Linear Regression

Sangheum Hwang

AI, machine learning, and deep learning

- Example: Linear regression
  - goal = build a system that can a vector $\boldsymbol{x} \in \mathbb{R}^p$ as input and predict the value of a scalar $y \in \mathbb{R}$ as its output
  - We define the output to be

$$\hat{y} = \boldsymbol{\beta}^T \boldsymbol{x} = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$$

  - Task T: to predict y from $\boldsymbol{x}$ by outputting $\hat{y} = \boldsymbol{\beta}^T \boldsymbol{x}$

AI, machine learning, and deep learning

- Example: Linear regression
  - Performance measure *P*: to compute the MSE on the test set

$$\mathrm{MSE}_{\mathrm{test}} = \frac{1}{n}\sum_{i}\left(\hat{y}_i^{(\mathrm{test})} - y_i^{(\mathrm{test})}\right)^2$$

  - How to gain experience by observing $\left(\boldsymbol{X}^{(\mathrm{train})}, \boldsymbol{y}^{(\mathrm{train})}\right)$? → minimize the MSE on the training set, $\mathrm{MSE}_{\mathrm{train}}$

AI, machine learning, and deep learning

- Example: Linear regression
  - Suppose that the training set $\{x_{i1}, x_{i2}, \ldots, x_{ip}, y_i\}$, $i = 1, \ldots, n$ is given.

$$\min \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$
$$= \frac{1}{2} \sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_p x_{ip})\right)^2$$

AI, machine learning, and deep learning

- Example: Linear regression
  - In vector/matrix notation,

$\hat{y}_1$ residual$_1$

$$y_1 = \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \cdots \beta_p x_{1p} + \epsilon_1$$

$$y_2 = \beta_0 + \beta_1 x_{21} + \beta_2 x_{22} + \cdots \beta_p x_{2p} + \epsilon_2$$

$$\vdots$$

$$y_n = \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \cdots \beta_p x_{np} + \epsilon_n$$

$$\Downarrow$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} = \boldsymbol{X\beta} + \boldsymbol{\epsilon}$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$$

$$\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

AI, machine learning, and deep learning

- Example: Linear regression

$$\min \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$
$$= \frac{1}{2}\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_p x_{ip})\right)^2$$

제곱

$$\min E = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) = \frac{1}{2}\varepsilon^T\varepsilon$$

$$\Rightarrow \frac{\partial E}{\partial \boldsymbol{\beta}} = -\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X\beta}) = 0$$

$$\Rightarrow -\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{X}^T\boldsymbol{X\beta} = 0$$

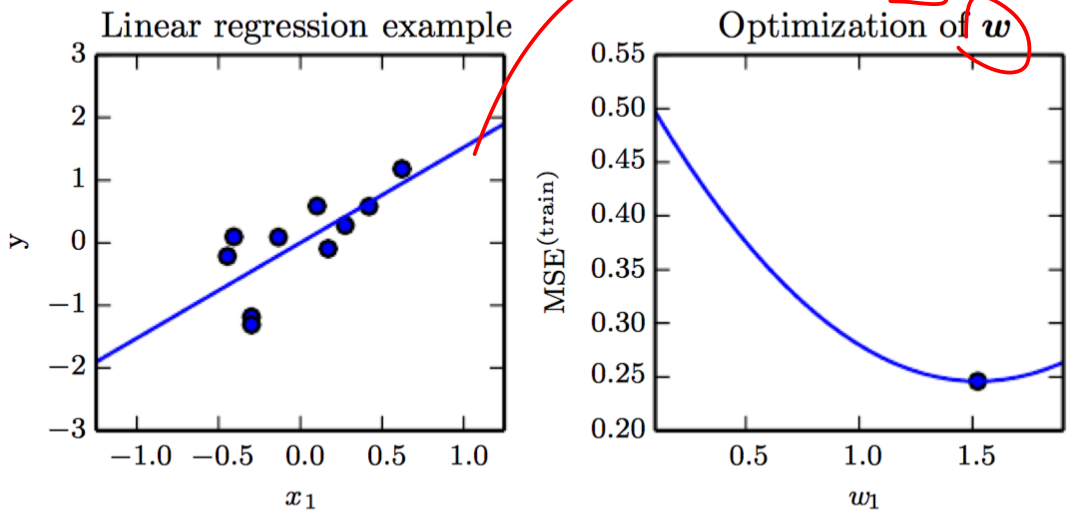
$$\boxed{\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}}$$

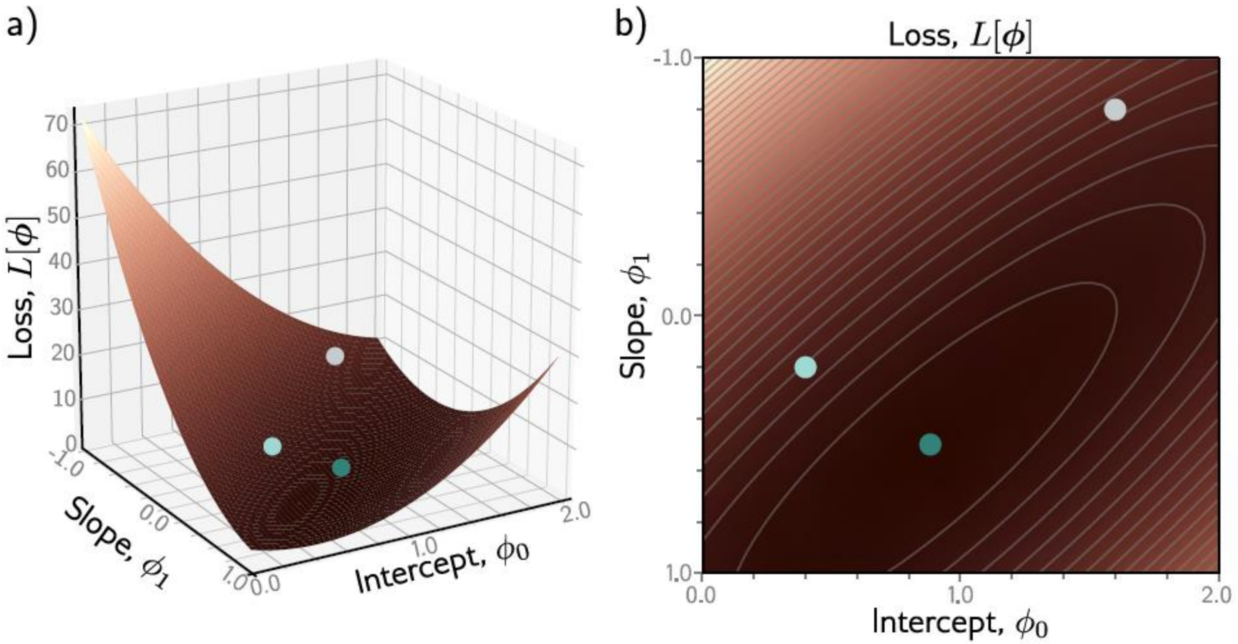
AI, machine learning, and deep learning

- Example: Linear regression
  - To minimize $\mathrm{MSE}_{\mathrm{train}}$, we can simply solve for where its gradient is **0**.

$$\nabla_{\boldsymbol{\beta}} \mathrm{MSE}_{\mathrm{train}} = 0 \rightarrow \boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$ closed-form solution

slope





Reading assignment

- “Understanding DL” book
  - Chapter 2. Supervised Learning