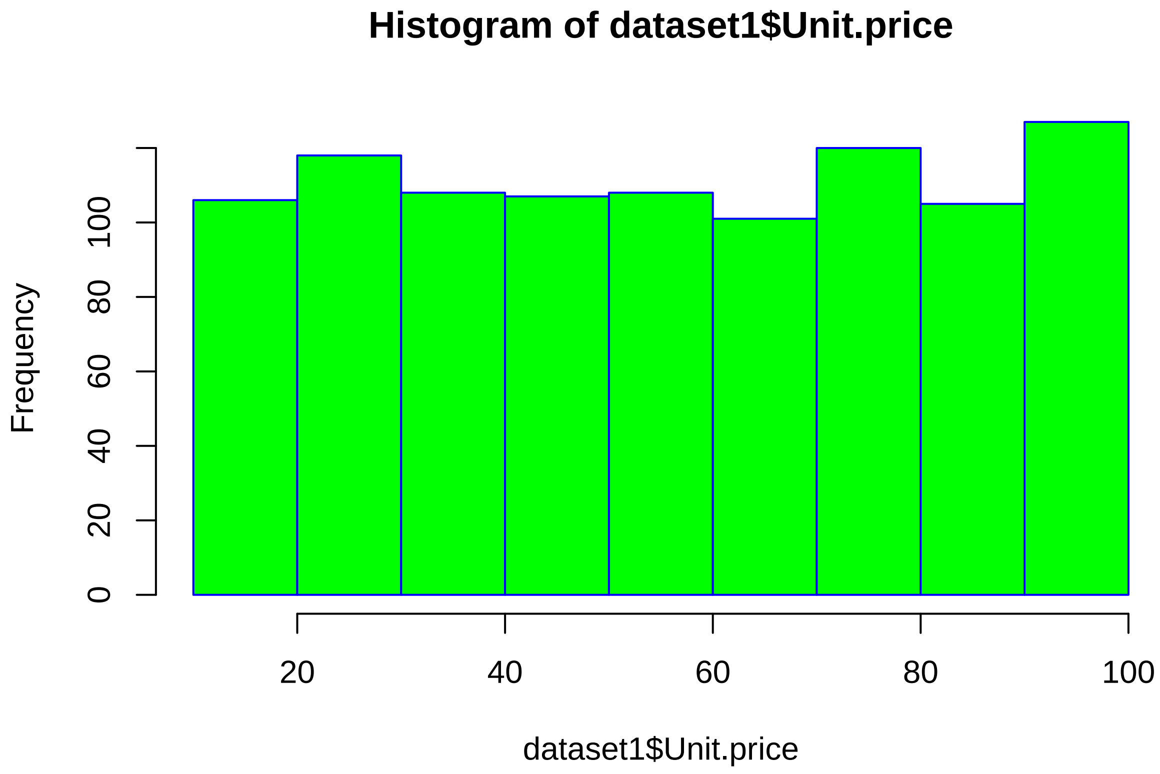
Change data types

Column data types

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

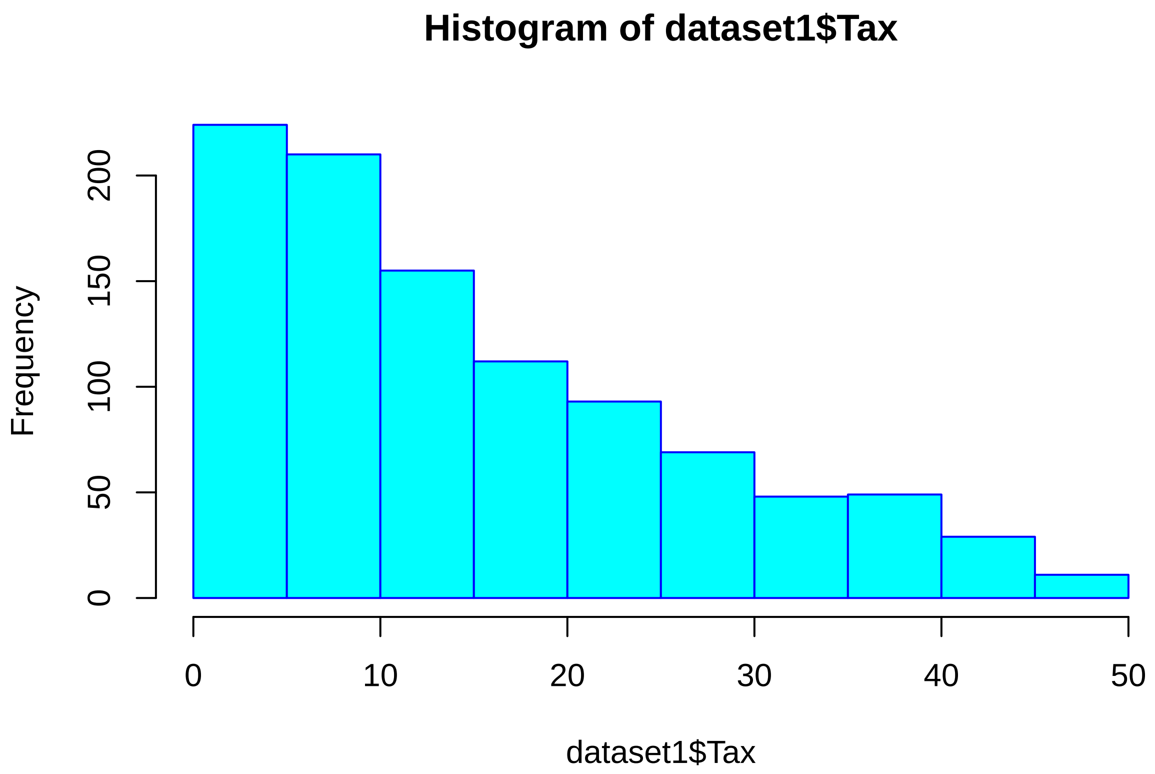
UNIVARIATE ANALYSIS

Unit.price



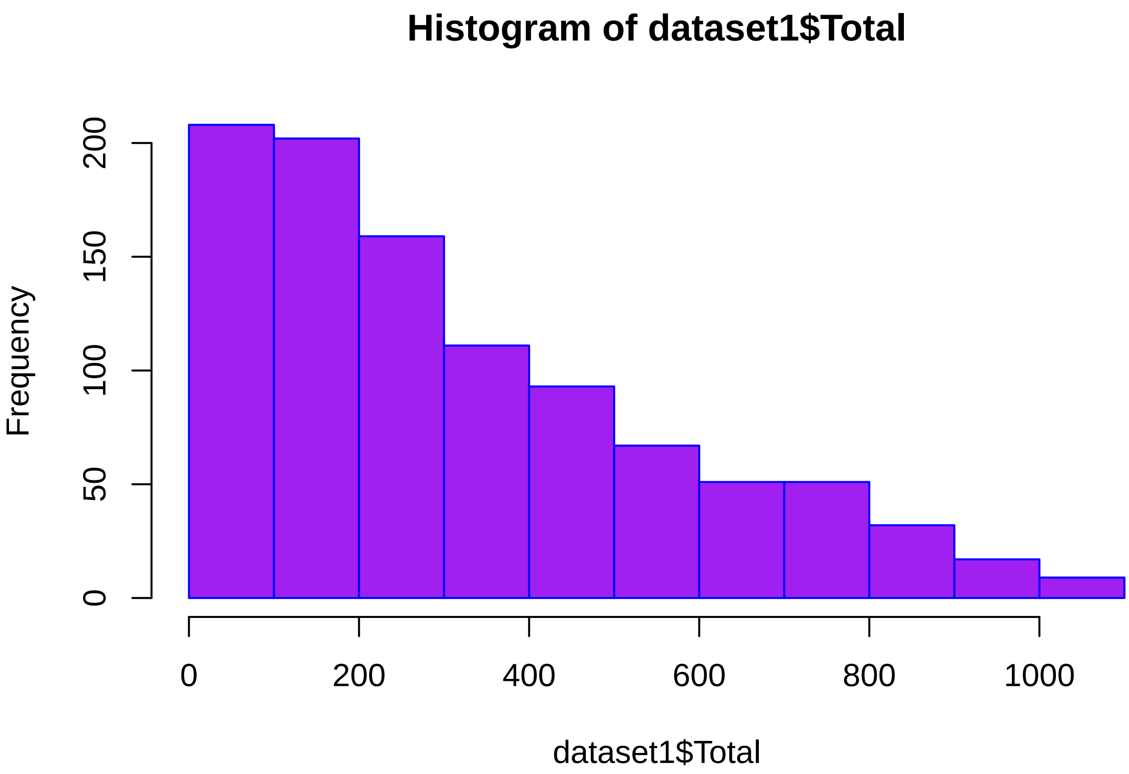
```
## [1] "mean"

## [1] 55.67213

## [1] "median"

## [1] 55.23

## [1] "mode"

## [1] 83.77
```

Tax



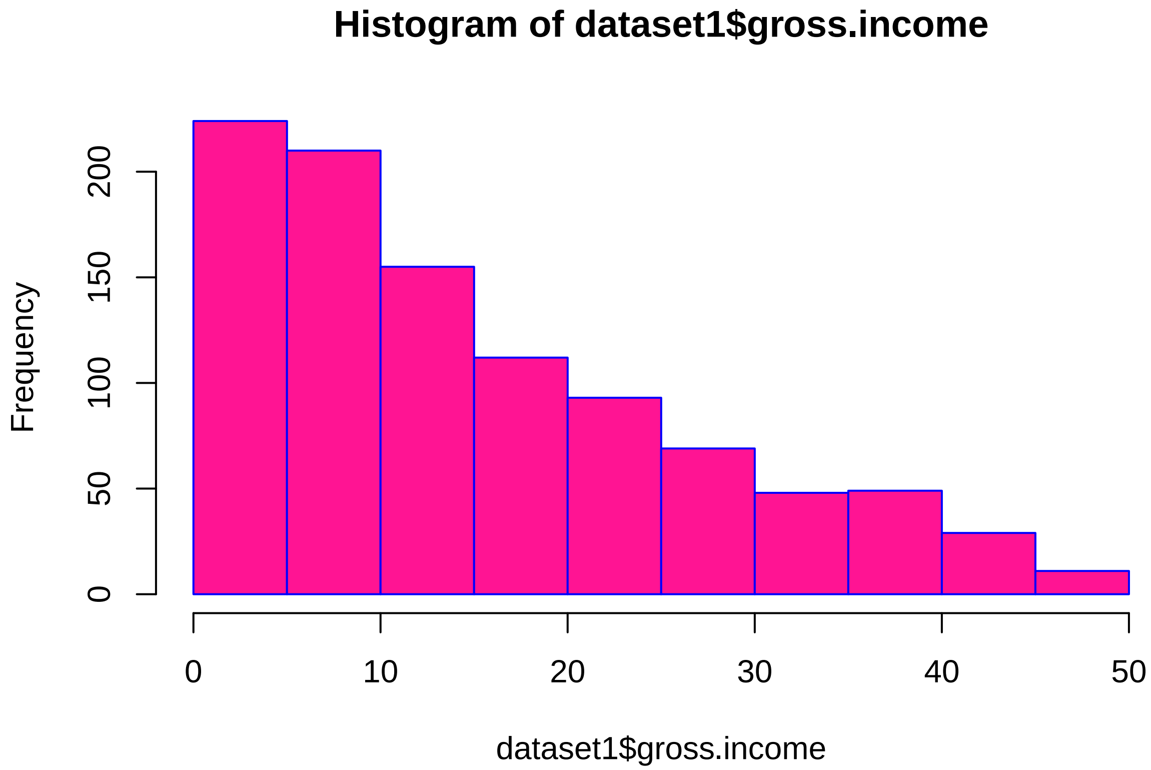
```
## [1] "mean"

## [1] 15.379369

## [1] "median"

## [1] 12.088

## [1] "mode"

## [1] 39.48
```

Total



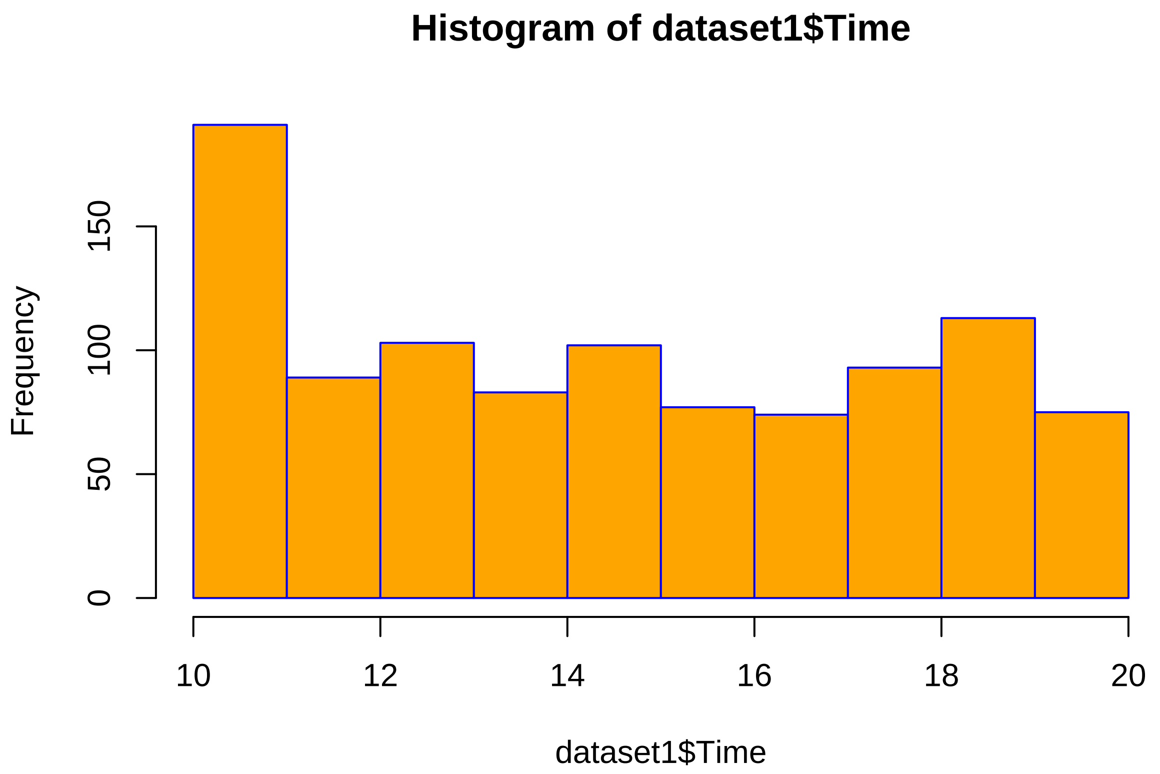
```
## [1] "mean"
## [1] 322.966749
## [1] "median"
## [1] 253.848
## [1] "mode"
## [1] 829.08
```

gross.income



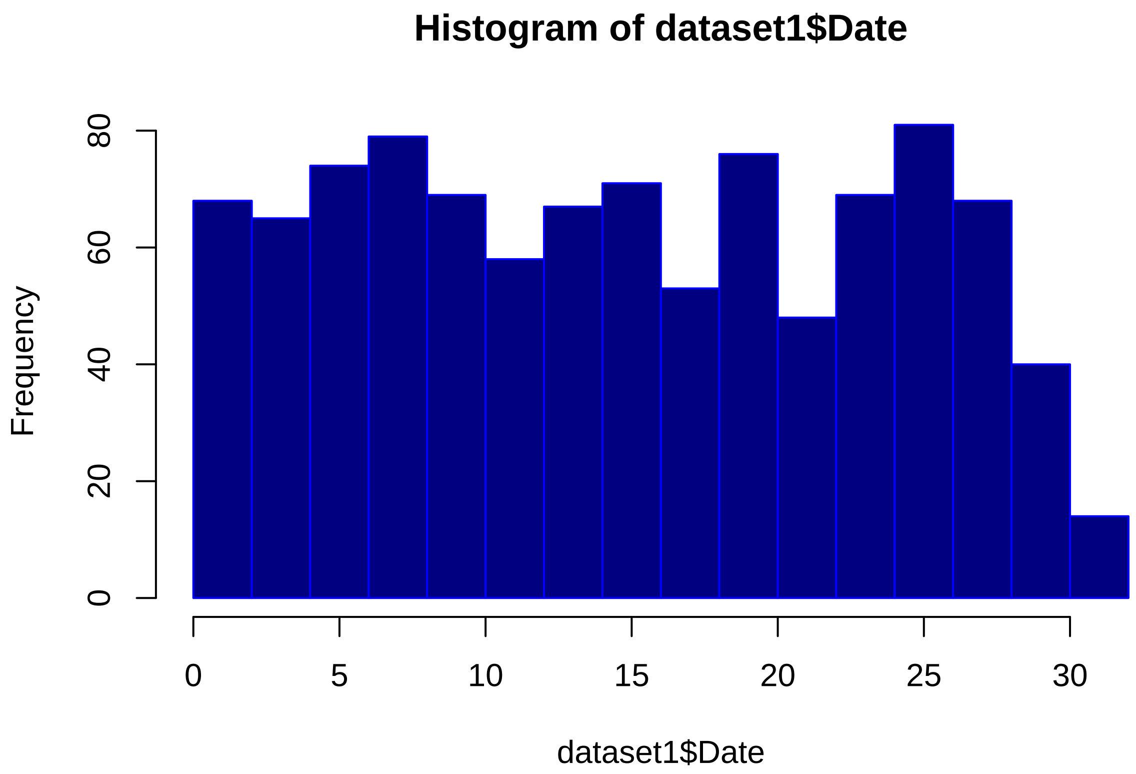
```
## [1] "mean"

## [1] 15.379369

## [1] "median"

## [1] 12.088

## [1] "mode"

## [1] 39.48
```

Time



```
## [1] "mean"
## [1] 14.91
## [1] "median"
## [1] 15
## [1] "mode"
## [1] 19
```

Date



```
## [1] "median"
```

```
## [1] 15
```

```
## [1] "mode"
```

```
## [1] 15
```

Correlation Matrix

Scaling

At this point we fit data to a a range of between 0 and 1.

```
##      Branch                 Customer.type              Gender
##  Min.   :-1.2083653760200   Min.   :-0.997502870195   Min.   :-0.997502870195
##  1st Qu.:-1.2083653760200   1st Qu.:-0.997502870195   1st Qu.:-0.997502870195
##  Median : 0.0146765025427   Median :-0.997502870195   Median :-0.997502870195
##  Mean   : 0.0000000000000   Mean   : 0.000000000000   Mean   : 0.000000000000
##  3rd Qu.: 1.2377183811000   3rd Qu.: 1.001500877690   3rd Qu.: 1.001500877690
##  Max.   : 1.2377183811000   Max.   : 1.001500877690   Max.   : 1.001500877690
##
##   Product.line               Unit.price                  Quantity
##  Min.   :-1.429394146310   Min.   :-1.7208065499700   Min.   :-1.542708079680
##  1st Qu.:-0.846443841940   1st Qu.:-0.8604434718100   1st Qu.:-0.858580328160
##  Median :-0.263493537574   Median :-0.0166875335707   Median :-0.174452576638
##  Mean   : 0.000000000000   Mean   : 0.0000000000000   Mean   : 0.000000000000
##  3rd Qu.: 0.902407071159   3rd Qu.: 0.8402786296020   3rd Qu.: 0.851739050645
##  Max.   : 1.485357375530   Max.   : 1.6715792129000   Max.   : 1.535866802170
##
##       Tax                       Date
##  Min.   :-1.270056422320   Min.   :-1.6398339305000
##  1st Qu.:-0.807467325851   1st Qu.:-0.8346405022250
##  Median :-0.281101550735   Median :-0.0294470739484
##  Mean   : 0.000000000000   Mean   : 0.0000000000000
##  3rd Qu.: 0.603466249580   3rd Qu.: 0.8907739869390
##  Max.   : 2.926905952750   Max.   : 1.8109950478300
##
##       Time                     Payment
##  Min.   :-1.5407030683600   Min.   :-1.20533432891000
##  1st Qu.:-0.9131254437750   1st Qu.:-1.20533432891000
##  Median : 0.0282409931065   Median :-0.00120413019871
##  Mean   : 0.0000000000000   Mean   : 0.00000000000000
##  3rd Qu.: 0.9696074299880   3rd Qu.: 1.20292606852000
##  Max.   : 1.5971850545800   Max.   : 1.20292606852000
##
##       cogs                 gross.margin.percentage  gross.income
##  Min.   :-1.270056422320   Min.   : NA             Min.   :-1.270056422320
##  1st Qu.:-0.807467325851   1st Qu.: NA             1st Qu.:-0.807467325851
##  Median :-0.281101550735   Median : NA             Median :-0.281101550735
##  Mean   : 0.000000000000   Mean   :NaN             Mean   : 0.000000000000
##  3rd Qu.: 0.603466249580   3rd Qu.: NA             3rd Qu.: 0.603466249580
##  Max.   : 2.926905952750   Max.   : NA             Max.   : 2.926905952750
##                            NA's   :1000
##      Rating
##  Min.   :-1.7297417000100
##  1st Qu.:-0.8569282475870
##  Median : 0.0158852048341
##  Mean   : 0.0000000000000
##  3rd Qu.: 0.8886986572550
##  Max.   : 1.7615121096800
##
```

PRINCIPAL COMPONENT ANALYSIS