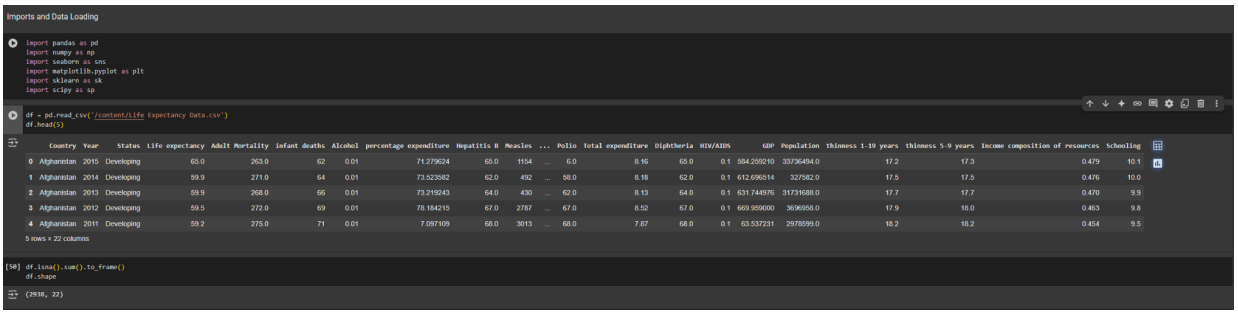
Here I'm importing the modules to be used later and loading the data.



The I look at the data I'm working with, I find that it's integers, floats and strings

```
[51] df.info()
     print(df.columns)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Country                          2938 non-null   object
 1   Year                             2938 non-null   int64
 2   Status                           2938 non-null   object
 3   Life expectancy                  2928 non-null   float64
 4   Adult Mortality                  2928 non-null   float64
 5   infant deaths                    2938 non-null   int64
 6   Alcohol                          2744 non-null   float64
 7   percentage expenditure           2938 non-null   float64
 8   Hepatitis B                      2385 non-null   float64
 9   Measles                          2938 non-null   int64
 10   BMI                             2904 non-null   float64
 11  under-five deaths                2938 non-null   int64
 12  Polio                            2919 non-null   float64
 13  Total expenditure                2712 non-null   float64
 14  Diphtheria                       2919 non-null   float64
 15   HIV/AIDS                        2938 non-null   float64
 16  GDP                              2490 non-null   float64
 17  Population                       2286 non-null   float64
 18   thinness  1-19 years            2904 non-null   float64
 19   thinness 5-9 years             2904 non-null   float64
 20  Income composition of resources  2771 non-null   float64
 21  Schooling                        2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
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')
```

I noticed there were al ot of null values in the data, so instead of dropping them I replaced them with the mean of each

```
[52] df['Life expectancy '] = df['Life expectancy '].fillna(df['Life expectancy '].mean())
     df['Adult Mortality'] = df['Adult Mortality'].fillna(df['Adult Mortality'].mean())
     df['Alcohol'] = df['Alcohol'].fillna(df['Alcohol'].mean())
     df['Hepatitis B'] = df['Hepatitis B'].fillna(df['Hepatitis B'].mean())
     df[' BMI '] = df[' BMI '].fillna(df[' BMI '].mean())
     df['Polio'] = df['Polio'].fillna(df['Polio'].mean())
     df['Total expenditure'] = df['Total expenditure'].fillna(df['Total expenditure'].mean())
     df['Diphtheria '] = df['Diphtheria '].fillna(df['Diphtheria '].mean())
     df['GDP'] = df['GDP'].fillna(df['GDP'].mean())
     df['Population'] = df['Population'].fillna(df['Population'].mean())
     df[' thinness  1-19 years'] = df[' thinness  1-19 years'].fillna(df[' thinness  1-19 years'].mean())
     df[' thinness 5-9 years'] = df[' thinness 5-9 years'].fillna(df[' thinness 5-9 years'].mean())
     df['Income composition of resources'] = df['Income composition of resources'].fillna(df['Income composition of resources'].mean())
     df['Schooling'] = df['Schooling'].fillna(df['Schooling'].mean())
```

I select the relevant columns to work with

Process the Data

```
[53] df_select = df[['Country', 'Year', 'Life expectancy ', 'GDP', 'Total expenditure']]
     df_select_2015 = df_select[df_select['Year'] == 2015]
     df_select_2015.head()
     # df_select_2015.isnull().sum()
```

	Country	Year	Life expectancy	GDP	Total expenditure
0	Afghanistan	2015	65.0	584.259210	8.16000
16	Albania	2015	77.8	3954.227830	6.00000
32	Algeria	2015	75.6	4132.762920	5.93819
48	Angola	2015	52.4	3695.793748	5.93819
64	Antigua and Barbuda	2015	76.4	13566.954100	5.93819

Using the describe function I find the metadata

Descriptive Analysis

```
[54] df_select.describe(include='all')
```

	Country	Year	Life expectancy	GDP	Total expenditure
count	2938	2938.000000	2938.000000	2938.000000	2938.000000
unique	193	NaN	NaN	NaN	NaN
top	Afghanistan	NaN	NaN	NaN	NaN
freq	16	NaN	NaN	NaN	NaN
mean	NaN	2007.518720	69.224932	7483.158469	5.938190
std	NaN	4.613841	9.507640	13136.800417	2.400274
min	NaN	2000.000000	36.300000	1.681350	0.370000
25%	NaN	2004.000000	63.200000	580.486996	4.370000
50%	NaN	2008.000000	72.000000	3116.561755	5.938190
75%	NaN	2012.000000	75.600000	7483.158469	7.330000
max	NaN	2015.000000	89.000000	119172.741800	17.600000

Using the corr() function I find the relationship between the 3 variables with 1 being very strong.

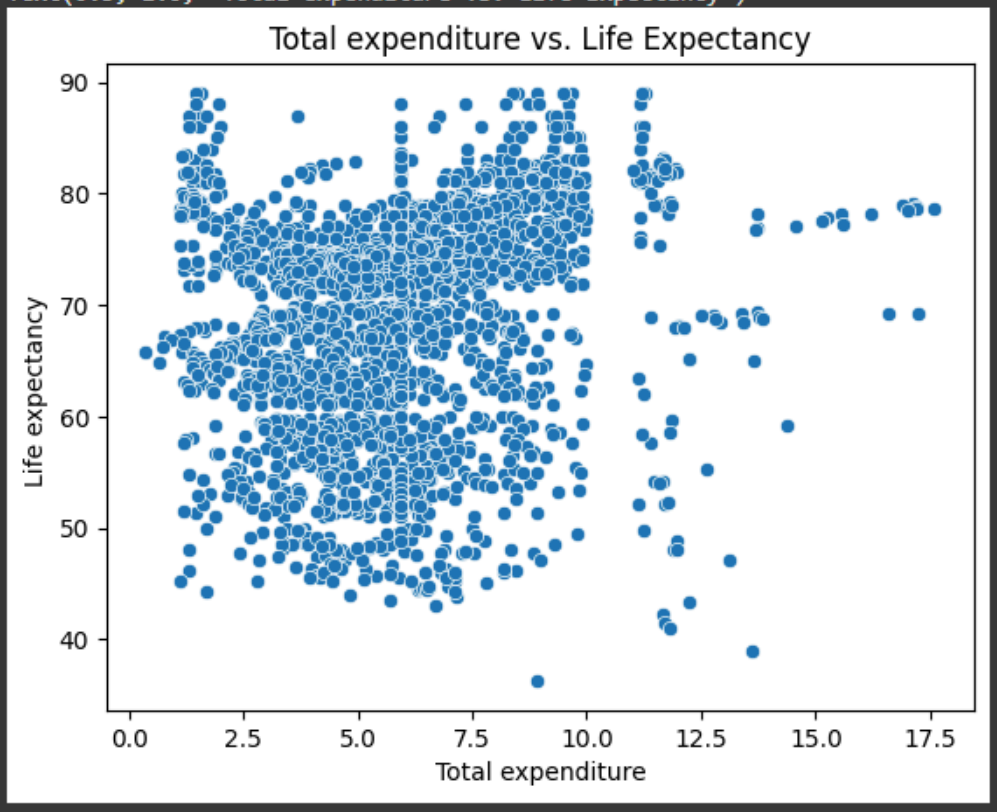
```
[55] df_select[['Life expectancy ', 'GDP', 'Total expenditure']].corr()
```

	Life expectancy	GDP	Total expenditure
Life expectancy	1.000000	0.430493	0.207981
GDP	0.430493	1.000000	0.121467
Total expenditure	0.207981	0.121467	1.000000

Now plotting the scatterpot.

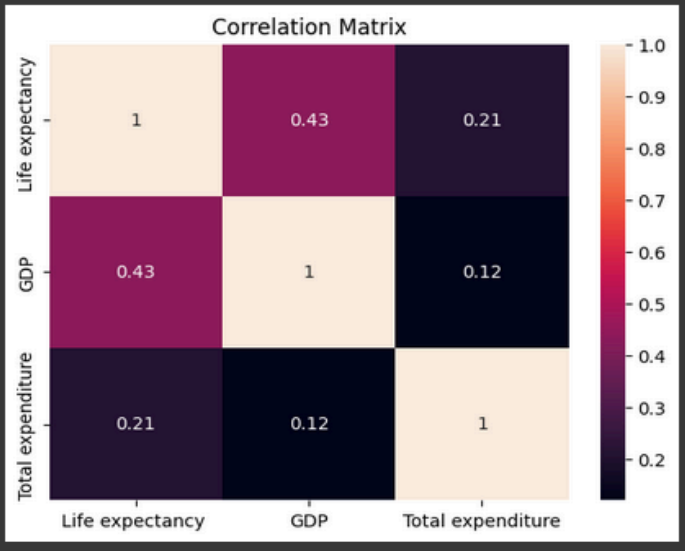
```
[56] sns.scatterplot(data=df_select, x='Total expenditure', y='Life expectancy ')
     plt.title('Total expenditure vs. Life Expectancy')
```

```
Text(0.5, 1.0, 'Total expenditure vs. Life Expectancy')
```



Then I use heatmap to visualise the correlation matrices

```
correlation_matrices= df_select[['Life expectancy ', 'GDP', 'Total expenditure']].corr()
sns.heatmap(correlation_matrices, annot=True)
plt.title('Correlation Matrix')
plt.show()
```



After which I split the data into train and test(The beauty of Gemini, I pressed '\t' working code was generated)

Predictive Analysis

```
Generated code may be subject to a licence | 1325ankit/NYC_Taxi_Time_Prediction
X = df_select[['Total expenditure']]
y = df_select['Life expectancy ']
```

```
[59] Generated code may be subject to a licence | SARIT42/ML-Notebooks
X_train, X_test, y_train, y_test = sk.model_selection.train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[60] Generated code may be subject to a licence | Chavarria18/Deteccion-Noticias-Falsas
model = sk.linear_model.LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

Using the R squared to measure the accuracy of my model
And the mean squared error to measure the its inaccuracy

```
Generated code may be subject to a licence | Chavarria18/Deteccion-Noticias-Falsas
y_prediction = model.predict(X_test)
```

```
[62] r_squared =sk.metrics.r2_score(y_test, y_prediction)
print(f'R-squared: {r_squared}')
```

```
R-squared: 0.06592611341873733
```

```
[63] mean_squeare_error = sk.metrics.mean_squared_error(y_test, y_prediction)
print(f'Mean Squared Error: {mean_squeare_error}')
```

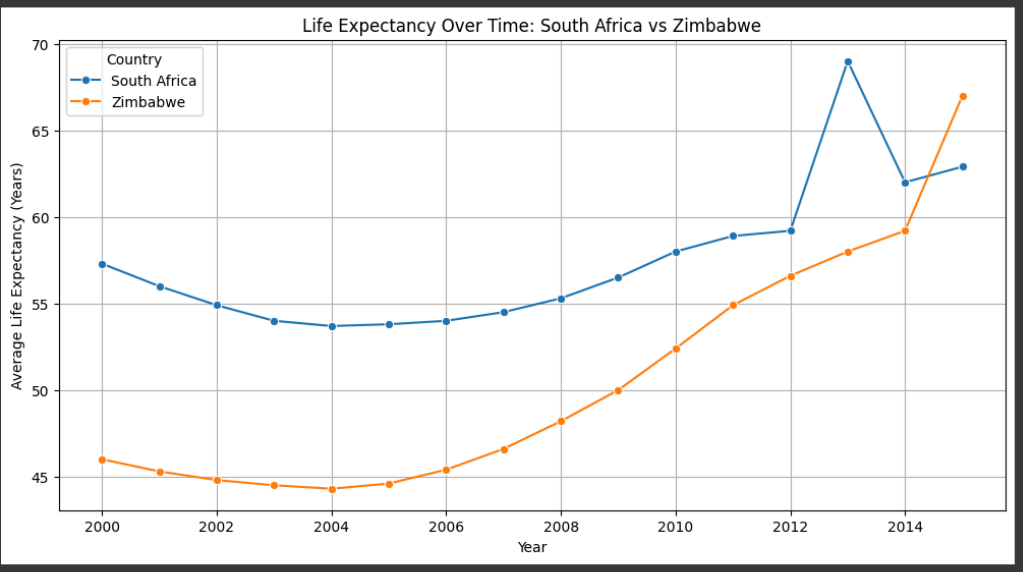
```
Mean Squared Error: 80.92374981595093
```

```
[64] selected_countries = ['South Africa', 'Zimbabwe']
select_countries = df[df_select['Country'].isin(selected_countries)]
```

```
[65] country_year_stats = select_countries.groupby(['Country', 'Year']).agg({'Life expectancy ': 'mean', 'Total expenditure': 'mean'}).reset_index()
```

Now getting the average life expectancy of South africa and Zimbabwe over the years.

```
plt.figure(figsize=(12, 6))
sns.lineplot(data=country_year_stats, x='Year', y='Life expectancy ', hue='Country', marker='o')
plt.title('Life Expectancy Over Time: South Africa vs Zimbabwe')
plt.ylabel('Average Life Expectancy (Years)')
plt.grid(True)
plt.show()
```



Conclusion:

It seems as the years progress and countries keep investing more and more into their healthcare their life expectancy becomes prolonged. As expected the bigger the economy of the country(higher GDP) it seems to have a positive correlation with life expectancy. Therefore, developed countries have a much more positive life expectancy than developing countries.