FINAL CAPSTONE PROJECT

- Dalam proyek praktik ini, kita akan melatih model XG-Boost untuk memprediksi harapan hidup.
- Data diperoleh dari Organisasi Kesehatan Dunia (WHO) dan Situs PBB dan berisi fitur-fitur seperti tahun, status, harapan hidup, kematian orang dewasa, kematian bayi, persentase pengeluaran, alkohol, dll.

> 1. Impor data “Life_Expectancy_Data.csv” dengan Pandas
> 2. Periksa apakah ada missing value pada data, lakukan rekayasa fitur untuk menghilangkan atau mengisi missing value
> 3. Berapa banyak memori yang terpakai dari DataFrame tersebut
> 4. Hitung nilai minimum, rata-rata dan maksimum dari life expectancy
> 5. Plot histogram, pairplot dan heatmap dari matriks korelasi untuk semua fitur
> 6. Plot scatterplot antara “Income composition of resources” dan “life expectancy”, gunakan “status” sebagai atribut hue. Berikan komentar pada plot yang dibentuk.
> 7. Plot scatterplot antara “Schooling” dan “life expectancy”, gunakan “status” sebagai atribut hue. Berikan komentar pada plot yang dibentuk.
> 8. Bagi data menjadi 80% data latih dan 20% data testing
> 9. Latih model XG-boost
> 10. Evaluasi model regresi yang telah dilatih, apa itu R2?
> 11. Plot prediksi dari model yang dilatih vs keluaran sebenarnya (ground-truth)

In [13]:

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

In [14]:

```
life_df = pd.read_csv('Life_Expectancy_Data.csv')
```

In [15]:

```
life_df.isnull().sum()
```

Out[15]:

```
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
```

In [16]:

```
life_df2 = life_df.dropna()
```

In [17]:

```
life_df2.isnull().sum()
```

Out[17]:

```
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
```

In [18]:

```python
life_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1649 entries, 0 to 2937
Data columns (total 21 columns):
 #   Column                           Non-Null Count  Dtype  
---  ------                           --------------  -----  
 0   Year                             1649 non-null   int64  
 1   Status                           1649 non-null   object 
 2   Life expectancy                  1649 non-null   float64
 3   Adult Mortality                  1649 non-null   float64
 4   infant deaths                    1649 non-null   int64  
 5   Alcohol                          1649 non-null   float64
 6   percentage expenditure           1649 non-null   float64
 7   Hepatitis B                      1649 non-null   float64
 8   Measles                          1649 non-null   int64  
 9    BMI                             1649 non-null   float64
 10  under-five deaths                1649 non-null   int64  
 11  Polio                            1649 non-null   float64
 12  Total expenditure                1649 non-null   float64
 13  Diphtheria                       1649 non-null   float64
 14   HIV/AIDS                        1649 non-null   float64
 15  GDP                              1649 non-null   float64
 16  Population                       1649 non-null   float64
 17   thinness  1-19 years            1649 non-null   float64
 18   thinness 5-9 years              1649 non-null   float64
 19  Income composition of resources  1649 non-null   float64
 20  Schooling                        1649 non-null   float64
dtypes: float64(16), int64(4), object(1)
memory usage: 283.4+ KB
```

In [19]:

```python
life_df2['Life expectancy '].max()
```

Out[19]:

```
89.0
```

In [20]:

```python
life_df2['Life expectancy '].min()
```

Out[20]:

```
44.0
```

In [21]:

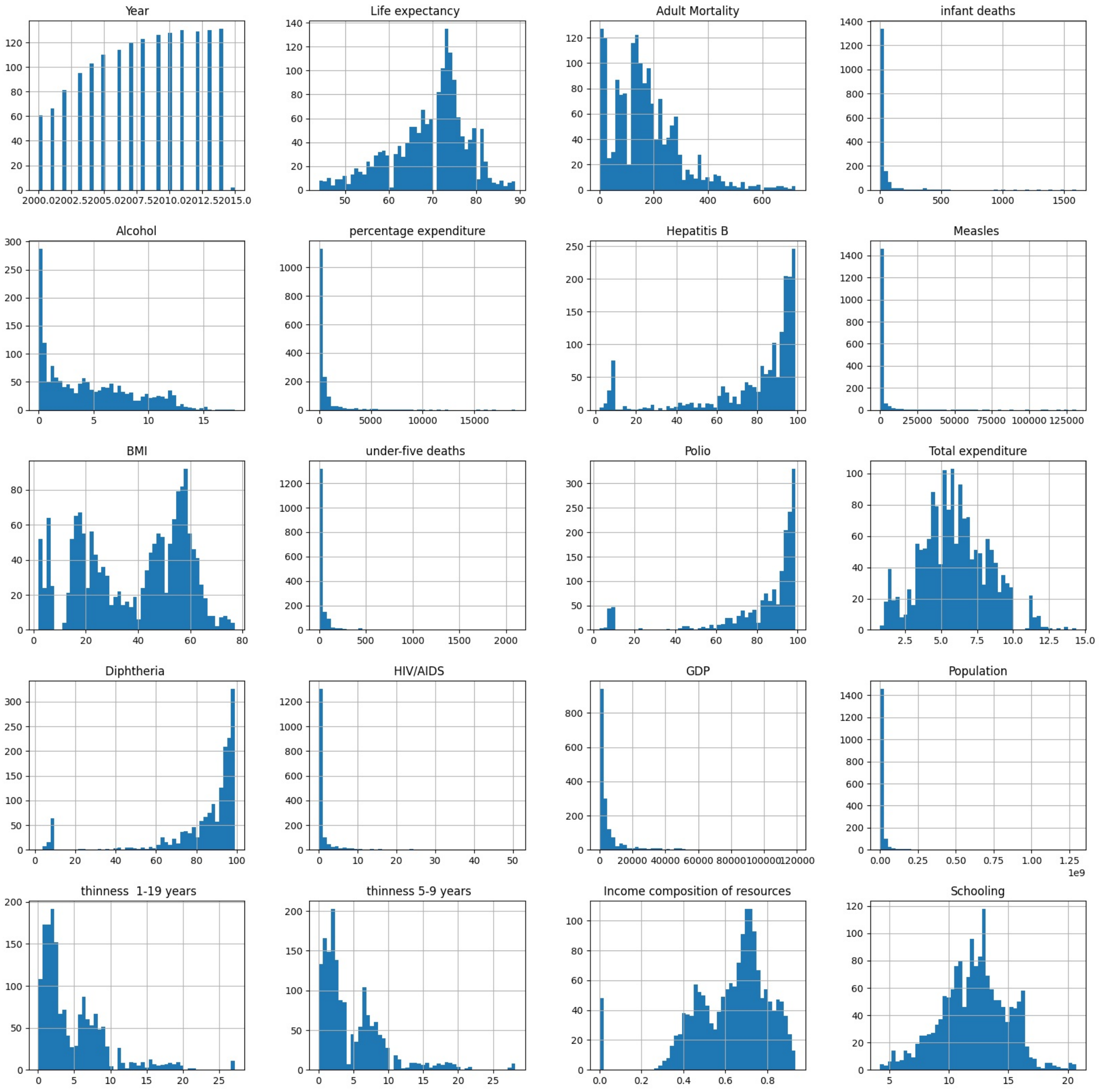
```python
life_df2['Life expectancy '].mean()
```

Out[21]:

```
69.3023044269254
```

In [22]:

```python
# plt.figure(figsize=(20,20))s
life_df2.hist(bins=50, figsize=(20,20))
plt.show()
```



```
In [29]: # sns.pairplot(life_df2)
         # plt.show()
```

```
In [45]: life_df2.corr()
```

Out[45]:

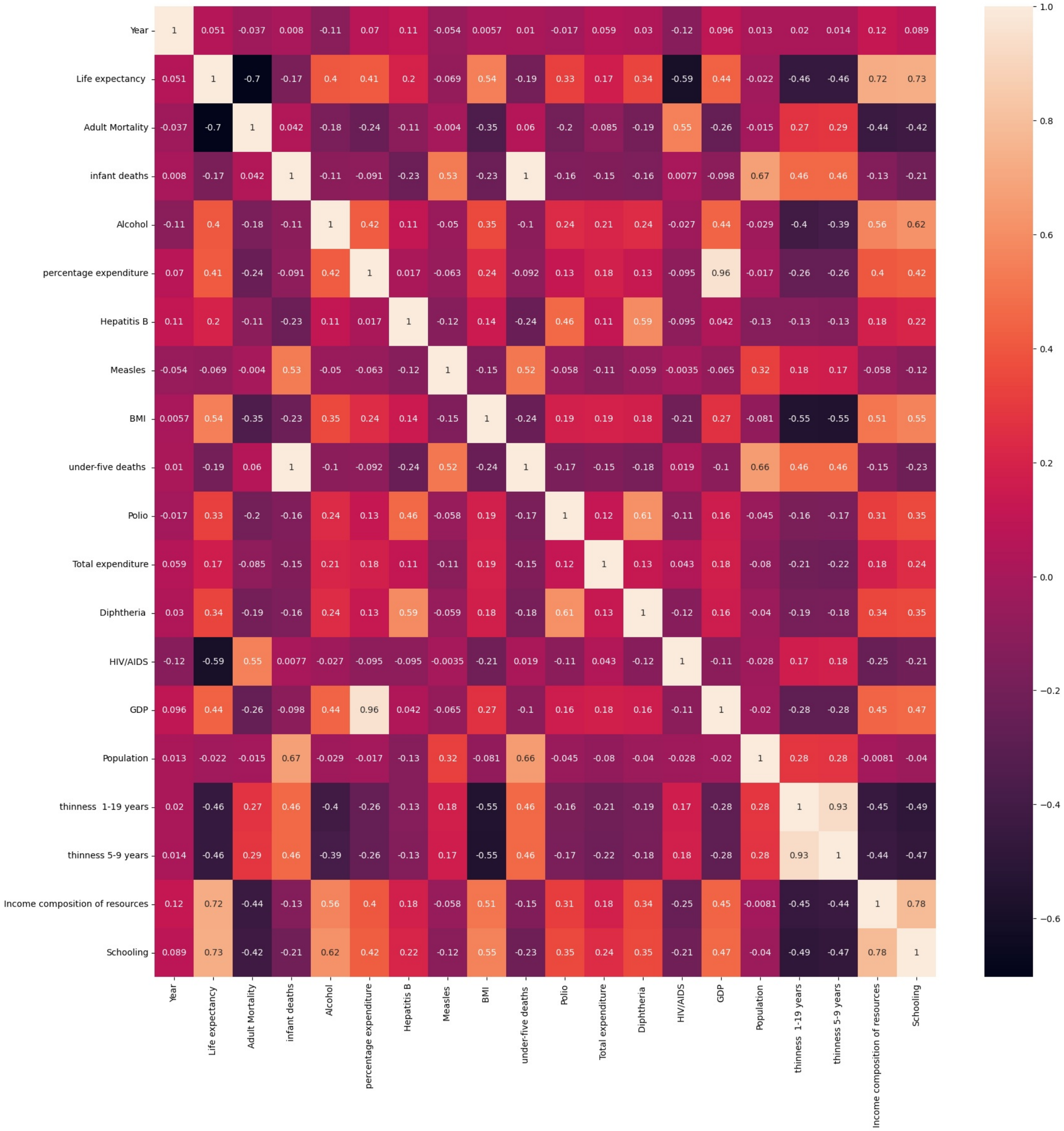
	Year	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio	expen
Year	1.000000	0.050771	-0.037092	0.008029	-0.113365	0.069553	0.114897	-0.053822	0.005739	0.010479	-0.016699	0.0
Life expectancy	0.050771	1.000000	-0.702523	-0.169074	0.402718	0.409631	0.199935	-0.068881	0.542042	-0.192265	0.327294	0.1
Adult Mortality	-0.037092	-0.702523	1.000000	0.042450	-0.175535	-0.237610	-0.105225	-0.003967	-0.351542	0.060365	-0.199853	-0.0
infant deaths	0.008029	-0.169074	0.042450	1.000000	-0.106217	-0.090765	-0.231769	0.532680	-0.234425	0.996906	-0.156929	-0.1
Alcohol	-0.113365	0.402718	-0.175535	-0.106217	1.000000	0.417047	0.109889	-0.050110	0.353396	-0.101082	0.240315	0.2
percentage expenditure	0.069553	0.409631	-0.237610	-0.090765	0.417047	1.000000	0.016760	-0.063071	0.242738	-0.092158	0.128626	0.1
Hepatitis B	0.114897	0.199935	-0.105225	-0.231769	0.109889	0.016760	1.000000	-0.124800	0.143302	-0.240766	0.463331	0.1
Measles	-0.053822	-0.068881	-0.003967	0.532680	-0.050110	-0.063071	-0.124800	1.000000	-0.153245	0.517506	-0.057850	-0.1
BMI	0.005739	0.542042	-0.351542	-0.234425	0.353396	0.242738	0.143302	-0.153245	1.000000	-0.242137	0.186268	0.1
under-five deaths	0.010479	-0.192265	0.060365	0.996906	-0.101082	-0.092158	-0.240766	0.517506	-0.242137	1.000000	-0.171164	-0.1
Polio	-0.016699	0.327294	-0.199853	-0.156929	0.240315	0.128626	0.463331	-0.057850	0.186268	-0.171164	1.000000	0.1
Total expenditure	0.059493	0.174718	-0.085227	-0.146951	0.214885	0.183872	0.113327	-0.113583	0.189469	-0.145803	0.119768	1.0
Diphtheria	0.029641	0.341331	-0.191429	-0.161871	0.242951	0.134813	0.588990	-0.058606	0.176295	-0.178448	0.609245	0.1
HIV/AIDS	-0.123405	-0.592236	0.550691	0.007712	-0.027113	-0.095085	-0.094802	-0.003522	-0.210897	0.019476	-0.107885	0.0
GDP	0.096421	0.441322	-0.255035	-0.098092	0.443433	0.959299	0.041850	-0.064768	0.266114	-0.100331	0.156809	0.1
Population	0.012567	-0.022305	-0.015012	0.671758	-0.028880	-0.016792	-0.129723	0.321946	-0.081416	0.658680	-0.045387	-0.0
thinness 1-19 years	0.019757	-0.457838	0.272230	0.463415	-0.403755	-0.255035	-0.129406	0.180642	-0.547018	0.464785	-0.164070	-0.2
thinness 5-9 years	0.014122	-0.457508	0.286723	0.461908	-0.386208	-0.255635	-0.133251	0.174946	-0.554094	0.462289	-0.174489	-0.2
Income composition of resources	0.122892	0.721083	-0.442203	-0.134754	0.561074	0.402170	0.184921	-0.058277	0.510505	-0.148097	0.314682	0.1
Schooling	0.088732	0.727630	-0.421171	-0.214372	0.616975	0.422088	0.215182	-0.115660	0.554844	-0.226013	0.350147	0.2

In [46]:

```
plt.figure(figsize=(20,20))
sns.heatmap(life_df2.corr(), annot = True)
```

Out[46]:

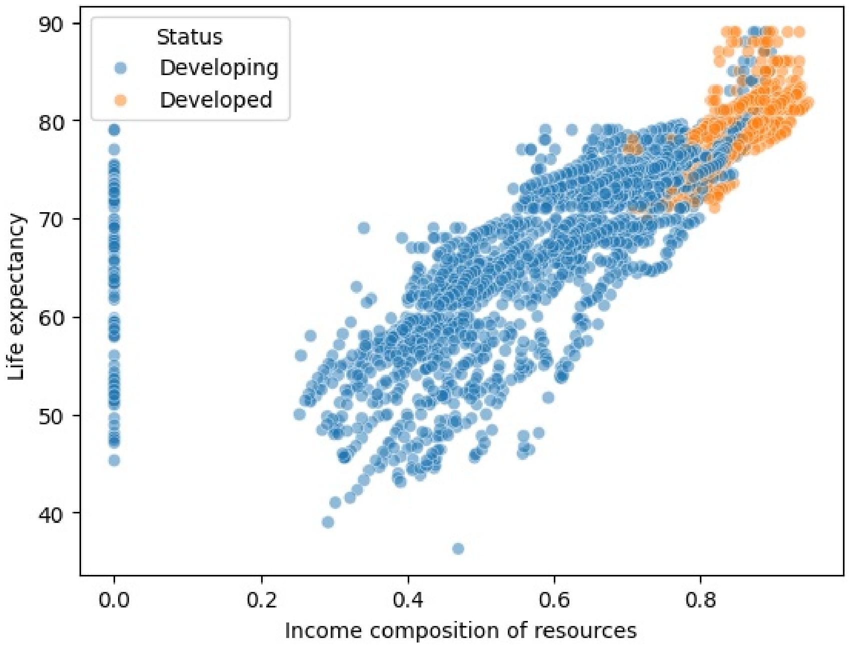
```
<Axes: >
```



```
In [52]: life_df.columns
```

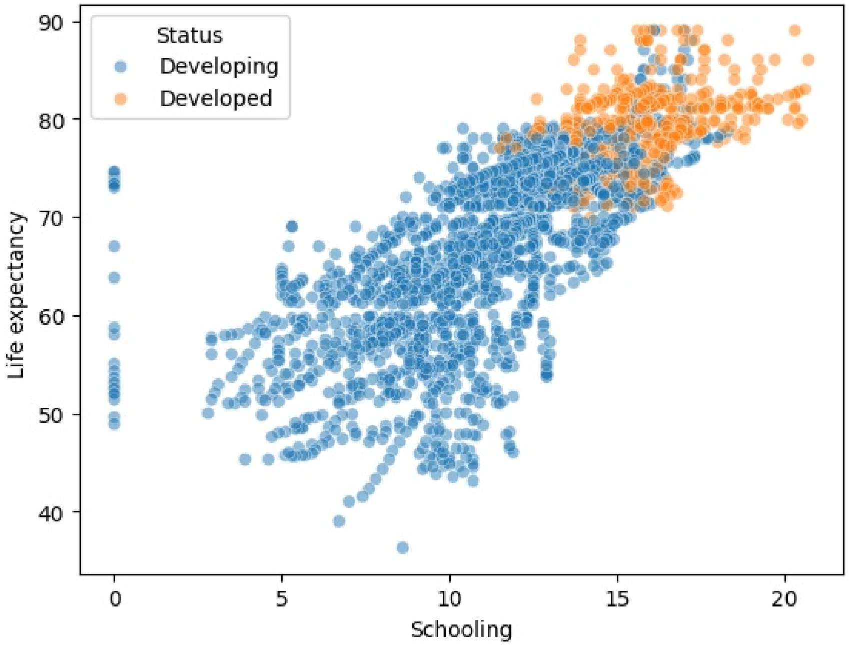
```
Out[52]: Index(['Year', 'Status', 'Life expectancy ', 'Adult Mortality',
                'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
                'Measles ', ' BMI ', 'under-five deaths ', 'Polio', 'Total expenditure',
                'Diphtheria ', ' HIV/AIDS', 'GDP', 'Population',
                ' thinness  1-19 years', ' thinness 5-9 years',
                'Income composition of resources', 'Schooling'],
               dtype='object')
```

```
In [61]: sns.scatterplot(x ='Income composition of resources', y='Life expectancy ', hue='Status', alpha = 0.5, data=lif
         plt.show()
```



In [62]:

```
sns.scatterplot(x ='Schooling', y='Life expectancy ', hue='Status', alpha = 0.5, data=life_df)
plt.show()
```



In [64]:

```
from sklearn.model_selection import train_test_split
```

In [66]:

```
life_df2.columns
```

Out[66]:

```
Index(['Year', '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')
```

In [69]:

```
life_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1649 entries, 0 to 2937
Data columns (total 20 columns):
 #   Column                           Non-Null Count  Dtype  
---  ------                           --------------  -----  
 0   Year                             1649 non-null   int64  
 1   Life expectancy                  1649 non-null   float64
 2   Adult Mortality                  1649 non-null   float64
 3   infant deaths                    1649 non-null   int64  
 4   Alcohol                          1649 non-null   float64
 5   percentage expenditure           1649 non-null   float64
 6   Hepatitis B                      1649 non-null   float64
 7   Measles                          1649 non-null   int64  
 8    BMI                             1649 non-null   float64
 9   under-five deaths                1649 non-null   int64  
 10  Polio                            1649 non-null   float64
 11  Total expenditure                1649 non-null   float64
 12  Diphtheria                       1649 non-null   float64
 13   HIV/AIDS                        1649 non-null   float64
 14  GDP                              1649 non-null   float64
 15  Population                       1649 non-null   float64
 16   thinness  1-19 years            1649 non-null   float64
 17   thinness 5-9 years              1649 non-null   float64
 18  Income composition of resources  1649 non-null   float64
 19  Schooling                        1649 non-null   float64
dtypes: float64(16), int64(4)
memory usage: 335.1 KB
```

In [73]:
```
X = life_df2.drop(columns=['Life expectancy '])
```

In [74]:
```
y = life_df2['Life expectancy ']
```

In [75]:
```
X.shape
```

Out[75]:
```
(1649, 19)
```

In [76]:
```
y.shape
```

Out[76]:
```
(1649,)
```

In [77]:
```
X = np.array(X)
y = np.array(y)
```

In [80]:
```
y = y.reshape(-1,1)
```

In [81]:
```
y.shape
```

Out[81]:
```
(1649, 1)
```

In [82]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0, train_size = 0.8)
```

In [84]:
```
X_train.shape
```

Out[84]:
```
(1319, 19)
```

In [86]:
```
X_test.shape
```

Out[86]:
```
(330, 19)
```

In [85]:
```
y_train.shape
```

Out[85]:
```
(1319, 1)
```

In [87]:
```
y_test.shape
```

Out[87]:
```
(330, 1)
```

In [88]:
```
!pip install xgboost
```

```
Requirement already satisfied: xgboost in /usr/local/lib/python3.10/dist-packages (2.0.3)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.25.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.11.4)
```

In [89]:
```
import xgboost as xgb

model = xgb.XGBRegressor(objective ='reg:squarederror', learning_rate = 0.1, max_depth = 2, n_estimators = 100)

model.fit(X_train, y_train)
```

Out[89]:

```
▾                              XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=Non
e,
             enable_categorical=False, eval_metric=None, feature_types=Non
e,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=0.1, max_bin=None
,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=2, max_leaves=None,
```

In [91]:

```
model.score(X_test, y_test)
```

Out[91]:

```
0.9405115978232569
```

In [95]:

```
y_predict = model.predict(X_test)
y_predict
```

Out[95]:

```
array([60.038383, 81.80574 , 75.29209 , 53.100655, 64.25743 , 80.127625,
       51.53877 , 55.695633, 48.17951 , 73.681786, 81.42267 , 73.883125,
       79.1772  , 73.21604 , 81.08935 , 51.591686, 52.12157 , 72.9015  ,
       72.93255 , 68.92449 , 77.93942 , 65.82495 , 65.066086, 73.06139 ,
       73.70144 , 66.88648 , 65.0939  , 57.762955, 80.006775, 71.37269 ,
       71.55293 , 50.989487, 59.635487, 70.11377 , 74.25845 , 71.43095 ,
       75.73187 , 81.825424, 81.08731 , 55.98042 , 76.76178 , 65.74226 ,
       81.48228 , 67.92013 , 82.76096 , 82.23607 , 51.569824, 74.6942  ,
       72.25836 , 81.97496 , 60.839916, 70.94349 , 59.301018, 81.10939 ,
       68.55446 , 81.87438 , 75.29864 , 78.588425, 68.54856 , 66.753334,
       72.93255 , 72.65823 , 68.66612 , 76.56913 , 76.19477 , 81.881386,
       72.87777 , 71.37372 , 72.988884, 74.13646 , 68.43472 , 81.81454 ,
       46.90968 , 81.34949 , 81.068184, 54.84374 , 75.81672 , 81.0613  ,
       64.789856, 67.83929 , 60.77602 , 76.579414, 66.539604, 66.253296,
       63.085014, 66.70991 , 73.85078 , 44.621185, 67.001945, 68.496216,
       64.1962  , 75.0734  , 76.1259  , 49.2858  , 71.68105 , 60.96845 ,
       70.03878 , 56.41933 , 50.029476, 53.181293, 48.255936, 62.52259 ,
       74.18237 , 68.27062 , 72.59304 , 80.79946 , 70.81421 , 52.321762,
       69.19996 , 60.235245, 72.643074, 68.25585 , 70.93557 , 60.70113 ,
       72.3372  , 73.59845 , 73.18729 , 69.19191 , 70.98424 , 70.351906,
       73.98763 , 74.73301 , 58.3044  , 73.68214 , 68.502686, 65.99409 ,
       82.23607 , 70.15541 , 69.93405 , 75.36959 , 53.748653, 75.38275 ,
       80.1633  , 76.19477 , 73.14676 , 81.422516, 77.93288 , 71.7825  ,
       81.16034 , 62.37273 , 72.69163 , 59.125366, 72.543274, 63.480213,
       73.96722 , 82.0041  , 61.99772 , 76.55361 , 82.657715, 69.99712 ,
       65.75027 , 67.71827 , 82.201836, 74.49699 , 82.657715, 59.284855,
       81.68797 , 73.496994, 69.28847 , 65.37262 , 82.76096 , 61.525894,
       56.55524 , 75.467545, 57.839043, 74.09439 , 78.52823 , 72.3474  ,
       45.676556, 77.212776, 64.64669 , 67.19851 , 68.84401 , 60.494385,
       74.31136 , 81.71829 , 66.71614 , 81.702095, 60.548874, 81.988655,
       69.42486 , 73.673775, 54.082905, 75.26634 , 54.968853, 64.58209 ,
       60.37126 , 68.46347 , 72.81395 , 63.83716 , 49.46759 , 75.07227 ,
       75.29326 , 72.84393 , 75.342064, 59.752228, 71.79592 , 54.060413,
       79.27006 , 50.319633, 73.601814, 69.328636, 72.02597 , 67.79603 ,
       71.80049 , 74.443504, 80.348885, 74.11126 , 75.9945  , 50.84247 ,
       60.561485, 65.32584 , 82.30043 , 80.412094, 66.75516 , 78.09529 ,
       72.26451 , 79.60585 , 64.48694 , 75.88517 , 71.43095 , 69.46476 ,
       80.34775 , 70.866905, 63.767677, 74.922806, 66.735085, 52.687035,
       81.414764, 61.154537, 82.10144 , 72.737526, 71.08861 , 70.53445 ,
       80.17305 , 55.48237 , 74.43556 , 82.14616 , 68.60575 , 81.262634,
       53.100655, 82.367065, 58.45281 , 73.9093  , 72.932755, 64.012596,
       75.09663 , 77.08788 , 76.32382 , 79.05808 , 75.34228 , 69.013954,
       82.13282 , 55.86899 , 77.30822 , 69.91755 , 75.29539 , 70.141464,
       74.73301 , 72.55791 , 73.81721 , 69.02893 , 72.5585  , 66.250435,
       81.0895  , 76.25064 , 67.51098 , 73.4791  , 49.295174, 67.33118 ,
       73.561966, 70.61961 , 81.54524 , 57.745495, 54.835594, 73.75155 ,
       71.273834, 57.5266  , 72.263824, 70.432755, 71.20065 , 56.81485 ,
       68.51811 , 68.28672 , 74.3441  , 72.52025 , 68.38485 , 73.09153 ,
       66.07627 , 69.86245 , 73.13646 , 51.258728, 66.54002 , 70.92247 ,
       63.91761 , 52.164833, 74.539116, 71.30629 , 72.84301 , 63.239166,
       51.390068, 74.04961 , 60.693752, 73.22912 , 74.58801 , 68.39462 ,
       73.83229 , 70.582466, 62.177853, 81.956055, 72.19832 , 64.53951 ,
       64.68613 , 74.25747 , 68.75119 , 66.63135 , 76.2344  , 72.72273 ,
       53.969097, 75.68926 , 75.29455 , 75.6966  , 72.86621 , 47.96772 ,
       82.8276  , 70.339005, 57.933857, 75.062126, 75.58726 , 56.679504],
      dtype=float32)
```

In [96]:

```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from math import sqrt

RMSE = float(format(np.sqrt(mean_squared_error(y_test, y_predict)),'.3f'))
MSE = mean_squared_error(y_test, y_predict)
MAE = mean_absolute_error(y_test, y_predict)
r2 = r2_score(y_test, y_predict)
```

```
print('RMSE =',RMSE, '\nMSE =',MSE, '\nMAE =',MAE, '\nR2 =', r2)
```

```
RMSE = 2.267
MSE = 5.140228127334427
MAE = 1.675479368730025
R2 = 0.9405115978232569
```

In [97]:

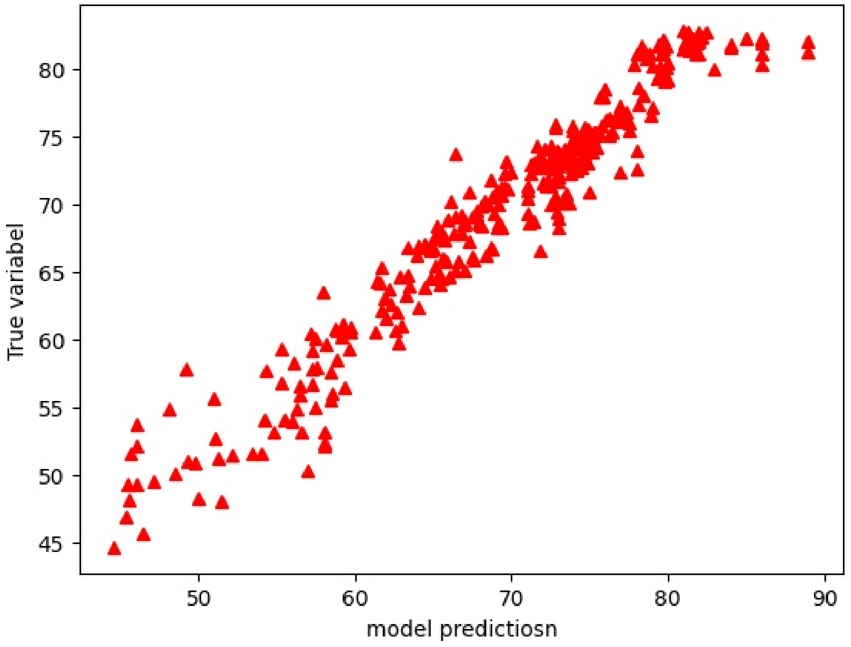
```
plt.plot(y_test, y_predict, '^', color = "r")
plt.xlabel('model predictiosn')
plt.ylabel('True variabel')
plt.show
```

Out[97]:

```
matplotlib.pyplot.show
def show(*args, **kwargs)

Display all open figures.

Parameters
----------
block : bool, optional
    Whether to wait for all figures to be closed before returning.
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js