Data Wrangling (Data Preprocessing)

Practical assessment 2

Charlie Lock

Setup

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

library(kableExtra)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 3 > 1' in coercion to 'logical(1)'
```

```
library(magrittr)
library(dplyr)
```

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

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

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

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

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:magrittr':
##
##     extract
```

```
library(outliers)
```

Student names, numbers and percentage of contributions

Group information

Student name	Student number	Percentage of contribution
Charlie Lock	s3785677	100%

Executive Summary

- Began by downloading the csv files from the given sources and reading them into this report using the read.csv() function.
- Identifying the variables and giving a brief explanation into what each variable represents
- Next, I've used the str() and class() functions to identify the data type of each variable as well as the structure of the data set
- As the "Number" variable from the road accidents data set was a character data type the necessary steps were taken to convert it to a numeric data type using the as.numeric() function. The gsub() function was also used to remove the commas from the values so the variable could be converted without an error.
- The next step involved selecting the variables that would be used and filtering out the observations that contained values that were not required.
- It also involved using the gather() function to convert the data set from wide format into long format as the "sex" variable was wrongly separated into two columns.
- Both data sets had to be filtered for different reasons. The road accidents data set had to be filtered to only contain observations from 2016 and the population projection data set had to be filtered to only contain observations with the "Male" and "Female" values in the "Sex" variable. *A new variable was created using the mutate() function.* The now merged data set was scanned for missing values and the observations with missing values were removed.
- An analysis using various graphs and functions was completed to find outliers.
- Finally, the data was scaled using a logarithmic transformation to fix the skewed data and create a new data set without outliers.

Data

The first data set that will be merged into one data set is the road traffic accidents data set which has been broken down by country, sex, year and age. The source for this data set is the World Health Organisation (WHO). Source: https://platform.who.int/mortality/themes/theme-details/topics/indicator-groups/indicator-group-details/MDB/road-traffic-accidents (https://platform.who.int/mortality/themes/theme-details/topics/indicator-groups/indicator-group-details/MDB/road-traffic-accidents)

The second data set is a population projection data set by country. The source fpr this data set is the United Nations. Source: https://population.un.org/wpp/Download/Standard/CSV/ (https://population.un.org/wpp/Download/Standard/CSV/)

```
# Import the data, provide your R codes here.
road_accidents <- read.csv("/Users/charlielock/Desktop/UNI/2023 Sem1/Data Preprocessi
ng/PracAssignment2/road_traffic_deaths1.csv")
head(road_accidents)
```

	Region.Code <chr>	Region.Name <chr>	Country.Code <chr>	Country.Name <chr>	Y… <int>	Sex <chr>	Age.g <chr>
1	OA	Oceania	AUS	Australia	2016	All	Age_a
2	OA	Oceania	AUS	Australia	2016	Male	Age_a
3	OA	Oceania	AUS	Australia	2016	Female	Age_a
4	CSA	Central and South America	BRA	Brazil	2016	All	Age_a
5	CSA	Central and South America	BLZ	Belize	2016	All	Age_a
6	EU	Europe	BGR	Bulgaria	2016	All	Age_a

6 rows | 1-8 of 13 columns

```
population1 <- read.csv("/Users/charlielock/Desktop/UNI/2023 Sem1/Data Preprocessing/
PracAssignment2/population1.csv")
head(population1)
```

	SortOrder <int>	Lo… <int>	No… <chr>	ISO3_co… <chr>	ISO2_co… <chr>	SDMX_c… <int>	LocTypeID <int>	LocTypeNa… <chr>	ParentID <int>
1	*NA*	1857				*NA*	*NA*		*NA*
2	*NA*	1857				*NA*	*NA*		*NA*
3	*NA*	1857				*NA*	*NA*		*NA*
4	*NA*	1857				*NA*	*NA*		*NA*
5	*NA*	1857				*NA*	*NA*		*NA*
6	*NA*	1857				*NA*	*NA*		*NA*

6 rows | 1-10 of 19 columns

I uploaded the two data sets into R by initially downloading them from their source and saving them on my desktop. I then proceeded to use the read.csv() function to read them into R Markdown as shown.

Road Accident Variables:

- Region.Code - The code of the given region (e.g. EU is the code for Europe)
- Region.Name - The name of the region
- Country.Code - The code of the given country (e.g. AUS is the code for Australia)
- Country.Name - The name of the country in which the road traffic accident deaths occurred
- Year - The calendar year in which the road traffic accident deaths occurred
- Sex - The gender/sex. Only the female and male observations will be used in this report.
- Age.group.code - The code for the given age group
- Age.Group - The given age range (e.g. 15-24)
- Number - The number of deaths caused by road traffic incidents

- Percentage.of.cause.specific.deaths.out.of.total.deaths - The percentage of overall deaths caused by road traffic accidents *Age.standardized.death.rate.per.100.000.standard.population - The death rate caused by road traffic accidents per 100 thousand people. Standardised by age.* Death.rate.per.100.000.population - The death rate caused by road traffic accidents per 100 thousand people.

Population Projection Variables:

- SortOrder - NA
- LocID - The location ID
- Location - The location of the population or population projection
- VarID - The projection variant ID
- Variant - The name of the projection variant
- Time - The calendar year of the population or population projection
- MidPeriod - The middle of that particular year (e.g. Start of July 2016 is 2016.5)
- PopMale - The population of males in thousands
- PopFemale - The population of females in thousands
- PopTotal - The total human population in thousands
- PopDensity - Total population per square km in thousands

The two data sets will be merged later in the report after some data cleaning processes have been undertaken.

Understand

```
str(road_accidents)
```

```
## 'data.frame':    297 obs. of  12 variables:
##  $ Region.Code                                                 : chr  "OA" "OA" "O
A" "CSA" ...
##  $ Region.Name                                                 : chr  "Oceania" "Oc
eania" "Oceania" "Central and South America" ...
##  $ Country.Code                                                : chr  "AUS" "AUS"
"AUS" "BRA" ...
##  $ Country.Name                                                : chr  "Australia"
"Australia" "Australia" "Brazil" ...
##  $ Year                                                        : int  2016 2016 201
6 2016 2016 2016 2016 2016 2016 2016 ...
##  $ Sex                                                         : chr  "All" "Male"
"Female" "All" ...
##  $ Age.group.code                                              : chr  "Age_all" "Ag
e_all" "Age_all" "Age_all" ...
##  $ Age.Group                                                   : chr  "[All]" "[Al
l]" "[All]" "[All]" ...
##  $ Number                                                      : chr  "1259,0000000
0" "916,00000000" "343,00000000" "36165,00000000" ...
##  $ Percentage.of.cause.specific.deaths.out.of.total.deaths     : chr  "0,79430172"
"1,11888795" "0,44756449" "2,76116338" ...
##  $ Age.standardized.death.rate.per.100.000.standard.population: chr  "4,76733870"
"7,13116253" "2,43300674" "16,67451412" ...
##  $ Death.rate.per.100.000.population                           : chr  "5,20443487"
"7,63140068" "2,81427400" "17,54194046" ...
```

```
class(road_accidents)
```

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

As can be identified from the output of the str() function, 2 of the variables that will be used in this report (Country.Name and Number) are both character data types and the “Year” variable is an integer data type. However, the “Number” variable should be a numeric data type and the following steps are taken to convert it into a numeric data type. The gsub() function was used to remove the commas from the values and then the as.numeric() function was used to complete the converting of the variable into the correct data type. The incorrect “Number” variable will be removed along with a number of others later in the report.

```
fatalities_vector1 <- gsub(",", "", road_accidents$Number)
fatalities_vector2 <- as.numeric(fatalities_vector1)
road_accidents_subset <- cbind(road_accidents, fatalities_vector2)
```

```
str(population1)
```

```
## 'data.frame':    550866 obs. of  18 variables:
##  $ SortOrder  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ LocID      : int  1857 1857 1857 1857 1857 1857 1857 1857 1857 1857 ...
##  $ Notes      : chr  "" "" "" "" ...
##  $ ISO3_code  : chr  "" "" "" "" ...
##  $ ISO2_code  : chr  "" "" "" "" ...
##  $ SDMX_code  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ LocTypeID  : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ LocTypeName: chr  "" "" "" "" ...
##  $ ParentID   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Location   : chr  "AUKUS" "AUKUS" "AUKUS" "AUKUS" ...
##  $ VarID      : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Variant    : chr  "Medium" "Medium" "Medium" "Medium" ...
##  $ Time       : int  1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 ...
##  $ MidPeriod  : num  1950 1952 1952 1954 1954 ...
##  $ PopMale    : num  101684 103204 104727 106093 107669 ...
##  $ PopFemale  : num  104830 106086 107332 108906 110510 ...
##  $ PopTotal   : num  206514 209290 212059 214999 218179 ...
##  $ PopDensity : num  12.1 12.3 12.4 12.6 12.8 ...
```

```
class(population1)
```

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

Using the output from the str() function it can be identified that the “PopMale”, “PopFemale” are both of the numeric data type, the “Time” variable is the integer data type and the “Location” variable is the character data type. These are all satisfactory so no data type comversions are required.

The structure of both the data sets is data frame as can be seen using the class() function.

Tidy & Manipulate Data I

```
population1_subset <- population1 %>% filter(Time == 2016)
population2_subset <- population1_subset %>% dplyr::select(Location, PopMale, PopFema
le)
colnames(population2_subset) <- c('Location', 'Male', 'Female')
population_tidy <- population2_subset %>% gather(Male, Female, key = "Sex", value =
"Population")
head(population_tidy)
```

	Location <chr>	Sex <chr>	Population <dbl>
1	AUKUS	Male	206265.38
2	African Group	Male	614029.34
3	African Union	Male	614308.18
4	African Union: Central Africa	Male	71873.59
5	African Union: Eastern Africa	Male	176852.16
6	African Union: Northern Africa	Male	99972.47

6 rows

The reason for the data set being untidy is that the gender variable is separated into two columns (PopMale and PopFemale) rather than being in only one column like it should be. To fix this issue the data set must be converted into the long data frame format from the wide data format which the original data set is in. To do this, I've used the gather function to combine the "PopMale" and "PopFemale" variables into one variable titled "Population".

The reason for only the year 2016 being used is that it is the last year in which a large number of countries had complete data on road accidents.

```
road_accidents_subset1 <- road_accidents %>% dplyr::select(Country.Name, Sex, Number)
road_accidents_subset2 <- road_accidents_subset1 %>% filter(Sex != "All", Sex != "Unk
nown")
colnames(road_accidents_subset2) <- c('Location', 'Sex', 'Number')
fatalities_vector <- gsub(",", "", road_accidents_subset2$Number)
fatalities_vector3 <- as.numeric(fatalities_vector)
road_accidents_subset4 <- cbind(road_accidents_subset2, fatalities_vector3)
road_accidents_subset4 <- mutate(road_accidents_subset4, Deaths = fatalities_vector3
/ 100000000)
road_accidents_tidy <- road_accidents_subset4 %>% dplyr::select(Location, Sex, Death
s)
head(road_accidents_tidy)
```

	Location <chr>	Sex <chr>	Deaths <dbl>
1	Australia	Male	916
2	Australia	Female	343
3	Belgium	Female	152

	Location <chr>	Sex <chr>	Deaths <dbl>
4	Belgium	Male	456
5	Austria	Male	273
6	Antigua and Barbuda	Female	0

6 rows

A couple steps have to be taken to clean the road accidents data set. The inessential variables are dropped from the data set by using the select() function to select the required variables. The "Sex" variable should also only include observations with "Male" and Female" so the observations with "All" and "Unknown" are filtered out of the data set. The "Number" variable also has the obvious error of the true value being multiplied by 100,000,000. The mutate() function is used to fix this error.

```
combined_dataset <- population_tidy %>% left_join(road_accidents_tidy)
```

```
## Joining with `by = join_by(Location, Sex)`
```

```
head(combined_dataset)
```

	Location <chr>	Sex <chr>	Population <dbl>	Deaths <dbl>
1	AUKUS	Male	206265.38	*NA*
2	African Group	Male	614029.34	*NA*
3	African Union	Male	614308.18	*NA*
4	African Union: Central Africa	Male	71873.59	*NA*
5	African Union: Eastern Africa	Male	176852.16	*NA*
6	African Union: Northern Africa	Male	99972.47	*NA*

6 rows

The final step in this section is to merge the two data sets and I've used the left_join() function to do this. The common variables between the two data sets are the "Sex" and "Location" variables.

Tidy & Manipulate Data II

```
# This is the R chunk for the Tidy & Manipulate Data II
complete_dataset <- mutate(combined_dataset, Road_Accident_Deaths_per_10k_people = (D
eaths/Population)*10)
complete_dataset %>% filter(Location == "Australia")
```

Location <chr>	Sex <chr>	Population <dbl>	Deaths <dbl>	Road_Accident_Deaths_per_10k_people <dbl>
Australia	Male	12007.64	916	0.7628476
Australia	Female	12188.06	343	0.2814230

2 rows

The next task is to create a new variable. I've used the mutate() function to add a new variable called "Road_Accident_Deaths_per_10k_people" to the data set. This has been done by dividing the "Deaths" variable by the "Population" variable and then multiplying it by 10.

Since the missing values have not yet been removed from the data set, the head() function will not give any output to this new variable other than NA so instead I've used the filter function to briefly show the output of this new variable for one country only (Australia).

Scan I

```
# This is the R chunk for the Scan I
head(is.na(complete_dataset), n=15)
```

```
##       Location   Sex Population Deaths Road_Accident_Deaths_per_10k_people
##  [1,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [2,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [3,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [4,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [5,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [6,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [7,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [8,]    FALSE FALSE      FALSE   TRUE                                TRUE
##  [9,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [10,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [11,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [12,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [13,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [14,]    FALSE FALSE      FALSE   TRUE                                TRUE
## [15,]    FALSE FALSE      FALSE   TRUE                                TRUE
```

```
complete_dataset1 <- na.omit(complete_dataset)
head(complete_dataset1)
```

	Location <chr>	Sex <chr>	Population <dbl>	Deaths <dbl>	Road_Accident_Deaths_per_10k_people <dbl>
255	Mauritius	Male	640.423	151	2.3578166
256	Mayotte	Male	124.696	3	0.2405851
260	Seychelles	Male	52.632	15	2.8499772
279	Egypt	Male	50509.730	6275	1.2423349
290	South Africa	Male	27418.165	4796	1.7492053
311	Kazakhstan	Male	8654.278	1796	2.0752742

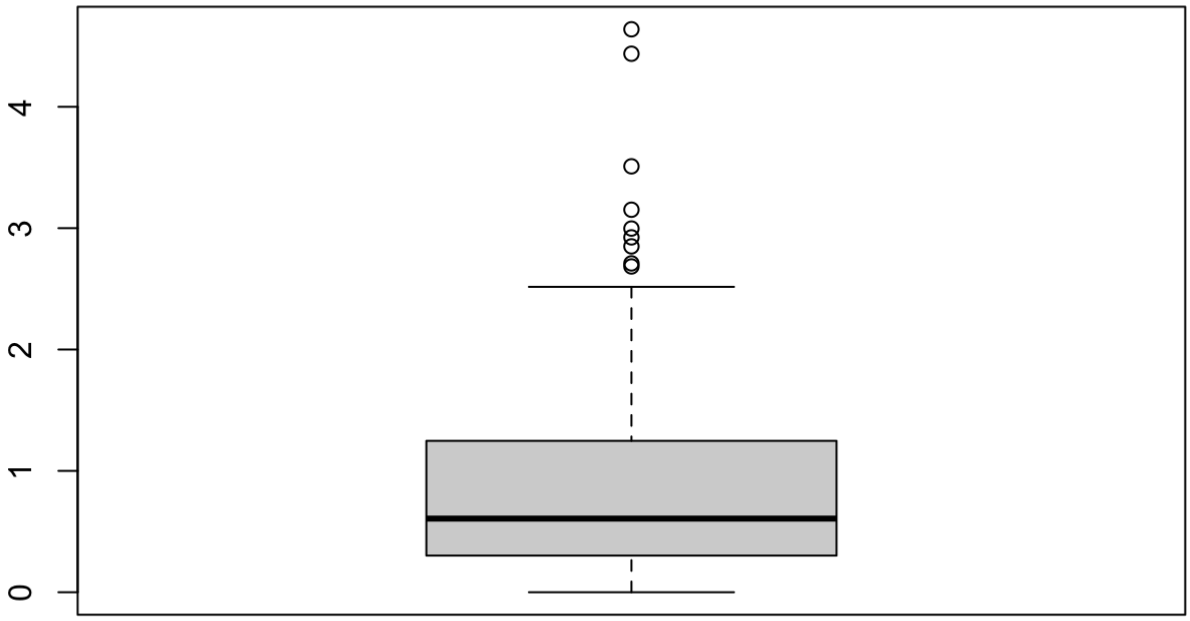
6 rows

The missing values (NA) can be removed from the data set as most of the missing values are in either countries with very small populations which would be insignificant to the data set or general regions (rather than specific countries) which wouldn't be used anyway as I'm looking at road accident deaths by country, not

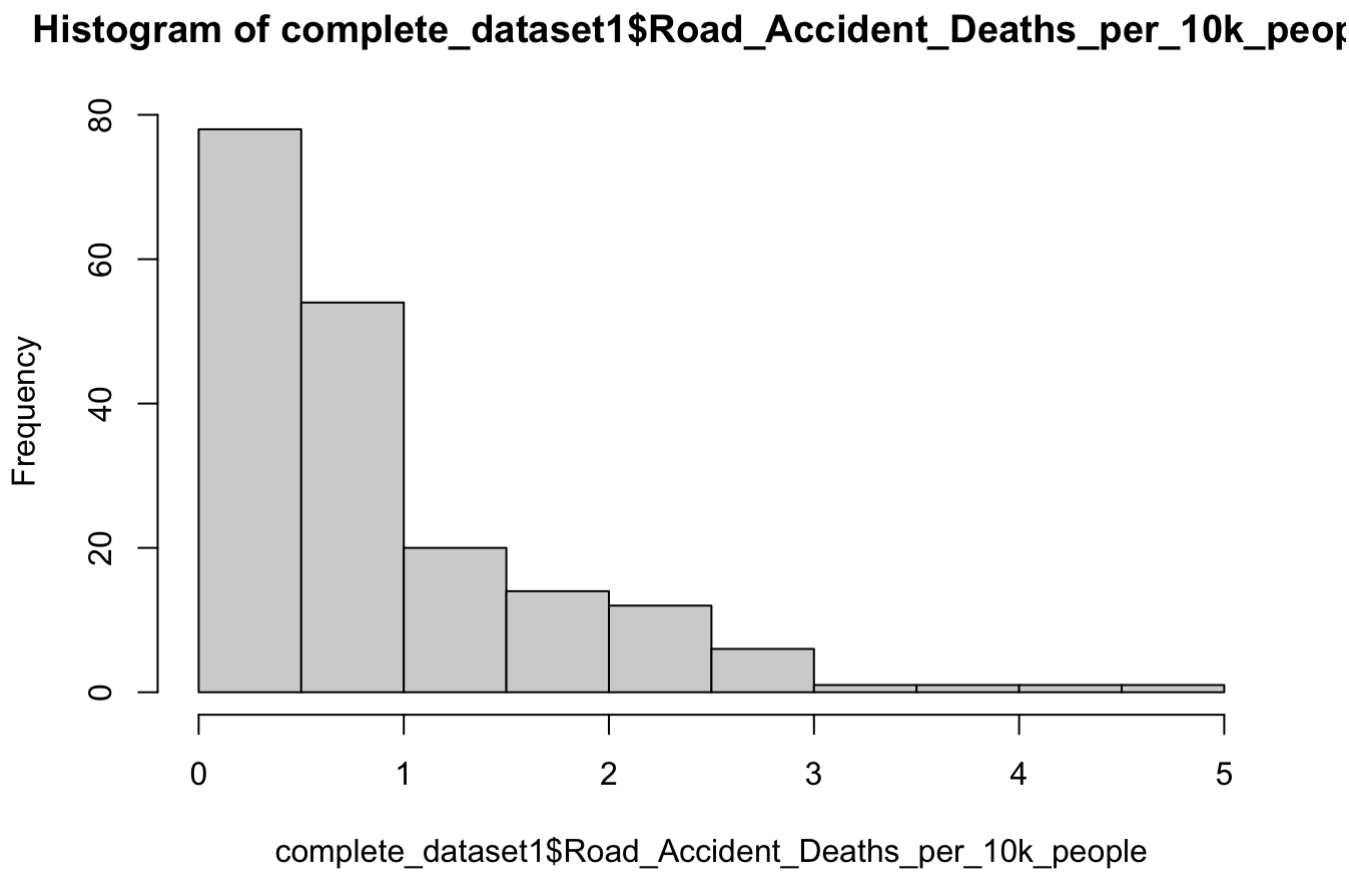
by region. I use the na.omit() function to remove the observations with NA in any of their columns.

Scan II

```
# This is the R chunk for the Scan II
complete_dataset1$Road_Accident_Deaths_per_10k_people %>% boxplot()
```




```
hist(complete_dataset1$Road_Accident_Deaths_per_10k_people)
```




```
accidents_zscore <- complete_dataset1$Road_Accident_Deaths_per_10k_people %>% scores
(type = "z")
summary(accidents_zscore)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.0475 -0.6866 -0.3242  0.0000  0.4368  4.4833
```

```
length(which(abs(accidents_zscore)>3))
```

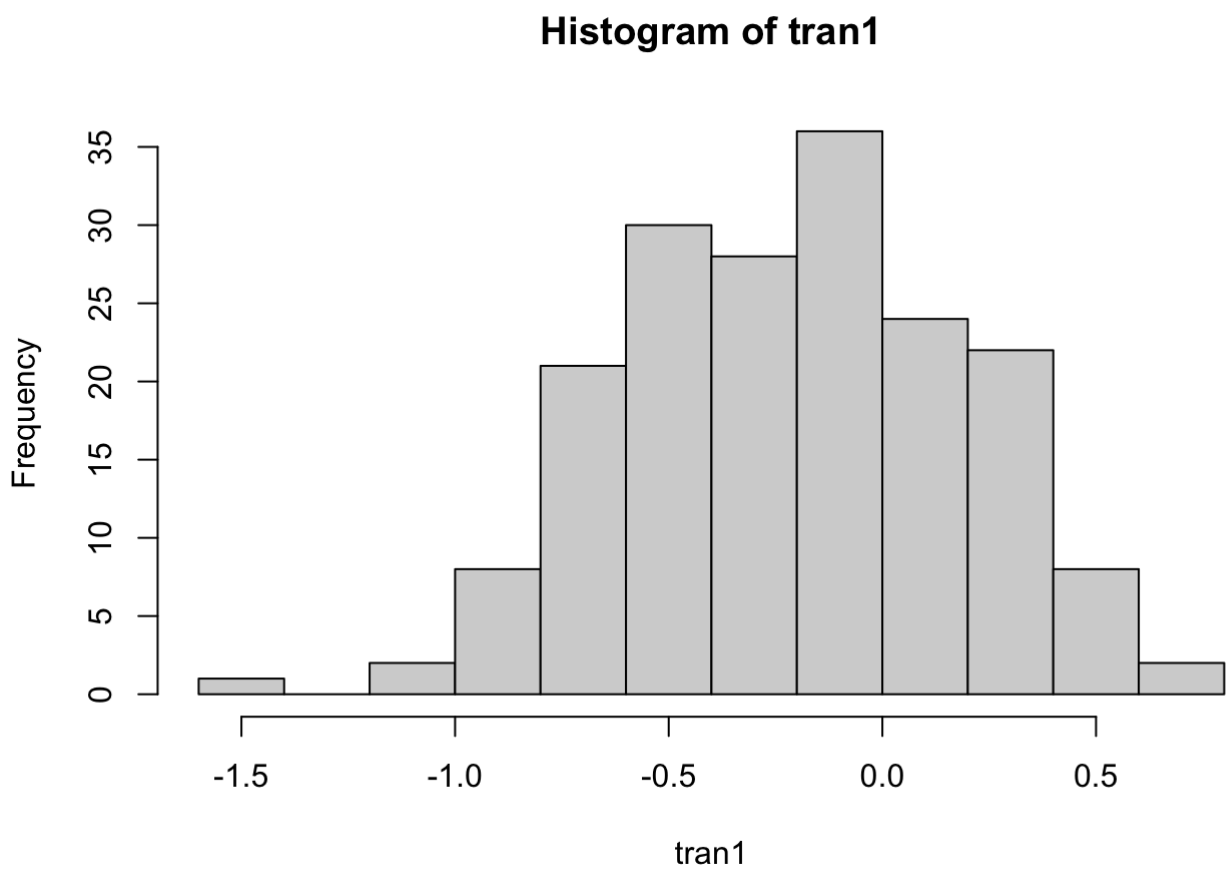
```
## [1] 3
```

Using the boxplot() function to create a box plot of the data it can be seen that the data contains a few values that would be considered outliers. The data is also heavily skewed to the right. The z-scores supports this information with 3 values considered to be outliers.

These issues will be fixed in the next section of the report using a transformation.

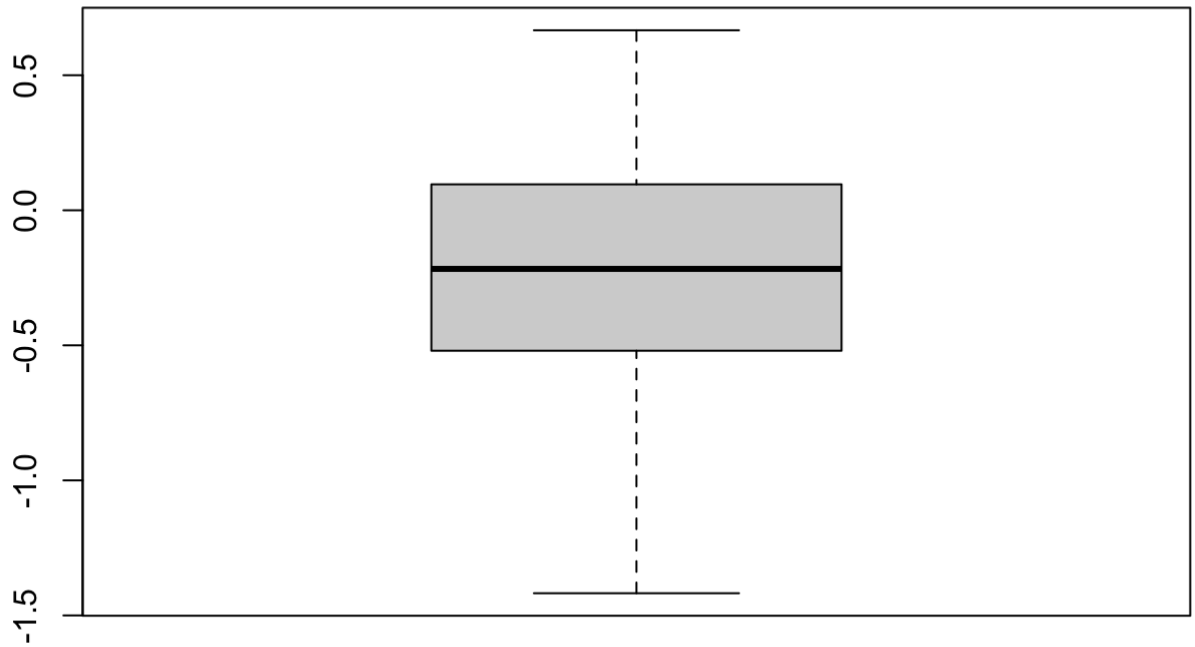
Transform

```
# This is the R chunk for the Transform Section
tran1 <- log10(complete_dataset1$Road_Accident_Deaths_per_10k_people)
hist(tran1)
```




```
boxplot(tran1)
```

```
## Warning in bplt(at[i], wid = width[i], stats = z$stats[, i], out =
## z$out[z$group == : Outlier (-Inf) in boxplot 1 is not drawn
```



As seen in the previous section of the report, the data is skewed to the right. To repair this issue, I have completed a logarithmic transformation using the log10() function to scale the data until it is closer to being normally distributed.

```
complete_zscore <- tran1 %>% scores(type = "z")
summary(complete_zscore)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      NA      NA      NA     NaN      NA      NA     188
```

```
length(which(abs(complete_zscore)>3))
```

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

Furthermore, the z-score is also re-calculated for the transformed data and the data set no longer contains any outliers.