Linear Regression Analysis Project

EE 381 Section 02 Probability & Stats Computing

Professor: Jody Shu

Group 2

Authors: Nhi Pham, Hadi Al Lawati

Date: 5/12/2022

I. **Introduction**:

Generally, when evaluating the quality of life of a country, there are a lot of factors that are needed to take into consideration such as its economy, resources, diseases, healthcare programs, mortality, average age of its citizens, etc,...For this project, to study how the quality of life of a country can be affected, we choose 3 elements that we think would be the major contributors and determine the relationship between them. Those elements are how well the country manages its resources, GDP value, which delivers an economic snapshot of a country and estimates the size of an economy and growth rate, and the life expectancy of the citizens. The outcome of this project can potentially be used to address issues relating to improving the life quality of people around the world.

II. **Problem Description**:

In this report, we will examine whether a country's utilization of its resources, also known as ‘income composition of resources’, and ‘GDP’ could be a factor in improving people's life expectancy. In other words, this report will discuss how these specific factors would help increase or decrease the period that a person may expect to live?

III. **Objectives**:

To approach this problem, we will use computer programming and open-sourced data to find the effect of GDP and income composition of resources on human life expectancy. The analysis would be based on the concept of multiple linear regression. The income composition of resources and the average GDP of countries are two independent variables, while the life expectancy of citizens is the dependent variable.

IV. Report body:

A. Methodologies used:

1. We will use the dataset obtained from Kaggle website: https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who , which provides annual information about the GDP, resources, and life expectancy of countries all over the world from a period of 2000 to 2015
2. Python programming language

B. Code:

1. Loading and reading data from the source using the code:

```
file_name = "project_data.csv"
x1 = "Income composition of resources"
x2 = "GDP"
y = "Life expectancy"

df = pd.read_csv(file_name)
```

2. Compute b_0, b_1, b_2:

Multiple linear regression is in form: **X * B = Y**

- X is a nx3 matrix, with the first column contains 1's, the second column contains value of x_{i1}, and the third column contains value of x_{i2}
- B is a 3x1 matrix containing b_0, b_1, b_2 that we need to find
- Y is a nx1 matrix that contains y values

To compute for b_0, b_1, b_2, we use the formula: $\mathbf{B = (X'X)^{-1} X'Y}$

In the program (Figure 1):

- X1, X2, Y are arrays storing the values of x1, x2, and y when extracting them from the csv file
- x is the numpy nx3 matrix of X
- y is the numpy nx1 matrix of Y

```
for index, row in df.iterrows():
    X1.append(row[x1])
    X2.append(row[x2])
    Y.append(row[y])
    temp = [1, row[x1], row[x2]]
    temp1 = [row[y]]
    final_list_x.append(temp)
    final_list_y.append(temp1)
    n+=1

x = np.array(final_list_x)#X
y = np.array(final_list_y)#Y
transposedx = np.transpose(x)#X'

test_x = np.dot(transposedx, x)#(X'X)
test_x_inv = np.linalg.inv(test_x) #(X'X)^-1
test_y = np.dot(transposedx, y)#(X'Y)
b = np.dot(test_x_inv, test_y)#(X'X)^-1*(X'Y)
```

Figure 1

After successfully creating matrix x and matrix y, we compute X'X and X'Y by using numpy dot product function and numpy transpose function. Then, we calculate $(X'X)^{-1}$ using np.linalg.inv(). Finally, we multiply together $(X'X)^{-1}$ X'Y to get matrix b which contains values of b_0, b_1, b_2 . From matrix b, we can build the regression line as below (Figure 2)

```
[[4.96081752e+01]
 [3.00871830e+01]
 [1.04520949e-04]]
----------------
b0: 49.60817518266464
b1: 30.087183039753317
b2: 0.00010452094900933289
y =  49.60817518266464  +  30.087183039753317 * x1 +  0.00010452094900933289 * x2
```

Figure 2

3. Compute SSE and MSE:

Using for loop, we compute the estimated value of y based on the regression line and stored in an numpy array called y_hat (Figure 3)

```
delta_y = 0.0
y_hat = []

for index, row in df.iterrows():
    delta_y = (b0+b1*(row[x1])+b2*(row[x2]))#y-hat
    y_hat.append([delta_y])#y-hat

y_hat_np = np.array(y_hat)
```

Figure 3

In figure 4, we find the error vector by the formula: **e = y - y_hat**. With e, we can compute SSE by the formula: **SSE = e'e**. After we get SSE, we can calculate **MSE = SSE/(n-3).** The results are printed out in figure 5

```
error_e = y-y_hat_np
print("---------------------------")
print("error: ", error_e)

print("-----------------------------------")
SSE = np.dot(np.transpose(error_e), error_e)

print("SSE: ", SSE)

MSE = (SSE/(n-3))

print("MSE: ", MSE)
```

Figure 4

```
---------------------------
error:  [[  0.91899681]
 [ -4.09371393]
 [ -3.9151818 ]
 ...
 [-17.66139644]
 [-17.21274121]
 [-16.72322309]]
-----------------------------------
SSE:  [[101444.59543233]]
MSE:  [[40.97116132]]
-------------------
```

Figure 5

4. Find s.e(b_0), s.e(b_1), s.e(b_2):

The standard error of b_i is square root of MSE multiplied by the square root of the i^{th} diagonal element of $(X'X)^{-1}$

```
lamda1 = test_x_inv[0][0]
lamda2 = test_x_inv[1][1]
lamda3 = test_x_inv[2][2]
se_b0_hat = math.sqrt(MSE) * math.sqrt(lamda1)
se_b1_hat = math.sqrt(MSE) * math.sqrt(lamda2)
se_b2_hat = math.sqrt(MSE) * math.sqrt(lamda3)
print("se_b0_hat = ", se_b0_hat)
print("se_b1_hat = ", se_b1_hat)
print("se_b2_hat = ", se_b2_hat)
```

Figure 6

We have stored the value of $(X'X)^{-1}$ computed in step 2 in a matrix called test_x_inv. In figure 6, we use variables lamda1, lamda2, lamda3 to store the first, second and third diagonal element of $(X'X)^{-1}$. The results are printed in figure 7

```
se_b0_hat =  0.4223622989397668
se_b1_hat =  0.6832175683164454
se_b2_hat =  1.0139577309562076e-05
```

Figure 7

5. Compute coefficient of correlation between x_1 and y, and between x_2 and y:

We use the formula: **r =** $\frac{\Sigma x_i y_i - n.\bar{x}.\bar{y}}{\sqrt{\Sigma(x_i)^2 - n(\bar{x})^2}\sqrt{\Sigma(y_i)^2 - n(\bar{y})^2}}$ to find the coefficient of correlation

First, we use for loop to compute for $\Sigma x1,\ \Sigma x2,\ \Sigma y,\ \Sigma x1^2, \Sigma x2^2,\ \Sigma y^2,\ \Sigma x1y,\ \Sigma x2y$ and store the values into variables sumx1, sumx2, sumy, sumx1_2, sumx2_2, sumy_2, sumx1y, sumx2y accordingly. We can also count for the size of sample data and

store it in an int variable n. Then, we calculate $\overline{x_i} = \frac{\Sigma x_1}{n}$ and $\overline{y} = \frac{\Sigma y}{n}$.

Finally, we plug all the numbers in the r formula (Figure 8) and get the results in Figure 9

```
sumx1 = sumx2 = sumy = sumx1_2 = sumx2_2 = sumy_2 = sumx1y = sumx2y = 0
b0 = b1 = b2 = 0
n = 0

for index, row in df.iterrows():
    sumx1 += row[x1]
    sumx2 += row[x2]
    sumy += row[y]
    sumx1_2 += row[x1]**2
    sumx2_2 += row[x2]**2
    sumy_2 += row[y]**2
    sumx1y += row[x1]*row[y]
    sumx2y += row[x2]*row[y]
```

```
x1_bar = sumx1/n
x2_bar = sumx2/n
y_bar = sumy/n

print("---Coefficient of correlation bw x1 and y----")
r_x1y = (sumx1y - n*x1_bar*y_bar)/(math.sqrt(sumx1_2 - n*((x1_bar)**2)) * (math.sqrt(sumy_2) - n*((y_bar)**2)))
print("r_x1y = ", r_x1y)

print("---Coefficient of correlation bw x2 and y----")

r_x2y = (sumx2y - n*x2_bar*y_bar)/(math.sqrt(sumx2_2 - n*((x2_bar)**2)) * (math.sqrt(sumy_2) - n*((y_bar)**2)))
print("r_x2y = ", r_x2y)
```

Figure 8

```
---Coefficient of correlation bw x1 and y----
r_x1y =  -2.94923114194712e-05
---Coefficient of correlation bw x2 and y----
r_x2y =  -1.851705706493134e-05
```

Figure 9

6. Hypothesis testing for linearity of b_1 and b_2 at α= 0.01:

   Two-side testing, df = n-2>120, $\alpha/2$=0.005

   t_critial = $t_{(0.995,\ \infty)}$=2.58

t_stat $= \frac{\widehat{bi}}{s.e(bi)}$

```
t1 = b1/se_b1_hat
t2 = b2/se_b2_hat
print("t1 = ", t1)
print()
print("t2 = ", t2)
```

```
t1 =  44.03748444861104

t2 =  10.308215600936832
```

a) Test H_0: $b_1 = 0$ vs H_A: $b_1 \neq 0$

By using computer programming, we get t1 = 44.0375

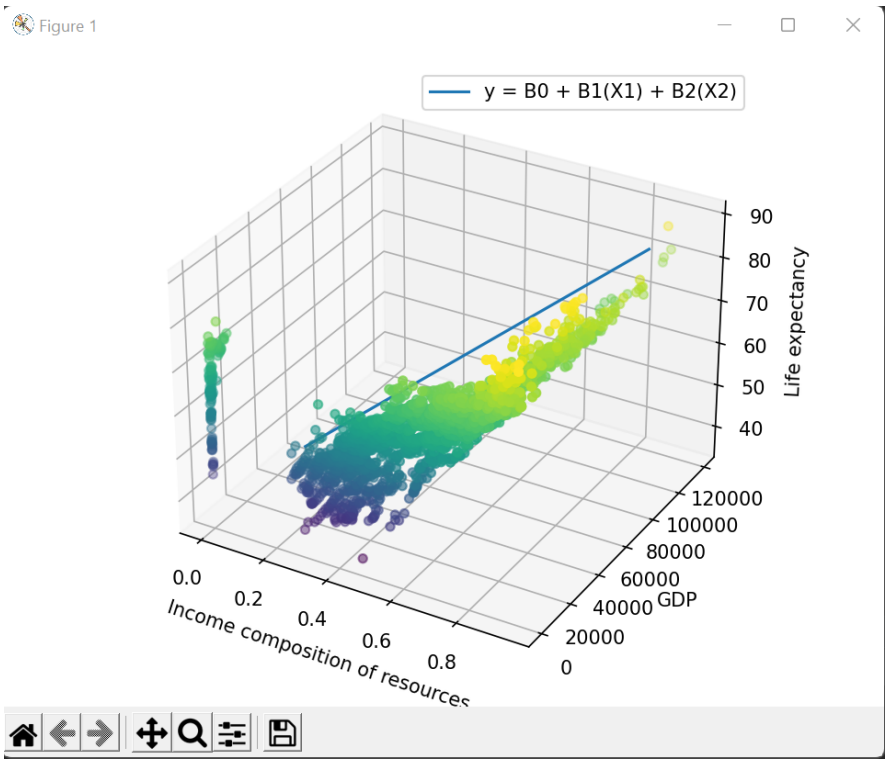
Because t1 > t_critical, meaning test statistic fall inside rejection area, we reject H_0

b) Test H_0: $b_2 = 0$ vs H_A: $b_2 \neq 0$

By using computer programming, we get t2 = 10.3082

Because t2 > t_critical, meaning test statistic fall inside rejection area, we reject H_0

C. **Data visualization:** This is the 3D graphs of the linear regression line along with scatter data points showing relationship between the income composition of resources of a country and its GDP versus the life expectancy of its citizen



V. Conclusion:

From the hypothesis testing, we can conclude that both b1 and b2 would not equal 0.

Also, by looking at the coefficient of correlation r between x_1 and y and between x_2 and y, since both r_x1y and r_x2y are negative and very small (almost equal 0), we can conclude that:

- the relationship between 'Income composition of resources' and 'Life expectancy' are inversely related, or nearly have no relationship
- the relationship between 'GDP' and 'Life expectancy' are inversely related, or nearly have no relationship

However, from the 3D graph, we can see that the regression line goes up when 'Income composition of resources' and 'GDP' increase. Therefore, it

supports our theory which is ‘income composition of resources’ and ‘GDP’ are factors in improving people's life expectancy.

Contribution:

- Nhi Pham:
    - Percentage of effort: 100%
    - Tasks: coding and writing report
- Hadi Al Lawati:
    - Percentage of effort:0%
    - Tasks: