DAT565/DIT407 Assignment 4

David Duong
thuan@chalmers.se

Savinjith Walisadeera
savwal@chalmers.se

September 30, 2024

Problem 1

We perform a 75:25 train-test split randomly of the data. We use this ratio because it gives us a sufficient amount of points to train the model and test it. We select randomly to make sure we do not fit specifically to any trends.

Problem 2

a) We identified the strongest linear relationship to LEB was Human Development Index. This parameter had the highest Pearson product-moment correlation coefficient.
b) The coefficient of determination was 0.8483

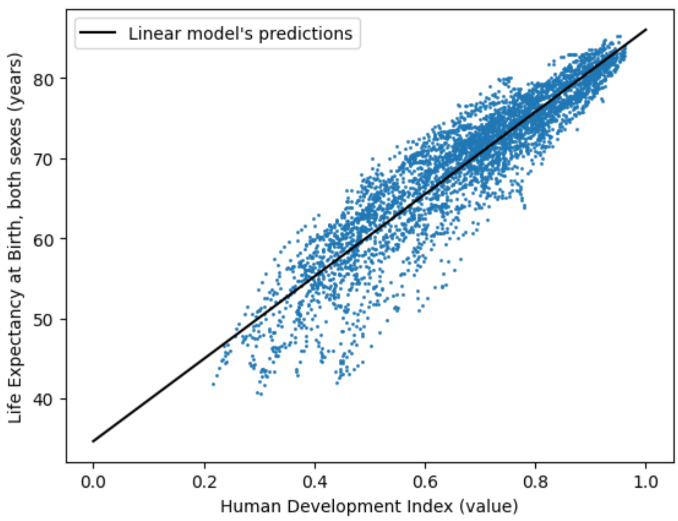

Figure 1: Scatter plot of human development index to life expectancy after birth with a single-variable linear model.

c) The correlation between HDI and LEB is: 0.912. The mean square error between the the test data and trained model was 14.60.

d) Human development index is a coefficient which measures: healthy and long lives, education level and standard of living, which are normalized to compute the index between 0 and 1 [1]. These factors are play a large role for the general public of a countries life expectancy after birth. Therefore it makes sense that HDI has a strong correlation coefficient with LEB.

Problem 3

We look at large difference in size between the min and max values and found that GDP per capita and LEB seemed logarithmicly correlated. Pearson coefficient before transformation: 0.651. Pearson coefficient after transformation: 0.837.

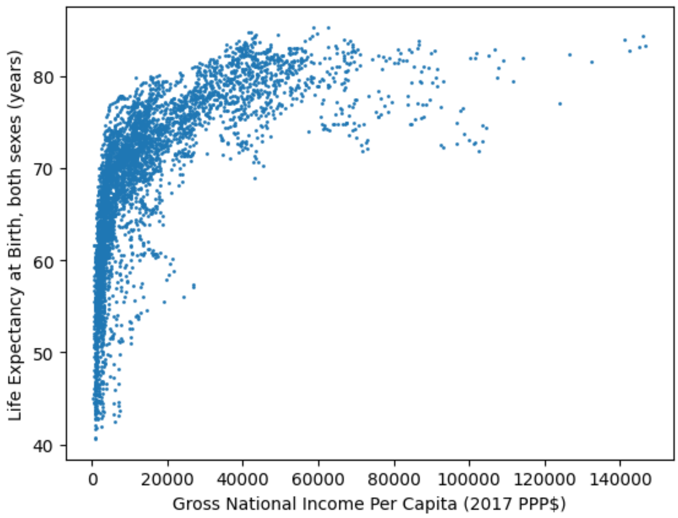

Figure 2: Scatter plot of GDP per capita to life expectancy after birth before transformation.

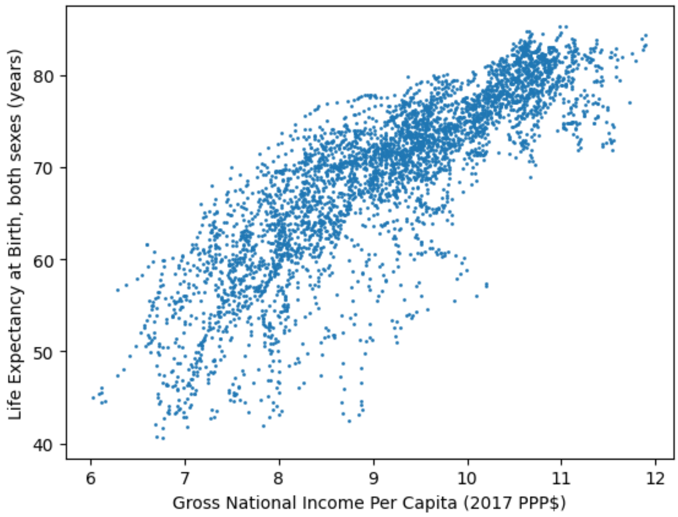

Figure 3: Scatter plot of GDP per capita to life expectancy after birth after transformation.

Problem 4

We chose the variables by using our knowledge on what impacts a country's life expectancy after birth combined with the absolute value of the Pearson product-moment correlation coefficient between all parameters against life expectancy after birth. With these variables we created a multiple linear regression model. The variables in the model are: expected years of schooling, crude birth rate, net reproduction rate gross national income per capita and gender development index. When we had missing values for a variable we omitted that row from the data set.

We have the Pearson correlation = 0.887, R^2=0.867, mean square error = 11.821, and the intercept value = 65.864.

Variables	k
Expected Years of Schooling	0.492
Crude birth rate	-1.198
Net reproduction rate	13.932
Gross national income per capita	1.873
Gender development index	-11.234

Table 1: Table of coefficients for each variable in the model.

This model had better model evaluation parameters than the single variable linear regression.

References

[1] Human Development Index (HDI) *United Nations Development Program*, 2024.

A Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv("life_expectancy.csv")
df

#Split the data 75-25
train_data, test_data = train_test_split(df, test_size=0.25, random_state=12)
train_data

data_no_country = train_data.drop('Country', axis=1)
cov_matrix = data_no_country.cov()
variances = data_no_country.var()
standard_deviations = np.sqrt(variances)


pearson_coef = cov_matrix['Life Expectancy at Birth, both sexes (years)']/(standard_deviations*standard_deviations['Life Expectancy at Birth, both sexes (years)'])

#We find that Human Development Index (value) has the strongest linear relationship with LEB
pearson_coef.sort_values()

train_LEB = train_data['Life Expectancy at Birth, both sexes (years)']
train_HDI = train_data['Human Development Index (value)']

strongest = train_data[['Life Expectancy at Birth, both sexes (years)','Human Development Index (value)']]

train_HDI.values.reshape(-1,1)
```

```python
model = LinearRegression().fit(train_HDI.values.reshape(-1,1), train_LEB.values)
print(model.coef_,model.intercept_)
R2 = r2_score(train_LEB.values.reshape(-1,1), model.predict(train_HDI.values.reshape(-1,1)))
R2

plt.scatter(train_HDI,train_LEB, s=1)
plt.plot(np.linspace(start=0,stop=1,num=1000),model.predict(np.linspace(start=0,stop=1,num=1000).reshape(-1,1)), color='black', label="Linear models' predictions")
plt.ylabel('Life Expectancy at Birth, both sexes (years)')
plt.xlabel('Human Development Index (value)')
plt.legend()

test_LEB = test_data['Life Expectancy at Birth, both sexes (years)']
test_HDI = test_data['Human Development Index (value)']

test_predicted_vals = model.predict(test_HDI.values.reshape(-1,1))
test = pd.DataFrame(test_LEB)
test_combined = test.assign(predicted_values=test_predicted_vals)
test_covariance = test_combined.cov()
var_test = test_LEB.var()
var_pred = test_predicted_vals.var()
test_corr = test_covariance['predicted_values']['Life Expectancy at Birth, both sexes (years)']/(np.sqrt(var_test*var_pred))

print('correlation: ',test_corr)
print('MSE: ',mean_squared_error(test_LEB.values,test_predicted_vals))

#Find large differances
i = 0
vals = []
for col in train_data.drop('Country', axis=1):
    diff = train_data.drop('Country', axis=1).max(axis=0)[col]-train_data.drop('Country', axis=1).min(axis=0)[col]
    vals.append((diff,i))
    i += 1

def return_first(list):
    return list[0]
```

```python
vals.sort(reverse=True,key=return_first)
vals

plt.scatter(np.log(train_data['Gross National Income Per Capita (2017 PPP$)']),train_data['Life Expectancy at Birth, both sexes (years)'],s=1)
plt.xlabel('Gross National Income Per Capita (2017 PPP$)')
plt.ylabel('Life Expectancy at Birth, both sexes (years)')

print('Pearson coefficient before transformation: ', pearson_coef['Gross National Income Per Capita (2017 PPP$)'])

log_GDP = np.log(train_data['Gross National Income Per Capita (2017 PPP$)'])

dframe = pd.DataFrame(train_LEB)
combined = dframe.assign(GDPPC=log_GDP)
GDP_covariance = combined.cov()
var_train = train_LEB.var()
var_GDP = log_GDP.var()

test_corr = GDP_covariance['GDPPC']['Life Expectancy at Birth, both sexes (years)']/(np.sqrt(var_train*var_GDP))

print('Pearson coefficient after transformation: ', test_corr)

#Expected Years of Schooling (years)
#Gross National Income Per Capita (2017 PPP$)
#Crude Birth Rate (births per 1,000 population)
#Adolescent Birth Rate (births per 1,000 women ages 15-19)
#Net Reproduction Rate (surviving daughters per woman)
#Mean Years of Schooling (years)
parameters = ['Expected Years of Schooling (years)',
              'Gross National Income Per Capita (2017 PPP$)',
              'Crude Birth Rate (births per 1,000 population)',
              'Net Reproduction Rate (surviving daughters per woman)',
              'Gender Development Index (value)']

train_data = train_data.dropna(subset=parameters)


```

```python
train_LEB = train_data['Life Expectancy at Birth, both sexes (years)']

train_EYS = train_data['Expected Years of Schooling (years)']
train_GNIPC = np.log(train_data['Gross National Income Per Capita (2017 PPP$)'])
train_CBR = train_data['Crude Birth Rate (births per 1,000 population)']
train_NRR = train_data['Net Reproduction Rate (surviving daughters per woman)']
train_GDI = train_data['Gender Development Index (value)']

arr = np.array([train_EYS, train_GNIPC, train_CBR, train_NRR, train_GDI])
arr = np.transpose(arr)

model = LinearRegression().fit(arr, train_LEB.values)
print(model.coef_,model.intercept_)

R2 = r2_score(train_LEB.values.reshape(-1,1), model.predict(arr))
print('R2-score: ',R2)


multi_train_predicted_vals = model.predict(arr)
multi_train = pd.DataFrame(train_LEB)
multi_train_combined = multi_train.assign(predicted_values=multi_train_predicted_vals)
train_covariance = multi_train_combined.cov()
var_train = train_LEB.var()
var_pred = multi_train_predicted_vals.var()
test_corr = test_covariance['predicted_values']['Life Expectancy at Birth, both sexes (years)']/(np.sqrt(var_test*var_pred))

print('correlation: ',test_corr)
print('MSE: ',mean_squared_error(train_LEB.values, multi_train_predicted_vals))
```