Linear Regression Analysis

Name: John Rome A. Belocora
Date:04/24/2024
Section: CPE22S3
Teacher: Engr. Roman Richard

```
!pip install hvplot
```

```
Requirement already satisfied: hvplot in /usr/local/lib/python3.10/dist-packages (0.9.2)
Requirement already satisfied: bokeh>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from hvplot) (3.3.4)
Requirement already satisfied: colorcet>=2 in /usr/local/lib/python3.10/dist-packages (from hvplot) (3.1.0)
Requirement already satisfied: holoviews>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from hvplot) (1.17.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from hvplot) (2.0.3)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.10/dist-packages (from hvplot) (1.25.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from hvplot) (24.0)
Requirement already satisfied: panel>=0.11.0 in /usr/local/lib/python3.10/dist-packages (from hvplot) (1.3.8)
Requirement already satisfied: param<3.0,>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from hvplot) (2.1.0)
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (3.1.3)
Requirement already satisfied: contourpy>=1 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (1.2.1)
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (9.4.0)
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (6.0.1)
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (6.3.3)
Requirement already satisfied: xyzservices>=2021.09.1 in /usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (2024.4.0)
Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.10/dist-packages (from holoviews>=1.11.0->hvplot) (3.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas->hvplot) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->hvplot) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas->hvplot) (2024.1)
Requirement already satisfied: markdown in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (3.6)
Requirement already satisfied: markdown-it-py in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (3.0.0)
Requirement already satisfied: linkify-it-py in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (2.0.3)
Requirement already satisfied: mdit-py-plugins in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (0.4.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (2.31.0)
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (4.66.2)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (6.1.0)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (4.11.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2>=2.9->bokeh>=1.0.0->hvplot) (2.1.
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas->hvplot) (1.16.0
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->panel>=0.11.0->hvplot) (0.5.1)
Requirement already satisfied: uc-micro-py in /usr/local/lib/python3.10/dist-packages (from linkify-it-py->panel>=0.11.0->hvplot) (1.0.3
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py->panel>=0.11.0->hvplot) (0.1.2
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot) (2.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot) (202
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import hvplot.pandas

from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.linear_model import LinearRegression

%matplotlib inline


LED = pd.read_csv('Life Expectancy Data.csv')


LED.head()
```

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepa
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	

Data Wrangling

As we can see here, We are trying to view the values of 'Life expectancy' but it keeps showing an error and I think it is because of the naming of the variables

```
LED['Life expectancy']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in get_loc(self,
key)
   3652         try:
-> 3653             return self._engine.get_loc(casted_key)
   3654         except KeyError as err:

                                 4 frames
pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'Life expectancy'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in get_loc(self,
key)
   3653             return self._engine.get_loc(casted_key)
   3654         except KeyError as err:
-> 3655             raise KeyError(key) from err
   3656         except TypeError:
   3657             # If we have a listlike key.  check indexing error will raise
```

Renaming Columns with Spaces into Underscore

I noticed that some of the variables have a excessive spacing, so now we can rename columns with spacing by replacing the spaces with underscores "_" making it more noticable if the variable has an excessive spacing

```
def rename_columns_with_spaces(LED):
    renamed_columns = {}
    for col in LED.columns:
        if ' ' in col:
            #Replace spaces with underscores
            new_col_name = col.replace(' ', '_')
            renamed_columns[col] = new_col_name
    #Rename columns in the DataFrame in place
    LED.rename(columns=renamed_columns, inplace=True)
    return LED

print("Before renaming:")
print(LED)

LED = rename_columns_with_spaces(LED)

print("\nAfter renaming:")
print(LED)
```

```
Before renaming:
          Country  Year  ... Income composition of resources  Schooling
0     Afghanistan  2015  ...                           0.479       10.1
1     Afghanistan  2014  ...                           0.476       10.0
2     Afghanistan  2013  ...                           0.470        9.9
3     Afghanistan  2012  ...                           0.463        9.8
4     Afghanistan  2011  ...                           0.454        9.5
...           ...   ...  ...                             ...        ...
2933     Zimbabwe  2004  ...                           0.407        9.2
2934     Zimbabwe  2003  ...                           0.418        9.5
2935     Zimbabwe  2002  ...                           0.427       10.0
2936     Zimbabwe  2001  ...                           0.427        9.8
2937     Zimbabwe  2000  ...                           0.434        9.8

[2938 rows x 22 columns]

After renaming:
          Country  Year  ... Income_composition_of_resources  Schooling
0     Afghanistan  2015  ...                           0.479       10.1
1     Afghanistan  2014  ...                           0.476       10.0
2     Afghanistan  2013  ...                           0.470        9.9
3     Afghanistan  2012  ...                           0.463        9.8
4     Afghanistan  2011  ...                           0.454        9.5
...           ...   ...  ...                             ...        ...
2933     Zimbabwe  2004  ...                           0.407        9.2
2934     Zimbabwe  2003  ...                           0.418        9.5
2935     Zimbabwe  2002  ...                           0.427       10.0
2936     Zimbabwe  2001  ...                           0.427        9.8
2937     Zimbabwe  2000  ...                           0.434        9.8

[2938 rows x 22 columns]
```

Now we have Identified that there really is a excessive spacing in the variables. for example the variable 'Life expectancy' turned into 'Life_expectancy_' which means that there really is a excessive spacing in some variables

```
#New output
LED.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/AIDS', 'GDP', 'Population',
       '_thinness__1-19_years', '_thinness_5-9_years',
       'Income_composition_of_resources', 'Schooling'],
      dtype='object')
```

Removing Excess Underscore

By just modifying the function, we can use the 'strip' method to remove excess underscores in the variable and a 'if' method which identifies if there are underscores in the beggining of the variable which resolves the issue

```
def excess_underscore(LED):
    renamed_columns = {}
    for col in LED.columns:
        if ' ' in col:
            new_col_name = col.replace(' ', '_')
            # Remove leading or trailing underscore if present next to a space
            new_col_name = new_col_name.strip('_')
            # If the column starts with an underscore, remove it
            if new_col_name.startswith('_'):
                new_col_name = new_col_name[1:]
            renamed_columns[col] = new_col_name
    LED.rename(columns=renamed_columns, inplace=True)
    return LED

print("Before renaming:")
print(LED)

LED = excess_underscore(LED)

print("\nAfter renaming:")
print(LED)
```

```
Before renaming:
          Country  Year  ... Income composition of resources  Schooling
0     Afghanistan  2015  ...                           0.479       10.1
1     Afghanistan  2014  ...                           0.476       10.0
2     Afghanistan  2013  ...                           0.470        9.9
3     Afghanistan  2012  ...                           0.463        9.8
4     Afghanistan  2011  ...                           0.454        9.5
...           ...   ...  ...                             ...        ...
2933     Zimbabwe  2004  ...                           0.407        9.2
2934     Zimbabwe  2003  ...                           0.418        9.5
2935     Zimbabwe  2002  ...                           0.427       10.0
2936     Zimbabwe  2001  ...                           0.427        9.8
2937     Zimbabwe  2000  ...                           0.434        9.8

[2938 rows x 22 columns]

After renaming:
          Country  Year      Status  ...  thinness_5-9_years  Income_composition_of_resources  Schooling
0     Afghanistan  2015  Developing  ...                17.3                            0.479       10.1
1     Afghanistan  2014  Developing  ...                17.5                            0.476       10.0
2     Afghanistan  2013  Developing  ...                17.7                            0.470        9.9
3     Afghanistan  2012  Developing  ...                18.0                            0.463        9.8
4     Afghanistan  2011  Developing  ...                18.2                            0.454        9.5
...           ...   ...         ...  ...                 ...                              ...        ...
2933     Zimbabwe  2004  Developing  ...                 9.4                            0.407        9.2
2934     Zimbabwe  2003  Developing  ...                 9.9                            0.418        9.5
2935     Zimbabwe  2002  Developing  ...                 1.3                            0.427       10.0
2936     Zimbabwe  2001  Developing  ...                 1.7                            0.427        9.8
2937     Zimbabwe  2000  Developing  ...                11.2                            0.434        9.8

[2938 rows x 22 columns]
```

```
#New Output
LED
```

	Country	Year	Status	Life_expectancy	Adult_Mortality	infant_deaths	Alco
0	Afghanistan	2015	Developing	65.0	263.0	62	0
1	Afghanistan	2014	Developing	59.9	271.0	64	0
2	Afghanistan	2013	Developing	59.9	268.0	66	0
3	Afghanistan	2012	Developing	59.5	272.0	69	0
4	Afghanistan	2011	Developing	59.2	275.0	71	0
...	...	...	...	...	...	...	
2933	Zimbabwe	2004	Developing	44.3	723.0	27	4
2934	Zimbabwe	2003	Developing	44.5	715.0	26	4
2935	Zimbabwe	2002	Developing	44.8	73.0	25	4
2936	Zimbabwe	2001	Developing	45.3	686.0	25	1
2937	Zimbabwe	2000	Developing	46.0	665.0	24	1

2938 rows × 22 columns

Identifying Missing Values

```
#Creating a function that can Identify missing values in each column
def missing_values(LED):
    #Count missing values in each column
    missing_values = LED.isnull().sum()
    return missing_values

missing_values = missing_values(LED)
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
Country                             0
Year                                0
Status                              0
Life_expectancy                    10
Adult_Mortality                    10
infant_deaths                       0
Alcohol                           194
percentage_expenditure              0
Hepatitis_B                       553
Measles                             0
BMI                                34
under-five_deaths                   0
Polio                              19
Total_expenditure                 226
Diphtheria                         19
HIV/AIDS                            0
GDP                               448
Population                        652
thinness__1-19_years               34
thinness_5-9_years                 34
Income_composition_of_resources   167
Schooling                         163
dtype: int64
```

We can see here the missing values for the variable 'Alcohol'

```
LED[LED['Alcohol'].isnull()]
```

	Country	Year	Status	Life_expectancy	Adult_Mortality	infant_deaths	Alcoh
32	Algeria	2015	Developing	75.6	19.0	21	Na
48	Angola	2015	Developing	52.4	335.0	66	Na
64	Antigua and Barbuda	2015	Developing	76.4	13.0	0	Na
80	Argentina	2015	Developing	76.3	116.0	8	Na
96	Armenia	2015	Developing	74.8	118.0	1	Na
...	...	...	...	...	...	...	
2858	Venezuela (Bolivarian Republic of)	2015	Developing	74.1	157.0	9	Na
2874	Viet Nam	2015	Developing	76.0	127.0	28	Na
2890	Yemen	2015	Developing	65.7	224.0	37	Na
2906	Zambia	2015	Developing	61.8	33.0	27	Na
2922	Zimbabwe	2015	Developing	67.0	336.0	22	Na

194 rows × 22 columns

Now we can see here that the Alcohol column of the Country Algeria consist of different values and also a NaN value.

```
LED[LED['Country']=='Algeria']
```

	Country	Year	Status	Life_expectancy	Adult_Mortality	infant_deaths	Alcohol	percentage_expenditure	Hepatitis_B	Measles	BM
32	Algeria	2015	Developing	75.6	19.0	21	NaN	0.000000	95.0	63	59.
33	Algeria	2014	Developing	75.4	11.0	21	0.01	54.237318	95.0	0	58.
34	Algeria	2013	Developing	75.3	112.0	21	0.53	544.450743	95.0	25	57.
35	Algeria	2012	Developing	75.1	113.0	21	0.66	555.926083	95.0	18	56.
36	Algeria	2011	Developing	74.9	116.0	21	0.56	509.002041	95.0	112	55.
37	Algeria	2010	Developing	74.7	119.0	21	0.45	430.717586	95.0	103	53.
38	Algeria	2009	Developing	74.4	123.0	20	0.50	352.063642	94.0	107	52.
39	Algeria	2008	Developing	74.1	126.0	20	0.46	43.087173	91.0	217	51.
40	Algeria	2007	Developing	73.8	129.0	20	0.44	320.323924	9.0	0	5.
41	Algeria	2006	Developing	73.4	132.0	20	0.36	270.240196	8.0	944	49.
42	Algeria	2005	Developing	72.9	136.0	19	0.50	2.548923	83.0	2302	48.
43	Algeria	2004	Developing	72.3	14.0	19	0.45	220.393699	81.0	3289	47.
44	Algeria	2003	Developing	71.7	146.0	20	0.34	25.018523	NaN	15374	47.
45	Algeria	2002	Developing	71.6	145.0	20	0.36	148.511984	NaN	5862	46.
46	Algeria	2001	Developing	71.4	145.0	20	0.23	147.986071	NaN	2686	45.
47	Algeria	2000	Developing	71.3	145.0	21	0.25	154.455944	NaN	0	44.

To fix this issue, we can get the mean of the Alchohol of the country Algeria and replace the missing value with the mean

Filling missing values with mean depending on each Country

We are filling the missing values for all columns except for 'Country' and 'Status' By getting the mean for each column values depending on the country then filling the missing value with the given mean

```
#Columns to fill missing values
#Excluding the 'Country' and 'Status' column since they are object dtypes
columns_to_fill = LED.columns.difference(['Country', 'Status'])

#Fill missing values for each column separately based on the country
for col in columns_to_fill:
    LED[col] = LED.groupby('Country')[col].transform(lambda x: x.fillna(x.mean()))

print(LED)
```

```
            Country  Year      Status  ...  thinness_5-9_years  Income_composition_of_resources  Schooling
0       Afghanistan  2015  Developing  ...                17.3                            0.479       10.1
1       Afghanistan  2014  Developing  ...                17.5                            0.476       10.0
2       Afghanistan  2013  Developing  ...                17.7                            0.470        9.9
3       Afghanistan  2012  Developing  ...                18.0                            0.463        9.8
4       Afghanistan  2011  Developing  ...                18.2                            0.454        9.5
...             ...   ...         ...  ...                 ...                              ...        ...
2933        Zimbabwe  2004  Developing  ...                 9.4                            0.407        9.2
2934        Zimbabwe  2003  Developing  ...                 9.9                            0.418        9.5
2935        Zimbabwe  2002  Developing  ...                 1.3                            0.427       10.0
2936        Zimbabwe  2001  Developing  ...                 1.7                            0.427        9.8
2937        Zimbabwe  2000  Developing  ...                11.2                            0.434        9.8

[2938 rows x 22 columns]
```

```
LED.isnull().sum()
```

```
Country                             0
Year                                0
Status                              0
Life_expectancy                    10
Adult_Mortality                    10
infant_deaths                       0
Alcohol                            17
percentage_expenditure              0
Hepatitis_B                       144
Measles                             0
BMI                                34
under-five_deaths                   0
Polio                               0
Total_expenditure                  32
Diphtheria                          0
HIV/AIDS                            0
GDP                               405
Population                        648
thinness__1-19_years               34
thinness_5-9_years                 34
Income_composition_of_resources   167
Schooling                         163
dtype: int64
```

Some missing values cannot be changed since they have no data collected in their Country. We can see Yemen's GDP and Population as an example below:

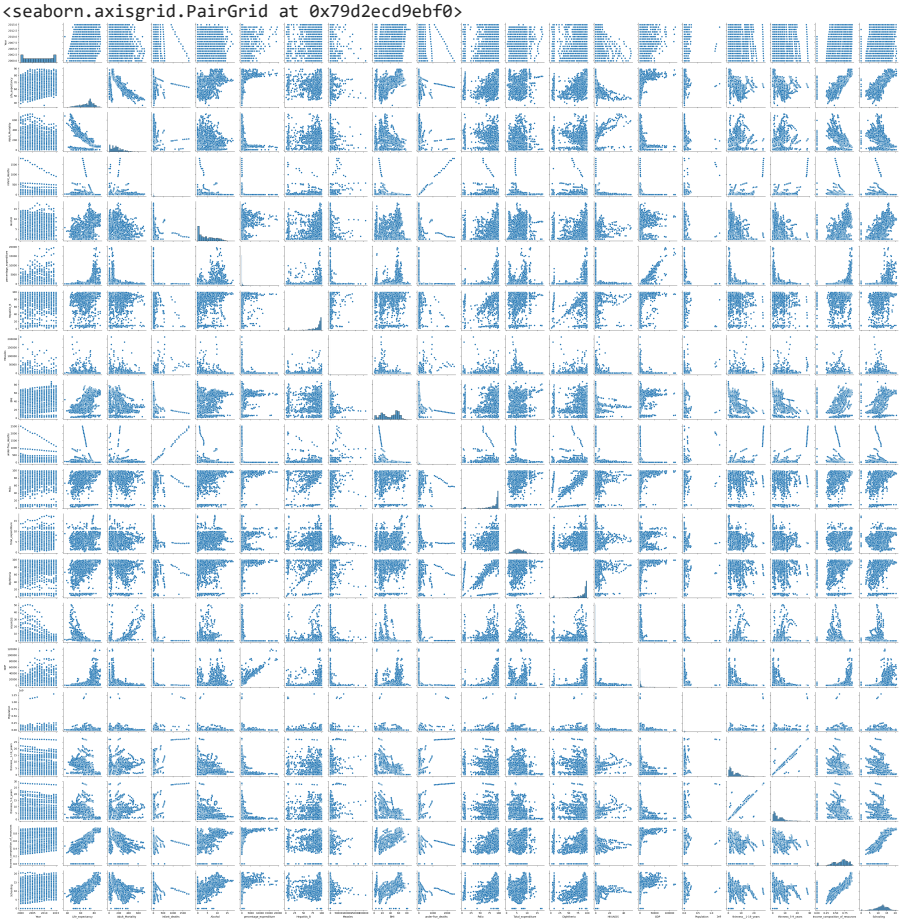
```
LED[LED['Country'] == 'Yemen']
```

	Country	Year	Status	Life_expectancy	Adult_Mortality	infant_deaths	Alcoho
2890	Yemen	2015	Developing	65.7	224.0	37	0.04733
2891	Yemen	2014	Developing	65.4	228.0	37	0.01000
2892	Yemen	2013	Developing	65.4	226.0	36	0.04000
2893	Yemen	2012	Developing	64.7	236.0	36	0.04000
2894	Yemen	2011	Developing	64.6	234.0	35	0.04000
2895	Yemen	2010	Developing	64.4	233.0	35	0.06000
2896	Yemen	2009	Developing	64.1	235.0	36	0.03000
2897	Yemen	2008	Developing	63.8	238.0	37	0.04000
2898	Yemen	2007	Developing	63.4	24.0	38	0.05000
2899	Yemen	2006	Developing	63.0	242.0	39	0.04000
2900	Yemen	2005	Developing	62.6	245.0	40	0.04000
2901	Yemen	2004	Developing	62.2	247.0	42	0.06000
2902	Yemen	2003	Developing	61.9	249.0	43	0.04000
2903	Yemen	2002	Developing	61.5	25.0	45	0.07000
2904	Yemen	2001	Developing	61.1	251.0	46	0.08000
2905	Yemen	2000	Developing	68.0	252.0	48	0.07000

Exploratory Data Analysis (EDA)

```
#Pair plot of the Dataframe
sns.pairplot(LED)
```

```
<seaborn.axisgrid.PairGrid at 0x79d2ecd9ebf0>
```



```
# Creating a new dataframe copy with the index of Country Column
df = LED.copy()
df.set_index('Country', inplace=True)
df
```

Country	Year	Status	Life_expectancy	Adult_Mortality	infant_deaths	Alcohol
Afghanistan	2015	Developing	65.0	263.0	62	0.01
Afghanistan	2014	Developing	59.9	271.0	64	0.01
Afghanistan	2013	Developing	59.9	268.0	66	0.01
Afghanistan	2012	Developing	59.5	272.0	69	0.01
Afghanistan	2011	Developing	59.2	275.0	71	0.01
...	...	...	...	...	...	...
Zimbabwe	2004	Developing	44.3	723.0	27	4.36
Zimbabwe	2003	Developing	44.5	715.0	26	4.06
Zimbabwe	2002	Developing	44.8	73.0	25	4.43
Zimbabwe	2001	Developing	45.3	686.0	25	1.72
Zimbabwe	2000	Developing	46.0	665.0	24	1.68

2938 rows × 21 columns

```
#Dropping Status column since it is a object datatype
df.drop('Status', axis=1, inplace=True)
df
```

Country	Year	Life_expectancy	Adult_Mortality	infant_deaths	Alcohol	percentage_
Afghanistan	2015	65.0	263.0	62	0.01	
Afghanistan	2014	59.9	271.0	64	0.01	
Afghanistan	2013	59.9	268.0	66	0.01	
Afghanistan	2012	59.5	272.0	69	0.01	
Afghanistan	2011	59.2	275.0	71	0.01	
...	...	...	...	...	...	
Zimbabwe	2004	44.3	723.0	27	4.36	
Zimbabwe	2003	44.5	715.0	26	4.06	
Zimbabwe	2002	44.8	73.0	25	4.43	
Zimbabwe	2001	45.3	686.0	25	1.72	
Zimbabwe	2000	46.0	665.0	24	1.68	

2938 rows × 20 columns

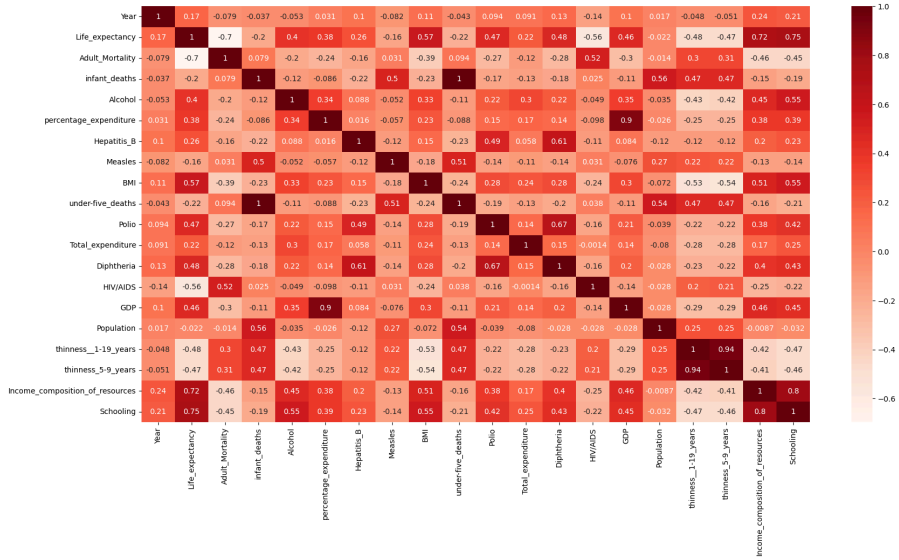
Next steps: View recommended plots

```
#Correlation of each values
df.corr()
```

	Year	Life_expectancy	Adult_Mortality	infant_deat
Year	1.000000	0.170033	-0.079052	-0.0374
Life_expectancy	0.170033	1.000000	-0.696359	-0.1965
Adult_Mortality	-0.079052	-0.696359	1.000000	0.0787
infant_deaths	-0.037415	-0.196557	0.078756	1.0000
Alcohol	-0.052990	0.404877	-0.195848	-0.1156
percentage_expenditure	0.031400	0.381864	-0.242860	-0.0856
Hepatitis_B	0.104333	0.256762	-0.162476	-0.2235
Measles	-0.082493	-0.157586	0.031176	0.5011
BMI	0.108974	0.567694	-0.387017	-0.2272
under-five_deaths	-0.042937	-0.222529	0.094146	0.9966
Polio	0.094158	0.465556	-0.274823	-0.1706
Total_expenditure	0.090740	0.218086	-0.115281	-0.1286
Diphtheria	0.134337	0.479495	-0.275131	-0.1751
HIV/AIDS	-0.139741	-0.556556	0.523821	0.0252
GDP	0.101620	0.461455	-0.296049	-0.1084
Population	0.016969	-0.021538	-0.013647	0.5568
thinness__1-19_years	-0.047876	-0.477183	0.302904	0.4657
thinness_5-9_years	-0.050929	-0.471584	0.308457	0.4713
Income_composition_of_resources	0.243468	0.724776	-0.457626	-0.1451
Schooling	0.209400	0.751975	-0.454612	-0.1937

```
# Heatmap correlation
plt.figure(figsize=(20,10))
sns.heatmap(df.corr(), annot=True,cmap='Reds')
```

```
<Axes: >
```



Training a Linear Regression Model

X and y arrays

```
X = df.drop('Adult_Mortality', axis=1).isnull()
y = df['under-five_deaths']
```

```
print("X=", X.shape,"\ny", y.shape)
```

```
X= (2938, 19)
y (2938,)
```

Train Test Split

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```

```
X_train.shape
```

```
(2056, 19)
```

```
X_test.shape
```

```
(882, 19)
```

Linear Regression

```
model = LinearRegression()
```

```
X_train.isnull().sum()
```

```
Year                             0
Life_expectancy                  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/AIDS                         0
GDP                              0
Population                       0
thinness__1-19_years             0
thinness_5-9_years               0
Income_composition_of_resources  0
Schooling                        0
dtype: int64
```

```
y.dropna()
```

```
Country
Afghanistan    83
Afghanistan    86
Afghanistan    89
Afghanistan    93
Afghanistan    97
               ..
Zimbabwe       42
Zimbabwe       41
Zimbabwe       40
Zimbabwe       39
Zimbabwe       39
Name: under-five_deaths, Length: 2938, dtype: int64
```

```
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Model Evaluation

```
model.coef_
```

```
array([ 0.00000000e+00, -6.25502432e+01,  5.61280495e+13,  1.76368811e+01,
       -8.21042429e+12,  1.02659100e+01, -3.39603354e+12, -3.69738331e+14,
        1.26386951e+13, -4.99657741e+15, -3.16179175e+01,  4.99657741e+15,
```

```
       -2.00000000e+00,  8.34327964e+00, -3.90079297e+01,  9.23400931e+13,
        2.77398238e+14,  4.40317951e+01,  1.46901191e+01])
```

```
pd.DataFrame(model.coef_, X.columns, columns=['Coeficients'])
```

	Coeficients
Year	0.000000e+00
Life_expectancy	-6.255024e+01
infant_deaths	5.612805e+13
Alcohol	1.763688e+01
percentage_expenditure	-8.210424e+12
Hepatitis_B	1.026591e+01
Measles	-3.396034e+12
BMI	-3.697383e+14
under-five_deaths	1.263870e+13
Polio	-4.996577e+15
Total_expenditure	-3.161792e+01