Simple Linear Regression
Resolution and Python tutorial

Sophie Marchand

April 2020

1 Shortcut

Given a set of n points (x, y), the slop b and the intercept a defining the best fitting straight line $y = a + bx$ through them are defined as:

$$a = \overline{y} - b \times \overline{x}$$

$$b = \frac{\sum_{i=0}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=0}^{n}(x_i - \overline{x})^2} = corr(x, y)\frac{std(y)}{std(x)}$$

with *corr* the expression of their correlation and *std* respective standard deviation. The score of the fitting error can be computed with the Root Mean Square Error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n}(y_i^{pred} - y_i)^2}{n}}$$

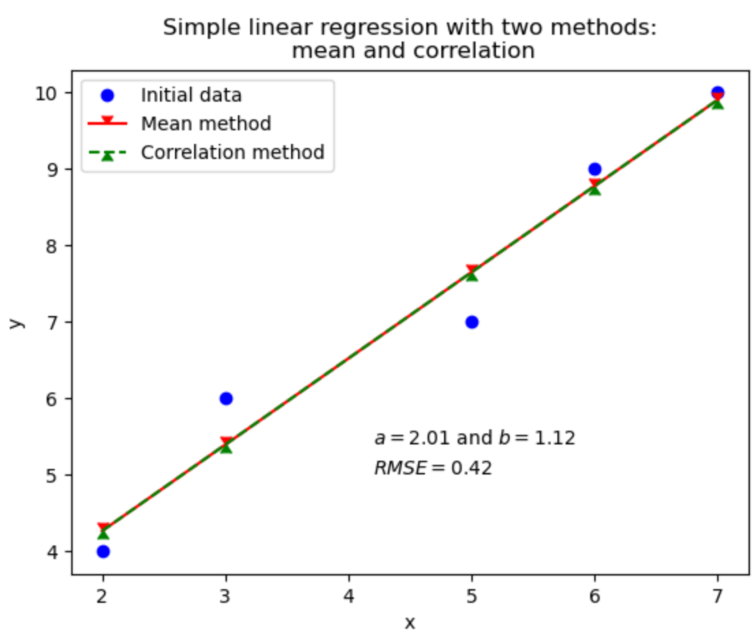

Figure 1: Output of the tutorial in section 3

2 Resolution details

Here, we provide a mathematical workflow for the solving the problem of finding the best fitting straight line f through a set of n points (x, y). To do so, we will use the linear least square fitting technique which consists in minimizing the sum of the squared residuals R^2 as defined by the equation (1).

$$R^2 = \sum_{i=0}^{n}[y_i - f(x_i)]^2 \text{ with } f(x_i) = a + bx_i \tag{1}$$

Then, minimizing R^2 requires to verifying the following expressions:

$$\begin{cases} \frac{\partial R^2}{\partial a} = -2\sum_{i=0}^{n}[y_i - (a + bx_i)] = 0 \\ \frac{\partial R^2}{\partial b} = -2\sum_{i=0}^{n}[y_i - (a + bx_i)]x_i = 0 \end{cases} \tag{2}$$

From (2), we obtain the system (3), defined as matrix at (4).

$$\begin{cases} na + b\sum_{i=0}^{n} x_i = \sum_{i=0}^{n} y_i \\ a\sum_{i=0}^{n} x_i + b\sum_{i=0}^{n} x_i^2 = \sum_{i=0}^{n} x_i y_i \end{cases} \tag{3}$$

$$\begin{bmatrix} n & \sum_{i=0}^{n} x_i \\ \sum_{i=0}^{n} x_i & \sum_{i=0}^{n} x_i^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n} y_i \\ \sum_{i=0}^{n} x_i y_i \end{bmatrix} \tag{4}$$

The expression (4) can be rearrange in:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} n & \sum_{i=0}^{n} x_i \\ \sum_{i=0}^{n} x_i & \sum_{i=0}^{n} x_i^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=0}^{n} y_i \\ \sum_{i=0}^{n} x_i y_i \end{bmatrix} \tag{5}$$

We then develop the inverse of the matrix in (5):

$$\begin{bmatrix} n & \sum_{i=0}^{n} x_i \\ \sum_{i=0}^{n} x_i & \sum_{i=0}^{n} x_i^2 \end{bmatrix}^{-1} = \frac{1}{n\sum_{i=0}^{n} x_i^2 - (\sum_{i=0}^{n} x_i)^2} \begin{bmatrix} \sum_{i=0}^{n} x_i^2 & -\sum_{i=0}^{n} x_i \\ -\sum_{i=0}^{n} x_i & n \end{bmatrix}$$

Using what precedes, we obtain the values of a and b from the equation (5):

$$\begin{cases} a = \frac{\overline{y}\sum_{i=0}^{n} x_i^2 - \overline{x}\sum_{i=0}^{n} x_i y_i}{\sum_{i=0}^{n} x_i^2 - n\overline{x}^2} \\ b = \frac{\sum_{i=0}^{n} x_i y_i - n\overline{xy}}{\sum_{i=0}^{n} x_i^2 - n\overline{x}^2} \end{cases} \tag{6}$$

Considering $n >> 2$, the expressions (6) can be simplified as follow:

$$\begin{cases} a = \overline{y} - b \times \overline{x} \\ b = \frac{\sum_{i=0}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=0}^{n}(x_i - \overline{x})^2} \end{cases} \tag{7}$$

Indeed, we can prove (7) by developing b then a:

$$b = \frac{\sum_{i=0}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=0}^{n}(x_i - \overline{x})^2} = \frac{\sum_{i=0}^{n}(x_i y_i + (n-2)x\overline{y}}{\sum_{i=0}^{n}(x_i^2 + (n-2)x^2} \simeq \frac{\sum_{i=0}^{n} x_i y_i - n\overline{xy}}{\sum_{i=0}^{n} x_i^2 - n\overline{x}^2}$$

$$a = \overline{y} - b \times \overline{x} = \frac{\overline{y}\sum_{i=0}^{n} x_i^2 - \overline{x}\sum_{i=0}^{n} x_i y_i}{\sum_{i=0}^{n}(x_i^2 + (n-2)x^2} \simeq \frac{\overline{y}\sum_{i=0}^{n} x_i^2 - \overline{x}\sum_{i=0}^{n} x_i y_i}{\sum_{i=0}^{n} x_i^2 - n\overline{x}^2}$$

An alternative expression for a then b knowing the expression of their correlation *corr* and respective standard deviation *std* is:

$$corr(x, y) = \frac{1}{n}\sum_{i=0}^{n}(x_i - \overline{x})(y_i - \overline{y}) \tag{8}$$

$$std(x) = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(x_i - \overline{x})^2} \tag{9}$$

$$b = \frac{\frac{1}{n}\sum_{i=0}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\frac{1}{n}\sum_{i=0}^{n}(x_i - \overline{x})^2}\sqrt{\frac{1}{n}\sum_{i=0}^{n}(x_i - \overline{x})^2}} \frac{\sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \overline{y})^2}}{\sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \overline{y})^2}} = corr(x, y)\frac{std(y)}{std(x)} \tag{10}$$

Finally, to measure the error score of the predicted y^{pred} made with the fitting function f compare to the y initial values, we can use the Root Mean Square Error $RMSE$ defined as follow:

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n}(y_i^{pred} - y_i)^2}{n}} \tag{11}$$

3 Python tutorial

The code source displayed below can be found on GitHub under Python_SimpleLinearRegression.py

```python
"""
Tutorial Simple Linear Regression

Author: Sophie Marchand
"""
import matplotlib as matplot
import matplotlib.pyplot as plt
import numpy as np

# Create a set of points (x,y)
x = np.array([2, 3, 5, 6, 7])
y = np.array([4, 6, 7, 9, 10])

# Compute the means, standard deviations and correlation
x_mean = np.mean(x)
y_mean = np.mean(y)
x_std = np.std(x)
y_std = np.std(y)
xy_corr = np.corrcoef(x, y)[0][1]

# Compute the slop b and the intercept a with means
b_from_mean = sum((x - x_mean) * (y - y_mean)) / sum(np.square(x - x_mean))
a_from_mean = y_mean - b_from_mean * x_mean

# Compute the slop b and the intercept a with standard deviations and correlation
b_from_corr = (xy_corr * y_std) / x_std
a_from_corr = y_mean - b_from_corr * x_mean

# Compute predicted values
y_predicted_from_mean = a_from_mean + b_from_mean * x
y_predicted_from_corr = a_from_corr + b_from_corr * x

# Compute Root Mean Square Error
error_from_mean = np.sqrt(np.sum(np.square(y_predicted_from_mean - y)) / len(y))
error_from_corr = np.sqrt(np.sum(np.square(y_predicted_from_corr - y)) / len(y))

# Print results
fig, ax = plt.subplots()
ax.plot(x, y, 'bo', label='Initial data')
ax.plot(x, y_predicted_from_mean, linestyle='-', marker=matplot.markers.CARETDOWN,
        color='r', label='Mean method')
ax.plot(x, y_predicted_from_corr, linestyle='--', marker=matplot.markers.CARETUP,
        color='g', label='Correlation method')
ax.set(xlabel='x', ylabel='y',
       title='Simple linear regression with two methods:\n mean and correlation')
ax.legend()
ax.text(4.2, 5.4, r'$a=$' + str(a_from_mean)[:4] +
        ' and $b=$' + str(b_from_mean)[:4], fontsize=10)
ax.text(4.2, 5, r'$RMSE=$' + str(error_from_mean)[:4], fontsize=10)
plt.show()
```