School of Information Studies
SYRACUSE UNIVERSITY

Life Expectancy(WHO)

Analysis on factors influencing Life Expectancy

IST 652

Submitted by:
Isha Halvaldar
Trisha Chakraborty
Vidisha Badhe
Information Management Graduate Student
Class of 2020

Data Description

Life Expectancy (WHO)

Our study will focus on immunization factors, mortality factors, economic factors, social factors and other health related factors as well. Since the observations in this dataset are based on different countries, it will be easier to determine the predicting factor which is contributing to lower value of life expectancy. This will help in suggesting a country which area should be given importance in order to efficiently improve the life expectancy of its population.

Dataset Description

- The project relies on accuracy of data. The Global Health Observatory (GHO) data repository under World Health Organization (WHO) keeps track of the health status as well as many other related factors for all countries. The data-sets are made available to the public for the purpose of health data analysis. This dataset is composed of data from all over the world from various countries aggregated by the World Health Organization (WHO for short). The data is an aggregate of many indicators for a particular country in a particular year. In essence, the data is multiple indicators in a time series separated by country.
- The dataset related to life expectancy, health factors for 193 countries has been collected from the same WHO data repository website and its corresponding economic data was collected from the United Nation website.
- This dataset consists of 25 Columns and 2938 rows.

Dataset Columns

- Country
- Year
- Status - Developed or Developing status
- Life expectancy - Life Expectancy in age
- Adult Mortality - Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population)
- Infant deaths - Number of Infant Deaths per 1000 population
- Alcohol - Alcohol, recorded per capita (15+) consumption (in litres of pure alcohol)
- Percentage Expenditure - Expenditure on health as a percentage of Gross Domestic Product per capita(%)
- Hepatitis B - Hepatitis B (HepB) immunization coverage among 1-year-olds (%)

- Measles - Measles - number of reported cases per 1000 population
- BMI - Average Body Mass Index of entire population
- Under-five deaths - Number of under-five deaths per 1000 population
- Polio - Polio (Pol3) immunization coverage among 1-year-olds (%)
- Total Expenditure - General government expenditure on health as a percentage of total government expenditure (%)
- Diphtheria - Diphtheria tetanus toxoid and pertussis (DTP3) immunization coverage among 1-year-olds (%)
- HIV/AIDS - Deaths per 1 000 live births HIV/AIDS (0-4 years)
- GDP - Gross Domestic Product per capita (in USD)
- Population - Population of the country
- Thinness 1-19 years - Prevalence of thinness among children and adolescents for Age 10 to 19 (%)
- Thinness 5-9 years - Prevalence of thinness among children for Age 5 to 9(%)
- Income composition of resources - Human Development Index in terms of income composition of resources (index ranging from 0 to 1)
- Schooling - Number of years of Schooling(years)
- Country Code
- Region - All the countries are divided into regions - East Asia and Pacific, Europe and Central Asia, Latin America and Caribbean, Middle East and North Africa, North America, South Asia, Sub-Saharan Africa
- Income Group - Income groups are divided into - High income, Low income, Uppermiddle income, Lower middle income

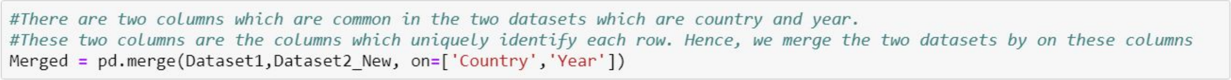
Data Source

- Source 1: Kaggle(https://www.kaggle.com/kumarajarshi/life-expectancy-who)
- Source 2: https://data.world/

Data Wrangling

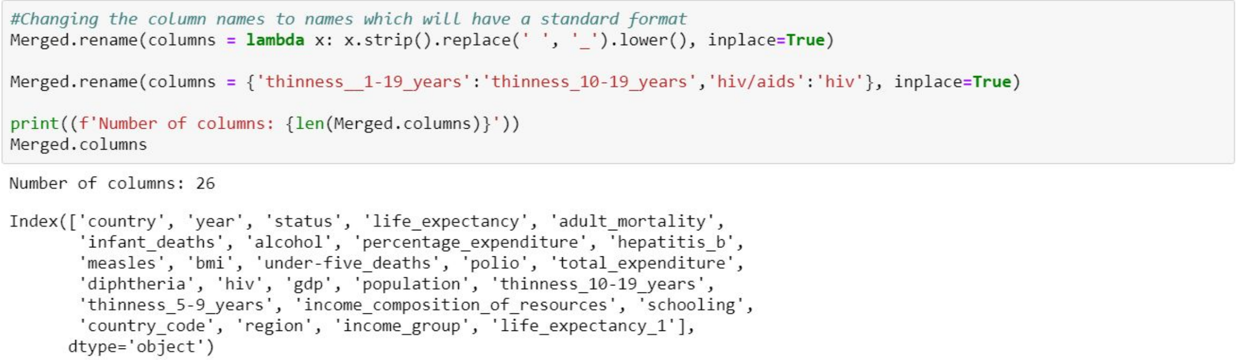
We started our analysis by first prepossessing our data so that we could have data cleant and in a standard format to perform analysis. This would help us to get accurate results and have strong and powerful data insights.

- In this project, we are using data from two different data sources. It was important to merge the two datasets based on the Country and Year so that there would not be duplicate records.



```
#There are two columns which are common in the two datasets which are country and year.
#These two columns are the columns which uniquely identify each row. Hence, we merge the two datasets by on these columns
Merged = pd.merge(Dataset1,Dataset2_New, on=['Country','Year'])
```

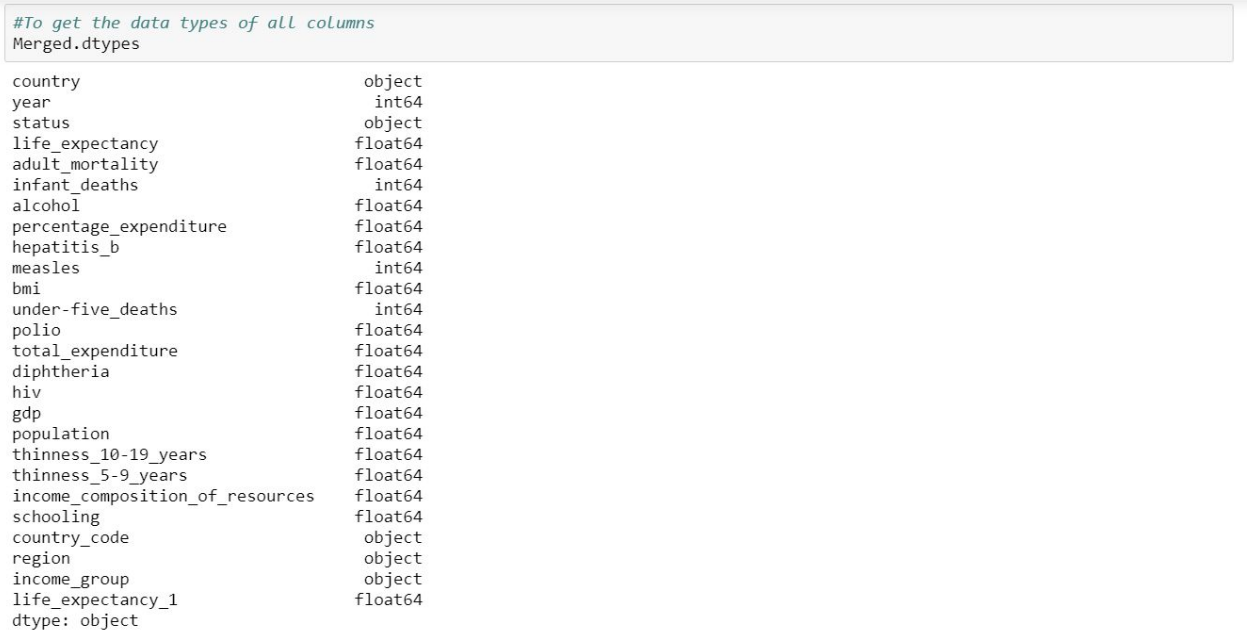
- We first remaned all the columns in the dataset by removing blank spaces of characters like ‘-’,’_’. This initial remaining of columns would help us in further analysing the data.



```
#Changing the column names to names which will have a standard format
Merged.rename(columns = lambda x: x.strip().replace(' ', '_').lower(), inplace=True)

Merged.rename(columns = {'thinness__1-19_years':'thinness_10-19_years','hiv/aids':'hiv'}, inplace=True)

print((f'Number of columns: {len(Merged.columns)}'))
Merged.columns
```

```
Number of columns: 26

Index(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv', 'gdp', 'population', 'thinness_10-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling',
       'country_code', 'region', 'income_group', 'life_expectancy_1'],
      dtype='object')
```

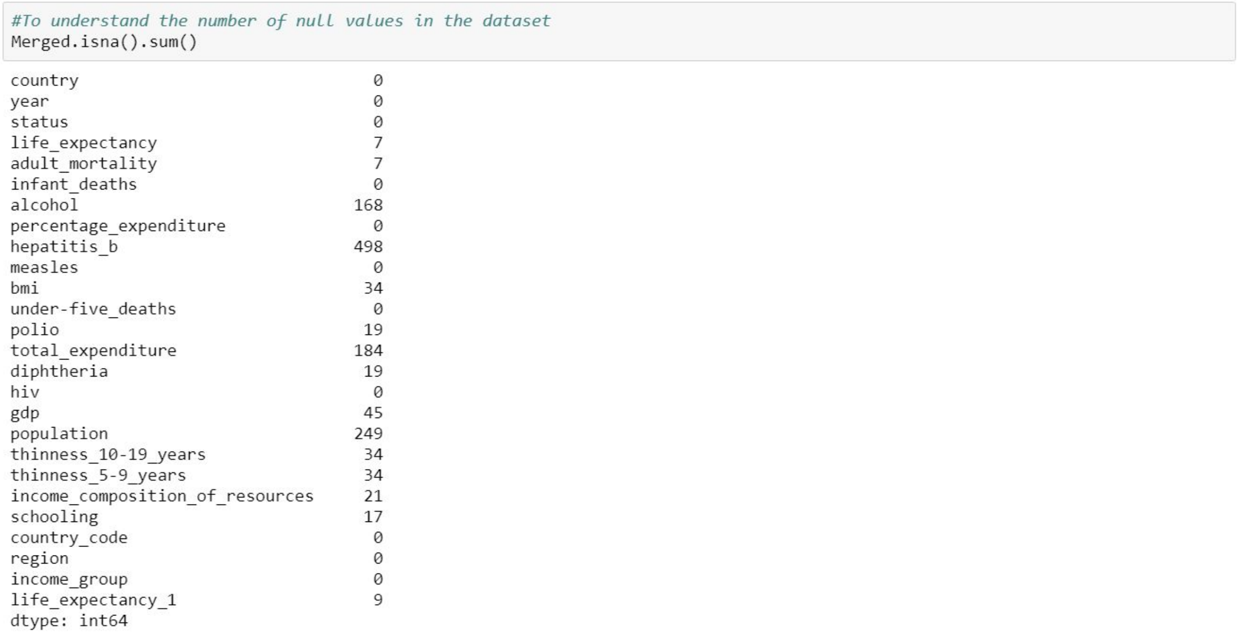
- We used the function ‘dtypes()’ to understand the data types of the columns.



```
#To get the data types of all columns
Merged.dtypes
```

```
country                             object
year                                 int64
status                              object
life_expectancy                    float64
adult_mortality                    float64
infant_deaths                        int64
alcohol                            float64
percentage_expenditure             float64
hepatitis_b                        float64
measles                              int64
bmi                                float64
under-five_deaths                    int64
polio                              float64
total_expenditure                  float64
diphtheria                         float64
hiv                                float64
gdp                                float64
population                         float64
thinness_10-19_years               float64
thinness_5-9_years                 float64
income_composition_of_resources    float64
schooling                          float64
country_code                        object
region                              object
income_group                        object
life_expectancy_1                  float64
dtype: object
```

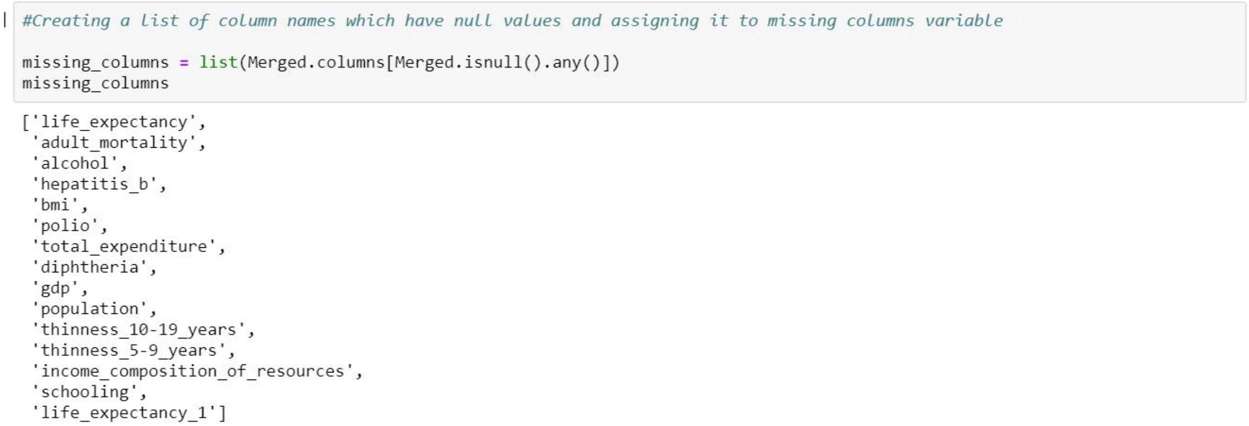
- We used the function ‘isna()’ to fill out null or NA values in the dataset. This gave us a comprehensive view of which columns had the null values.



```
#To understand the number of null values in the dataset
Merged.isna().sum()
```

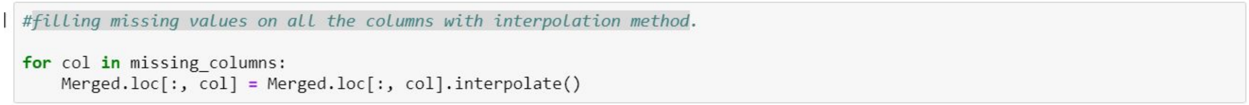
```
country                            0
year                               0
status                             0
life_expectancy                    7
adult_mortality                    7
infant_deaths                      0
alcohol                          168
percentage_expenditure             0
hepatitis_b                      498
measles                            0
bmi                               34
under-five_deaths                  0
polio                             19
total_expenditure                184
diphtheria                        19
hiv                                0
gdp                               45
population                       249
thinness_10-19_years              34
thinness_5-9_years                34
income_composition_of_resources   21
schooling                         17
country_code                       0
region                             0
income_group                       0
life_expectancy_1                  9
dtype: int64
```

- We then filled all the missing values in the columns by using ‘interpolate()’ function.

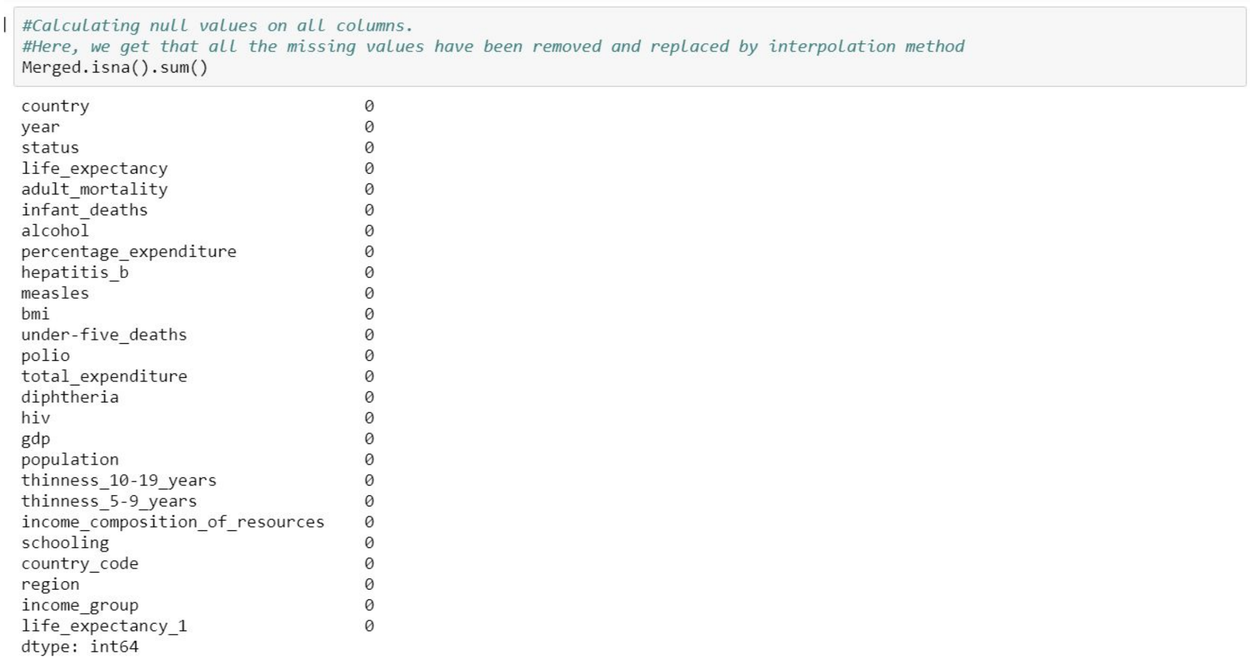


```
#Creating a list of column names which have null values and assigning it to missing columns variable

missing_columns = list(Merged.columns[Merged.isnull().any()])
missing_columns
```

```
['life_expectancy',
 'adult_mortality',
 'alcohol',
 'hepatitis_b',
 'bmi',
 'polio',
 'total_expenditure',
 'diphtheria',
 'gdp',
 'population',
 'thinness_10-19_years',
 'thinness_5-9_years',
 'income_composition_of_resources',
 'schooling',
 'life_expectancy_1']
```



```
#filling missing values on all the columns with interpolation method.

for col in missing_columns:
    Merged.loc[:, col] = Merged.loc[:, col].interpolate()
```



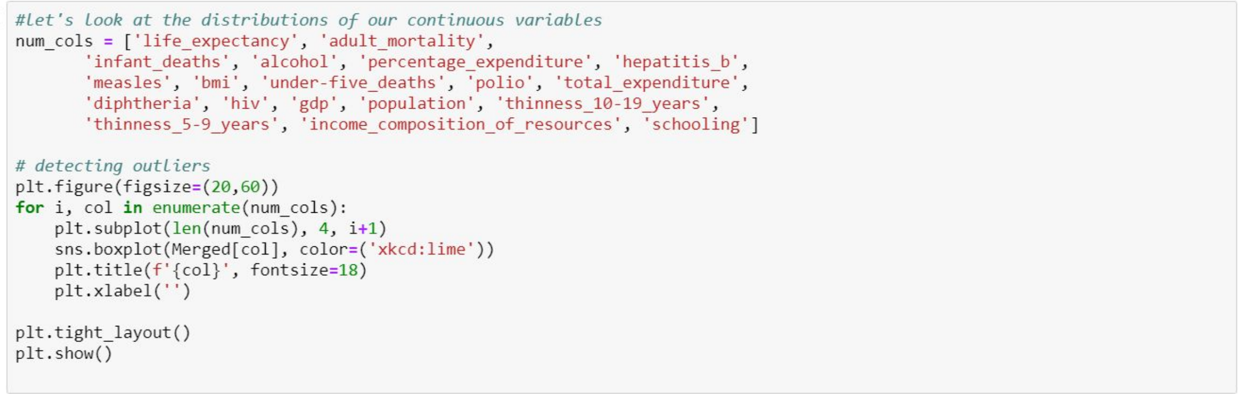
```
#Calculating null values on all columns.
#Here, we get that all the missing values have been removed and replaced by interpolation method
Merged.isna().sum()
```

```
country                            0
year                               0
status                             0
life_expectancy                    0
adult_mortality                    0
infant_deaths                      0
alcohol                            0
percentage_expenditure             0
hepatitis_b                        0
measles                            0
bmi                                0
under-five_deaths                  0
polio                              0
total_expenditure                  0
diphtheria                         0
hiv                                0
gdp                                0
population                         0
thinness_10-19_years               0
thinness_5-9_years                 0
income_composition_of_resources    0
schooling                          0
country_code                       0
region                             0
income_group                       0
life_expectancy_1                  0
dtype: int64
```

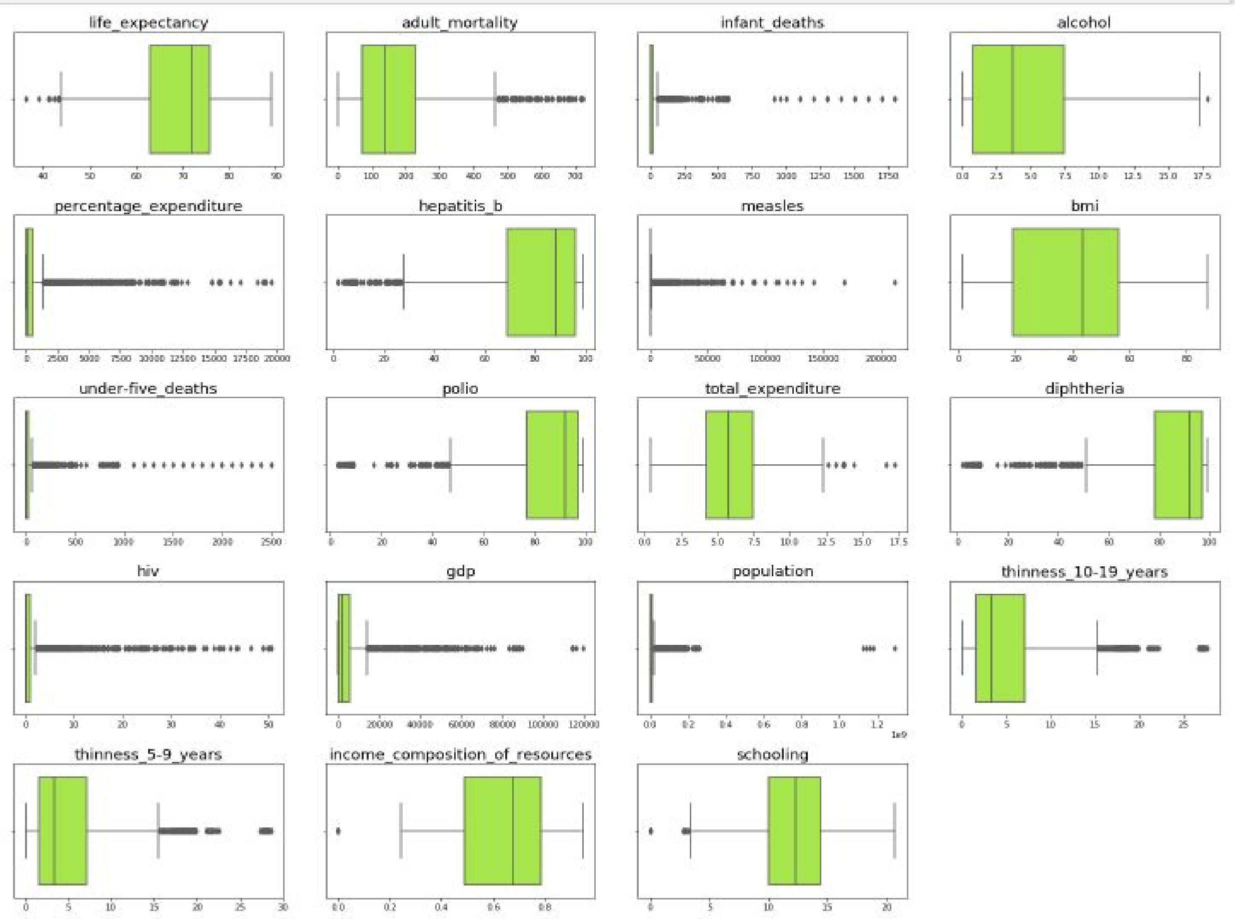
Data Analysis

We started our initial analysis by performing exploratory data analysis.

To begin, we wanted to see the spread data for each column. This would allow us to view the skewness of the data and also identify if there are any outliers. We used a box plot to view the median, first quartile, third quartile and the outliers.

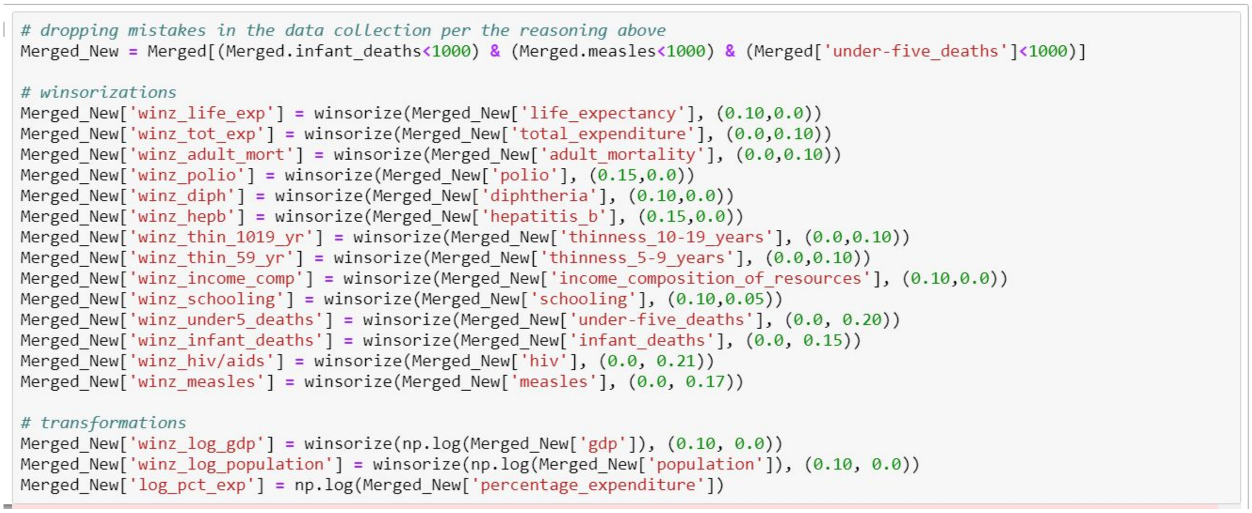


```
#let's look at the distributions of our continuous variables
num_cols = ['life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv', 'gdp', 'population', 'thinness_10-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling']

# detecting outliers
plt.figure(figsize=(20,60))
for i, col in enumerate(num_cols):
    plt.subplot(len(num_cols), 4, i+1)
    sns.boxplot(Merged[col], color=('xkcd:lime'))
    plt.title(f'{col}', fontsize=18)
    plt.xlabel('')

plt.tight_layout()
plt.show()
```



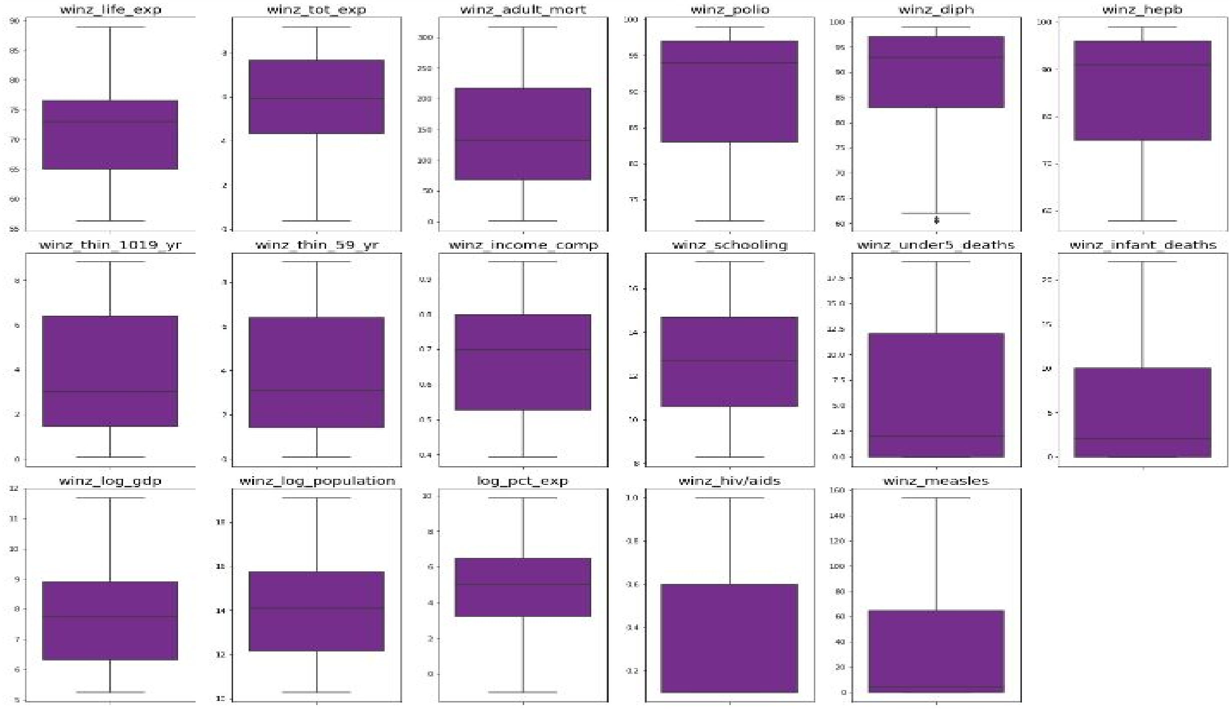
It was important to handle the outliers, as not handling would result in biased conclusions. We need to handle them to have more reliable analysis. We have 2 ways to handle outliers:

- **Dropping** : Dropping will result in losing data. Hence, it will restrict us from gaining powerful and correct insights.
- **Winsorization** : It limits the values of the outliers. We can cap the outliers with the value of specified percentile. In that way, we can limit outliers' affect on our analysis.



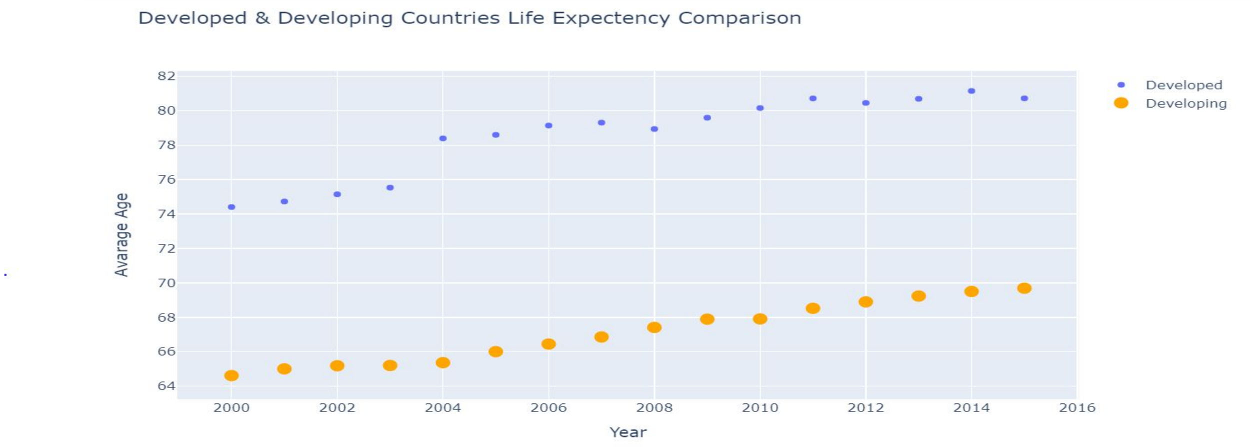
```
# dropping mistakes in the data collection per the reasoning above
Merged_New = Merged[(Merged.infant_deaths<1000) & (Merged.measles<1000) & (Merged['under-five_deaths']<1000)]

# winsorizations
Merged_New['winz_life_exp'] = winsorize(Merged_New['life_expectancy'], (0.10,0.0))
Merged_New['winz_tot_exp'] = winsorize(Merged_New['total_expenditure'], (0.0,0.10))
Merged_New['winz_adult_mort'] = winsorize(Merged_New['adult_mortality'], (0.0,0.10))
Merged_New['winz_polio'] = winsorize(Merged_New['polio'], (0.15,0.0))
Merged_New['winz_diph'] = winsorize(Merged_New['diphtheria'], (0.10,0.0))
Merged_New['winz_hepb'] = winsorize(Merged_New['hepatitis_b'], (0.15,0.0))
Merged_New['winz_thin_1019_yr'] = winsorize(Merged_New['thinness_10-19_years'], (0.0,0.10))
Merged_New['winz_thin_59_yr'] = winsorize(Merged_New['thinness_5-9_years'], (0.0,0.10))
Merged_New['winz_income_comp'] = winsorize(Merged_New['income_composition_of_resources'], (0.10,0.0))
Merged_New['winz_schooling'] = winsorize(Merged_New['schooling'], (0.10,0.05))
Merged_New['winz_under5_deaths'] = winsorize(Merged_New['under-five_deaths'], (0.0, 0.20))
Merged_New['winz_infant_deaths'] = winsorize(Merged_New['infant_deaths'], (0.0, 0.15))
Merged_New['winz_hiv/aids'] = winsorize(Merged_New['hiv'], (0.0, 0.21))
Merged_New['winz_measles'] = winsorize(Merged_New['measles'], (0.0, 0.17))

# transformations
Merged_New['winz_log_gdp'] = winsorize(np.log(Merged_New['gdp']), (0.10, 0.0))
Merged_New['winz_log_population'] = winsorize(np.log(Merged_New['population']), (0.10, 0.0))
Merged_New['log_pct_exp'] = np.log(Merged_New['percentage_expenditure'])
```

We then again plotted the box plot to view the data graphically.

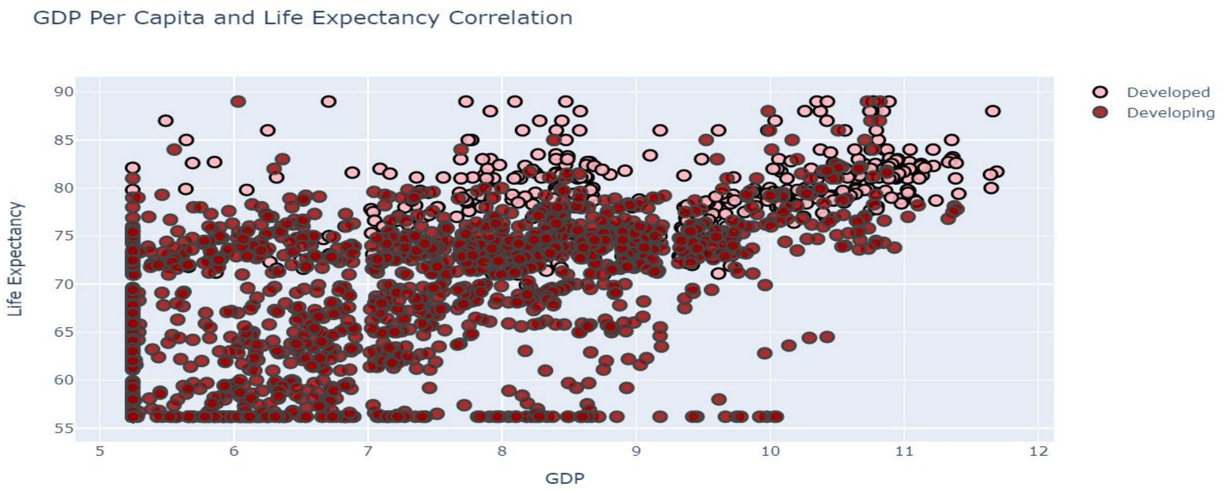


We viewed the data by plotting graphs against the life expectancy.

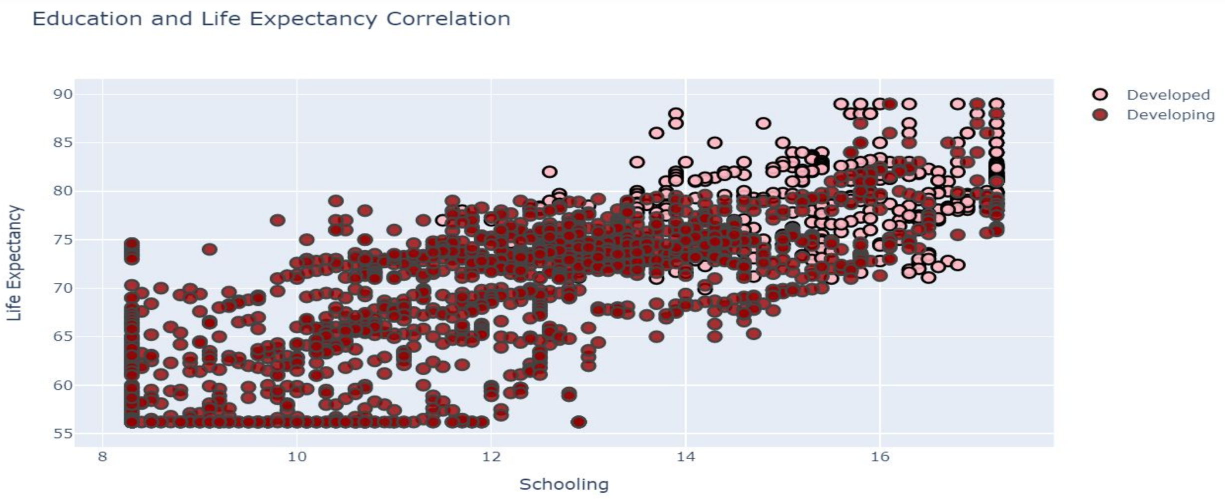
- Developed & Developing Countries Life Expectancy Comparison



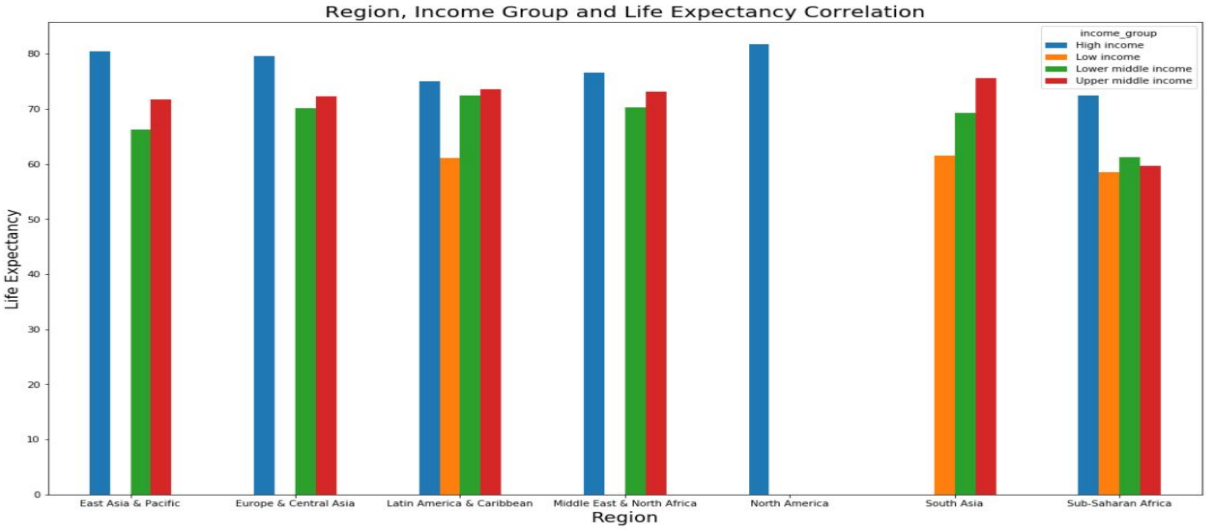
- GDP Per Capita and Life Expectancy Correlation



- Education and Life Expectancy Correlation

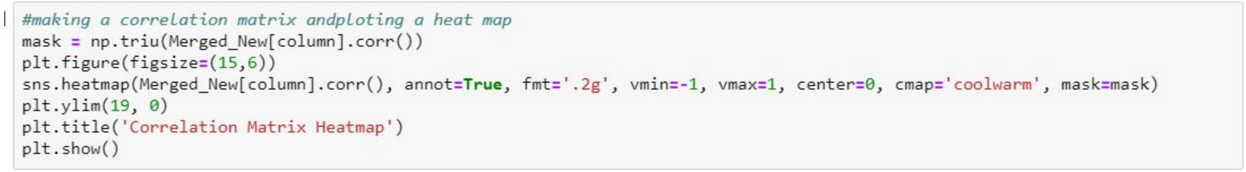


- Region, Income Group and Life Expectancy Correlation

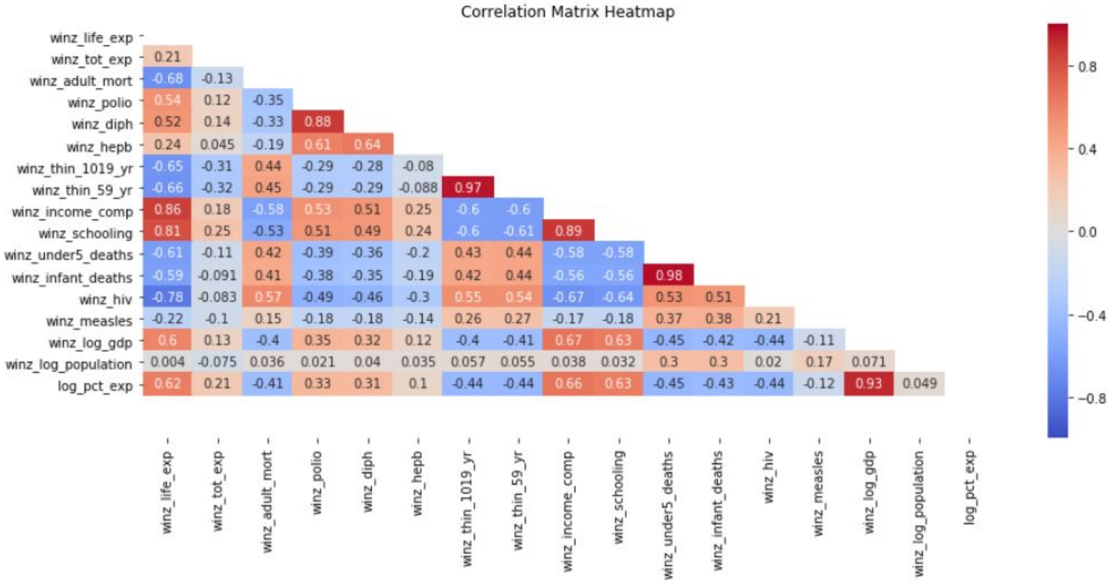


Questions answered through our analysis:

1. **What are the predicting variables affecting life expectancy?**
   Plotting a correlation matrix would help us understand which variables positively and negatively have a high or low correlation to life expectancy.



```
#making a correlation matrix andploting a heat map
mask = np.triu(Merged_New[column].corr())
plt.figure(figsize=(15,6))
sns.heatmap(Merged_New[column].corr(), annot=True, fmt='.2g', vmin=-1, vmax=1, center=0, cmap='coolwarm', mask=mask)
plt.ylim(19, 0)
plt.title('Correlation Matrix Heatmap')
plt.show()
```

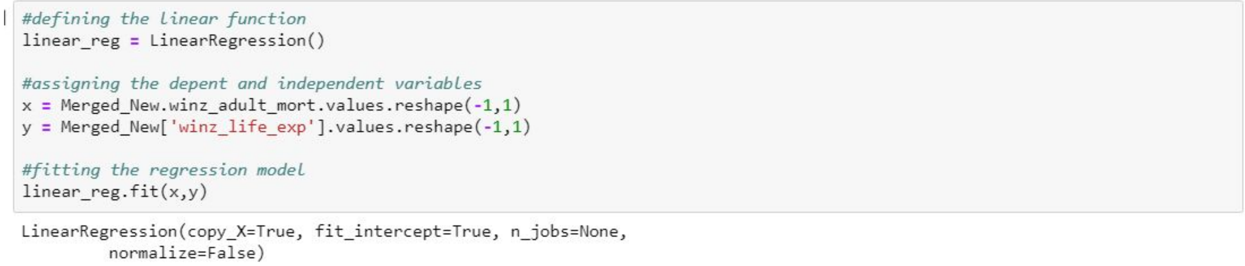


The above heatmap displays a number of important correlations between variables. Some general takeaways from the graphic above:

- Life Expectancy (target variable) appears to be relatively highly correlated (negatively or positively) with:
- Adult Mortality (negative)
- HIV/AIDS (negative)
- Income Composition of Resources (positive)
- Schooling (positive)
- Life expectancy (target variable) is extremely lowly correlated to population (nearly no correlation at all)
- Infant deaths and Under Five deaths are extremely highly correlated
- Percentage Expenditure and GDP are relatively highly correlated
- Hepatitis B vaccine rate is relatively positively correlated with Polio and Diphtheria vaccine rates
- Polio vaccine rate and Diphtheria vaccine rate are very positively correlated
- HIV/AIDS is relatively negatively correlated with Income Composition of Resources
- Thinness of 5-9 Year olds rate and Thinness of 10-15 Year olds rate is extremely highly correlated
- Income Composition of Resources and Schooling are very highly correlated

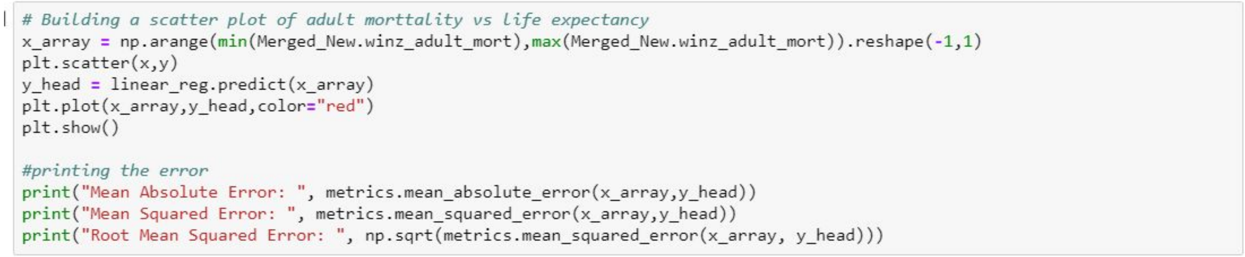
2. **What is the impact of Immunization coverage on life expectancy?**
   From the correlation matrix we understood that there is a very low correlation between the immunization coverages - Polio, Hepatitis B and Diphtheria.

3. **How does Adult mortality rates affect life expectancy?**
   We created a linear regression model to know how adult mortality affects life expectancy.

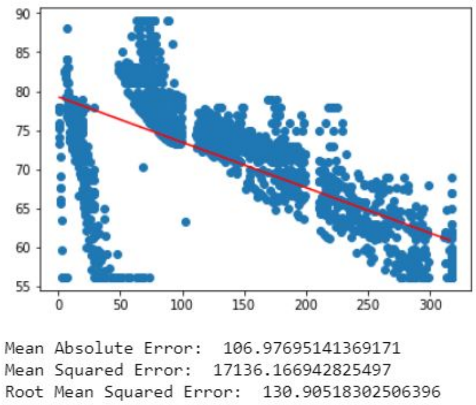


```
#defining the linear function
linear_reg = LinearRegression()

#assigning the depent and independent variables
x = Merged_New.winz_adult_mort.values.reshape(-1,1)
y = Merged_New['winz_life_exp'].values.reshape(-1,1)

#fitting the regression model
linear_reg.fit(x,y)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```



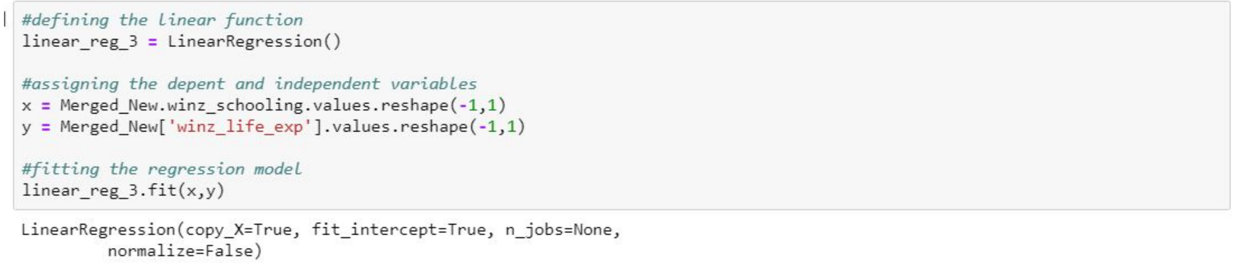
```
# Building a scatter plot of adult morttality vs life expectancy
x_array = np.arange(min(Merged_New.winz_adult_mort),max(Merged_New.winz_adult_mort)).reshape(-1,1)
plt.scatter(x,y)
y_head = linear_reg.predict(x_array)
plt.plot(x_array,y_head,color="red")
plt.show()

#printing the error
print("Mean Absolute Error: ", metrics.mean_absolute_error(x_array,y_head))
print("Mean Squared Error: ", metrics.mean_squared_error(x_array,y_head))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(x_array, y_head)))
```



```
Mean Absolute Error:  106.97695141369171
Mean Squared Error:  17136.166942825497
Root Mean Squared Error:  130.90518302506396
```

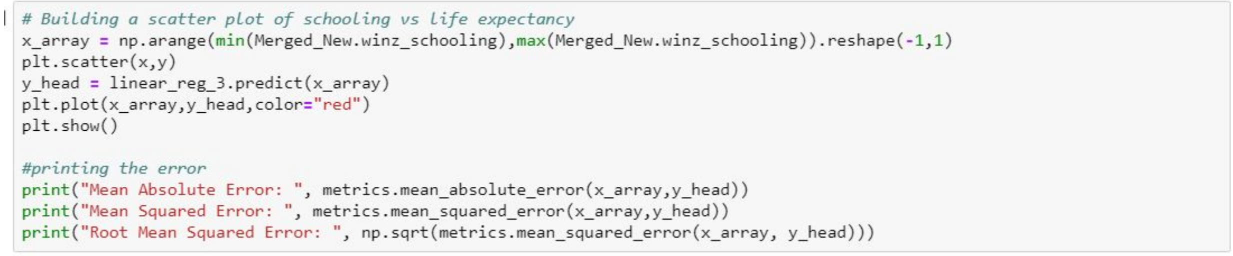
The linear regression scatter plot shows that adult mortality is directly negatively correlated to life expectancy. Lower the adult mortality rate will increase the life expectancy.

4. **What is the impact of schooling on the lifespan of humans?**
   We created a linear regression model to know how the number of schooling years affects life expectancy.

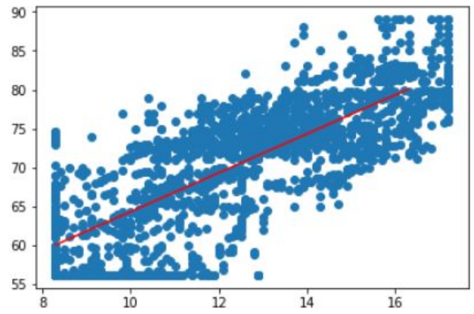


```
#defining the linear function
linear_reg_3 = LinearRegression()

#assigning the depent and independent variables
x = Merged_New.winz_schooling.values.reshape(-1,1)
y = Merged_New['winz_life_exp'].values.reshape(-1,1)

#fitting the regression model
linear_reg_3.fit(x,y)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```



```
# Building a scatter plot of schooling vs life expectancy
x_array = np.arange(min(Merged_New.winz_schooling),max(Merged_New.winz_schooling)).reshape(-1,1)
plt.scatter(x,y)
y_head = linear_reg_3.predict(x_array)
plt.plot(x_array,y_head,color="red")
plt.show()

#printing the error
print("Mean Absolute Error: ", metrics.mean_absolute_error(x_array,y_head))
print("Mean Squared Error: ", metrics.mean_squared_error(x_array,y_head))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(x_array, y_head)))
```



```
Mean Absolute Error:  57.772554200920894
Mean Squared Error:  3352.6737945326668
Root Mean Squared Error:  57.90227797360538
```

The plot above explains that as the number of years of schooling increases the life expectancy of countries increases.

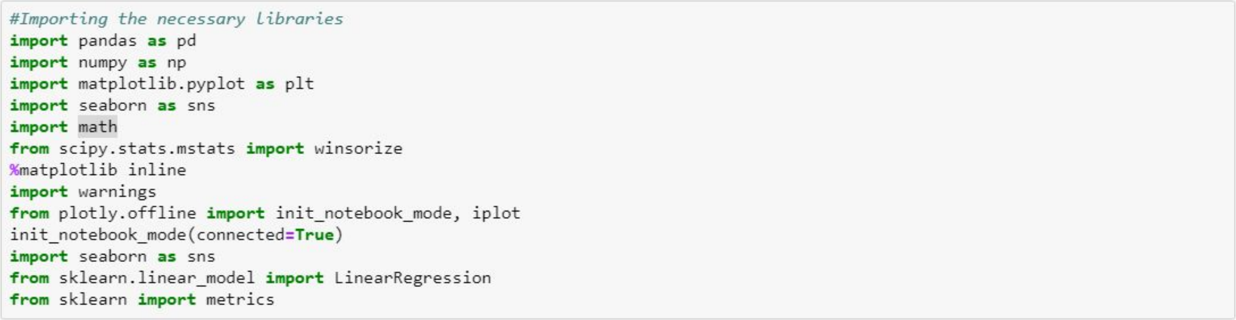
Python Program Description and Output

Description of the program:

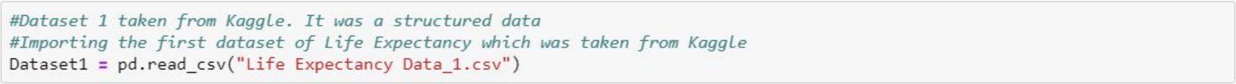
1. **Load the dataset**

   Installed packages for the data analysis.

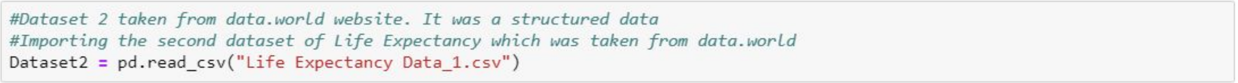
   Read the csv file in the panda dataframe.



```
#Importing the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import math
from scipy.stats.mstats import winsorize
%matplotlib inline
import warnings
from plotly.offline import init_notebook_mode, iplot
init_notebook_mode(connected=True)
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

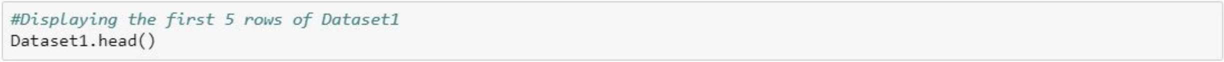


```
#Dataset 1 taken from Kaggle. It was a structured data
#Importing the first dataset of Life Expectancy which was taken from Kaggle
Dataset1 = pd.read_csv("Life Expectancy Data_1.csv")
```

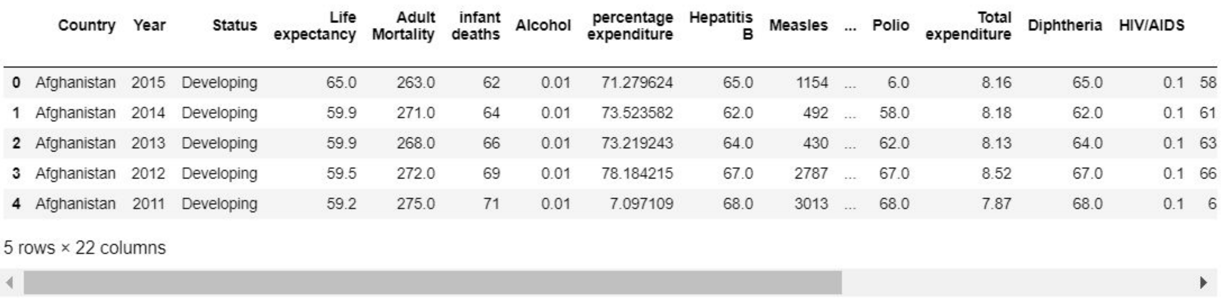


```
#Dataset 2 taken from data.world website. It was a structured data
#Importing the second dataset of Life Expectancy which was taken from data.world
Dataset2 = pd.read_csv("Life Expectancy Data_1.csv")
```

2. **View both the datasets**

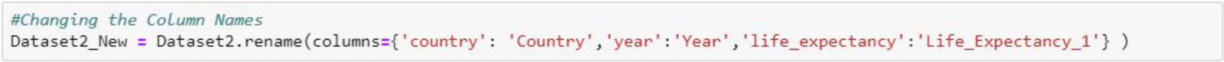


```
#Displaying the first 5 rows of Dataset1
Dataset1.head()
```

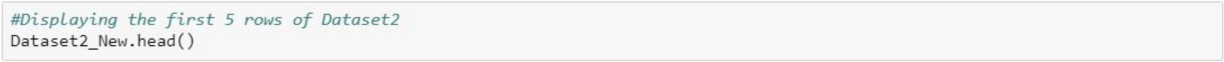


	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	Polio	Total expenditure	Diphtheria	HIV/AIDS	
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	6.0	8.16	65.0	0.1	58
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	58.0	8.18	62.0	0.1	61
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	62.0	8.13	64.0	0.1	63
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	67.0	8.52	67.0	0.1	66
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	68.0	7.87	68.0	0.1	6

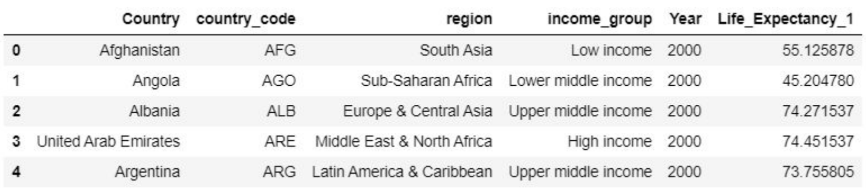
5 rows × 22 columns



```
#Changing the Column Names
Dataset2_New = Dataset2.rename(columns={'country': 'Country','year':'Year','life_expectancy':'Life_Expectancy_1'} )
```

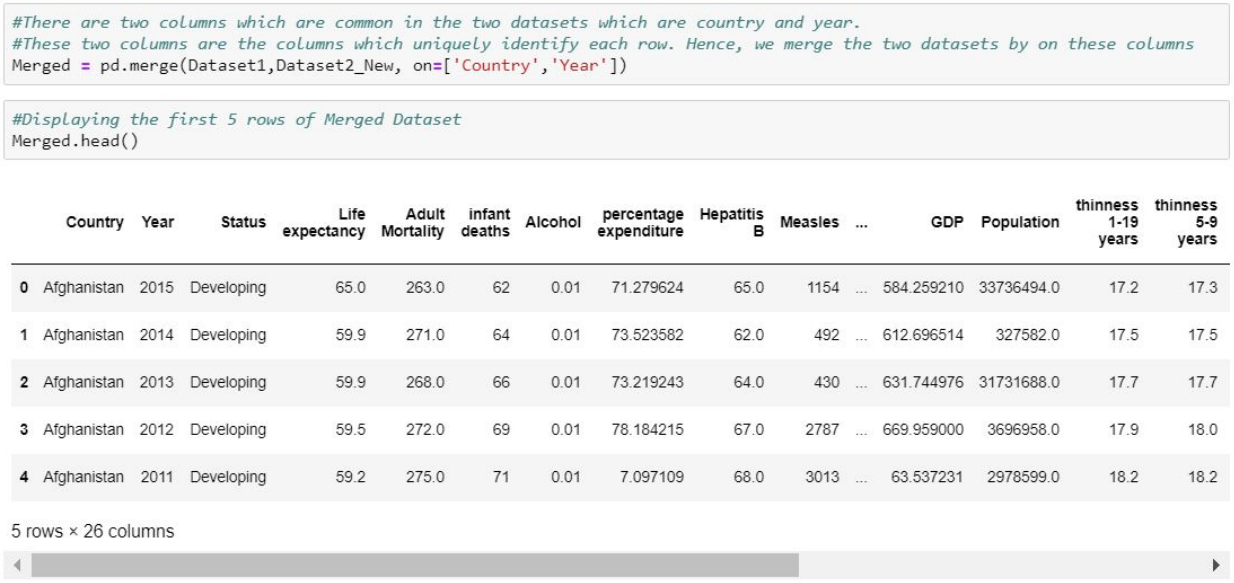


```
#Displaying the first 5 rows of Dataset2
Dataset2_New.head()
```



	Country	country_code	region	income_group	Year	Life_Expectancy_1
0	Afghanistan	AFG	South Asia	Low income	2000	55.125878
1	Angola	AGO	Sub-Saharan Africa	Lower middle income	2000	45.204780
2	Albania	ALB	Europe & Central Asia	Upper middle income	2000	74.271537
3	United Arab Emirates	ARE	Middle East & North Africa	High income	2000	74.451537
4	Argentina	ARG	Latin America & Caribbean	Upper middle income	2000	73.755805

3. Merge the two datasets on Country and Year



```
#There are two columns which are common in the two datasets which are country and year.
#These two columns are the columns which uniquely identify each row. Hence, we merge the two datasets by on these columns
Merged = pd.merge(Dataset1,Dataset2_New, on=['Country','Year'])
```

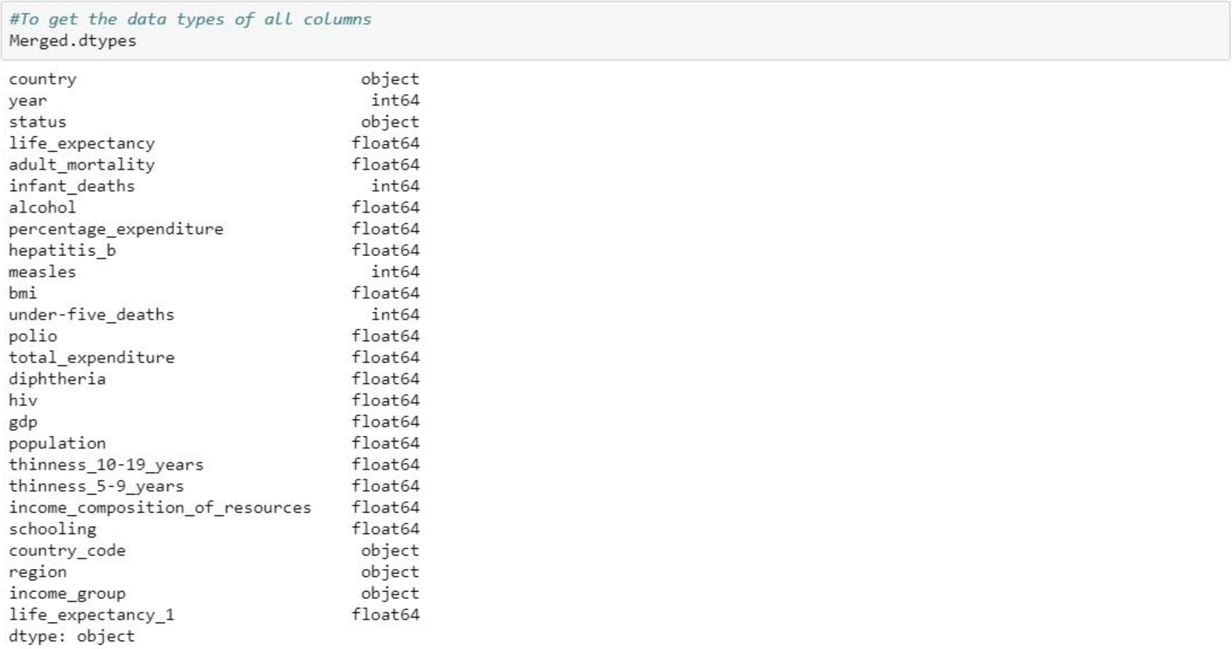
```
#Displaying the first 5 rows of Merged Dataset
Merged.head()
```

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	GDP	Population	thinness 1-19 years	thinness 5-9 years
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	584.259210	33736494.0	17.2	17.3
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	612.696514	327582.0	17.5	17.5
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	631.744976	31731688.0	17.7	17.7
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	669.959000	3696958.0	17.9	18.0
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	63.537231	2978599.0	18.2	18.2

5 rows × 26 columns

4. Statistical Description

We have used the functions dtypes to view the data types and statistical information about the dataset.

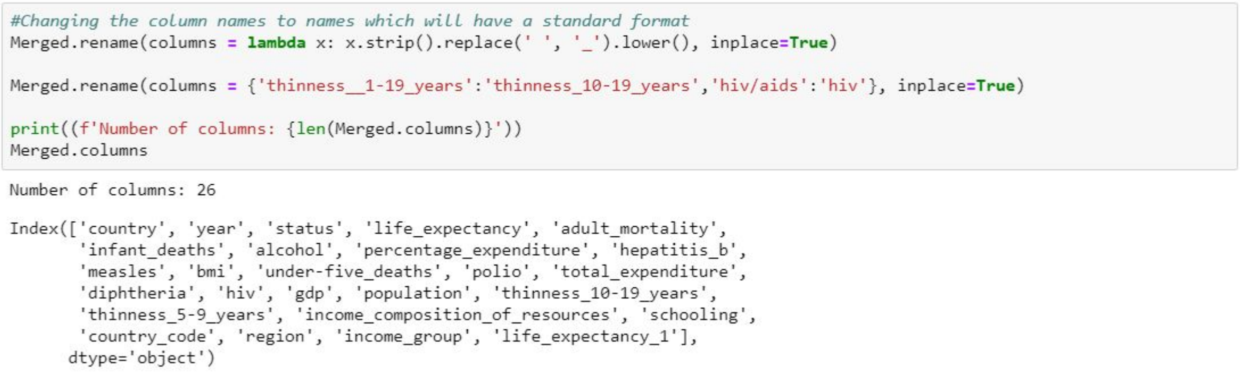


```
#To get the data types of all columns
Merged.dtypes
```

```
country                             object
year                                 int64
status                              object
life_expectancy                    float64
adult_mortality                    float64
infant_deaths                        int64
alcohol                            float64
percentage_expenditure             float64
hepatitis_b                        float64
measles                              int64
bmi                                float64
under-five_deaths                    int64
polio                              float64
total_expenditure                  float64
diphtheria                         float64
hiv                                float64
gdp                                float64
population                         float64
thinness_10-19_years               float64
thinness_5-9_years                 float64
income_composition_of_resources    float64
schooling                          float64
country_code                        object
region                              object
income_group                        object
life_expectancy_1                  float64
dtype: object
```

5. Data Cleaning

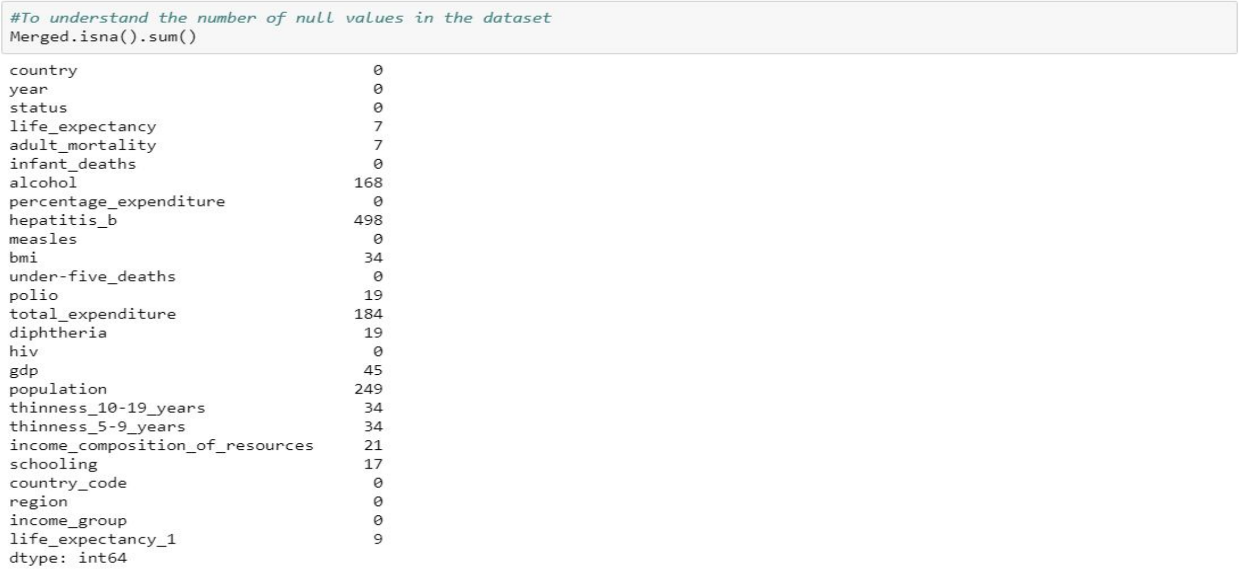
Remaned all the columns, removing blanks and special characters.



```
#Changing the column names to names which will have a standard format
Merged.rename(columns = lambda x: x.strip().replace(' ', '_').lower(), inplace=True)

Merged.rename(columns = {'thinness__1-19_years':'thinness_10-19_years','hiv/aids':'hiv'}, inplace=True)

print((f'Number of columns: {len(Merged.columns)}'))
Merged.columns
```

```
Number of columns: 26

Index(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv', 'gdp', 'population', 'thinness_10-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling',
       'country_code', 'region', 'income_group', 'life_expectancy_1'],
      dtype='object')
```

With the function isna, we could view which columns have NA values.

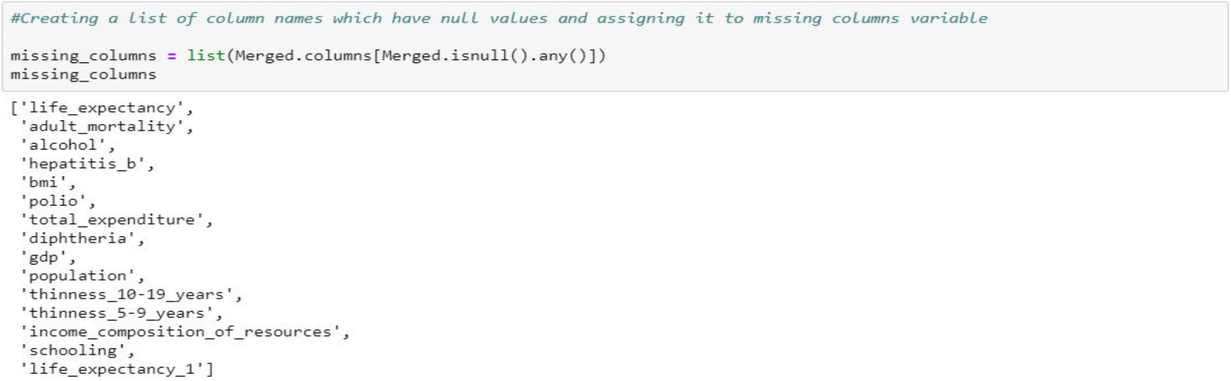


```
#To understand the number of null values in the dataset
Merged.isna().sum()
```

```
country                              0
year                                 0
status                               0
life_expectancy                      7
adult_mortality                      7
infant_deaths                        0
alcohol                            168
percentage_expenditure               0
hepatitis_b                        498
measles                              0
bmi                                 34
under-five_deaths                    0
polio                               19
total_expenditure                  184
diphtheria                          19
hiv                                  0
gdp                                 45
population                         249
thinness_10-19_years                34
thinness_5-9_years                  34
income_composition_of_resources     21
schooling                           17
country_code                         0
region                               0
income_group                         0
life_expectancy_1                    9
dtype: int64
```

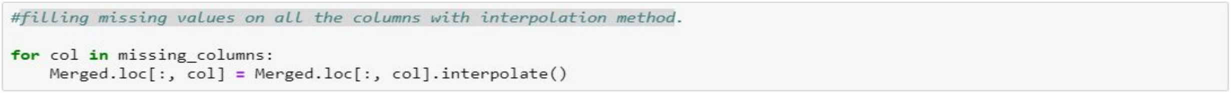
Dealing with missing values

- Fill nulls - Interpolate



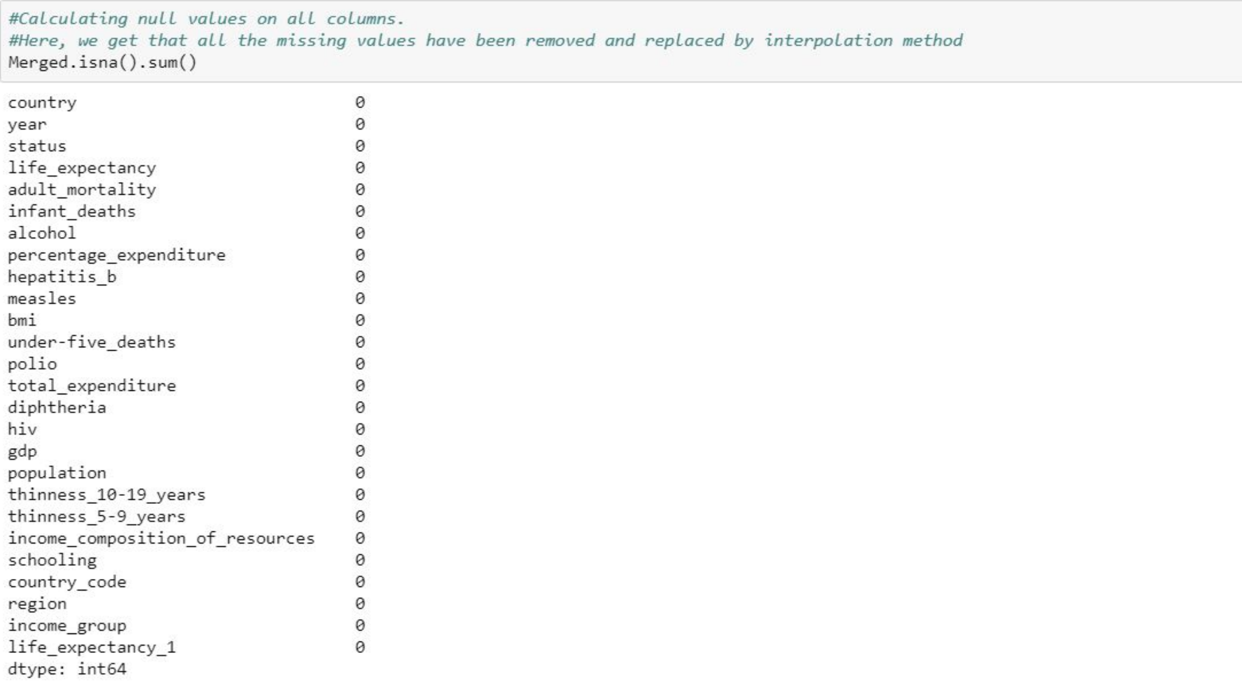
```
#Creating a list of column names which have null values and assigning it to missing columns variable

missing_columns = list(Merged.columns[Merged.isnull().any()])
missing_columns
```

```
['life_expectancy',
 'adult_mortality',
 'alcohol',
 'hepatitis_b',
 'bmi',
 'polio',
 'total_expenditure',
 'diphtheria',
 'gdp',
 'population',
 'thinness_10-19_years',
 'thinness_5-9_years',
 'income_composition_of_resources',
 'schooling',
 'life_expectancy_1']
```



```
#filling missing values on all the columns with interpolation method.

for col in missing_columns:
    Merged.loc[:, col] = Merged.loc[:, col].interpolate()
```

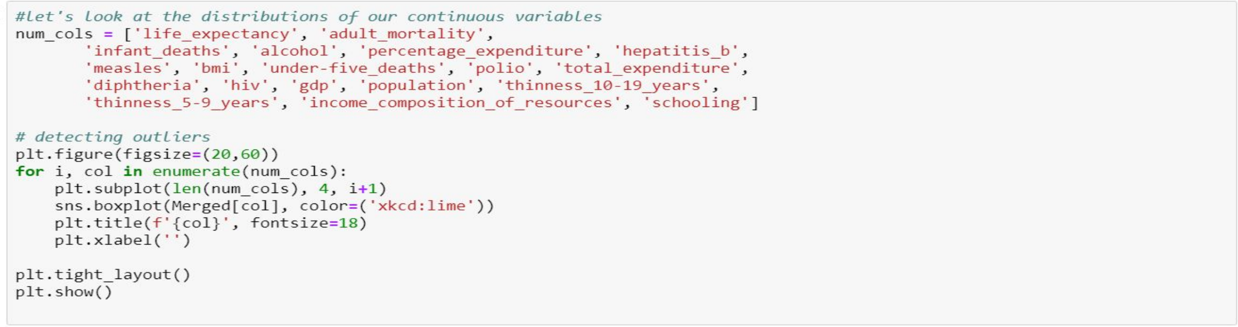
Null values are removed by interpolation method.



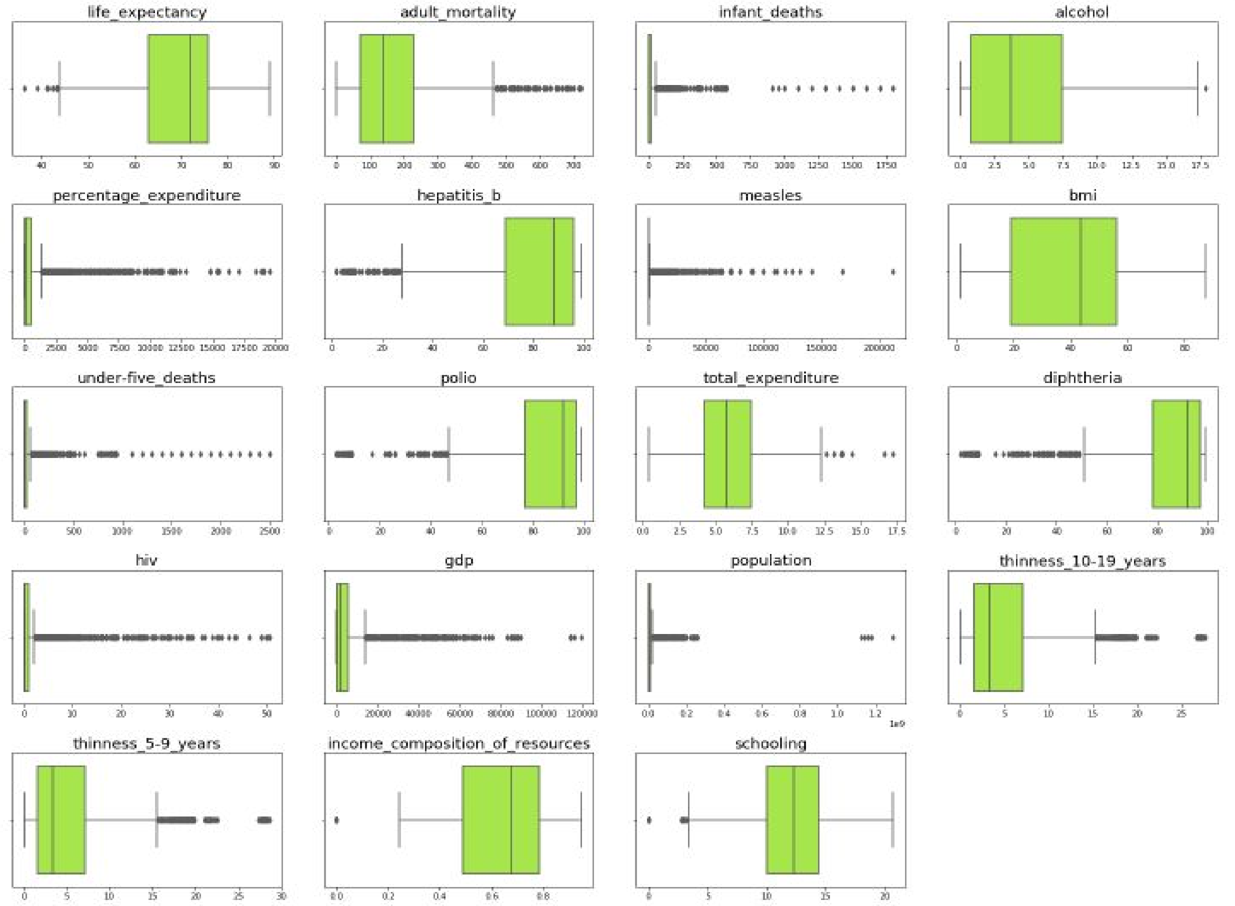
```
#Calculating null values on all columns.
#Here, we get that all the missing values have been removed and replaced by interpolation method
Merged.isna().sum()
```

```
country                          0
year                             0
status                           0
life_expectancy                  0
adult_mortality                  0
infant_deaths                    0
alcohol                          0
percentage_expenditure           0
hepatitis_b                      0
measles                          0
bmi                              0
under-five_deaths                0
polio                            0
total_expenditure                0
diphtheria                       0
hiv                              0
gdp                              0
population                       0
thinness_10-19_years             0
thinness_5-9_years               0
income_composition_of_resources  0
schooling                        0
country_code                     0
region                           0
income_group                     0
life_expectancy_1                0
dtype: int64
```

We wanted to see the spread data for each column. This would allow us to view the skewness of the data and also identify if there are any outliers. We used a box plot to view the median, first quartile, third quartile and the outliers.

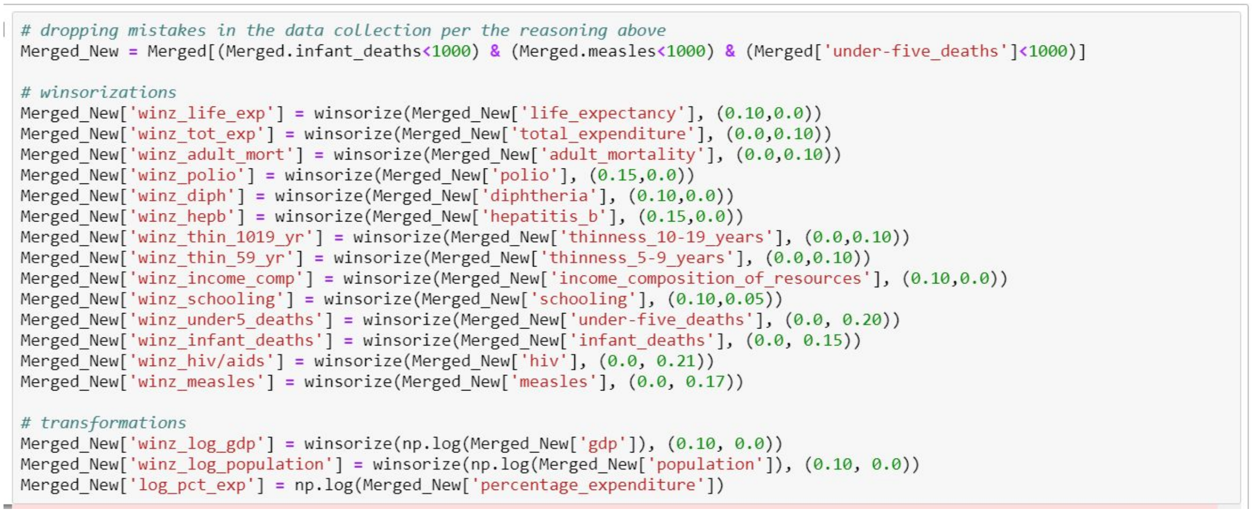


```
#let's look at the distributions of our continuous variables
num_cols = ['life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv', 'gdp', 'population', 'thinness_10-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling']

# detecting outliers
plt.figure(figsize=(20,60))
for i, col in enumerate(num_cols):
    plt.subplot(len(num_cols), 4, i+1)
    sns.boxplot(Merged[col], color=('xkcd:lime'))
    plt.title(f'{col}', fontsize=18)
    plt.xlabel('')

plt.tight_layout()
plt.show()
```



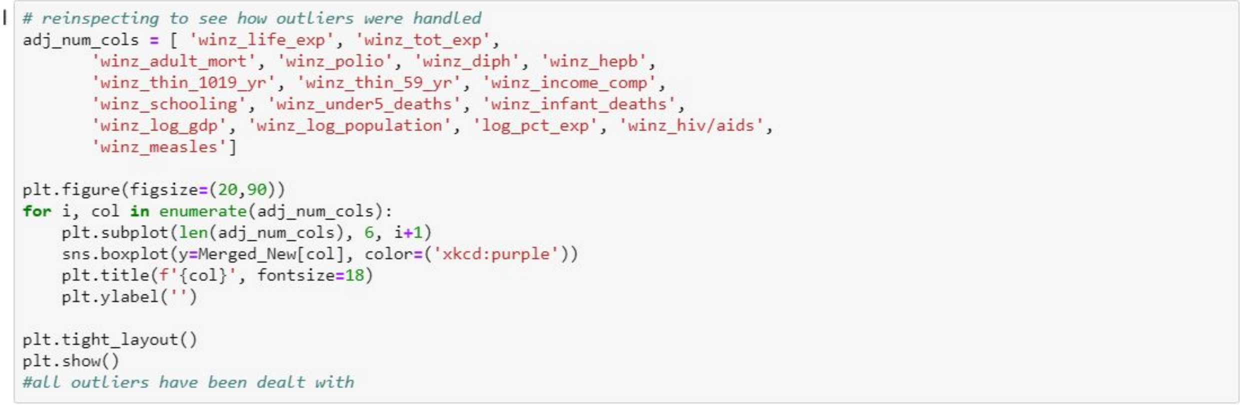
It was important to handle the outliers, as not handling would result in biased conclusions. We need to handle them to have more reliable analysis. We have 2 ways to handle outliers:

- **Dropping** : Dropping will result in losing data. Hence, it will restrict us from gaining powerful and correct insights.
- **Winsorization** : It limits the values of the outliers. We can cap the outliers with the value of specified percentile. In that way, we can limit outliers' affect on our analysis.

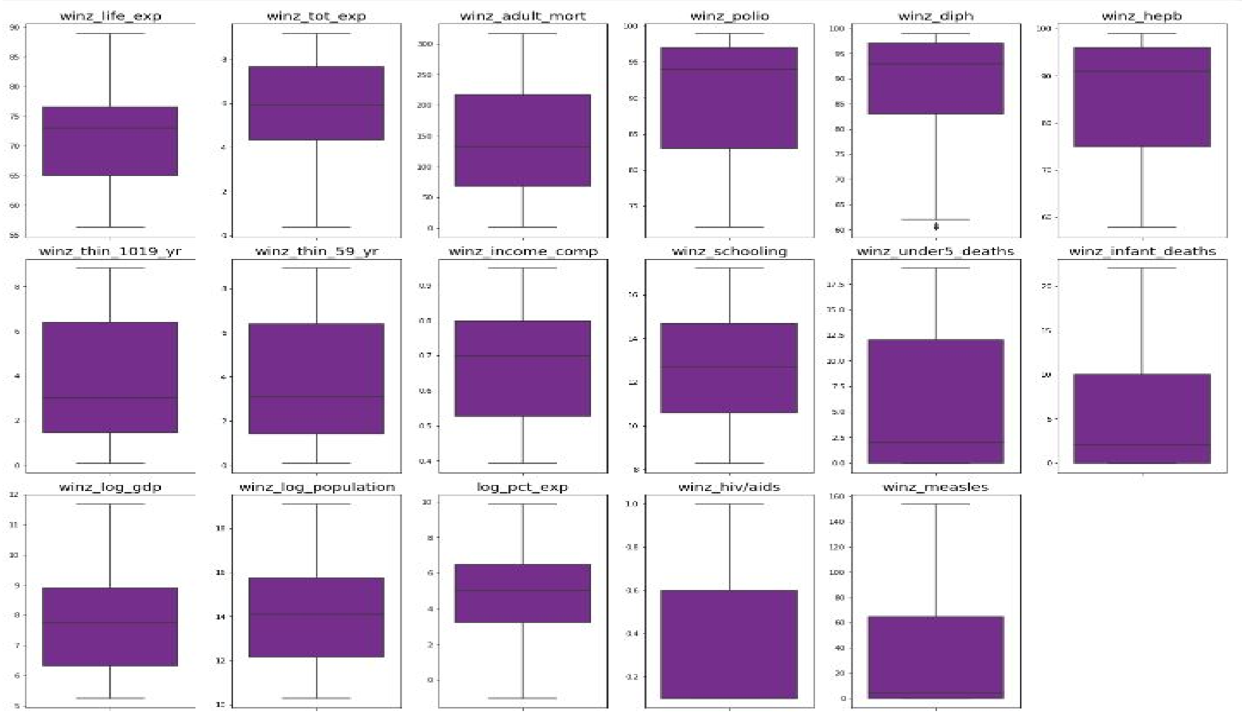


```
# dropping mistakes in the data collection per the reasoning above
Merged_New = Merged[(Merged.infant_deaths<1000) & (Merged.measles<1000) & (Merged['under-five_deaths']<1000)]

# winsorizations
Merged_New['winz_life_exp'] = winsorize(Merged_New['life_expectancy'], (0.10,0.0))
Merged_New['winz_tot_exp'] = winsorize(Merged_New['total_expenditure'], (0.0,0.10))
Merged_New['winz_adult_mort'] = winsorize(Merged_New['adult_mortality'], (0.0,0.10))
Merged_New['winz_polio'] = winsorize(Merged_New['polio'], (0.15,0.0))
Merged_New['winz_diph'] = winsorize(Merged_New['diphtheria'], (0.10,0.0))
Merged_New['winz_hepb'] = winsorize(Merged_New['hepatitis_b'], (0.15,0.0))
Merged_New['winz_thin_1019_yr'] = winsorize(Merged_New['thinness_10-19_years'], (0.0,0.10))
Merged_New['winz_thin_59_yr'] = winsorize(Merged_New['thinness_5-9_years'], (0.0,0.10))
Merged_New['winz_income_comp'] = winsorize(Merged_New['income_composition_of_resources'], (0.10,0.0))
Merged_New['winz_schooling'] = winsorize(Merged_New['schooling'], (0.10,0.05))
Merged_New['winz_under5_deaths'] = winsorize(Merged_New['under-five_deaths'], (0.0, 0.20))
Merged_New['winz_infant_deaths'] = winsorize(Merged_New['infant_deaths'], (0.0, 0.15))
Merged_New['winz_hiv/aids'] = winsorize(Merged_New['hiv'], (0.0, 0.21))
Merged_New['winz_measles'] = winsorize(Merged_New['measles'], (0.0, 0.17))

# transformations
Merged_New['winz_log_gdp'] = winsorize(np.log(Merged_New['gdp']), (0.10, 0.0))
Merged_New['winz_log_population'] = winsorize(np.log(Merged_New['population']), (0.10, 0.0))
Merged_New['log_pct_exp'] = np.log(Merged_New['percentage_expenditure'])
```

We then again plotted the box plot to view the data graphically.



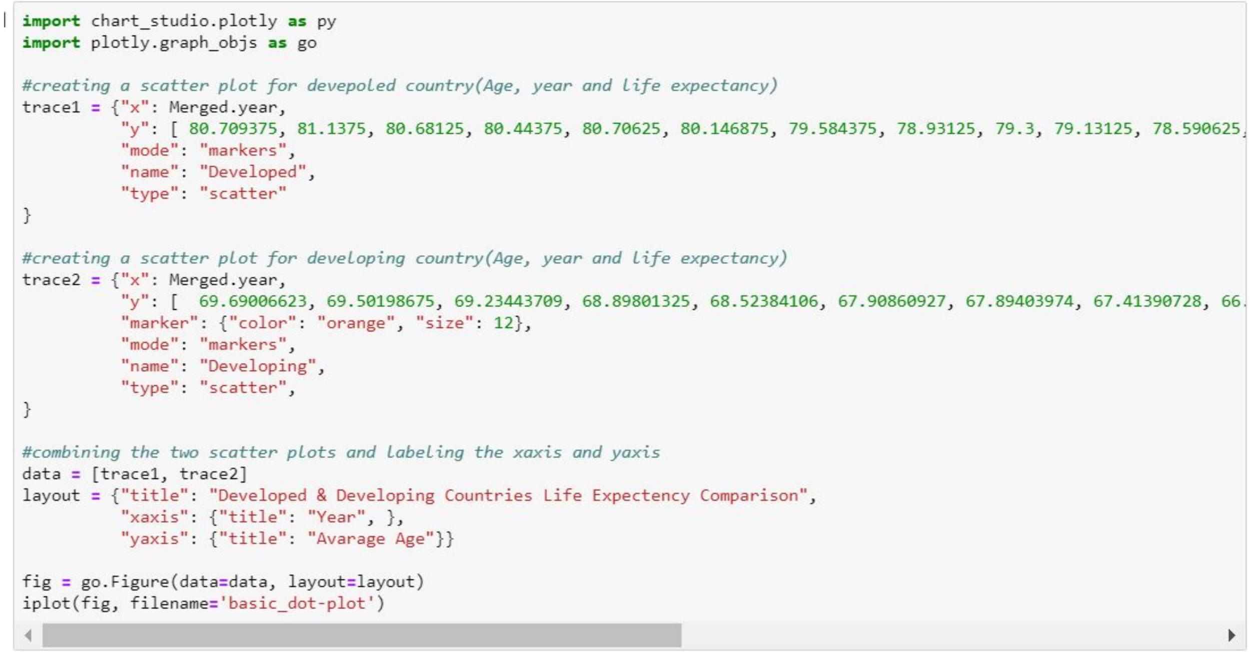
```
# reinspecting to see how outliers were handled
adj_num_cols = [ 'winz_life_exp', 'winz_tot_exp',
       'winz_adult_mort', 'winz_polio', 'winz_diph', 'winz_hepb',
       'winz_thin_1019_yr', 'winz_thin_59_yr', 'winz_income_comp',
       'winz_schooling', 'winz_under5_deaths', 'winz_infant_deaths',
       'winz_log_gdp', 'winz_log_population', 'log_pct_exp', 'winz_hiv/aids',
       'winz_measles']

plt.figure(figsize=(20,90))
for i, col in enumerate(adj_num_cols):
    plt.subplot(len(adj_num_cols), 6, i+1)
    sns.boxplot(y=Merged_New[col], color=('xkcd:purple'))
    plt.title(f'{col}', fontsize=18)
    plt.ylabel('')

plt.tight_layout()
plt.show()
#all outliers have been dealt with
```



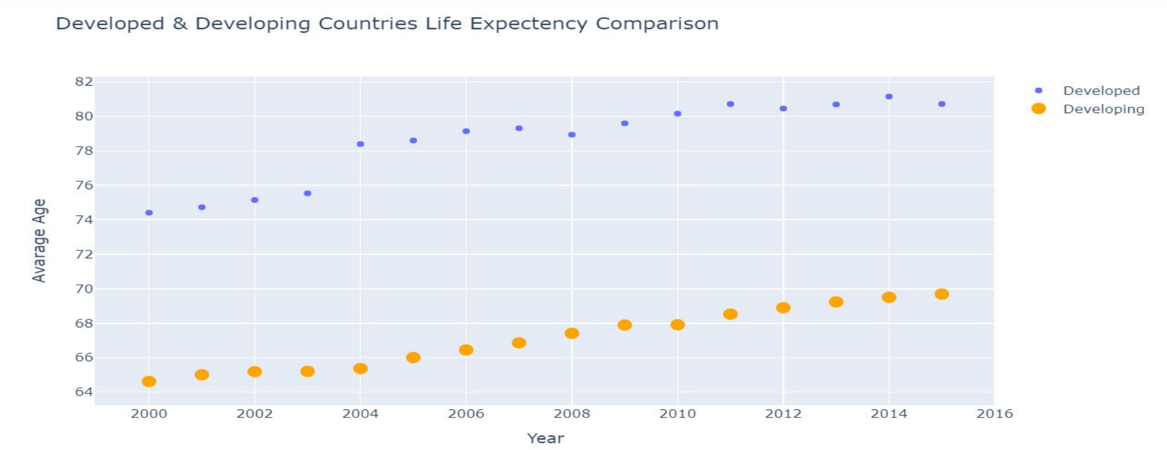
6. Exploratory Data Analysis

We viewed the data by plotting graphs against the life expectancy.

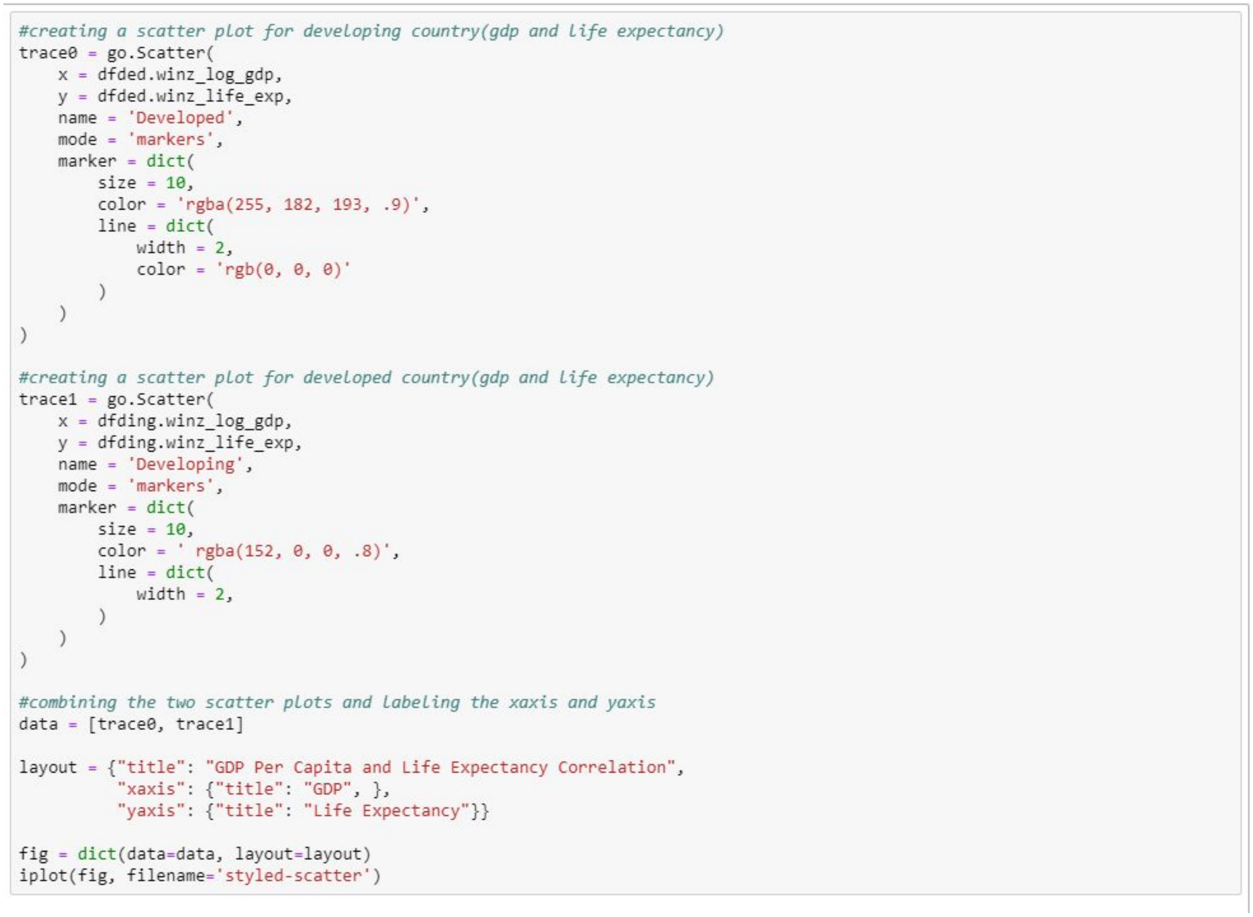
- Developed & Developing Countries Life Expectancy Comparison



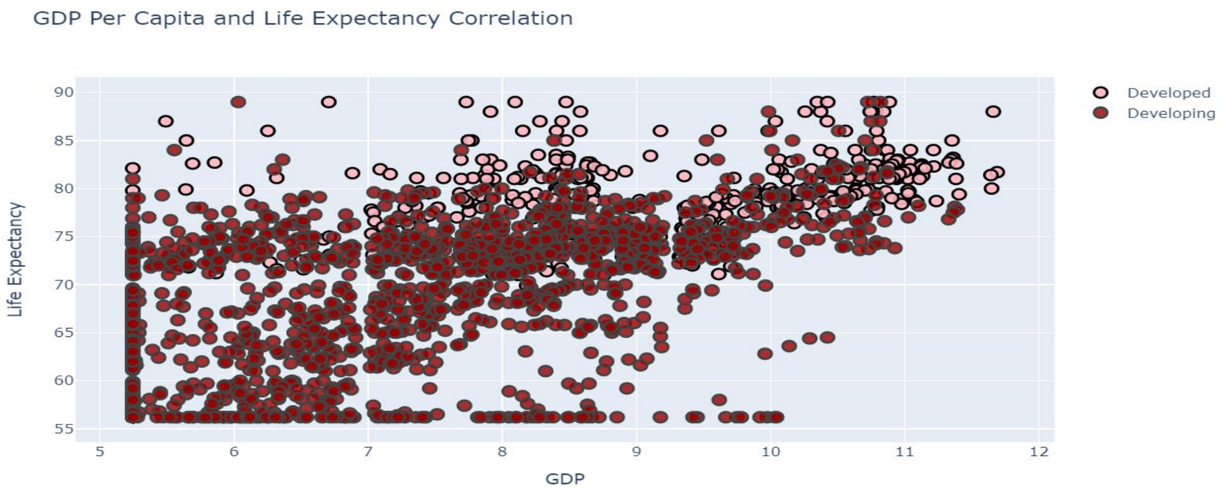
```
import chart_studio.plotly as py
import plotly.graph_objs as go

#creating a scatter plot for devepoled country(Age, year and life expectancy)
trace1 = {"x": Merged.year,
          "y": [ 80.709375, 81.1375, 80.68125, 80.44375, 80.70625, 80.146875, 79.584375, 78.93125, 79.3, 79.13125, 78.590625,
          "mode": "markers",
          "name": "Developed",
          "type": "scatter"
}

#creating a scatter plot for developing country(Age, year and life expectancy)
trace2 = {"x": Merged.year,
          "y": [  69.69006623, 69.50198675, 69.23443709, 68.89801325, 68.52384106, 67.90860927, 67.89403974, 67.41390728, 66.
          "marker": {"color": "orange", "size": 12},
          "mode": "markers",
          "name": "Developing",
          "type": "scatter",
}

#combining the two scatter plots and labeling the xaxis and yaxis
data = [trace1, trace2]
layout = {"title": "Developed & Developing Countries Life Expectency Comparison",
          "xaxis": {"title": "Year", },
          "yaxis": {"title": "Avarage Age"}}

fig = go.Figure(data=data, layout=layout)
iplot(fig, filename='basic_dot-plot')
```



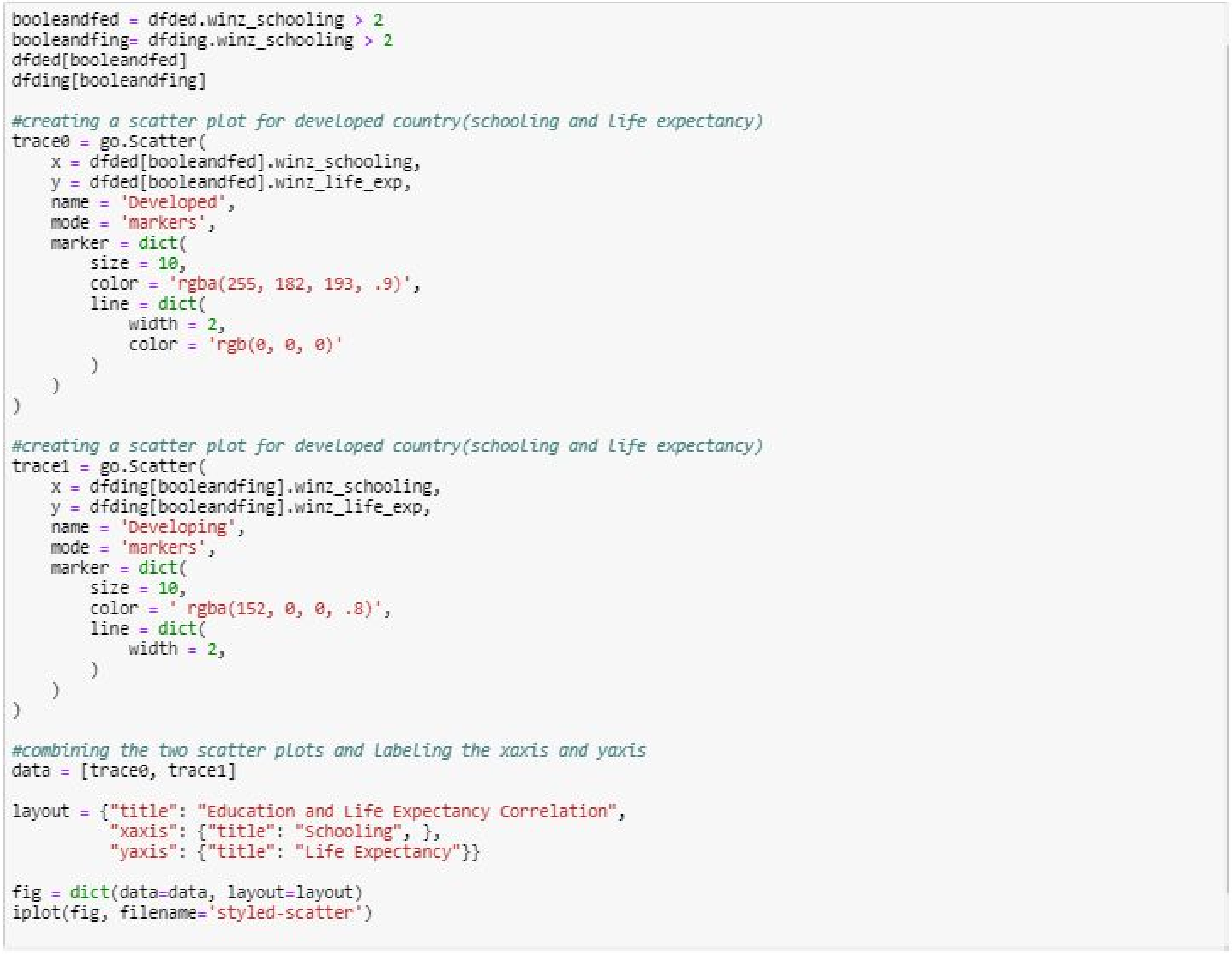
- GDP Per Capita and Life Expectancy Correlation



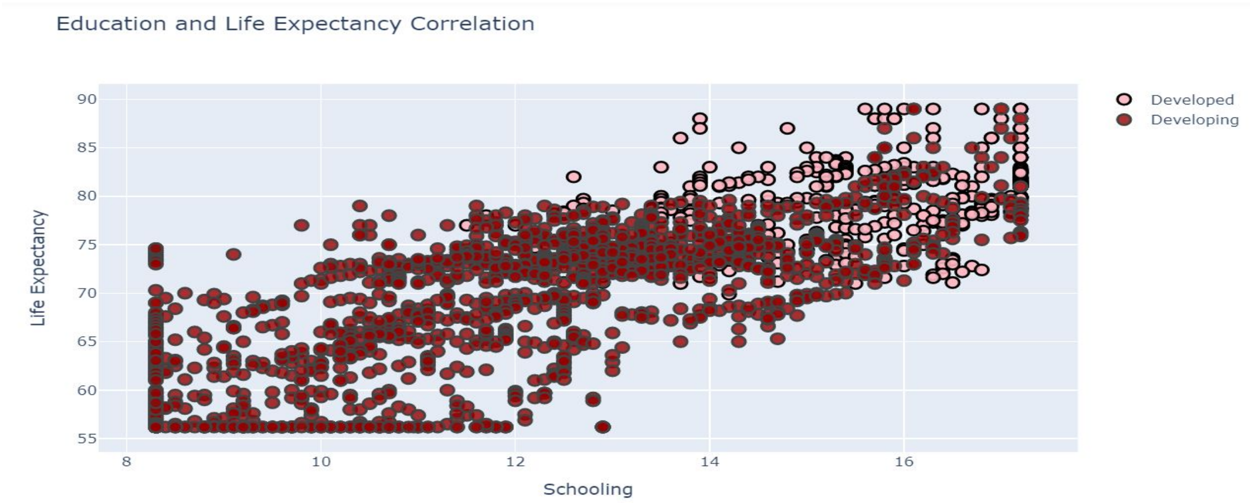
```
#creating a scatter plot for developing country(gdp and life expectancy)
trace0 = go.Scatter(
    x = dfded.winz_log_gdp,
    y = dfded.winz_life_exp,
    name = 'Developed',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(255, 182, 193, .9)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)

#creating a scatter plot for developed country(gdp and life expectancy)
trace1 = go.Scatter(
    x = dfding.winz_log_gdp,
    y = dfding.winz_life_exp,
    name = 'Developing',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = ' rgba(152, 0, 0, .8)',
        line = dict(
            width = 2,
        )
    )
)

#combining the two scatter plots and labeling the xaxis and yaxis
data = [trace0, trace1]

layout = {"title": "GDP Per Capita and Life Expectancy Correlation",
          "xaxis": {"title": "GDP", },
          "yaxis": {"title": "Life Expectancy"}}

fig = dict(data=data, layout=layout)
iplot(fig, filename='styled-scatter')
```



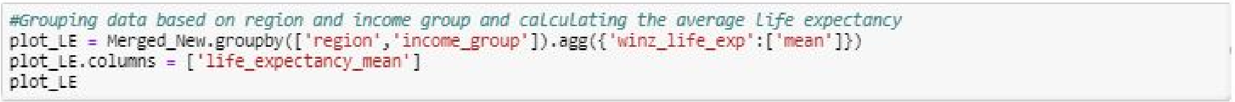
- Education and Life Expectancy Correlation



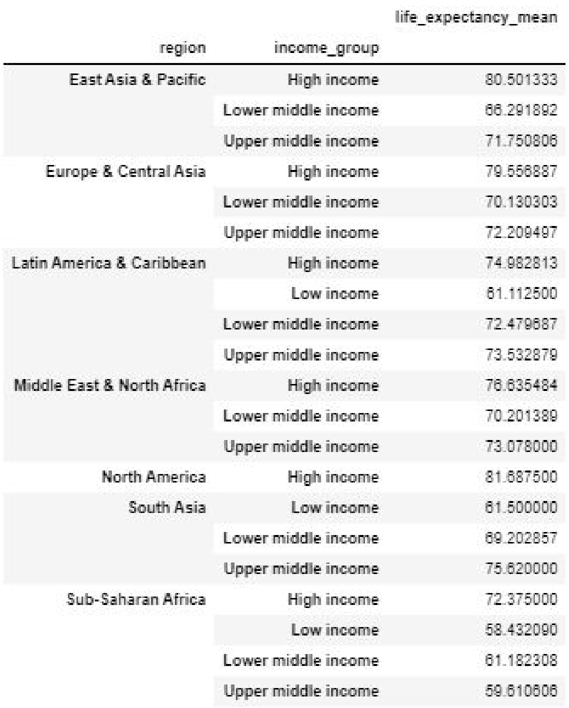
```
booleandfed = dfded.winz_schooling > 2
booleandfing= dfding.winz_schooling > 2
dfded[booleandfed]
dfding[booleandfing]

#creating a scatter plot for developed country(schooling and life expectancy)
trace0 = go.Scatter(
    x = dfded[booleandfed].winz_schooling,
    y = dfded[booleandfed].winz_life_exp,
    name = 'Developed',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = 'rgba(255, 182, 193, .9)',
        line = dict(
            width = 2,
            color = 'rgb(0, 0, 0)'
        )
    )
)

#creating a scatter plot for developed country(schooling and life expectancy)
trace1 = go.Scatter(
    x = dfding[booleandfing].winz_schooling,
    y = dfding[booleandfing].winz_life_exp,
    name = 'Developing',
    mode = 'markers',
    marker = dict(
        size = 10,
        color = ' rgba(152, 0, 0, .8)',
        line = dict(
            width = 2,
        )
    )
)

#combining the two scatter plots and labeling the xaxis and yaxis
data = [trace0, trace1]

layout = {"title": "Education and Life Expectancy Correlation",
          "xaxis": {"title": "Schooling", },
          "yaxis": {"title": "Life Expectancy"}}

fig = dict(data=data, layout=layout)
iplot(fig, filename='styled-scatter')
```



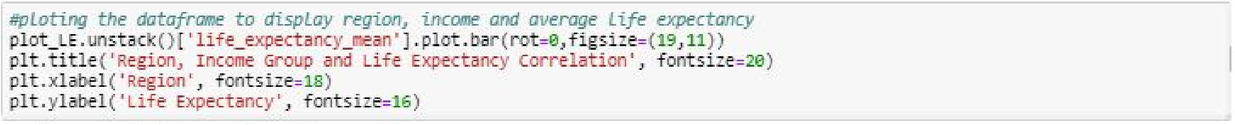
- Region, Income Group and Life Expectancy Correlation



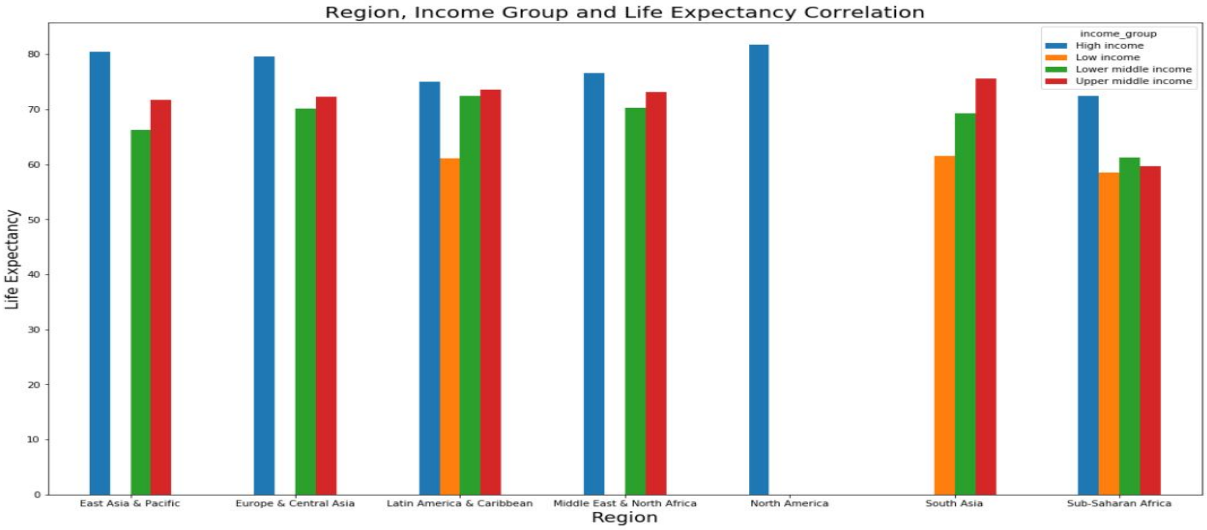
```
#Grouping data based on region and income group and calculating the average life expectancy
plot_LE = Merged_New.groupby(['region','income_group']).agg({'winz_life_exp':['mean']})
plot_LE.columns = ['life_expectancy_mean']
plot_LE
```



region	income_group	life_expectancy_mean
East Asia & Pacific	High income	80.501333
	Lower middle income	66.291892
	Upper middle income	71.750806
Europe & Central Asia	High income	79.556887
	Lower middle income	70.130303
	Upper middle income	72.209497
Latin America & Caribbean	High income	74.982813
	Low income	61.112500
	Lower middle income	72.479687
	Upper middle income	73.532879
Middle East & North Africa	High income	76.635484
	Lower middle income	70.201389
	Upper middle income	73.078000
North America	High income	81.687500
South Asia	Low income	61.500000
	Lower middle income	69.202857
	Upper middle income	75.620000
Sub-Saharan Africa	High income	72.375000
	Low income	58.432090
	Lower middle income	61.182308
	Upper middle income	59.610606



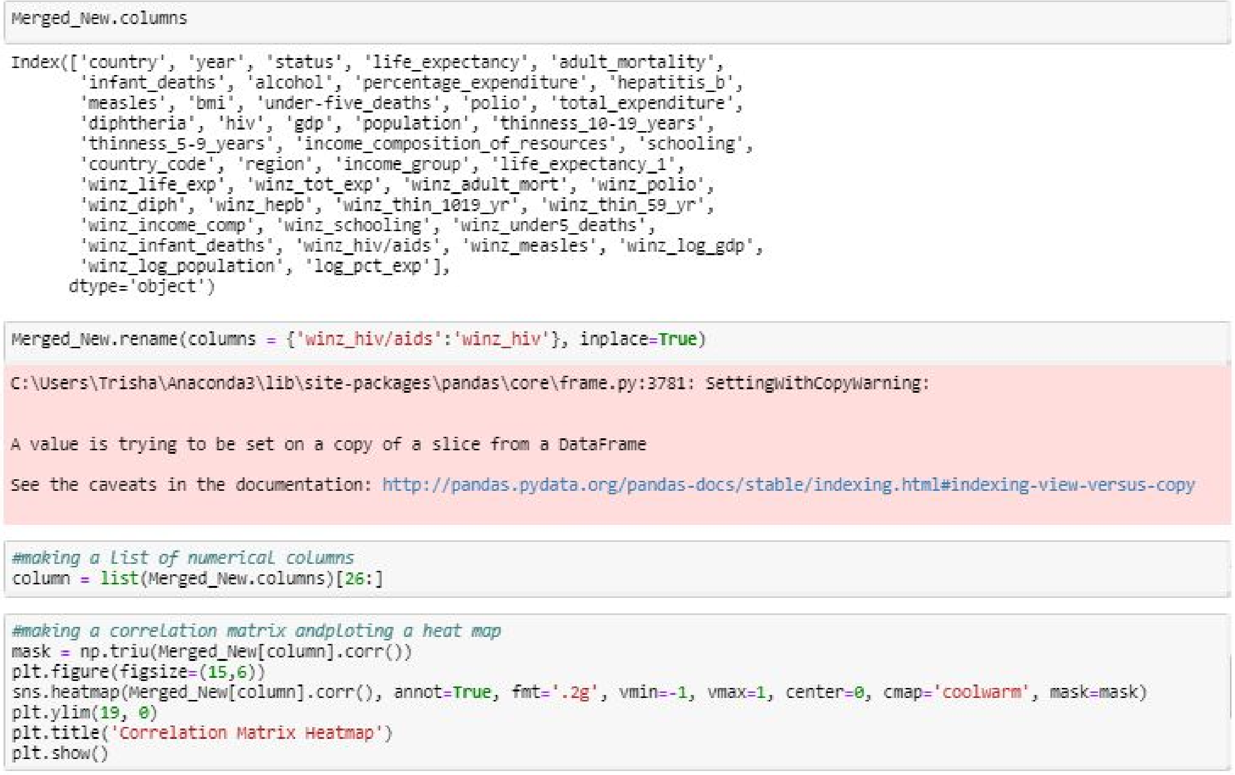
```
#ploting the dataframe to display region, income and average life expectancy
plot_LE.unstack()['life_expectancy_mean'].plot.bar(rot=0,figsize=(19,11))
plt.title('Region, Income Group and Life Expectancy Correlation', fontsize=20)
plt.xlabel('Region', fontsize=18)
plt.ylabel('Life Expectancy', fontsize=16)
```



7. Data Analysis

a. Question 1: What are the predicting variables affecting life expectancy?

Plotting a correlation matrix would help us understand which variables positively and negatively have a high or low correlation to life expectancy.



```
Merged_New.columns
```

```
Index(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_deaths', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under-five_deaths', 'polio', 'total_expenditure',
       'diphtheria', 'hiv', 'gdp', 'population', 'thinness_10-19_years',
       'thinness_5-9_years', 'income_composition_of_resources', 'schooling',
       'country_code', 'region', 'income_group', 'life_expectancy_1',
       'winz_life_exp', 'winz_tot_exp', 'winz_adult_mort', 'winz_polio',
       'winz_diph', 'winz_hepb', 'winz_thin_1019_yr', 'winz_thin_59_yr',
       'winz_income_comp', 'winz_schooling', 'winz_under5_deaths',
       'winz_infant_deaths', 'winz_hiv/aids', 'winz_measles', 'winz_log_gdp',
       'winz_log_population', 'log_pct_exp'],
      dtype='object')
```

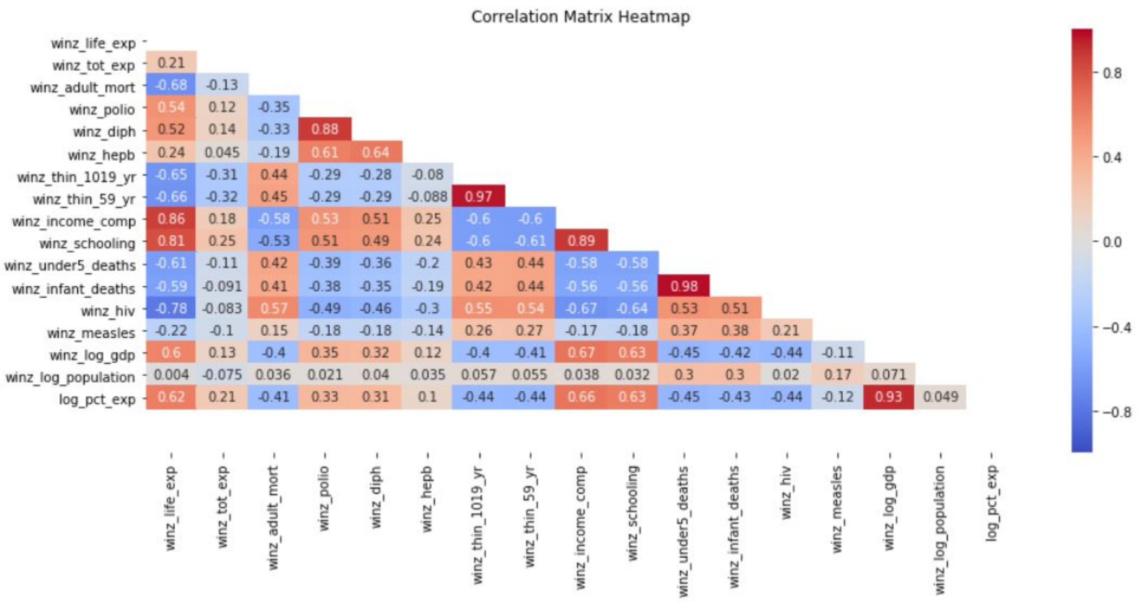
```
Merged_New.rename(columns = {'winz_hiv/aids':'winz_hiv'}, inplace=True)
```

```
C:\Users\Trisha\Anaconda3\lib\site-packages\pandas\core\frame.py:3781: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

```
#making a list of numerical columns
column = list(Merged_New.columns)[26:]
```

```
#making a correlation matrix andploting a heat map
mask = np.triu(Merged_New[column].corr())
plt.figure(figsize=(15,6))
sns.heatmap(Merged_New[column].corr(), annot=True, fmt='.2g', vmin=-1, vmax=1, center=0, cmap='coolwarm', mask=mask)
plt.ylim(19, 0)
plt.title('Correlation Matrix Heatmap')
plt.show()
```

Output:



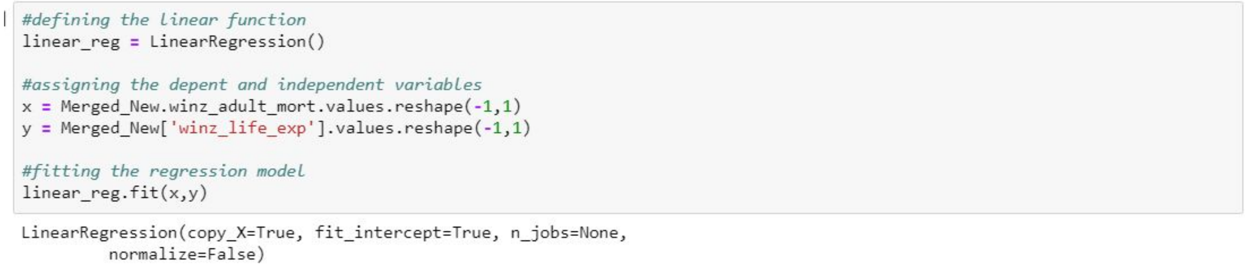
The above heatmap displays a number of important correlations between variables. Some general takeaways from the graphic above:

- Life Expectancy (target variable) appears to be relatively highly correlated (negatively or positively) with:
- Adult Mortality (negative)
- HIV/AIDS (negative)
- Income Composition of Resources (positive)
- Schooling (positive)
- Life expectancy (target variable) is extremely lowly correlated to population (nearly no correlation at all)
- Infant deaths and Under Five deaths are extremely highly correlated
- Percentage Expenditure and GDP are relatively highly correlated
- Hepatitis B vaccine rate is relatively positively correlated with Polio and Diphtheria vaccine rates
- Polio vaccine rate and Diphtheria vaccine rate are very positively correlated
- HIV/AIDS is relatively negatively correlated with Income Composition of Resources
- Thinness of 5-9 Year olds rate and Thinness of 10-15 Year olds rate is extremely highly correlated
- Income Composition of Resources and Schooling are very highly correlated

b. **Question 2: What is the impact of Immunization coverage on life Expectancy?**
From the correlation matrix we understood that there is a very low correlation between the immunization coverages - Polio, Hepatitis B and Diphtheria

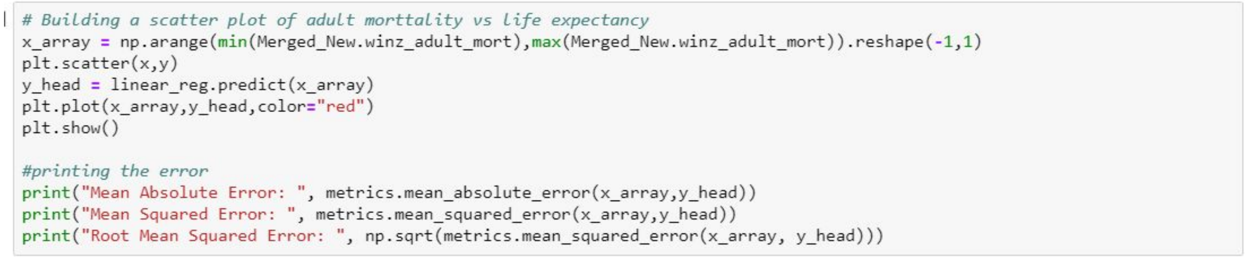
c. **How does Adult mortality rates affect life expectancy?**

We created a linear regression model to know how adult mortality affects life expectancy.



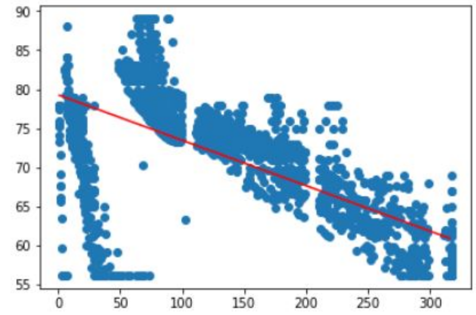
```
#defining the linear function
linear_reg = LinearRegression()

#assigning the depent and independent variables
x = Merged_New.winz_adult_mort.values.reshape(-1,1)
y = Merged_New['winz_life_exp'].values.reshape(-1,1)

#fitting the regression model
linear_reg.fit(x,y)
```

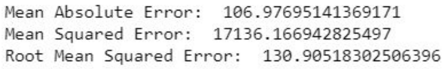
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```



```
# Building a scatter plot of adult morttality vs life expectancy
x_array = np.arange(min(Merged_New.winz_adult_mort),max(Merged_New.winz_adult_mort)).reshape(-1,1)
plt.scatter(x,y)
y_head = linear_reg.predict(x_array)
plt.plot(x_array,y_head,color="red")
plt.show()

#printing the error
print("Mean Absolute Error: ", metrics.mean_absolute_error(x_array,y_head))
print("Mean Squared Error: ", metrics.mean_squared_error(x_array,y_head))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(x_array, y_head)))
```

Output:

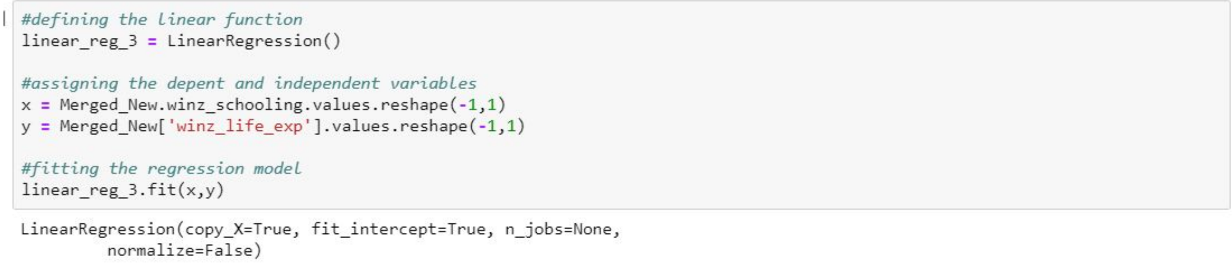




```
Mean Absolute Error:  106.97695141369171
Mean Squared Error:  17136.166942825497
Root Mean Squared Error:  130.90518302506396
```

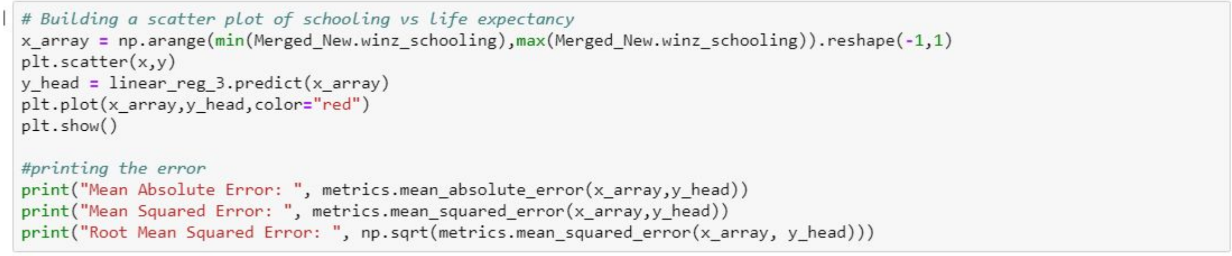
The linear regression scatter plot shows that adult mortality is directly negatively correlated to life expectancy. Lower the adult mortality rate will increase the life expectancy.

d. **What is the impact of schooling on the lifespan of humans?**
We created a linear regression model to know how the number of schooling years affects life expectancy.



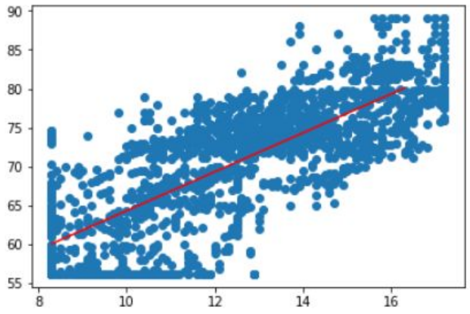
```
#defining the linear function
linear_reg_3 = LinearRegression()

#assigning the depent and independent variables
x = Merged_New.winz_schooling.values.reshape(-1,1)
y = Merged_New['winz_life_exp'].values.reshape(-1,1)

#fitting the regression model
linear_reg_3.fit(x,y)
```

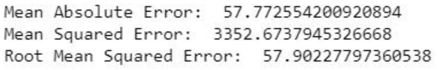
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```



```
# Building a scatter plot of schooling vs life expectancy
x_array = np.arange(min(Merged_New.winz_schooling),max(Merged_New.winz_schooling)).reshape(-1,1)
plt.scatter(x,y)
y_head = linear_reg_3.predict(x_array)
plt.plot(x_array,y_head,color="red")
plt.show()

#printing the error
print("Mean Absolute Error: ", metrics.mean_absolute_error(x_array,y_head))
print("Mean Squared Error: ", metrics.mean_squared_error(x_array,y_head))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(x_array, y_head)))
```

Output:



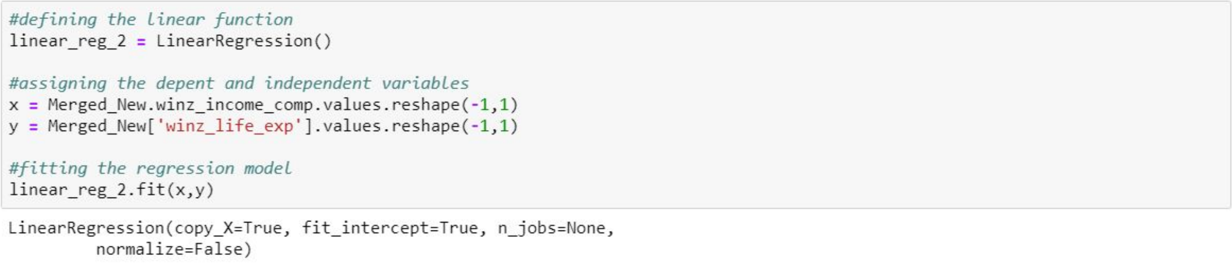


```
Mean Absolute Error:  57.772554200920894
Mean Squared Error:  3352.6737945326668
Root Mean Squared Error:  57.90227797360538
```

The plot above explains that as the number of years of schooling increases the life expectancy of countries increases.

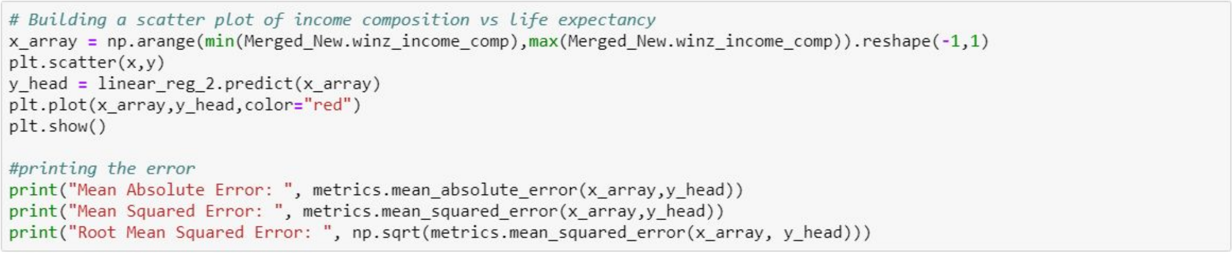
e. **As income composition was on of the tp most affecting factors of life expectancy will create a linear model**

   We created a linear regression model to know how the income composition affects life expectancy.



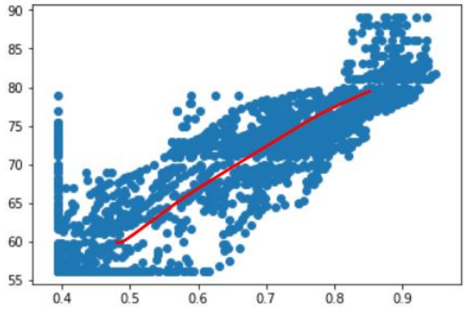
```
#defining the linear function
linear_reg_2 = LinearRegression()

#assigning the depent and independent variables
x = Merged_New.winz_income_comp.values.reshape(-1,1)
y = Merged_New['winz_life_exp'].values.reshape(-1,1)

#fitting the regression model
linear_reg_2.fit(x,y)
```

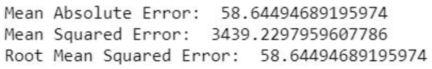
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```



```
# Building a scatter plot of income composition vs life expectancy
x_array = np.arange(min(Merged_New.winz_income_comp),max(Merged_New.winz_income_comp)).reshape(-1,1)
plt.scatter(x,y)
y_head = linear_reg_2.predict(x_array)
plt.plot(x_array,y_head,color="red")
plt.show()

#printing the error
print("Mean Absolute Error: ", metrics.mean_absolute_error(x_array,y_head))
print("Mean Squared Error: ", metrics.mean_squared_error(x_array,y_head))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(x_array, y_head)))
```

Output:





```
Mean Absolute Error:  58.64494689195974
Mean Squared Error:  3439.2297959607786
Root Mean Squared Error:  58.64494689195974
```

The plot above explains that as the income composition increases the life expectancy of countries increases.

Conclusion

- Developed countries have higher life expectancy than developing countries. Developing countries had a steady life expectancy from the year 2000 to 2015 when correlated to GDP. Whereas, when the GDP increased every year from 200 to 2015 there was a linear increase in life expectancy in developing countries.
- Immunization coverages like Polio, Hepatitis B, Diphtheria do not have high correlation with life expectancy. This indicates that immunization coverages do not affect life expectancy.
- Adult Mortality, Number of years of Education, Income Composition and the highest factors which affect life expectancy.
- Adult Mortality is directly negatively correlated to life expectancy. This shows as the adult mortality decreases the life expectancy increases across all countries.
- The more number of years people educate themselves the life expectancy of the country increases.
- Income composition directly affects life expectancy. Higher income composition in a country directs to a better life expectancy.
- When citizens have more schooling years, they will tend to have a better income composition which will lead to higher life expectancy in the country.

Team Member Roles

Data Cleaning - Trisha Chakraborty
Exploratory Data Analysis - Vidisha Badhe
Linear Regression - Isha Havaldar