MATH2349 Semester 2, 2018

Code ▾

Assignment 3

Udeshika Dissanayake (s3400652)

Required Packages

Below packages and libraries in R have been used in for this Exercise.

Hide

```
library(knitr)
library(ggplot2)
library(readr)
library(tidyr)
library(deductive)
library(validate)
library(Hmisc)
library(stringr)
library(lubridate)
library(outliers)
library(MVN)
library(MASS)
library(caret)
library(dplyr)
```

Executive Summary

The data sets used in this exercise contain world population evolution (from 1960 to 2017) and countries income classification for 264 observations. Firstly, the variables in the data sets have been carefully examined in order to get a proper understanding on the data sets. The two data sets have been merged using the common variable of Country Code. Then, the structure and attributes of the merged data set have been carefully checked. Data types of a few variables have been converted to have a better representation of the data. In order to make the data set tidier, unnecessary variables for the exercise have been dropped. Also a few variables have been re-labeled to have a better representation. After that, the data set have been transformed from wide format to long format. Subsequently, the missing values and special values in the data set have been appropriately treated. The outliers of the data set have been investigated using the z-score method. The numerical variable of "Total_population" has been checked for its distribution using Histogram and identified that its not normal, but strongly right-skewed. By using logarithm base e (ln) transformation, this variable has been converted to normally distributed representation for convenient analysis.

Data

Data source: World Development Indicators - Population dynamics

Retrieved from: https://data.worldbank.org/indicator/SP.POP.TOTL (https://data.worldbank.org/indicator/SP.POP.TOTL)

Data set 1: Total Population

The date set contains total population of 264 countries from year 1960 to 2017. Following are the variables in the data set.

Variables:
* Country Name
* Country Code
* Indicator Name: Population, total
* Indicator Code: SP.POP.TOTL
* Total Population values from the year 1960 to 2017

The total population data set has been loaded in to R Studio using the readr function and then print the data frame to inspect the data. The data set has been labeled as total_pop.

Hide

```
# Skip first 4 rows in csv & load data set
headers = read.csv("API_SP.POP.TOTL_DS2_en_csv_v2_10134466.csv", skip = 4, header
= F,
                   nrows = 1, as.is = T)
total_pop = read.csv("API_SP.POP.TOTL_DS2_en_csv_v2_10134466.csv",skip = 5, header
= F)
colnames(total_pop)= headers
head(total_pop, 4)
```

	Country Name <fctr>	Country Code <fctr>	Indicator Name <fctr>	Indicator Code <fctr>	1960 <dbl>	19 <dl
1	Aruba	ABW	Population, total	SP.POP.TOTL	54211	554
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8996351	91667
3	Angola	AGO	Population, total	SP.POP.TOTL	5643182	57530
4	Albania	ALB	Population, total	SP.POP.TOTL	1608800	16598

4 rows | 1-9 of 63 columns

Data set 2: Income group

The data set contains the classification of income (Income Group) of 264 countries for the year 2017. Following are the variables in the data set.

Variables:
* Country Code
* Region
* IncomeGroup

The income group data set has been loaded in to R Studio using the readr function and then print the data frame to inspect the data frame. The data set has been labeled as income_group.

Hide

```
income_group<-read_csv("Metadata_Country_API_SP.POP.TOTL_DS2_en_csv_v2_10134466.cs
v")
head(income_group, 4)
```

Country Code <chr>	Region <chr>	IncomeGroup <chr>
ABW	Latin America & Caribbean	High income
AFG	South Asia	Low income
AGO	Sub-Saharan Africa	Lower middle income
ALB	Europe & Central Asia	Upper middle income

4 rows | 1-3 of 6 columns

There are 46 records in this data set with no input (blank) for the variable, income group. These records do not represent really countries, but rather represent a group of countries based on geographical regions or economical similarities. A few such examples are ARB - Arab World, CEB - Central Europe and the Balatics, and LMC - Lower Middle Income.

Join two Data sets

The key variable ‘Country Code’ has been used to combine ‘income_group’ and ‘total_pop’ data frames. The combined data frame has been labeled as ‘total_pop_combined’.

Hide

```
total_pop_combined<- total_pop %>% left_join(income_group,by="Country Code")
```

Understand

The function head() has been used to view the first 4 rows of the combined data frame. Then the str () function has been used to check the structure of the data set (i.e. dimension, column names/attributes, types of variables, and levels of categorical variables). The data set consists with 264 observations and 68 variables. As shown below, columns Country Name, Indicator Name, and Indicator Code are factor variables while population of years are numerical variables. Also the Country Code, IncomeGroup, Region, SpecialNotes, TableName, and X6 are character variables.

Hide

```
head(total_pop_combined, 4)
```

	Country Name <fctr>	Country Code <chr>	Indicator Name <fctr>	Indicator Code <fctr>	1960 <dbl>	19 <dl
1	Aruba	ABW	Population, total	SP.POP.TOTL	54211	554
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8996351	91667
3	Angola	AGO	Population, total	SP.POP.TOTL	5643182	57530
4	Albania	ALB	Population, total	SP.POP.TOTL	1608800	16598

4 rows | 1-9 of 68 columns

Hide

```
str(total_pop_combined) # Checking the structure of the data set
```

```
'data.frame':	264 obs. of  68 variables:
 $ Country Name  : Factor w/ 264 levels "Afghanistan",..: 11 1 6 2 5 8 250 9 10
4 ...
 $ Country Code  : chr  "ABW" "AFG" "AGO" "ALB" ...
 $ Indicator Name: Factor w/ 1 level "Population, total": 1 1 1 1 1 1 1 1 1 1 ...
 $ Indicator Code: Factor w/ 1 level "SP.POP.TOTL": 1 1 1 1 1 1 1 1 1 1 ...
 $ 1960          : num  54211 8996351 5643182 1608800 13411 ...
 $ 1961          : num  55438 9166764 5753024 1659800 14375 ...
 $ 1962          : num  56225 9345868 5866061 1711319 15370 ...
 $ 1963          : num  56695 9533954 5980417 1762621 16412 ...
 $ 1964          : num  57032 9731361 6093321 1814135 17469 ...
 $ 1965          : num  57360 9938414 6203299 1864791 18549 ...
 $ 1966          : num  57715 10152331 6309770 1914573 19647 ...
 $ 1967          : num  58055 10372630 6414995 1965598 20758 ...
 $ 1968          : num  58386 10604346 6523791 2022272 21890 ...
 $ 1969          : num  58726 10854428 6642632 2081695 23058 ...
 $ 1970          : num  59063 11126123 6776381 2135479 24276 ...
 $ 1971          : num  59440 11417825 6927269 2187853 25559 ...
 $ 1972          : num  59840 11721940 7094834 2243126 26892 ...
 $ 1973          : num  60243 12027822 7277960 2296752 28232 ...
 $ 1974          : num  60528 12321541 7474338 2350124 29520 ...
 $ 1975          : num  60657 12590286 7682479 2404831 30705 ...
 $ 1976          : num  60586 12840299 7900997 2458526 31777 ...
 $ 1977          : num  60366 13067538 8130988 2513546 32771 ...
 $ 1978          : num  60103 13237734 8376147 2566266 33737 ...
 $ 1979          : num  59980 13306695 8641521 2617832 34818 ...
 $ 1980          : num  60096 13248370 8929900 2671997 36067 ...
 $ 1981          : num  60567 13053954 9244507 2726056 37500 ...
 $ 1982          : num  61345 12749645 9582156 2784278 39114 ...
 $ 1983          : num  62201 12389269 9931562 2843960 40867 ...
 $ 1984          : num  62836 12047115 10277321 2904429 42706 ...
 $ 1985          : num  63026 11783050 10609042 2964762 44600 ...
 $ 1986          : num  62644 11601041 10921037 3022635 46517 ...
 $ 1987          : num  61833 11502761 11218268 3083605 48455 ...
 $ 1988          : num  61079 11540888 11513968 3142336 50434 ...
 $ 1989          : num  61032 11777609 11827237 3227943 52448 ...
 $ 1990          : num  62149 12249114 12171441 3286542 54509 ...
 $ 1991          : num  64622 12993657 12553446 3266790 56671 ...
 $ 1992          : num  68235 13981231 12968345 3247039 58888 ...
 $ 1993          : num  72504 15095099 13403734 3227287 60971 ...
 $ 1994          : num  76700 16172719 13841301 3207536 62677 ...
 $ 1995          : num  80324 17099541 14268994 3187784 63850 ...
 $ 1996          : num  83200 17822884 14682284 3168033 64360 ...
 $ 1997          : num  85451 18381605 15088981 3148281 64327 ...
 $ 1998          : num  87277 18863999 15504318 3128530 64142 ...
 $ 1999          : num  89005 19403676 15949766 3108778 64370 ...
 $ 2000          : num  90853 20093756 16440924 3089027 65390 ...
 $ 2001          : num  92898 20966463 16983266 3060173 67341 ...
 $ 2002          : num  94992 21979923 17572649 3051010 70049 ...
 $ 2003          : num  97017 23064851 18203369 3039616 73182 ...
 $ 2004          : num  98737 24118979 18865716 3026939 76244 ...
 $ 2005          : num  100031 25070798 19552542 3011487 78867 ...
```

```
 $ 2006          : num  100832 25893450 20262399 2992547 80991 ...
 $ 2007          : num  101220 26616792 20997687 2970017 82683 ...
 $ 2008          : num  101353 27294031 21759420 2947314 83861 ...
 $ 2009          : num  101453 28004331 22549547 2927519 84462 ...
 $ 2010          : num  101669 28803167 23369131 2913021 84449 ...
 $ 2011          : num  102053 29708599 24218565 2905195 83751 ...
 $ 2012          : num  102577 30696958 25096150 2900401 82431 ...
 $ 2013          : num  103187 31731688 25998340 2895092 80788 ...
 $ 2014          : num  103795 32758020 26920466 2889104 79223 ...
 $ 2015          : num  104341 33736494 27859305 2880703 78014 ...
 $ 2016          : num  104822 34656032 28813463 2876101 77281 ...
 $ 2017          : num  105264 35530081 29784193 2873457 76965 ...
 $ NA            : logi  NA NA NA NA NA NA ...
 $ Region        : chr  "Latin America & Caribbean" "South Asia" "Sub-Saharan Afric
a" "Europe & Central Asia" ...
 $ IncomeGroup   : chr  "High income" "Low income" "Lower middle income" "Upper mid
dle income" ...
 $ SpecialNotes  : chr  "SNA data for 2000-2011 are updated from official governmen
t statistics; 1994-1999 from UN databases. Base year "| __truncated__ "Fiscal year
end: March 20; reporting period for national accounts data is calendar year, estima
ted to insure co"| __truncated__ NA NA ...
 $ TableName     : chr  "Aruba" "Afghanistan" "Angola" "Albania" ...
 $ X6            : chr  NA NA NA NA ...
```

Data type conversions

The IncomeGroup and Region variables are more appropriate to be a factor data type. However in the data set they are defined as character variables. In below steps, these two variables are converted to factor data types and define the levels.

Hide

```
total_pop_combined$IncomeGroup<- factor(total_pop_combined$IncomeGroup,
                                        levels=c('Low income',
                                                 'Lower middle income',
                                                 'Upper middle income',
                                                 'High income'),
                                        labels = c('Low',
                                                   'Lower middle',
                                                   'Upper middle',
                                                   'High'),
                                        ordered=TRUE)
total_pop_combined$Region <- factor(total_pop_combined$Region,
                                  levels=c('Latin America & Caribbean',
                                           'South Asia',
                                           'Sub-Saharan Africa',
                                           'Europe & Central Asia',
                                           'Middle East & North Africa',
                                           'East Asia & Pacific',
                                           'North America'))
```

The variables Table Name, Special notes, Indicator Code, Indicator Name, and X6 are not required for this exercise, therefore will be dropped in future step. These variables were not considered for data type conversions.

Below is the first four rows of the data type convered variables in the data frame after the conversions.

Hide

```
check_pop<- total_pop_combined %>% select(c(1:2,64:65))
head(check_pop, 4)
```

	Country Name	Country Code	Region	IncomeGroup
	<fctr>	<chr>	<fctr>	<ord>
1	Aruba	ABW	Latin America & Caribbean	High
2	Afghanistan	AFG	South Asia	Low
3	Angola	AGO	Sub-Saharan Africa	Lower middle
4	Albania	ALB	Europe & Central Asia	Upper middle

4 rows

Tidy & Manipulate Data I

Drop unwanted colums

As explained in the previous section, the variables Table Name, Special notes, Indicator Code, Indicator Name, and X6 are dropped to tidy up the data set.

Hide

```
total_pop_comb_tidy<- total_pop_combined %>%
  select(-`TableName`,-X6,-`Indicator Name`,-`Indicator Code`,-`NA`,-SpecialNotes)
```

Tidy up the Data

Checking the data set against the Tidy Data Principles, it deemed the data set is untidy as some values of a real variable are in columns (example- column names 1960 to 2017 represent value of the year variable). The gather() function of “tidyr” has been used to transform the data frame from wide format to long format.

Hide

```
total_pop_comb_tidy<- total_pop_comb_tidy %>%
  gather(key="Year", value ="Total_population", 3:60)
#Converting the data type of column "Year"
total_pop_comb_tidy$Year <- as.integer(total_pop_comb_tidy$Year)
```

Relabelling column names.

Some of the variable names have been re-labelled to have more meaningful names as bellow:

Hide

```
total_pop_comb_tidy <- rename(total_pop_comb_tidy,
                              Country_Name=`Country Name`,
                              Country_Code=`Country Code`,
                              IncomeGroup2017=IncomeGroup)
# First 4 obervations of the data frame
head(total_pop_comb_tidy, 4)
```

	Country_Na... <fctr>	Country_Code <chr>	Region <fctr>	IncomeGroup2017 <ord>
1	Aruba	ABW	Latin America & Caribbean	High
2	Afghanistan	AFG	South Asia	Low
3	Angola	AGO	Sub-Saharan Africa	Lower middle
4	Albania	ALB	Europe & Central Asia	Upper middle

Tidy & Manipulate Data II

Annual population growth in 2017

As described below, a new variable has been created as Growth_2017, which represents the annual growth of population. A subset of data has been created by filtering the tidy up data set for 2016 and 2017 observations. The formula outlined in below code has been used to get the values for Growth_2017. Subsequently, the data set has been filtered out only for 2017 observations. Below are the first four entries of the data set total_pop_2017.

Hide

```
# Filter rows with 2016 & 2017
total_pop_2017<- total_pop_comb_tidy %>% filter(Year==2017|Year==2016)
# Calculate populatin growth 2017
total_pop_growth_2017<- total_pop_comb_tidy %>% filter(Year==2017|Year==2016) %>%
  group_by(Country_Name) %>% mutate(Growth_2017 =
                                      ((Total_population/lag(Total_population) -
1) * 100)) %>%
  filter(Year==2017) %>% select(-(Year))
head(total_pop_growth_2017, 4)
```

Country_Na... <fctr>	Country_Code <chr>	Region <fctr>	IncomeGroup2017 <ord>	T
Aruba	ABW	Latin America & Caribbean	High	
Afghanistan	AFG	South Asia	Low	
Angola	AGO	Sub-Saharan Africa	Lower middle	
Albania	ALB	Europe & Central Asia	Upper middle	

4 rows

Scan I

Checking and dealing with missing values

Below codes have been executed to identify the missing values in the ‘Total_population’ column:

Hide

```
#rows with missing values in the column Total_population
total_pop_comb_tidy_missing<- total_pop_comb_tidy %>% subset(is.na(Total_populatio
n))
head(total_pop_comb_tidy_missing)
```

	Country_Name <fctr>	Country_Code <chr>	Region <fctr>	I
109	Not classified	INX	NA	
195	West Bank and Gaza	PSE	Middle East & North Africa	
213	Serbia	SRB	Europe & Central Asia	
224	Sint Maarten (Dutch part)	SXM	Latin America & Caribbean	
373	Not classified	INX	NA	
459	West Bank and Gaza	PSE	Middle East & North Africa	

6 rows | 1-6 of 6 columns

Hide

```
#Calculating total missing values in the column Total_population in each country
total_pop_comb_tidy_missing_count<- total_pop_comb_tidy_missing %>% group_by(Countr
y_Name) %>%
  summarise(NA_count=sum(is.na(Total_population)))
total_pop_comb_tidy_missing_count
```

Country_Name <fctr>	NA_count <int>
Eritrea	6
Kuwait	3
Not classified	58
Serbia	30
Sint Maarten (Dutch part)	38
West Bank and Gaza	30

6 rows

Above are the population missing frequencies for each countries.
The “Not Classified” observations are not related to any real countries and those have no data for any other variables. Therefore those Not Classified entries can safely be dropped.

The missing population values for other aforementioned countries have been replaced by population data in adjacent year.

Hide

```
# Drop Not Classified  entries
total_pop_comb_tidy_clean<-  total_pop_comb_tidy[!total_pop_comb_tidy$Country_Name=
="Not classified",]
# Replace missing population by population data in adjacent year
total_pop_comb_tidy_clean<- total_pop_comb_tidy_clean %>% group_by(Country_Name) %
>%
  fill(Total_population,.direction = "down")
total_pop_comb_tidy_clean<- total_pop_comb_tidy_clean %>% group_by(Country_Name) %
>%
  fill(Total_population,.direction = "up")
```

Similarly, below codes have been executed to identify & fix the missing values in other variables.

Hide

```
#rows with missing values in the column Country_Name
total_pop_comb_tidy_clean %>% subset(is.na(Country_Name))
```

0 rows

Hide

```
#rows with missing values in the column Country_Code
total_pop_comb_tidy_clean %>% subset(is.na(Country_Code))
```

0 rows

Hide

```
#rows with missing values in the column Region
total_pop_comb_tidy_clean %>% subset(is.na(Region))
```

Country_Name <fctr>	Country_Code <chr>	Reg… <fctr>	IncomeGroup2017 <ord>	Y… <int>	Total
Arab World	ARB	NA	NA	1960	
Arab World	ARB	NA	NA	1961	
Arab World	ARB	NA	NA	1962	
Arab World	ARB	NA	NA	1963	
Arab World	ARB	NA	NA	1964	
Arab World	ARB	NA	NA	1965	
Arab World	ARB	NA	NA	1966	
Arab World	ARB	NA	NA	1967	
Arab World	ARB	NA	NA	1968	
Arab World	ARB	NA	NA	1969	

1-10 of 2,668 rows	Previous 1 2 3 4 5 6 ... 100 Next

Hide

```
#Calculating total missing values in the column Region in each country
total_pop_comb_tidy_clean_count<- total_pop_comb_tidy_clean %>% subset(is.na(Regio
n)) %>%
  group_by(Country_Name) %>% summarise(NA_count=sum(is.na(Region)))
total_pop_comb_tidy_clean_count
```

Country_Name <fctr>	NA_count <int>
Arab World	58
Caribbean small states	58
Central Europe and the Baltics	58
Early-demographic dividend	58
East Asia & Pacific	58
East Asia & Pacific (excluding high income)	58
East Asia & Pacific (IDA & IBRD countries)	58
Euro area	58
Europe & Central Asia	58
Europe & Central Asia (excluding high income)	58

1-10 of 46 rows Previous 1 2 3 4 5 Next

There were 2668 entries with missing Region values as can be seen above. When those missing values are classified against frequencies, it shows that there were 46 countries with fully missing (from 1960-2017) region value. Carefully considering, it is understood these are not real countries, but rather regions themselves. Therefore those observations can safely be dropped down.

Hide

```
total_pop_clean<- total_pop_comb_tidy_clean[complete.cases(total_pop_comb_tidy_clea
n),]
```

Finally, below codes have been executed to identify the missing values in the IncomeGroup2017 and Year columns.

Hide

```
#rows with missing values in the column IncomeGroup2017
total_pop_clean %>% subset(is.na(IncomeGroup2017))
```

0 rows

Hide

```
#rows with missing values in the column Year
total_pop_clean %>% subset(is.na(Year))
```

0 rows

check for special values

Below codes have been run to identify the inconsistencies or special values in Total_population column.

Hide

```
#check input whether they are not infinite or NA unsing a fuction called is.special
is.special<- function(x){
  if(is.numeric(x)) !is.finite(x) else is.na(x)
  }
is.special<- function(x){
  if(is.numeric(x)) !is.finite(x)
  }
#Applly is.special function to the Total_population column
a<- sapply(total_pop_clean$Total_population, is.special)
#Count the number of special values in the Total_population column
length(a[a=="TRUE"])
```

```
[1] 0
```

Similarly, respective code has been run to identify the inconsistencies or special values in other columns. It was noticed that zero inconsistencies or special values were identified in other columns.

To make sure there is no further missing values/inconsistencies, below code has been run.

Hide

```
total_pop_clean[!complete.cases(total_pop_clean),]
```

0 rows

Below is the final data set after performing all above mentioned data preprocessing.

Hide

```
total_pop_clean
```

Country_Na... <fctr>	Country_Code <chr>	Region <fctr>	IncomeGroup2017 <ord>	Y... <int>
Afghanistan	AFG	South Asia	Low	1960
Afghanistan	AFG	South Asia	Low	1961
Afghanistan	AFG	South Asia	Low	1962

Country_Na…	Country_Code	Region	IncomeGroup2017	Y…	T
<fctr>	<chr>	<fctr>	<ord>	<int>	
Afghanistan	AFG	South Asia	Low	1963	
Afghanistan	AFG	South Asia	Low	1964	
Afghanistan	AFG	South Asia	Low	1965	
Afghanistan	AFG	South Asia	Low	1966	
Afghanistan	AFG	South Asia	Low	1967	
Afghanistan	AFG	South Asia	Low	1968	
Afghanistan	AFG	South Asia	Low	1969	

1-10 of 12,586 rows

Previous **1** 2 3 4 5 6 ... 100 Next

Scan II - Checking for outliers

The only numerical variable to be checked for outliers in the data set is Total_population. The outliers have been investigated using the z-score method for Total_population as shown in below code:

Hide

```
z.scores <- total_pop_clean$Total_population %>%  scores(type = "z")
z.scores %>% summary()
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-0.2376 -0.2332 -0.1972  0.0000 -0.1094 13.6001 
```

Hide

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

```
  [1]  2379  2380  2381  2382  2383  2384  2385  2386  2387  2388  2389  2390  2391
2392  2393  2394
 [17]  2395  2396  2397  2398  2399  2400  2401  2402  2403  2404  2405  2406  240
7  2408  2409  2410
 [33]  2411  2412  2413  2414  2415  2416  2417  2418  2419  2420  2421  2422  242
3  2424  2425  2426
 [49]  2427  2428  2429  2430  2431  2432  2433  2434  2435  2436  5163  5164  516
5  5166  5167  5168
 [65]  5169  5170  5171  5172  5173  5174  5175  5176  5177  5178  5179  5180  518
1  5182  5183  5184
 [81]  5185  5186  5187  5188  5189  5190  5191  5192  5193  5194  5195  5196  519
7  5198  5199  5200
 [97]  5201  5202  5203  5204  5205  5206  5207  5208  5209  5210  5211  5212  521
3  5214  5215  5216
[113]  5217  5218  5219  5220 12006
```

Hide

```
Total_pop_outliers<- total_pop_clean[which( abs(z.scores) >3 ),] %>% count(Country_
Name)
Total_pop_outliers
```

Country_Name <fctr>	**n** <int>
China	58
India	58
United States	1

3 rows

China and India came as outliers in the data set due to its large number of population. Hence, the analysis for China and India should be done separate to the other countries The last entry (2017) for United States has also come as an outlier, but could be ignored as an outlier after checking the value.
China and India has been dropped from the dataframe as below:

Hide

```
Total_pop_last <- total_pop_clean %>% filter(Country_Name!="China", Country_Name!
="India")
Total_pop_last
```

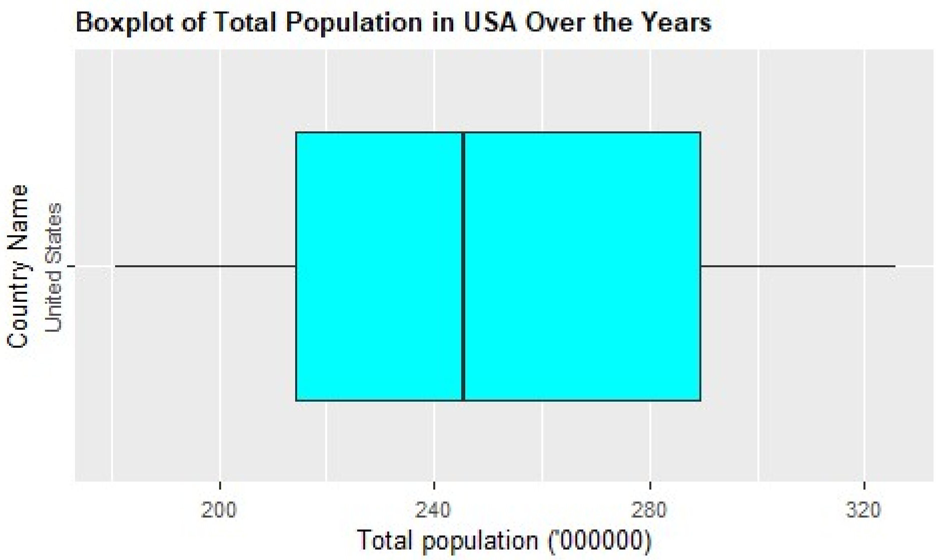
Country_Na… <fctr>	Country_Code <chr>	Region <fctr>	IncomeGroup2017 <ord>	Y… <int>	T
Afghanistan	AFG	South Asia	Low	1960	
Afghanistan	AFG	South Asia	Low	1961	
Afghanistan	AFG	South Asia	Low	1962	
Afghanistan	AFG	South Asia	Low	1963	
Afghanistan	AFG	South Asia	Low	1964	
Afghanistan	AFG	South Asia	Low	1965	
Afghanistan	AFG	South Asia	Low	1966	
Afghanistan	AFG	South Asia	Low	1967	
Afghanistan	AFG	South Asia	Low	1968	
Afghanistan	AFG	South Asia	Low	1969	

1-10 of 12,470 rows Previous **1** 2 3 4 5 6 ... 100 Next

In addition to the above, the outliers have been checked in Total_population variable for each countries separately. Below code shows the box plot and z-score derived for a sample country United States.

Hide

```
library(car)
#Drawing the boxplot to see the outliers for China
total_pop_clean_USA<- Total_pop_last %>% filter(Country_Name=="United States")
plot1<- ggplot(data=total_pop_clean_USA,
               aes(x=Country_Name,y=Total_population/1000000))
plot1<- plot1+geom_boxplot(outlier.colour="red", outlier.shape=8, outlier.size=4, f
ill="cyan")+theme(legend.position = "none",
        plot.title = element_text(lineheight=1, face="bold",size=10),
        axis.text.x = element_text(hjust = 0.5, vjust = 0.5,size=8),
        axis.text.y = element_text(hjust = 0.5, vjust = 0.5,angle=90,size=8),
        axis.title = element_text(hjust = 0.5, vjust = 0.5,size=10),
        legend.title = element_blank())+
  labs(x="Country Name", y="Total population ('000000)",
       title_main="Boxplot of Total Population in USA Over the Years")+
  coord_flip()
plot1
```



z- score approach to see the outliers for USA

Hide

```
z.score_USA <- total_pop_clean_USA$Total_population %>%  scores(type = "z")
z.score_USA %>% summary()
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-1.6210 -0.8470 -0.1289  0.0000  0.8773  1.7092 
```

Hide

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

```
integer(0)
```

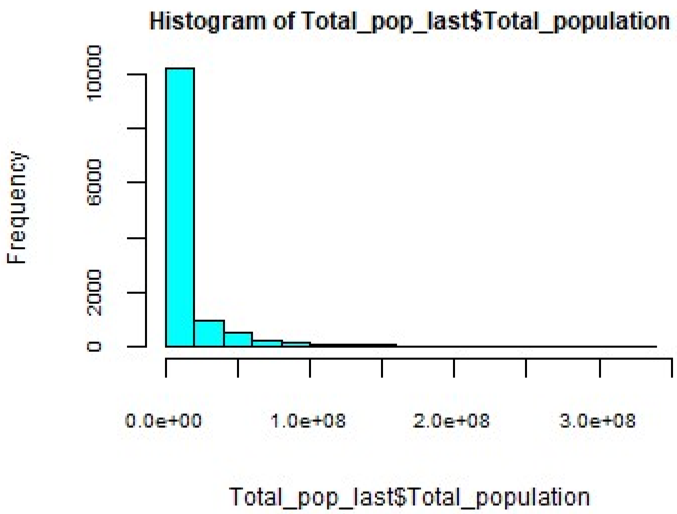
As per the boxplot & z-score, it can be concluded that there are no outliers in Total_population in z.score_USA. Similarly, the outliers have been investigated for other 58 countries.

Transform

The only variable qualified for transformation in the data set is Total_population.The distribution of Total_population is strongly right skewed and not normal.
The below figure shows the histogram of Total_population:

Hide

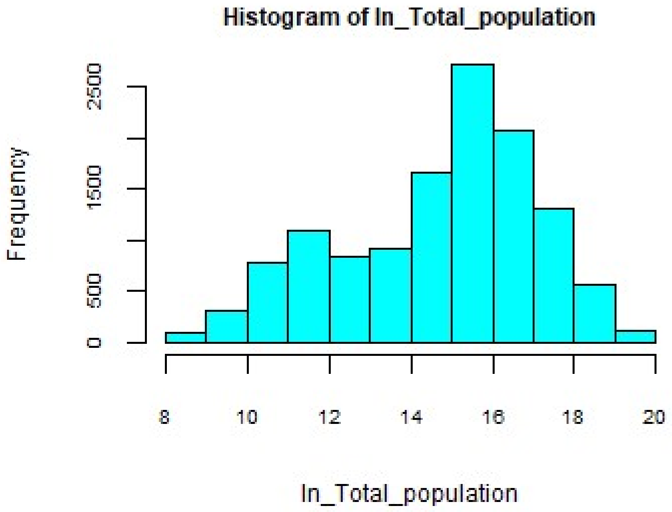
```
#Histogram of Total_population
hist(Total_pop_last$Total_population,
     border="black",col="cyan",cex.main=0.75,cex.axis=0.6,cex.lab=0.75)
```



Different transformations (logarithm, roots, reciprocal & BoxCox) have been applied to Total_population to eliminate the right skewness and to make it normal. The logarithm base e (ln) transformation is identified to be the best transformation out of all for this particular data set. Below is the histogram after the transformation. It can be seen that it is much close to normal after the transformation.

Hide

```
#Histogram of ln_Total_population
ln_Total_population<-log(Total_pop_last$Total_population)
hist(ln_Total_population,
     border="black",col="cyan",cex.main=0.75,cex.axis=0.6,cex.lab=0.75)
```



Histogram of ln_Total_population
Frequency
0
500
1500
2500
8
10
12
14
16
18
20
ln_Total_population