Linear Regression, Method 1

Chris Cornwell

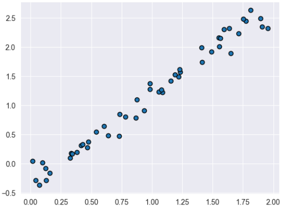
Feb 11, 2025

Outline

Outline

The goal

- Setting: have points in the plane, say n of them. Say the points are $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- **Goal:** *Model* them as being "noisy" points from a line, finding "best fit" line (the closest linear model). This line is also called the **least squares regression** (LSR) line.
- **Running example:** A data set, `'Example1.csv'`, with 50 points, is available here; these points are displayed in the plot.



Outline

Overview of linear regression task

The procedure

Implementing the procedure

Examples

Finding the LSR line

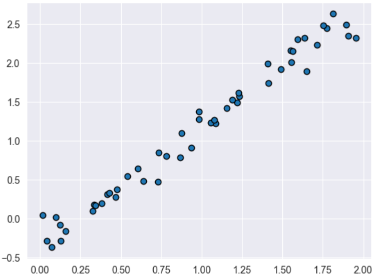


Figure: Our running example

How do we find the LSR line?

Can get the slope *m*, intercept *b* simply from using the `polyfit` function in NumPy. If `x`, `y` are the arrays with *x*- and *y*-coordinates:

```
np.polyfit(x,y,1)
```

Finding the LSR line

But, how are the slope, intercept found?

- If a slope m and intercept b existed so that $(x_1, y_1), \ldots, (x_{50}, y_{50})$ *were* points on $y = mx + b$, then

$$y_i = mx_i + b$$

would hold for all $1 \leq i \leq 50$.

1. Write those 50 equations as a matrix equation. Setting:

$$A = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_{50} & 1 \end{bmatrix}; \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{50} \end{bmatrix},$$

and writing[1] $\mathbf{p} = \begin{bmatrix} m \\ b \end{bmatrix}$, the matrix equation is $A\mathbf{p} = \mathbf{y}$.

[1] Will use $\mathbf{p}$ for this vector, for the rest of these slides.

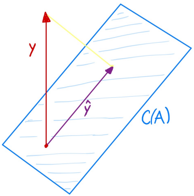
Finding the LSR line

Now the equation $A\mathbf{p} = \mathbf{y}$ does not have a solution (those points are *not* on a line).

Next idea: (*thinking of noise being in direction of* $\mathbf{y}$)

- Get vector $\hat{\mathbf{y}}$ that is *as close to* $\mathbf{y}$ *as possible*, so that $A\mathbf{p} = \hat{\mathbf{y}}$ has a solution.
- For each i, we either increase or decrease y_i by a (hopefully small) amount, $y_i \rightsquigarrow \hat{y}_i$. We make $|\mathbf{y} - \hat{\mathbf{y}}|$ as small as possible.

2. Done by solving $A^T A\mathbf{p} = A^T\mathbf{y}$ (normal equations).
   If $\mathbf{p} = \begin{bmatrix} \hat{m} \\ \hat{b} \end{bmatrix}$ is the solution,
   then $\hat{\mathbf{y}}$ is given by $A\begin{bmatrix} \hat{m} \\ \hat{b} \end{bmatrix}$.



Normal equation

Why does solving $A^T A\mathbf{p} = A^T\mathbf{y}$ give the right thing?

Related to orthogonal vectors in $\mathbb{R}^n$ (in the example, $\mathbb{R}^{50}$).

- If $A\mathbf{p} = \hat{\mathbf{y}}$ has a solution, then $\hat{\mathbf{y}} \in \mathbb{R}^{50}$ is in column space of A.
- $\mathbf{y} = \mathbf{z}_1 + \mathbf{z}_2$, where $\mathbf{z}_1$ in null space of A^T and $\mathbf{z}_2$ in column space of A.
  (*Note: Null space of A^T orthogonal to column space of A.*)
- As $\mathbf{z}_2$ in column space, $\exists\, \hat{\mathbf{p}}$ so that $A\hat{\mathbf{p}} = \mathbf{z}_2$.But then,

$$A^T(A\hat{\mathbf{p}}) = A^T\mathbf{z}_2 = A^T(\mathbf{y} - \mathbf{z}_1) = A^T\mathbf{y}.$$

And $\mathbf{z}_2$ is closest, since subtracted $\mathbf{z}_1$ from $\mathbf{y}$, orthogonal to column space:

$$\mathbf{z}_2 = \hat{\mathbf{y}}.$$

Finding the LSR line

Normal equations:

$$(A^T A)\mathbf{p} = A^T \mathbf{y}.$$

Note:

- $A^T A$ is 2×2 matrix, $A^T \mathbf{y} \in \mathbb{R}^2$, and $A^T A$ is invertible as long as there exists $x_i \neq x_j$.

3. Pretty straightforward to find solution to $(A^T A)\mathbf{p} = A^T \mathbf{y}$.

So, three steps:

1. Write the n equations in matrix form. (get matrix A, vector $\mathbf{y}$)
2. Get matrix $A^T A$ and vector $A^T \mathbf{y}$ for normal equations: $A^T A\mathbf{p} = A^T \mathbf{y}$.
3. Use a method to solve normal equations for $\mathbf{p}$.

Outline

Solving normal equation, in pseudocode

Procedure to carry out the steps:

1. Write the n equations in matrix form. (get matrix A, vector $\mathbf{y}$)
2. Get matrix $A^T A$ and vector $A^T \mathbf{y}$ for normal equations: $A^T A \mathbf{p} = A^T \mathbf{y}$.
3. Use a method to solve normal equations for $\mathbf{p}$.

Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, as a NumPy array (call it `D`, with shape `(n,2)`):

```
A ← [D[:,0], all 1s] # 2-column matrix
y ← D[:,1]
# next, get 2x2 matrix and 2-vector
Compute A.T times A; compute A.T times y
Solve normal eq'ns (numpy solve, or use inverse)
return solution
```

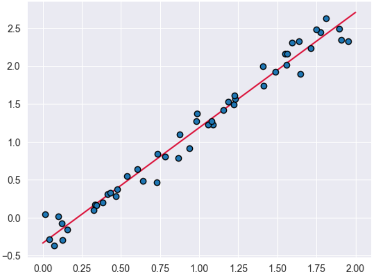
Outline

Result on running example

For the data (linked to above) with 50 points, the LSR line comes out close to

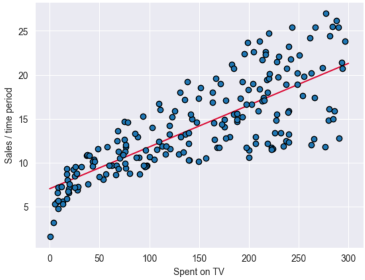
$$y = 1.520275x - 0.33458.$$

A plot of the line (in red), alongside the points, looks as follows.



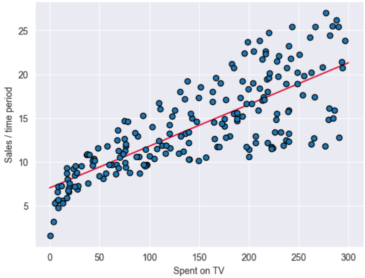
Another example, Advertising data

In the DataSets folder, the `'Advertising.csv'` file contains data on amounts spent on TV, Radio, and Newspaper advertising for different brands, as well as amounts in Sales (per day??).
We will look more at this data later. For now, plotted here are the columns (`'TV'`, `'Sales'`).



Another example, Advertising data

If you switch the role of x- and y-coordinates, you can still do linear regression; i.e., for purpose of a thought experiment, predict the TV data as the *response*, instead of the Sales.
The LSR line for the data is then **not** the same line, if you switch the roles of TV and Sales in the algorithm to get $\hat{\mathbf{p}}$.



Another example, Advertising data

If you switch the role of x- and y-coordinates, you can still do linear regression; i.e., for purpose of a thought experiment, predict the TV data as the *response*, instead of the Sales.

The LSR line for the data is then **not** the same line, if you switch the roles of TV and Sales in the algorithm to get $\hat{\mathbf{p}}$.