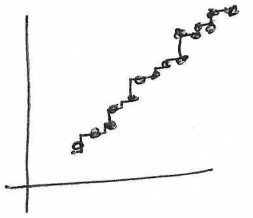
9

# Linear Regression

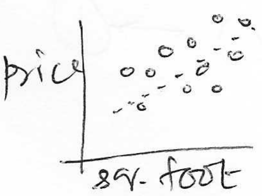
Regression → Predicting a continuous variable given some other variables.



← basically an interpolation. (doesn't work well in most of the cases).

## Linear fitting to data.

We want to fit a linear function to an observed set of points $X = [x_1 \cdots x_N]$ with associated labels $Y = [y_1, \cdots, y_N]$. Once we fit the function, we want to use it to predict the $y$ for new $x$.



search for line that best fits these data points.

I/P: $\vec{x} \in \mathbb{R}^d$

O/P: $y \in \mathbb{R}$

The model we'll use:

$$\hat{y} = w_0 + w_1 x_1 + \cdots + w_d x_d$$

$$= [1\ x_1 \cdots x_d] \begin{bmatrix} w_0 \\ \vdots \\ w_d \end{bmatrix}$$

$$= x^T w \quad (\text{or } w^T x \rightarrow \text{why?})$$

bias

why?

However, we may not find the true mapping function $f: X \rightarrow Y$, but we'll try to approximate it as much as possible.

Residuals/errors: $e_i = y_i - \hat{y}_i$

A natural loss function is squarred loss:

$$L(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$$

& we want this to be as small as possible

Least sequares estimation: Minimize w.r.t. $w$

$$L(\vec{w}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - w^T x_i)^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T w)^2$$

Many ways to solve this (e.g. taking partial derivatives $\frac{\partial L}{\partial w_0} = 0$, $\frac{\partial L}{\partial w_1} = 0, \cdots$ & solve a system of eqn's). We'll use matrix notation & avoid solving system of eqn's.

Let,

$$X = \begin{bmatrix} — \vec{x}_1 — \\ — \vec{x}_2 — \\ \vdots \\ — \vec{x}_n — \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ \vdots & & & \\ 1 & x_{N1} & \cdots & x_{Nd} \end{bmatrix}$$

$X \Rightarrow N \times (d+1)$
$Y \Rightarrow N \times 1$
$W \Rightarrow (d+1) \times 1.$

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}, \quad W = \begin{bmatrix} w_0 \\ \vdots \\ w_d \end{bmatrix}. \quad \hat{Y} = XW$$

Hence, $L(W) = \frac{1}{N} \|Y - \hat{Y}\|_2^2$ ($L_2$ norm squarred)

$$= \frac{1}{N} \|Y - XW\|_2^2$$

$$= \frac{1}{N} (Y - XW)^T (Y - XW)$$

$$L(W) = \frac{1}{N} [Y^TY + W^TX^TXW - 2Y^TXW] \quad \text{(how?!)} \rightarrow ①$$

Now we need to take $\frac{\partial L(W)}{\partial W} = 0$

But $\vec{w}$ is a vector, so we need to take care of the derivatives. We'll get back to eqn ①. Let's see some derivatives first.

$$\frac{\partial(\vec{w}^T\vec{x})}{\partial \vec{w}} = \Bigg[ \underbrace{\frac{\partial(\vec{w}^T\vec{x})}{\partial w_0}} \cdots \frac{\partial(\vec{w}^T\vec{x})}{\partial w_d} \Bigg]$$

$$\downarrow$$

$$\frac{\partial[\sum_{j=0}^{d} w_j x_j]}{\partial w_0} = x_0 \quad (=1 \text{ in this case})$$

So, $\frac{\partial(\vec{w}^T\vec{x})}{\partial \vec{w}} = [x_0 \; x_1 \cdots x_d] \Rightarrow \boxed{\frac{\partial(\vec{w}^T\vec{x})}{\partial \vec{w}} = \vec{x}^T}$

or, $\boxed{\frac{\partial(X^TW)}{\partial W} = X^T} \rightarrow ②$

Now let's compute $\frac{\partial}{\partial \vec{w}}(\vec{w}^T A \vec{w})$, where A is a symmetric matrix

$$\frac{\partial(\vec{w}^T A \vec{w})}{\partial \vec{w}} = \Bigg[ \underbrace{\frac{\partial(W^TAW)}{\partial w_0}} \cdots \frac{\partial(W^TAW)}{\partial w_d} \Bigg]$$

$$\downarrow$$

$$\frac{\partial}{\partial w_0}(W^TAW) = \frac{\partial}{\partial w_0} \Big[ \sum_{i=0}^{d} \sum_{j=0}^{d} w_i a_{ij} w_j \Big]$$

$$\Rightarrow \frac{\partial}{\partial w_0}(w^TAw) = \frac{\partial}{\partial w_0}\Big[a_{00}w_0^2 + \sum_{i\neq 0} a_{ij} w_i w_0 + \sum_{j\neq 0} a_{ij} w_0 w_j + \sum_{i\neq 0}\sum_{j\neq 0} w_i a_{ij} w_j\Big]$$

$$= 2a_{00} w_0 + \sum_{i\neq 0} a_{ij} w_i + \sum_{j\neq 0} a_{ij} w_j$$

$$= 2a_{00} w_0 + 2\sum_{i\neq 0} a_{ij} w_i \quad (\text{how?!})$$

$$= 2\sum_{i=0}^{d} a_{i0} w_i$$

$$= 2w^T A_0 \quad (A_0 \text{ is the first column of } A \rightarrow A_0 = \begin{pmatrix} a_{00} \\ a_{10} \\ \vdots \\ a_{d0} \end{pmatrix})$$

$$\Rightarrow \frac{\partial(w^TAw)}{\partial w} = [2w^TA_0 \;\; 2w^TA_1 \;\; \dots \;\; 2w^TA_d] = \boxed{2w^TA} \longrightarrow ③$$

Now let's get back to eqn ①:

$$\frac{\partial L(w)}{\partial w} = \frac{\partial}{\partial w}\Big[\frac{1}{n}(y^Ty - 2y^TXw + w^TX^TXw)\Big]$$

$$= \frac{1}{n}(-2y^TX + 2w^TX^TX) \quad (\text{using } 2, 3)$$

Now set $\frac{\partial L}{\partial w} = 0 \Rightarrow \frac{1}{n}(-2y^TX + 2w^TX^TX) = 0$

$$\Rightarrow w^TX^TX = y^TX$$

$$\Rightarrow (X^TX)w = X^Ty \quad (\text{taking transpose})$$

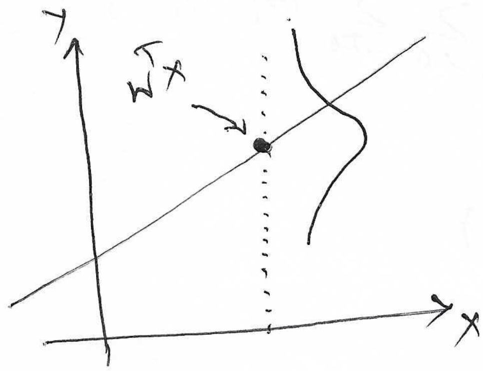
$$\Rightarrow \boxed{\hat{w}_{OLS} = (X^TX)^{-1}X^Ty}$$

$(X^TX)^{-1}X^T$ is called the pseudo-inverse of $X$.
↓

def $X^+ = (X^TX)^{-1}X^T$

Then $X^+X = (X^TX)^{-1}X^TX = I$

# Probabilistic view of linear regression



Let $X \sim p(X)$ (some unknown distribution)

& let, $Y|X=x \sim \mathcal{N}(w^T x, \sigma^2)$

The error is,

$$e = y - w^T x$$
$$\sim \mathcal{N}(0, \sigma^2)$$ (zero-mean Gaussian)

Let's see the MLE estimator of $w$ (assume $\sigma^2$ is constant)

$$\hat{w}_{MLE} = \underset{\vec{w}}{\operatorname{argmax}} \log p(D/\vec{w})$$

$$= \underset{\vec{w}}{\operatorname{argmax}} \sum_{i=1}^{N} \log p(e_i/\vec{w})$$

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{N} \left[ - \frac{(y_i - w^T x_i)^2}{\underset{\substack{\uparrow \\ \text{const.}}}{2\sigma^2}} - \frac{1}{2} \log(\cancel{2\pi\sigma^2}) \right]$$

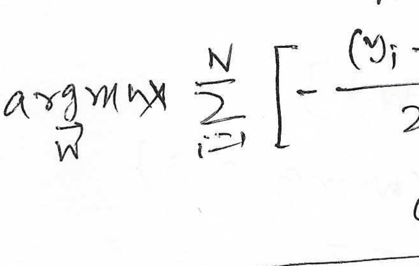
doesn't depend on $w$

Dataset:
$\{(x_1, y_1) \cdots\}$
↓ convert into errors
$e_1 \; \cdots \; e_n$
Since errors are Gaussians,

$$p(e) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[ -\frac{(y - w^T x)^2}{2\sigma^2} \right]$$

$$\Rightarrow \boxed{\begin{aligned} \hat{w}_{MLE} &= \underset{\vec{w}}{\operatorname{argmin}} \sum_{i=1}^{N} (y_i - w^T x_i)^2 \\ \hat{w}_{MLE} &= \hat{w}_{OLS} \end{aligned}}$$

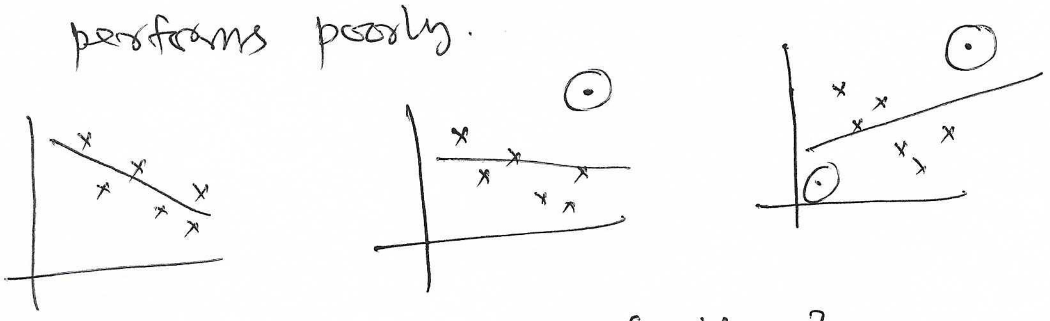
So the assumption we're making is that errors are coming from a standard Gaussian.



probability of points to be here is very small. If there are points like this, linear regression fails.

OLS is not robust to these points, aka outliers.

Our model doesn't consider "outliers", so it performs poorly.



↓ how to fix these?

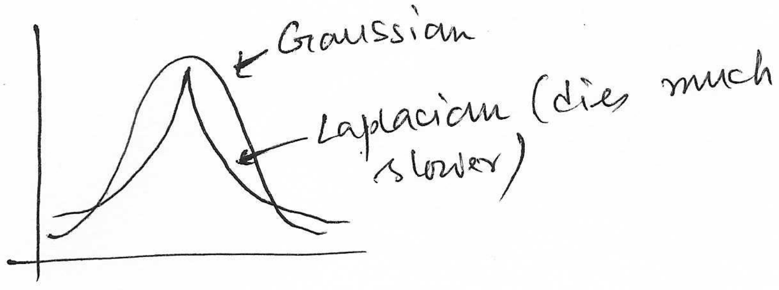
Instead of Gaussian,

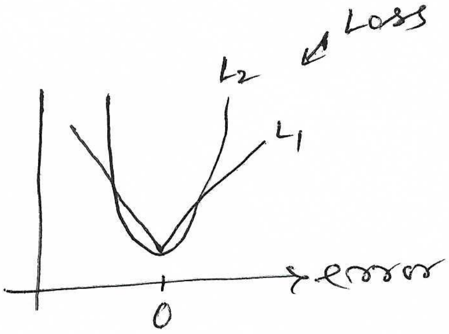
$$Y|X=x = \text{Laplacian}(W^T x, b)$$

$$Y \sim \text{Laplacian}(\mu, b)$$

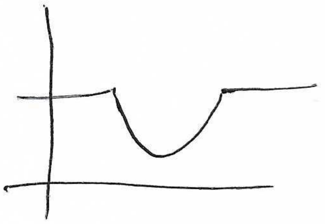
$$\Rightarrow P(y|\mu, b) = \frac{1}{2b} \exp\left[-\frac{|y-\mu|}{b}\right]$$

$$\hat{W}_{MLE} = \arg\min_{W} \sum |y_i - W^T x_i|$$





A robust loss function should look like:



(why?)

Why don't we do that?

↳ Convexity!! extremely hard to optimize.