Data Wrangling (Data Preprocessing)

Practical assessment 2

Karthik Kolume (Student ID : S3857825)

Submission Steps:

Required packages

The packages required to reproduce the report are,

```
# This is the R chunk for the required packages
library(outliers)
library(readr)
library(dplyr)
```

```
##
## 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(readxl)
library(gdata)
```

```
## gdata: read.xls support for 'XLS' (Excel 97-2004) files ENABLED.
```

```
##
```

```
## gdata: read.xls support for 'XLSX' (Excel 2007+) files ENABLED.
```

```
##
## Attaching package: 'gdata'
```

```
## The following objects are masked from 'package:dplyr':
##
##     combine, first, last
```

```
## The following object is masked from 'package:stats':
##
##     nobs
```

```
## The following object is masked from 'package:utils':
##
##     object.size
```

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

```
library(tidyr)
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(rvest)
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
library(knitr)
```

Executive Summary

This report describes the preprocessing steps of “World happiness report” for each year from year 2005 to year 2021 for 162 countries and represent their Happiness Score and the Rank of each country based on the their Ladder score.

Steps taken to preprocess the data are:

- Data extracted/downloaded from https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021 (https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021) and loaded in the R-studio.
- Understanding the Data used by using summary and Structure functions and converting the relevant data types to factors for better processing of data.
- Tidy and Manipulate the Data set in order to find the Ranks of the Countries with highest Happiness/Ladder score.
- Scanning the data to find the missing values present in the data set. Replacing the missing values with the average score over all the years of consideration.
- Transforming the data to get the normal distribution and understanding/Analyzing the data insights and interpreting the right results.

Data

- Two sets of Data have been extracted/Downloaded from site https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021 (https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021) and loaded in the R-studio as shown in the below R-chunk.
- Used the World happiness datasets. Dataset 1 consists of the World happiness report from year 2005 to year 2020 and Dataset 2 consists of the World happiness report of year 2021 alone.
- These datasets consists of many variables along with the Countries and Ladder score. We are considering only 2 variables from the original datasets i.e; Countries and Ladder score.

```
setwd("~/Documents/RMIT/RMIT Work/DW_worksheets")
World_happiness_2021<- read_csv("world-happiness-report-2021.csv")
```

```
##
## ── Column specification ────────────────────────────────────────────────────
─
## cols(
##   .default = col_double(),
##   `Country name` = col_character(),
##   `Regional indicator` = col_character()
## )
## ℹ Use `spec()` for the full column specifications.
```

```
World_happiness<- read_csv("world-happiness-report.csv")
```

```
##
## ── Column specification ────────────────────────────────────────────────────
─
## cols(
##   `Country name` = col_character(),
##   year = col_double(),
##   `Life Ladder` = col_double(),
##   `Log GDP per capita` = col_double(),
##   `Social support` = col_double(),
##   `Healthy life expectancy at birth` = col_double(),
##   `Freedom to make life choices` = col_double(),
##   Generosity = col_double(),
##   `Perceptions of corruption` = col_double(),
##   `Positive affect` = col_double(),
##   `Negative affect` = col_double()
## )
```

```
head(World_happiness)
```

Country name	y...	Life Ladder	Log GDP per capita	Social support	Healthy
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	
Afghanistan	2008	3.724	7.370	0.451	
Afghanistan	2009	4.402	7.540	0.552	
Afghanistan	2010	4.758	7.647	0.539	
Afghanistan	2011	3.832	7.620	0.521	
Afghanistan	2012	3.783	7.705	0.521	
Afghanistan	2013	3.572	7.725	0.484	

6 rows | 1-6 of 11 columns

```
head(World_happiness_2021)
```

Country name <chr>	Regional indicator <chr>	Ladder score <dbl>	Standard error of ladder score <dbl>
Finland	Western Europe	7.842	0.032
Denmark	Western Europe	7.620	0.035
Switzerland	Western Europe	7.571	0.036
Iceland	Western Europe	7.554	0.059
Netherlands	Western Europe	7.464	0.027
Norway	Western Europe	7.392	0.035

6 rows | 1-5 of 20 columns

Understand

To understand the Data,

- Firstly, Structure of the data is found using str() function.
- The data sets used consists of multiple data types(Numeric and Character)
- The variable “year” has been changed from numeric to Ordered factor in the below R-chunk because the variable “year” is a categorical variable, which needs to be ordered and labelled.
- Summary and Structure functions are used below to understand the data efficiently.

```
# This is the R chunk for the Understand Section
str(World_happiness)
```

```
## spec_tbl_df [1,949 × 11] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Country name                    : chr [1:1949] "Afghanistan" "Afghanistan" "A
fghanistan" "Afghanistan" ...
##  $ year                            : num [1:1949] 2008 2009 2010 2011 2012 ...
##  $ Life Ladder                     : num [1:1949] 3.72 4.4 4.76 3.83 3.78 ...
##  $ Log GDP per capita              : num [1:1949] 7.37 7.54 7.65 7.62 7.71 ...
##  $ Social support                  : num [1:1949] 0.451 0.552 0.539 0.521 0.521
0.484 0.526 0.529 0.559 0.491 ...
##  $ Healthy life expectancy at birth: num [1:1949] 50.8 51.2 51.6 51.9 52.2 ...
##  $ Freedom to make life choices    : num [1:1949] 0.718 0.679 0.6 0.496 0.531 0.
578 0.509 0.389 0.523 0.427 ...
##  $ Generosity                      : num [1:1949] 0.168 0.19 0.121 0.162 0.236 0
.061 0.104 0.08 0.042 -0.121 ...
##  $ Perceptions of corruption       : num [1:1949] 0.882 0.85 0.707 0.731 0.776 0
.823 0.871 0.881 0.793 0.954 ...
##  $ Positive affect                 : num [1:1949] 0.518 0.584 0.618 0.611 0.71 0
.621 0.532 0.554 0.565 0.496 ...
##  $ Negative affect                 : num [1:1949] 0.258 0.237 0.275 0.267 0.268
0.273 0.375 0.339 0.348 0.371 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Country name` = col_character(),
##   ..   year = col_double(),
##   ..   `Life Ladder` = col_double(),
##   ..   `Log GDP per capita` = col_double(),
##   ..   `Social support` = col_double(),
##   ..   `Healthy life expectancy at birth` = col_double(),
##   ..   `Freedom to make life choices` = col_double(),
##   ..   Generosity = col_double(),
##   ..   `Perceptions of corruption` = col_double(),
##   ..   `Positive affect` = col_double(),
##   ..   `Negative affect` = col_double()
##   .. )
```

```
World_happiness$year<- factor(World_happiness$year,levels= c('2005', '2006', '2007'
, '2008', '2009','2010','2011','2012','2013','2014','2015','2016','2017','2018','20
19','2020'), ordered=TRUE,
                    labels = c('Year 1', 'Year 2', 'Year 3', 'Year 4', 'Year 5','Ye
ar 6','Year 7','Year 8','Year 9','Year 10','Year 11','Year 12','Year 13','Year 14',
'Year 15','Year 16'))

str(World_happiness)
```

```
## spec_tbl_df [1,949 × 11] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Country name                    : chr [1:1949] "Afghanistan" "Afghanistan" "A
fghanistan" "Afghanistan" ...
##  $ year                            : Ord.factor w/ 16 levels "Year 1"<"Year 2"<.
.: 4 5 6 7 8 9 10 11 12 13 ...
##  $ Life Ladder                     : num [1:1949] 3.72 4.4 4.76 3.83 3.78 ...
##  $ Log GDP per capita              : num [1:1949] 7.37 7.54 7.65 7.62 7.71 ...
##  $ Social support                  : num [1:1949] 0.451 0.552 0.539 0.521 0.521
0.484 0.526 0.529 0.559 0.491 ...
##  $ Healthy life expectancy at birth: num [1:1949] 50.8 51.2 51.6 51.9 52.2 ...
##  $ Freedom to make life choices    : num [1:1949] 0.718 0.679 0.6 0.496 0.531 0.
578 0.509 0.389 0.523 0.427 ...
##  $ Generosity                      : num [1:1949] 0.168 0.19 0.121 0.162 0.236 0
.061 0.104 0.08 0.042 -0.121 ...
##  $ Perceptions of corruption       : num [1:1949] 0.882 0.85 0.707 0.731 0.776 0
.823 0.871 0.881 0.793 0.954 ...
##  $ Positive affect                 : num [1:1949] 0.518 0.584 0.618 0.611 0.71 0
.621 0.532 0.554 0.565 0.496 ...
##  $ Negative affect                 : num [1:1949] 0.258 0.237 0.275 0.267 0.268
0.273 0.375 0.339 0.348 0.371 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Country name` = col_character(),
##   ..   year = col_double(),
##   ..   `Life Ladder` = col_double(),
##   ..   `Log GDP per capita` = col_double(),
##   ..   `Social support` = col_double(),
##   ..   `Healthy life expectancy at birth` = col_double(),
##   ..   `Freedom to make life choices` = col_double(),
##   ..   Generosity = col_double(),
##   ..   `Perceptions of corruption` = col_double(),
##   ..   `Positive affect` = col_double(),
##   ..   `Negative affect` = col_double()
##   .. )
```

```
summary(World_happiness)
```

```
##  Country name            year       Life Ladder    Log GDP per capita
##  Length:1949        Year 13: 147   Min.   :2.375   Min.   : 6.635
##  Class :character   Year 7 : 146   1st Qu.:4.640   1st Qu.: 8.464
##  Mode  :character   Year 10: 145   Median :5.386   Median : 9.460
##                     Year 15: 144   Mean   :5.467   Mean   : 9.368
##                     Year 11: 143   3rd Qu.:6.283   3rd Qu.:10.353
##                     Year 8 : 142   Max.   :8.019   Max.   :11.648
##                     (Other):1082                   NA's   :36
##  Social support   Healthy life expectancy at birth Freedom to make life choices
##  Min.   :0.2900   Min.   :32.30                    Min.   :0.2580
##  1st Qu.:0.7498   1st Qu.:58.69                    1st Qu.:0.6470
##  Median :0.8355   Median :65.20                    Median :0.7630
##  Mean   :0.8126   Mean   :63.36                    Mean   :0.7426
##  3rd Qu.:0.9050   3rd Qu.:68.59                    3rd Qu.:0.8560
##  Max.   :0.9870   Max.   :77.10                    Max.   :0.9850
##  NA's   :13       NA's   :55                       NA's   :32
##    Generosity      Perceptions of corruption Positive affect  Negative affect
##  Min.   :-0.3350   Min.   :0.0350            Min.   :0.3220   Min.   :0.0830
##  1st Qu.:-0.1130   1st Qu.:0.6900            1st Qu.:0.6255   1st Qu.:0.2060
##  Median :-0.0255   Median :0.8020            Median :0.7220   Median :0.2580
##  Mean   : 0.0001   Mean   :0.7471            Mean   :0.7100   Mean   :0.2685
##  3rd Qu.: 0.0910   3rd Qu.:0.8720            3rd Qu.:0.7990   3rd Qu.:0.3200
##  Max.   : 0.6980   Max.   :0.9830            Max.   :0.9440   Max.   :0.7050
##  NA's   :89        NA's   :110               NA's   :22       NA's   :16
```

```
str(World_happiness_2021)
```

```
## spec_tbl_df [149 × 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Country name                            : chr [1:149] "Finland" "Denmark" "
Switzerland" "Iceland" ...
##  $ Regional indicator                      : chr [1:149] "Western Europe" "Wes
tern Europe" "Western Europe" "Western Europe" ...
##  $ Ladder score                            : num [1:149] 7.84 7.62 7.57 7.55 7
.46 ...
##  $ Standard error of ladder score          : num [1:149] 0.032 0.035 0.036 0.0
59 0.027 0.035 0.036 0.037 0.04 0.036 ...
##  $ upperwhisker                            : num [1:149] 7.9 7.69 7.64 7.67 7.
52 ...
##  $ lowerwhisker                            : num [1:149] 7.78 7.55 7.5 7.44 7.
41 ...
##  $ Logged GDP per capita                   : num [1:149] 10.8 10.9 11.1 10.9 1
0.9 ...
##  $ Social support                          : num [1:149] 0.954 0.954 0.942 0.9
83 0.942 0.954 0.934 0.908 0.948 0.934 ...
##  $ Healthy life expectancy                 : num [1:149] 72 72.7 74.4 73 72.4
73.3 72.7 72.6 73.4 73.3 ...
##  $ Freedom to make life choices            : num [1:149] 0.949 0.946 0.919 0.9
55 0.913 0.96 0.945 0.907 0.929 0.908 ...
```

```
##  $ Generosity                                : num [1:149] -0.098 0.03 0.025 0.1
6 0.175 0.093 0.086 -0.034 0.134 0.042 ...
##  $ Perceptions of corruption                 : num [1:149] 0.186 0.179 0.292 0.6
73 0.338 0.27 0.237 0.386 0.242 0.481 ...
##  $ Ladder score in Dystopia                  : num [1:149] 2.43 2.43 2.43 2.43 2
.43 2.43 2.43 2.43 2.43 2.43 ...
##  $ Explained by: Log GDP per capita          : num [1:149] 1.45 1.5 1.57 1.48 1.
5 ...
##  $ Explained by: Social support              : num [1:149] 1.11 1.11 1.08 1.17 1
.08 ...
##  $ Explained by: Healthy life expectancy     : num [1:149] 0.741 0.763 0.816 0.7
72 0.753 0.782 0.763 0.76 0.785 0.782 ...
##  $ Explained by: Freedom to make life choices: num [1:149] 0.691 0.686 0.653 0.6
98 0.647 0.703 0.685 0.639 0.665 0.64 ...
##  $ Explained by: Generosity                  : num [1:149] 0.124 0.208 0.204 0.2
93 0.302 0.249 0.244 0.166 0.276 0.215 ...
##  $ Explained by: Perceptions of corruption   : num [1:149] 0.481 0.485 0.413 0.1
7 0.384 0.427 0.448 0.353 0.445 0.292 ...
##  $ Dystopia + residual                       : num [1:149] 3.25 2.87 2.84 2.97 2
.8 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Country name` = col_character(),
##   ..   `Regional indicator` = col_character(),
##   ..   `Ladder score` = col_double(),
##   ..   `Standard error of ladder score` = col_double(),
##   ..   upperwhisker = col_double(),
##   ..   lowerwhisker = col_double(),
##   ..   `Logged GDP per capita` = col_double(),
##   ..   `Social support` = col_double(),
##   ..   `Healthy life expectancy` = col_double(),
##   ..   `Freedom to make life choices` = col_double(),
##   ..   Generosity = col_double(),
##   ..   `Perceptions of corruption` = col_double(),
##   ..   `Ladder score in Dystopia` = col_double(),
##   ..   `Explained by: Log GDP per capita` = col_double(),
##   ..   `Explained by: Social support` = col_double(),
##   ..   `Explained by: Healthy life expectancy` = col_double(),
##   ..   `Explained by: Freedom to make life choices` = col_double(),
##   ..   `Explained by: Generosity` = col_double(),
##   ..   `Explained by: Perceptions of corruption` = col_double(),
##   ..   `Dystopia + residual` = col_double()
##   .. )
```

```
summary(World_happiness_2021)
```

```
##  Country name       Regional indicator  Ladder score
##  Length:149         Length:149         Min.   :2.523
##  Class :character   Class :character   1st Qu.:4.852
```

```
##  Mode  :character   Mode  :character   Median :5.534
##                                        Mean   :5.533
##                                        3rd Qu.:6.255
##                                        Max.   :7.842
##  Standard error of ladder score  upperwhisker    lowerwhisker
##  Min.   :0.02600                Min.   :2.596   Min.   :2.449
##  1st Qu.:0.04300                1st Qu.:4.991   1st Qu.:4.706
##  Median :0.05400                Median :5.625   Median :5.413
##  Mean   :0.05875                Mean   :5.648   Mean   :5.418
##  3rd Qu.:0.07000                3rd Qu.:6.344   3rd Qu.:6.128
##  Max.   :0.17300                Max.   :7.904   Max.   :7.780
##  Logged GDP per capita Social support   Healthy life expectancy
##  Min.   : 6.635        Min.   :0.4630   Min.   :48.48
##  1st Qu.: 8.541        1st Qu.:0.7500   1st Qu.:59.80
##  Median : 9.569        Median :0.8320   Median :66.60
##  Mean   : 9.432        Mean   :0.8147   Mean   :64.99
##  3rd Qu.:10.421        3rd Qu.:0.9050   3rd Qu.:69.60
##  Max.   :11.647        Max.   :0.9830   Max.   :76.95
##  Freedom to make life choices   Generosity       Perceptions of corruption
##  Min.   :0.3820               Min.   :-0.28800   Min.   :0.0820
##  1st Qu.:0.7180               1st Qu.:-0.12600   1st Qu.:0.6670
##  Median :0.8040               Median :-0.03600   Median :0.7810
##  Mean   :0.7916               Mean   :-0.01513   Mean   :0.7274
##  3rd Qu.:0.8770               3rd Qu.: 0.07900   3rd Qu.:0.8450
##  Max.   :0.9700               Max.   : 0.54200   Max.   :0.9390
##  Ladder score in Dystopia Explained by: Log GDP per capita
##  Min.   :2.43             Min.   :0.0000
##  1st Qu.:2.43             1st Qu.:0.6660
##  Median :2.43             Median :1.0250
##  Mean   :2.43             Mean   :0.9772
##  3rd Qu.:2.43             3rd Qu.:1.3230
##  Max.   :2.43             Max.   :1.7510
##  Explained by: Social support Explained by: Healthy life expectancy
##  Min.   :0.0000               Min.   :0.0000
##  1st Qu.:0.6470               1st Qu.:0.3570
##  Median :0.8320               Median :0.5710
##  Mean   :0.7933               Mean   :0.5202
##  3rd Qu.:0.9960               3rd Qu.:0.6650
##  Max.   :1.1720               Max.   :0.8970
##  Explained by: Freedom to make life choices Explained by: Generosity
##  Min.   :0.0000                             Min.   :0.000
##  1st Qu.:0.4090                             1st Qu.:0.105
##  Median :0.5140                             Median :0.164
##  Mean   :0.4987                             Mean   :0.178
##  3rd Qu.:0.6030                             3rd Qu.:0.239
##  Max.   :0.7160                             Max.   :0.541
##  Explained by: Perceptions of corruption Dystopia + residual
##  Min.   :0.0000                          Min.   :0.648
##  1st Qu.:0.0600                          1st Qu.:2.138
##  Median :0.1010                          Median :2.509
##  Mean   :0.1351                          Mean   :2.430
```

```
##  3rd Qu.:0.1740                          3rd Qu.:2.794
##  Max.   :0.5470                          Max.   :3.482
```

Tidy & Manipulate Data I

In this section, we Tidy the data and manipulate the data to meet our requirements from this data.

Steps:

- Remove the unwanted columns which are not required for the preprocessing.
- The dataset 1 cannot to used to determine the score for each country alone.In order to tidy this data we need to use the package called “tidyr”.Since, multiple variables are stored in rows, the pivot_wider() function to generate columns from rows.
- Variable “Country name” is changed to “Country”.
- Similarly we rename another dataset as well.

```
# This is the R chunk for the Tidy & Manipulate Data I
World_happiness <- World_happiness [,-c(4,5,6,7,8,9,10,11)]
head(World_happiness)
```

Country name <chr>	year <ord>	Life Ladder <dbl>
Afghanistan	Year 4	3.724
Afghanistan	Year 5	4.402
Afghanistan	Year 6	4.758
Afghanistan	Year 7	3.832
Afghanistan	Year 8	3.783
Afghanistan	Year 9	3.572

6 rows

```
WH_Tidy<-pivot_wider(World_happiness, names_from = "year", values_from = "Life Ladd
er")
WH_Tidy<- WH_Tidy%>% rename(c("Country" = "Country name"))
head(WH_Tidy)
```

Country <chr>	Year 4 <dbl>	Year 5 <dbl>	Year 6 <dbl>	Year 7 <dbl>	Year 8 <dbl>	Year 9 <dbl>	Year 10 <dbl>	Year 11 <dbl>	Year 12 <dbl>
Afghanistan	3.724	4.402	4.758	3.832	3.783	3.572	3.131	3.983	4.220
Albania	*NA*	5.485	5.269	5.867	5.510	4.551	4.814	4.607	4.511

Algeria	*NA*	*NA*	5.464	5.317	5.605	*NA*	6.355	*NA*	5.341
Angola	*NA*	*NA*	*NA*	5.589	4.360	3.937	3.795	*NA*	*NA*
Argentina	5.961	6.424	6.441	6.776	6.468	6.582	6.671	6.697	6.427
Armenia	4.652	4.178	4.368	4.260	4.320	4.277	4.453	4.348	4.325

6 rows | 1-10 of 17 columns

```
World_happiness_2021 <- World_happiness_2021 [,-c(2,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)]
World_happiness_2021<- World_happiness_2021%>% rename(c("Country" = "Country name",
                      "Year 17" = "Ladder score"))
head(World_happiness_2021)
```

Country <chr>	**Year 17** <dbl>
Finland	7.842
Denmark	7.620
Switzerland	7.571
Iceland	7.554
Netherlands	7.464
Norway	7.392

6 rows

Tidy & Manipulate Data II

Join both the datasets,find the average score and rank them accordingly.

Steps:

- We join both the datasets using left_join function. Joining is done with respect to variable “Country”.
- New column “Average” is introduced to find the Average/mean of the ladder Score for each country across all the years considered.
- Changed the datatype of “Average” to numeric.
- Assigned the order of the rows with respect to “Average” in the decreasing format in order to find the countries which have highest and lowest scores.
- To assign the rankings of the countries,new column “Rank” is introduced and assigned accordingly starting from 1 being the highest Score.

```
# This is the R chunk for the Tidy & Manipulate Data II
merge<- WH_Tidy %>% left_join(World_happiness_2021, by = "Country")
head(merge)
```

Country <chr>	Year 4 <dbl>	Year 5 <dbl>	Year 6 <dbl>	Year 7 <dbl>	Year 8 <dbl>	Year 9 <dbl>	Year 10 <dbl>	Year 11 <dbl>	Year 12 <dbl>
Afghanistan	3.724	4.402	4.758	3.832	3.783	3.572	3.131	3.983	4.220
Albania	*NA*	5.485	5.269	5.867	5.510	4.551	4.814	4.607	4.511
Algeria	*NA*	*NA*	5.464	5.317	5.605	*NA*	6.355	*NA*	5.341
Angola	*NA*	*NA*	*NA*	5.589	4.360	3.937	3.795	*NA*	*NA*
Argentina	5.961	6.424	6.441	6.776	6.468	6.582	6.671	6.697	6.427
Armenia	4.652	4.178	4.368	4.260	4.320	4.277	4.453	4.348	4.325

6 rows | 1-10 of 18 columns

```
merge<- merge %>% mutate(Average= rowMeans(merge[,2:18], na.rm=TRUE))
head(merge)
```

Country <chr>	Year 4 <dbl>	Year 5 <dbl>	Year 6 <dbl>	Year 7 <dbl>	Year 8 <dbl>	Year 9 <dbl>	Year 10 <dbl>	Year 11 <dbl>	Year 12 <dbl>
Afghanistan	3.724	4.402	4.758	3.832	3.783	3.572	3.131	3.983	4.220
Albania	*NA*	5.485	5.269	5.867	5.510	4.551	4.814	4.607	4.511
Algeria	*NA*	*NA*	5.464	5.317	5.605	*NA*	6.355	*NA*	5.341
Angola	*NA*	*NA*	*NA*	5.589	4.360	3.937	3.795	*NA*	*NA*
Argentina	5.961	6.424	6.441	6.776	6.468	6.582	6.671	6.697	6.427
Armenia	4.652	4.178	4.368	4.260	4.320	4.277	4.453	4.348	4.325

6 rows | 1-10 of 19 columns

```
merge<-merge[,c(1,19,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18)]
merge$Average<-as.numeric(merge$Average)
merge<-merge[order(merge$Average,decreasing=TRUE),]
head(merge)
```

Country <chr>	Average <dbl>	Year 4 <dbl>	Year 5 <dbl>	Year 6 <dbl>	Year 7 <dbl>	Year 8 <dbl>	Year 9 <dbl>	Year 10 <dbl>	Year 11 <dbl>
Denmark	7.676625	7.971	7.683	7.771	7.788	7.520	7.589	7.508	7.514

Finland	7.614643	7.671	*NA*	7.393	7.354	7.420	7.445	7.385	7.448
Switzerland	7.550364	*NA*	7.525	*NA*	*NA*	7.776	*NA*	7.493	7.572
Norway	7.501455	7.632	*NA*	*NA*	*NA*	7.678	*NA*	7.444	7.603
Netherlands	7.466133	7.631	*NA*	7.502	7.564	7.471	7.407	7.321	7.324
Iceland	7.458444	6.888	*NA*	*NA*	*NA*	7.591	7.501	*NA*	7.498

6 rows | 1-10 of 19 columns

```
merge<- merge %>% mutate(Rank = row_number())
head(merge)
```

Country <chr>	**Average** <dbl>	**Year 4** <dbl>	**Year 5** <dbl>	**Year 6** <dbl>	**Year 7** <dbl>	**Year 8** <dbl>	**Year 9** <dbl>	**Year 10** <dbl>	**Year 11** <dbl>
Denmark	7.676625	7.971	7.683	7.771	7.788	7.520	7.589	7.508	7.514
Finland	7.614643	7.671	*NA*	7.393	7.354	7.420	7.445	7.385	7.448
Switzerland	7.550364	*NA*	7.525	*NA*	*NA*	7.776	*NA*	7.493	7.572
Norway	7.501455	7.632	*NA*	*NA*	*NA*	7.678	*NA*	7.444	7.603
Netherlands	7.466133	7.631	*NA*	7.502	7.564	7.471	7.407	7.321	7.324
Iceland	7.458444	6.888	*NA*	*NA*	*NA*	7.591	7.501	*NA*	7.498

6 rows | 1-10 of 20 columns

```
merge<-merge[,c(1,2,20,18,17,15,3,4,5,6,7,8,9,10,11,12,13,14,16,19)]
head(merge)
```

Country <chr>	**Average** <dbl>	**R...** <int>	**Year 1** <dbl>	**Year 2** <dbl>	**Year 3** <dbl>	**Year 4** <dbl>	**Year 5** <dbl>	**Year 6** <dbl>	**Year 7** <dbl>
Denmark	7.676625	1	8.019	*NA*	7.834	7.971	7.683	7.771	7.788
Finland	7.614643	2	*NA*	7.672	*NA*	7.671	*NA*	7.393	7.354
Switzerland	7.550364	3	*NA*	7.473	*NA*	*NA*	7.525	*NA*	*NA*
Norway	7.501455	4	*NA*	7.416	*NA*	7.632	*NA*	*NA*	*NA*
Netherlands	7.466133	5	7.464	*NA*	7.452	7.631	*NA*	7.502	7.564
Iceland	7.458444	6	*NA*	*NA*	*NA*	6.888	*NA*	*NA*	*NA*

6 rows | 1-10 of 20 columns

Scan I

Scan the Dataset obtained and identify if there are any missing values, special values and obvious errors (i.e. inconsistencies)in the dataset. Steps:

- Identification of missing values present in the data set using is.na() function.
- Finding the number of missing values using sum(is.na()) function.
- Replacing the missing values with the average value of the row.
- Identification of special values and obvious errors (i.e. inconsistencies) using is.special function as shown in the below R-chunk which is zero in our data.

```
# This is the R chunk for the Scan I
head(is.na(merge))
```

```
##      Country Average  Rank Year 1 Year 2 Year 3 Year 4 Year 5 Year 6 Year 7
## [1,]   FALSE   FALSE FALSE  FALSE   TRUE  FALSE  FALSE  FALSE  FALSE  FALSE
## [2,]   FALSE   FALSE FALSE   TRUE  FALSE   TRUE  FALSE   TRUE  FALSE  FALSE
## [3,]   FALSE   FALSE FALSE   TRUE  FALSE   TRUE   TRUE  FALSE   TRUE   TRUE
## [4,]   FALSE   FALSE FALSE   TRUE  FALSE   TRUE  FALSE   TRUE   TRUE   TRUE
## [5,]   FALSE   FALSE FALSE  FALSE   TRUE  FALSE  FALSE   TRUE  FALSE  FALSE
## [6,]   FALSE   FALSE FALSE   TRUE   TRUE   TRUE  FALSE   TRUE   TRUE   TRUE
##      Year 8 Year 9 Year 10 Year 11 Year 12 Year 13 Year 14 Year 15 Year 16
## [1,]  FALSE  FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
## [2,]  FALSE  FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
## [3,]  FALSE   TRUE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
## [4,]  FALSE   TRUE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
## [5,]  FALSE  FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
## [6,]  FALSE  FALSE    TRUE   FALSE   FALSE   FALSE    TRUE   FALSE   FALSE
##      Year 17
## [1,]   FALSE
## [2,]   FALSE
## [3,]   FALSE
## [4,]   FALSE
## [5,]   FALSE
## [6,]   FALSE
```

```
sum(is.na(merge))
```

```
## [1] 724
```

```
merge$'Year 4'[is.na(merge$'Year 4')]<- merge$Average[is.na(merge$'Year 4')]
merge$'Year 5'[is.na(merge$'Year 5')]<- merge$Average[is.na(merge$'Year 5')]
merge$'Year 6'[is.na(merge$'Year 6')]<- merge$Average[is.na(merge$'Year 6')]
merge$'Year 7'[is.na(merge$'Year 7')]<- merge$Average[is.na(merge$'Year 7')]
merge$'Year 8'[is.na(merge$'Year 8')]<- merge$Average[is.na(merge$'Year 8')]
merge$'Year 9'[is.na(merge$'Year 9')]<- merge$Average[is.na(merge$'Year 9')]
merge$'Year 10'[is.na(merge$'Year 10')]<- merge$Average[is.na(merge$'Year 10')]
merge$'Year 11'[is.na(merge$'Year 11')]<- merge$Average[is.na(merge$'Year 11')]
merge$'Year 12'[is.na(merge$'Year 12')]<- merge$Average[is.na(merge$'Year 12')]
merge$'Year 13'[is.na(merge$'Year 13')]<- merge$Average[is.na(merge$'Year 13')]
merge$'Year 14'[is.na(merge$'Year 14')]<- merge$Average[is.na(merge$'Year 14')]
merge$'Year 15'[is.na(merge$'Year 15')]<- merge$Average[is.na(merge$'Year 15')]
merge$'Year 16'[is.na(merge$'Year 16')]<- merge$Average[is.na(merge$'Year 16')]
merge$'Year 17'[is.na(merge$'Year 17')]<- merge$Average[is.na(merge$'Year 17')]
merge$'Year 1'[is.na(merge$'Year 1')]<- merge$Average[is.na(merge$'Year 1')]
merge$'Year 2'[is.na(merge$'Year 2')]<- merge$Average[is.na(merge$'Year 2')]
merge$'Year 3'[is.na(merge$'Year 3')]<- merge$Average[is.na(merge$'Year 3')]

is.special <- function(x){
if (is.numeric(x)) (is.infinite(x) | is.nan(x))
}
sum(is.special(sapply(merge, is.special)))
```

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

```
head(merge)
```

Country <chr>	Average <dbl>	R... <int>	Year 1 <dbl>	Year 2 <dbl>	Year 3 <dbl>	Year 4 <dbl>	Year 5 <dbl>	Year 6 <dbl>	
Denmark	7.676625	1	8.019000	7.676625	7.834000	7.971000	7.683000	7.771000	7.
Finland	7.614643	2	7.614643	7.672000	7.614643	7.671000	7.614643	7.393000	7.
Switzerland	7.550364	3	7.550364	7.473000	7.550364	7.550364	7.525000	7.550364	7.
Norway	7.501455	4	7.501455	7.416000	7.501455	7.632000	7.501455	7.501455	7.
Netherlands	7.466133	5	7.464000	7.466133	7.452000	7.631000	7.466133	7.502000	7.
Iceland	7.458444	6	7.458444	7.458444	7.458444	6.888000	7.458444	7.458444	7.

6 rows | 1-10 of 20 columns

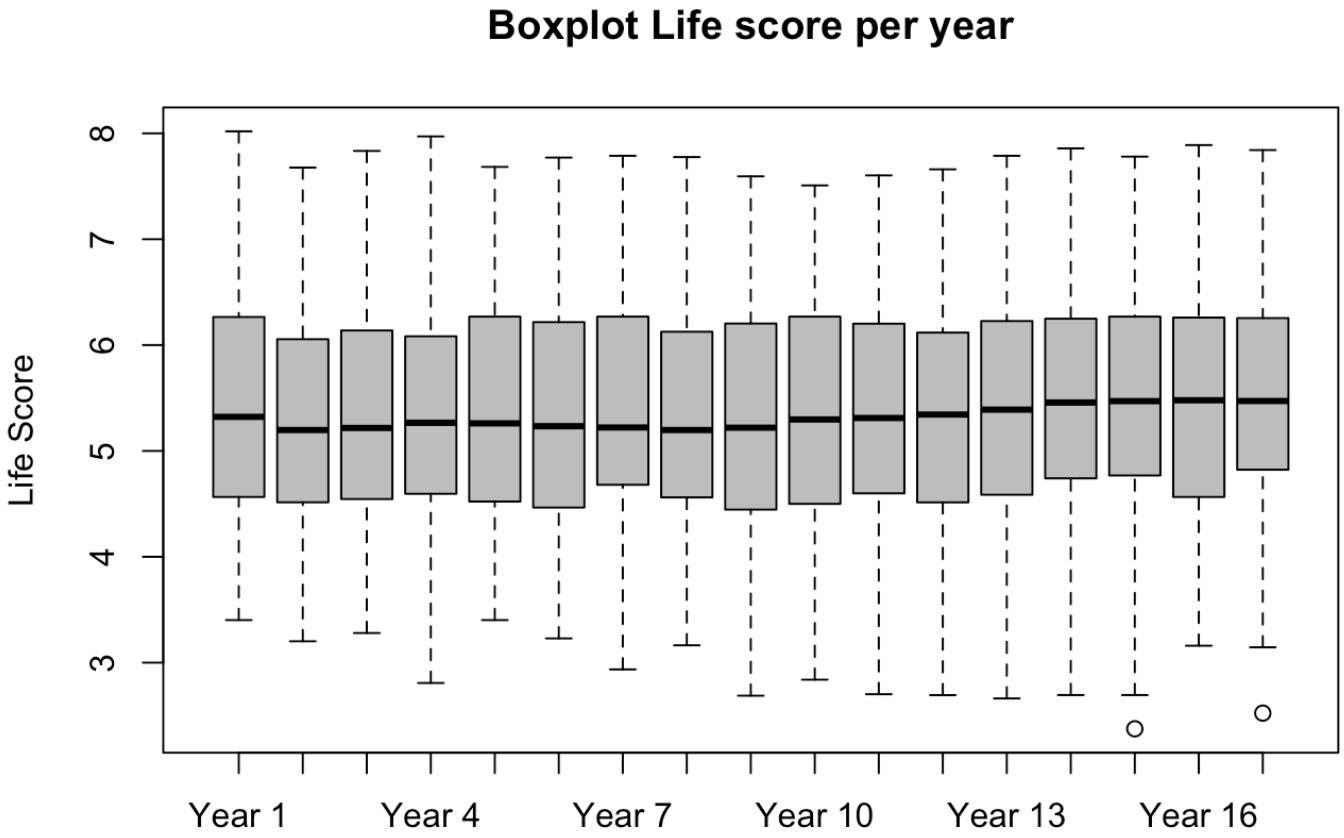
Scan II

Scan the numeric data for outliers.

Steps:

- Apply boxplot() function to numeric data and find if there are any outliers in the dataset.
- From the Box plot, we can find that there is a chance of having an outlier in the “Year 15” and “Year 17” columns which can be ignored because, If the outlier does not change the results but does affect assumptions, you may drop the outlier.
- Find the summary of the column “Average” using z.scores for the normally distributed data. From the method of z score, we found that there are no possible outliers present in the data.

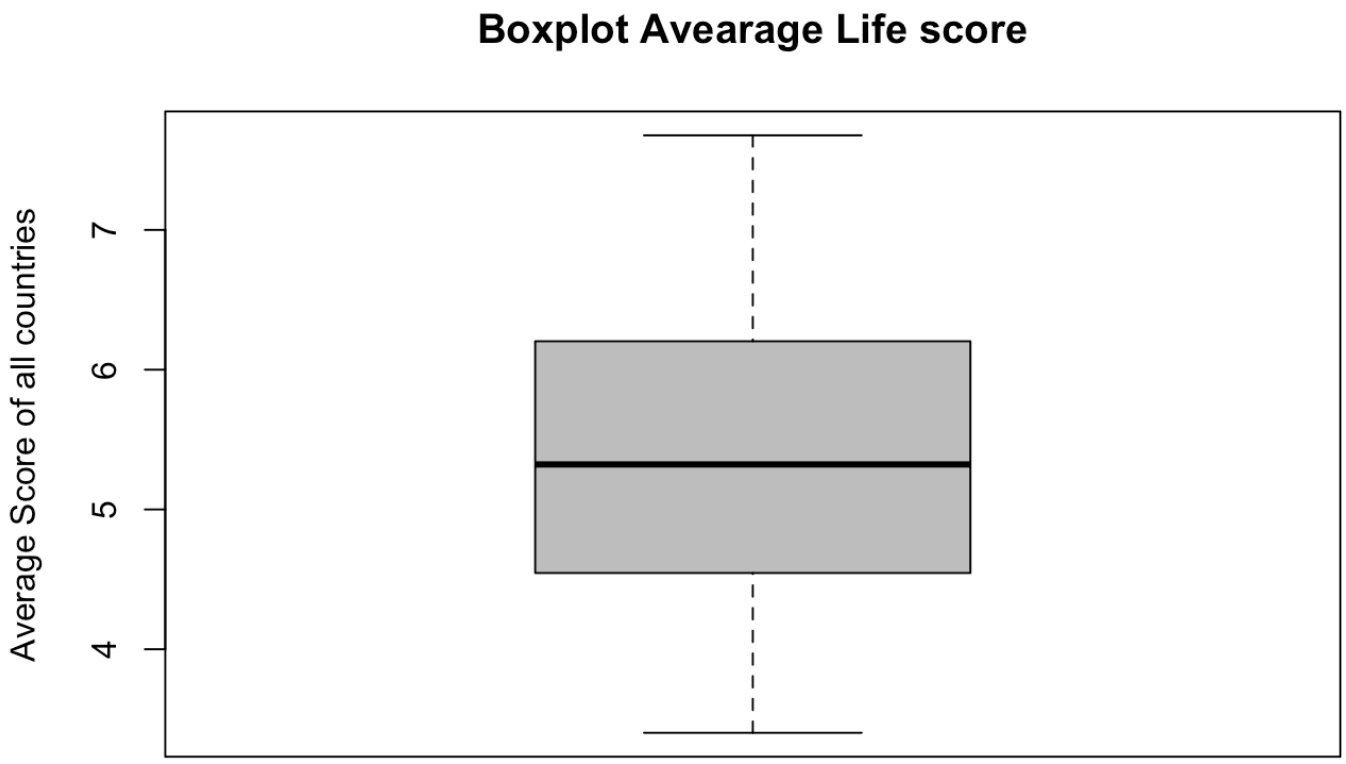
```
# This is the R chunk for the Scan II

merge[,4:20] %>% boxplot(main="Boxplot Life score per year", ylab="Life Score", col
= "grey")
```



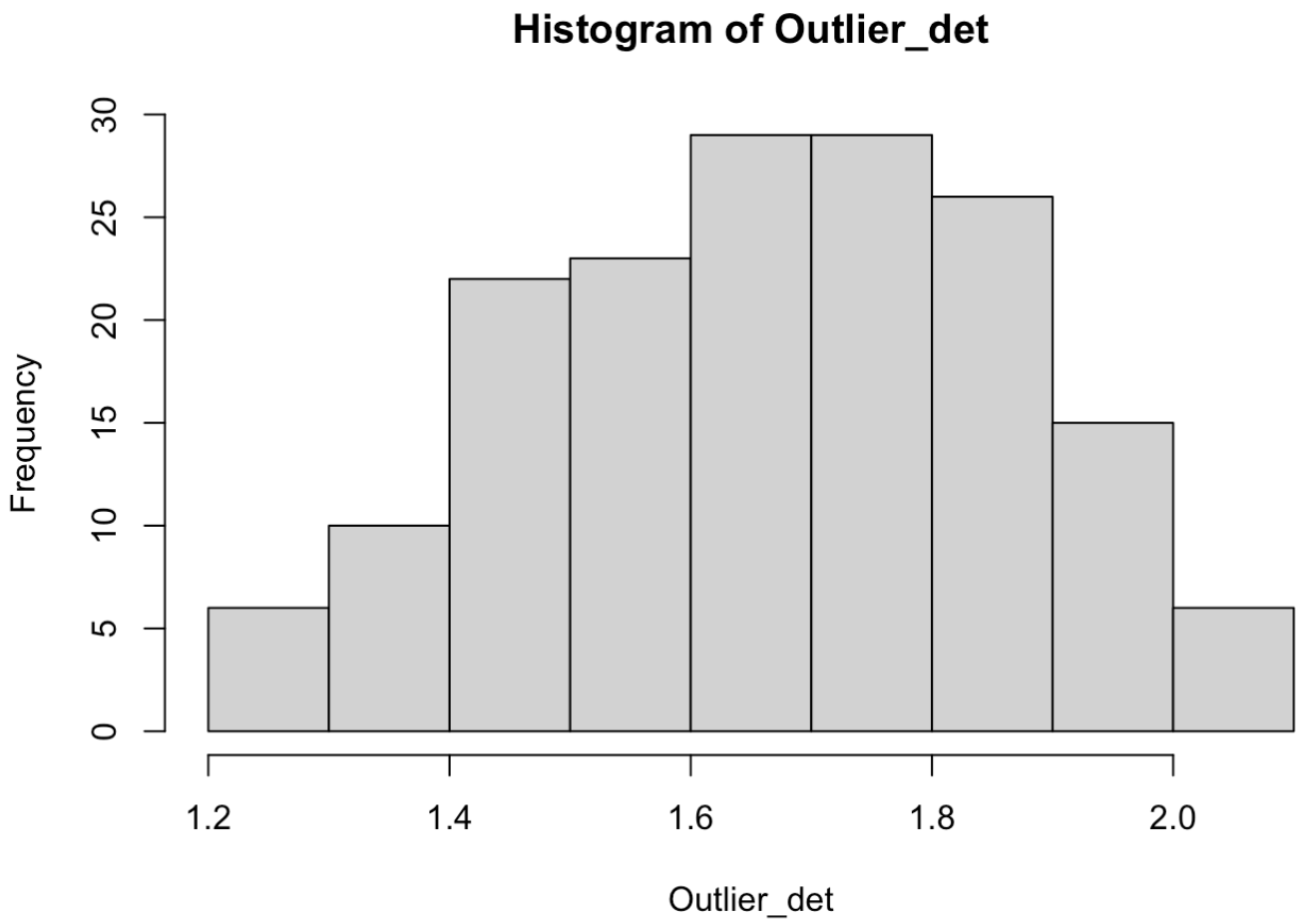

```
head(merge$Average)
```

```
## [1] 7.676625 7.614643 7.550364 7.501455 7.466133 7.458444
```

```
merge$Average %>% boxplot(main="Boxplot Avearage Life score", ylab="Average Score o
f all countries", col = "grey")
```




```
Outlier_det <- log(merge$Average)
hist(Outlier_det)
```




```
z.scores <- Outlier_det %>%  scores(type = "z")
z.scores %>% summary()
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -2.26450 -0.78224  0.01623  0.00000  0.77883  1.88346
```

```
which(abs(z.scores) >3)
```

```
## integer(0)
```

Transform

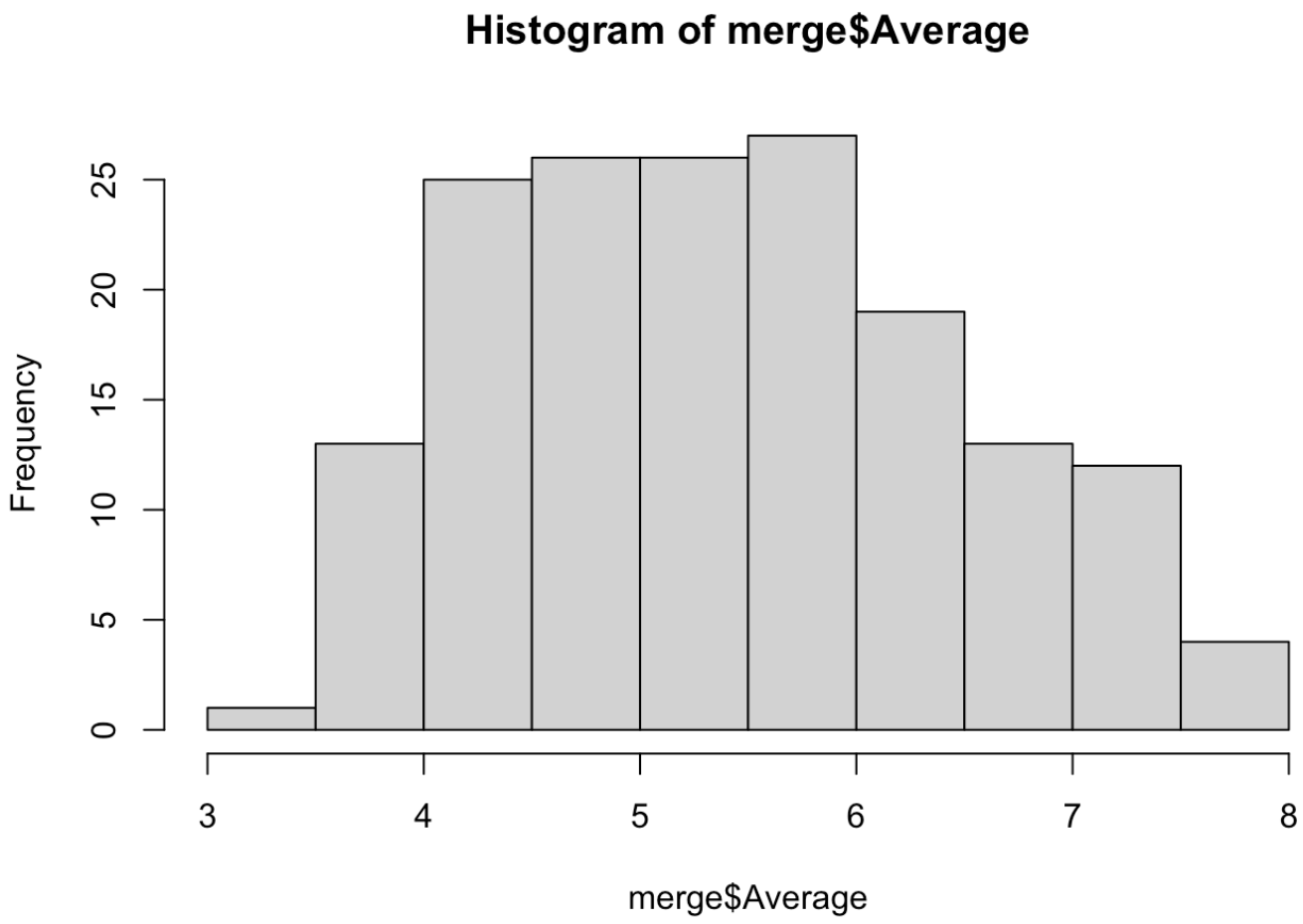
Data transformation is the most important step in the data preprocessing for the development and deployment of statistical analysis and machine learning models.

The purpose of this transformation is to decrease the skewness and convert the distribution into a normal distribution.

Steps:

- The variable Average is not normally distributed from the qq plot and shapiro wilk test performed below.
- In order to achieve normal distribution we need to Transform the data by using different Transformation methods such as Mathematical operations(log, square-root, square, etc.) and BoxCox method.
- For our data, since it is slightly not symmetric, we use Logarithmic method to transform the data and achieve the linear Distribution.
- From the obtained symmetric distribution, we can use tranformed data for the development and deployment of statistical analysis and machine learning models further and gain insights and predict the future trends.

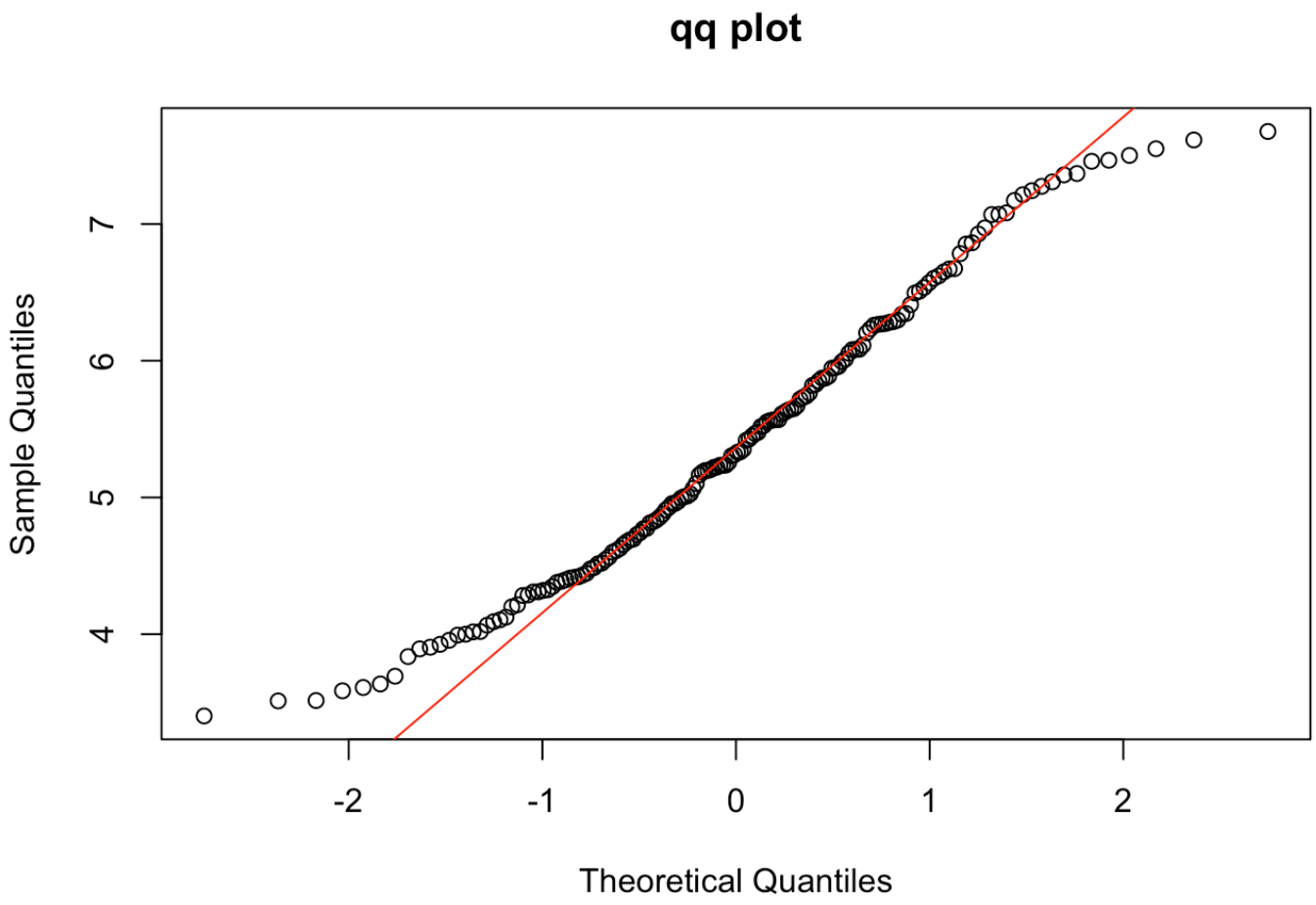
```
# This is the R chunk for the Transform Section
hist(merge$Average)
```



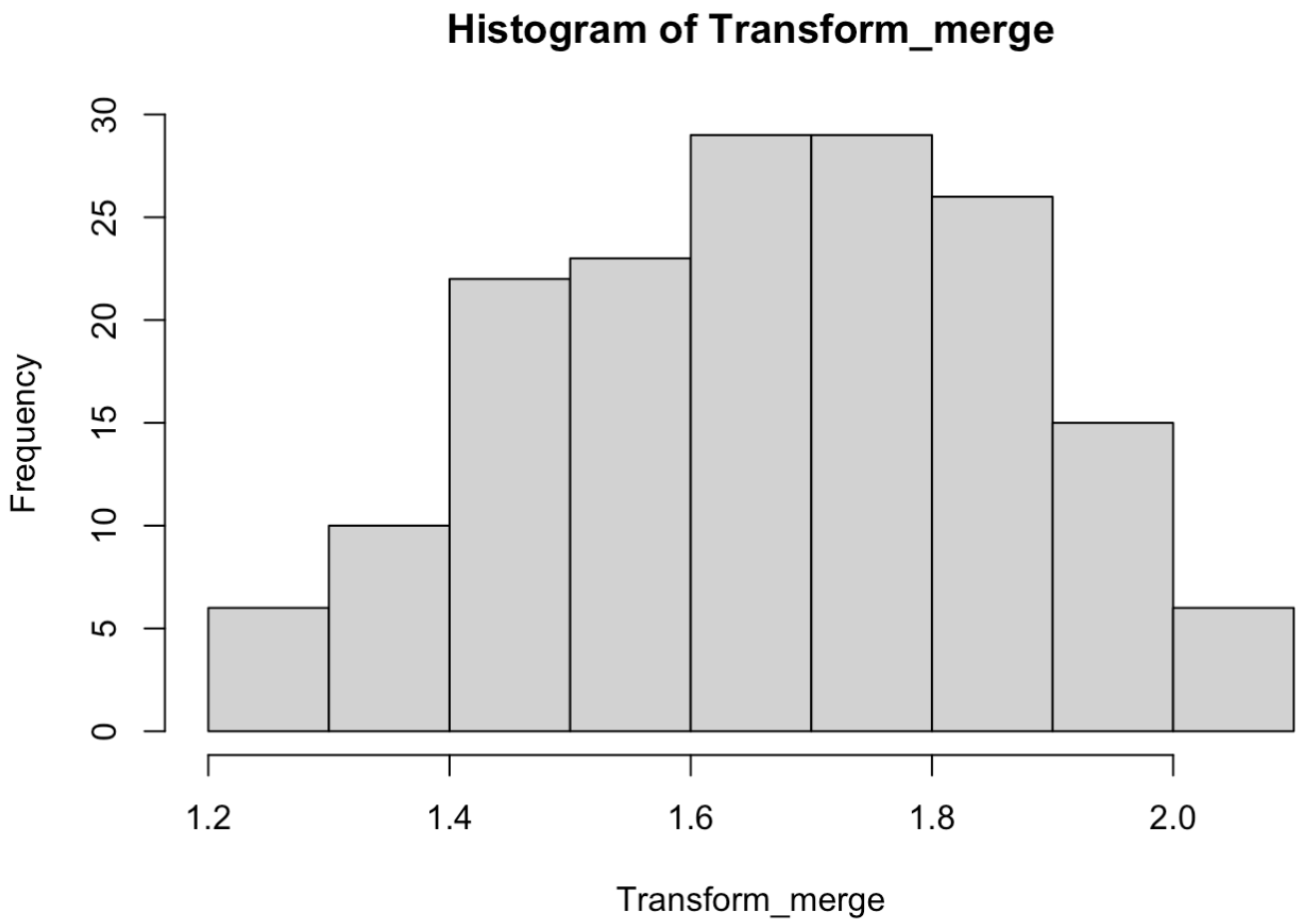

```
shapiro.test(merge$Average)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  merge$Average
## W = 0.97652, p-value = 0.006397
```

```
qqnorm(merge$Average,main="qq plot")
qqline(merge$Average,lwd=1,col="red")
```



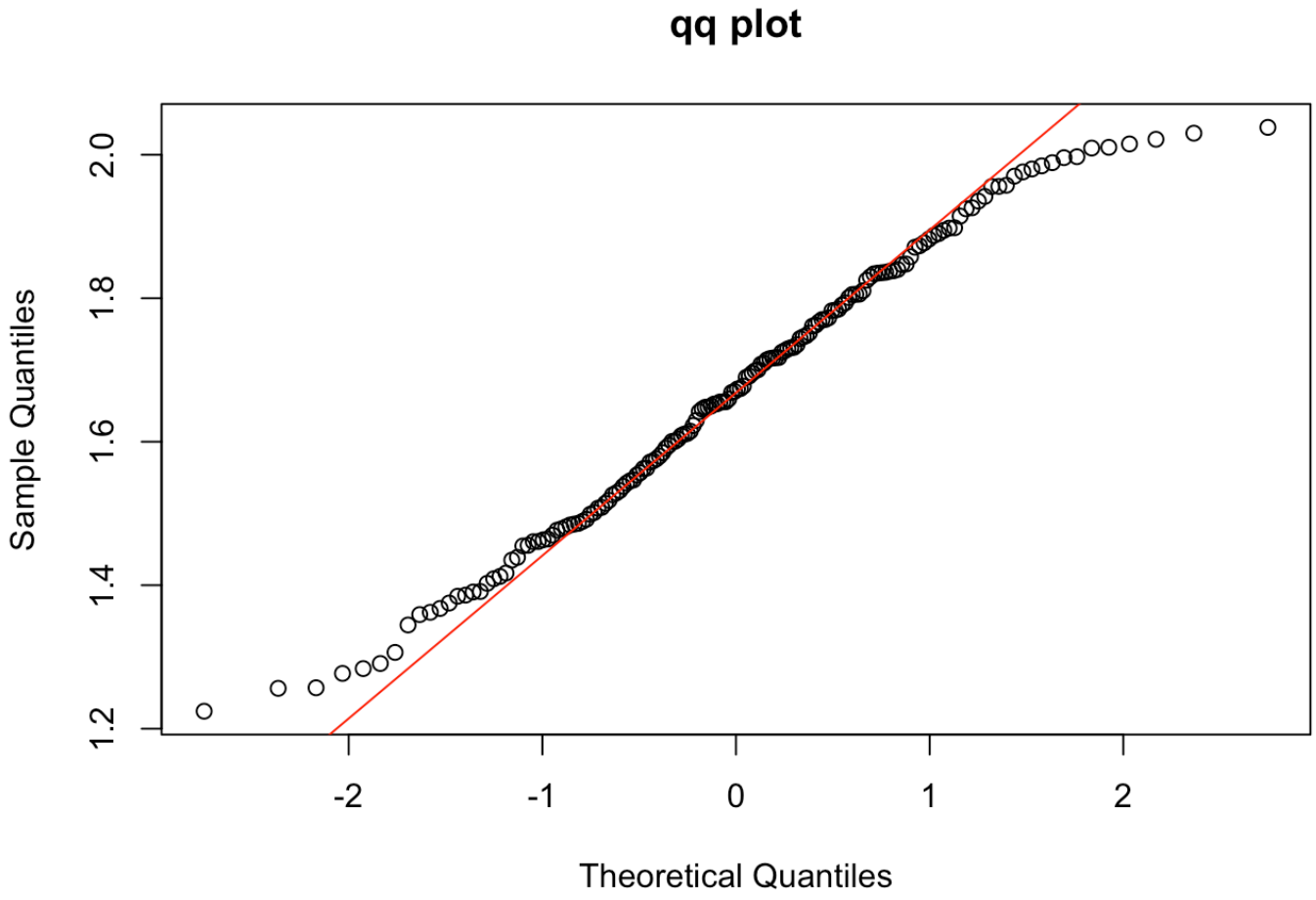

```
Transform_merge <- log(merge$Average)
hist(Transform_merge )
```




```
shapiro.test(Transform_merge )
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Transform_merge
## W = 0.98315, p-value = 0.04156
```

```
qqnorm(Transform_merge,main="qq plot")
qqline(Transform_merge,lwd=1,col="red")
```


qq plot
Sample Quantiles
Theoretical Quantiles
1.2
1.4
1.6
1.8
2.0
-2
-1
0
1
2