<u>Lin Regr:</u> (super, regr problem)

model form: $Y = f(\underset{\sim}{x}) = \underset{\sim}{x}^T \beta$

<u>learn:</u> $\hat{f}(\underset{\sim}{x}) = \underset{\sim}{x}^T \hat{\beta}$

How do I "learn" $\hat{\beta}$?

<u>Want:</u> $Y \approx \hat{f}(\underset{\sim}{x}) = \underset{\sim}{x}^T \hat{\beta}$

Need some measure of "goodness"

<u>Ordinary Least Squares (OLS) regression</u>

Going to use a squared-error loss L to measure goodness of fit

On training data

$$N \qquad T\,)^2$$

$$L(\beta) = \sum_{n=1}^{N} \Big(\underbrace{y_n}_{\text{target to predict}} - \underbrace{\underset{\sim}{x}_n^T \beta}_{\text{prediction}}\Big)^2$$

<u>Goal</u>: choose $\hat{\beta}$ to minimize L

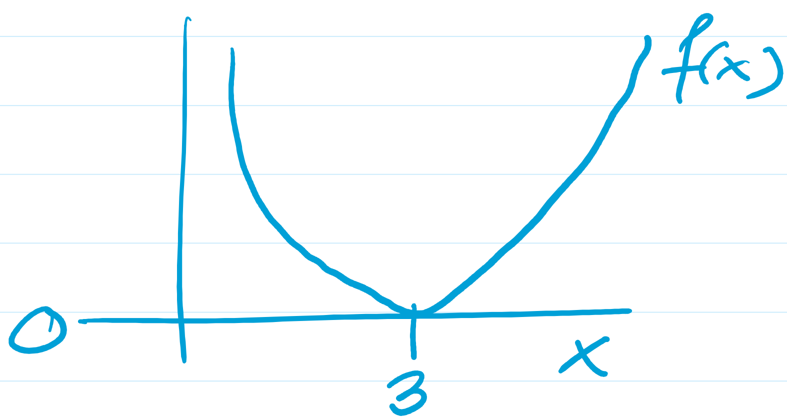
$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}\; L(\beta)$$

---

argmin = value that minimizes

$$\min_x f(x) = 0$$

$$\underset{x}{\operatorname{argmin}}\, f(x) = 3$$



Simple enough to get closed form soln.

Let our design mtx be

$$X = \begin{bmatrix} - \; x_n^T \; - \end{bmatrix} \in \mathbb{R}^{N \times P}$$

and $y = (y_1, \dots, y_N)^T \in \mathbb{R}^N$

$\mathbb{R}^P$

then
$$L(\beta) = \| y - X\beta \|^2$$
$$= \sum_{n=1}^{N} (y_n - x_n^T \beta)^2$$

So then
$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \underbrace{\| y - X\beta \|^2}_{L(\beta):\ \text{fn of } \beta}$$

At this point, Calc 3 problem.

Take deriv wrt β and set equal to zero, solve.

Solve.

Can show:

gradient $= \frac{\partial L}{\partial \beta} = -2(y - X\beta)^T X$

$1 \times P$

Set equal to zero. (Solve for β)

$$\cancel{-}\cancel{2}(y - X\beta)^T X = 0$$

$$\Rightarrow \quad y^T X - (X\beta)^T X = 0$$

$$\Rightarrow \quad y^T X = \beta^T X^T X$$

$$\Rightarrow \quad \boxed{X^T y = X^T X \beta}$$ Normal equations

If $X^T X$ is invertible, then multiply each side by $(X^T X)^{-1}$ to get

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

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

So, overall for lin Regr.

(1) Calc. $\hat{\beta} = (X^T X)^{-1} X^T y$

(2) $\hat{f}(\underset{\sim}{x}_{new}) = \underset{\sim}{x}_{new}^T \hat{\beta}$

Consider predictions on training data

$$\hat{y} = \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} - \underset{\sim}{x}_1^T \hat{\beta} - \\ \vdots \\ - \underset{\sim}{x}_N^T \hat{\beta} - \end{bmatrix} = \begin{bmatrix} - \underset{\sim}{x}_1^T - \\ - x_N^T - \end{bmatrix} \hat{\beta}$$

$$= X \hat{\beta}$$

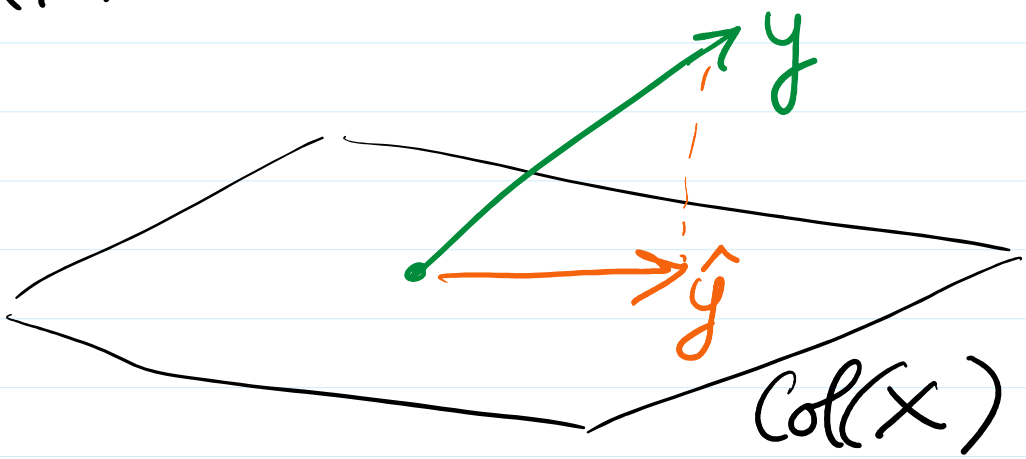
$$= \underbrace{X (X^T X)^{-1} X^T}_{P_X \,=\, \text{proj. onto } Col(X)} y$$

on train.

Col(X)

on train, regr. is just proj. y onto Col(X).

$= P_x y$



Another view

rank(X) = p

$$\underset{n \times p}{X} = U D V^T \qquad D = \underset{n \times p}{\begin{bmatrix} D_* \\ \hline 0 \end{bmatrix}}$$

$$X^T X = V D^T \cancel{U^T U}^{I} D V^T$$

$$= V D^T D V^T \qquad D^T D = [D_* \mid 0] \begin{bmatrix} D_* \\ \hline 0 \end{bmatrix}$$

$$= V D_*^2 V^T \qquad = D_*^2$$

$= D_*^{-}$

$$(X^TX)^{-1} = (V D_*^2 V^T)^{-1}$$

$$= V D_*^{-2} V^T$$

$$X(X^TX)^{-1}X^T = U D \overbrace{\cancel{V^T} V}^{I} D_*^{-2} \overbrace{\cancel{V^T} V}^{I} D^T U^T$$

$$= U \underbrace{D D_*^{-2} D^T}_{\begin{bmatrix} D_* \\ 0 \end{bmatrix} D_*^{-2} \begin{bmatrix} D_* \mid 0 \end{bmatrix}} U^T$$

$$= \left[\begin{array}{c|c} I_p & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$P_X = U_{1:p} U_{1:p}^T$$

$$\hat{y} = P_X y = \overbrace{U_{1:p} U_{1:p}}^{} {}^T y$$

proj. of y onto u_j

$$\hat{y} = P_X y = U_{1:p} U_{1:p}^T y$$

$$= \sum_{j=1}^{p} u_j u_j^T y$$

proj. onto u_j

jth col of U

Another way of getting $\hat{y}$ is

1. Let $X = UDV^T$
2. proj. y onto each of first p cols of U
3. Sum them up

Linear regression is quite flexible.

e.g. $\hat{f}(\underset{\sim}{x}) = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x_j^2$

this is still lin. regr.

Still linear in $\hat{\beta}$.

$$X = \begin{bmatrix} | & | & | & \\ 1 & x_1^2 & x_2^2 & \cdots \\ | & | & | & \end{bmatrix}$$

e.g. $\hat{f}(\underline{x}) = \beta_1 x_1^3 + \beta_2 \sin(x_2) + \beta_3 x_1 x_2 + \cdots$

$$X = \begin{bmatrix} | & | & | \\ x_1^3 & \sin(x_2) & x_1 x_2 \\ | & | & | \end{bmatrix}$$

What about categorical inputs/feats?

e.g. race, gender, color, ...

Can I build a model like

$$\hat{f}(\underline{x}) = \beta_0 + \beta_1 (\text{color})$$

One-hot encoding

$$\begin{bmatrix} R \\ Y \\ B \\ R \\ B \end{bmatrix} \longrightarrow \begin{array}{ccc} R & Y & B \\ \left[\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{matrix}\right. & & \left.\right] \end{array}$$

Issue: what if I include a intercept?

$$X = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ \vdots & 0 & 0 & 1 \\ \vdots & & \vdots & \\ 1 & & \vdots & \end{bmatrix}$$

← one of my cols is a LC of others

$\Longrightarrow$

X^TX not invertible

So what people do typically is throw out one col:

$$X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ \vdots & \vdots & \\ 1 & \vdots & \end{bmatrix}$$

More general issue is that to get $\hat{\beta}$ I solved normal eqns

$$\boxed{X^T X \beta = X^T y}$$

and sometims $X^T X$ not invertible.

Thrm:

$$X^T X \text{ is invertible} \Leftrightarrow \text{rank}(X) = p$$

$= \#$ cols of X

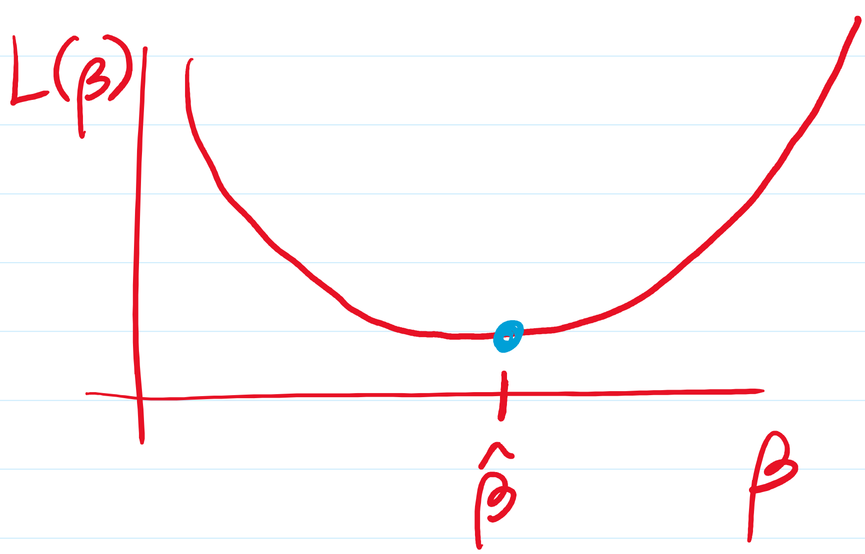
---

when does this happen in reality?

1. Accidentally include a var twice
2. One var is a LC of others
   (e.g. one-hot encoding)
3. If # cols (P) > # rows (N)
4. This can be problematic if X^TX is "almost" not invertible

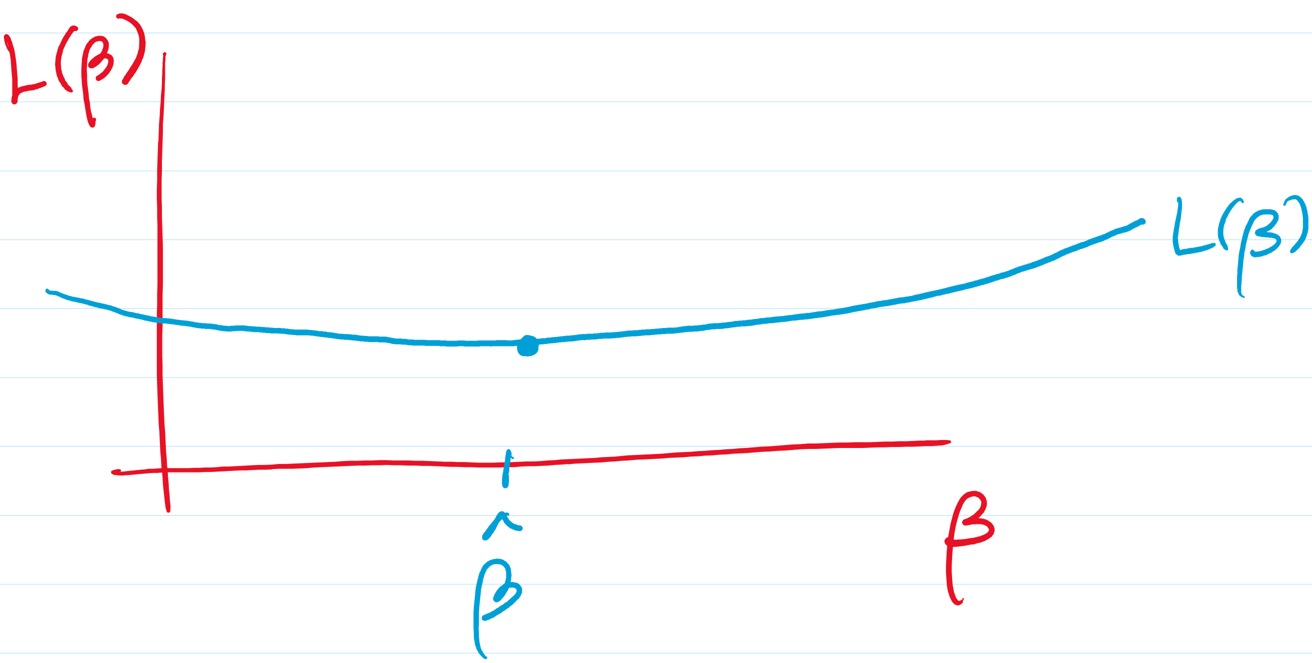
e.g. one var is approx. to a LC of others.

1. X^TX is invertible



(2) $X^T X$ not invertible

(3) $X^T X$ almost not invertible



$\hat\beta$ very sensitive