420-SN1-RE Programming in Science - Lab Exercise 17

November 16, 2024

Introduction

In this lab you will use Python's `statistics` module for basic statistical calculations, use `pandas` with a simple data file, and practice plotting skills using `matplotlib.pyplot`. You will also learn how to use the `help()` function to explore new functions in Python libraries.

Exercise 1

In this section we'll do a basic plot using some measured data, and add a "line of best fit" using a technique called *linear regression*. You will see linear regression in your probability and statistics course, so we will not cover the theory of the technique here. However, the basic idea is that, given a set of x and y values measured from an experiment, linear regression will determine the slope and intercept of a line that is the "best" fit for those measurements.

1. **Setting up**

   - Create a Python file called `lab17-ex1.py`
   - Import the necessary libraries:

     ```
     import matplotlib.pyplot as plt
     import statistics as st
     ```

   - Define the lists `x` (for time) and `y` (for position):

     ```
     x = [0, 1, 2, 3, 4, 5] # time in seconds
     y = [3.1738, 4.1605, 6.0860, 7.6511, 8.9429, 11.6408] # position in metres
     ```

2. **Explore with `help()`**

   - Use the `help()` function to learn about some of the functions of the `statistics` module: `mean`, `stdev` and `linear_regression`:

     ```
     help(st.linear_regression)
     ```

3. **Calculate and print statistics**

   - Use the `mean()` and `stdev()` functions from the `statistics` module to get the average and standard deviation of the data in `y`.
   - Using f-strings, print the results to 4 decimal places; make sure it is clear what each value is.

4. **Line of best fit - Generate Data**

   (a) Recall that the equation of a line is: `y = mx + b`

   (b) Use `st.linear_regression` to find the slope (`m`) and intercept (`b`) of the best-fit line:
   `m, b = st.linear_regression(x, y)`

(c) In order to analyze the correlation between the data for `y` as a function of `x`, we will graph the line of best fit.

(d) Use the equation of the line with the values of `x` provided in step 1, the calculated slope(m) and intercept(b) from the previous step to generate the data of the line of best fit that can be graphed, create a list called `best_y`.

(e) Create an empty list `best_y=[]` and use a loop to calculate a value for `best_y` for each value in the list `x`.

5. **Line of best fit - Plot the data**

- Use `plt.scatter()` to plot the initial data (from step 1); Plot `y` as a function of `x`.
- Use the data from step 4, to plot the line of best fit with `plt.plot()`
- Add labels for the axes, including units.
- You can add text annotations to the graph so that you can show the values for the slope and intercept on the graph:

  ```
  plt.text(0, 11, f'm: {m:.4f} b: {b:.4f}')
  ```

  The first two values are the x and y coordinates for the text. These are given in the coordinates of the axes, you may adjust these values if you prefer to locate the information elsewhere on your graph.
- Add a title showing your family name and student ID number using `plt.suptitle()`
- Your final graph should resemble Figure 1.

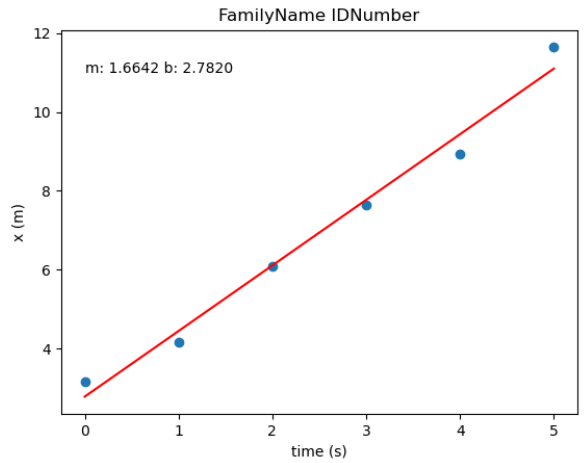

Figure 1: The completed graph for Exercise 1

Exercise 2

It is often useful to use linear regression to fit a line to a set of measurements, as you did in Exercise 1. However, it is important to remember that the quantitative results may or may not make actual sense,

depending on the nature of the underlying data. Using charts to examine your data visually can help avoid mistakes in data analysis.

Anscombe's quartet is an artificially-generated data set that illustrates this idea. We have provided this small dataset in a file named `anscombe.csv`.

1. **Setting up**

   - Create a Python file called `lab17-ex2.py`. Be sure that this file is in the same folder as your `lab17.pdf` and `anscombe.csv`.
   - Import the necessary libraries:

     ```
     import matplotlib.pyplot as plt
     import pandas as pd
     import statistics as st
     ```

   - Read the data:

     ```
     df = pd.read_csv('anscombe.csv')
     ```

     The file contains 4 small datasets. The columns are named `x1`, `y1`, `x2`, `y2`, `x3`, `y3`, `x4`, and `y4`.
   - Extract the columns data:
     Use the column names (labels) to extract each column data in a Python variable with the same name as the label.
   - **Important:** Note that steps 2 to 4 below will be completed for each of the 4 data sets.
   - **Optional:** You may want to use a big loop (over 4 data sets) to avoid repetition in your code but it is more of a challenge to do so! It is OK to repeat the code 4 times, it should not be long.

2. **Calculate and print statistics**

   - Use the `mean()` and `stdev()` functions from the `statistics` module to calculate the average and standard deviation of the y-values in each of the four datasets.
   - Use f-strings to print the results to 2 decimal places; make sure it is clear what each value is.

3. **Extract Column Data as lists**

   - When you extract a column from a pandas data frame (e.g. `x1 = df['x1']`), the returned object is a pandas series. In IDLE, print(x1) to see how it looks like.
   - To extract a column data as an actual Python list, use the following command:

     ```
     x1_list = x1.tolist()
     print(x1_list) #a normal Python list
     ```

   - The above step is required if we need to use the column data for any further calculations in Python, as in the next steps.
   - Similarly, create the lists `x2_list, x3_list and x4_list`.

4. **Line of best fit**

   - You will analyze the correlation between the data for each `y` as a function of `x`.

- Use `st.linear_regression` for each `xi` and `yi` to find the slope (`mi`) and intercept (`bi`) of the best-fit line. You will obtain the values: `m1, m2, m3, m4, b1, b2, b3, b4`
- To create and graph the line of best fit for each data set, you will need: the xi_list values and the slope(mi) and intercept(bi) for each data set to create a list called `best_yi`; where i is the id for each data set (i.e. replace i by 1,2,3,4)
- Similar to Exercise 1, you can now calculate a value for `best_yi` for each value in xi_list `xi`.

5. **Plot the data**

- Create a figure consisting of four subplots, arranged in 4 rows using `plt.subplot()`.
- Use `plt.scatter()` to plot each `yi` as a function of the corresponding `xi`.
- Plot the line of best fit `best_yi` as a function of xi_list with `plt.plot()`. Use a different color for the line.
- Add labels for the four axes, using the column names of the `DataFrame`.
- Add text annotations to the graph to show the values for the slope and intercept:

```
plt.text(min(xi), 101, f"m:{m:.2f} b:{b:.2f}")
```

- Add a title to your graph showing your family name and student ID number.
- Your final graph should resemble Figure 2. You might want to use `plt.tight_layout()` to get Python to fix the figure layout.
- Look at your final graph. What do you notice about the statistics, and the lines of best fit for the four datasets? Do you think the linear fit makes sense in all four cases?

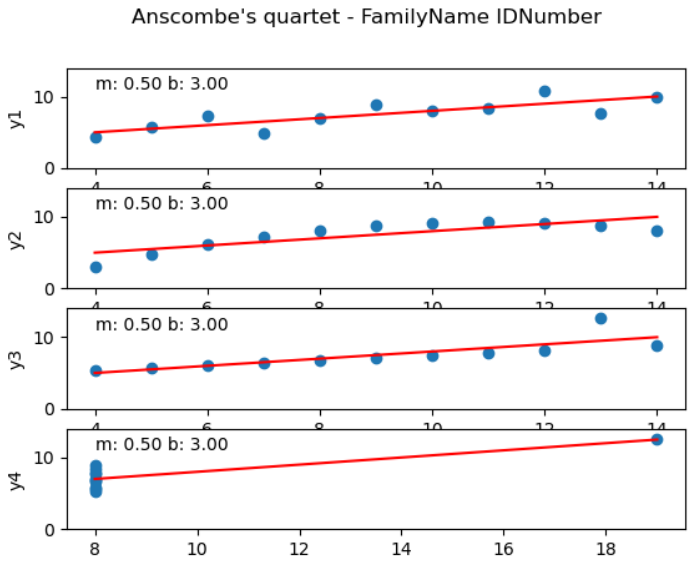

Figure 2: The completed graph for Exercise 2

Submission Guidelines

Combine all of your Python files into a single ZIP file (Lab17.zip), and upload the results to Omnivox.