

DAT565/DIT407 Assignment 4

Ola Bratt
ola.bratt@gmail.com

Patrick Attimont
patrickattimont@gmail.com

2024-02-15

This paper is addressing the assignment 3 study queries within the *Introduction to Data Science & AI* course, DIT407 at the University of Gothenburg and DAT565 at Chalmers. The main source of information for this project is derived from the lectures and Skiena [2]. Assignment 4 is about correlation and linear regression.

Problem 1: Splitting the data

The dataset is large enough to be separated into a train and a test set. We use the function `train_test_split` with a test size of 0.2.

Problem 2: Single-variable model

To identify the variable with the strongest linear relationship with the target variable, we use the `corr` function specifying the Pearson method to get a correlation matrix between all the variables (Fig 1), and take the column corresponding to the target variable.

The variable with the highest absolute pearson coefficient is the Human development index, with a value of 0.92. Fig 2 shows the linear regression applied on the test set, using the model built on the train set. The model has a slope of 48.2, an intercept of 37.3, and a determination coefficient of 0.86. These values obviously slightly change depending on the seed used to split the data. The mean squared error between the test and predicted sets is 9.45.

The human development index measures a country's social and economic development. As specified on UNDP's website ([1]), it is calculated based on the following factors: life expectancy, education, and standard of living. Thus the strong relationship between the two variables comes from the fact that life expectancy at birth is used to calculate the human development index.

Problem 3: Non-linear relationship

To explore variables with non-linear relationships, we initially assess Spearman correlation coefficients, Figure 3. Initially, we eliminate the variable exhibiting the strongest Pearson correlation, namely 'Human Development Index (value)'. Subsequently, we examine the Spearman correlation.

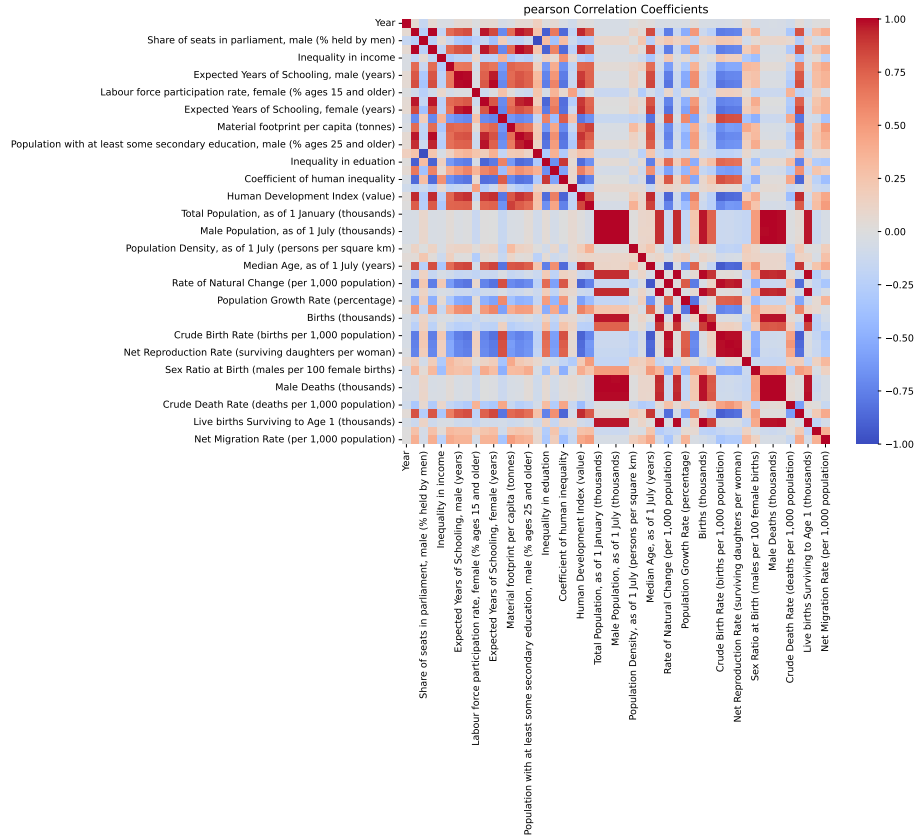


Figure 1: Correlation Pearson

Our analysis reveals that 'Median Age, as of 1 July (years)' exhibits the most robust Spearman correlation. Subsequently, we develop a linear model using this variable to determine the mean squared error, which amounts to 13.58, Figure 5a.

Next, we apply transformations to the variable using logarithmic, square root, and reciprocal functions, respectively Figure 4. For each transformation, we train a linear model to ascertain the resulting mean squared error. The logarithmic transformation yields the lowest mean squared error, measuring 11.52. Figure 5b.

Before transformation, the Pearson correlation registers at 0.898, while after transformation, it increases marginally to 0.913.

Problem 4: Multiple linear regression

We utilized Spearman correlation analysis as a method to distinguish variables exhibiting strong correlations with our target. By setting a predetermined threshold, we aimed to identify variables exerting significant influence on the target variable.

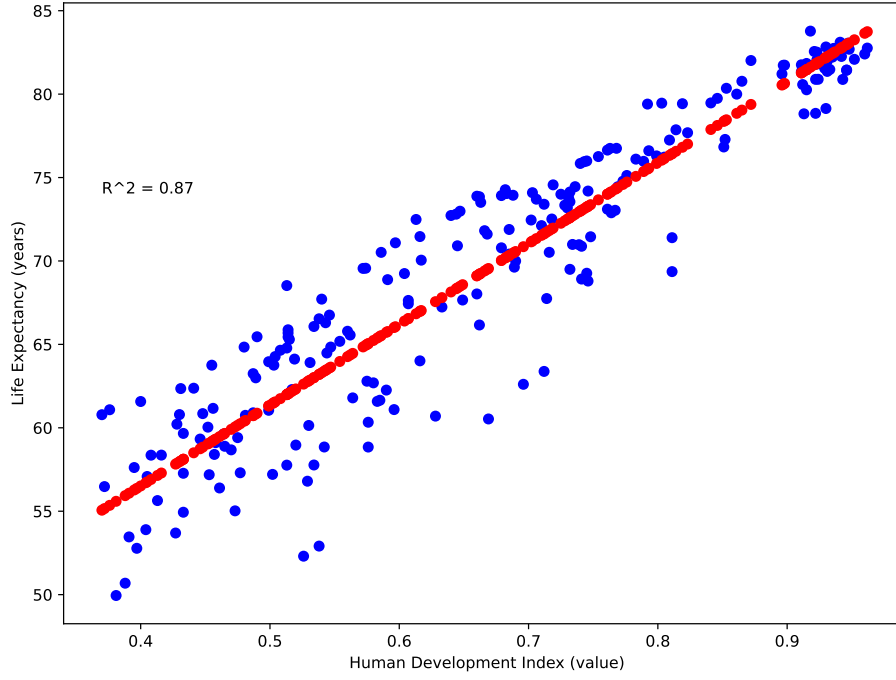


Figure 2: Linear Regression Human Development Index (value)

After conducting experiments with different threshold values, we observed that a threshold of 0.85 yielded satisfactory results, achieving a balance between inclusivity and relevance.

The variables identified through this process are: Expected Years of Schooling for females, the Coefficient of Human Inequality, Gross National Income Per Capita for the year 2017, Median Age as of 1 July, Rate of Natural Change per 1,000 population, Crude Birth Rate measured in births per 1,000 population, Total Fertility Rate denoting live births per woman, and Net Reproduction Rate denoting surviving daughters per woman.

Upon conducting a multiple linear regression analysis on the selected variables, we obtained a mean squared error of 2.03, indicative of the model's predictive accuracy. Additionally, the model achieved an R-squared score of 0.95, suggesting a high degree of variance explained by the model.

The coefficients derived from the regression model provide insights into the strength and direction of the relationships between the independent variables and the target variable. The coefficients for the model are as follows: [19, -571, 243338, 61, 1.7, -2.1, 1.9, 3], with an intercept of 78.24.

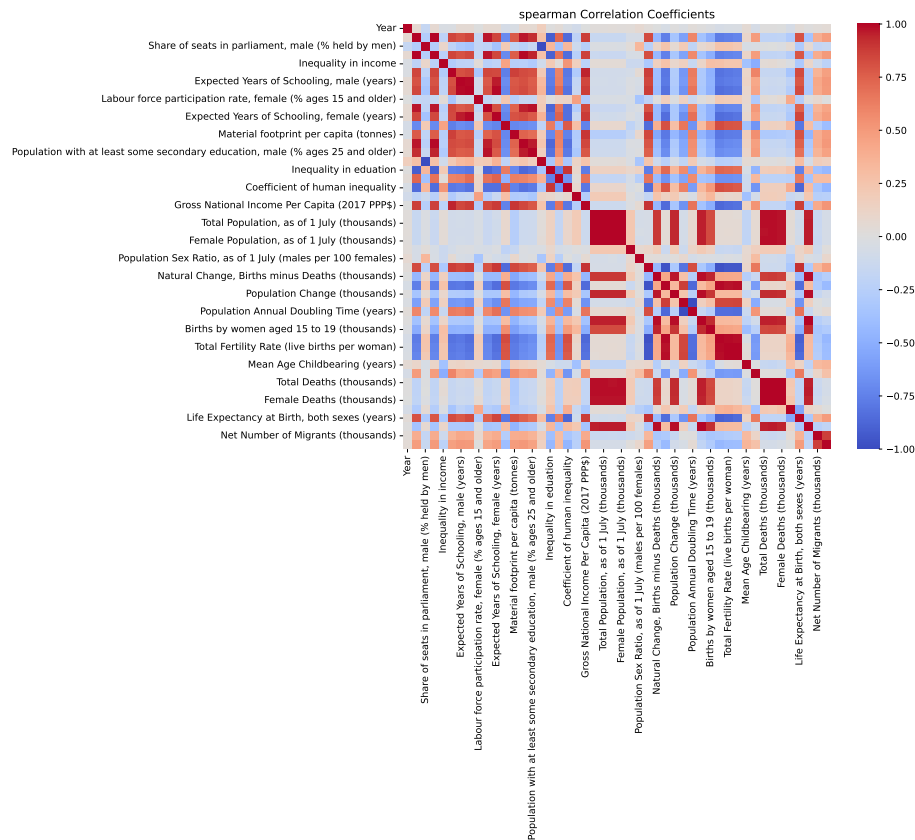


Figure 3: Correlation Spearman

References

- [1] Human development reports. *Human Development Index (HDI)*. Retrieved 2024-02-14. 2023. URL: <https://hdr.undp.org/data-center/human-development-index#/indicies/HDI>.
- [2] Steven S Skiena. *The Data Science Design Manual*. Retrieved 2024-01-20. 2024. URL: <https://ebookcentral.proquest.com/lib/gu/detail.action?docID=6312797>.

Appendix: Source Code

```

1 from matplotlib import pyplot
2 import numpy as np
3 import pandas as pd
4 from sklearn.model_selection import train_test_split
5 from sklearn.linear_model import LinearRegression
6 import seaborn as sns
7 from sklearn.metrics import mean_squared_error
8 from sklearn.metrics import r2_score
9

```

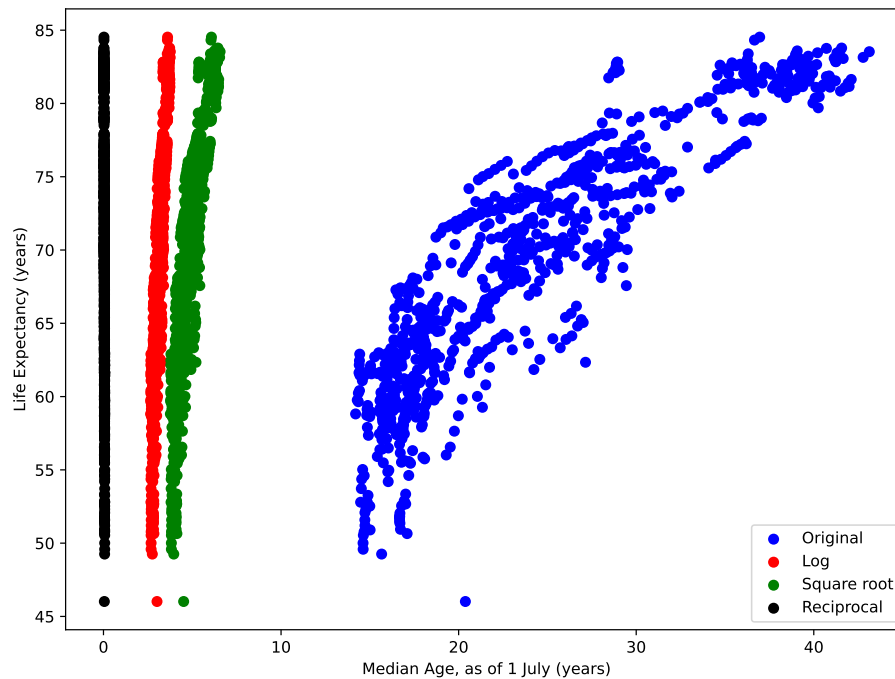
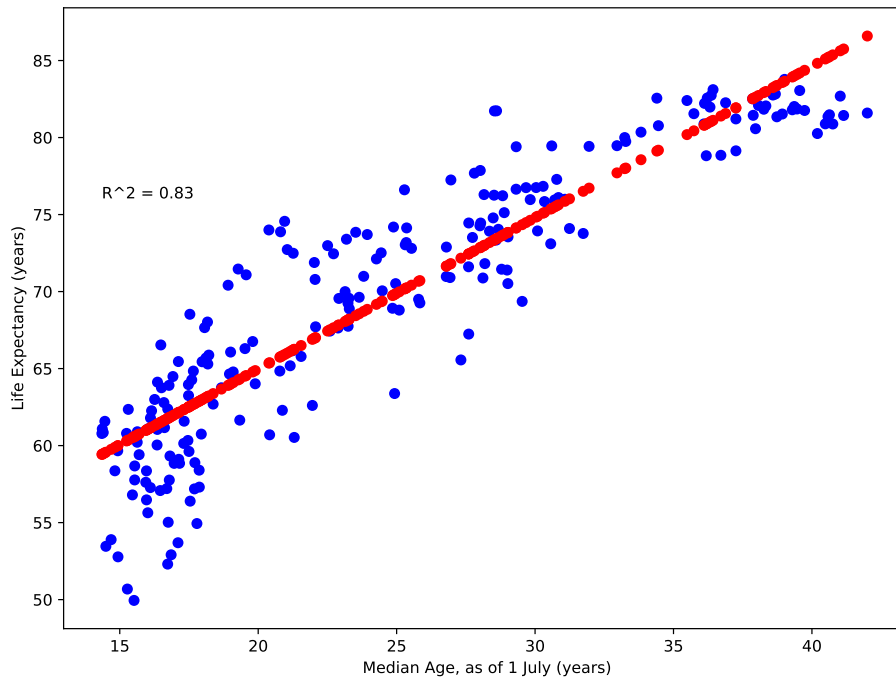


Figure 4: Linear transformation

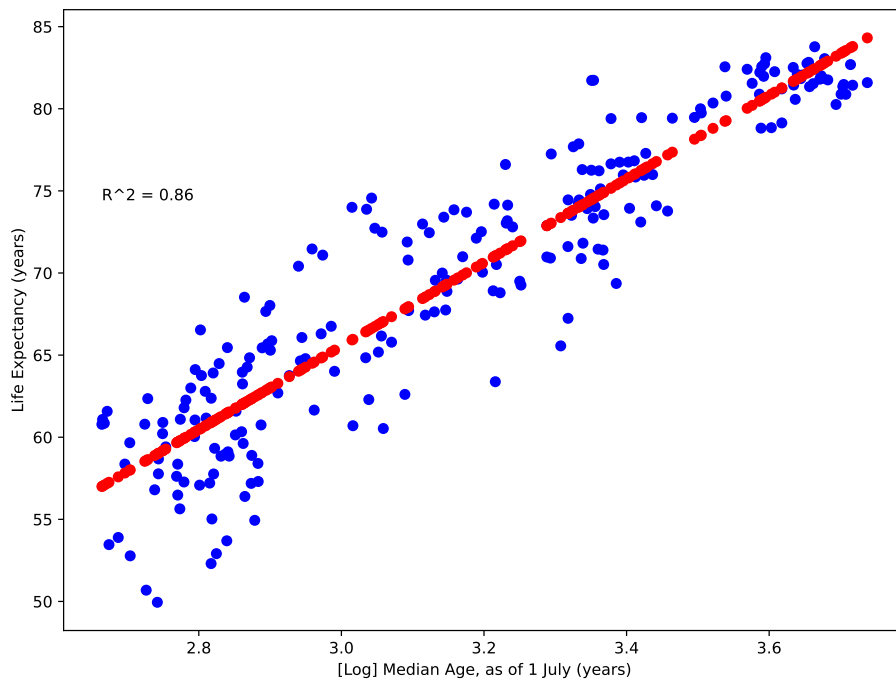
```

10 def calculate_correlation(data, variable, method):
11     # Compute Pearson correlation coefficients
12     correlation_matrix = data.corr(method = method)
13
14     # Extract correlation coefficients of the target variable (life
15     # ↳ expectancy)
16     correlation_with_life_expectancy = correlation_matrix[variable]
17     # Remove the target variable from the correlation coefficients
18     correlation_without_life_expectancy =
19     # ↳ correlation_with_life_expectancy.drop(variable)
20
21     # Find the variable with the highest absolute correlation
22     # ↳ coefficient
23     strongest_correlation_variable =
24     # ↳ correlation_without_life_expectancy.abs().idxmax()
25     strongest_correlation_coefficient =
26     # ↳ correlation_without_life_expectancy.abs().max()
27
28     print(f"The variable '{strongest_correlation_variable}' has the
29     # ↳ -strongest-" + method + f"-correlation-with-a-
30     # ↳ coefficient-of-{strongest_correlation_coefficient:.2f}."
31     # ↳ )
32
33     fig, ax = pyplot.subplots(figsize=(10, 8))
34     sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm')
35     ax.set_title(method + '-Correlation-Coefficients')
36     fig.savefig(method + "_correlation.pdf", bbox_inches='tight')
37
38     return strongest_correlation_variable, correlation_matrix

```



(a) Linear Regression Median Age (original)



(b) Linear Regression Median Age (log)

Figure 5: Linear Regression Median Age

```

32
33 def train_linear_regression_model(X_train, X_test, y_train, y_test,
    ↪ variables, prefix = ''):
34     # Train a linear regression model using the variable with the
    ↪ strongest correlation
35     model = LinearRegression().fit(X_train, y_train)
36     # Make predictions
37     y_pred = model.predict(X_test)
38     r2 = r2_score(y_test, y_pred)
39     _, rows = X_test.shape
40     if rows == 1:
41         fig, ax = pyplot.subplots(figsize=(8, 6), layout='
    ↪ constrained')
42         ax.scatter(X_test, y_test, color='blue')
43         ax.scatter(X_test, y_pred, color='red')
44         ax.set_xlabel(prefix + "-" + variables)
45         ax.set_ylabel('Life-Expectancy-(years)')
46         ax.text(0.1, 0.7, f'R^2={r2:.2f}', ha='center', va='
    ↪ center', transform=ax.transAxes)
47         filename = prefix + "_linear_regression_" + variables + ".
    ↪ pdf"
48         filename = filename.replace('-', '_').lower()
49         fig.savefig(filename, bbox_inches='tight')
50
51
52 mse = mean_squared_error(y_test, y_pred)
53 print("Trained-model-with-the-following-variables-" + prefix +
    ↪ ":-", variables)
54 print(f"The-mean-squared-error-for-is-{mse:.2f}.")
55 print(f"The-r2-score-is-{r2:.2f}.")
56 print("Coefficients:-", model.coef_)
57 print("Intercept-of-the-line:-", model.intercept_)
58
59 def transform_variable(X_train, y_train, correlation_variable):
60     pd.options.mode.chained_assignment = None
61
62     X_train_selected = X_train[[correlation_variable]]
63
64     X_train_selected['log'] = np.log(X_train[[correlation_variable
    ↪ ]])
65     X_train_selected['sqrt'] = np.sqrt(X_train[[
    ↪ correlation_variable]])
66     X_train_selected['reciprocal'] = 1/(X_train[[
    ↪ correlation_variable]])
67
68     fig, ax = pyplot.subplots(figsize=(8, 6), layout='constrained')
69     ax.scatter(X_train_selected.iloc[:, 0], y_train, color='blue',
    ↪ label='Original')
70     ax.scatter(X_train_selected['log'], y_train, color='red', label
    ↪ ='Log')
71     ax.scatter(X_train_selected['sqrt'], y_train, color='green',
    ↪ label='Square-root')
72     ax.scatter(X_train_selected['reciprocal'], y_train, color='
    ↪ black', label='Reciprocal')
73     ax.set_ylabel('Life-Expectancy-(years)')
74     ax.set_xlabel(correlation_variable)
75
76     ax.legend()
77     fig.savefig("linear_transformation.pdf", bbox_inches='tight')
78
79 file_path = "../life_expectancy.csv"
80 life_expectancy = pd.read_csv(file_path, sep=',').dropna()

```

```

81 LEB = 'Life-Expectancy-at-Birth,-both-sexes-(years)'
82 life_expectancy.set_index('Country', inplace=True)
83
84 life_expectancy_train, life_expectancy_test = train_test_split(
85     ↪ life_expectancy, test_size=0.2)
86
87 X_train = life_expectancy_train.drop(LEB, axis=1)
88 X_test = life_expectancy_test.drop(LEB, axis=1)
89 y_train = life_expectancy_train[LEB]
90 y_test = life_expectancy_test[LEB]
91
92
93 strongest_pearson_correlation_variable, correlation_pearson =
94     ↪ calculate_correlation(life_expectancy_train, LEB, 'pearson')
95
96 train_linear_regression_model(X_train[[
97     ↪ strongest_pearson_correlation_variable]], X_test[[
98     ↪ strongest_pearson_correlation_variable]], y_train, y_test,
99     ↪ strongest_pearson_correlation_variable)
100
101 strongest_spearman_correlation_variable, correlation_spearman =
102     ↪ calculate_correlation(life_expectancy_train.drop(
103     ↪ strongest_pearson_correlation_variable, axis=1), LEB, '
104     ↪ spearman')
105
106 train_linear_regression_model(X_train[[
107     ↪ strongest_spearman_correlation_variable]], X_test[[
108     ↪ strongest_spearman_correlation_variable]], y_train, y_test,
109     ↪ strongest_spearman_correlation_variable)
110
111 transform_variable(X_train, y_train,
112     ↪ strongest_spearman_correlation_variable)
113
114 train_linear_regression_model(np.log(X_train[[
115     ↪ strongest_spearman_correlation_variable]]), np.log(X_test[[
116     ↪ strongest_spearman_correlation_variable]]), y_train, y_test,
117     ↪ strongest_spearman_correlation_variable, "[Log]")
118
119 train_linear_regression_model(np.sqrt(X_train[[
120     ↪ strongest_spearman_correlation_variable]]), np.sqrt(X_test[[
121     ↪ strongest_spearman_correlation_variable]]), y_train, y_test,
122     ↪ strongest_spearman_correlation_variable, "[Sqrt]")
123
124 train_linear_regression_model(1/(X_train[[
125     ↪ strongest_spearman_correlation_variable]]), 1/(X_test[[
126     ↪ strongest_spearman_correlation_variable]]), y_train, y_test,
127     ↪ strongest_spearman_correlation_variable, "[Reciprocal]")
128
129
130 threshold = 0.85
131 correlation_spearman_no_LEB = correlation_spearman.drop([LEB])
132
133 relevant_variables = correlation_spearman_no_LEB[abs(
134     ↪ correlation_spearman_no_LEB['Life-Expectancy-at-Birth,-both-
135     ↪ sexes-(years)']) > threshold].index.tolist()
136
137 train_linear_regression_model(X_train[relevant_variables], X_test[
138     ↪ relevant_variables], y_train, y_test, relevant_variables)

```