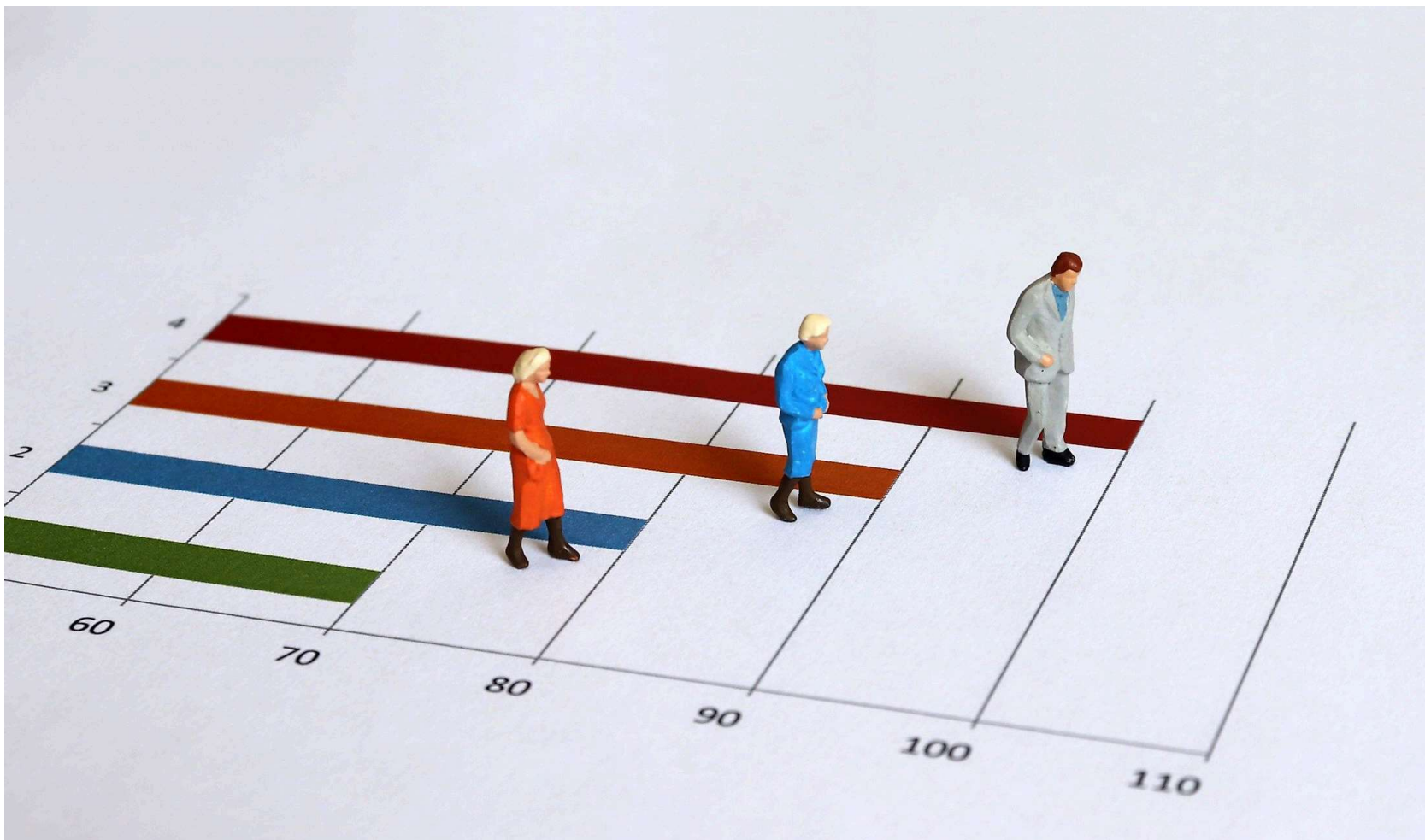
LIFE EXPECTANCY ANALYSIS AND PREDICTION



Life expectancy

Life expectancy is the average time a human expected to live based on various factors.

=> Developing a model that can accurately predict life expectancy based on historical data and trends.

=> Identifying the key factors that contribute to life expectancy, such as lifestyle choices, medical history, and environmental factors.

=> Building a tool that can be used by healthcare providers to identify patients who are at risk of developing life-threatening diseases.

=> Creating a model that can be used by insurance companies to develop more accurate pricing models and reduce the risk of losses due to unexpected deaths.

=> Informing public policy decisions related to healthcare, retirement, and social security.

Platform used - GOOGLE COLAB

Application

Life Expectancy is the essential factor in deciding a person's risk factor and the probability they will make a case. Insurance agencies think about age, way of life decisions, and a few different components while deciding premium rates for singular life coverage strategies. It very well may be utilized by specialists to make important inquires about out of it and along these lines, realize something that will help increment the hope think about the effect of a particular factor on the normal life expectancy of individuals in a particular nation.

Data Preprocessing:

- Missing values were imputed using the mean.
- Outliers were handled using robust scaling.
- Label encoding was used to convert categorical variables to numerical values.

Exploratory Data Analysis:

- The distribution of life expectancy was visualized using a histogram.
- A heatmap was used to explore the correlations between different variables.
- Scatter plots were used to analyze the relationships between GDP and life expectancy, and between BMI and life expectancy.

Model Creation and Evaluation:

- Three models were used to predict life expectancy: linear regression, random forest regressor, and SVR.
- The models were evaluated using mean squared error (MSE) and R^2 score.
- Random forest regressor achieved the best performance with an MSE of 0.12 and an R^2 score of 0.95.

Hyperparameter tuning was performed on the random forest model to further improve its performance

Description of the columns:

LifeExp: Life expectancy at birth (in years)

Status: Development status of the country (Developed or Developing)

Adult_Mortality: Adult mortality rate (per 1000 population)

Infant_Deaths: Number of infant deaths per 1000 live births

Alcohol: Alcohol consumption per capita (in liters of pure alcohol)

Percentage_expenditure: Expenditure on health as a percentage of GDP

Hepatitis_B: Hepatitis B immunization coverage among 1-year-olds (%) Measles: Number of reported measles cases per 1000 population

BMI: Average body mass index (kg/m^2)

Under_five_deaths: Number of deaths under the age of 5 per 1000 live births

Polio: Polio immunization coverage among 1-year-olds (%)

Total_expenditure: Total health expenditure as a percentage of GDP

Diphtheria: Diphtheria immunization coverage among 1-year-olds (%)

HIV_AIDS: Deaths from HIV/AIDS per 1000 population

GDP: Gross domestic product per capita (in US dollars)

Population: Population of the country

Thinness_1-19_years: Prevalence of thinness among children and adolescents aged 1-19 (%)

Thinness_5-9_years: Prevalence of thinness among children aged 5-9 (%)

```
# ! pip install --upgrade
```

```
# ! pip install --upgrade imbalanced-learn scikit-learn
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder, StandardScaler,RobustScaler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score
```

```
df=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Life Expectancy Data.csv")
df.head()
```

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	Polio	Total expenditure	Diphtheria	HIV/AIDS	G
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	6.0	8.16	65.0	0.1	584.2592
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	58.0	8.18	62.0	0.1	612.6965
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	62.0	8.13	64.0	0.1	631.7449
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	67.0	8.52	67.0	0.1	669.9590
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	68.0	7.87	68.0	0.1	63.5372

5 rows × 22 columns

```
# from google.colab import drive
# drive.mount('/content/drive')
```

```
df.info()
```

```
<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
```

```
df.describe()
```

	Year	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio	Tota expenditur
count	2938.000000	2928.000000	2928.000000	2938.000000	2744.000000	2938.000000	2385.000000	2938.000000	2904.000000	2938.000000	2919.000000	2712.000
mean	2007.518720	69.224932	164.796448	30.303948	4.602861	738.251295	80.940461	2419.592240	38.321247	42.035739	82.550188	5.9381
std	4.613841	9.523867	124.292079	117.926501	4.052413	1987.914858	25.070016	11467.272489	20.044034	160.445548	23.428046	2.4983
min	2000.000000	36.300000	1.000000	0.000000	0.010000	0.000000	1.000000	0.000000	1.000000	0.000000	3.000000	0.370
25%	2004.000000	63.100000	74.000000	0.000000	0.877500	4.685343	77.000000	0.000000	19.300000	0.000000	78.000000	4.260
50%	2008.000000	72.100000	144.000000	3.000000	3.755000	64.912906	92.000000	17.000000	43.500000	4.000000	93.000000	5.755
75%	2012.000000	75.700000	228.000000	22.000000	7.702500	441.534144	97.000000	360.250000	56.200000	28.000000	97.000000	7.4925
max	2015.000000	89.000000	723.000000	1800.000000	17.870000	19479.911610	99.000000	212183.000000	87.300000	2500.000000	99.000000	17.600

```
df.shape
```

```
(2938, 22)
```

DATA PREPROCESSING

```
new_cols = []
for col in df.columns:
  new_cols.append(col.strip())

df.columns = new_cols
df.head()
```

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	Polio	Total expenditure	Diphtheria	HIV/AIDS	G
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	6.0	8.16	65.0	0.1	584.2592
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	58.0	8.18	62.0	0.1	612.6965
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	62.0	8.13	64.0	0.1	631.7449
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	67.0	8.52	67.0	0.1	669.9590
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	68.0	7.87	68.0	0.1	63.5372

5 rows × 22 columns

```
df.isna().sum()
```

```
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
```

```
df.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')
```

```
df.drop(columns=['Country','Year'],inplace=True)
```

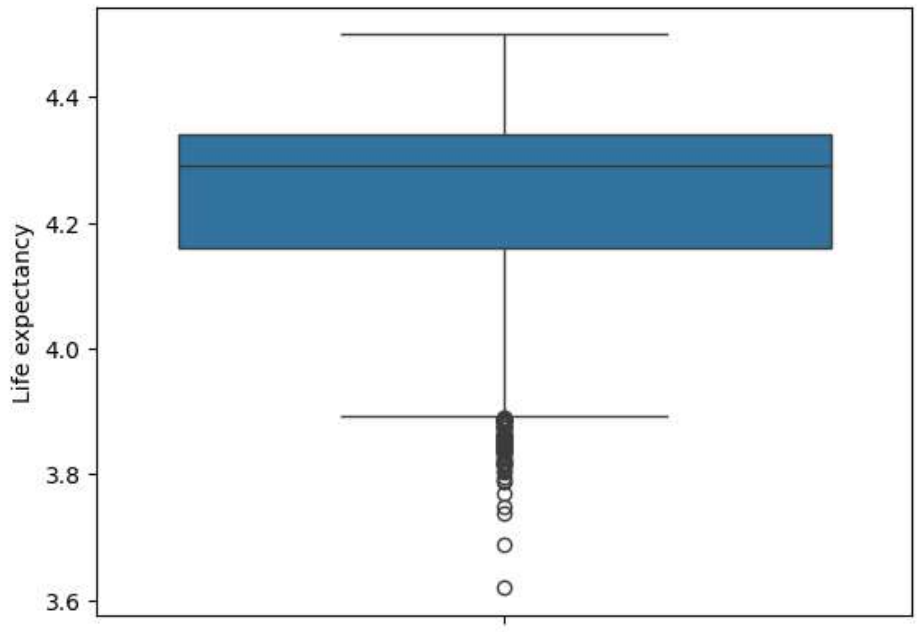
```
columns=['Adult Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles', 'BMI', 'under-five deaths', 'Polio', 'Total expenditure',
       'Diphtheria', 'HIV/AIDS', 'GDP', 'Population',
       'thinness  1-19 years', 'thinness 5-9 years',
       'Income composition of resources', 'Schooling']
for col in columns:
  df[col]=df[col].fillna(df[col].mean())
df
# mean cuz it data normally distributed, symmetricaly distribiuted ,impact on analysis(statitical analysis)
```

	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio	Total expenditure	Diphtheria	HIV/AIDS	GDP	Po
0	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	19.1	83	6.0	8.16	65.0	0.1	584.259210	3
1	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	18.6	86	58.0	8.18	62.0	0.1	612.696514	
2	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	18.1	89	62.0	8.13	64.0	0.1	631.744976	3
3	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	17.6	93	67.0	8.52	67.0	0.1	669.959000	
4	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	17.2	97	68.0	7.87	68.0	0.1	63.537231	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2933	Developing	44.3	723.0	27	4.36	0.000000	68.0	31	27.1	42	67.0	7.13	65.0	33.6	454.366654	1
2934	Developing	44.5	715.0	26	4.06	0.000000	7.0	998	26.7	41	7.0	6.52	68.0	36.7	453.351155	1
2935	Developing	44.8	73.0	25	4.43	0.000000	73.0	304	26.3	40	73.0	6.53	71.0	39.8	57.348340	
2936	Developing	45.3	686.0	25	1.72	0.000000	76.0	529	25.9	39	76.0	6.16	75.0	42.1	548.587312	1
2937	Developing	46.0	665.0	24	1.68	0.000000	79.0	1483	25.5	39	78.0	7.10	78.0	43.5	547.358878	1

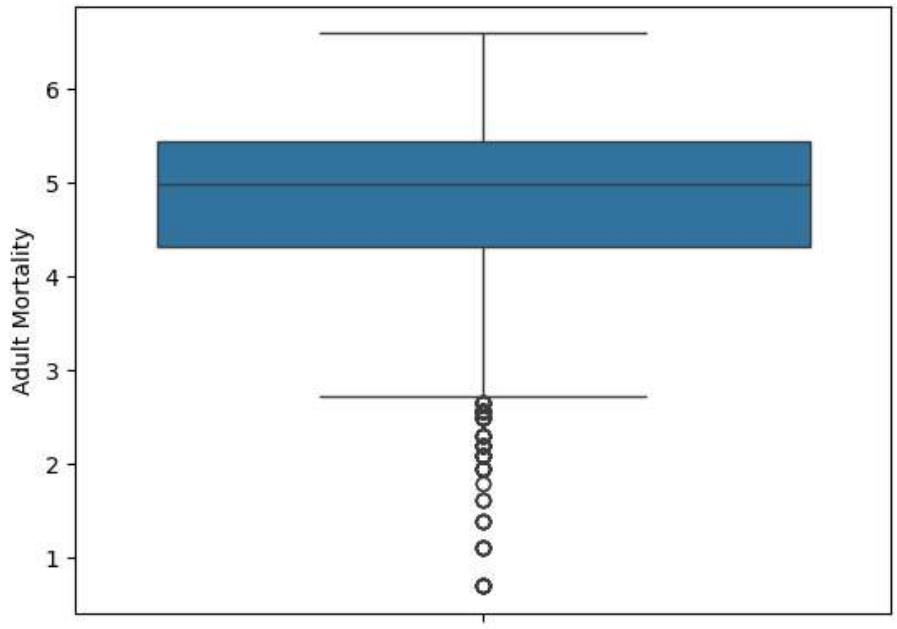
2938 rows × 20 columns

```
lst=[]
for i in df.columns:
  if df.dtypes[i]=='int64' or df.dtypes[i]=='float64':
    print(i)
    sns.boxplot(np.log1p(df[i]))
    plt.show()
    lst.append(i)
```

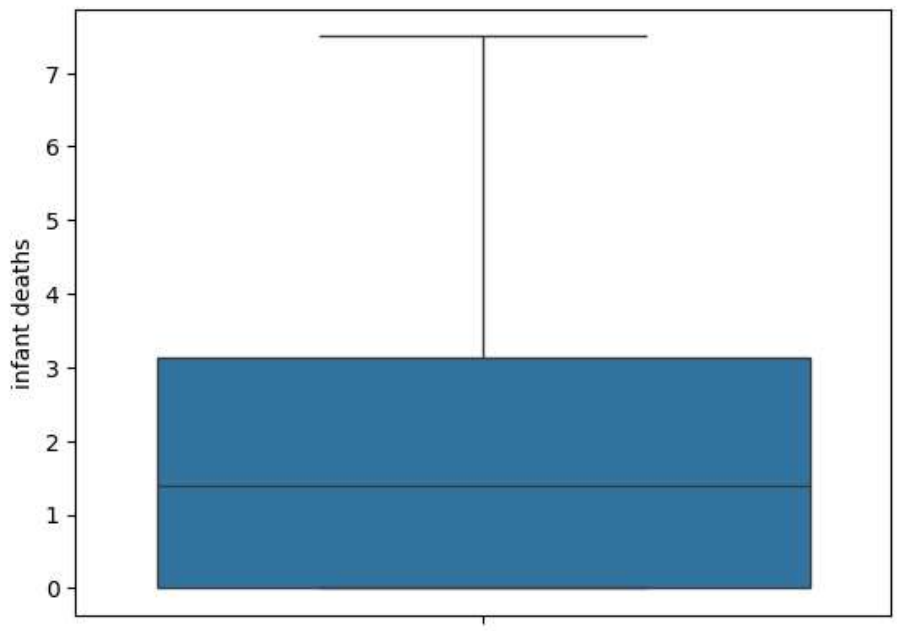
Life expectancy



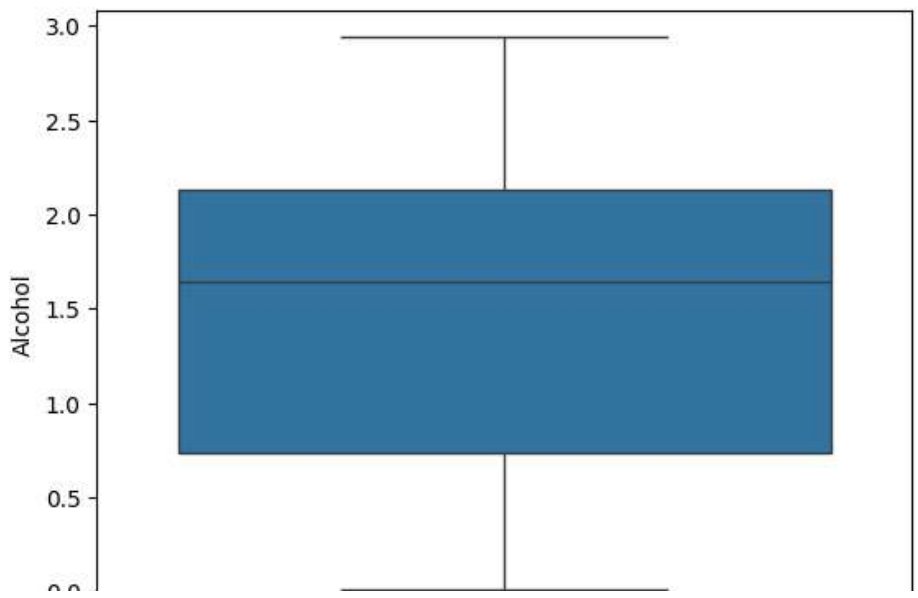
Adult Mortality



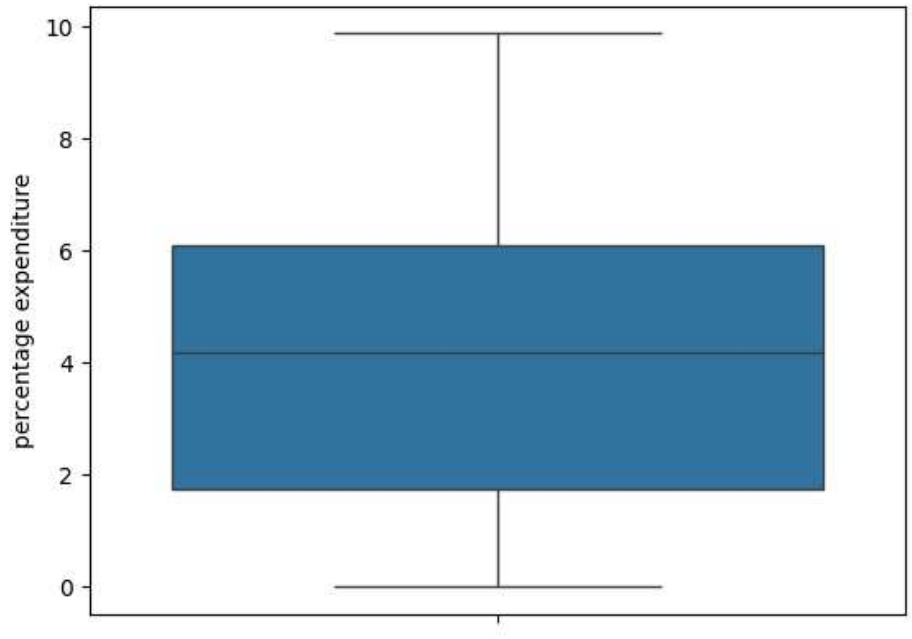
infant deaths



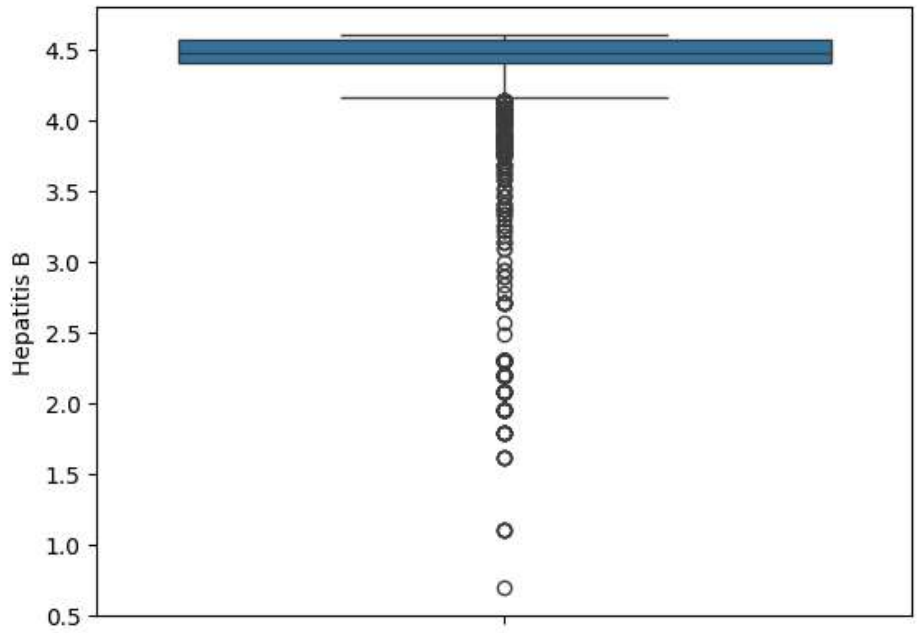
Alcohol



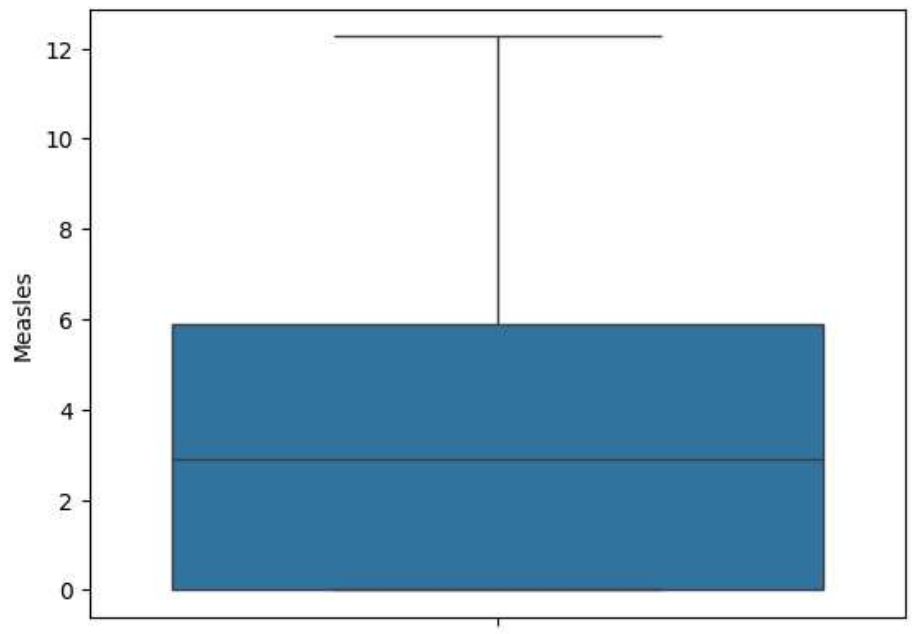
percentage expenditure



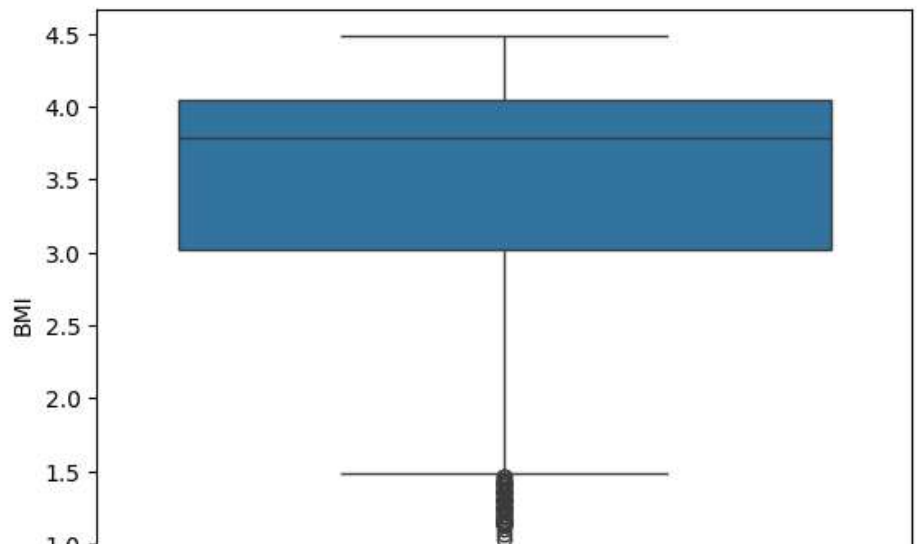
Hepatitis B

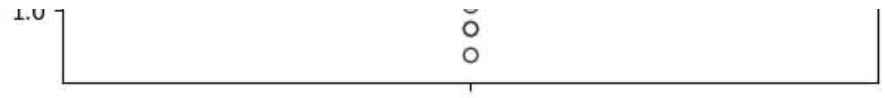


Measles

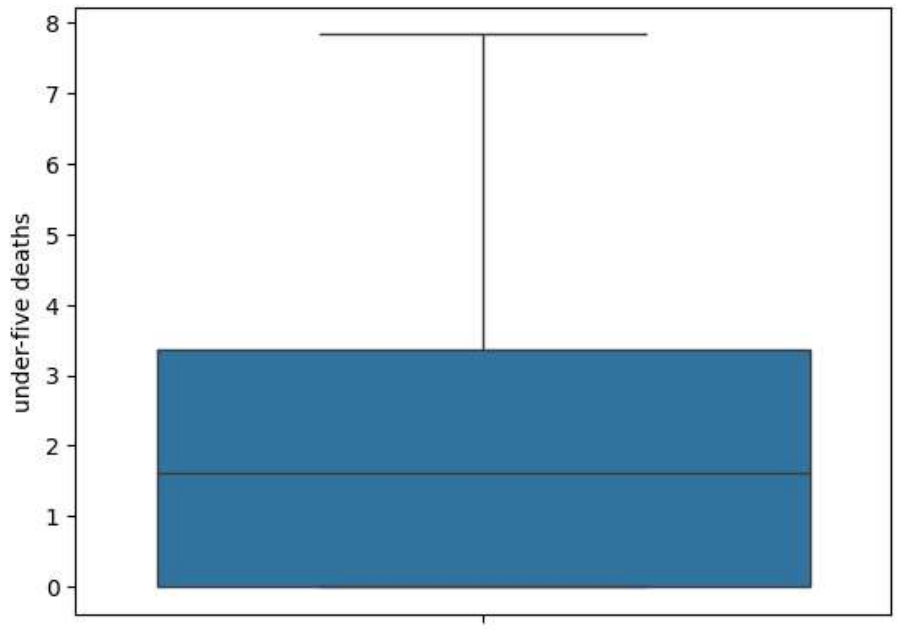


BMI

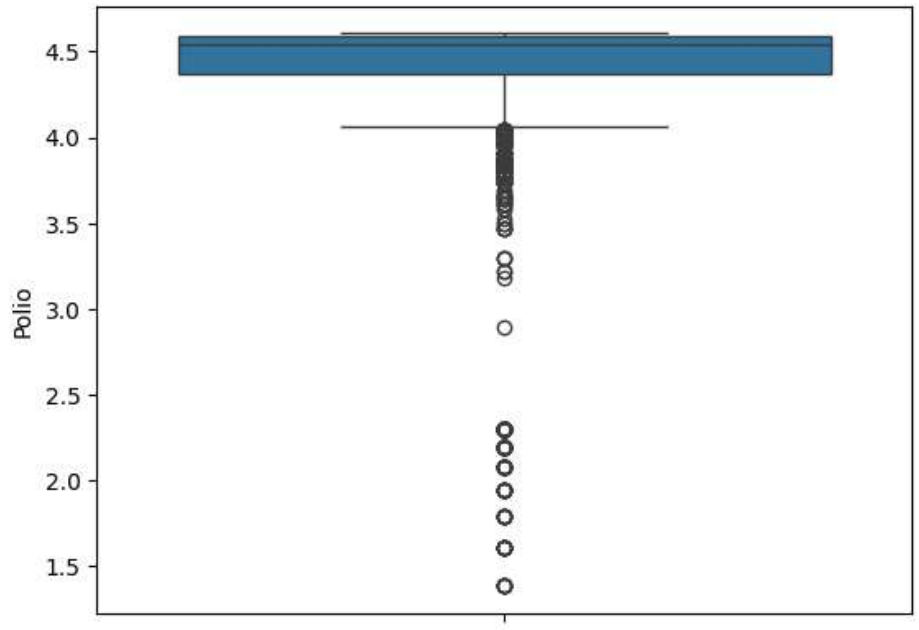




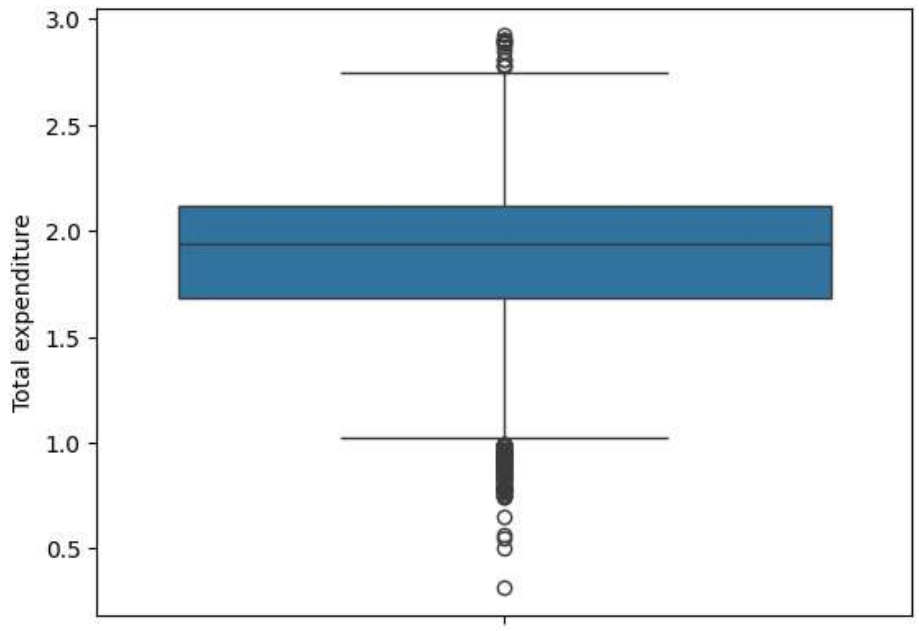
under-five deaths



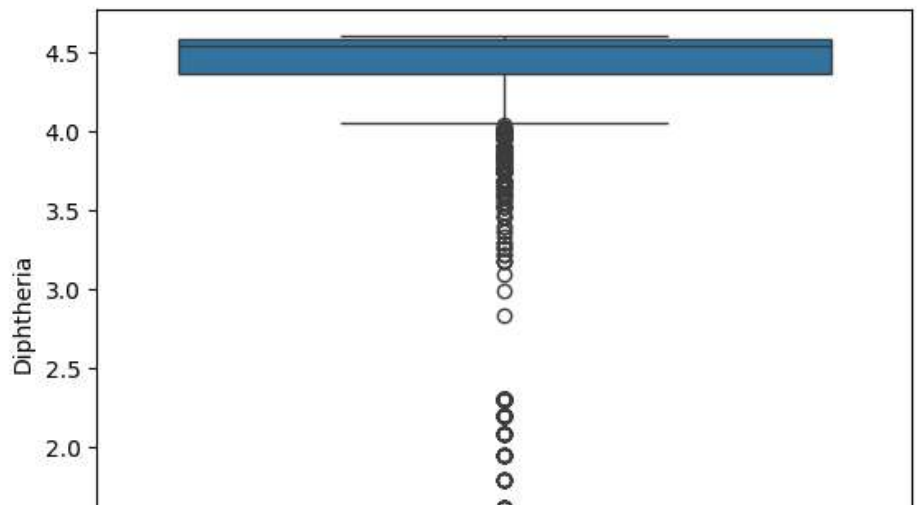
Polio

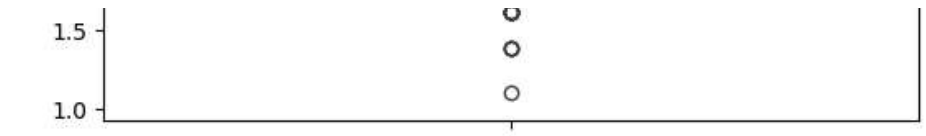


Total expenditure

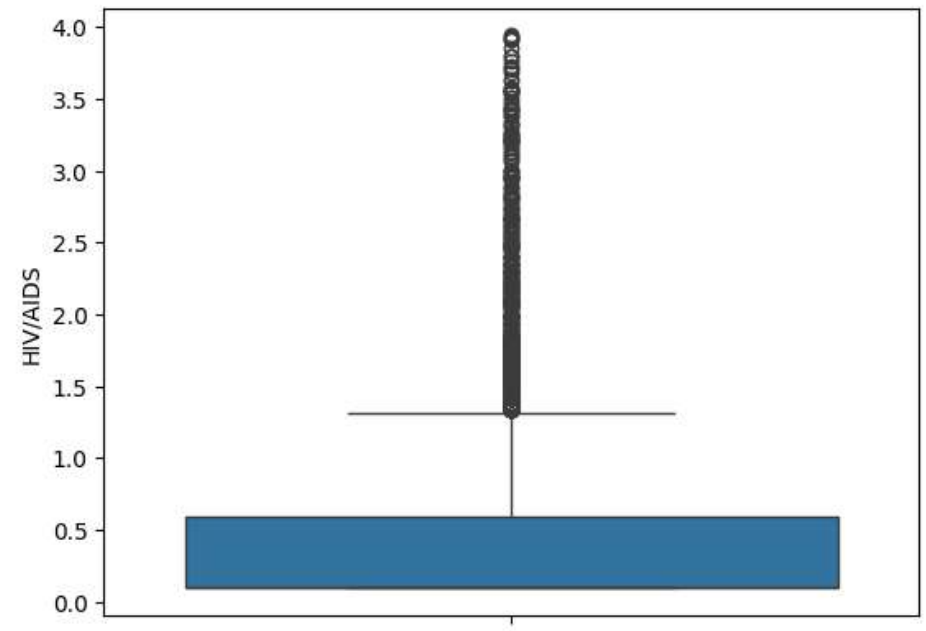


Diphtheria

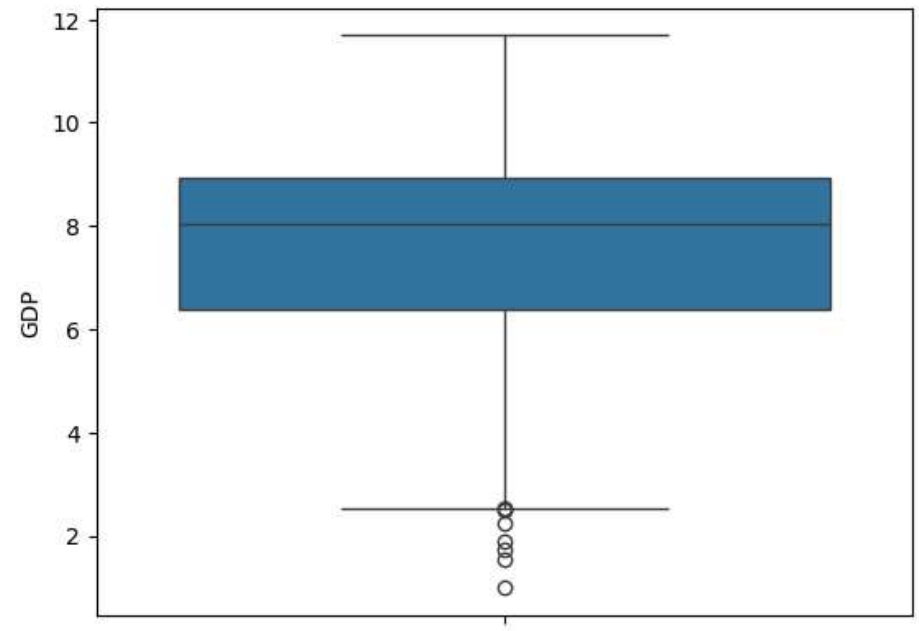




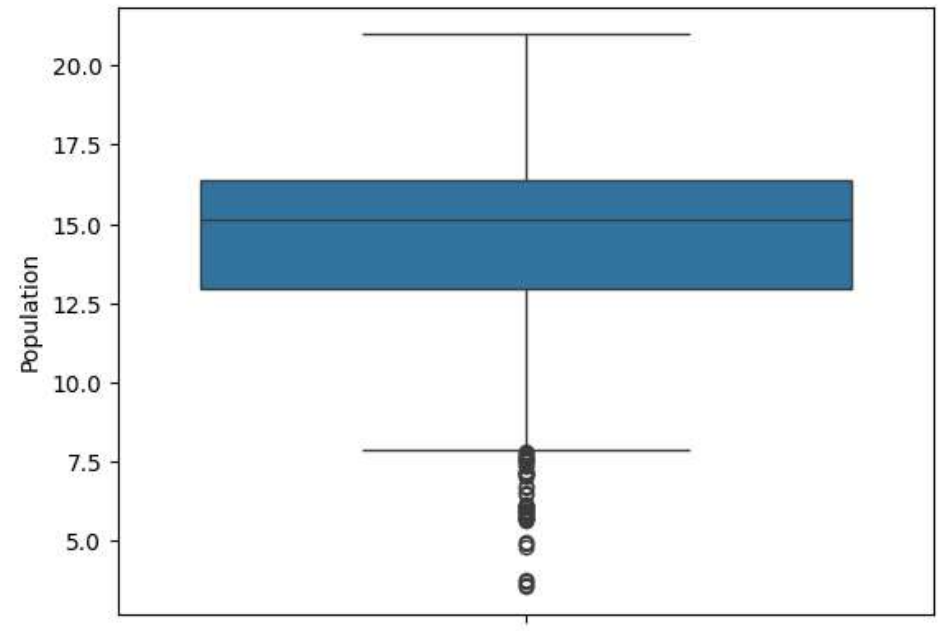

```
HIV/AIDS
```



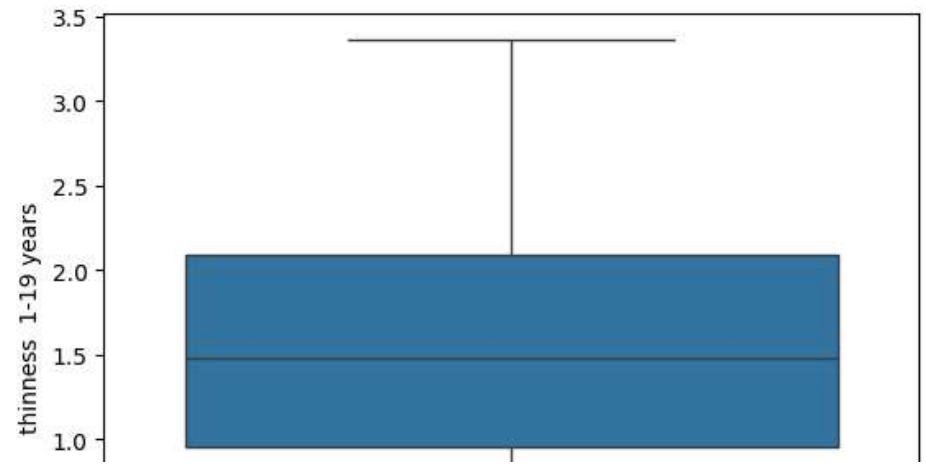

```
GDP
```



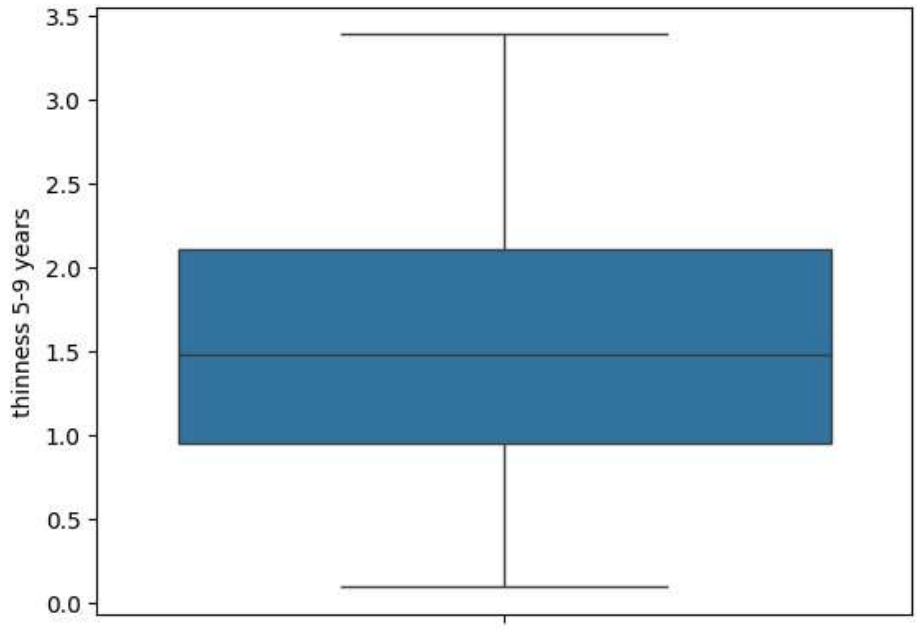

```
Population
```



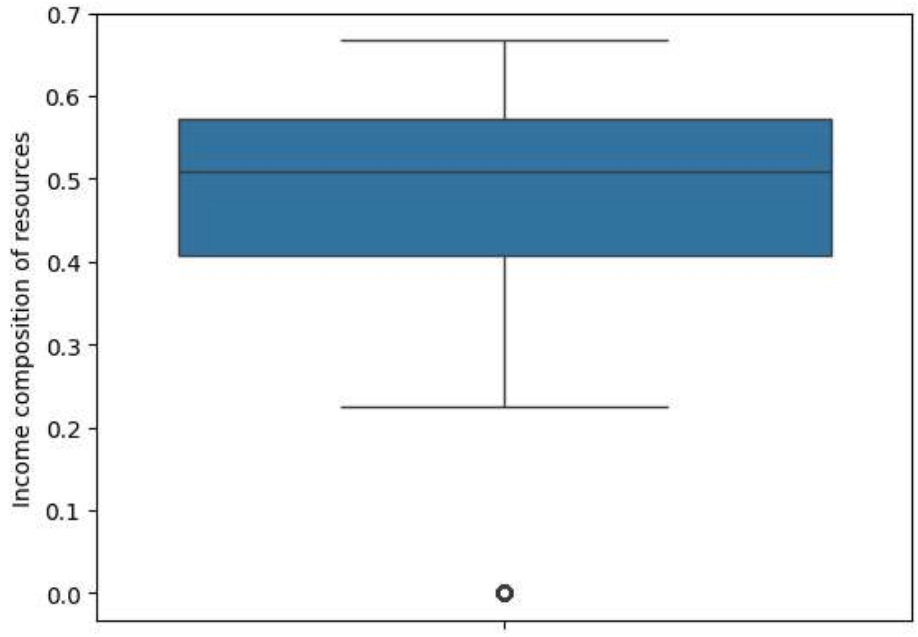

```
thinness  1-19 years
```



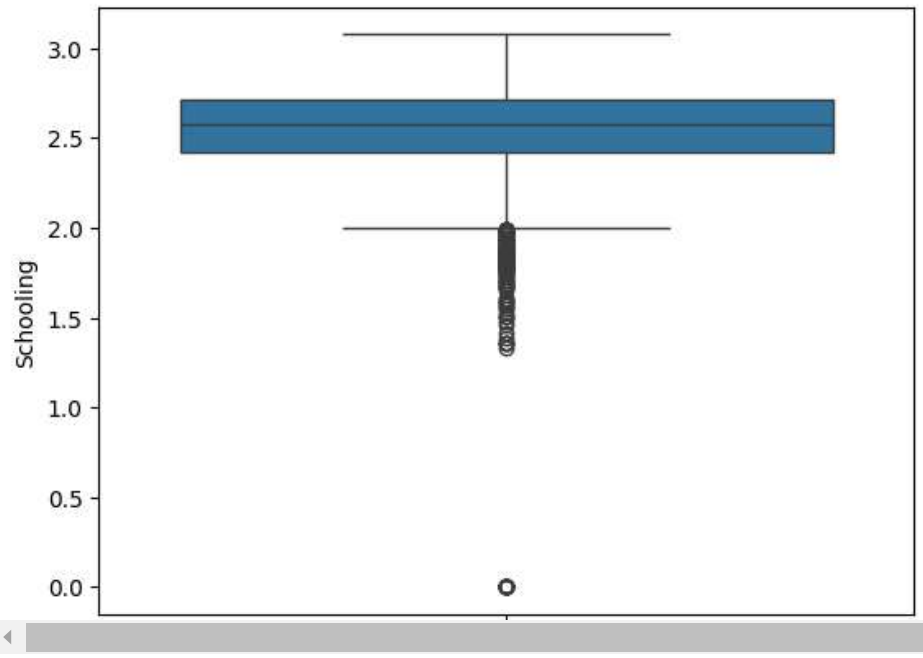
thinness 5-9 years



Income composition of resources



Schooling




```
cat_columns=[ 'Status']
le_dict={}
for col in cat_columns:
  le_dict[col]=LabelEncoder()
  df[col]=le_dict[col].fit_transform(df[col])
```

```
df.Status.value_counts()
```

```
Status
1    2416
```

```
0     512
Name: count, dtype: int64
```

```
lst= ['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']

for i in lst:
    df[i]=np.log1p(df[i])
robust=RobustScaler()
df[lst]=robust.fit_transform(df[lst])

'''
 Used to scale the data considering th oulayers before scaling features  log of fetaure increase the edfficiency of the scaler Robust scaling is a method used to
 the features of a dataset in a way that is robust to outliers. This method scales the data according to the interquartile range (IQR), which is the range betwee
 quartile (25th percentile) and the third quartile (75th percentile). Unlike standard scaling methods, which are influenced by extreme values, robust scaling foc
 central part of the data distribution.

 x_sccaled=x-x_mean/iqr

'''
```

```
'\n Used to scale the data considering th oulayers before scaling features  log of fetaure increase the edfficiency of the scaler Robust scaling is a method
used to scale\n the features of a dataset in a way that is robust to outliers. This method scales the data according to the interquartile range (IQR), which
is the range between the first\n quartile (25th percentile) and the third quartile (75th percentile). Unlike standard scaling methods, which are influenced b
```

```
df.dropna(inplace=True)
df = df.reset_index()
```

```
df.info()
```

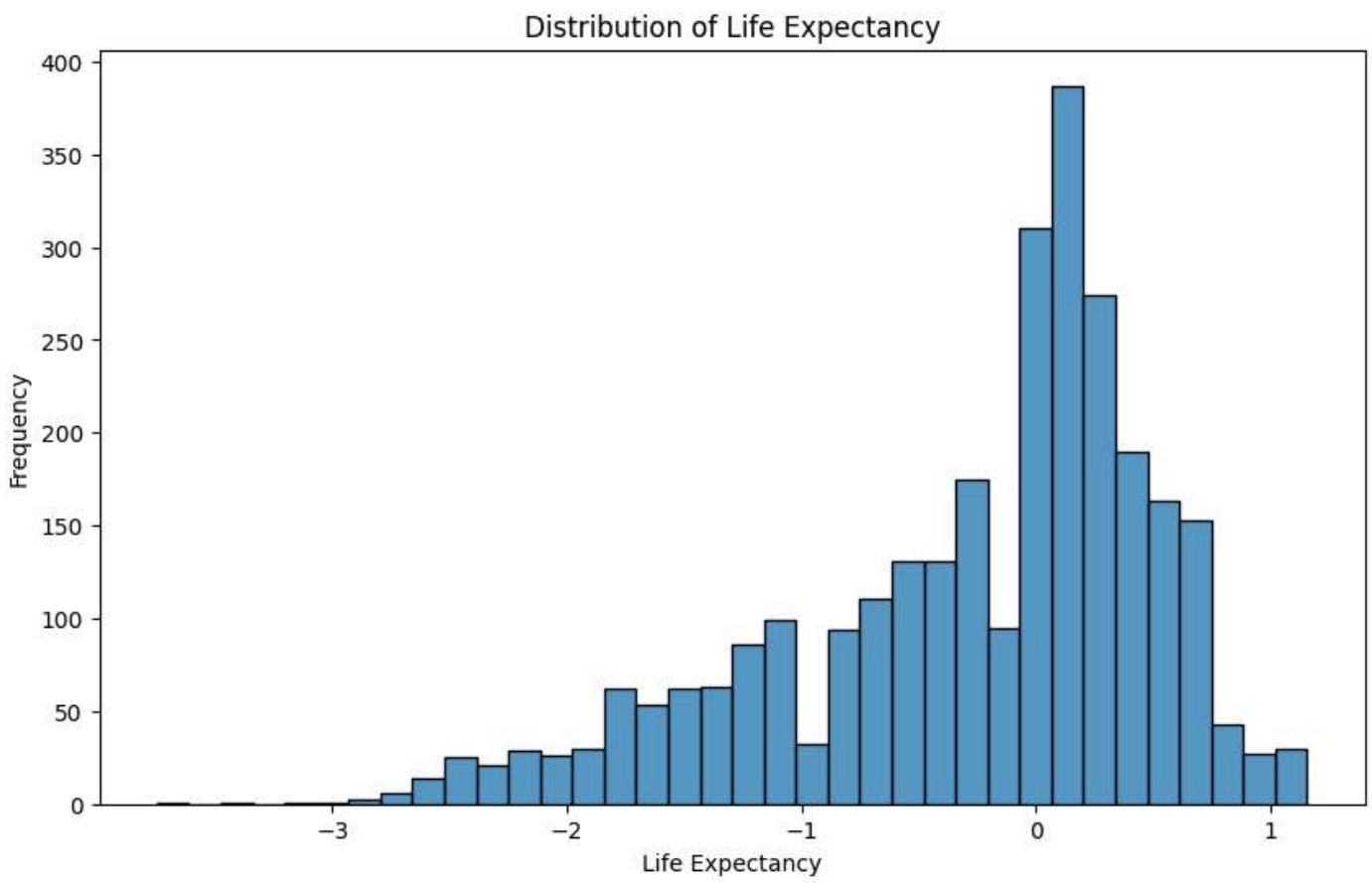
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2928 entries, 0 to 2927
Data columns (total 21 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   index                            2928 non-null   int64
 1   Status                           2928 non-null   int64
 2   Life expectancy                  2928 non-null   float64
 3   Adult Mortality                  2928 non-null   float64
 4   infant deaths                    2928 non-null   float64
 5   Alcohol                          2928 non-null   float64
 6   percentage expenditure           2928 non-null   float64
 7   Hepatitis B                      2928 non-null   float64
 8   Measles                          2928 non-null   float64
 9   BMI                              2928 non-null   float64
 10  under-five deaths                2928 non-null   float64
 11  Polio                            2928 non-null   float64
 12  Total expenditure                2928 non-null   float64
 13  Diphtheria                       2928 non-null   float64
 14  HIV/AIDS                         2928 non-null   float64
 15  GDP                              2928 non-null   float64
 16  Population                       2928 non-null   float64
 17  thinness  1-19 years             2928 non-null   float64
 18  thinness 5-9 years               2928 non-null   float64
 19  Income composition of resources  2928 non-null   float64
 20  Schooling                        2928 non-null   float64
dtypes: float64(19), int64(2)
memory usage: 480.5 KB
```

```
df.isna().sum()
```

```
index                            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
```

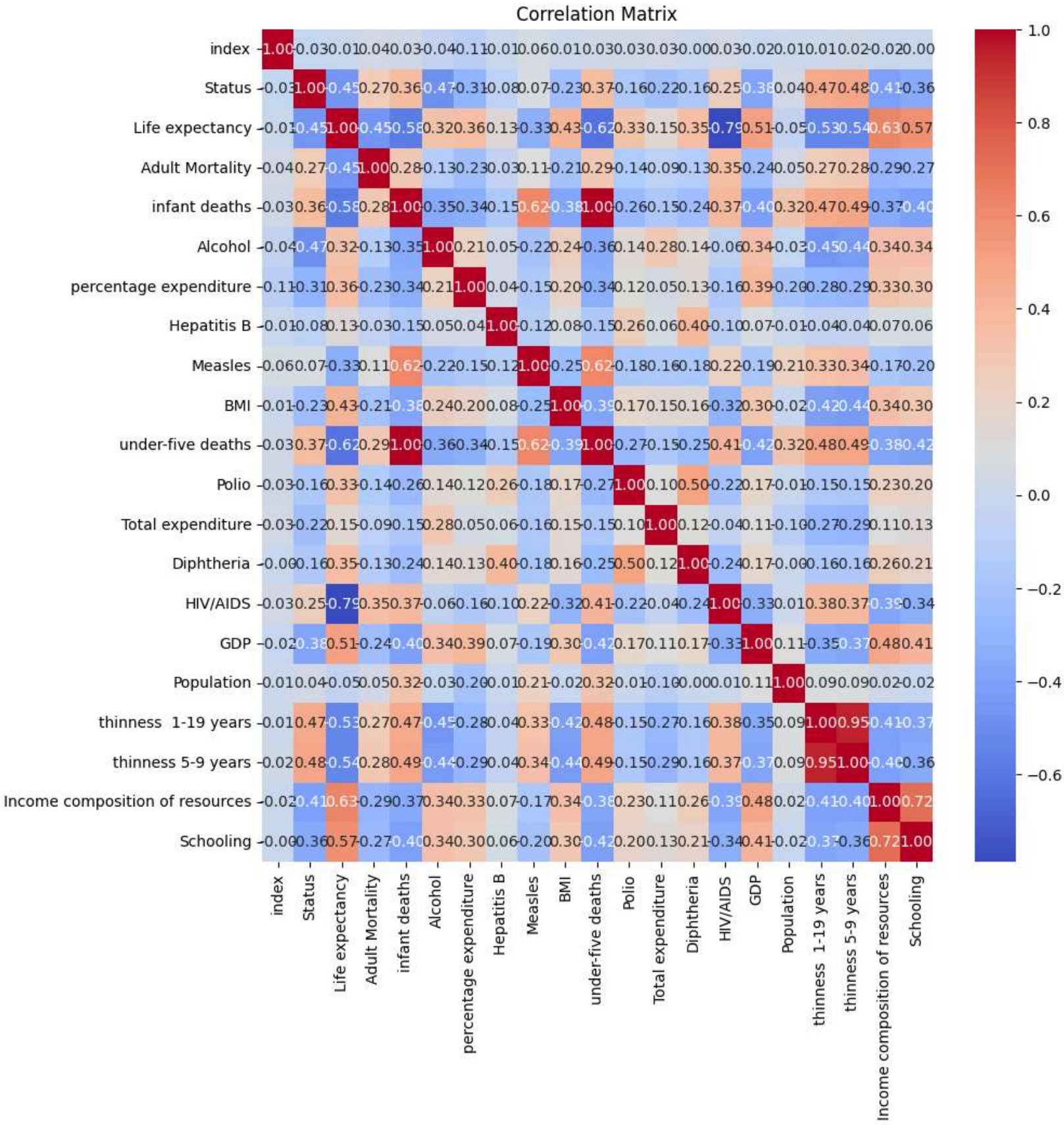
EXPLORATORY DATA ANALYSIS

```
plt.figure(figsize=(10, 6))
sns.histplot(df['Life expectancy'])
plt.title('Distribution of Life Expectancy')
plt.xlabel('Life Expectancy')
plt.ylabel('Frequency')
plt.show()
```



1. *Central Tendency:*
   - The most frequent life expectancy range is between 70 and 75 years, indicating that a significant number of observations fall within this range.
2. *Skewness:*
   - The distribution appears to be slightly left-skewed, with a longer tail towards the lower life expectancy values. This suggests that while most observations are clustered around higher life expectancies, there are some lower values that are less frequent but still present.
3. *Spread and Range:*
   - Life expectancy values range from around 40 to 90 years. The majority of the data falls between 50 and 80 years, with fewer observations at the extreme ends of the spectrum.

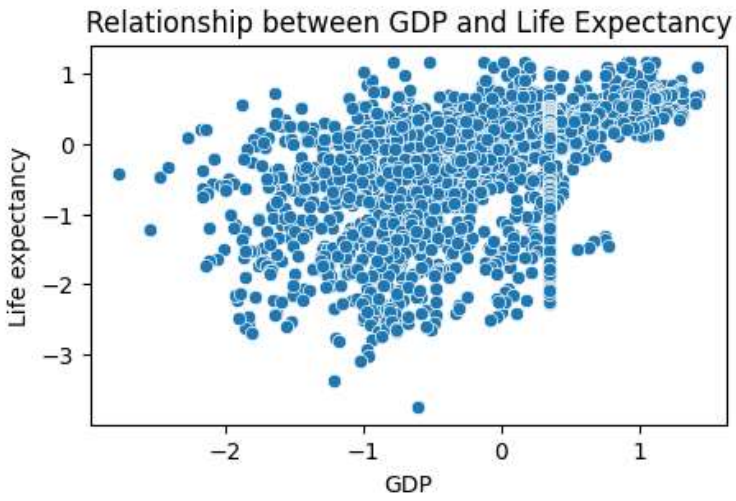
```
plt.figure(figsize=(10 ,10))
numeric_data = df.select_dtypes(include=['float64', 'int64'])
correlation_matrix = numeric_data.corr()
sns.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



Insights that can be derived from the heatmap:

1. *Strong Correlations*:
   - *BMI and Life Expectancy*: There seems to be a strong positive correlation between BMI and life expectancy (dark red color).
   - *GDP and Life Expectancy*: GDP also shows a strong positive correlation with life expectancy.
   - *Adult Mortality and Under-five Deaths*: High correlation, indicating regions with high adult mortality also have high under-five deaths.
2. *Negative Correlations*:
   - *BMI and Under-five Deaths*: There is a strong negative correlation between BMI and under-five deaths.
   - *GDP and Under-five Deaths*: Higher GDP is negatively correlated with under-five deaths.
   - *Schooling and Under-five Deaths*: Better schooling appears to be negatively correlated with under-five deaths.
3. *Interesting Observations*:
   - *Alcohol Consumption*: Alcohol consumption shows varying degrees of correlation with different indicators, but none are extremely strong.
   - *Measles and Schooling*: There is a negative correlation, suggesting that better schooling might correlate with lower measles incidence.
4. *Health Expenditure*: Total health expenditure shows moderate to strong positive correlations with several positive health outcomes (e.g., life expectancy, lower under-five deaths).

```
plt.figure(figsize=(5, 3))
sns.scatterplot(x='GDP', y='Life expectancy', data=df)
plt.title('Relationship between GDP and Life Expectancy')
plt.show()
```



- There is a positive correlation between GDP and life expectancy.
- The highest average life expectancy is in Europe, while the lowest is in Africa.
- There are outliers in the data, particularly for countries with high GDP and low life expectancy.

Additional Insights:

- Infant mortality rate and under-five mortality rate have a strong negative correlation with life expectancy.
- Alcohol consumption and BMI have a weak positive correlation with life expectancy.
- The percentage expenditure on healthcare has a weak positive correlation with life expectancy.
- The prevalence of HIV/AIDS has a strong negative correlation with life expectancy.
- The income composition of resources has a weak positive correlation with life expectancy.
- Schooling has a weak positive correlation with life expectancy.

FEATURE EXTRACTION

```
x=df.drop(columns='Life expectancy')
y=df['Life expectancy']
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=42)
```

MODEL CREATION

USING

1 LINEAR REGRESSION

2 RANDOMFOREST REGRESSOR

3 SVR

```
linear_model = LinearRegression()
random_forest_model = RandomForestRegressor()
svr_model = SVR()

linear_model.fit(x_train, y_train)
random_forest_model.fit(x_train, y_train)
svr_model.fit(x_train, y_train)

linear_predictions = linear_model.predict(x_test)
random_forest_predictions = random_forest_model.predict(x_test)
svr_predictions = svr_model.predict(x_test)

print("Linear Regression MSE:", mean_squared_error(y_test, linear_predictions))
print("Random Forest MSE:", mean_squared_error(y_test, random_forest_predictions))
print("SVR MSE:", mean_squared_error(y_test, svr_predictions))

linear_accuracy = r2_score(y_test, linear_predictions)
random_forest_accuracy = r2_score(y_test, random_forest_predictions)
svr_accuracy = r2_score(y_test, svr_predictions)

print("Linear Regression Accuracy:", linear_accuracy)
print("Random Forest Accuracy:", random_forest_accuracy)
print("SVR Accuracy:", svr_accuracy)

# lin  - asumes lin relationship  input var  and tag var, estimates coeff sum squared diff b/w a & p
# rand - it is an ensemble tech for regre tasks , build multiples des tree dur training and avg their pred avg  to impr,
# svr  - used for regre, find aprox the target values with specified margin tolerernce
```

```
Linear Regression MSE: 0.11017220894431294
Random Forest MSE: 0.026126341772865713
SVR MSE: 0.7356243260186037
Linear Regression Accuracy: 0.8373373405012308
Random Forest Accuracy: 0.9614260231643699
SVR Accuracy: -0.08610520210825401
```

Best parameters

```
random_forest_params = {'n_estimators': [10, 50, 100], 'max_depth': [None, 5, 10]}

random_forest_grid = GridSearchCV(random_forest_model, random_forest_params, cv=5)
random_forest_grid.fit(x_train, y_train)

#
```

```
▸ GridSearchCV
▸ estimator: RandomForestRegressor
  ▸ RandomForestRegressor
```

```
print("Random Forest MSE:", mean_squared_error(y_test, random_forest_predictions))
print("Random Forest Accuracy:", random_forest_accuracy)

# mse avg sqrd diff b/w at& p
#r2 prop of variance dep var explained by ind var 0 -1 1-rss/tss
```

```
Random Forest MSE: 0.026126341772865713
Random Forest Accuracy: 0.9614260231643699
```

RANDOM FOREST REGRESSOR STANDS OUT THE BEST MODEL

time taken 0.45s

Conclusion

- The analysis revealed that GDP, BMI, and schooling have positive correlations with life expectancy, while infant mortality rate, under-five mortality rate, and HIV/AIDS prevalence have negative correlations.
- The random forest regressor was identified as the best model for predicting life expectancy based on the available data.

Additional Notes:

- The project demonstrates the use of various data preprocessing techniques, exploratory data analysis methods, and machine learning models for analyzing a real-world dataset.

- The results and insights obtained from this project can be used to inform policy decisions and interventions aimed at improving health

```
# dict1={"model":random_forest_grid,"label_encoder":le_dict,"Scaler":robust}
# import pickle

# with open('model.pkl', 'wb') as f:
#     pickle.dump(dict1, f)
# f.close()
```

```
# with open('model.pkl', 'rb') as f:
#     loaded_model = pickle.load(f)
```

```
import pickle
dict1={"model":random_forest_grid,"label_encoder":le_dict,"Scaler":robust}
file=open('file.pkl','wb')
pickle.dump(dict1,file)
```

```
file1=open('file.pkl','rb')
res=pickle.load(file1)
res/
```

```
{'model': GridSearchCV(cv=5, estimator=RandomForestRegressor(),
                 param_grid={'max_depth': [None, 5, 10],
                             'n_estimators': [10, 50, 100]}),
     'label_encoder': {'Status': LabelEncoder()},
     'Scaler': RobustScaler()}
```

```
loaded_model
```

```
{'model': GridSearchCV(cv=5, estimator=RandomForestRegressor(),
                 param_grid={'max_depth': [None, 5, 10],
                             'n_estimators': [10, 50, 100]}),
     'label_encoder': {'Status': LabelEncoder()},
     'Scaler': RobustScaler()}
```

```
def prediction(status, adult_mortality, infant_deaths, alcohol, percentage_expenditure, hepatitis, measles, bmi,
               under_five_deaths, polio, total_expenditure, diphtheria, hiv_aids, gdp, population, thinness_large,
               thinness_small, income_composition, schooling):
    # Transform categorical variable
    status = le_Status.transform([status])[0]
```