CSE422: Artificial Intelligence

Project Report

Project topic:

Life Expectancy Prediction (WHO)

Submitted by

Group no: 10

Section: 03

Student Name	Student ID
Mahir Tajwar Rahman	22299422
Asiful Islam Mahir	22299318

Table of Contents

1. Introduction

This project explores the factors influencing life expectancy across various countries by leveraging machine learning techniques. It examines the impact of health, economic, and social factors, utilizing data from the World Health Organization (WHO) and the United Nations (UN). The primary objective is to understand the role of variables such as immunization rates and human development in determining life expectancy.

The project aims to address the critical question of what drives disparities in life expectancy globally. By studying these patterns and relationships, it seeks to provide insights that could inform policy-making and resource allocation. The motivation stems from the desire to better understand how modifiable factors can improve public health and quality of life, ultimately contributing to a more equitable world.

2. Dataset Description

- **Source:**

  Link: https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who

  Reference: World Health Organisation (WHO)

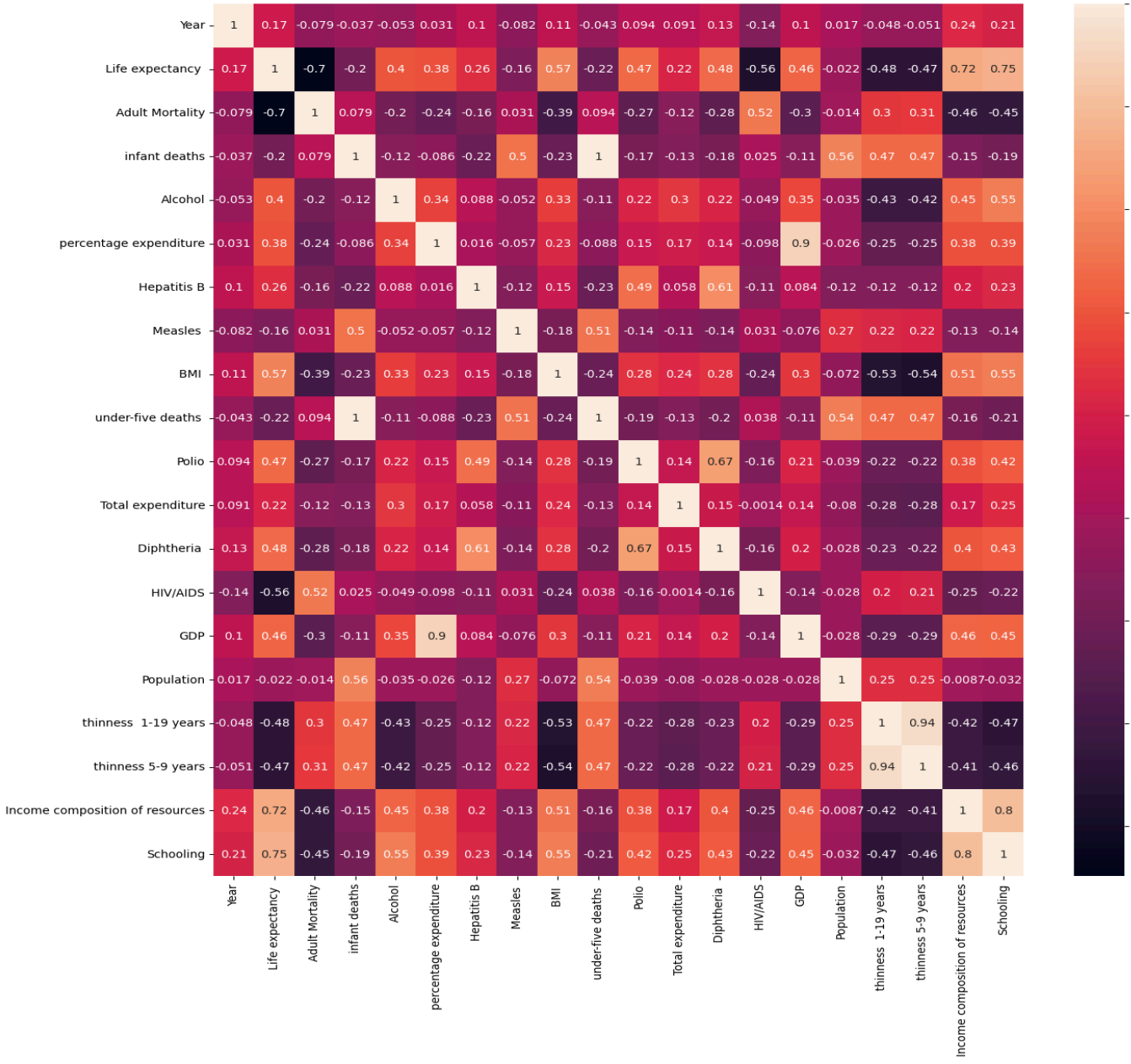
- **Features:** There are 21 features. 22 columns (21 Features, 1 Target)

- **Classification/Regression:** Regression problem (Life expectancy is measured in years (e.g., 75.3 years) It's a continuous numerical value. There are no discrete classes or categories to predict. As we know Regression problems predict continuous numerical values so we are considering this as a regression problem)

- **Data points:** 2938 rows

```
<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
```

- **Feature types:**
    - **Quantitative**: GDP, Life expectancy, Adult Mortality, etc.
    - **Categorical**: Country, Status (Developing/Developed)
- **Correlation:** We have applied joint plot, Pair plot, histogram, and boxplot to understand the relation between life expectancy for every single feature, and we applied heatmap to understand the correlations between the features



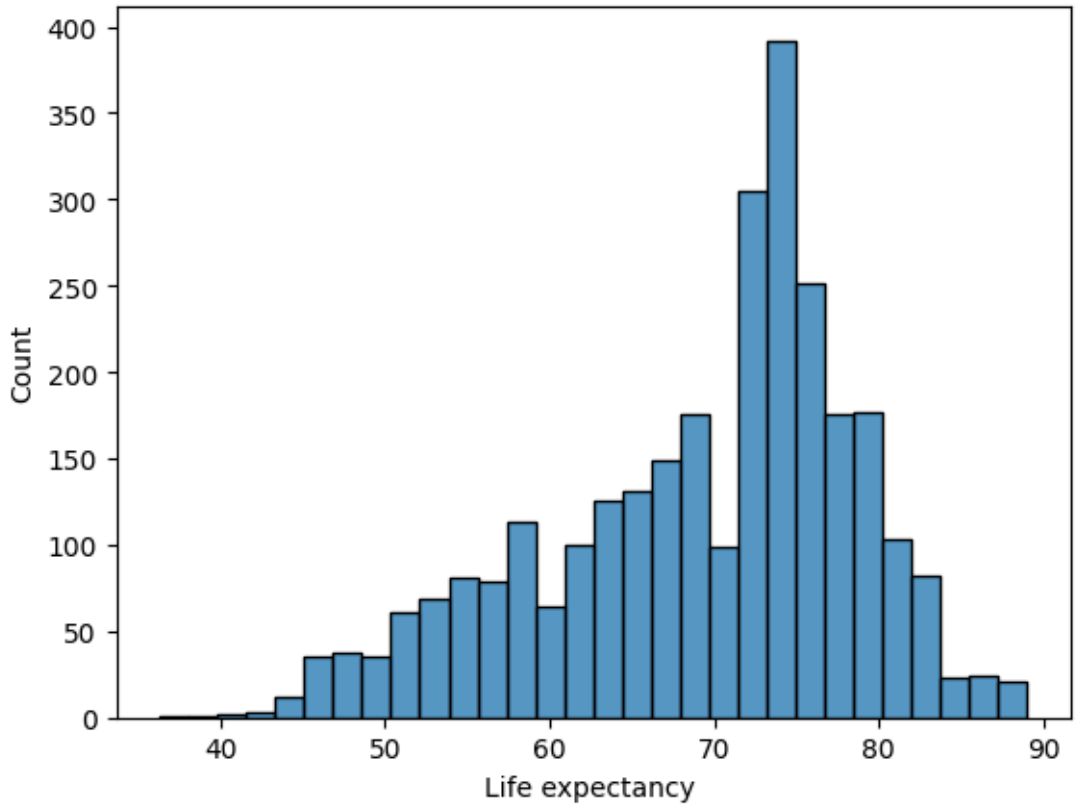
1 Correlation between the features

The color scale helps interpret the relationships:

- Light/warm colors (towards 1.0): Strong positive correlation
- Dark/cool colors (towards -1.0): Strong negative correlation
- Neutral colors (around 0): Little to no correlation

- **Imbalance Dataset:** This visual representation helps us understand data distribution and identify unusual patterns, allowing us to understand the data better.

  As this is a regression problem we don't have any output features. Here we are attaching the life expectancy histogram. The graph is not normally distributed.



3. Dataset Preprocessing:

- **Null values:** We have many null values for dataset features. If the percentage of null values in a column is above 50%, we may delete the column. Otherwise, we will try to fix that. However, we observed not a single feature with more than 50% null value.

  We have used KNN imputer to fill the null values, as we had numerical values here.

1. Null values Before

```
2 life_expectancy.isnull().sum()
```

	0
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

2: Null values after

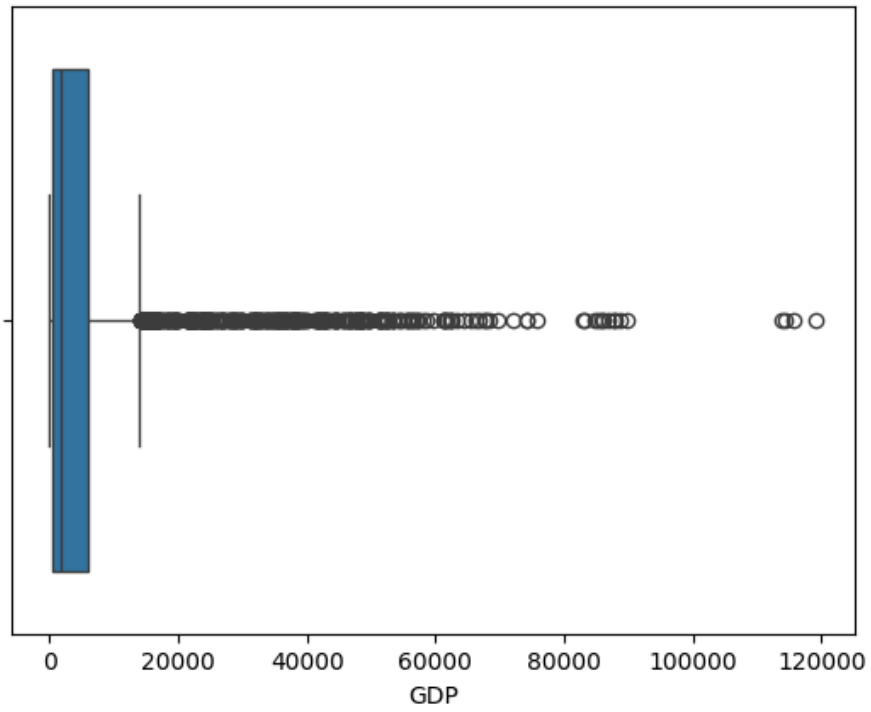
```
1 life_expectancy.isnull().sum()
```

	0
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/AIDS	0
GDP	0
Population	0
thinness 1-19 years	0
thinness 5-9 years	0
Income composition of resources	0
Schooling	0

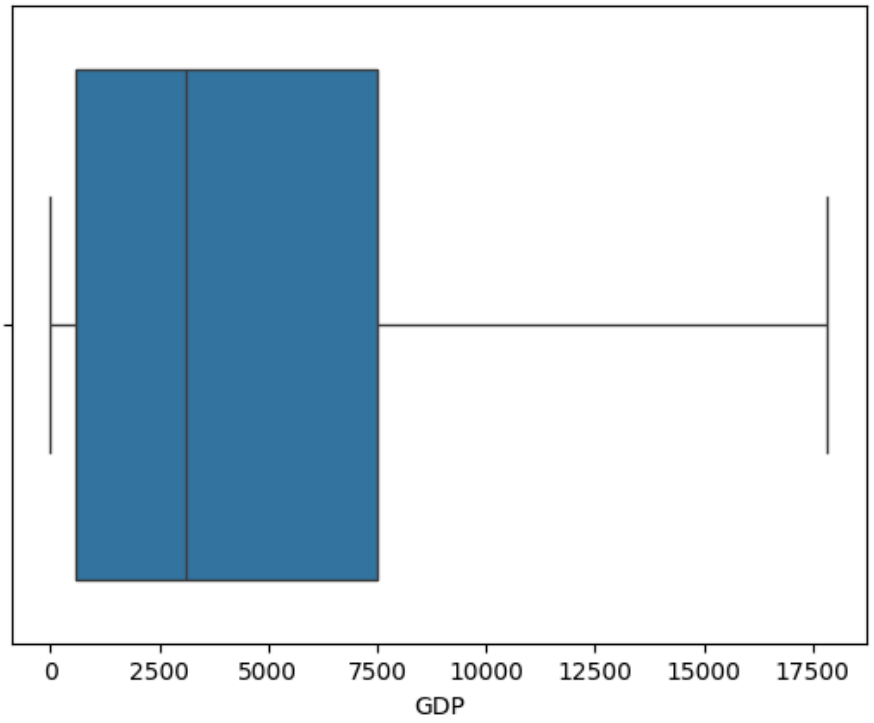
dtype: int64

- **Outliers:** As we know, we can only do outlier treatment if the number of outliers is negligible and the values are continuous numerical values. To observe how many outliers are there for each feature, we did box plotting in the very beginning, and after that using the whisker function which sets a range and compares the range, we found the outliers and replaced the outliers with the lower or upper whisker (capping).

1: Before outlier treatment



2: After outlier treatment



- **Duplicate and Garbage values:** No duplicate or garbage values in our dataset.

```
[ ]  1 #finding duplicates
     2 life_expectancy.duplicated().sum()

     0
```

```
1 #identifying garbage values
2 for i in life_expectancy.select_dtypes(include="object").columns:
3   print(life_expectancy[i].value_counts())
4   print("***"*10)
```

```
Country
Afghanistan             16
Peru                    16
Nicaragua               16
Niger                   16
Nigeria                 16
                        ..
Niue                     1
San Marino               1
Nauru                    1
Saint Kitts and Nevis    1
Dominica                 1
Name: count, Length: 193, dtype: int64
******************************
Status
Developing    2426
Developed      512
Name: count, dtype: int64
******************************
```

- **Categorical values:** For fitting the data into a model we need all the columns in numerical form as some machine learning models cannot properly learn from categorical values. converting object columns into numerical is called encoding. We had 2 categorical columns(Country and status) and have applied one hot encoding with pd.getdummies to solve this problem.

Categorical values

```
1 life_expectancy["Country"].unique()
```

```
array(['Afghanistan', 'Albania', 'Algeria', 'Angola',
       'Antigua and Barbuda', 'Argentina', 'Armenia', 'Australia',
       'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh',
       'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bhutan',
       'Bolivia (Plurinational State of)', 'Bosnia and Herzegovina',
       'Botswana', 'Brazil', 'Brunei Darussalam', 'Bulgaria',
       'Burkina Faso', 'Burundi', "Côte d'Ivoire", 'Cabo Verde',
       'Cambodia', 'Cameroon', 'Canada', 'Central African Republic',
       'Chad', 'Chile', 'China', 'Colombia', 'Comoros', 'Congo',
       'Cook Islands', 'Costa Rica', 'Croatia', 'Cuba', 'Cyprus',
       'Czechia', "Democratic People's Republic of Korea",
       'Democratic Republic of the Congo', 'Denmark', 'Djibouti',
       'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Estonia',
       'Ethiopia', 'Fiji', 'Finland', 'France', 'Gabon', 'Gambia',
       'Georgia', 'Germany', 'Ghana', 'Greece', 'Grenada', 'Guatemala',
       'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti', 'Honduras',
       'Hungary', 'Iceland', 'India', 'Indonesia',
       'Iran (Islamic Republic of)', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati',
       'Kuwait', 'Kyrgyzstan', "Lao People's Democratic Republic",
       'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Lithuania',
       'Luxembourg', 'Madagascar', 'Malawi', 'Malaysia', 'Maldives',
       'Mali', 'Malta', 'Marshall Islands', 'Mauritania', 'Mauritius',
       'Mexico', 'Micronesia (Federated States of)', 'Monaco', 'Mongolia',
       'Montenegro', 'Morocco', 'Mozambique', 'Myanmar', 'Namibia',
       'Nauru', 'Nepal', 'Netherlands', 'New Zealand', 'Nicaragua',
       'Niger', 'Nigeria', 'Niue', 'Norway', 'Oman', 'Pakistan', 'Palau',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines',
       'Poland', 'Portugal', 'Qatar', 'Republic of Korea',
       'Republic of Moldova', 'Romania', 'Russian Federation', 'Rwanda',
       'Saint Kitts and Nevis', 'Saint Lucia',
       'Saint Vincent and the Grenadines', 'Samoa', 'San Marino',
       'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Seychelles', 'Sierra Leone', 'Singapore', 'Slovakia', 'Slovenia',
       'Solomon Islands', 'Somalia', 'South Africa', 'South Sudan',
       'Spain', 'Sri Lanka', 'Sudan', 'Suriname', 'Swaziland', 'Sweden',
       'Switzerland', 'Syrian Arab Republic', 'Tajikistan', 'Thailand',
       'The former Yugoslav republic of Macedonia', 'Timor-Leste', 'Togo',
       'Tonga', 'Trinidad and Tobago', 'Tunisia', 'Turkey',
       'Turkmenistan', 'Tuvalu', 'Uganda', 'Ukraine',
       'United Arab Emirates',
       'United Kingdom of Great Britain and Northern Ireland',
       'United Republic of Tanzania', 'United States of America',
       'Uruguay', 'Uzbekistan', 'Vanuatu',
       'Venezuela (Bolivarian Republic of)', 'Viet Nam', 'Yemen',
       'Zambia', 'Zimbabwe'], dtype=object)
```

```
1 life_expectancy["Status"].unique()
```

```
array(['Developing', 'Developed'], dtype=object)
```

Categorical values after encoding

```
1 encoded_life_expectancy = pd.get_dummies(data=life_expectancy, columns=["Country","Status"], drop_first=True)
2 encoded_life_expectancy
```

Year	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	...	Country_United States of America	Country_Uruguay	Country_Uzbekistan	Country_Vanuatu	Country_Venezuela (Bolivarian Republic of)	Country_Viet Nam	Country_Yemen	Country_Zambia	Country_Zimbabwe	Status_Developing
015.0	65.0	263.0	62.0	0.01	71.279624	65.0	1154.0	19.1	83.0	...	False	False	False	False	False	False	False	False	False	True
014.0	59.9	271.0	64.0	0.01	73.523582	62.0	492.0	18.6	86.0	...	False	False	False	False	False	False	False	False	False	True
013.0	59.9	268.0	66.0	0.01	73.219243	64.0	430.0	18.1	89.0	...	False	False	False	False	False	False	False	False	False	True
012.0	59.5	272.0	69.0	0.01	78.184215	67.0	2787.0	17.6	93.0	...	False	False	False	False	False	False	False	False	False	True
011.0	59.2	275.0	71.0	0.01	7.097109	68.0	3013.0	17.2	97.0	...	False	False	False	False	False	False	False	False	False	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
004.0	44.3	723.0	27.0	4.36	0.000000	68.0	31.0	27.1	42.0	...	False	False	False	False	False	False	False	False	True	True
003.0	44.5	715.0	26.0	4.06	0.000000	7.0	998.0	26.7	41.0	...	False	False	False	False	False	False	False	False	True	True
002.0	44.8	73.0	25.0	4.43	0.000000	73.0	304.0	26.3	40.0	...	False	False	False	False	False	False	False	False	True	True
001.0	45.3	686.0	25.0	1.72	0.000000	76.0	529.0	25.9	39.0	...	False	False	False	False	False	False	False	False	True	True
000.0	46.0	665.0	24.0	1.68	0.000000	79.0	1483.0	25.5	39.0	...	False	False	False	False	False	False	False	False	True	True

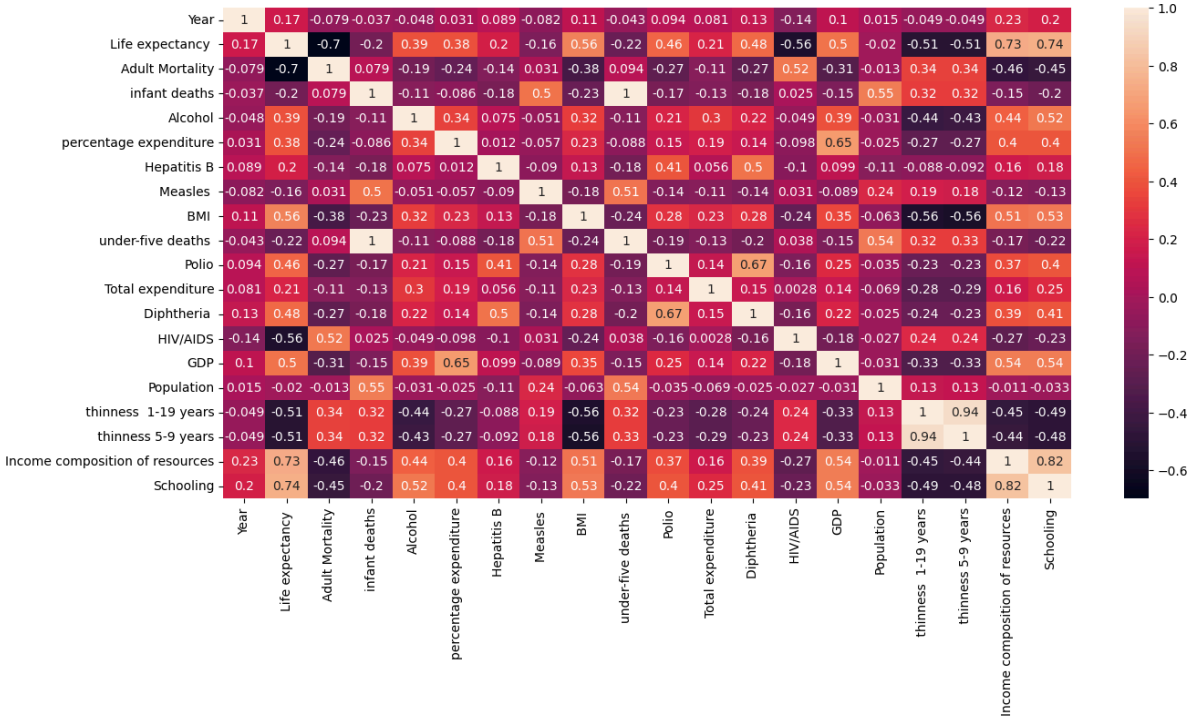
3 × 213 columns

Status_Developing
True
True
True
True
True
...
True
True
True
True
True

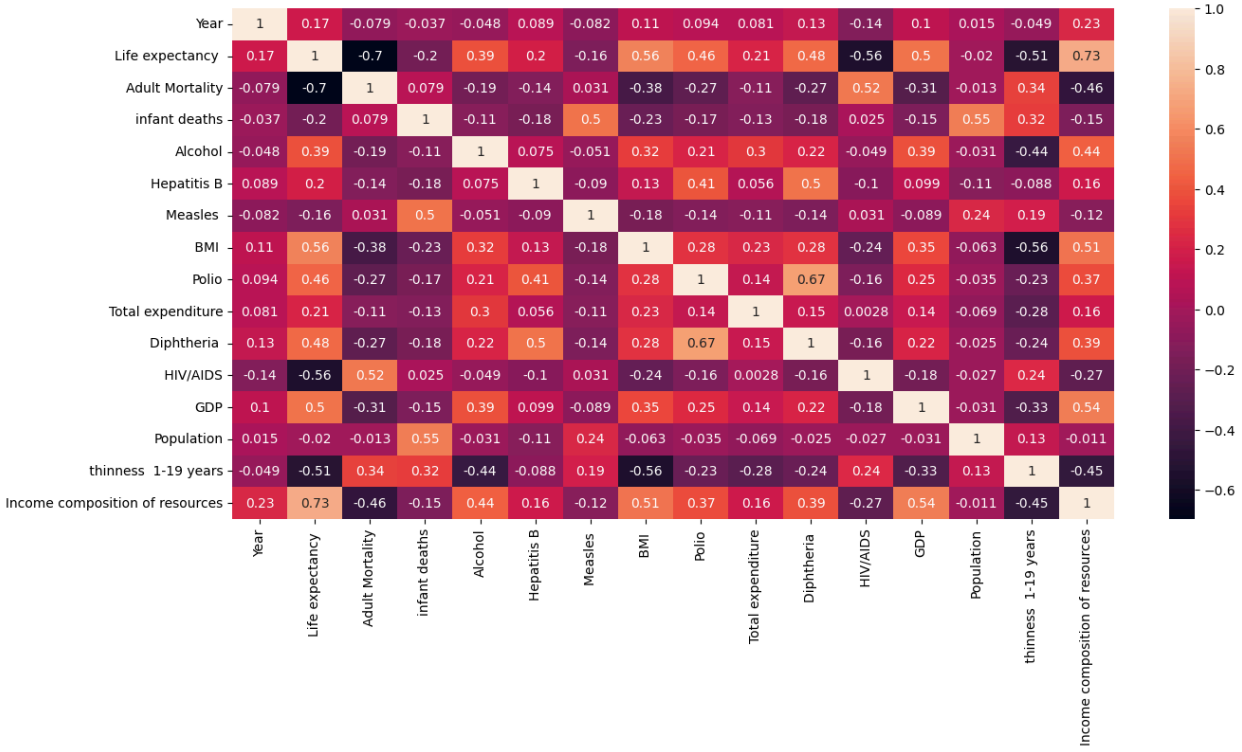
- **Feature selection:**

Using heatmap we have found out that many columns are correlated among themselves by more than 75% (e.g., Under five death and infant death correlates by 100%). So er dropped one (Under-five deaths) of them.

Before dropping columns



After dropping columns that correlate more than 75%



4. Feature scaling:

Our dataset has features with very different ranges:

- GDP: Could be in billions
- Life Expectancy: Between 0-100 years
- Mortality Rates: Usually between 0-1000
- Percentages (like immunization): 0-100%

Before using scaling

```
per-feature minimum before scaling:
 Year                          2000.0
Adult Mortality                   1.0
infant deaths                     0.0
Alcohol                          0.01
percentage expenditure            0.0
                                  ...
Country_Viet Nam                False
Country_Yemen                   False
Country_Zambia                  False
Country_Zimbabwe                False
Status_Developing               False
Length: 212, dtype: object
per-feature maximum before scaling:
 Year                               2015.0
Adult Mortality                      723.0
infant deaths                       1800.0
Alcohol                              17.31
percentage expenditure         19099.04506
                                       ...
Country_Viet Nam                      True
Country_Yemen                         True
Country_Zambia                        True
Country_Zimbabwe                      True
Status_Developing                     True
Length: 212, dtype: object
```

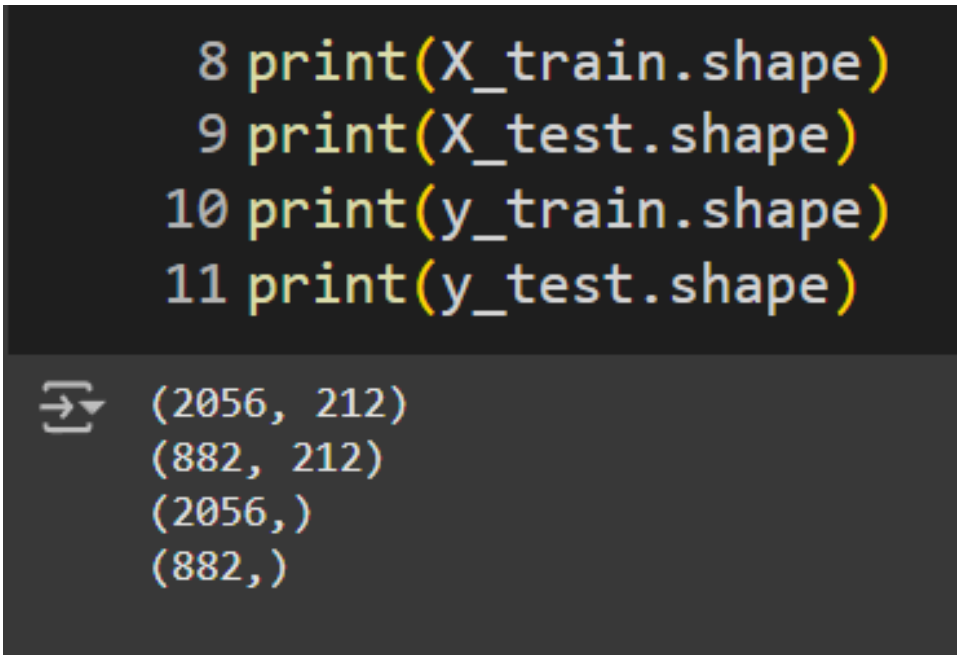
- Some algorithms might not work properly without scaling. So, we have used the min-max scaler to avoid this.

After using min-max scaler

```
per-feature minimum after scaling:
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
per-feature maximum after scaling:
 [1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

5. Dataset splitting:

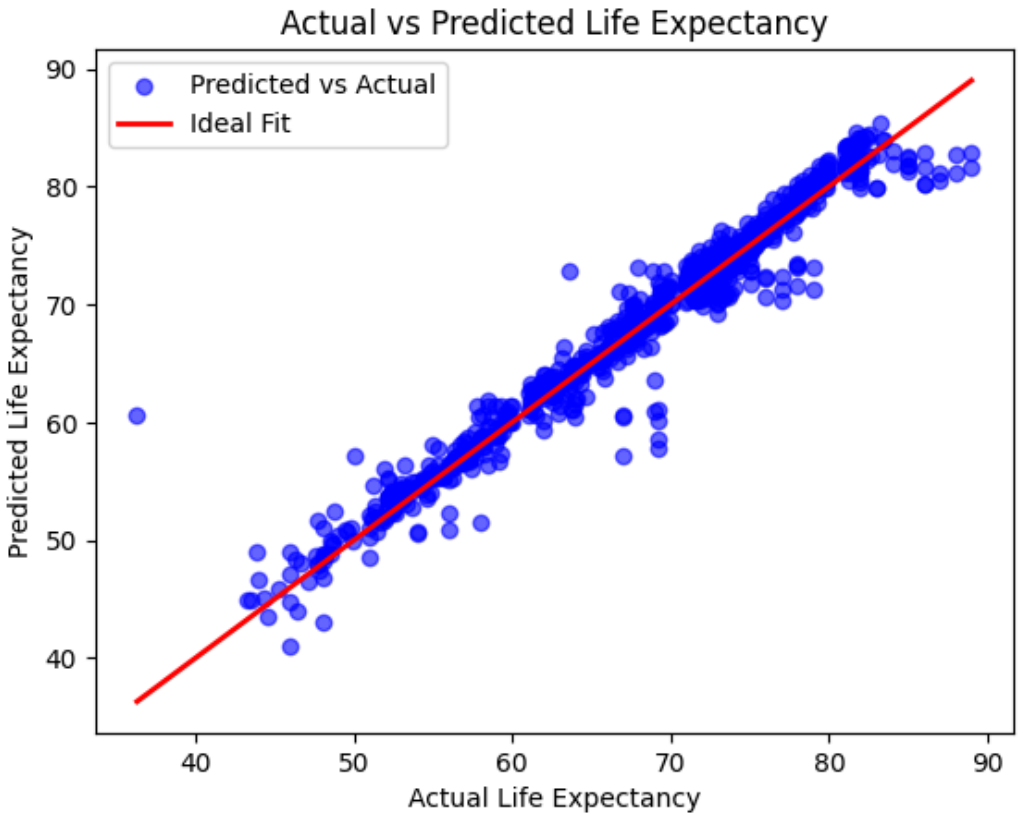
To train the model data splitting was done randomly as this is a regression problem. We have split the dataset into Train set (70%) and Test set (30%).



6. Model Training and Testing:

- **Linear regression:**

  Created a scatter plot that visually compares the model's predictions (`y_pred_lr`) with the true values (`y_test`). This is a way to quickly assess how well the model's predictions align with reality.



Then we calculated the Evaluation metrics:

MAE: Average absolute difference between predicted and actual values

MSE: Average squared difference (penalizes larger errors more)

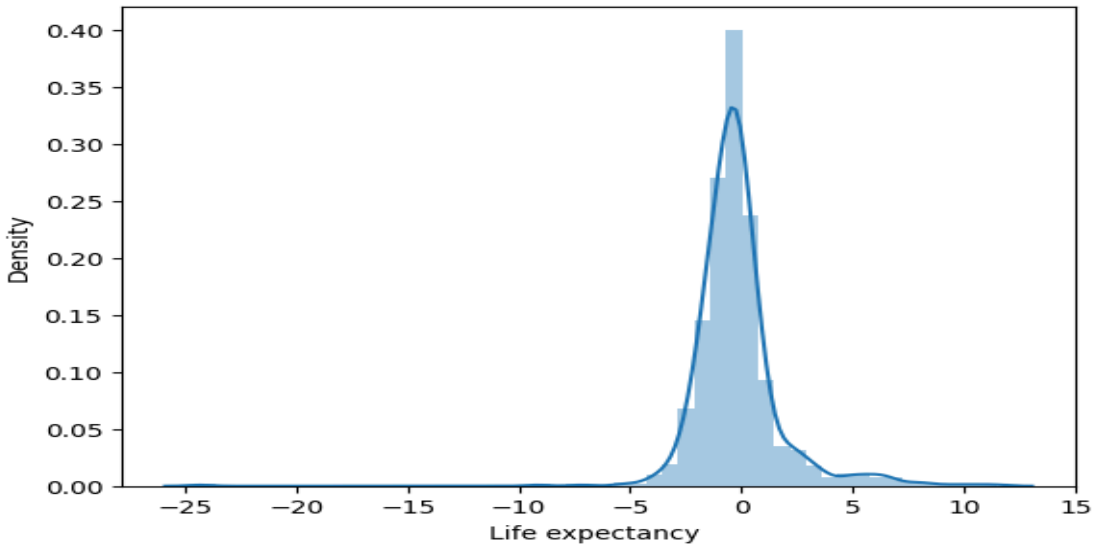
RMSE: The square root of MSE, gives error in the same units as life expectancy

R-squared: How well the model explains the variance in data (1 is perfect)

```
MAE : 1.244478595042193
MSE : 4.157572059468552
RMSE : 2.0390125206747878
R-squared : 0.9555107475290541
```

The numerical metrics help us to get a more precise understanding of how the model is performing, which lets us know how well this regression model is performing.

Analyzing model errors: Residual distribution.



Then we measured the coefficients of the linear regression model to see how each feature impacts life expectancy. The bigger the coefficient, the more that feature impacts the life expectancy.

	Coeffecient
Year	3.838201
Adult Mortality	-0.964366
infant deaths	114.457999
Alcohol	-1.299372
percentage expenditure	2.494415
...	...
Country_Viet Nam	15.212573
Country_Yemen	5.520683
Country_Zambia	-0.295114
Country_Zimbabwe	-1.133478
Status_Developing	-19.246146

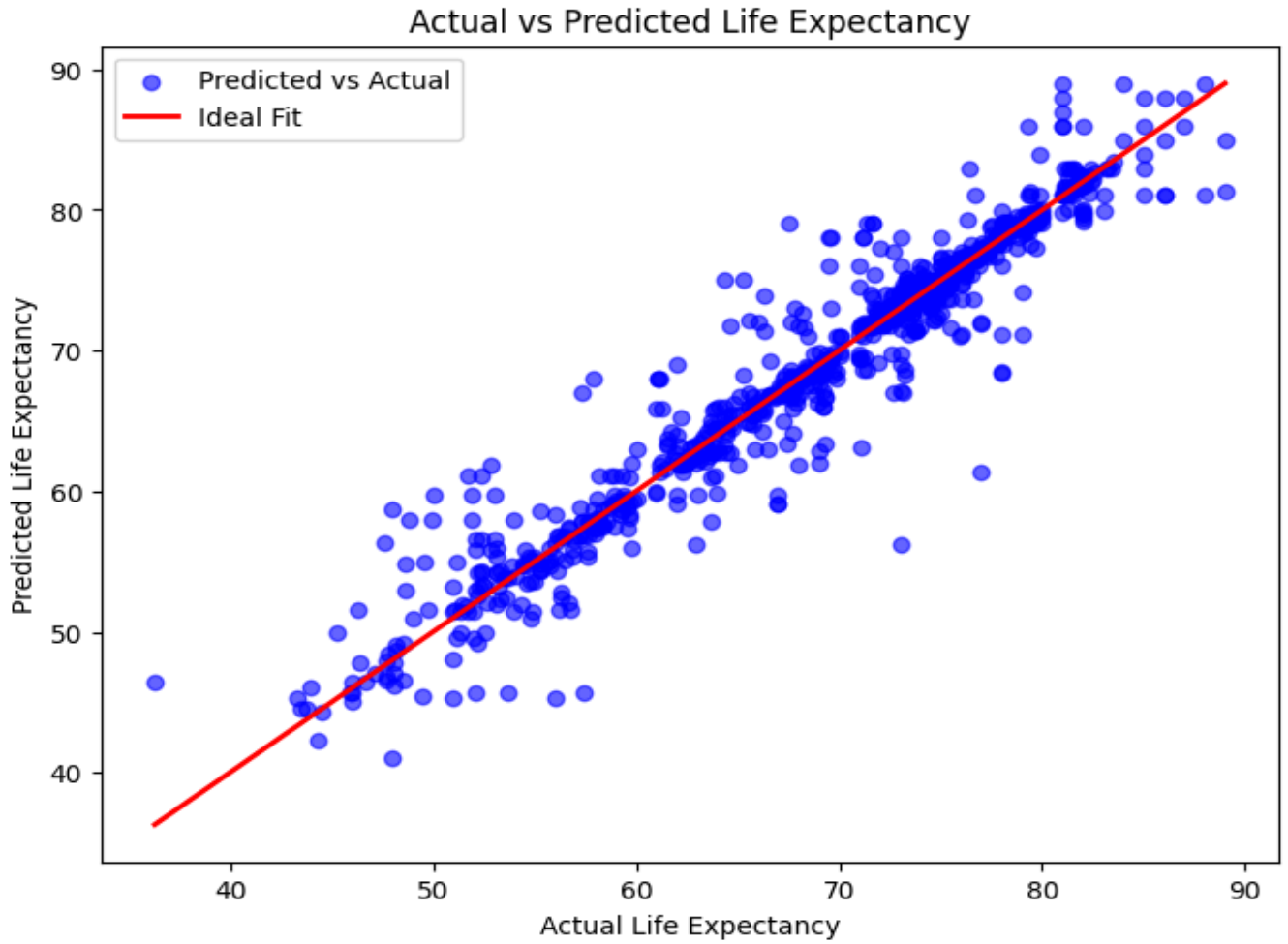
212 rows × 1 columns

- **Decision Tree Regressor Model:**

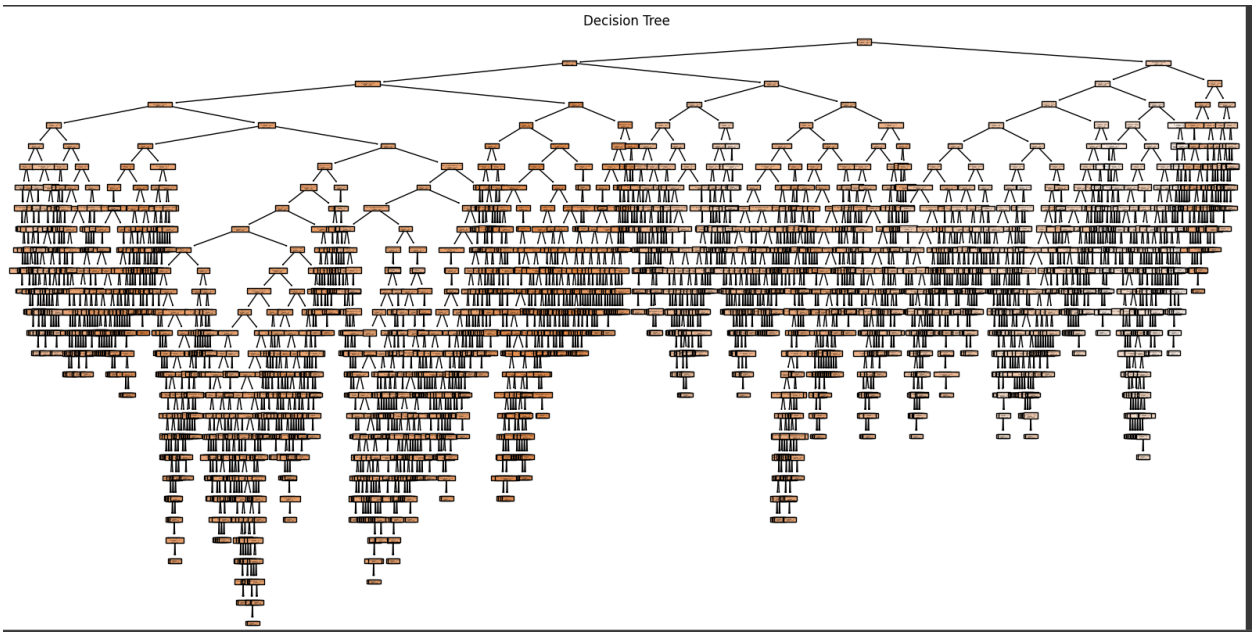
We calculated the common error metrics (MAE, MSE, RMSE, R-squared) to measure how accurate the model's predictions are compared to actual values

```
MAE : 1.5703798960385615
MSE : 7.3649407782503085
RMSE : 2.713842437992727
R-squared : 0.9211894093402614
```

Created a visualization comparing predicted vs actual life expectancy values, with the red line showing perfect predictions and scattered points showing actual model performance



This is a visualization of the decision tree, illustrating the hierarchy of decision nodes and splits based on feature thresholds that guide predictions.



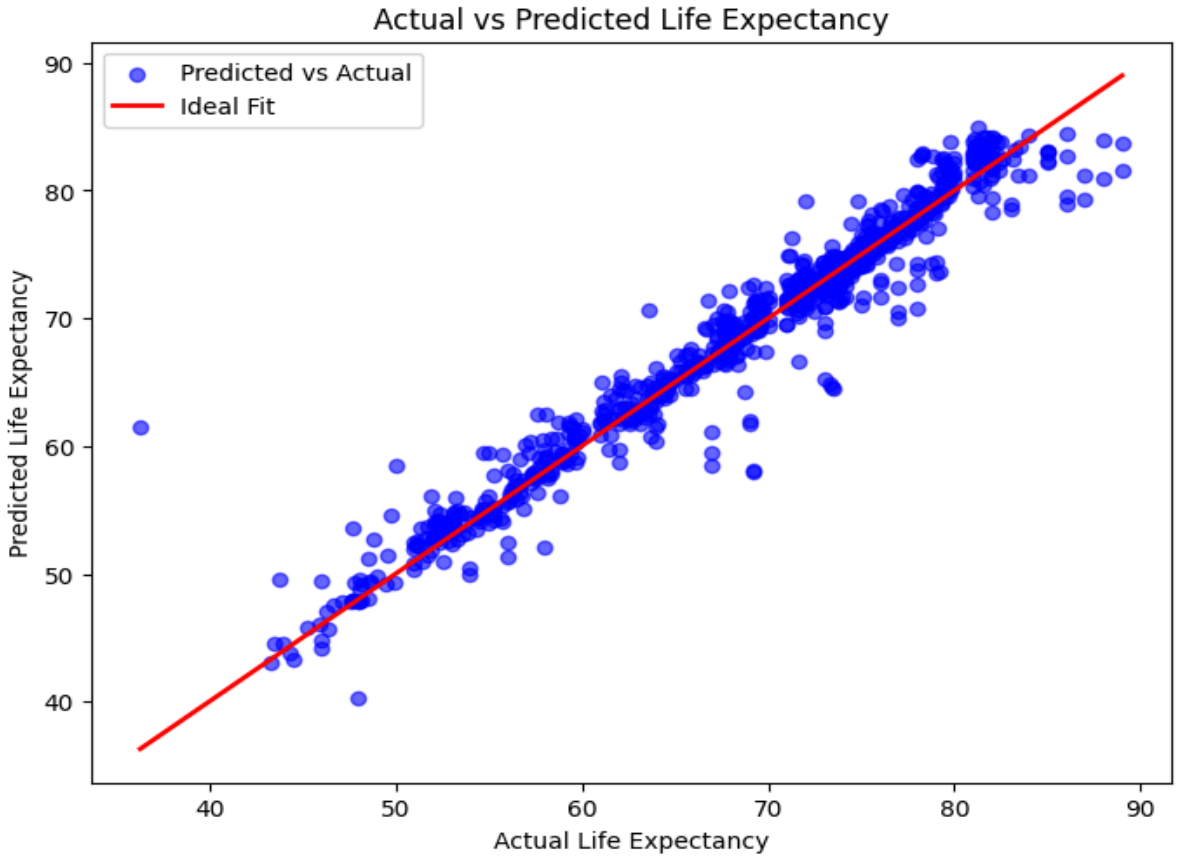
- **Neural Networks:**

  We created a Neural Network with 3 layers (64 neurons, 32 neurons, and 1 output neuron) for predicting life expectancy

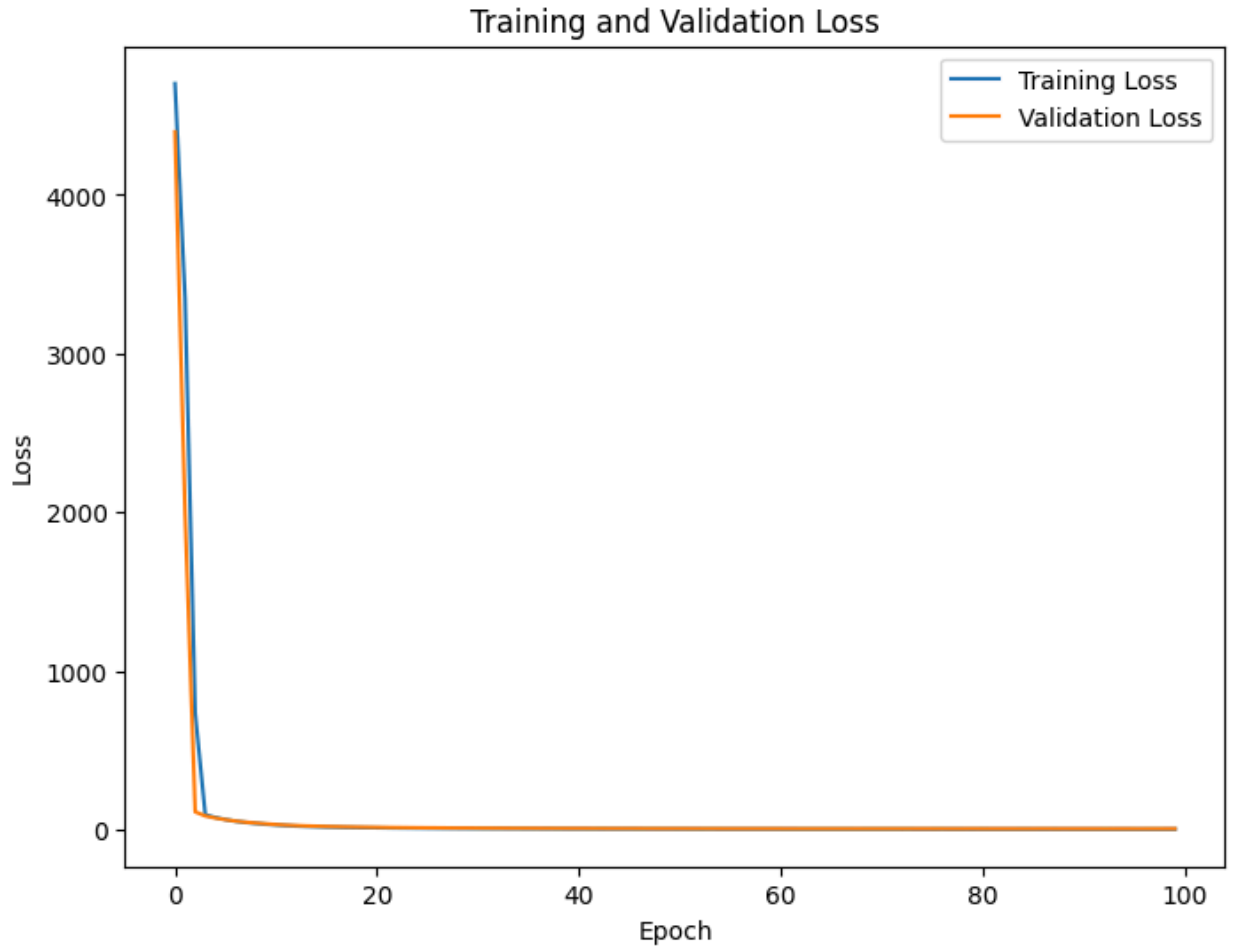
  Trained the model for 100 epochs using the Adam optimizer and mean squared error as the loss function

  Evaluated model performance using standard metrics (MAE, MSE, RMSE, R-squared) and prints the results. And then Created a visualization comparing predicted vs actual life expectancy values.

```
28/28 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
MAE : 1.2979854345321655
MSE : 4.756730564788126
RMSE : 2.1809930226362773
R-squared : 0.9490992858316989
```

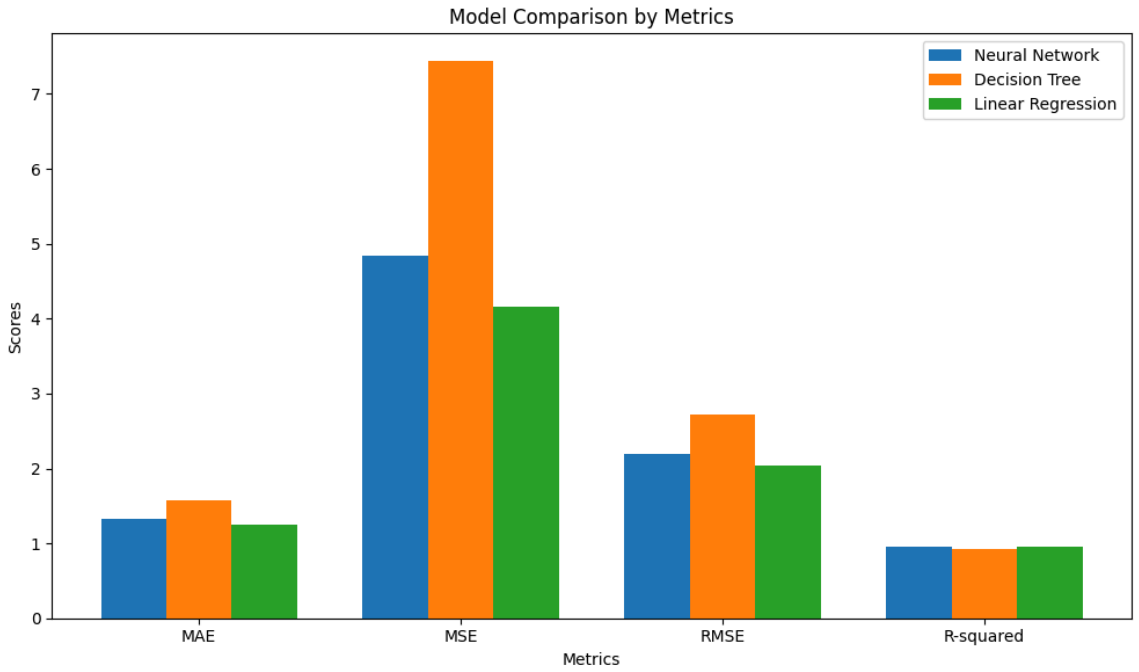


Created a plot showing how the model's loss (error) changes during training over time. Which helps us to identify if the model is overfitting.



7. Comparison Analysis:

We compared three machine learning models (Neural Network, Decision Tree, and Linear Regression) across four metrics: MAE, MSE, RMSE, and R-squared. Which gives us:



Based on the metrics shown:

- Decision Tree has higher error rates (MAE, MSE, RMSE) compared to others
- All models have similar R-squared values (around 1)
- Neural Network and Linear Regression show comparable performance with slightly lower error rates than Decision Tree.

Model Selection:

We have to go with Linear Regression since it:

- Has the lowest error rates (MAE, MSE, RMSE)
- Has the highest R-squared value
- Is simpler to implement and interpret
- Would require less computational resources

Unless our data contains specific non-linear patterns, linear regression appears to be the most practical choice.

8. Conclusion:

In this project, we utilized the "Life Expectancy (WHO)" dataset to develop a predictive model for life expectancy. The process involved several key steps: data preprocessing, feature scaling, and dataset splitting. Subsequently, we applied three Machine Learning models—Linear Regression, Decision Tree Regressor, and Neural Networks—to predict life expectancy.

The performance of these models was evaluated and compared based on relevant metrics. Following the analysis, Linear Regression was identified as the most suitable model for this problem, demonstrating superior performance relative to the other approaches.