Required packages

**library**(dplyr) *# for pipe operator*

**library**(ggplot2) *# for display of plots*

**library**(readr) *# to import datasets*

**library**(lattice) *# to display graphs*

**library**(lubridate)*# for operations on date and time variables*

**library**(tidyr) *# used to tidy data*

**library**(forecast) *# used for Box-Cox transformation*

**library**(validate) *# to use functions on dataframes*

Aim/Objective

* To perform necessary data pre-processing operations on the dataset to infer and perform analysis on the data

Executive Summary

* First, the necessary packages are installed and loaded to use required operators and function for data pre-processing. The two datasets of interest are then imported into R workspace to perform required operations.
* Before performing the pre-processing task, the two datasets are merged using left join condition in order to gather the details of only those orders which has delivery status.
* The summary of merged dataset is presented which indicates statistical and general information of all the variables.
* The required variables are converted into suitable data types such as factor, character, numeric, etc. as required
* Post datatype conversions, the NA values are identified and omitted to perform required operations.
* Although the dataset is in UNTIDY format considering the presence of date and time in the same variable, we are interested to find processing time of the order.
* Another variable named delivery\_time\_in\_hours is mutated to display the delivery time
* The dataset is then TIDIED to separate order, month and day from the order purchased and order delivery variables.
* The histogram of the price variable is presented to understand the frequency distribution of price values. The data set is then filtered considering that more than 80 % of values in the dataset lies below 500.
* The box plot of the filtered price data indicates the presence of outliers using TUKEYS’s Method of Outlier Detection. The outliers observed are handled using the CAPPING method via a user-defined function which caps the values above 95percentile of the data.
* The histogram displayed after removing the outliers indicates that the data is right skewed and we need to NORMALIZE it.
* Lastly, data transformation using SQUARE ROOT METHOD is done to reduce the skewness of the price variable

Data

* The dataset named “Brazilian E-Commerce Public Dataset by Olist”, has over 100,000 Orders with product, customer and reviews info.
* Source : <https://www.kaggle.com/olistbr/brazilian-ecommerce>
* We have only considered 2 files amongst these whole dataset as desired by the assignment requirements which are olist\_orders\_dataset and olist\_order\_item\_dataset
* These two data files have the variables as Order\_id, product\_id and seller\_id which are alphanumeric, order\_item\_id, price and freight\_value which are numeric, order\_status is character but will be converted into ordered factor, order\_purchase\_timestamp and order\_delivered\_customer\_date are date time variables denoting the time of order and the time of delivery which are again in the character type but will be converted into date and time datatype.

\*The variable headings are self explanatory for both files

Importing and Reading the data set

data1 <- read\_csv("olist\_orders\_items\_dataset.csv")

data2 <- read\_csv("olist\_orders\_dataset.csv")

Order Items Dataset

head(data1)

|  |
| --- |
|  |

| **order\_id**  <chr> | **order\_item\_id**  <dbl> | **product\_id**  <chr> |  |
| --- | --- | --- | --- |
| 00010242fe8c5a6d1ba2dd792cb16214 | 1 | 4244733e06e7ecb4970a6e2683c13e61 |  |
| 00018f77f2f0320c557190d7a144bdd3 | 1 | e5f2d52b802189ee658865ca93d83a8f |  |
| 000229ec398224ef6ca0657da4fc703e | 1 | c777355d18b72b67abbeef9df44fd0fd |  |
| 00024acbcdf0a6daa1e931b038114c75 | 1 | 7634da152a4610f1595efa32f14722fc |  |
| 00042b26cf59d7ce69dfabb4e55b4fd9 | 1 | ac6c3623068f30de03045865e4e10089 |  |
| 00048cc3ae777c65dbb7d2a0634bc1ea | 1 | ef92defde845ab8450f9d70c526ef70f |  |

6 rows | 1-3 of 6 columns

Overall Orders Dataset

head(data2)

|  |
| --- |
|  |

| **order\_id**  <chr> | **customer\_id**  <chr> | **order\_status**  <chr> |  |
| --- | --- | --- | --- |
| e481f51cbdc54678b7cc49136f2d6af7 | 9ef432eb6251297304e76186b10a928d | delivered |  |
| 53cdb2fc8bc7dce0b6741e2150273451 | b0830fb4747a6c6d20dea0b8c802d7ef | delivered |  |
| 47770eb9100c2d0c44946d9cf07ec65d | 41ce2a54c0b03bf3443c3d931a367089 | delivered |  |
| 949d5b44dbf5de918fe9c16f97b45f8a | f88197465ea7920adcdbec7375364d82 | delivered |  |
| ad21c59c0840e6cb83a9ceb5573f8159 | 8ab97904e6daea8866dbdbc4fb7aad2c | delivered |  |
| a4591c265e18cb1dcee52889e2d8acc3 | 503740e9ca751ccdda7ba28e9ab8f608 | delivered |  |

6 rows | 1-3 of 5 columns

Joining Data set

Left join ensures that we only pickup the data which has delivery status

data <- left\_join(data2, data1, by = "order\_id")

head(data, 10)

|  |
| --- |
|  |

| **order\_id**  <chr> | **customer\_id**  <chr> | **order\_status**  <chr> |  |
| --- | --- | --- | --- |
| e481f51cbdc54678b7cc49136f2d6af7 | 9ef432eb6251297304e76186b10a928d | delivered |  |
| 53cdb2fc8bc7dce0b6741e2150273451 | b0830fb4747a6c6d20dea0b8c802d7ef | delivered |  |
| 47770eb9100c2d0c44946d9cf07ec65d | 41ce2a54c0b03bf3443c3d931a367089 | delivered |  |
| 949d5b44dbf5de918fe9c16f97b45f8a | f88197465ea7920adcdbec7375364d82 | delivered |  |
| ad21c59c0840e6cb83a9ceb5573f8159 | 8ab97904e6daea8866dbdbc4fb7aad2c | delivered |  |
| a4591c265e18cb1dcee52889e2d8acc3 | 503740e9ca751ccdda7ba28e9ab8f608 | delivered |  |
| 136cce7faa42fdb2cefd53fdc79a6098 | ed0271e0b7da060a393796590e7b737a | invoiced |  |
| 6514b8ad8028c9f2cc2374ded245783f | 9bdf08b4b3b52b5526ff42d37d47f222 | delivered |  |
| 76c6e866289321a7c93b82b54852dc33 | f54a9f0e6b351c431402b8461ea51999 | delivered |  |
| e69bfb5eb88e0ed6a785585b27e16dbf | 31ad1d1b63eb9962463f764d4e6e0c9d | delivered |  |

1-10 of 10 rows | 1-3 of 10 columns

Understand

* To understand datatypes presented in the dataset, we are using summary() function

summary(data)

## order\_id customer\_id order\_status

## Length:113425 Length:113425 Length:113425

## Class :character Class :character Class :character

## Mode :character Mode :character Mode :character

##

##

##

##

## order\_purchase\_timestamp order\_delivered\_customer\_date order\_item\_id

## Length:113425 Length:113425 Min. : 1.000

## Class :character Class :character 1st Qu.: 1.000

## Mode :character Mode :character Median : 1.000

## Mean : 1.198

## 3rd Qu.: 1.000

## Max. :21.000

## NA's :775

## product\_id seller\_id price freight\_value

## Length:113425 Length:113425 Min. : 0.85 Min. : 0.00

## Class :character Class :character 1st Qu.: 39.90 1st Qu.: 13.08

## Mode :character Mode :character Median : 74.99 Median : 16.26

## Mean : 120.65 Mean : 19.99

## 3rd Qu.: 134.90 3rd Qu.: 21.15

## Max. :6735.00 Max. :409.68

## NA's :775 NA's :775

* The “Order\_Status” Variable is an Ordinal variable andis converted into factor and leveled accordingly

data$order\_status <- data$order\_status %>% factor(levels = c("unavailable", "created", "cancelled", "approved", "processing","invoiced", "shipped", "delivered"), ordered = TRUE)

class(data$order\_status) *# Checking class of variable after conversion*

## [1] "ordered" "factor"

levels(data$order\_status) *# levelling order\_status variable*

## [1] "unavailable" "created" "cancelled" "approved" "processing"

## [6] "invoiced" "shipped" "delivered"

TIDY dataset

* Currently the dataset is in UNTIDY format due to the presence of date, and time values in the same variable.
* We would be first checking for the missing values in order to add delivery time variable and take necessary measures before hand
* Converting required variables into Date and Time Values

data$order\_purchase\_timestamp <- data$order\_purchase\_timestamp %>% dmy\_hm()

data$order\_delivered\_customer\_date <- data$order\_delivered\_customer\_date %>% dmy\_hm()

str(data)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 113425 obs. of 10 variables:

## $ order\_id : chr "e481f51cbdc54678b7cc49136f2d6af7" "53cdb2fc8bc7dce0b6741e2150273451" "47770eb9100c2d0c44946d9cf07ec65d" "949d5b44dbf5de918fe9c16f97b45f8a" ...

## $ customer\_id : chr "9ef432eb6251297304e76186b10a928d" "b0830fb4747a6c6d20dea0b8c802d7ef" "41ce2a54c0b03bf3443c3d931a367089" "f88197465ea7920adcdbec7375364d82" ...

## $ order\_status : Ord.factor w/ 8 levels "unavailable"<..: 8 8 8 8 8 8 6 8 8 8 ...

## $ order\_purchase\_timestamp : POSIXct, format: "2017-10-02 10:56:00" "2018-07-24 20:41:00" ...

## $ order\_delivered\_customer\_date: POSIXct, format: "2017-10-10 21:25:00" "2018-08-07 15:27:00" ...

## $ order\_item\_id : num 1 1 1 1 1 1 1 1 1 1 ...

## $ product\_id : chr "87285b34884572647811a353c7ac498a" "595fac2a385ac33a80bd5114aec74eb8" "aa4383b373c6aca5d8797843e5594415" "d0b61bfb1de832b15ba9d266ca96e5b0" ...

## $ seller\_id : chr "3504c0cb71d7fa48d967e0e4c94d59d9" "289cdb325fb7e7f891c38608bf9e0962" "4869f7a5dfa277a7dca6462dcf3b52b2" "66922902710d126a0e7d26b0e3805106" ...

## $ price : num 30 118.7 159.9 45 19.9 ...

## $ freight\_value : num 8.72 22.76 19.22 27.2 8.72 ...

Scanning for missing values

Scan I

colSums(is.na(data))

## order\_id customer\_id

## 0 0

## order\_status order\_purchase\_timestamp

## 706 0

## order\_delivered\_customer\_date order\_item\_id

## 3229 775

## product\_id seller\_id

## 775 775

## price freight\_value

## 775 775

sum(is.na(data))

## [1] 7810

* This shows that the data has a total of 7810 missing values
* Thus, before finding the delivery time of the order, we will remove the NA values

new\_data <- na.omit(data) *# removing NA values and keeping only rows with complete cases*

colSums(is.na(new\_data)) *# Checking for removed NA values*

## order\_id customer\_id

## 0 0

## order\_status order\_purchase\_timestamp

## 0 0

## order\_delivered\_customer\_date order\_item\_id

## 0 0

## product\_id seller\_id

## 0 0

## price freight\_value

## 0 0

Tidy & Manipulate Data I

* We will mutate a new variable Delivery time in “dmy\_hms” format to guage the delivery time of each individual order
* We are interested to find the total time for deliverty in hours, thus we are adding another variable named delivery time before tidying the data set

mutate\_data <- mutate(new\_data, delivery\_time\_in\_hours = (ymd\_hms(order\_delivered\_customer\_date) - ymd\_hms(order\_purchase\_timestamp)))

mutate\_data$delivery\_time\_in\_days <- round(mutate\_data$delivery\_time\_in\_hours)

head(mutate\_data$delivery\_time\_in\_hours) %>% round(0)

## Time differences in hours

## [1] 202 331 225 317 69 397

Tidy & Manipulate Data II

* The order\_status data set is the untidy dataset as the year, month, day,hours and minutes values are present in the order\_purchase\_timestamp variable and order\_delivered\_customer\_date variable

mutate\_data <- mutate\_data %>% separate (order\_purchase\_timestamp, into = c("Date", "Time"), sep = " ")

mutate\_data <- mutate\_data %>% separate (Date, into = c("Year","Month", "Day"), sep = "-")

mutate\_data <- mutate\_data %>% separate (order\_delivered\_customer\_date, into = c("O\_Date", "O\_Time"), sep = " ")

mutate\_data <- mutate\_data %>% separate (O\_Date, into = c("O\_Year","O\_Month", "O\_Day"), sep = "-")

select(mutate\_data,Year,Month, Day,O\_Year, O\_Month, O\_Day) %>% head()

|  |
| --- |
|  |

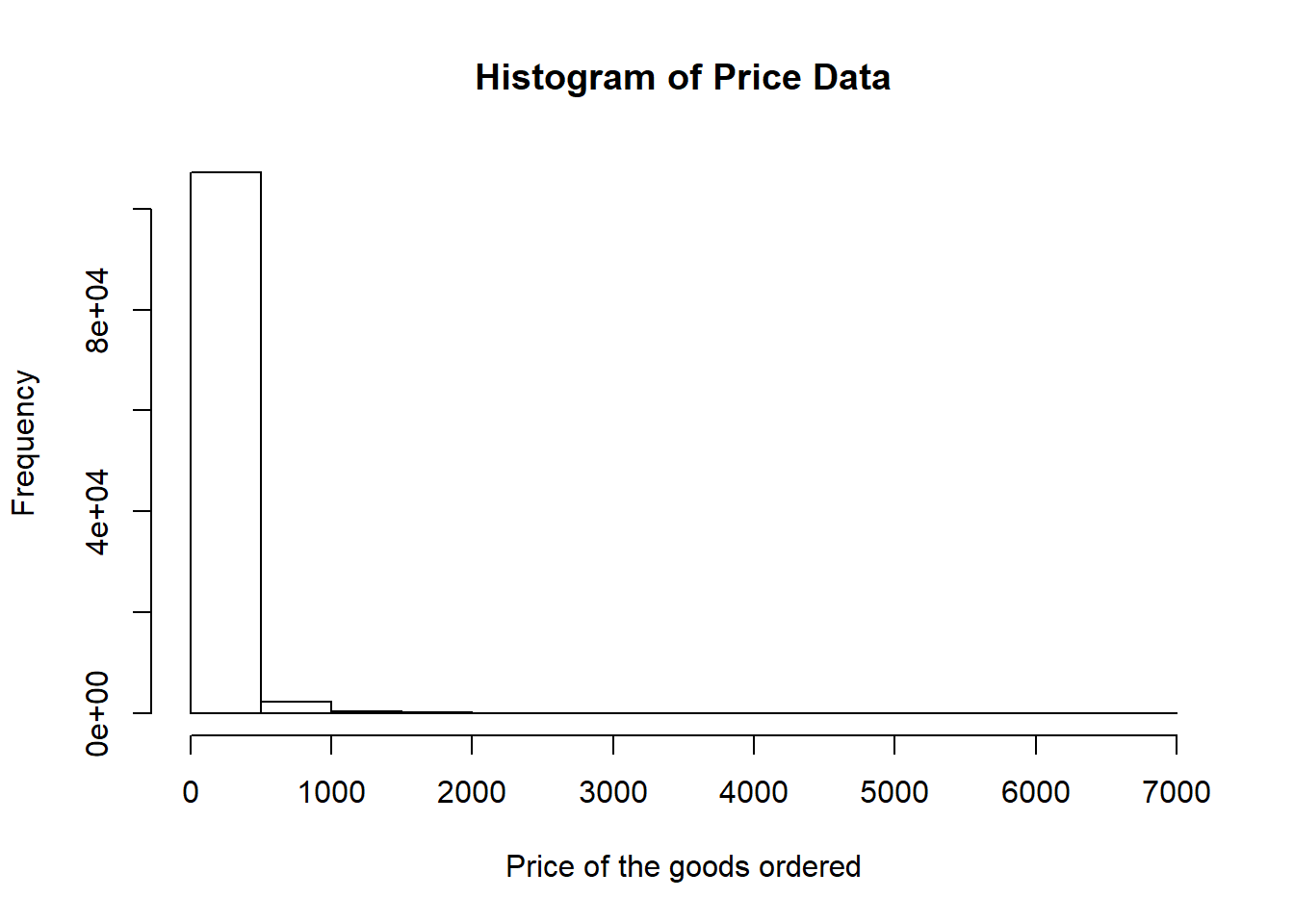
| **Year**  <chr> | **Month**  <chr> | **Day**  <chr> | **O\_Year**  <chr> | **O\_Month**  <chr> | **O\_Day**  <chr> |
| --- | --- | --- | --- | --- | --- |
| 2017 | 10 | 02 | 2017 | 10 | 10 |
| 2018 | 07 | 24 | 2018 | 08 | 07 |
| 2018 | 08 | 08 | 2018 | 08 | 17 |
| 2017 | 11 | 18 | 2017 | 12 | 02 |
| 2018 | 02 | 13 | 2018 | 02 | 16 |
| 2017 | 07 | 09 | 2017 | 07 | 26 |

6 rows

* The above outputs displays only the varibles which are created due to separate function

Ploting the histogram of price variable

hist(mutate\_data$price, xlab = "Price of the goods ordered", main = "Histogram of Price Data")



* As, the majority of price values exists between 0 to 500. We would be filtering the dataset gain price values less that 500.

box\_price <- mutate\_data %>% filter(price <= 500)

head(box\_price$price,20)

## [1] 29.99 118.70 159.90 45.00 19.90 147.90 59.99 19.90 149.99 99.00

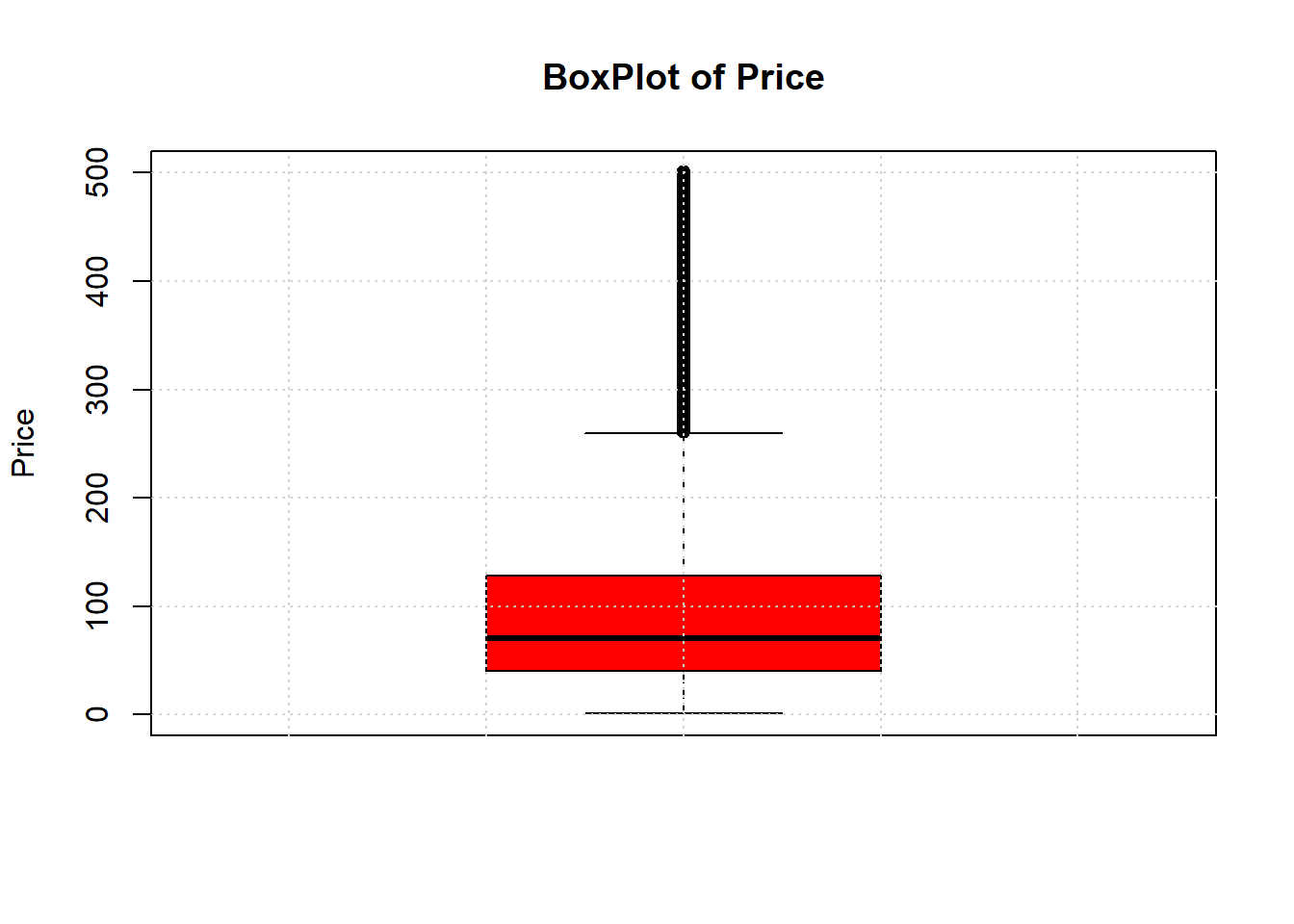
## [11] 99.00 98.00 31.90 19.90 38.25 132.40 27.99 17.90 76.00 109.90

Scan II

* Based on below box plot, the TUKEY’s Method of Outlier detection is used to detect the outliers

boxplot(box\_price$price, main = "BoxPlot of Price", ylab = "Price" , col = "red" )

grid()



Handling outliers

* To handle outliers from the filterd data set, we are using capping method where the values above the upper limit are replaced by 95 percentile
* The user defined function below is taken from Stackflow and is used to handle outliers.

cap<- **function** (x){ *## Capping the dataset to handle outliers*

quantiles <- quantile(x,c(.05, 0.25, 0.75, 0.95))

x[ x < quantiles[2] - 1.5\*IQR(x) ] <- quantiles[1]

x[x > quantiles[3] + 1.5\*IQR(x) ] <-quantiles[4]

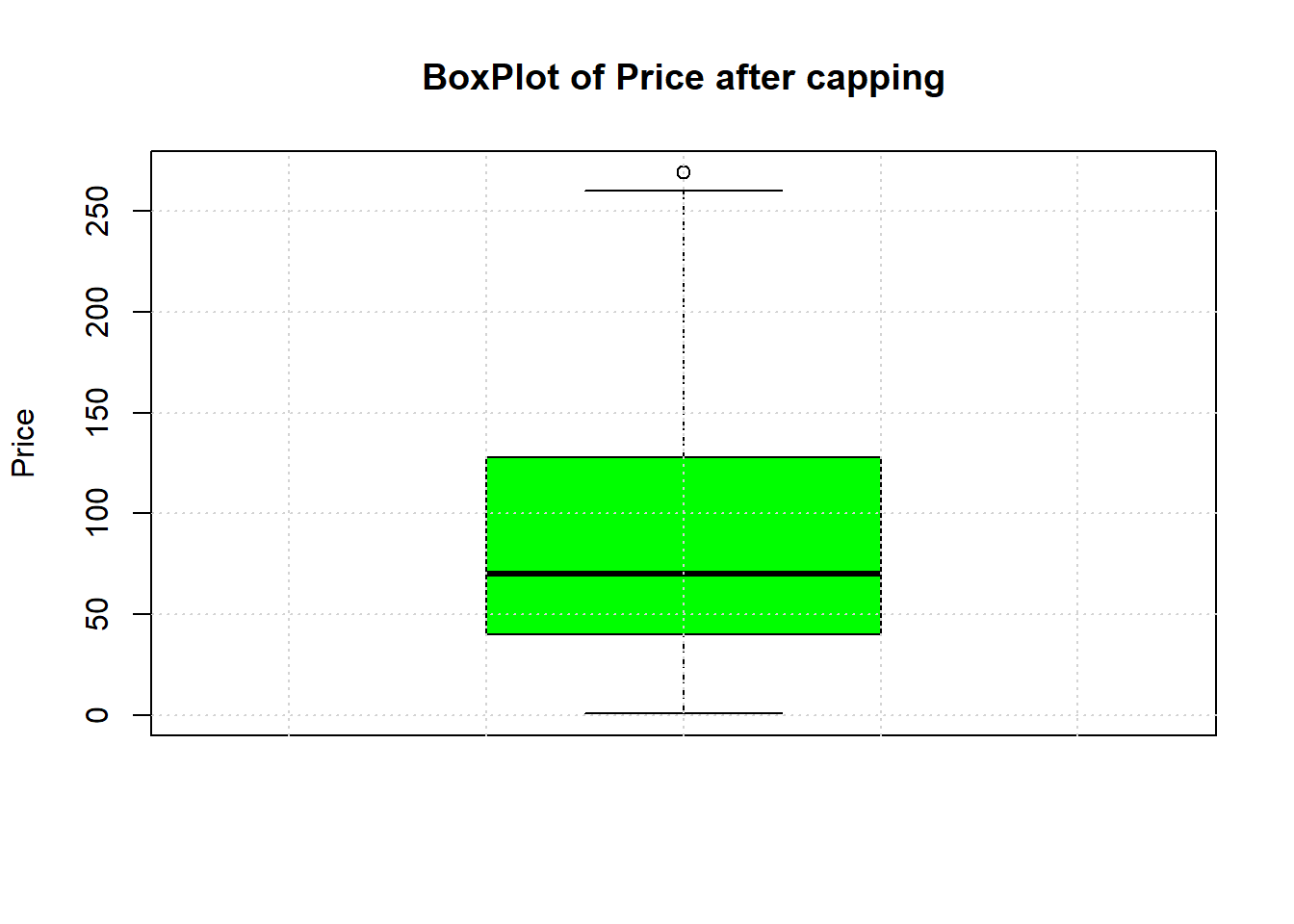
x

}

box\_price$price<- box\_price$price %>% cap()

boxplot(box\_price$price, main = "BoxPlot of Price after capping", ylab= "Price", col ="green") *## Box Plot of Required Dataset*

grid()

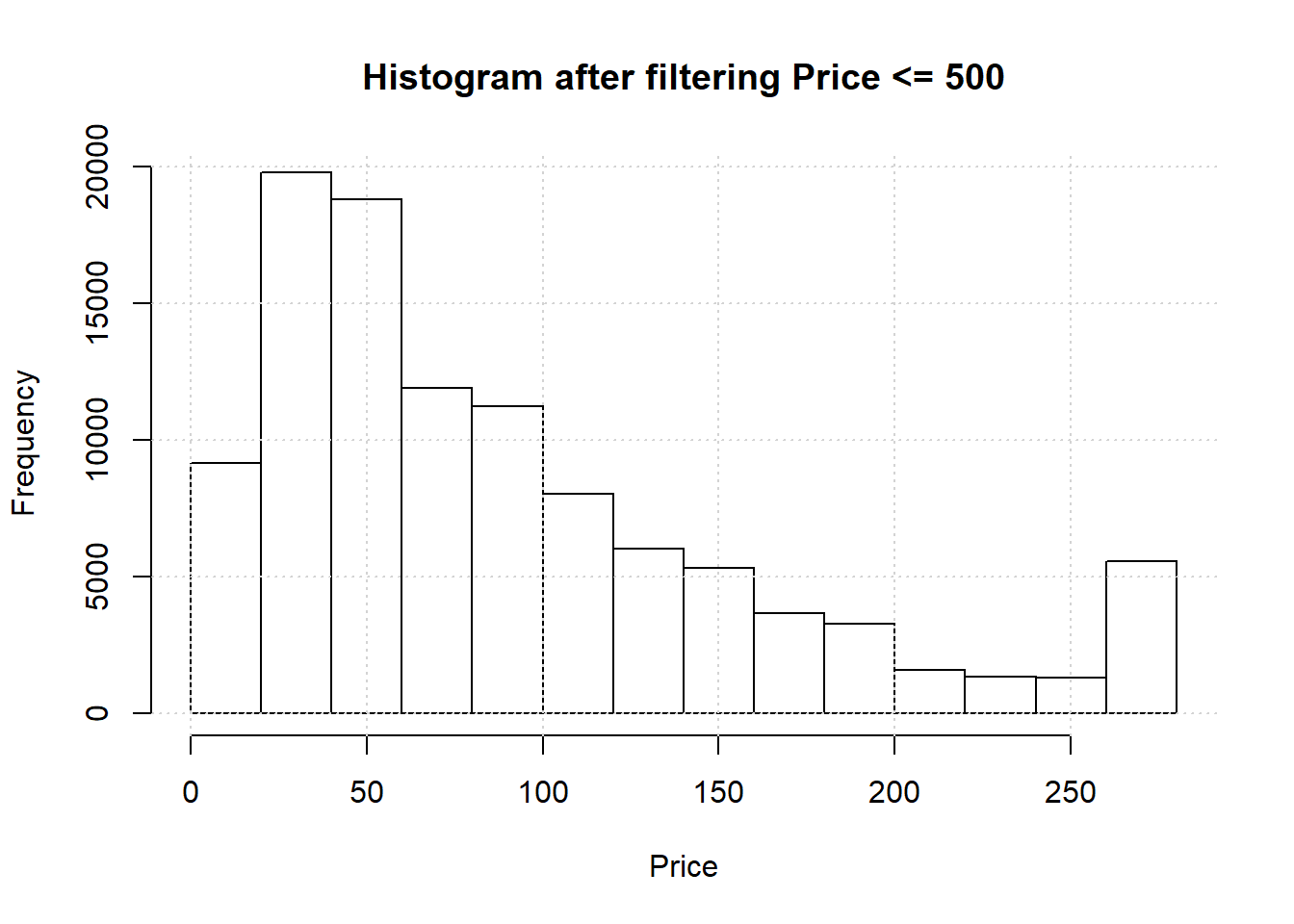


Before Transformation

* The below histogram plot of filtered price variable (value less than 500) shows that the data is right skewed. Thus, appropriate measures are taken to reduce skewness of the dataset

hist(box\_price$price, xlab = "Price", main="Histogram after filtering Price <= 500")

grid()

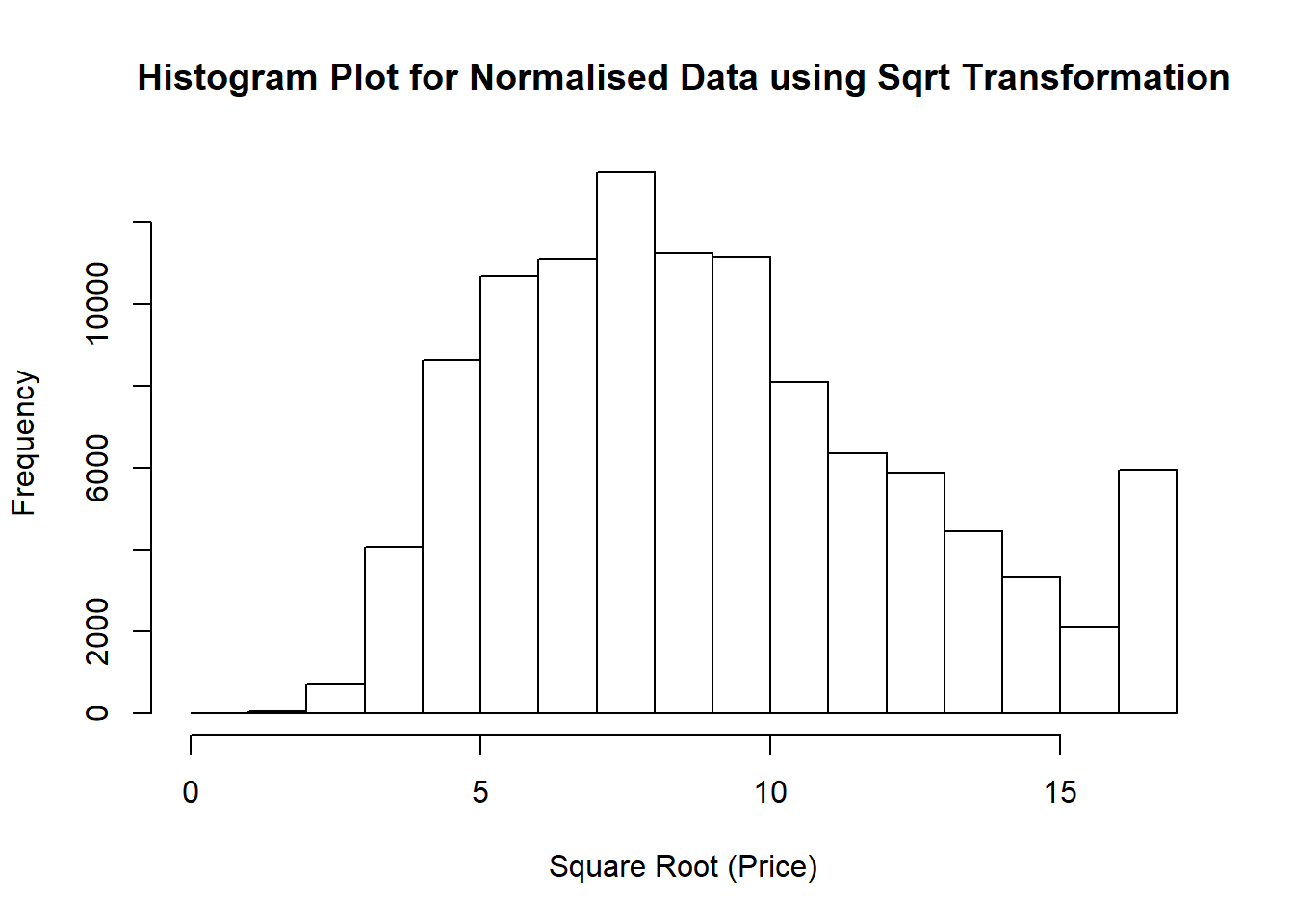


Post Transformation

* After trying several transformations such box-cox, log10, log, min max normalisation and z-score, we found Sqrt method as the most appropriate method for the transformation.

squareroottransform <- sqrt(box\_price$price)

hist(squareroottransform, xlab = "Square Root (Price)", main = "Histogram Plot for Normalised Data using Sqrt Transformation")



Conclusion

* Hence, by performing various operations on the datasets like mutation, datatype conversions, filtering, using capping functions etc., we conclude that the obtained dataset can now be used to interpret data and derive inferences from it as we have successfully pre-preprocessed the data for further analysis as and when required.