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

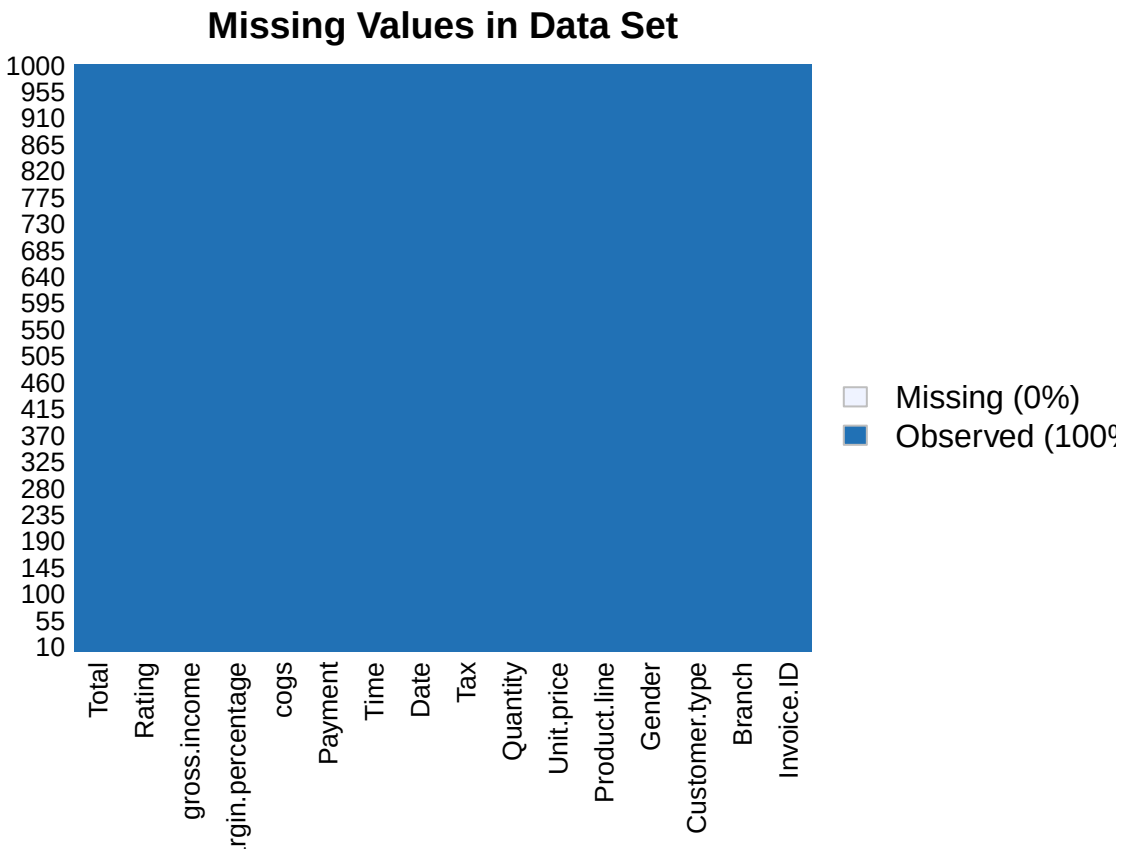
```
library(Amelia)
```

```
## 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

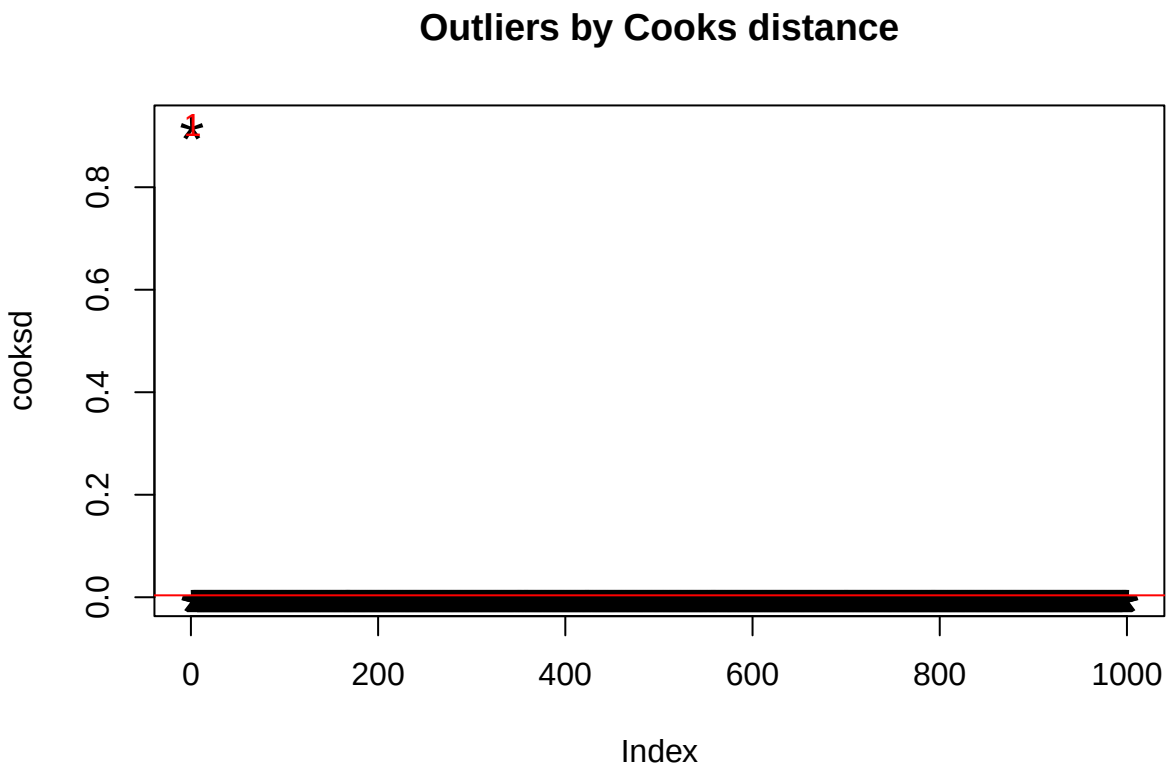
```
dataset1 <- subset( dataset1, select = -Invoice.ID )
```

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
```

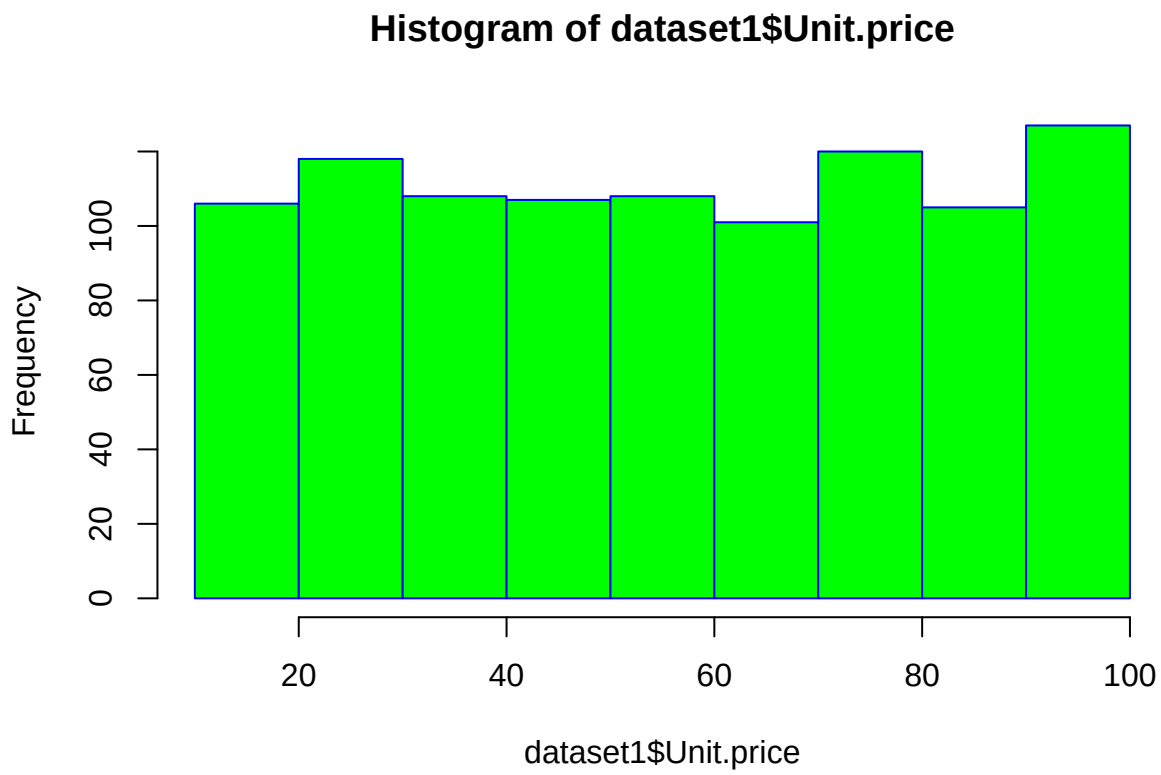
Change data types

Column data types

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

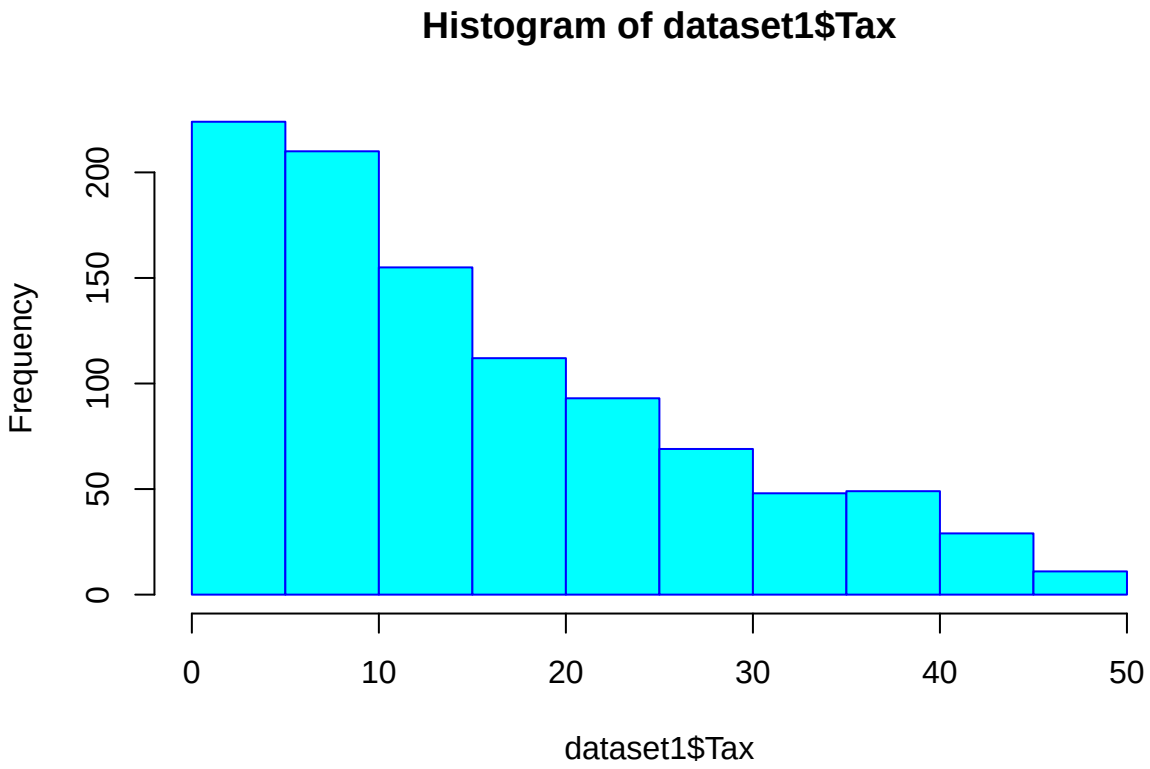
UNIVARIATE ANALYSIS

Unit.price



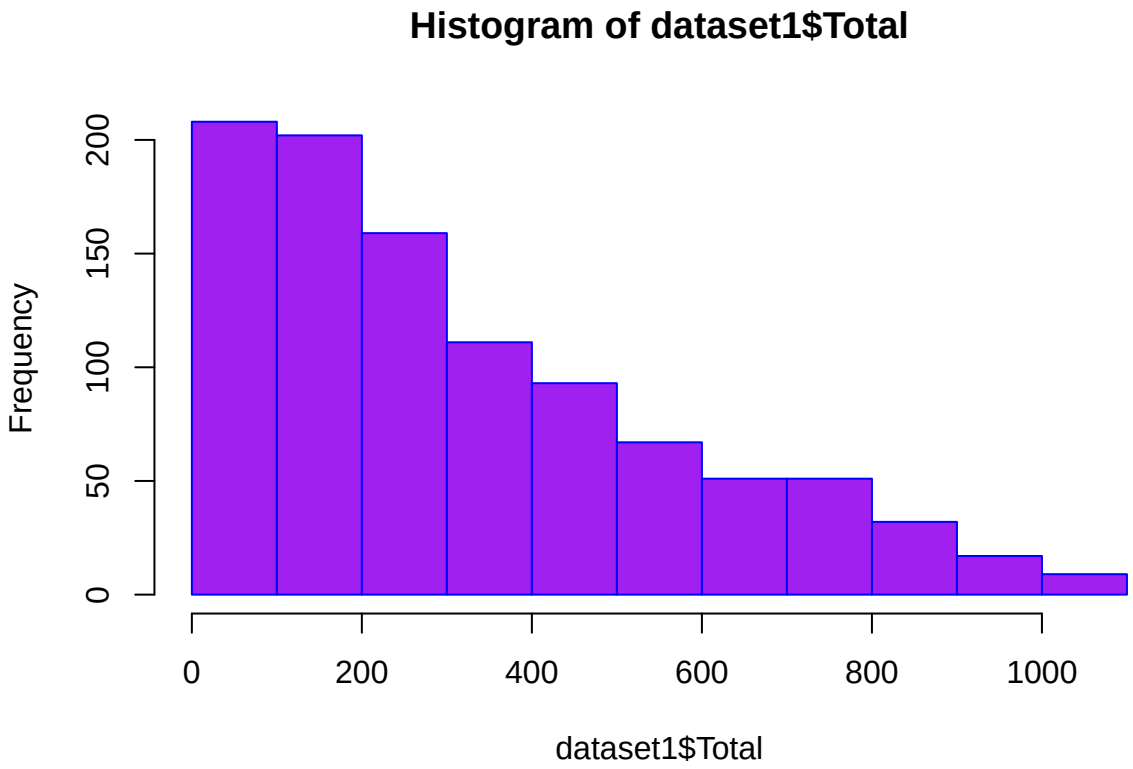

```
## [1] "mean"
## [1] 55.67213
## [1] "median"
## [1] 55.23
## [1] "mode"
## [1] 83.77
```

Tax



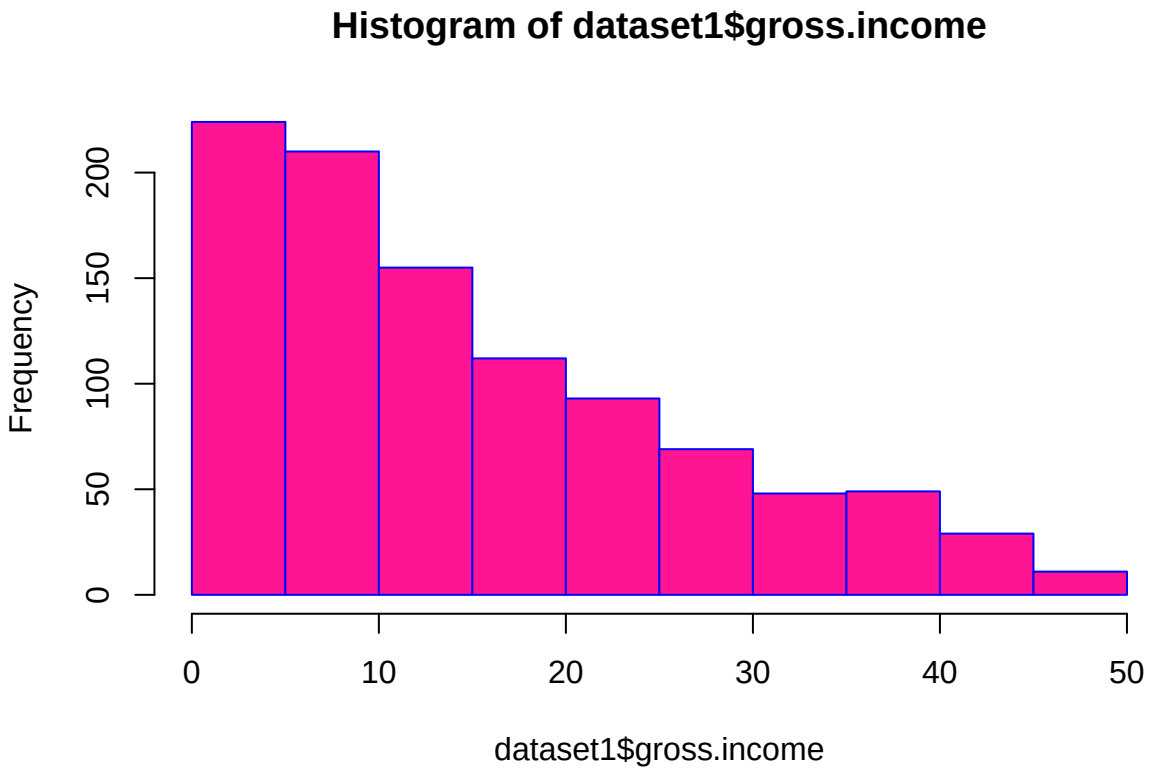

```
## [1] "mean"
## [1] 15.379369
## [1] "median"
## [1] 12.088
## [1] "mode"
## [1] 39.48
```

Total



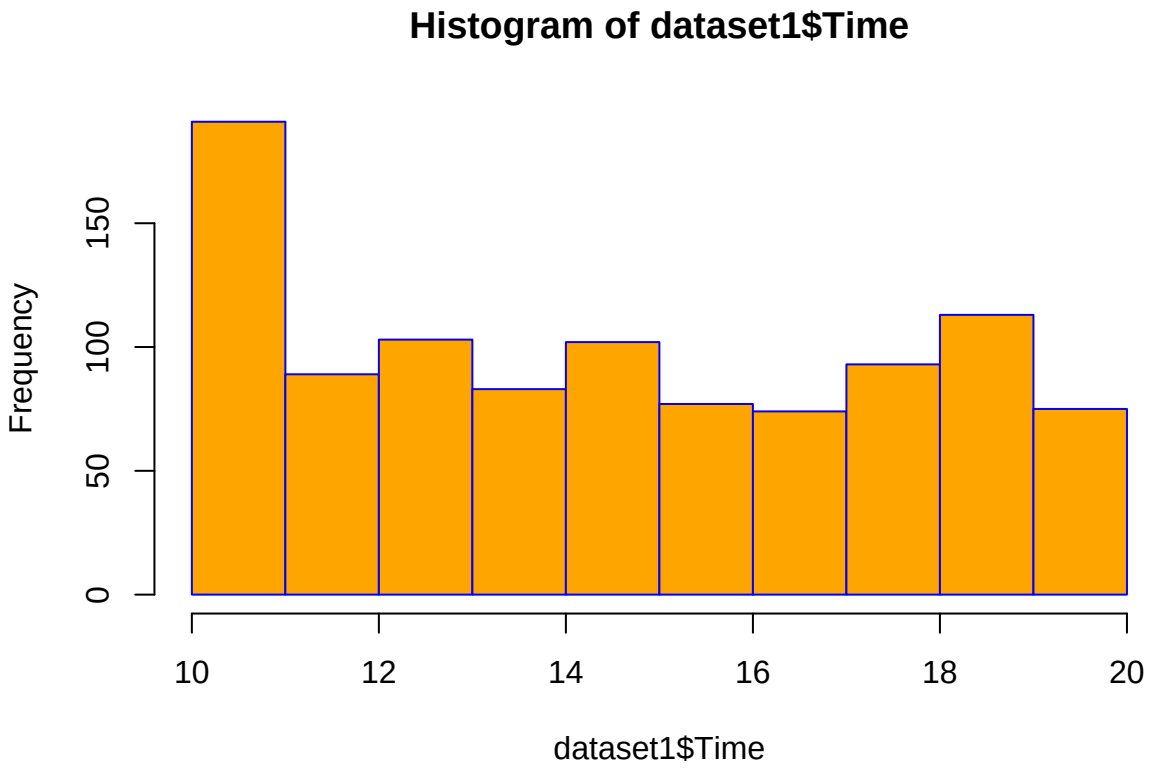

```
## [1] "mean"
## [1] 322.966749
## [1] "median"
## [1] 253.848
## [1] "mode"
## [1] 829.08
```

gross.income



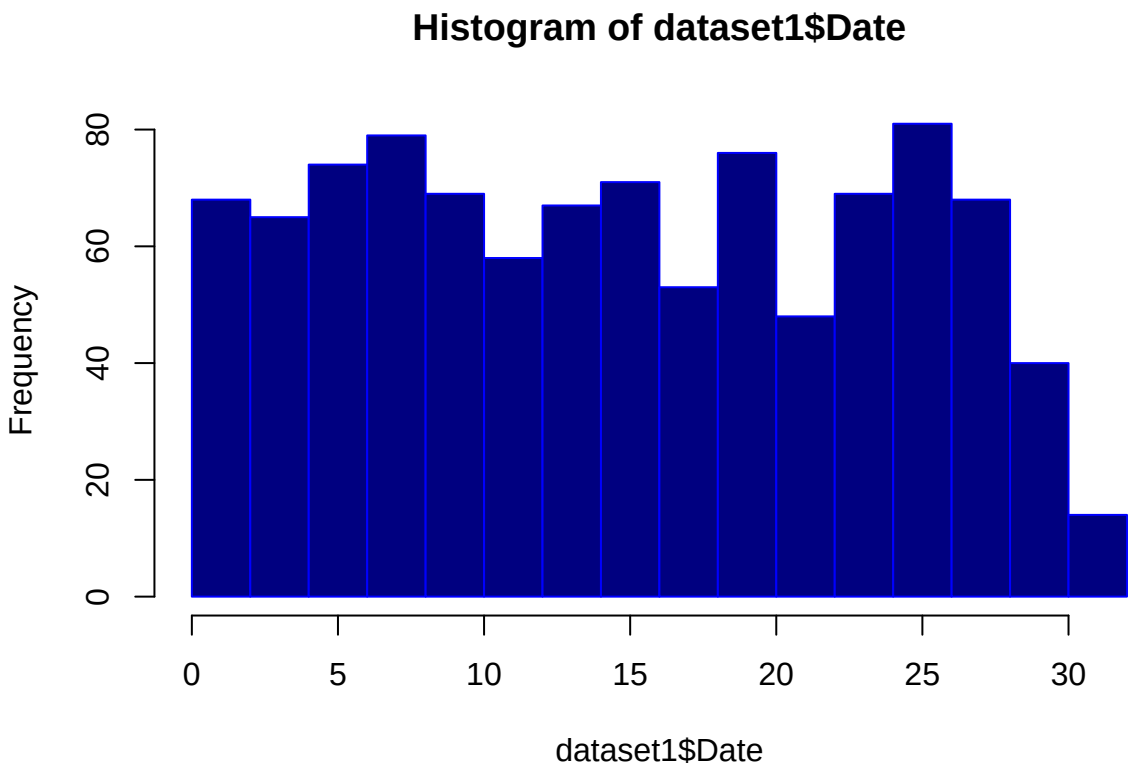

```
## [1] "mean"
## [1] 15.379369
## [1] "median"
## [1] 12.088
## [1] "mode"
## [1] 39.48
```

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

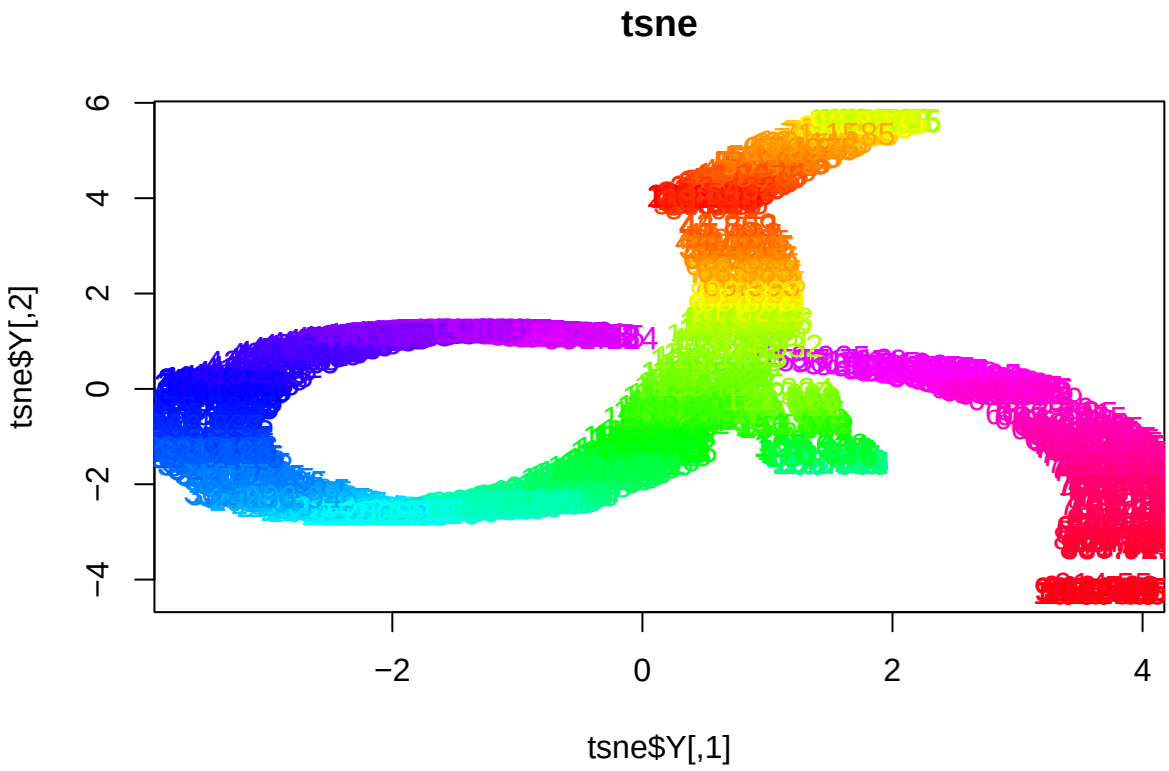
At this point we fit data to a a range of between 0 and 1.

T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING(t-SNE)

```
## Performing PCA
## Read the 1000 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 14.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.08 seconds (sparsity = 0.048264)!
## Learning embedding...
```

```
## Iteration 50: error is 71.991879 (50 iterations in 0.11 seconds)
## Iteration 100: error is 61.330624 (50 iterations in 0.10 seconds)
## Iteration 150: error is 58.906093 (50 iterations in 0.10 seconds)
## Iteration 200: error is 57.789531 (50 iterations in 0.10 seconds)
## Fitting performed in 0.40 seconds.
```

```
## Performing PCA
## Read the 1000 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 14.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.08 seconds (sparsity = 0.048264)!
## Learning embedding...
## Iteration 50: error is 69.495284 (50 iterations in 0.10 seconds)
## Iteration 100: error is 59.605726 (50 iterations in 0.09 seconds)
## Iteration 150: error is 57.724202 (50 iterations in 0.10 seconds)
## Iteration 200: error is 56.931747 (50 iterations in 0.10 seconds)
## Fitting performed in 0.39 seconds.
```



Part 2: Feature Selection

Importance of features can be estimated from data by building a Learning Vector Quantization (LVQ) model. The varImp is then used to estimate the variable importance, which is printed and plotted.

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:psych':
##
##     outlier
## The following object is masked from 'package:gridExtra':
##
##     combine
## The following object is masked from 'package:ggplot2':
##
##     margin
## The following object is masked from 'package:dplyr':
##
##     combine
##                              IncNodePurity
## Unit.price                5711103.233488476
## Quantity                  8042322.493624696
## Time                       250662.422682511
## cogs                     22805674.575103264
## gross.margin.percentage         0.000000000
## gross.income             23297593.900623526
## Rating                     246471.188408402
```