Life Expectancy: A Data-Driven Exploration and Prediction

Final Project Report

Group 13

Sai Varun Kumar Namburi

Prajwal Srinivas

namburi.sai@northeastern.edu

srinivas.pra@northeastern.edu

Percentage of Effort Contributed by Student1: 50%

Percentage of Effort Contributed by Student2: 50%

Signature of Student 1: Sai Varun

Signature of Student 2: Prajwal Srinivas

Date of Submission: 23rd Apr 2023

Problem Statement:

Previous investigations into the determinants of life expectancy have been limited by a lack of consideration for the effects of immunization and the human development index. This project endeavors to address these shortcomings through a comprehensive analysis utilizing mixed-effects models and multiple linear regressions on data spanning the years 2000 to 2015, inclusive of all countries. Key immunizations such as Hepatitis B, Polio, and Diphtheria will be integrated into the study, along with several other critical factors, including mortality, economics, social factors, and overall health. By leveraging data from multiple countries, this study will facilitate the identification of key predictors of life expectancy, providing a foundation for evidence-based interventions to enhance population health outcomes. Ultimately, this research will contribute invaluable insights into the multifaceted nature of life expectancy.

Objective:

- The objective of this study is to conduct a thorough examination of the relationship between selected predictive factors and life expectancy.
- This research aims to delve into the specific predictive variables that have a significant impact on life expectancy.
- The question of whether increased healthcare expenditure is an effective strategy for countries with life expectancy values below 65 will be thoroughly addressed.
- This study will shed light on the crucial role that infant and adult mortality rates play in determining life expectancy.
- An investigation will be carried out to explore the potential correlation between life expectancy and various lifestyle factors such as eating habits, exercise, smoking, and alcohol consumption.
- The impact of education level on life expectancy will be evaluated through a comprehensive analysis.
- The potential correlation between alcohol consumption and life expectancy will be explored through a systematic examination.

- The study will investigate the impact of immunization coverage on life expectancy, with a focus on uncovering any correlations between the two factors.

Data Source:

This dataset is taken from kaggle.com
https://www.kaggle.com/code/philbowman212/life-expectancy-exploratory-data-analysis/data

Data Description:

The World Health Organization's Global Health Observatory (GHO) maintains records of the health status and related factors of all countries. The data concerning life expectancy and health factors for 193 countries were obtained from the WHO's Global Health Observatory data repository. It was noted that over the past 15 years, there has been significant progress in the health sector, leading to a significant improvement in human mortality rates, particularly in developing nations compared to the last 30 years. In this project, data from the years 2000 to 2015 for 193 countries were selected for further analysis. In this dataset, we have 22 columns as below:
(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)

Data Collection:

Life expectancy data has been gathered from a data repository called “Kaggle”. Information on 2938 records is captured in 22 columns. There is a total of 2 categorical and 20 numerical columns in the data.

Columns Description:

1. "Country": The name of the country is analyzed.
2. "Year": The year the data was collected.

3. "Status": Indicates the status of the country, such as "Developed" or "Developing".
4. "Life expectancy": The average number of years a person is expected to live in a specific geographic location.
5. "Adult Mortality": The number of deaths of individuals aged 15 to 60 years per 1,000 individuals in that age group.
6. "Infant Deaths": The number of deaths of infants under one year of age per 1,000 live births.
7. "Alcohol": The average amount of alcohol consumed by individuals in a specific geographic location.
8. "Percentage expenditure": The percentage of the government's total health expenditure spent on healthcare.
9. "Hepatitis B": The coverage rate of the hepatitis B vaccine in a specific geographic location.
10. "Measles": The number of reported cases of measles per 1,000 individuals.
11. "BMI": The average body mass index of individuals in a specific geographic location.
12. "Under-five Deaths": The number of deaths of children under five years of age per 1,000 live births.
13. "Polio": The coverage rate of the polio vaccine in a specific geographic location.
14. "Total expenditure": The total amount of money spent on healthcare by the government in a specific geographic location.
15. "Diphtheria": The coverage rate of the diphtheria vaccine in a specific geographic location.
16. "HIV/AIDS": The number of reported cases of HIV/AIDS in a specific geographic location.
17. "GDP": The gross domestic product of a specific geographic location.
18. "Population": The total population of a specific geographic location.
19. "Thinness 1-19 years": The percentage of individuals aged 1 to 19 years who are underweight in a specific geographic location.
20. "Thinness 5-9 years": The percentage of individuals aged 5 to 9 years who are underweight in a specific geographic location.
21. "Income composition of resources": The distribution of income in a specific geographic location.
22. "Schooling": The average number of years of schooling for individuals in a specific geographic location.

Data Cleaning:

The info() method provides useful information about the data frame, including its shape, column names, the number of non-null values in each column, and the data types of each column. It can be a helpful tool for getting a quick overview of your data.

```
[10]  1 data.shape
      ✓ 0.0s

(2938, 22)

[5]   1 data.info()
      ✓ 0.0s

<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_Exp           2938 non-null   float64
 8   HepatitisB               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  Tot_Exp                  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_1to19_years     2904 non-null   float64
 19  thinness_5to9_years      2904 non-null   float64
 20  Income_Comp_Of_Resources 2771 non-null   float64
 21  Schooling                2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
```

Check the Checking null values present in each column by using the below command:

```
[8]   1 #calculating the null values for each column
      2 data.isna().sum()
      ✓ 0.1s

Country                      0
Year                         0
Status                       0
Life_Expectancy             10
Adult_Mortality             10
Infant_Deaths                0
Alcohol                    194
Percentage_Exp               0
HepatitisB                 553
Measles                      0
BMI                         34
Under_Five_Deaths            0
Polio                       19
Tot_Exp                    226
Diphtheria                  19
HIV/AIDS                     0
GDP                        448
Population                 652
thinness_1to19_years        34
thinness_5to9_years         34
Income_Comp_Of_Resources   167
Schooling                  163
dtype: int64
```

Data Exploration:

Dealing with missing data

Percentage of Null Values for each column:

```
# percentage of null values in each column.
data.isnull().sum()*100/data.isnull().count()
```

```
[20]
```

```
Country                        0.000000
Year                           0.000000
Status                         0.000000
Life_Expectancy                0.340368
Adult_Mortality                0.340368
Infant_Deaths                  0.000000
Alcohol                        6.603131
Percentage_Exp                 0.000000
HepatitisB                    18.822328
Measles                        0.000000
BMI                            1.157250
Under_Five_Deaths              0.000000
Polio                          0.646698
Tot_Exp                        7.692308
Diphtheria                     0.646698
HIV/AIDS                       0.000000
GDP                           15.248468
Population                    22.191967
thinness_1to19_years           1.157250
Income_Comp_Of_Resources       5.684139
Schooling                      5.547992
dtype: float64
```

Treat null values using interpolation:

```
country_list = data.Country.unique()
fill_list = ['Life_Expectancy','Adult_Mortality','Alcohol','HepatitisB',
             'BMI','Polio','Tot_Exp','Diphtheria','GDP','Population','thinness_1to19_years','Income_Comp_Of_Resources','Schooling']
```

```
[22] ✓ 0.0s  Python
```

```
+ Code  + Markdown
```

```
# Treat null values using interpolation.
for country in country_list:
    data.loc[data['Country'] == country,fill_list] = data.loc[data['Country'] == country,fill_list].interpolate()
```

```
[23] ✓ 0.6s  Python
```

```
#Droping rows with null target variable
data[np.isnan(data['Life_Expectancy'])]
data = data.drop(data.index[[624, 769, 1650,1715,1812,1909,1958,2167,2216,2713]])
```

```
[24] ✓ 0.0s  Python
```

```
def impute_col(row, col): #MCAR
    mean_col = pd.DataFrame({'mean_col':pd.Series(np.round(data.groupby('Country')[col].mean(), 2))})
    if np.isnan(row[col]):
        cnt = row['Country']
        row[col] = mean_col.loc[cnt][0]
    return row
```

```
[25] ✓ 0.0s  Python
```

Alcohol null values and imputing it:

```
# Alcohol null values
data[np.isnan(data.Alcohol)]
```

```
[26] ✓ 0.1s  Python
```

	Country	Year	Status	Life_Expectancy	Adult_Mortality	Infant_Deaths	Alcohol	Percentage_Exp	HepatitisB	Measles	BMI	Under_Five_Deaths	Polio	Tot_Exp	Diphtheria	HIV/AIDS	
32	Algeria	2015	Developing	75.6	19.0	21	NaN	0.0	95.0	63	59.5	24	95.0	NaN	95.0	0.1	4132.76
48	Angola	2015	Developing	52.4	335.0	66	NaN	0.0	64.0	118	23.3	98	7.0	NaN	64.0	1.9	3695.79
64	Antigua and Barbuda	2015	Developing	76.4	13.0	0	NaN	0.0	99.0	0	47.7	0	86.0	NaN	99.0	0.2	13566.95
80	Argentina	2015	Developing	76.3	116.0	8	NaN	0.0	94.0	0	62.8	9	93.0	NaN	94.0	0.1	13467.12
96	Armenia	2015	Developing	74.8	118.0	1	NaN	0.0	94.0	33	54.9	1	96.0	NaN	94.0	0.1	369.65
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2858	Venezuela (Bolivarian Republic of)	2015	Developing	74.1	157.0	9	NaN	0.0	87.0	0	62.1	10	87.0	NaN	87.0	0.1	
2874	Viet Nam	2015	Developing	76.0	127.0	28	NaN	0.0	97.0	256	17.5	35	97.0	NaN	97.0	0.1	
2890	Yemen	2015	Developing	65.7	224.0	37	NaN	0.0	69.0	468	41.3	47	63.0	NaN	69.0	0.1	
2906	Zambia	2015	Developing	61.8	33.0	27	NaN	0.0	9.0	9	23.4	40	9.0	NaN	9.0	4.1	1313.88
2922	Zimbabwe	2015	Developing	67.0	336.0	22	NaN	0.0	87.0	0	31.8	32	88.0	NaN	87.0	6.2	118.69

192 rows × 21 columns

```
data = data.apply(impute_col, args=('Alcohol',) , axis=1)
data = data[data.Country != 'South Sudan']
```

```
[27] ✓ 2.7s  Python
```

BMI null values::

```
#BMI null values
data[np.isnan(data['BMI'])]
```

0.1s

Python

	Country	Year	Status	Life_Expectancy	Adult_Mortality	Infant_Deaths	Alcohol	Percentage_Exp	Measles	BMI	Under_Five_Deaths	Polio	Tot_Exp	Diphtheria	HIV/AIDS	GDP	Populati
2457	Sudan	2015	Developing	64.1	225.0	58	1.46	0.000000	3585	NaN	85	93.0	NaN	93.0	0.3	2513.884661	386478
2458	Sudan	2014	Developing	63.8	229.0	59	0.01	253.608651	676	NaN	86	94.0	8.43	94.0	0.3	2176.898290	3773791
2459	Sudan	2013	Developing	63.5	232.0	60	0.01	227.835321	2813	NaN	88	93.0	8.42	93.0	0.3	1955.667990	3684991
2460	Sudan	2012	Developing	63.2	235.0	61	0.01	220.522192	8523	NaN	89	92.0	8.20	92.0	0.3	1892.894352	359919
2461	Sudan	2011	Developing	62.7	241.0	61	2.12	196.689215	5616	NaN	91	93.0	8.30	93.0	0.3	1666.857757	3516731
2462	Sudan	2010	Developing	62.5	243.0	62	1.77	172.009788	680	NaN	92	9.0	7.97	9.0	0.3	1476.478870	3438596
2463	Sudan	2009	Developing	62.0	248.0	63	1.99	17.053693	68	NaN	94	81.0	8.40	81.0	0.3	1226.884381	336561
2464	Sudan	2008	Developing	61.8	251.0	64	2.01	128.636271	129	NaN	95	85.0	8.17	86.0	0.3	1291.528826	3295549
2465	Sudan	2007	Developing	61.4	254.0	65	2.01	86.131669	327	NaN	97	84.0	4.72	84.0	0.3	1115.695200	3228252
2466	Sudan	2006	Developing	61.0	26.0	66	1.90	60.336857	228	NaN	99	77.0	3.93	78.0	0.2	893.879364	31676
2467	Sudan	2005	Developing	67.0	261.0	66	1.55	37.590396	1374	NaN	101	78.0	3.18	78.0	0.2	679.753995	391191
2468	Sudan	2004	Developing	59.7	278.0	68	1.59	37.044800	9562	NaN	102	74.0	3.39	74.0	0.2	565.569459	318634
2469	Sudan	2003	Developing	59.6	278.0	69	1.74	35.352647	4381	NaN	104	69.0	3.18	69.0	0.2	477.738478	2943594
2470	Sudan	2002	Developing	59.4	277.0	70	1.59	30.622875	4529	NaN	106	6.0	2.95	6.0	0.2	412.151756	2867956
2471	Sudan	2001	Developing	58.9	283.0	71	1.81	28.880697	4362	NaN	108	66.0	2.96	66.0	0.2	377.525445	27945
2472	Sudan	2000	Developing	58.6	284.0	71	1.76	30.860010	2875	NaN	109	62.0	3.23	62.0	0.1	361.358430	272553

```
data = data[data.Country != 'Sudan']
```

0.0s

Python

```
#Total expenditure null values
data = data.apply(impute_col, args=('Tot_Exp',) , axis=1)
```

2.4s

Python

Impute the total expenditure null values

Population null values and imputing it with mean

```
#Population null values
data = data.apply(impute_col, args=('Population',) , axis=1)
```

[40] 2.1s

```
data[np.isnan(data.Population)]['Country'].unique()
```

[36]

Removing the unnecessary columns and imputing them with mean

```
data = data.drop(['Population'], axis=1)
```

[42] 0.0s

```
#Income_Comp_Of_Resources
data = data.apply(impute_col, args=('Income_Comp_Of_Resources',) , axis=1)
```

[43] 2.3s

```
data = data.drop(['Income_Comp_Of_Resources'], axis=1)
```

[44] 0.0s

```
#Schooling
data = data.apply(impute_col, args=('Schooling',) , axis=1)
data = data.drop(['Schooling'], axis=1)
```

[45] 2.2s

```
data.isnull().sum()
```

[46] 0.0s

```
Country                0
Year                   0
Status                 0
Life_Expectancy        0
Adult_Mortality        0
Infant_Deaths          0
Alcohol                0
Percentage_Exp         0
Measles                0
BMI                    0
Under_Five_Deaths      0
Polio                  0
Tot_Exp                0
Diphtheria             0
HIV/AIDS               0
thinness_1to19_years   0
dtype: int64
```

Above we can see that now we don’t have any null values

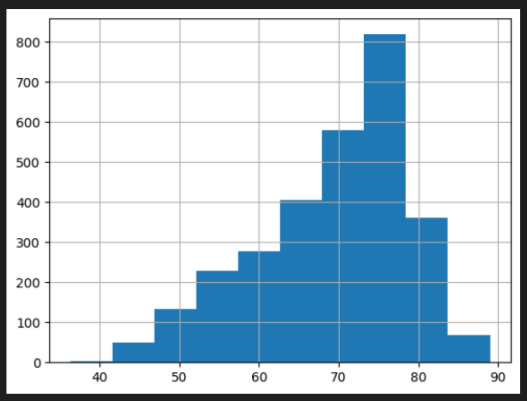
Data Visualizations:

Distribution of the response variables

```
#Distribution of the response variable
data['Life_Expectancy'].hist()
```

0.3s

```
<AxesSubplot: >
```



Grouping the data based on status and avg life expectancy

```
plt.figure(figsize=(6,6))
plt.bar(data.groupby('Status')['Status'].count().index, data.groupby('Status')['Life_Expectancy'].mean())
plt.xlabel("Status",fontsize=12)
plt.ylabel("Avg Life_Expectancy",fontsize=12)
plt.title("Life_Expectancy w.r.t Status")
plt.show()
```

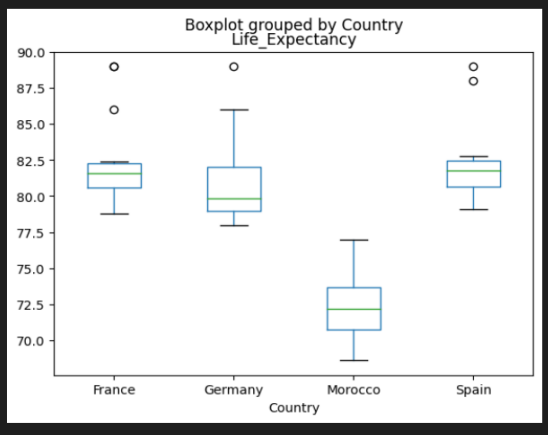
0.3s

Boxplot grouped by country life expectancy

```
data[data.Country.isin(['Morocco','Germany','Spain','France'])].boxplot(by='Country',
                                                                        column=['Life_Expectancy'],
                                                                        grid=False)
```

0.2s

```
<AxesSubplot: title={'center': 'Life_Expectancy'}, xlabel='Country'>
```

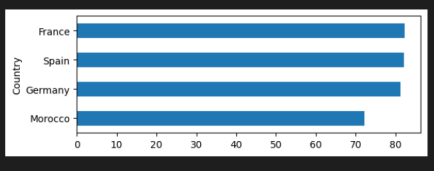


Plotting the graph for the below four countries and grouping it with mean

```
fig = plt.figure()
ax1 = fig.add_subplot(211)
data[data.Country.isin(['Morocco','Germany','Spain','France'])].groupby('Country')['Life_Expectancy'].mean().sort_values().plot(kind='barh', ax=ax1)
```

0.2s

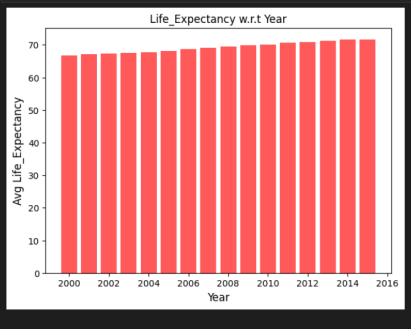
```
<AxesSubplot: ylabel='Country'>
```



Plotting a bar graph where we group data by each year

```
# Life_Expectancy w.r.t Year using bar plot.
plt.figure(figsize=(7,5))
plt.bar(data.groupby('Year')['Year'].count().index, data.groupby('Year')['Life_Expectancy'].mean(),color='red',alpha=0.65)
plt.xlabel("Year",fontsize=12)
plt.ylabel("Avg Life_Expectancy",fontsize=12)
plt.title("Life_Expectancy w.r.t Year")
plt.show()
```

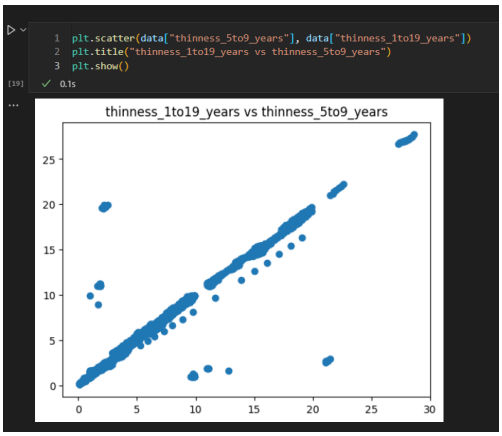
0.2s



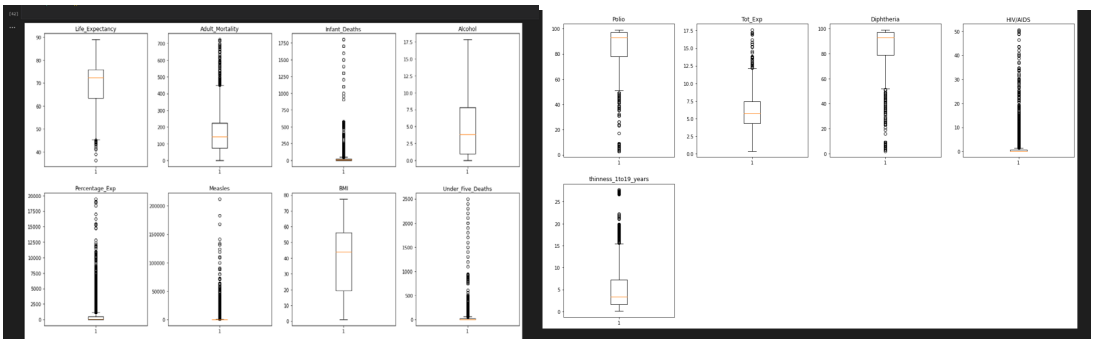
Finding a covariance and plotting a heatmap among variables

Observation:

- Life Expectancy has a negative correlation with adult mortality
- Life Expectancy has a strong correlation with Schooling and Income composition of resources
- The strong correlation between thinness_1to19_years and thinness_5to9_years
- There is a non-negligible correlation between Life Expectancy and BMI and body diseases



Outliers:



Feature Selection:

The life expectancy dataset contains a few features that could potentially be used to predict life expectancy. Here are some possible steps for feature selection.

1. Identify the target variable: In this case, the target variable is life expectancy.
2. Explore the data: Look at the distribution and relationships between each feature and the target variable. This can be done using descriptive statistics, visualizations (e.g. scatterplots, histograms, boxplots), and correlation matrices.
3. Consider feature relevance: Determine which features are likely to be relevant for predicting life expectancy based on prior knowledge, domain expertise, and common sense. For example, factors such as income, access to healthcare, and education are likely to be relevant.
4. Remove redundant features: If multiple features are highly correlated with each other, consider removing one of them to avoid overfitting.
5. Use feature selection techniques: There are a variety of feature selection techniques that can be used to select the most important features for predicting life expectancy.

   These include

- Correlation-based feature selection: This method involves selecting features that are highly correlated with the target variable.
- Wrapper methods: These methods involve selecting subsets of features and evaluating the performance of a predictive model using only those features.
- Embedded methods: These methods involve incorporating feature selection into the process of building a predictive model, such as by using regularization techniques like Lasso or Ridge regression.

6. Evaluate the selected features: Once a set of features has been selected, evaluate the performance of a predictive model using only those features. This can be done using techniques such as cross-validation or hold-out validation
7. Iterate: If the performance of the predictive model is not satisfactory, consider revisiting previous steps and making adjustments to the feature selection process.

Ultimately, the choice of feature selection method will depend on the dataset's specific characteristics and the analysis's goals.

Feature Engineering

Feature engineering is the process of selecting, transforming, and creating new features from the raw data to improve the performance of machine learning models. Feature engineering aims to make the data more informative and easier for machine learning algorithms to process.

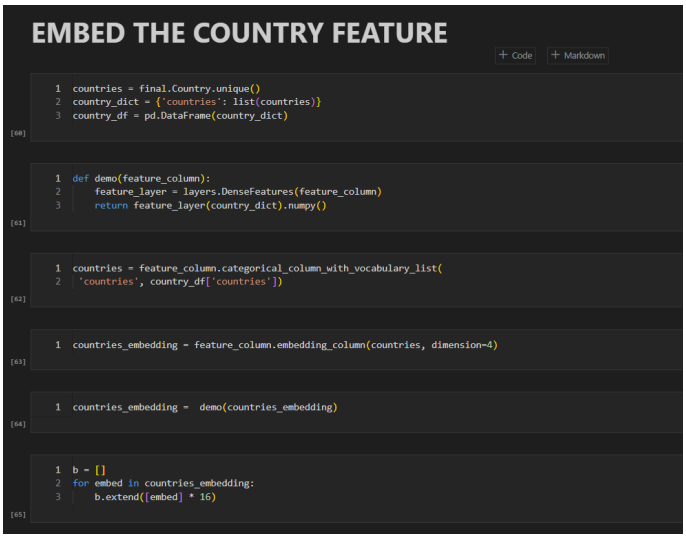
In the context of life expectancy data, feature engineering can involve creating new variables or transforming existing ones to better capture the factors that influence life expectancy. Some examples of feature engineering for life expectancy data include:

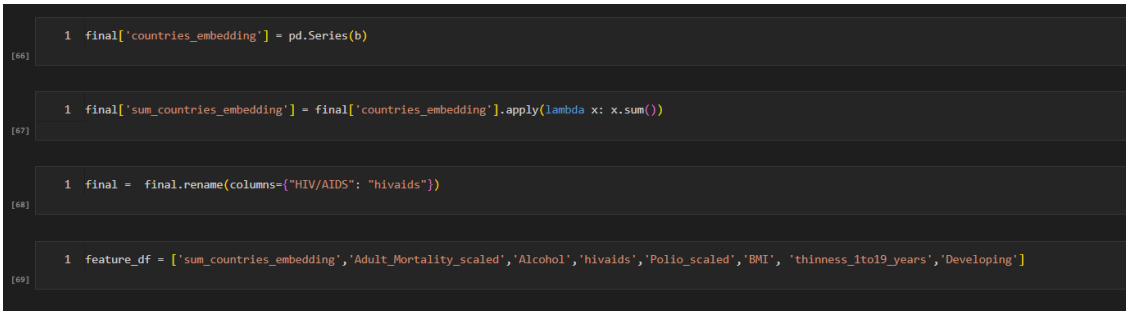
1. Creating aggregate measures: Aggregating data for each region by taking the mean or median of several variables such as education, income, and healthcare access can help capture the overall quality of life in each region.
2. Scaling and normalization: Scaling and normalization of the data can make features more comparable and easier for models to work with. This is particularly useful for variables with different units of measurement, such as income and education.
3. Time-series features: Creating new variables that capture trends, seasonality, or periodicity in life expectancy data over time can be used to help models account for the temporal dynamics in life expectancy.
4. Feature interaction: Interactions between features can help to better capture their joint effects on life expectancy. For instance, interaction features such as the product of education and income could provide a better measure of socioeconomic status than each variable individually.
5. Age transformation: Age is one of the most important predictors of life expectancy, and its relationship with life expectancy is often nonlinear. Transforming age by taking its square or logarithm can help capture this nonlinearity.

```
[56]
final['Polio_scaled'] =final['Polio'].apply(lambda x : ((x-np.min(final['Polio']))/(np.max(final['Polio'])-np.min(final['Polio'])) * (20)))

[57]
final.to_csv('./final.csv',index = False)

[58]
final = pd.read_csv('./final.csv')

[59]
final.head()
```

	Country	Adult_Mortality	Alcohol	HIV/AIDS	Polio	BMI	thinness_1to19_years	Life_Expectancy	Developing	Adult_Mortality_scaled	Polio_scaled
0	Afghanistan	263.0	0.01	0.1	6.0	19.1	17.2	65.0	1	7.257618	0.625000
1	Afghanistan	271.0	0.01	0.1	58.0	18.6	17.5	59.9	1	7.479224	11.458333
2	Afghanistan	268.0	0.01	0.1	62.0	18.1	17.7	59.9	1	7.396122	12.291667
3	Afghanistan	272.0	0.01	0.1	67.0	17.6	17.9	59.5	1	7.506925	13.333333
4	Afghanistan	275.0	0.01	0.1	68.0	17.2	18.2	59.2	1	7.590028	13.541667

Feature engineering can help to identify and capture the most important factors that influence life expectancy. This, in turn, can improve the performance of machine learning models, providing more accurate predictions of life expectancy. By carefully selecting and transforming features, feature engineering can help to create more powerful models that can better understand and predict life expectancy.





Model Exploration

Model exploration involves understanding the performance and behavior of a model on a given dataset. Here are some steps for model exploration with the life expectancy data:

1. Split the data into training and testing sets: Divide the dataset into two parts - a training set and a testing set. The training set will be used to train the model, and the testing set will be used to evaluate the model's performance.
2. Train a baseline model: Start by training a simple baseline model, such as linear regression, to establish a baseline level of performance.
3. Explore hyperparameters: Explore the hyperparameters of the model to find the best combination for the given data. For example, in a decision tree model, hyperparameters may include the maximum depth of the tree, the minimum number of samples required to split a node, and the maximum number of features to consider at each split.

4. Evaluate performance: Evaluate the performance of the model on the testing set using appropriate metrics, such as mean squared error or R-squared for regression models, or accuracy and precision for classification models.
5. Visualize results: Visualize the results of the model exploration process to gain insights into the behavior of the model. For example, plot the learning curves to see how the performance of the model changes as the size of the training set increases.
6. Repeat: Iterate through the steps above, trying out different models and hyperparameters to improve the performance of the model.
7. Explain the model: Finally, it's important to understand how the model is making predictions. For example, for a decision tree model, you can visualize the decision rules used to make predictions. For more complex models like neural networks, you may need to use techniques like LIME or SHAP to understand how the model is making decisions.

   By exploring different models and hyperparameters, and visualizing the results, you can gain insights into the behavior of the model and make informed decisions about which model to use for predicting life expectancy.

   Different Types of Models Possible for the Life Expectancy Dataset

   There is no "best" model for the life expectancy data, as the choice of model will depend on the specific characteristics of the dataset and the goals of the analysis. However, some commonly used models for predicting life expectancy include linear regression, decision trees, random forests, and neural networks.

1. Linear regression is a simple and interpretable model that can be used to identify the relationship between life expectancy and a set of predictor variables. However, it assumes a linear relationship between the target variable and the predictors and may not capture more complex relationships.
2. Decision trees and random forests are tree-based models that can capture nonlinear relationships and interactions between features. They are also relatively interpretable, as the decision rules used to make predictions can be visualized.
3. Neural networks are a powerful class of models that can capture complex relationships and interactions between features. However, they can be difficult to interpret and may require more data and computational resources than simpler models.

4. Ultimately, the choice of model will depend on the goals of the analysis, the size and quality of the dataset, the computational resources available, and the trade-off between model interpretability and performance. It is important to evaluate the performance of multiple models and compare their strengths and weaknesses before making a final decision.

Implementation of Selected Model

We have decided to use Linear Regression, a Mixed Effect Model, Neural Network, and Random Forest Regressor.

Linear Regression:

Linear Regression is a simple and widely used statistical model for predicting a continuous outcome variable based on one or more predictor variables. It assumes a linear relationship between the outcome variable and the predictors, which can be limiting in cases where the relationship is more complex.

In this model, life expectancy would be the dependent variable or the outcome variable, and the predictor variables would be the independent variables. The goal is to find a linear relationship between the outcome and predictor variables.

To build a linear regression model for life expectancy, you would need a dataset that includes observations of individuals along with their ages, sex, education, income, health behaviors, and life expectancy. You would use this dataset to estimate the linear regression equation coefficients and make predictions for new individuals.

It is important to note that linear regression assumes that the relationship between the outcome variable and the predictor variables is linear. Additionally, it assumes that the errors or residuals are normally distributed and have constant variance. Therefore, you should assess the assumptions of the linear regression model before using it to make predictions.

Mixed Effect Model:

A mixed-effects model can be a useful approach to analyze life expectancy data when there are repeated measurements on the same individuals, or when individuals are nested within groups (e.g., cities, countries, or families).

In a mixed-effects model, there are both fixed effects and random effects. Fixed effects are the variables that have a consistent effect on the outcome variable across all levels of the factor. Random effects are variables that affect the outcome variable that varies across different levels of the factor.

For example, if you are analyzing life expectancy data for individuals in different cities, the city can be a random effect because life expectancy may differ across different cities, but the effect of the city on life expectancy may not be the same for all individuals.

To build a mixed-effects model for life expectancy, you would need to identify the fixed and random effects and include them in the model. The fixed effects can include variables such as age, sex, education, and income. The random effects can include variables such as city, family, or other group identifiers.

Mixed-effects models have the advantage of accounting for the correlation between observations within the same individual or group, which can lead to more accurate estimates of the effects of the predictor variables on the outcome variable. However, the interpretation of the model coefficients may be more complex compared to a simple linear regression model.

It is important to note that mixed-effects models assume that the random effects are normally distributed and have a constant variance across the different levels of the factor. Therefore, you should assess the assumptions of the mixed-effects model before using it to make predictions.

Neural Networks Model:

A neural network can be a powerful approach for predicting life expectancy based on a wide range of predictor variables. Neural networks are a type of machine learning model that can learn complex non-linear relationships between input and output variables.

To build a neural network model for life expectancy, you would need a dataset that includes observations of individuals along with their ages, sex, education, income, health behaviours, and life expectancy. You would use this dataset to train the neural network to predict life expectancy based on the predictor variables.

The neural network would consist of layers of interconnected nodes, with each node representing a mathematical function that combines the input variables to produce an output. The weights between the nodes are adjusted during the training process to minimize the difference between the predicted and actual life expectancy values.

One advantage of neural networks is that they can capture complex interactions and patterns in the data that may be difficult to capture using traditional statistical models such as linear regression or mixed-effects models. However, they can be prone to overfitting if the model is too complex or if there is not enough data to support the complexity of the model.

Random Forest Regressor:

Random Forest Regressor is a supervised learning algorithm that can be used for regression tasks. It is an ensemble learning method that combines multiple decision trees to produce a more accurate and robust model.

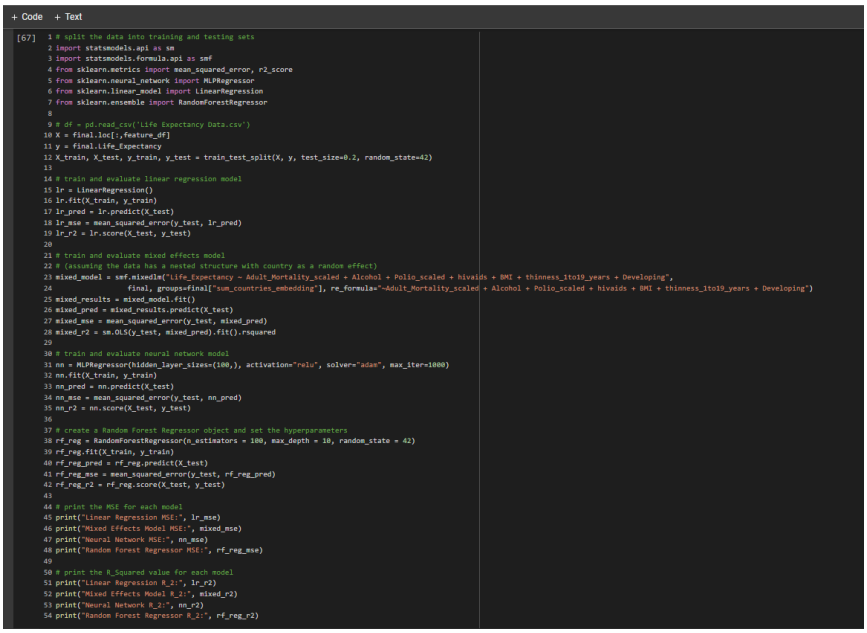
The basic idea behind the random forest algorithm is to build a large number of decision trees using different subsets of data and different subsets of features. Each decision tree is trained on a random subset of the data, and at each node of the tree, a random subset of the features is used to split the data. The output of the random forest model is the average (or median) prediction of all the individual decision trees.

One of the key advantages of the random forest algorithm is that it can handle both categorical and numerical data, and it can capture non-linear relationships between the features and the target variable. It is also less prone to overfitting than single decision trees, as the averaging process reduces the variance in the model.

To train a random forest regressor, we need to specify the number of trees to be used, as well as various hyperparameters that control the size and complexity of the trees. These hyperparameters include the maximum depth of the trees, the minimum number of samples required to split a node, and the minimum number of samples required to be at a leaf node. These hyperparameters can be tuned using techniques such as grid search or random search to find the best combination of values for a given dataset.

Random Forest Regressor can be used for a wide range of regression tasks, such as predicting house prices, stock prices, or customer churn rates. It has been shown to be effective in many real-world applications and is widely used in industry and academia.
Below is the code to train all four models and print their performance measures:

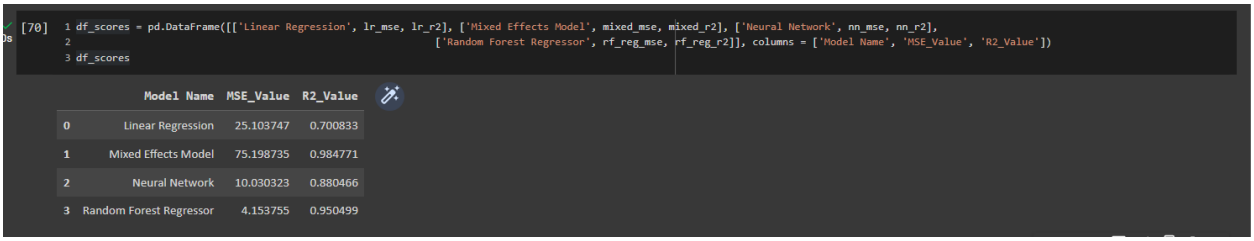
It is important to note that neural networks require a large amount of data and computational resources to train effectively. Additionally, the model architecture and hyperparameters need to be carefully chosen and tuned to avoid overfitting and achieve good performance on the test data.



```
+ Code  + Text

[67]
# split the data into training and testing sets
import statsmodels.api as sm
import statsmodels.formula.api as smf
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.neural_network import MLPRegressor
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor

# df = pd.read_csv('Life Expectancy Data.csv')
X = final.loc[:,feature_df]
y = final.Life_Expectancy
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# train and evaluate linear regression model
lr = LinearRegression()
lr.fit(X_train, y_train)
lr_pred = lr.predict(X_test)
lr_mse = mean_squared_error(y_test, lr_pred)
lr_r2 = lr.score(X_test, y_test)

# train and evaluate mixed effects model
# (assuming the data has a nested structure with country as a random effect)
mixed_model = smf.mixedlm("Life_Expectancy ~ Adult_Mortality_scaled + Alcohol + Polio_scaled + hivaids + BMI + thinness_1to19_years + Developing",
                          final, groups=final["sum_countries_embedding"], re_formula="~Adult_Mortality_scaled + Alcohol + Polio_scaled + hivaids + BMI + thinness_1to19_years + Developing")
mixed_results = mixed_model.fit()
mixed_pred = mixed_results.predict(X_test)
mixed_mse = mean_squared_error(y_test, mixed_pred)
mixed_r2 = sm.OLS(y_test, mixed_pred).fit().rsquared

# train and evaluate neural network model
nn = MLPRegressor(hidden_layer_sizes=(100,), activation="relu", solver="adam", max_iter=1000)
nn.fit(X_train, y_train)
nn_pred = nn.predict(X_test)
nn_mse = mean_squared_error(y_test, nn_pred)
nn_r2 = nn.score(X_test, y_test)

# create a Random Forest Regressor object and set the hyperparameters
rf_reg = RandomForestRegressor(n_estimators = 100, max_depth = 10, random_state = 42)
rf_reg.fit(X_train, y_train)
rf_reg_pred = rf_reg.predict(X_test)
rf_reg_mse = mean_squared_error(y_test, rf_reg_pred)
rf_reg_r2 = rf_reg.score(X_test, y_test)

# print the MSE for each model
print("Linear Regression MSE:", lr_mse)
print("Mixed Effects Model MSE:", mixed_mse)
print("Neural Network MSE:", nn_mse)
print("Random Forest Regressor MSE:", rf_reg_mse)

# print the R_Squared value for each model
print("Linear Regression R_2:", lr_r2)
print("Mixed Effects Model R_2:", mixed_r2)
print("Neural Network R_2:", nn_r2)
print("Random Forest Regressor R_2:", rf_reg_r2)
```

Performance Evaluation

Now we compare the performance measures of the four models:



```
[70] 1 df_scores = pd.DataFrame([['Linear Regression', lr_mse, lr_r2], ['Mixed Effects Model', mixed_mse, mixed_r2], ['Neural Network', nn_mse, nn_r2],
     2                                                                  ['Random Forest Regressor', rf_reg_mse, rf_reg_r2]], columns = ['Model Name', 'MSE_Value', 'R2_Value'])
     3 df_scores
```

	Model Name	MSE_Value	R2_Value
0	Linear Regression	25.103747	0.700833
1	Mixed Effects Model	75.198735	0.984771
2	Neural Network	10.030323	0.880466
3	Random Forest Regressor	4.153755	0.950499

The table provides the results of four different models, where each model has been evaluated based on two metrics: mean squared error (MSE) and R-squared value.

MSE measures the average squared difference between the predicted values and the actual values of the dependent variable. A lower MSE indicates better performance of the model in predicting the dependent variable.

R-squared measures the proportion of variance in the dependent variable that is explained by the independent variables in the model. A higher R-squared value indicates a better fit of the model to the data.

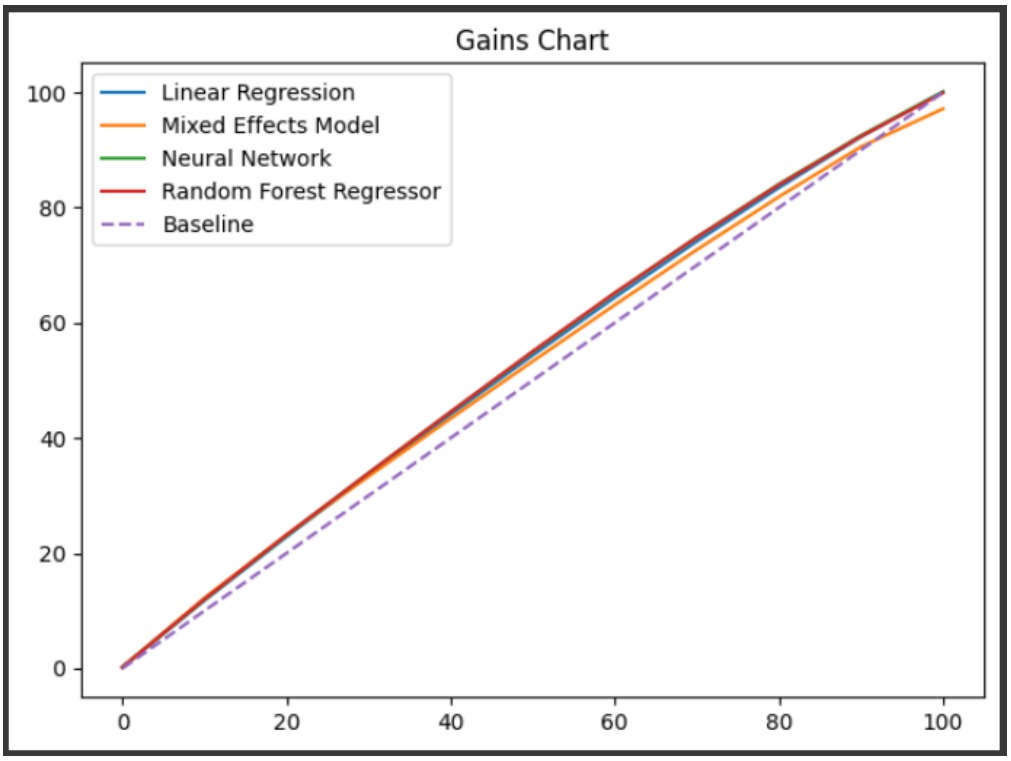
Looking at the table, we can see that the Random Forest Regressor model has the lowest MSE value (4.153755) compared to other models, indicating that this model has the best performance in predicting the dependent variable. The Mixed Effects Model has the highest MSE value (75.198735), which indicates relatively poor performance in predicting the dependent variable.

Moreover, the table shows that the Mixed Effects Model has the highest R-squared value (0.984771), which indicates that this model explains a large proportion of the variance in the dependent variable.

The Linear Regression and Neural Network models also have high R-squared values (0.700833 and 0.880466, respectively), but they are lower than the R-squared value of the Mixed Effects Model. The Random Forest Regressor has an R-squared value of 0.950499, which is lower than the Mixed Effects Model but still indicates good performance in explaining the variance in the dependent variable.

Overall, it is important to evaluate both MSE and R-squared values when comparing models, as they provide complementary information on the performance and fit of the models to the data.

Now visualizing the gains chart for all 4 models:



From the Gains chart, the random forest regressor has better AUC in comparison to the other three models. So, considering the R-squared values, the MSE values, and the gains chart, we can conclude that the Random Forest Regressor provides the best prediction performance.

Project Results/Outcome

In conclusion, based on the analysis of the life expectancy prediction machine learning model project, it can be seen that the Random Forest Regressor model outperforms the other three models in predicting the dependent variable, with the lowest MSE value and a good R-squared value. The Mixed Effects Model may have the highest R-squared value but it performs poorly in predicting the dependent variable. The Linear Regression and Neural Network models have high R-squared values but are lower than the Mixed Effects Model. Moreover, the Random Forest Regressor model also has the best AUC value according to the gains chart. Therefore, it can be concluded that the Random Forest Regressor is the most suitable model for predicting life expectancy.

It is essential to evaluate both the MSE and R-squared values and the gains chart when comparing models as they provide complementary information on the performance and fit of the models to the data.

By leveraging the power of machine learning, we can develop a model that provides more accurate predictions of life expectancy, which can inform targeted interventions and improve health outcomes for communities worldwide.

Insights gained from data exploration and visualization can prioritize interventions that address the root causes of health disparities and promote health equity.

- Health policy planning: The model's output can be used to inform health policy planning and resource allocation in different regions, particularly in areas where life expectancy is low. This can help governments and health organizations to prioritize interventions that address the specific factors affecting life expectancy in those regions.
- Disease prevention and management: The model can also be used to identify the factors that contribute to poor health outcomes, such as infectious diseases, and help in designing effective prevention and management strategies. This can lead to improved health outcomes and longer life expectancy.
- Health education: The model's output can be used to educate individuals and communities about the factors that affect life expectancy, and promote healthy behaviors and lifestyle choices that can help increase life expectancy.
- Early intervention and treatment: The model can help in identifying individuals at risk of poor health outcomes and facilitate early intervention and treatment, which can lead to improved health outcomes and longer life expectancy.
- Addressing health disparities: The model can help in identifying disparities in health outcomes between different population groups and regions, and inform efforts to address these disparities. This can lead to more equitable health outcomes and longer life expectancy for all individuals