Linear Regression

Thursday, January 16, 2020 13:34

Linear Regression

Real-value target function

(instead of yes/no, we estimate a real value)

Approximate some function $y_n = f(\underline{x}_n)$ by a linear function

MODEL SETUP

$\mathcal{D} = \{(\underline{x}_1, y_1) \dots\}$ same as before

$\underline{x}_i \in \mathbb{R}^{d+1}$ $y_i \in \mathbb{R}$ $\underline{w} \in \mathbb{R}^{d+1}$

Regression: $\hat{y}_n = \underline{w}^T \underline{x}_n$

(predict by $\underline{w}^T \cdot \underline{x}_n$)

COMPACT REPRESENTATION

Data matrix

$$X = \begin{bmatrix} x_{10} = 1, & x_{11}, & \dots x_{1d} \\ x_{20} = 1, & x_{21} \dots\dots & x_{2d} \\ x_{30} = 1, & x_{31} \dots\dots & x_{3d} \\ x_{N0} = 1, & x_{N1} \dots\dots & x_{Nd} \end{bmatrix} \in \mathbb{R}^{N \times (d+1)}$$

$$X = \begin{bmatrix} \underline{x}_1^T \\ \underline{x}_2^T \\ \vdots \\ \underline{x}_N^T \end{bmatrix}$$

we represent input data as a matrix for compact computation

Observation Vector

$$\underline{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N$$

$N \times (d+1)$ size

$[\, y_N \,]$

$$\Rightarrow \underset{N \text{ size}}{\hat{\underline{y}}} = \begin{bmatrix} \underline{x}_1^T \underline{w} \\ \underline{x}_2^T \underline{w} \\ \vdots \\ \underline{x}_N^T \cdot \underline{w} \end{bmatrix} = \underset{N \times (d+1) \text{ size}}{X} \underset{(d+1) \text{ size}}{\underline{w}}$$

ideally $\hat{\underline{y}} = \underline{y}$ after training.

Case ① $N < d+1$:

Many Solutions (# of unknowns > # of equations)

case ② $N = d+1$: (overfitting)

Single/Unique solution: $\hat{y} = y$

Case ③ $N > d+1$:

No solution for $\hat{y} = y$

$\Rightarrow \hat{y} \neq y$ (but approximation is good enough)

$\Rightarrow$ Typically we want $N \gg d+1$ so we do not suffer from overfitting

Loss Function

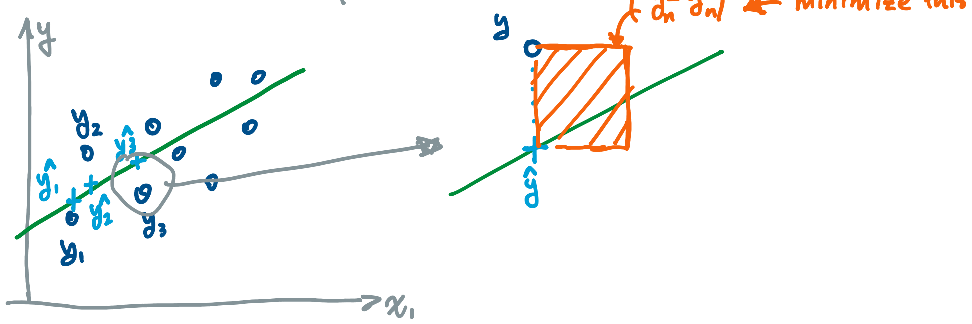
intuition: we can measu distance:

$$E_{in}(\underline{w}) = \frac{1}{N} \|\underline{y} - \hat{\underline{y}}\|_2^2 = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2$$

$$= \frac{1}{N} \|\underline{y} - X\underline{w}\|_2^2 = \frac{1}{N} \sum_{n=1}^{N} \underbrace{(y_n - \underline{x}_n^T \underline{w})^2}_{\text{error of each data : } e_n}$$

VISUAL EXAMPLE:

Conside 1-dimensional input



we want optimised weights

$$\underline{w}^* = \text{argmin}(E_{in}(\underline{w})) \qquad \underline{w} \in \mathbb{R}^{d+1}$$

$$= \text{argmin}\left(\frac{1}{N}\|\underline{y} - X\underline{w}\|^2\right)$$

How do we minimize this?

Recall $\|\underline{a}\|^2 = \underline{a}^T\underline{a}$

$$E_{in}(\underline{w}) = \frac{1}{N}(\underline{y} - X\underline{w})^T(\underline{y} - X\underline{w})$$

$$= \frac{1}{N}\left(\underline{y}^T\underline{y} - \underbrace{\underline{w}^T X\underline{y} - \underline{y}^T X\underline{w}}_{-2\underline{y}^T X\underline{w}} + \underline{w}^T X^T X\underline{w}\right)$$

$$= \frac{1}{N}\left(\underline{y}^T\underline{y} - 2\underline{y}^T X\underline{w} + \underline{w}^T X^T X\underline{w}\right)$$

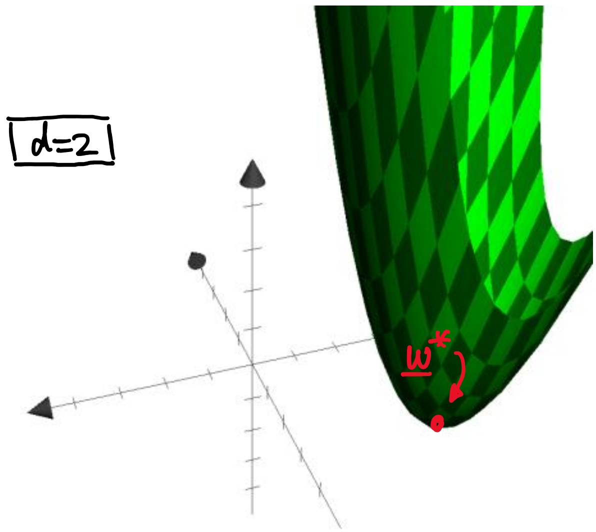
concavity determined by X^TX

↳ if all eigenvalues are ⊕/⊖/mix of ⊕ and ⊖

this looks like quadratic.

Quadratic form in $\underline{w}$ is convex

⇒ there exists a local minimum.



To get to w^*, find global minimum.

$$\nabla(E_{in}(\underline{w})) = 0$$

LINEAR ALG RULES

- $f(\underline{w}) = \underline{w}^T\underline{v} = \underline{v}^T\underline{w}$

$$f(\underline{w}) = \underline{w}^T\underline{v} = \underline{v}^T\underline{w}$$

$$\nabla f(\underline{w}) = \underline{v}$$

$$\nabla(\underline{w}^TA\underline{w}) = (A + A^T)\underline{w} = 2A\underline{w}$$

$$\nabla(E_{in}(\underline{w})) = \frac{1}{N}\left(0 - 2\underline{y}^T\underline{x} + 2X^TX\underline{w}^*\right) = 0$$

$$\Rightarrow \quad \cancel{2}\underline{y}^T\underline{x} = \cancel{2}X^TX\underline{w}^*$$

$$\Rightarrow \quad \boxed{\underline{w}^* = (X^TX)^{-1}X\underline{y}}$$

Least square solution.

Finally, we can use w^* to predict:

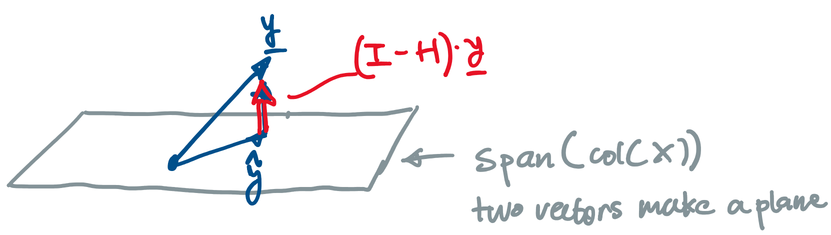
$$\hat{y} = X\underline{w}^* = \underbrace{X(X^TX)^{-1}X^T}_{H \text{ matrix}} \cdot \underline{y}$$

Special: This is a closed form solution.

Geometric Interpetation

$$X\underline{w} = \sum_{i=0}^{d} w_i \cdot \underline{q}_i$$

$\underline{q}$ are columns of X



Regularization

Helps with overfitting

Regularized Least Square (Regression)

$$\operatorname{argmin}\left(\frac{1}{N}\|\underline{y}-X\underline{w}\|^2 + \underbrace{\lambda}_{\lambda \geq 0}\|\underline{w}\|^2\right)$$

We add a term to penalise large weights.

$$f(\underline{w}) = \overbrace{\frac{1}{N}\|\underline{y}-X\underline{w}\|^2}^{E_{in}(\underline{w})} + \lambda\underbrace{\|\underline{w}\|^2}_{\underline{w}^T\underline{w}}$$

$$\nabla f(\underline{w}) = \frac{1}{N}\left(2X^TX\underline{w} + 2X^T\underline{y}\right) + 2\underline{w}\lambda = 0$$

$$\Rightarrow \underline{w}^* = (X^TX + \lambda I)^{-1}X^T\underline{y}$$

How to choose "λ" ?

$\Rightarrow$ selected using validation

collected data broken into three types:

$\rightarrow$ training data : $\lambda_1, \lambda_2 \ldots \lambda_M \xRightarrow{\text{gives}} \underline{w}^*_{\lambda_1}, w^*_{\lambda_2}, \ldots w^*_{\lambda_M}$

try different λ

also gives

$\rightarrow$ validation data : $\underbrace{E_{valid_1}, E_{valid_2} \ldots E_{valid_M}}_{\hookrightarrow \text{ pick } \underline{w}^*_{\lambda_m}}$

$\rightarrow$ test data : $E_{test}(\underline{w}^*_{\lambda_m})$
(not involved in training)