Data Wrangling (Data Preprocessing)

Practical assessment 2

Ruiyang Fu And Jie Chen

Setup

```
# Load the necessary packages required to reproduce the report. For example:

library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.1.3
```

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.1.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
##
## 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
```

```
library(kableExtra)
```

```
## Warning: package 'kableExtra' was built under R version 4.1.3
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

Student names, numbers and percentage of contributions

Group information

Student name	Student number	Percentage of contribution
Ruiyang Fu	s3679150	50%
Jie Chen	s3956275	50%

Executive Summary

This report is created by extracting the data from two xlsx files and understanding the data

a: To prepossess the rental report data, input new column names and tidy the data by changing it into a longer format

b: Changing data types to correct types.

c: Then dropping unnecessary columns and rows to prepare for merging.

d: The missing values are scanned and outliers for preprocessing the crime statistics data.

e: The data columns are changed into a more comprehensive format and tidied up.

f: Unnecessary rows and columns are dropped and both datasets are joined.

g: A transformation is performed for data values to fix skewness and reduce bias.

Data

Import VictimReport data and rent data, display their head and tail for viewing. Because the data is messy and not merged, so there is no summary for the time being, we will summarize the data in the subsequent tidy

```
# Import the data
VictimReports <- read_xlsx("Data_Tables_LGA_Harm_Caused_Visualisation_Year_Ending_December_202
1.xlsx", sheet = "Table 01")
rent <- read_excel("Affordable_lettings_by_local_government_area_-_September_quarter_2021.xlsx"
,
                   sheet = "lga aff total", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
## * `` -> ...3
## * `` -> ...4
## * `` -> ...5
## * ...
```

```
# Checking the VictimReports data head
head(VictimReports)
```

Y…	Year ending	Police Region	Local Government Area	Harm Caused Flag	Vict
<dbl>	<chr>	<chr>	<chr>	<chr>	
2021	December	1 North West Metro	Banyule	High Harm	
2021	December	1 North West Metro	Banyule	Low Harm	
2021	December	1 North West Metro	Banyule	Medium Harm	
2021	December	1 North West Metro	Brimbank	High Harm	
2021	December	1 North West Metro	Brimbank	Low Harm	
2021	December	1 North West Metro	Brimbank	Medium Harm	

6 rows

```
# Checking the VictimReports data tail
tail(VictimReports)
```

Y…	Year ending	Police Region	Local Government Area	Harm Caused Flag	Victim Rep
<dbl>	<chr>	<chr>	<chr>	<chr>	<
2012	December	4 Western	West Wimmera	High Harm	
2012	December	4 Western	West Wimmera	Low Harm	
2012	December	4 Western	West Wimmera	Medium Harm	
2012	December	4 Western	Yarriambiack	High Harm	
2012	December	4 Western	Yarriambiack	Low Harm	
2012	December	4 Western	Yarriambiack	Medium Harm	

6 rows

```
# Checking the rent data head
head(rent)
```

...1	...2	...3	...4
<chr>	<chr>	<chr>	<chr>
LGA Affordable rental - All bedrooms	*NA*	*NA*	*NA*
NA	Mar 2000	*NA*	Jun 2000
NA	Affordable	Percent	Affordable
Alpine	70	0.80500000000000005	69
Ararat	75	1	59
Ballarat	833	0.71099999999999997	440

6 rows | 1-4 of 175 columns

```
# Checking the rent data tail
tail(rent)
```

...1 <chr>	...2 <chr>	...3 <chr>	...4 <chr>	...5 <chr>	...6 <chr>	...7 <chr>
Yarriambiack	11	1	15	1	11	1
Table Total	12455	0.307	10169	0.30599999999999999	13488	0.37
NA	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
Metro	4935	0.16200000000000001	4199	0.16700000000000001	6422	0.2310000000
Non-Metro	7520	0.74	5970	0.73699999999999999	7066	0.8159999999
Table Total	12455	0.307	10169	0.30599999999999999	13488	0.37

6 rows | 1-8 of 175 columns

Understand

```
# On the end of the column, there are 4 columns will not be used.
#They are summarised data which need to be deleted.
#dropping th last 4 rows
rent <- rent[-c(83:nrow(rent)), ]
#We realise the column name does work well with the data,
#Therefore, mannually add in the column names.
#Dropping the first column which is the big label of the dataset
rent <- rent[-1,]
colnames(rent) <- c('Council_Name',
                    'Mar_2000_Affordable','Mar_2000_Percent',
                    'Jun_2000_Affordable','Jun_2000_Percent',
                    'Sep_2000_Affordable','Sep_2000_Percent',
                    'Dec_2000_Affordable','Dec_2000_Percent',
                    'Mar_2001_Affordable','Mar_2001_Percent',
                    'Jun_2001_Affordable','Jun_2001_Percent',
                    'Sep_2001_Affordable','Sep_2001_Percent',
                    'Dec_2001_Affordable','Dec_2001_Percent',
                    'Mar_2002_Affordable','Mar_2002_Percent',
                    'Jun_2002_Affordable','Jun_2002_Percent',
                    'Sep_2002_Affordable','Sep_2002_Percent',
                    'Dec_2002_Affordable','Dec_2002_Percent',
                    'Mar_2003_Affordable','Mar_2003_Percent',
                    'Jun_2003_Affordable','Jun_2003_Percent',
                    'Sep_2003_Affordable','Sep_2003_Percent',
                    'Dec_2003_Affordable','Dec_2003_Percent',
                    'Mar_2004_Affordable','Mar_2004_Percent',
                    'Jun_2004_Affordable','Jun_2004_Percent',
                    'Sep_2004_Affordable','Sep_2004_Percent',
                    'Dec_2004_Affordable','Dec_2004_Percent',
                    'Mar_2005_Affordable','Mar_2005_Percent',
                    'Jun_2005_Affordable','Jun_2005_Percent',
                    'Sep_2005_Affordable','Sep_2005_Percent',
                    'Dec_2005_Affordable','Dec_2005_Percent',
                    'Mar_2006_Affordable','Mar_2006_Percent',
                    'Jun_2006_Affordable','Jun_2006_Percent',
                    'Sep_2006_Affordable','Sep_2006_Percent',
                    'Dec_2006_Affordable','Dec_2006_Percent',
                    'Mar_2007_Affordable','Mar_2007_Percent',
                    'Jun_2007_Affordable','Jun_2007_Percent',
                    'Sep_2007_Affordable','Sep_2007_Percent',
                    'Dec_2007_Affordable','Dec_2007_Percent',
                    'Mar_2008_Affordable','Mar_2008_Percent',
                    'Jun_2008_Affordable','Jun_2008_Percent',
                    'Sep_2008_Affordable','Sep_2008_Percent',
                    'Dec_2008_Affordable','Dec_2008_Percent',
                    'Mar_2009_Affordable','Mar_2009_Percent',
                    'Jun_2009_Affordable','Jun_2009_Percent',
                    'Sep_2009_Affordable','Sep_2009_Percent',
                    'Dec_2009_Affordable','Dec_2009_Percent',
                    'Mar_2010_Affordable','Mar_2010_Percent',
                    'Jun_2010_Affordable','Jun_2010_Percent',
                    'Sep_2010_Affordable','Sep_2010_Percent',
                    'Dec_2010_Affordable','Dec_2010_Percent',
                    'Mar_2011_Affordable','Mar_2011_Percent',
                    'Jun_2011_Affordable','Jun_2011_Percent',
```

```
                    'Sep_2011_Affordable','Sep_2011_Percent',
                    'Dec_2011_Affordable','Dec_2011_Percent',
                    'Mar_2012_Affordable','Mar_2012_Percent',
                    'Jun_2012_Affordable','Jun_2012_Percent',
                    'Sep_2012_Affordable','Sep_2012_Percent',
                    'Dec_2012_Affordable','Dec_2012_Percent',
                    'Mar_2013_Affordable','Mar_2013_Percent',
                    'Jun_2013_Affordable','Jun_2013_Percent',
                    'Sep_2013_Affordable','Sep_2013_Percent',
                    'Dec_2013_Affordable','Dec_2013_Percent',
                    'Mar_2014_Affordable','Mar_2014_Percent',
                    'Jun_2014_Affordable','Jun_2014_Percent',
                    'Sep_2014_Affordable','Sep_2014_Percent',
                    'Dec_2014_Affordable','Dec_2014_Percent',
                    'Mar_2015_Affordable','Mar_2015_Percent',
                    'Jun_2015_Affordable','Jun_2015_Percent',
                    'Sep_2015_Affordable','Sep_2015_Percent',
                    'Dec_2015_Affordable','Dec_2015_Percent',
                    'Mar_2016_Affordable','Mar_2016_Percent',
                    'Jun_2016_Affordable','Jun_2013_Percent',
                    'Sep_2016_Affordable','Sep_2016_Percent',
                    'Dec_2016_Affordable','Dec_2016_Percent',
                    'Mar_2017_Affordable','Mar_2017_Percent',
                    'Jun_2017_Affordable','Jun_2017_Percent',
                    'Sep_2017_Affordable','Sep_2017_Percent',
                    'Dec_2017_Affordable','Dec_2017_Percent',
                    'Mar_2018_Affordable','Mar_2018_Percent',
                    'Jun_2018_Affordable','Jun_2018_Percent',
                    'Sep_2018_Affordable','Sep_2018_Percent',
                    'Dec_2018_Affordable','Dec_2018_Percent',
                    'Mar_2019_Affordable','Mar_2019_Percent',
                    'Jun_2019_Affordable','Jun_2019_Percent',
                    'Sep_2019_Affordable','Sep_2019_Percent',
                    'Dec_2019_Affordable','Dec_2019_Percent',
                    'Mar_2020_Affordable','Mar_2020_Percent',
                    'Jun_2020_Affordable','Jun_2020_Percent',
                    'Sep_2020_Affordable','Sep_2020_Percent',
                    'Dec_2020_Affordable','Dec_2020_Percent',
                    'Mar_2021_Affordable','Mar_2021_Percent',
                    'Jun_2021_Affordable','Jun_2021_Percent',
                    'Sep_2021_Affordable','Sep_2021_Percent')
#After adding in the new colnames
#the old 2 rows need to be removed.
rent <- rent[-c(1,2),]
```

Data Background Data used in this report are found on the Victoria government data website. The first one is the quarterly rental report on affordable letting sorted by local government area from 2000 to 2021. The second is harm caused by crime statistics sorted by local government area from 2012 December to 2021 December.

Rental report quarterly affordable letting by LGA The rental report data is about cheap affordable housing available to rent within the local government governed area and the percentage of affordable housing within the local area rental market. The data is sorted into quarterly years with a separation of March, June, September, and December, from March 2000 to 2021 September. Under each month of the year, there are Affordable values and Percentage Values. Affordable value is the count of affordable housing within the local government area, and the percentage is proportionality within the local government area rental market.

Crime statics agency data table harm caused by LGA
The crime statistic data is about the number of harms been caused by crime reported sorted by local government area by the end of the year from 2012 to 2021. The data column includes the year, year ending, police region, local government area, harm caused flag and victim reports. The year consists of 2012 to 2021; the year ending only contains December, which is the reporting period for such a report. Local government includes 79 different local council names. Police regions are 4 different levels that indicate the different areas of the police force. Harm caused flag is a rating from High to medium-low, indicating different crimes. Victims' reports are the number of sufferers who suffered from the crime.

Tidy & Manipulate Data I

```
# First convert the VictimReports data to wide format
VictimReports <- pivot_wider(VictimReports, names_from = "Harm Caused Flag", values_from = "Vic
tim Reports")
# Because in our survey, we don't need to discuss whether the harm is high or low, a new variab
le needs to be created to count the sum
VictimReports <- mutate(VictimReports, NumberOfVictim = `High Harm` + `Low Harm` + `Medium Harm
`)
# Also delete the existing High Harm, low Harm, Medium Harm variables
VictimReports <- subset(VictimReports, select = -c(`High Harm`, `Low Harm`, `Medium Harm`))

# The rent data is not tidy due to the multiple variable are stored in rows
# First convert the whole data set into individual column contain majority of
# the variable.
rent <- rent %>% pivot_longer(!Council_Name, names_to= "date_ca",
                              values_to = "percentage_af")
# Now the data set is untidy due to multiple variable are stored in  one column
# The time is mixed with the year and other things, it need to be separated
rent <- separate(rent, col=date_ca,
                 into=c('Month','Year','Afford_Percent'), sep='_')
# Now finally separate the Afford_Percent by creating a
# unique identifier due to some values are repeated
# and pivot_wider cannot processed and comes back with
# warning
rent <- rent %>%
  group_by(Afford_Percent) %>%
  mutate(row = row_number()) %>%
  tidyr::pivot_wider(names_from = Afford_Percent, values_from = percentage_af) %>%
  select(-row)
# Changing to different type of data and rounding up decemal places.
rent$Affordable <- as.integer(rent$Affordable)
rent$Percent <- as.numeric(rent$Percent)
rent$Percent <- round(rent$Percent , digit=2)
```

First of all for the VictimReport table, we don't need to discuss whether the harm is high or low, what we need is an overall Victim number, so we use the pivot_wider function to convert from long data to wide data Then use the mutate function to integrate and create new variables Finally delete the unneeded variables with subset.

For rent data, we first convert the entire dataset into a single column containing most of the variables. Then use the separate function to separate the time and year etc from date_ca Since it is now the last to separate Afford_Percent by creating a unique identifier, pivot_wider cannot handle it and returns a warning because some values are duplicated Finally, convert Affordable and percent into the corresponding data format

Tidy & Manipulate Data II

```
VictimReports <- VictimReports %>% filter(Year < 2021)
VictimReports$`Local Government Area` <- as.factor(VictimReports$`Local Government Area`)
# Because the minimum year of VictimReports is in 2012, we filter out the years less than 2012
 in the rent table.
rent <- rent %>% filter(Year >2011)
# Because the Dec of the two sets of data is in years, the Mar, Jun, and Sep in the rent table
 are actually useless data for us, so filter out
rent <- rent %>% filter(Month == "Dec")
# Here, the rename function in the plyr package is used to rename the VictimReport data
# so that the key names of the two tables are unified for the next join
library(plyr)
```

```
## Warning: package 'plyr' was built under R version 4.1.3
```

```
## ------------------------------------------------------------------------------
```

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

```
VictimReports <- rename(VictimReports, c(`Local Government Area` = "Council_Name"))
# The reason why plyr is canceled here is because many functions in plyr and dplyr conflict.
# If it is not canceled, the next functions will call the plyr package, which will cause code e
rrors.
detach("package:plyr", unload = TRUE)
# In order to join the two tables, year and Council_Name need to have the same data structure
# so here the year in VictimReport data is converted into a character structure
VictimReports$Year <- as.character(VictimReports$Year)
# Arrange the two tables in positive order by region name and year to achieve data alignment
VictimReports <- VictimReports %>% arrange(Year, Council_Name)
rent <- rent %>% arrange(Year, Council_Name)
# Perform Join operation
RentForVictim <- left_join(VictimReports, rent, by = c("Council_Name", "Year"))
# Because our data unit is one year, the Month and Year ending variables are not necessary her
e, delete them
RentForVictim <- subset(RentForVictim, select = -c(Month, `Year ending`))
head(RentForVictim)
```

Year	Police Region	Council_Name	NumberOfVictim	Affordable	Percent
<chr>	<chr>	<chr>	<dbl>	<int>	<dbl>
2012	2 Eastern	Alpine	146	69	0.80
2012	4 Western	Ararat	574	69	0.85
2012	4 Western	Ballarat	5243	597	0.56
2012	1 North West Metro	Banyule	3512	37	0.04
2012	2 Eastern	Bass Coast	1186	266	0.66
2012	2 Eastern	Baw Baw	1263	180	0.70

6 rows

First, we use filter to filter out the data in the rent data that is less than 2011 and Month is not equal to Dec, so that the correspondence between VictimReport data and rent data can be achieved. After that, we used the rename function in the plyr package to make the VictimReport data and rent data have the same variable name for better JOIN operation The plyr package needs to be canceled here, because the plyr package and the dplyr package have many functions that are the same, which will cause conflicts. For the next operation, the loading of the plyr package must be canceled. Here we need two keys to link the two data, so convert the Year in VictimReport to character format to achieve data correspondence. Then sort the two tables in ascending order by Year and Council_Name variables Finally, perform left_join to merge Because we observed that the amount of data is the same after processing the data on both sides, we can use left_join directly Because Year, Year ending, Month, Month variables have overlapping parts, we delete the overlapping variables Finally, the header of the sorted data is displayed

Scan I

```
# First look at the statistics of the two tables
library(editrules)
```

```
## Warning: package 'editrules' was built under R version 4.1.3
```

```
## Loading required package: igraph
```

```
## Warning: package 'igraph' was built under R version 4.1.3
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
```

```
## The following object is masked from 'package:tidyr':
##
##     crossing
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

```
##
## Attaching package: 'editrules'
```

```
## The following objects are masked from 'package:igraph':
##
##     blocks, normalize
```

```
## The following object is masked from 'package:dplyr':
##
##     contains
```

```
## The following objects are masked from 'package:tidyr':
##
##     contains, separate
```

```
summary(RentForVictim$NumberOfVictim)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      49     431    1494    2734    4443   13872
```

```
summary(RentForVictim$Affordable)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     0.0    22.0    52.0   114.6   135.0  1098.0       9
```

```
summary(RentForVictim$Percent)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.0600  0.4450  0.4330  0.7275  1.0000       9
```

```
# Count how many rows have NA values
sum(is.na(RentForVictim))
```

```
## [1] 18
```

```
# Determine the position of the row where these NA values are located
which(is.na(RentForVictim))
```

```
##  [1] 2897 2976 3055 3134 3213 3292 3371 3450 3529 3608 3687 3766 3845 3924 4003
## [16] 4082 4161 4240
```

```
# Calculate the percentage of these NA values
18/711
```

```
## [1] 0.02531646
```

```
# The result is 3%, because it does not exceed 5%, so excluding these Missing data will not cau
se a big deviation in the data
```

```
# So here is the operation to exclude Missing data
RentForVictim <- na.omit(RentForVictim)
# Use sapply and the function we created to check for special values
sum(sapply(RentForVictim, is.infinite))
```

```
## [1] 0
```

```
sum(sapply(RentForVictim, is.finite))
```

```
## [1] 2106
```

```
sum(sapply(RentForVictim, is.nan))
```

```
## [1] 0
```

```
# To keep the data accurate, three rules were created to keep my data from exceeding the criter
ia for each variable
(Rule1 <- editset(c("Percent >= 0", "Percent <= 1")))
```

```
##
## Edit set:
## num1 : 0 <= Percent
## num2 : Percent <= 1
```

```
(Rule2 <- editset(c("NumberOfVictim >= 0")))
```

```
##
## Edit set:
## num1 : 0 <= NumberOfVictim
```

```
(Rule3 <- editset(c("Affordable >= 0")))
```

```
##
## Edit set:
## num1 : 0 <= Affordable
```

```
sum(violatedEdits(Rule1, RentForVictim))
```

```
## [1] 0
```

```
sum(violatedEdits(Rule2, RentForVictim))
```

```
## [1] 0
```

```
sum(violatedEdits(Rule3, RentForVictim))
```

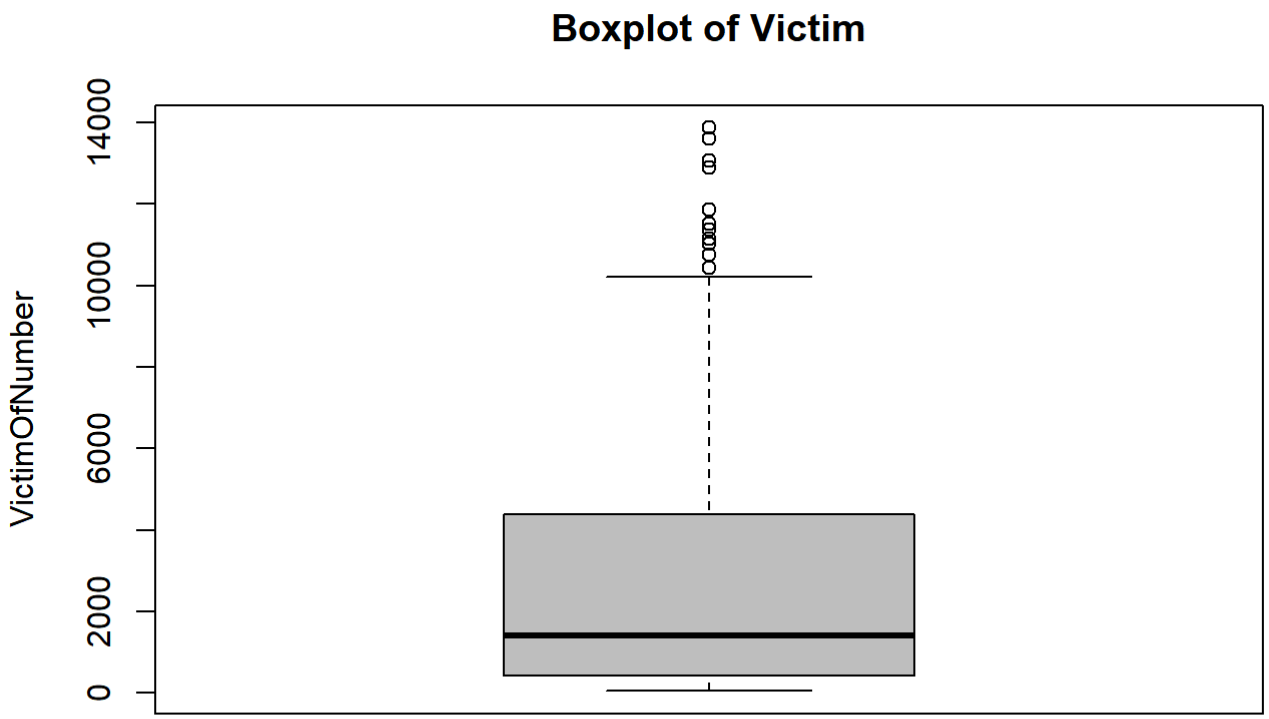
```
## [1] 0
```

```
# The results are all 0, proving that the current data is all good
```

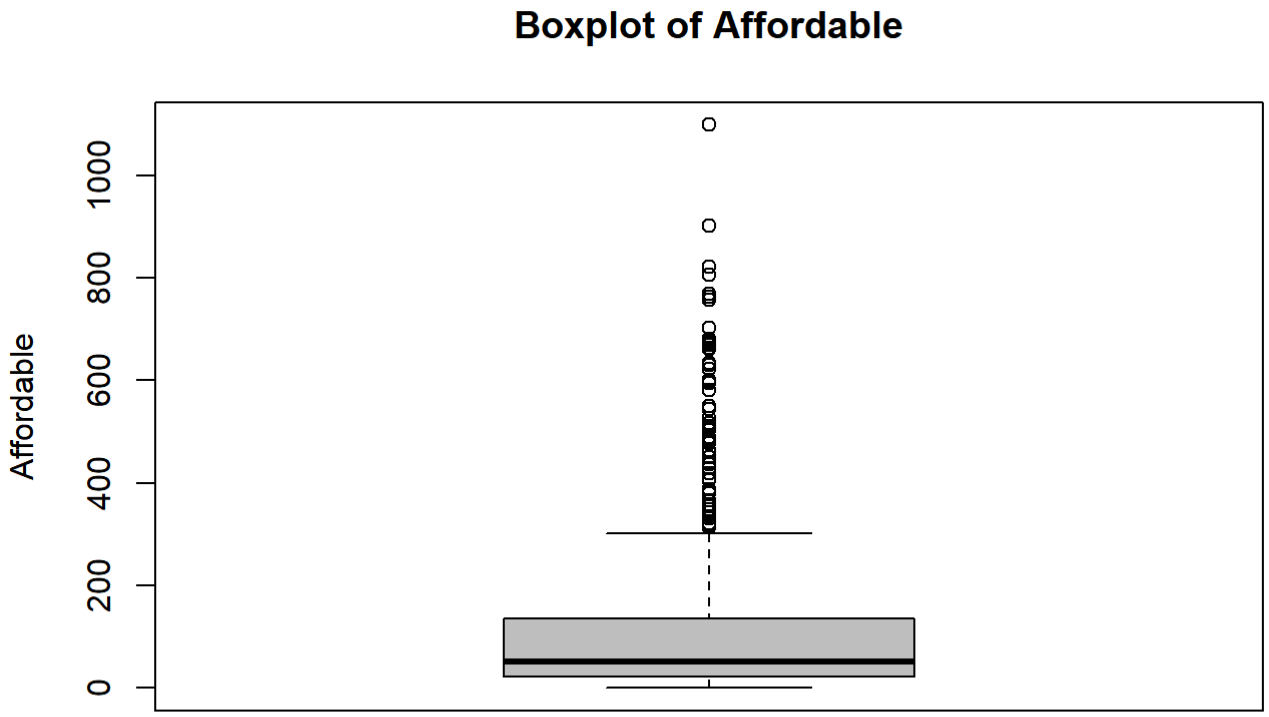
Here first load the editrules package for subsequent data scanning Also here are statistics for NumberOfVictim, Percent and Affordable variables Next, scan for the presence of NA data and determine the location of the NA data Then use 18/711 to judge the percentage of NA to determine our next judgment. The percentage is around 2.5%, not reaching 5%, so even removing the NA value will not have a great impact on the data. So use na.omit to remove NA values. After inspection, it is found that there are no outliers. Then I need to create rules with rules in the editrules package to ensure that the code is within the standards of my rules.

Scan II

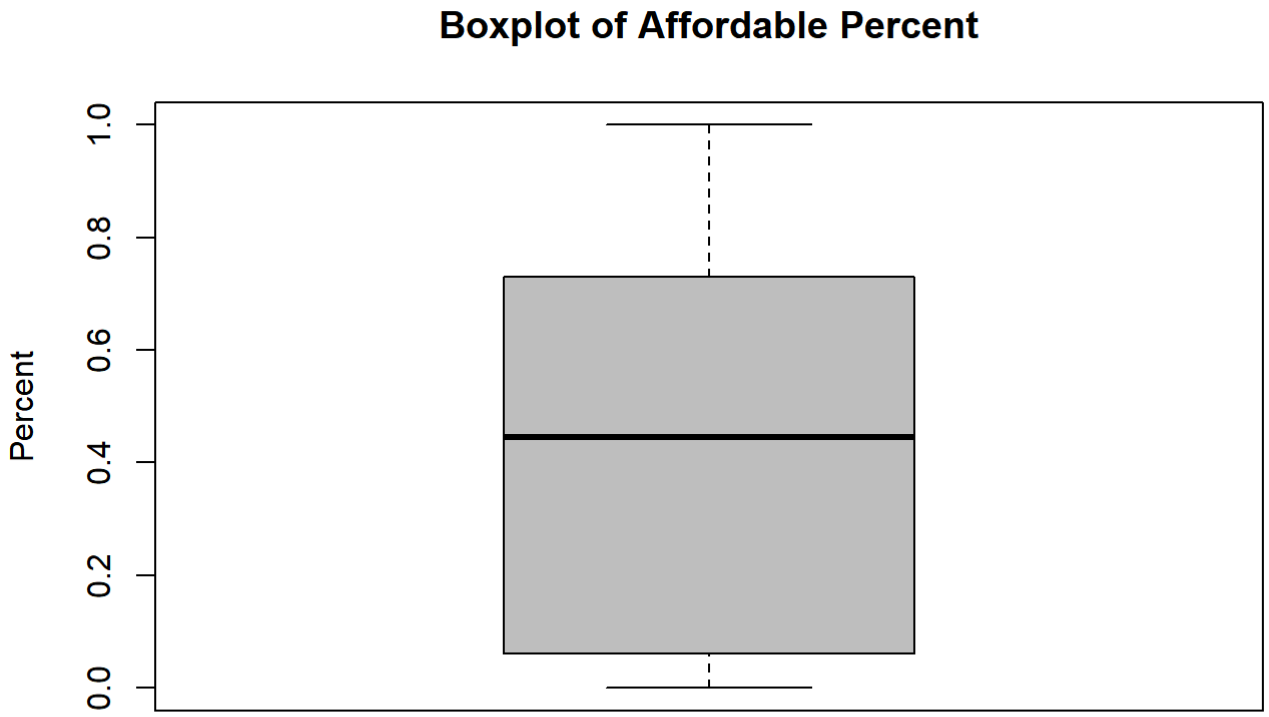
```
# First look at the boxplot of three numeric variables
RentForVictim$NumberOfVictim %>% boxplot(main = "Boxplot of Victim",
                                ylab = "VictimOfNumber",
                                col = "grey")
```




```
RentForVictim$Affordable %>% boxplot(main = "Boxplot of Affordable",
                            ylab = "Affordable",
                            col = "grey")
```



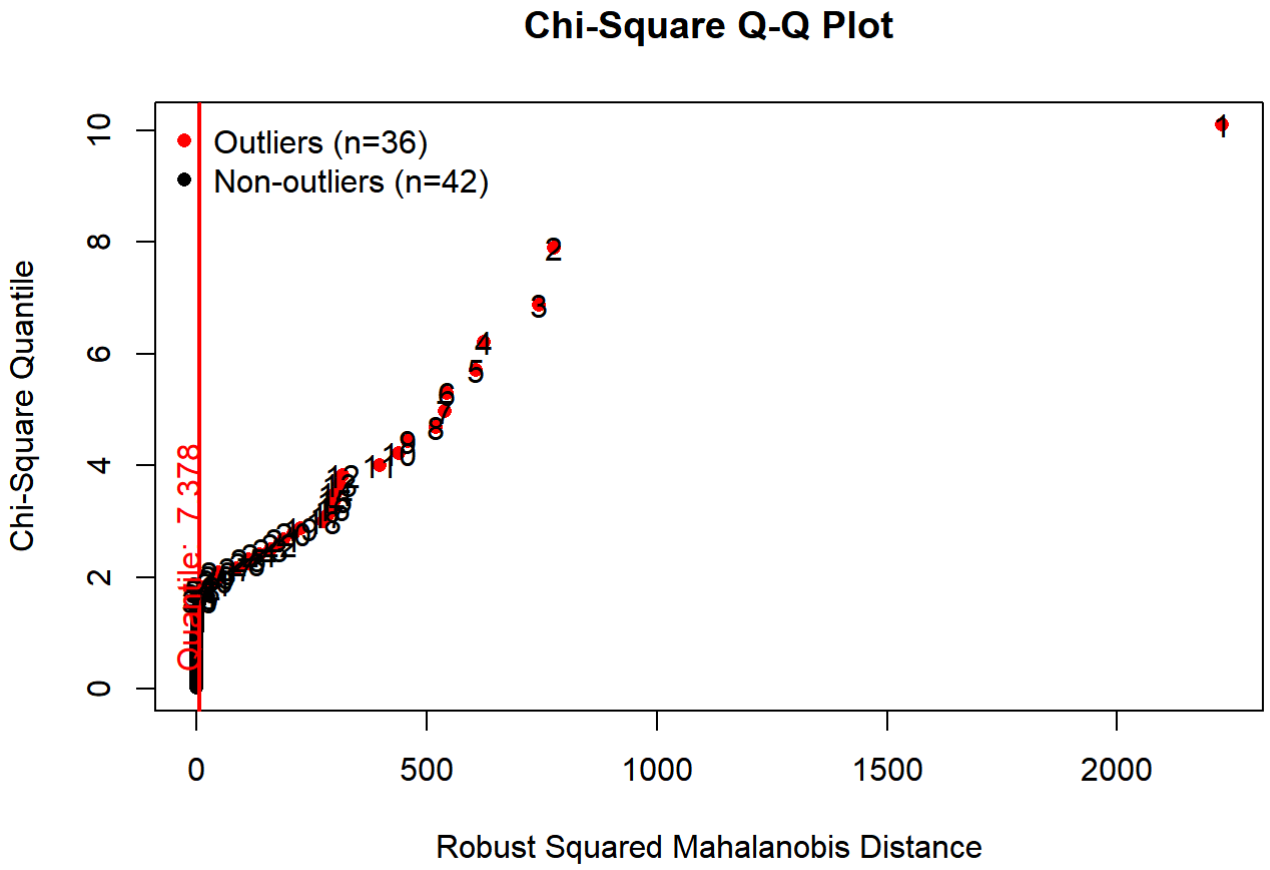

```
RentForVictim$Percent %>% boxplot(main = "Boxplot of Affordable Percent",
                          ylab = "Percent",
                          col = "grey")
```



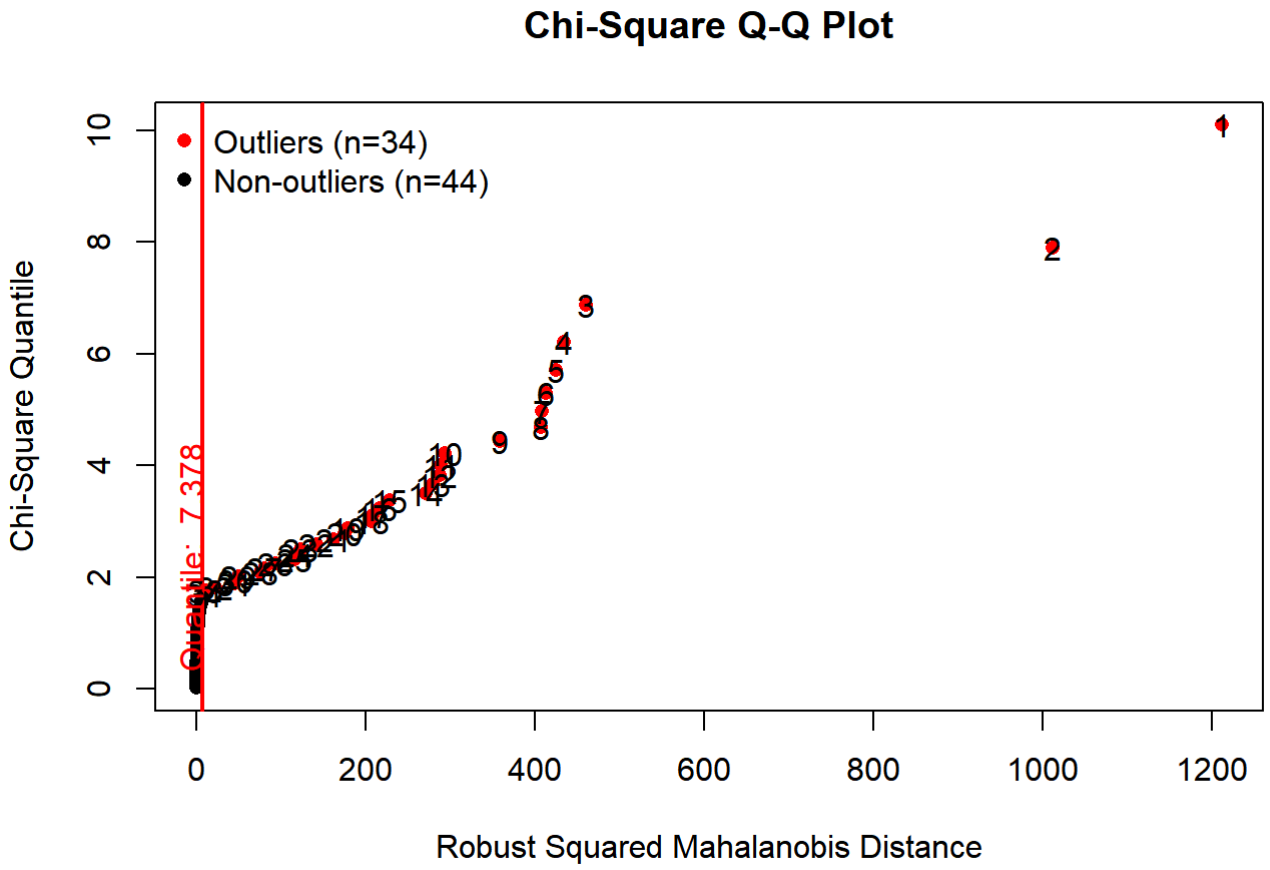

```
# It can be seen that in addition to Percent, Affordable and NumberOfVictim still have many out
liers
# MVN package is used here to detect multivariate outliers
library(MVN)
```

```
## Warning: package 'MVN' was built under R version 4.1.3
```

```
versicolor2012 <- RentForVictim %>% filter(Year == "2012") %>%
  dplyr::select("NumberOfVictim", "Affordable")
results <- mvn(data = versicolor2012, multivariateOutlierMethod = "quan",
               showOutliers = TRUE)
```




```
versicolor2020 <- RentForVictim %>% filter(Year == "2020") %>%
  dplyr::select("NumberOfVictim", "Affordable")
results <- mvn(data = versicolor2020, multivariateOutlierMethod = "quan",
               showOutliers = TRUE)
```




```
# Here we find that no matter which year it is, there are a lot of outliers
# so we can't directly delete outliers here, which will destroy the data
# So I chose to cap it using a capping technique that replaces observations outside the lower b
ound with values at the 5th percentile and values above the upper bound with values at the 95th
percentile.
# create the function
cap <- function(x){
  quantiles <- quantile(x, c(0.5, 0.25, 0.75, .95))
  x[x < quantiles[2] - 1.5*IQR(x)] <- quantiles[1]
  x[x > quantiles[3] + 1.5*IQR(x)] <- quantiles[4]
  x
}

# Extract subsets and observe changes
RentForVictim_sub <- RentForVictim %>% dplyr::select(NumberOfVictim, Affordable)
summary(RentForVictim_sub)
```

```
##  NumberOfVictim      Affordable
##  Min.    :   49.0   Min.    :   0.0
##  1st Qu.:  422.5   1st Qu.:  22.0
##  Median : 1411.5   Median :  52.0
##  Mean    : 2708.6   Mean    : 114.6
##  3rd Qu.: 4366.5   3rd Qu.: 135.0
##  Max.    :13872.0   Max.    :1098.0
```

```
RentForVictim_capped <- sapply(RentForVictim_sub, FUN = cap)
summary(RentForVictim_capped)
```

```
##  NumberOfVictim      Affordable
##  Min.   :   49.0   Min.   :  0.0
##  1st Qu.:  422.5   1st Qu.: 22.0
##  Median : 1411.5   Median : 52.0
##  Mean   : 2651.9   Mean   :111.2
##  3rd Qu.: 4366.5   3rd Qu.:135.0
##  Max.   :10215.0   Max.   :481.8
```

For the second part of the scan, I first created boxplots of the three variables and observed that most of the outliers were concentrated in NumberOfVictim and Affordable. So next I use the MVN package to do multivariate outlier detection by year. I picked a representative beginning (2012) and ending (2021) for analysis Outliers occupy almost 50% of the data, so deletion cannot be performed. So create cap function to limit the value beyond the limit. Finally, use the cap function to change the data, and compare the data before the change.

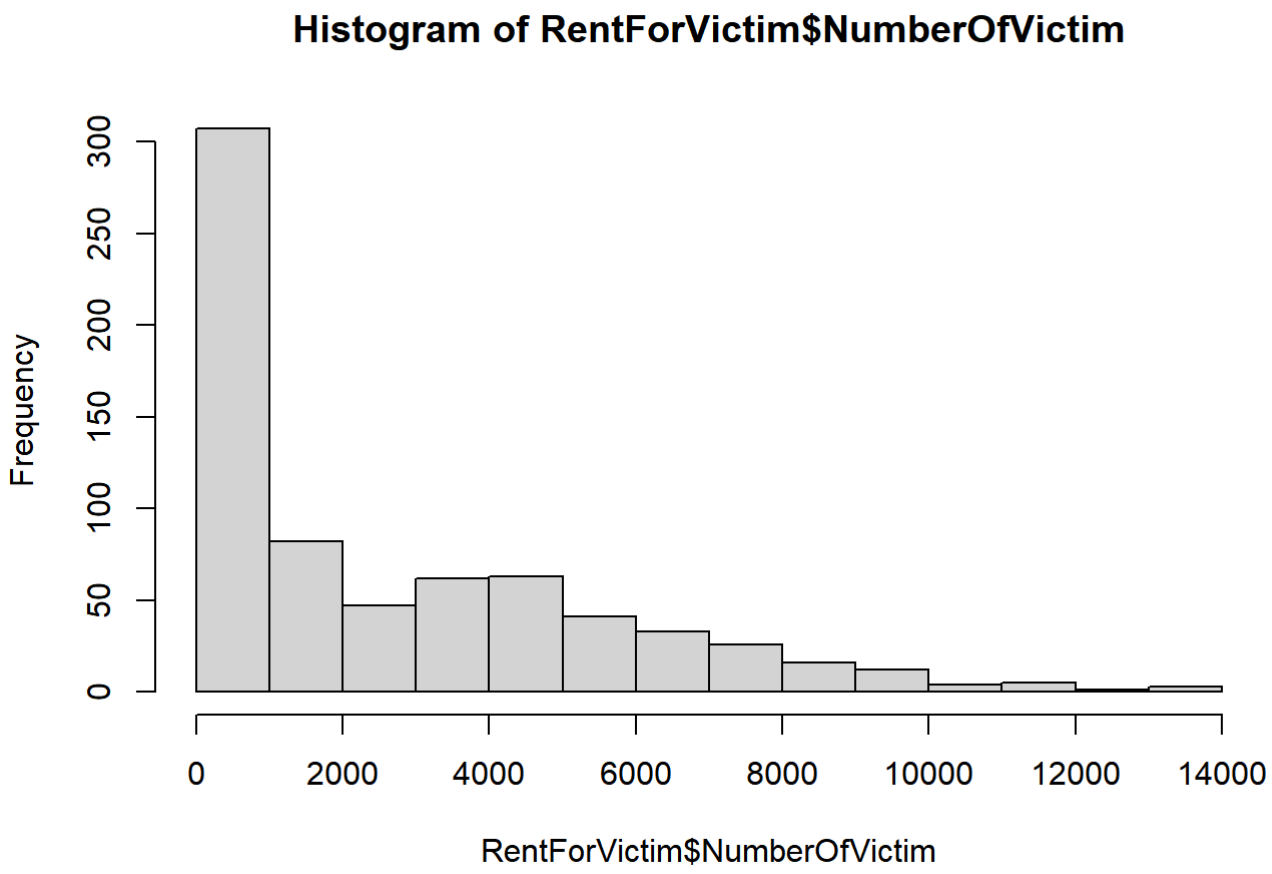
Transform

```
library(forecast)
```

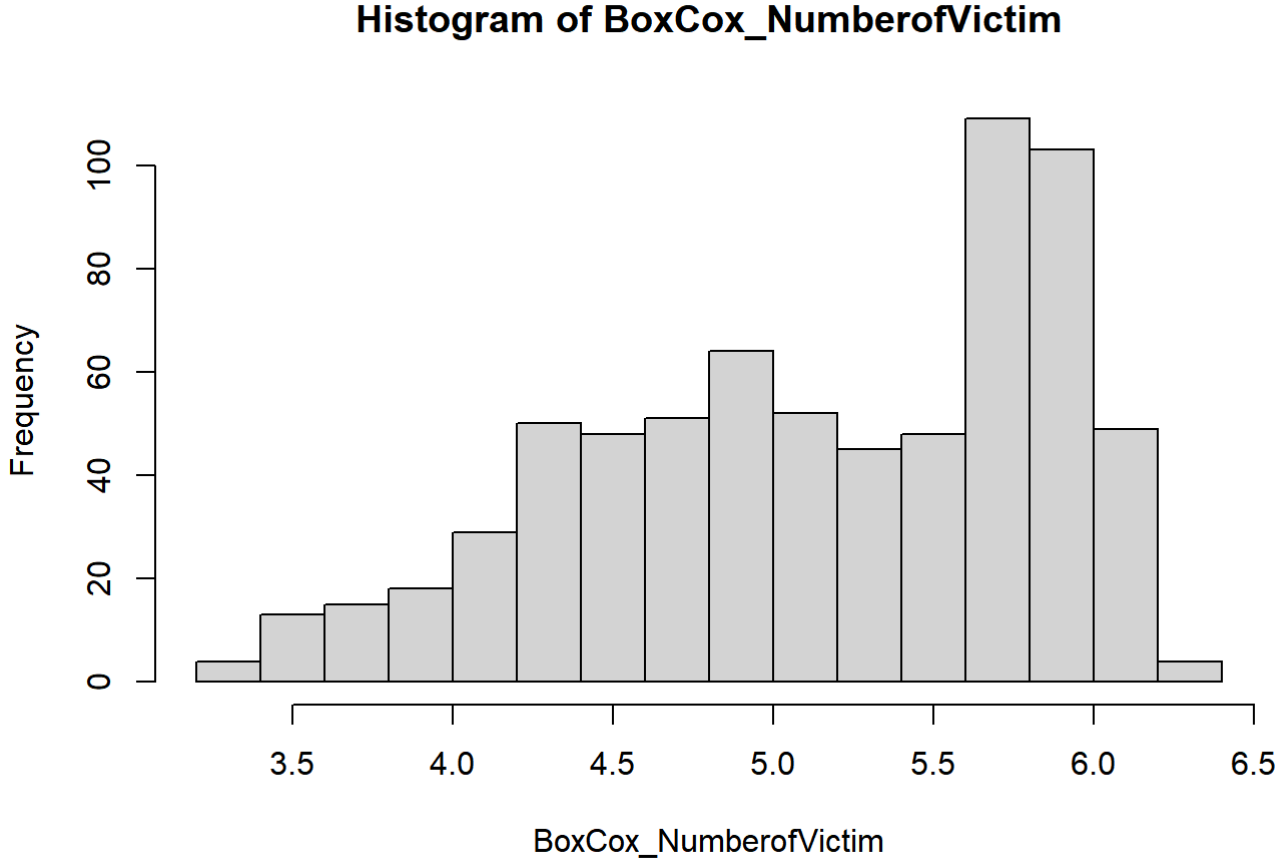
```
## Warning: package 'forecast' was built under R version 4.1.3
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

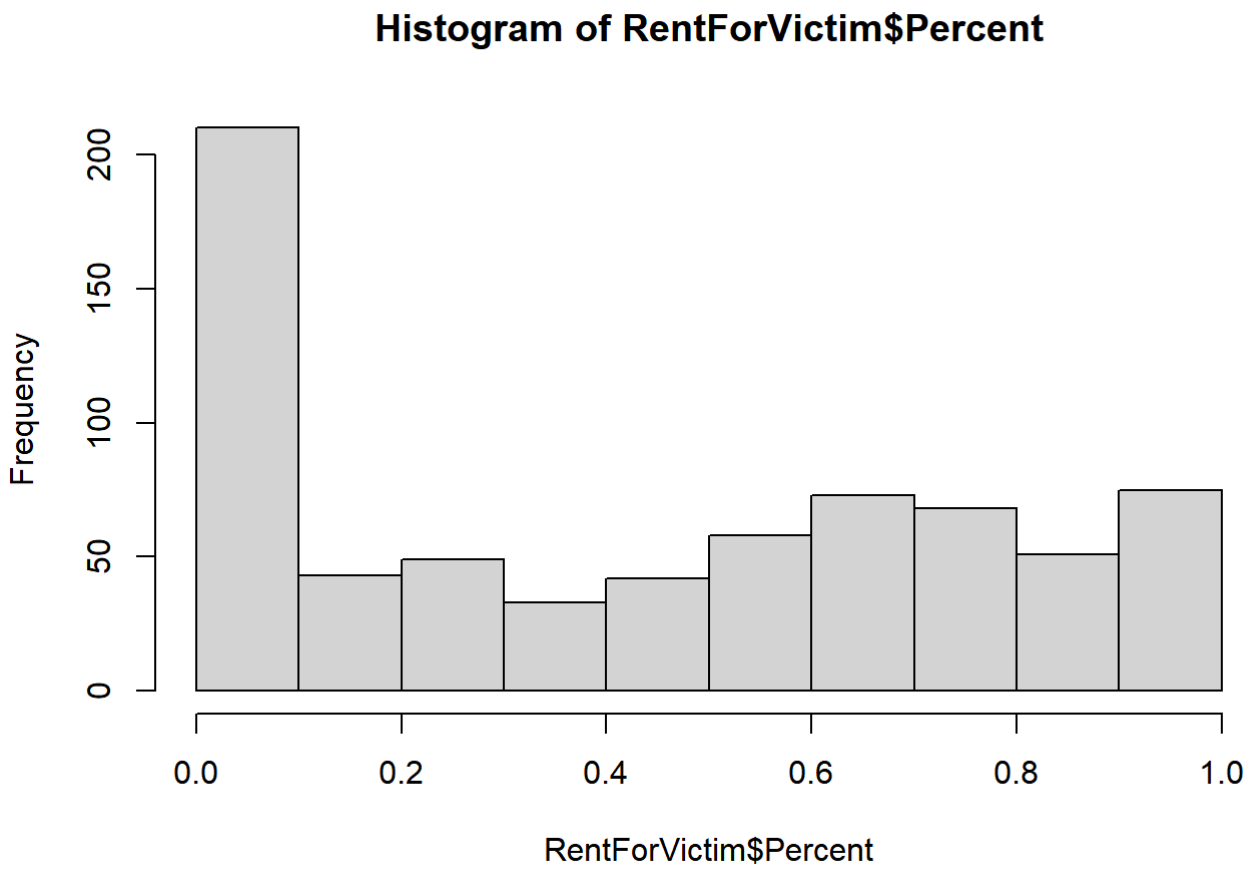
```
#For number of victims, the most effective method was BoxCox method. However, it is still unabl
e fully converted the skewness into symmetrical distribution. But, with the current knowledge,
 it is the best we can achieve.
hist(RentForVictim$NumberOfVictim)
```



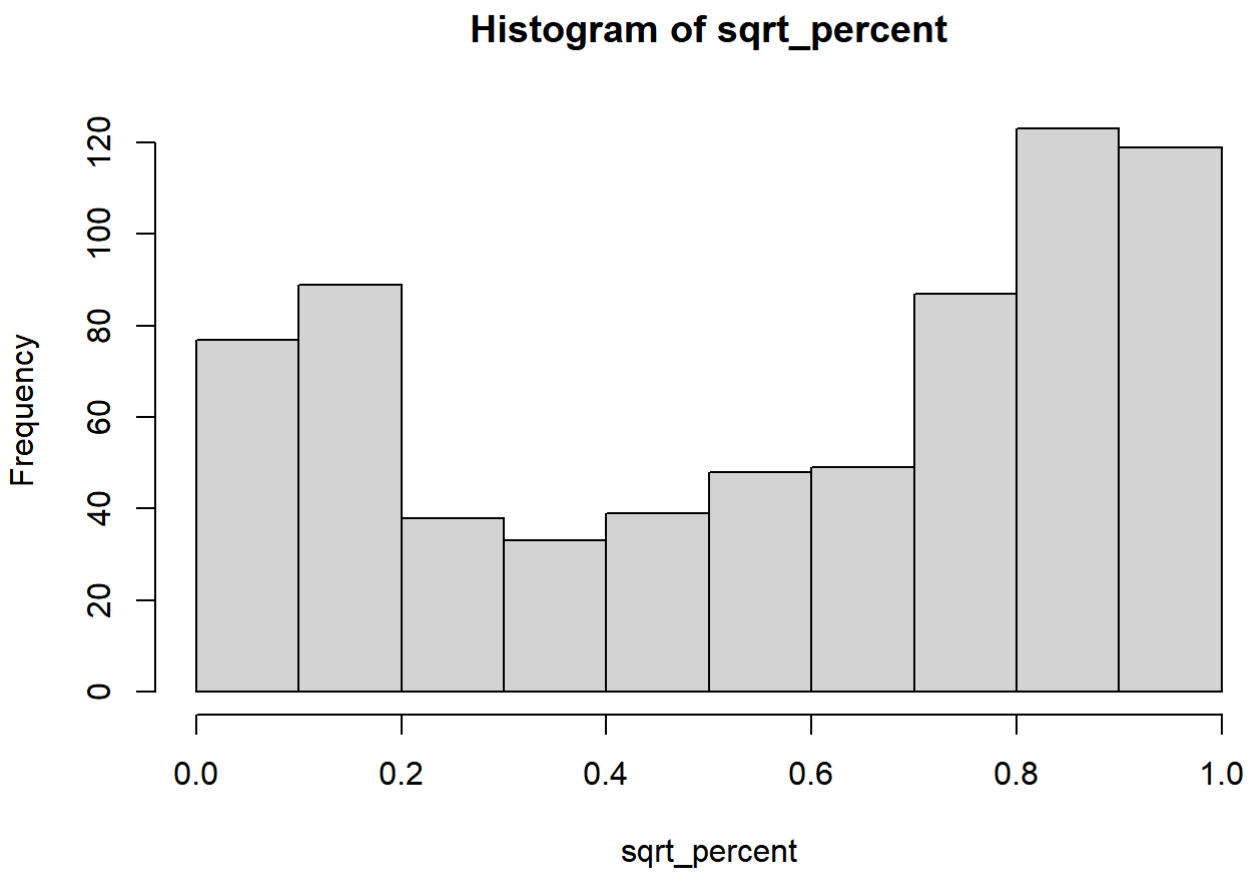

```
BoxCox_NumberofVictim <- BoxCox(RentForVictim$NumberOfVictim, lambda = "auto")
hist(BoxCox_NumberofVictim )
```



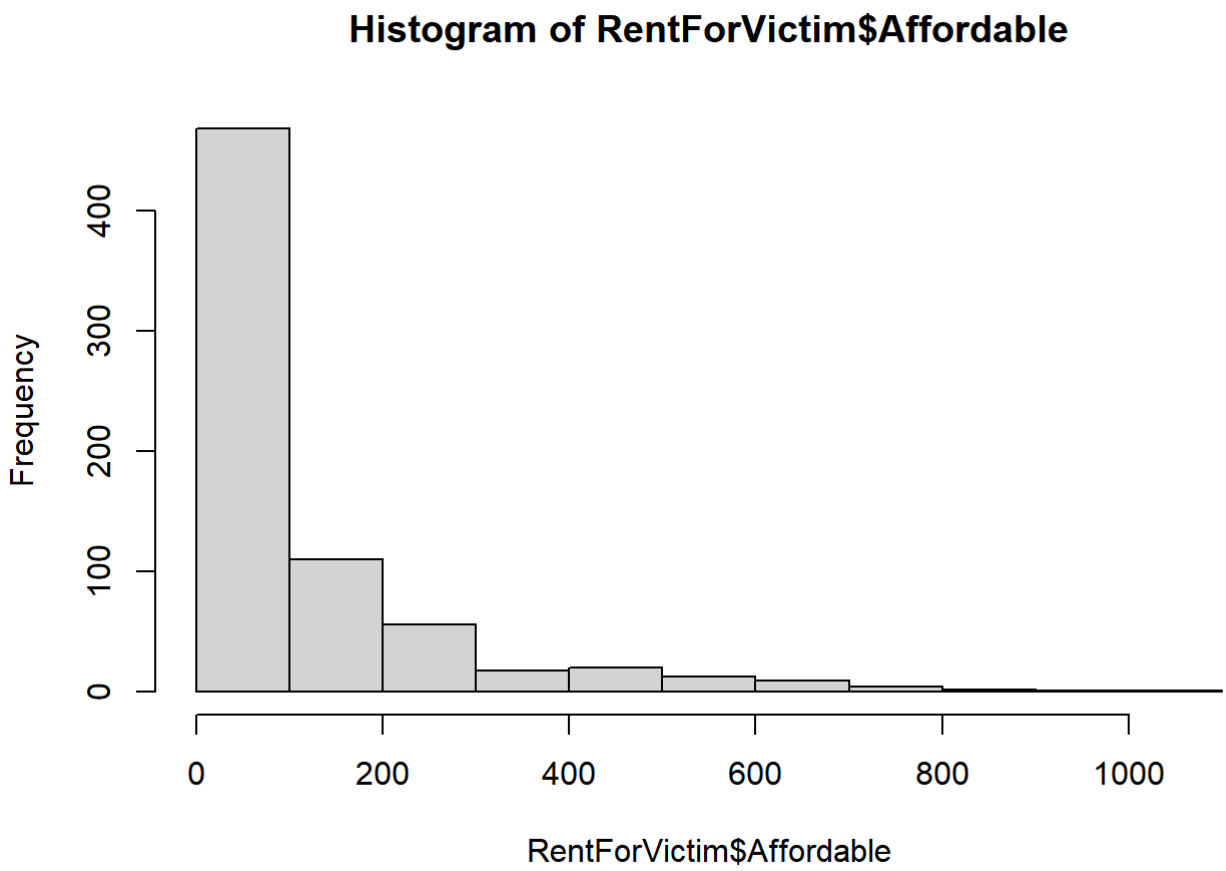
```
#For the Percent. The most effective method is square root transformation method. However, this method kind of making the skewness into more left skewed rather than symmetrical. We tempt other method, BoxCox transformation and log10 transformation does not give good result.
hist(RentForVictim$Percent)
```



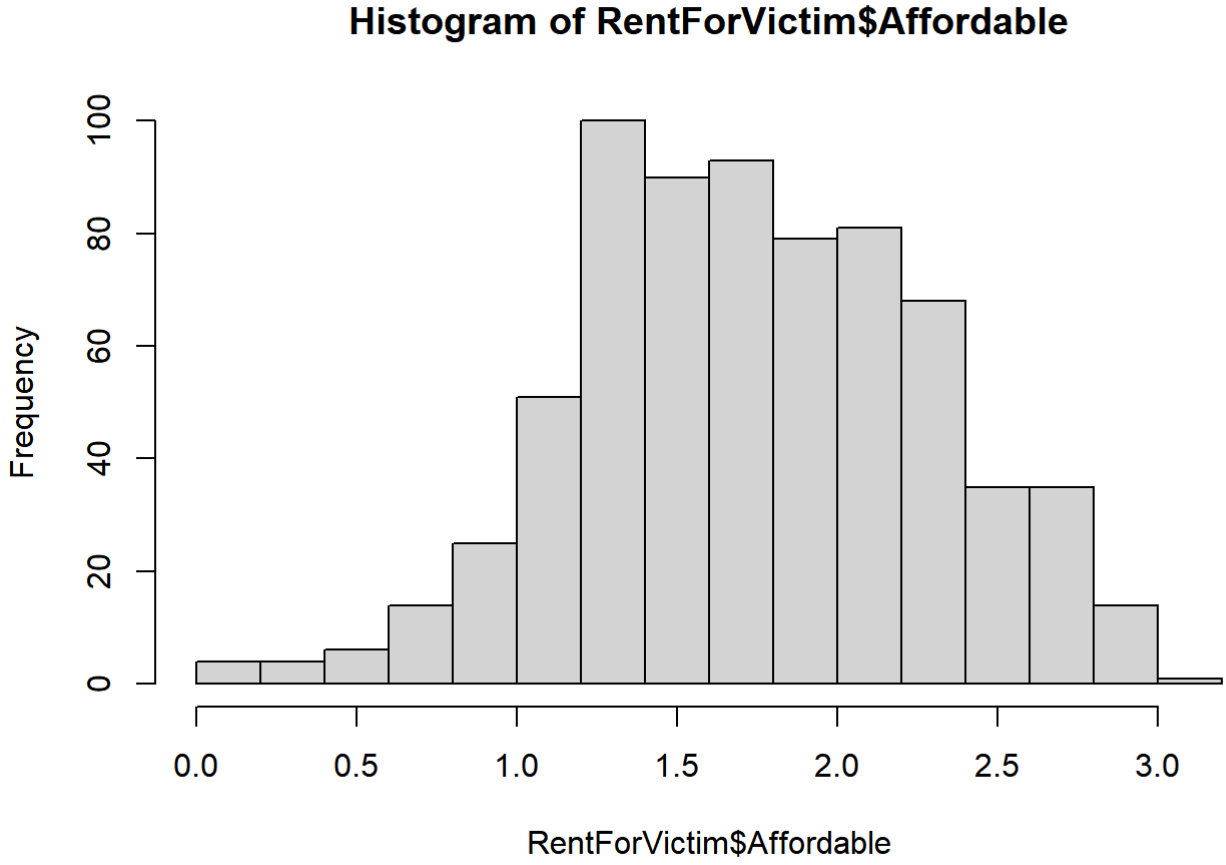

```
sqrt_percent<-sqrt(RentForVictim$Percent)
hist(sqrt_percent)
```




```
# Affordable
# For affordable column, the most effective was log10 transformation. As the graph shown that i
t effectively correct skewness into symmetrical distribution.
hist(RentForVictim$Affordable)
```




```
RentForVictim$Affordable <- log10(RentForVictim$Affordable)
hist(RentForVictim$Affordable)
```



We checked the skewness of three variables within the merged dataset. The number of victims, Affordable and Percent. All three of the variables have the trend of right skewness. This will impact any further model making causing bias. We performed square root, Box Cox also log 10 transformation on individual variables. For number of victims, the most effective method was BoxCox method. However, it is still unable fully converted the skewness into symmetrical distribution. But, with the current knowledge, it is the best we can achieve. For affordable column, the most effective was log10 transformation. As the graph shown that it effectively correct skewness into symmetrical distribution. For the Percent. The most effective method is square root transformation method. However, this method kind of making the skewness into more left skewed rather than symmetrical. We tempt other method, BoxCox transformation and log10 transformation does not give good result.