Lecture 1: Linear Regression

with least-squares fitting

Zejian Li
(li.zejian@ictp.it)

16 Oct. 2024

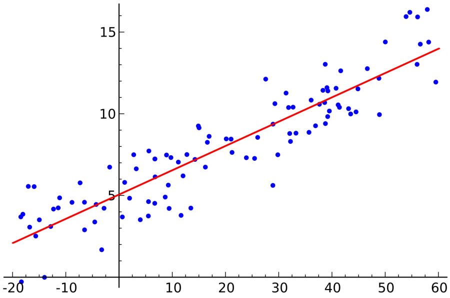
Curve fitting - Introduction

The process of constructing a parametrized function $f(\boldsymbol{x}; \beta)$ that has the best fit to a series of data points $\{(\boldsymbol{x}_i, y_i)\}_i$.

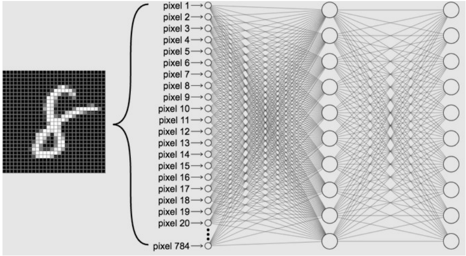
For example...

- Linear regression:

$$f(x; \beta_1, \beta_2) = \beta_1 + \beta_2 x$$

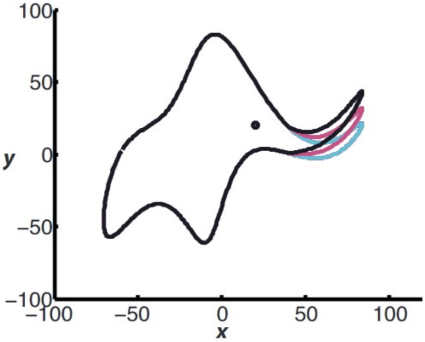


- Training an AI (supervised learning): $f(\boldsymbol{x}; \beta)$ is an complicated nonlinear function (often represented as a "neural network") that can be trained (optimizing the parameters β) to learn patterns in the input $\boldsymbol{x}$.



Curve (over-)fitting

"With four parameters I can fit an elephant, and with five I can make him wiggle his trunk."
— John von Neumann



The Fermi-Neumann elephant.
(See Am. J. Phys. 1 June 2010; 78 (6): 648–649)

Linear least-squares fitting

The procedure for fitting a linear function by minimizing the sum of the squares of the residuals of the points from the curve.

- Input: dataset with N points $\{(x_1, y_1), \dots, (x_N, y_N)\}$.
- Assumption: errors ε_i are only in y_i,

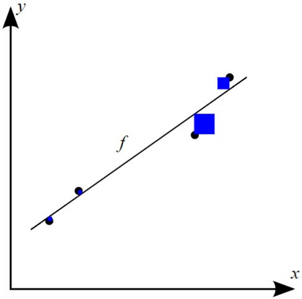
$$y_i = f(x_i) + \varepsilon_i \,,$$

  and are normally distributed $\varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$.
- Linear model (the function we want to fit):

$$f(x; \beta_1, \beta_2) = \beta_1 + \beta_2 x \,.$$

- Sum of squared residuals (the quantity to be minimized):

$$S_{\mathrm{res}} \equiv \sum_i [y_i - f(x_i)]^2 = \sum_i [y_i - (\beta_1 + \beta_2 x_i)]^2 \,.$$



Squares of residuals

Linear least-squares fitting

We now minimize the sum of squared residuals:

$$S_{\text{res}}(\beta_1, \beta_2) \equiv \sum_i [y_i - f(x_i)]^2 = \sum_i [y_i - (\beta_1 + \beta_2 x_i)]^2 \,.$$

- We require the partial derivatives $\partial_{\boldsymbol{\beta}} S_{\text{res}}$ to be zero at the minimum:

$$\frac{\partial S_{\text{res}}}{\partial \beta_1} = -2 \sum_{i=1}^{N} [y_i - \beta_1 - \beta_2 x_i] = 0 \,,$$

$$\frac{\partial S_{\text{res}}}{\partial \beta_2} = -2 \sum_{i=1}^{N} [y_i - \beta_1 - \beta_2 x_i] x_i = 0 \,.$$

- This is a linear system for (β_1, β_2) that you know how to solve with Cramer's rule (see last lecture):

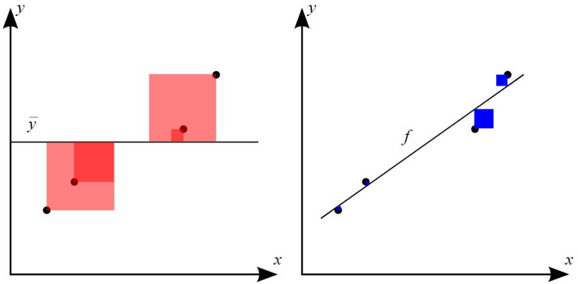
$$\begin{bmatrix} N & \Sigma_i x_i \\ \Sigma_i x_i & \Sigma_i x_i^2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \Sigma_i y_i \\ \Sigma_i x_i y_i \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} = \begin{bmatrix} N & \Sigma_i x_i \\ \Sigma_i x_i & \Sigma_i x_i^2 \end{bmatrix}^{-1} \begin{bmatrix} \Sigma_i y_i \\ \Sigma_i x_i y_i \end{bmatrix} ,$$

- Estimator for the fit function: $\hat{f}(x) \equiv f(x; \hat{\beta}_1, \hat{\beta}_2)$.

Quality of the fit and error estimation

- Overall quality of the fit: coefficient of determination R^2:

$$R^2 \equiv 1 - \frac{S_{\mathrm{res}}}{S_{\mathrm{tot}}}\,, \quad S_{\mathrm{tot}} \equiv \sum_i (y_i - \overline{y})^2\,, \quad S_{\mathrm{res}} \equiv \sum_i [y_i - \hat{f}(x_i)]^2\,, \quad \overline{y} \equiv \frac{1}{N}\sum_{i=1}^{N} y_i\,.$$



Quality of the fit and error estimation

- Overall quality of the fit: coefficient of determination R^2:

$$R^2 \equiv 1 - \frac{S_{\mathrm{res}}}{S_{\mathrm{tot}}}\,, \quad S_{\mathrm{tot}} \equiv \sum_i (y_i - \overline{y})^2\,, \quad S_{\mathrm{res}} = \sum_i [y_i - \hat{f}(x_i)]^2\,, \quad \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i\,.$$

- Estimator for the variance σ^2 of the error $\varepsilon_i = y_i - f(x_i)$:

$$\hat{\sigma}^2 = \sum_{i=1}^{N} \frac{\varepsilon_i^2}{N-2}\,.$$

- Standard errors (SE) for the fit parameters:

$$\widehat{\mathrm{SE}}(\hat{\beta}_1) = \hat{\sigma}\sqrt{\frac{1}{N} + \frac{\overline{x}^2}{\sum_i (x_i - \overline{x})^2}}\,, \quad \widehat{\mathrm{SE}}(\hat{\beta}_2) = \hat{\sigma}\frac{1}{\sqrt{\sum_i (x_i - \overline{x})^2}}\,.$$

General case of multiple variables

- Dataset: $\{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,N}$ with $\boldsymbol{x}_i \in \mathbb{R}^{1\times D}$ being $D-$dimensional row vectors.
- Linear model: $f(\boldsymbol{x};\beta) = \boldsymbol{x}\beta$ with $\beta \in \mathbb{R}^{D\times 1}$ being the column vector of linear weights (fit parameters). The intercept can be absorbed into β by adding an entry of constant 1 into $\boldsymbol{x}$.
- Error assumption:

$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i\,, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbb{I})\,.$$

- Notation:

$$\mathbf{X} \equiv \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix}, \quad \boldsymbol{y} \equiv \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}.$$

- Estimators for the fit parameters and their covariances:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}\,, \quad \widehat{\mathrm{Var}}(\hat{\beta}) = \hat{\sigma}^2(\mathbf{X}^T\mathbf{X})^{-1}\,, \quad \hat{\sigma}^2 = \sum_{i=1}^{N} \frac{\varepsilon_i^2}{N-D}\,.$$

- $N-D$ is the degree of freedom of the estimate in order to provide an unbiased estimation. Read more on Wikipedia: Unbiased estimation of standard deviation .

Assignment: estimate Hubble's constant

In 1929, Edwin Hubble noted a remarkable linear relationship in our universe: the greater the distance d to a galaxy – the larger its velocity of recession v_r, which shows that the universe is expanding. This phenomena is expressed as: $v_r = H_0 d$ known as Hubble's Law where the slope of the best fit line through the observation data is known as the Hubble Constant (read his original paper here!). Today, astronomers use exploding stars called Type 1A supernova to more accurately determine speeds and distances across the universe. In the text file `hubble_data.txt` we can find a list of speeds (in km/s) and distances (in megaparsec, $1 \text{ parsec} \simeq 3.26 \text{ ly}$) for 15 Type 1A supernovae. **Write a Fortran program to perform a linear fit of the data and estimate the Hubble constant.**

- Read the speeds and distances into two separate arrays.
- Perform the fit with the linear model $v_r(d) = \beta_1 + \beta_2 d$ and print the estimates for β_1, β_2, their standard errors and the coefficient of determination R^2 in a text file `fit.txt`.
- You can use the built-in `sum` function for the summation of arrays.
- **Bonus question**: Perform the fit without the intercept, i.e. with the linear model $v_r(d) = \beta d$ (which is actually easier).

Submit your code as `Ass09.YourLastName.f90` to `li.zejian@ictp.it` before the next lesson.