DM954 Autonomous Sensing, Learning and Reasoning – Individual Assignment

Name: Barbara Weroba Obayi **Class Code:** DM954 **Date:** 23 October 2025

Linear Regression

Introduction

This assignment aimed to demonstrate how to apply linear regression in Python to understand the relationship between two variables. The task involved analysing a dataset generated by the `generate_data.py` script, identifying any issues within it, cleaning it algorithmically, and then creating a reliable linear model.

First, the dataset was inspected to identify potential issues and address them without compromising the client's data. Next, a cleaned and representative dataset was created and saved as `treated_set.dat`, which was used for the regression analysis.

The intercept (a) and slope (b) of the regression line were manually calculated using the least-squares method. The model was then visualised by plotting the regression line together with the cleansed dataset to help understand the relationship between the two variables.

Finally, the `scipy.stats.linregress` function was used to verify the results and obtain additional metrics such as R^2, p-value, and standard error. These results were compared with manual calculations to ensure consistency and accuracy. The results were then interpreted so that the client could understand what the regression results meant.

1 Data Issues and Treatment

Before performing the regression, The dataset was examined to check for any data quality issues that might affect the results [Haig, 2018].

1.1 Checking for Outliers

Boxplots for both the x and y variables were plotted to visually inspect extreme or unusual values (outliers). The boxplots provides a visual summary of how the data is distributed. The box itself shows where most of the data points lie, roughly representing the middle half of all values. The line inside the box marks the median, or middle value. The "whiskers" extend to the lowest and highest values that still fall within a normal range for the dataset, while any dots/circles appearing beyond those whiskers represent unusually high or low values which are potential outliers that differ significantly from the rest of the dataset.

Both boxplots showed that all data points were within the normal range; there were no dots or extreme values outside the whiskers. This means that there were no visible outliers in either x or y. The data appeared consistent and well-distributed; therefore, no extreme points needed to be removed or corrected [Tukey, 1977].

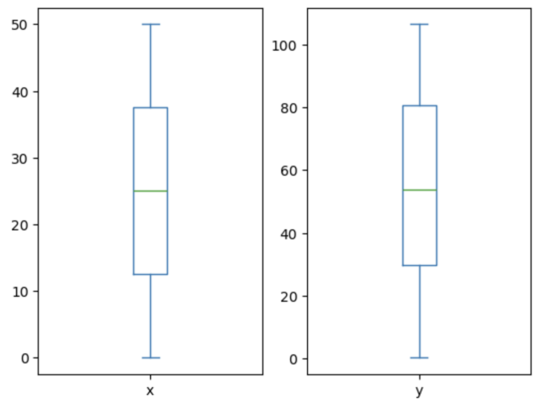

Figure 1: Boxplots for x and y showing no visible outliers.

To confirm this, A scatter plot of x against y was created to visually check the data pattern. The points formed a clear upward trend, reinforcing that the dataset followed a linear pattern and was ready for regression analysis.

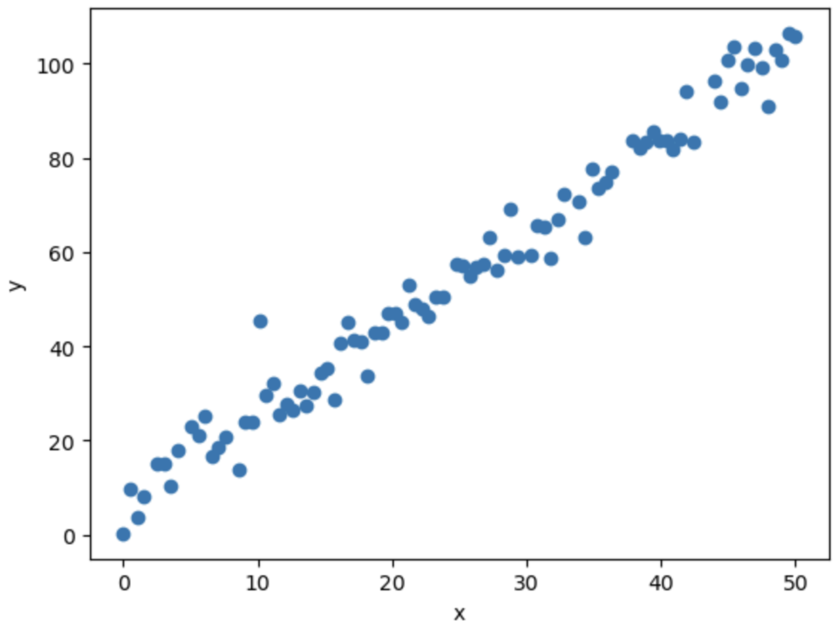

Figure 2: Scatter plot of y versus x showing a linear pattern.

1.2 Checking for Missing and Duplicate Values

Next, the `df.isna().sum()` function was used to check for missing or blank entries while the `df.duplicated().any()` function for duplicate records. The results showed:

- x had no missing values.
- y had 10 missing values.
- No duplicate rows were found in the dataset; the `df.duplicated().any()` function returned `False`.

Missing and duplicate values can affect the accuracy of regression results; therefore, it was important to determine whether these gaps occurred randomly or followed a pattern.

1.3 Investigating Missingness

To investigate whether the missing y values were random or related to x, box plots were plotted comparing the x values where y was missing vs. where it was present side-by-side. The idea was to visually compare how the values of x are distributed when y is missing versus when it's not. If the two boxplots looked very different, it would suggest that the missingness in y isn't random but somehow related to x.

This method was chosen because it is a simple and intuitive way to spot patterns in missingness during exploratory data analysis, especially when working with numerical variables. It gives a quick sense of whether the missing values are likely to be random (MCAR) or not, without needing formal statistical tests[Alsufyani et al., 2024].

In this case, the two boxplots looked very similar, with the same overall range, and their middle values (the medians) differed by approximately ~ 5 units. Considering that x ranges from 0 to 50 ($x \in [0, 50]$), this is a small difference.

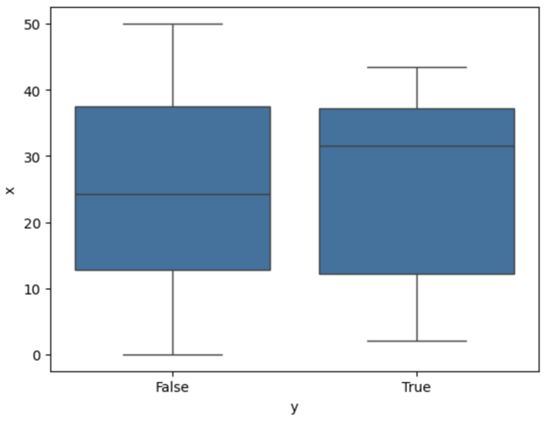

Figure 3: Boxplots showing that missing y values occurred randomly with respect to x.

This suggests that the missing y values occurred randomly and not because of anything specific about the x values (MCAR). Therefore, it was safe to remove those 10 rows before modeling, without worrying that it would affect the accuracy of the results [Emmanuel et al., 2021].

2 Final Clean Dataset

After confirming that the missing y values occurred randomly, listwise deletion was performed so that the dataset would include only complete entries. In Python, this was done using the `dropna()` function to remove the entire entry that was related to the rows where y was missing, and the cleaned data was then saved as `treated_set.dat`.

The cleaned dataset now contained 90 rows, each with valid x, y pairs. This ensured that the data used for regression was complete and reliable, allowing the model to focus only on the true relationship between the two variables without being influenced by missing values.

2.1 Correlation Analysis

Prior to using the linear regression model, the Pearson correlation coefficient was used to analyse the relationship between x and y. With values near 1 or -1 signifying a strong positive or negative correlation, respectively, this statistical measure quantifies the direction and strength of the linear association between the two variables.

According to the correlation matrix (Figure 4), there is a strong positive correlation between x and y ($r = 0.99$). This implies that x tends to increase proportionately as y increases, confirming the idea that a linear model is suitable for explaining their relationship.[Warne, 2020].

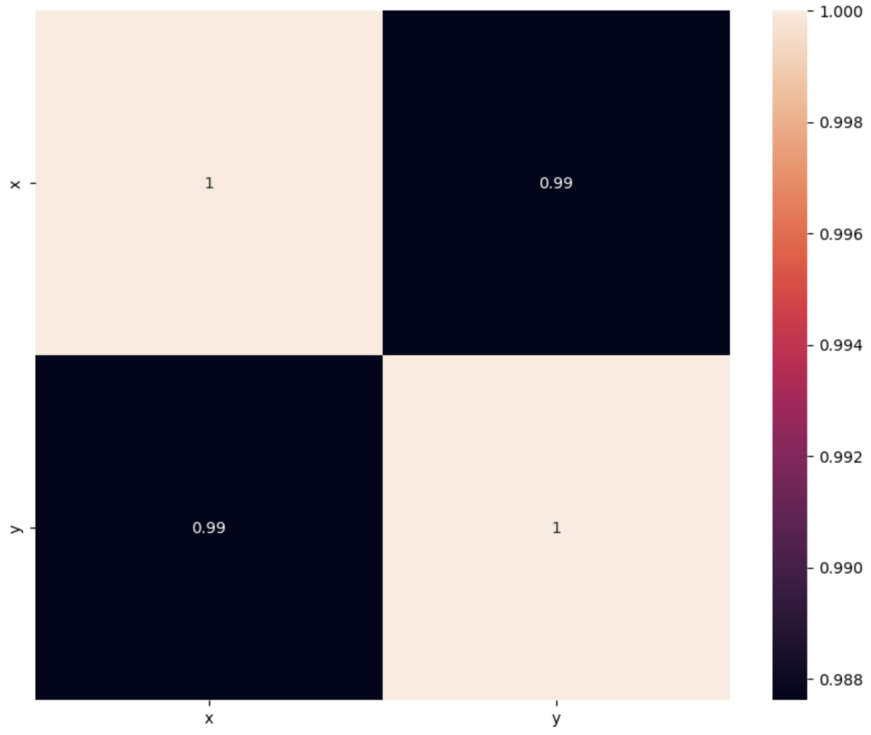

Figure 4: Correlation matrix and heatmap showing the strength of the linear relationship between x and y.

3 Linear Regression

To understand how x influences y, a linear regression model was used. This method helps draw a straight line through the data that best describes the general relationship between the two variables. The line follows the standard least-squares equation,

$$y = a + b\,x, \tag{1}$$

where:

a: the point where the line starts on the y-axis (called the intercept); when x is 0 and

b: the steepness of the line (called the slope), showing how much y changes when x increases by one unit

To find these two numbers (closed-form estimators), the formula that measures how far each point is from the average of all points was used: The slope b shows how tightly the data points move together; if they rise together, the slope will be positive. Once b is known, intercept a was calculated.

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \tag{2}$$

$$a = \bar{y} - b\,\bar{x}, \tag{3}$$

with $\bar{x} = \frac{1}{n}\sum_i x_i$ and $\bar{y} = \frac{1}{n}\sum_i y_i$.

When (2)–(3) was applied to `treated_set.dat`, the following was obtained:

$$\text{Intercept } (a) = 6.1300,$$
$$\text{Slope } (b) = 1.9500.$$

This means that when x is 0, the value of y starts at approximately 6.13 ($y \approx 6.13$), and each unit increase in x increases y by about 1.95.. The data shows a clear upward trend; as x increases, y also increases [Freedman, 2009].

4 Plotting the Regression Model

After calculating the regression line, it was plotted with the cleaned dataset to show how well the model fit the data. In the plot below:

- The blue dots represent the actual data points from the cleaned file, `treated_set.dat`.
- The blue line represents the regression model, which best describes the relationship between x and y.

The line runs almost exactly through the middle of the points, indicating a strong and consistent upward trend. This means that as x increases, y also increases at a steady rate. There is only a small amount of natural variation around the line, which is normal in real data. Overall, the plot confirms that the model provides an excellent summary of the relationship between the two variables.

5 Statistical Summary Using SciPy

To double-check the results from the manual calculation, the built-in `stats.linregress` function from the SciPy library was used. This function automatically computes the slope, intercept, and other key statistics that describe how well the data fit the regression line.

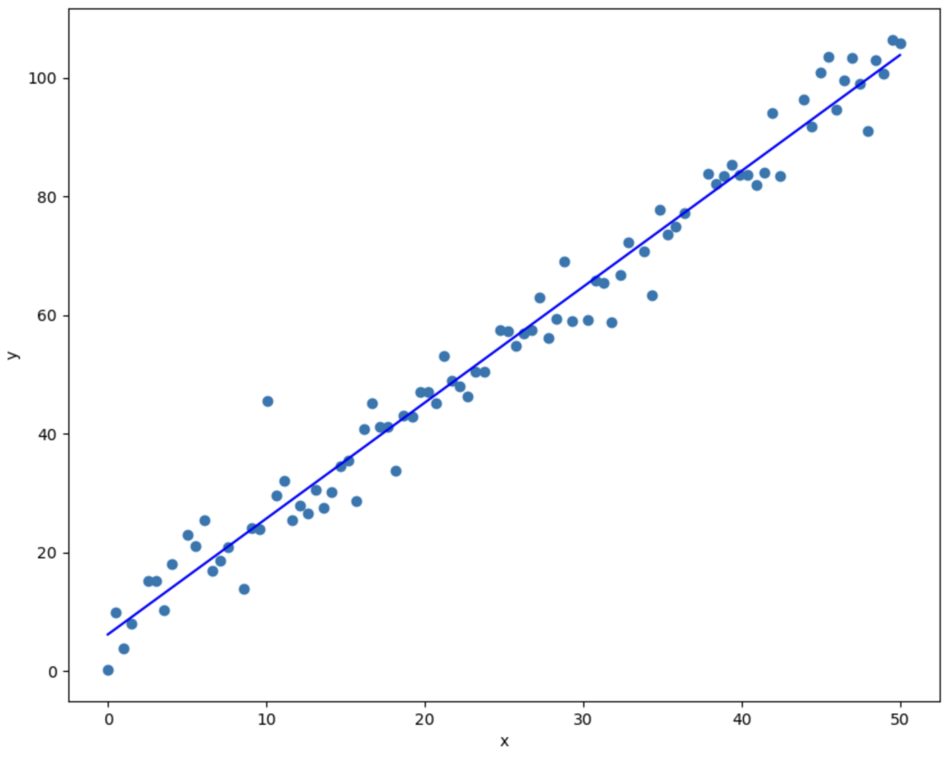

Figure 5: Cleaned data (points) and fitted least-squares line (solid).

Results:

Slope (b)	1.9537
Intercept (a)	6.1268
R^2	0.9754
p-value	1.39×10^{-72}
Standard error	0.0331

The results from SciPy matched the values calculated manually.

What these results mean:

- The R^2 value (0.975) shows that the regression line explains approximately 97.5% of the variation in y. This means that the model fits the data extremely well; almost all changes in y can be predicted from x.
- The p-value is extremely small (close to zero), which means that the relationship between x and y is statistically significant and not due to chance.
- The standard error (0.0331) was very low, indicating that the estimated slope was highly reliable.

When plotted, the regression line (in red) passes neatly through the middle of the blue data points, confirming a strong and consistent positive relationship.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{4}$$

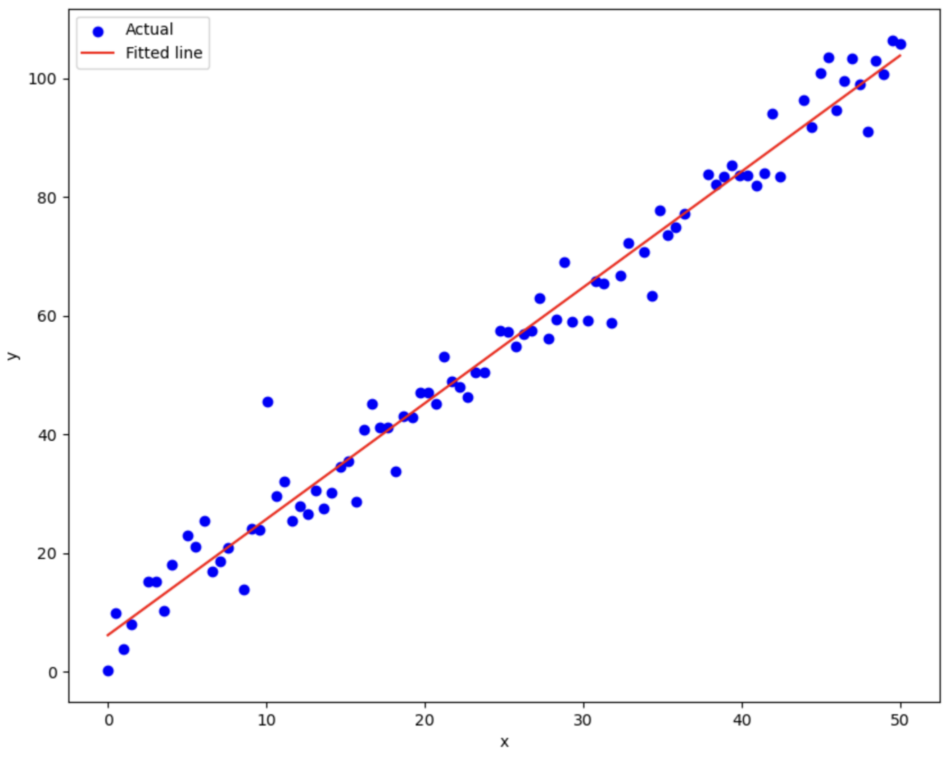

Figure 6: Regression line (in red) plotted using Scikit-learn.

Appendix

Python code to reproduce the steps of the regression analysis:

```
import pandas as pd #for tabular data
import numpy as np #for advanced mathematics
import sklearn #for machine learning and processing of the data
import seaborn as sns #for data visualisation
import matplotlib.pyplot as plt #for data visualisation
from sklearn.linear_model import LinearRegression
from scipy.stats import linregress

1.Exploratory Data Analysis
# Load the dataset
df_dirty = pd.read_csv('dirty_regression_data.csv')
#preview data
df.head()
# understanding the data
df.info()
#Summary of numeric columns
df.describe()

1.1 Checking for Outliers
#plot box plot
df.plot(kind="box", subplots=True)
plt.suptitle("Box plot to investigate outliers")
plt.show()
#plot scatter plot
plt.scatter(df['x'],df['y'])
plt.xlabel('x')
plt.ylabel('y')
plt.show()

1.2 Checking for missing and duplicate values
df.isna().sum()
df.duplicated().any()

1.3 Investigating Missingness
#checking whether y missing values depend on x
sns.boxplot(x=df['y'].isna(), y=df['x'])
plt.show()

1.4 Dealing with missing values
```

```
#cleaning and saving data
df_clean= df.dropna(subset=['y'])
df_clean.to_csv("treated_set.dat", index= False)
df_clean.info()

#Plot scatter plot
plt.scatter(df_clean['x'],df_clean['y'])
plt.xlabel('x')
plt.ylabel('y')
plt.show()

#calculate the correlation matrix and visualise it as a heatmap
cor= df.corr()
plt.rcParams['figure.figsize']= (10,8)
sns.heatmap(cor,xticklabels=cor.columns.values,yticklabels= cor.columns.values,annot= True)
plt.show()

2. Linear regression
#load and preview clean data
treated_set= pd.read_csv("treated_set.dat")
treated_set.head()

#linear regression manually using OLS
x = treated_set['x'].to_numpy(dtype=float)
y = treated_set['y'].to_numpy(dtype=float)
#calculate the mean of the two columns
x_mean = x.mean()
y_mean = y.mean()
b = np.sum((x - x_mean) * (y - y_mean)) / np.sum((x - x_mean)**2)
a = y_mean - b * x_mean
print('custom regression coefficients', a, b)

#plotting the values
x_max= np.max(treated_set['x'])
x_min= np.min(treated_set['x'])
x=np.linspace(x_min,x_max, 100)
y= a + b*x
#plotting the line
plt.plot(x,y, color= '#0000ff', label= 'Linear regression')
plt.scatter(df_clean['x'],df_clean['y'])
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.title('Linear Regression using OLS')
plt.show()
```

Manual calculation of R-squared

```
# actual y values
y_actual = treated_set['y'].to_numpy()
# predicted y values
y_pred = a + b * treated_set['x'].to_numpy()
# mean of actual y
y_mean = np.mean(y_actual)
# total sum of squares
SS_tot = np.sum((y_actual - y_mean)**2)
# residual sum of squares
SS_res = np.sum((y_actual - y_pred)**2)
# R-squared
R2_manual = 1 - (SS_res / SS_tot)
print("Manual R-squared:", R2_manual)
```

Linear regression with Sklearn

```
# separate features (x) and target (y)
X = treated_set[['x']]   # double brackets to make it a 2D array
y = treated_set['y']
# create and fit the model
model = LinearRegression()
model.fit(X, y)
# get coefficients
a = model.intercept_
b = model.coef_[0]
print('sklearn regression coefficients:')
print('Intercept (a):', a)
print('Slope (b):', b)
#make predictions and plot
import matplotlib.pyplot as plt
y_pred = model.predict(X)
plt.scatter(X, y, color='blue', label='Actual')
plt.plot(X, y_pred, color='red', label='Fitted line')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

```
Linear regression with SciPy stats.linregress
# perform linear regression using SciPy
result = stats.linregress(treated_set['x'], treated_set['y'])
# print results
print("Slope (b):", result.slope)
print("Intercept (a):", result.intercept)
print("R-squared:", result.rvalue**2)
print("p-value:", result.pvalue)
print("Standard error:", result.stderr)
```

References

[Alsufyani et al., 2024] Alsufyani, S., Forshaw, M., and Johansson Fernstad, S. (2024). Visualization of missing data: A state-of-the-art survey. *arXiv preprint arXiv:2410.03712.*

[Emmanuel et al., 2021] Emmanuel, T., Maupong, T., Mpoeleng, D., Abu-Mahfouz, A. M., and Hancke, G. P. (2021). A survey on missing data in machine learning. *Journal of Big Data*, 8(1):1–37.

[Freedman, 2009] Freedman, D. A. (2009). *Statistical Models: Theory and Practice.* Cambridge University Press, Cambridge, 2nd edition.

[Haig, 2018] Haig, B. D. (2018). Exploratory data analysis. In *The Philosophy of Quantitative Methods: Understanding Statistics.* Oxford University Press, New York. Online edition, Oxford Academic, accessed 22 Oct. 2025.

[Tukey, 1977] Tukey, J. W. (1977). *Exploratory Data Analysis.* Addison-Wesley, Reading, Massachusetts.

[Warne, 2020] Warne, R. T. (2020). Correlation. In *Statistics for the Social Sciences: A General Linear Model Approach*, pages 334–374. Cambridge University Press, Cambridge. Chapter.