Validation for model selection

→ Consider a set of M models, $\mathcal{H}_1, \mathcal{H}_2, \ldots \mathcal{H}_M$

e.g. linear, quadratic, logistic etc.

→ Different models have different regularization parameters.

→ How to choose the model?

Use D_{train} to learn $g_1^-, \ldots, g_M^-$

→ Evaluate each on the validation set

$$E_m = E_{val}(g_m^-), \quad m = 1, \ldots, M.$$

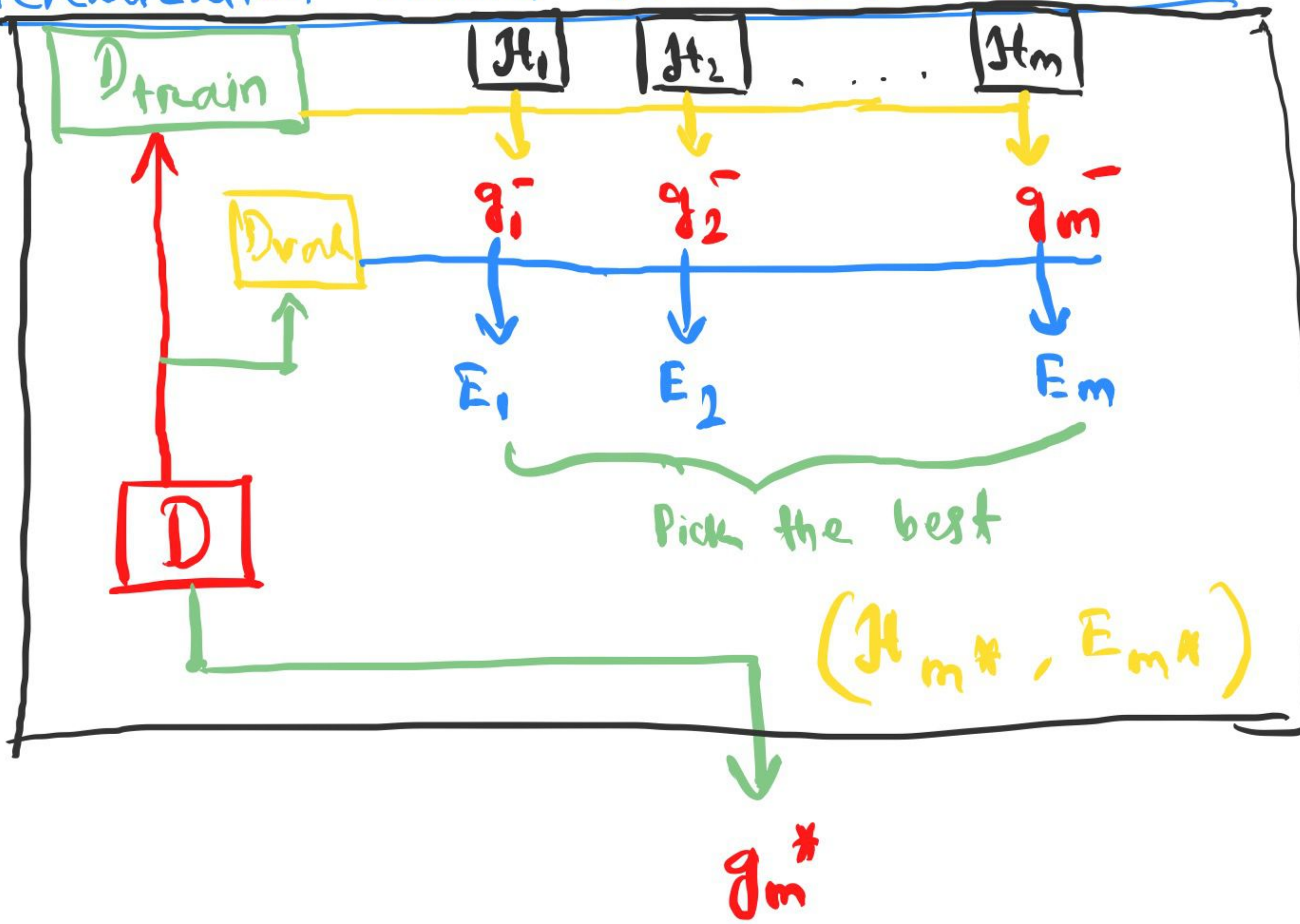
→ E_m is an unbiased estimate of the out-sample error $E_{out}(g_m^-)$.

→ Select the one with the minimum validation error

$$m^* = \underset{m}{\arg\min}\ E_m$$

So the model $\mathcal{H}_{m^*}$ is the best model.

Generalization bound for model selection



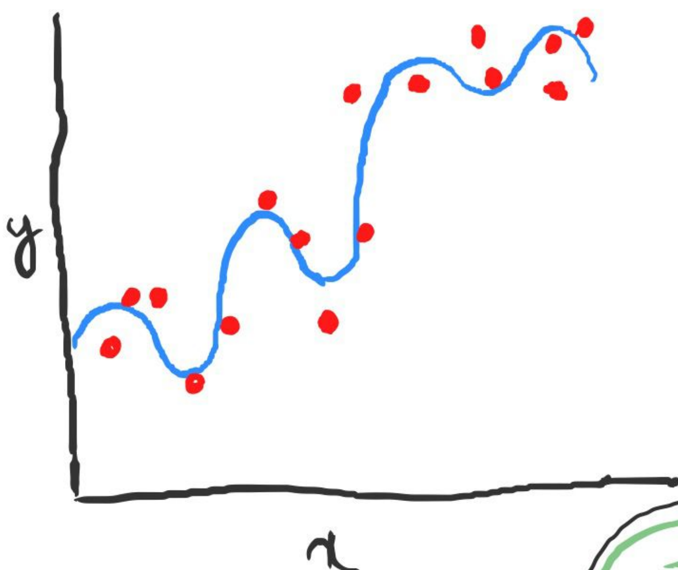
If we choose $g_{m^*}^-$ from $g_1^-, \dots, g_m^-$, you are effectively considering

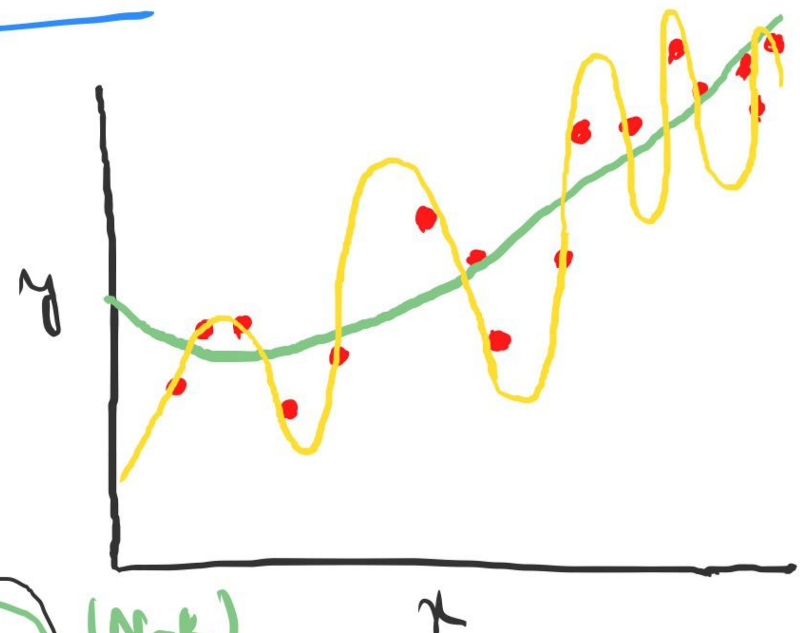
$$\mathcal{H}_{val} = \{ g_1^-, \dots, g_m^- \}.$$

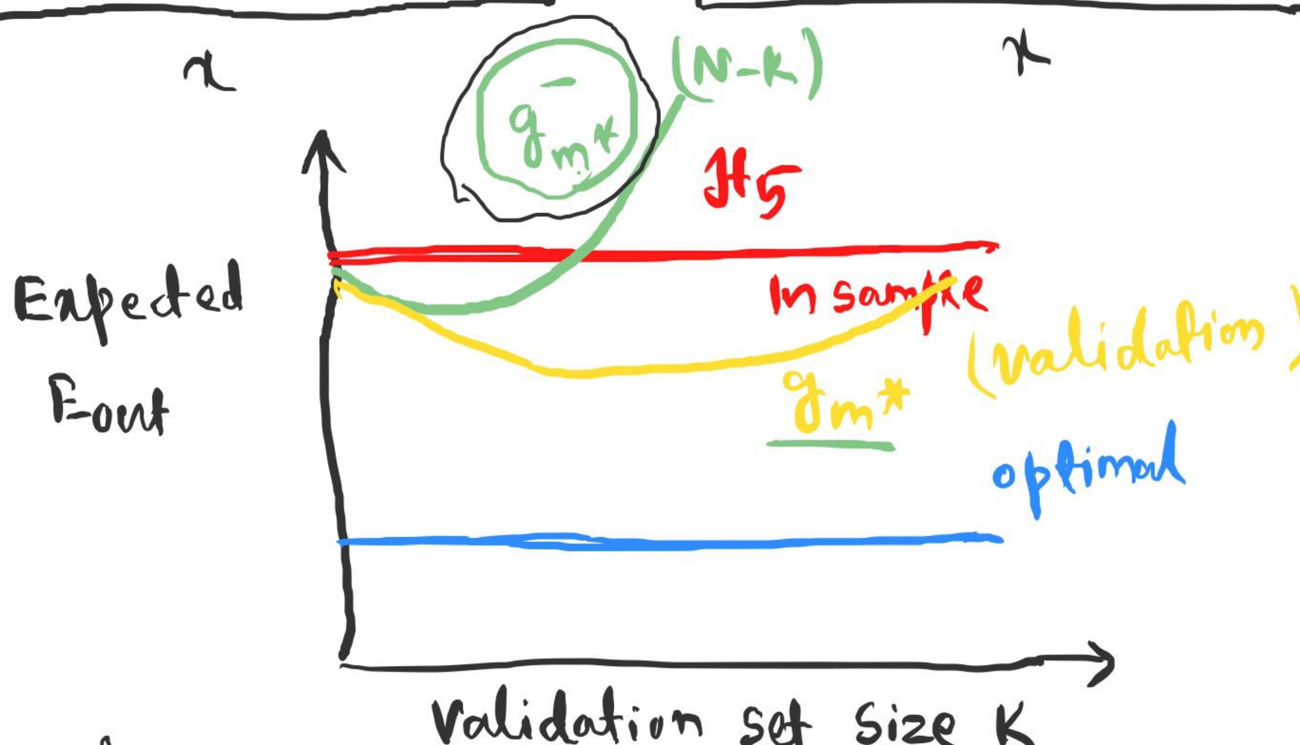
→ Generalization bound

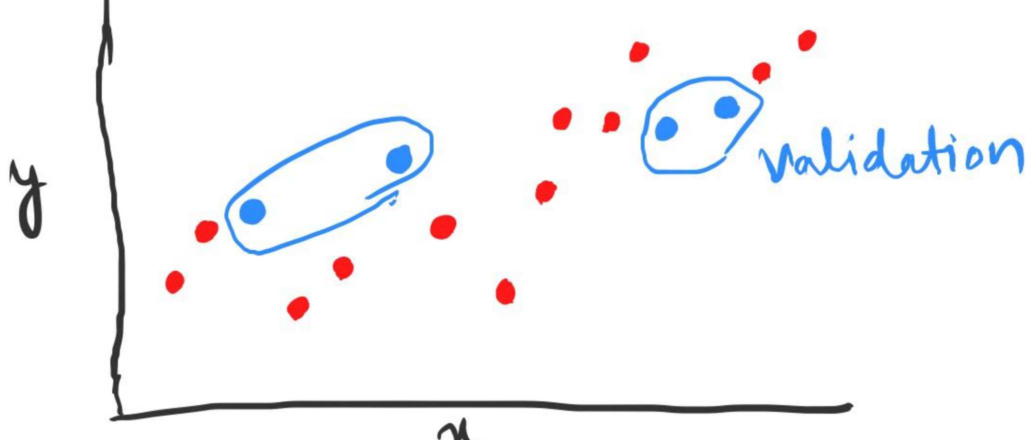
$$E_{out}(g_{m^*}^-) \leq E_{val}(g_{m^*}^-) + O\left(\sqrt{\frac{\log m}{K}}\right).$$

Case study $\mathcal{H}_2$ vs $\mathcal{H}_5$









Observations :

Validation and N-K samples for training

→ $\mathbb{E}[E_{out}(g_{m^*}^-)]$ drops then rise,

→ compared to in-sample, $\mathbb{E}[E_{out}(g_{m^*}^-)]$ uses few samples to validate.

→ This gives a good estimate of out-sample error.

→ As K increases, the estimate improves, if K is too large, then only N-K << for training, so $\mathbb{E}[E_{out}(g_{m^*}^-)]$ rise.

Validation and N samples for training

→ $\mathbb{E}[E_{out}(g_{m^*})]$ will be lower.

→ One should always recycle the validation data for training the final hypothesis.

Cross Validation

A principled way to estimate the out-sample error, without suffering from small K problem.

→ We use the leave-one-out approach, corresponds to a validation set of size $K=1$.

→ Let the data set be

$$D_n = \{(x_1, y_1), \ldots, (x_{n-1}, y_{n-1}), \cancel{(x_n, y_n)}, (x_{n+1}, y_{n+1}), \ldots\ldots, (x_N, y_N)\}$$

Knock out one

→ Remove the nth sample.

→ Learn the hypothesis function

$$g_n^- = \text{learn from } D_n$$

→ Compute the error $e_n = E_{val}(g_n^-)$

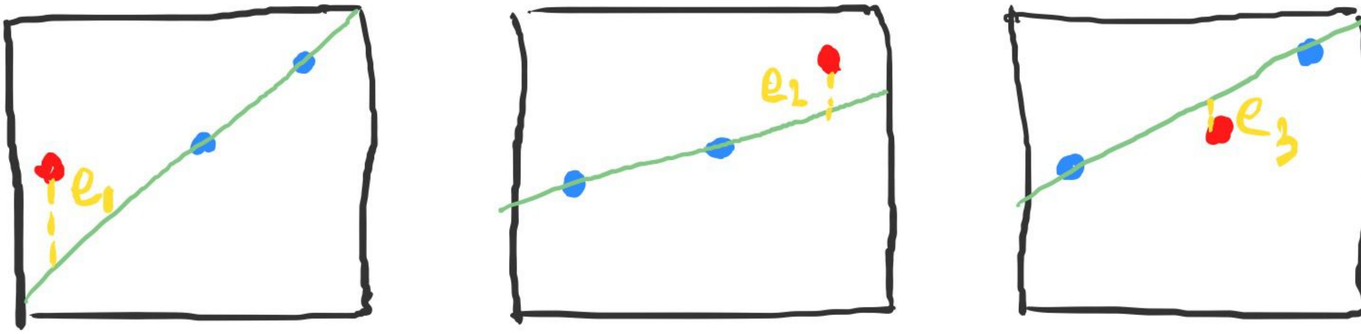
$$= e(g_n^-(x_n), y_n)$$

e_n is the error made by g_n^- on its validation set which is just a single data pt. (x_n, y_n).

→ This will give you $e_1, e_2, \ldots, e_N$.

→ Compute the average (cross validation error)

$$E_{CV} = \frac{1}{N} \sum_{n=1}^{N} e_n$$



Let we have 3 data pts.
2 for training & 1 for validation.

Validation : Use K samples to validate

Cross validation : Recycle the N samples to validate.

P: Cross validation for linear regression

In a linear regression model we have the problem of finding the ω^*

$$\boxed{\omega^* = (A^T A + \lambda I)^{-1} A^T y}$$

The linear regression algorithm is based on minimizing the squared error between $h(x)$ and y

$$E_{out}(h) = \mathbb{E}\left[(h(x) - y)^2\right]$$

is taken w.r.t. joint Prob. dist. $P(x,y)$

Since P is unknown, E_{out} can not be computed.

→ So we resort to E_{in}

$$E_{in}(h) = \frac{1}{N} \sum_{n=1}^{N} \left(h(x_n) - y_n\right)^2$$

→ In linear regression, h takes the form of a linear combination of the components of x.

$$h(x) = \sum_{i=0}^{d} w_i x_i = W^T x \quad , \quad W \in \mathbb{R}^{d+1}$$

In-sample error is a func of W

$$E_{in}(W) = \frac{1}{N} \sum_{n=1}^{N} \left(W^T x_n - y_n\right)^2$$

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

A is the data matrix with x_n as row vectors and y be the column vector with components are target values y_n

$$= \frac{1}{N} \left(W^T A^T A W - 2 W^T A^T Y - Y^T Y \right)$$

The linear regression algorithm is derived by minimizing $E_{in}(w)$ over all possible w, as formulated by the optimization problem

$$w_{lin} = \underset{w}{\operatorname{argmin}}\ E_{in}(w)$$

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

$$\Rightarrow \frac{2}{N}\left(A^T A w - A^T y \right) = 0$$

$$\Rightarrow \quad A^T A w = A^T y$$

$$\boxed{w^* = \left(A^T A + \lambda I\right)^{-1} A^T y}$$

Q: How to estimate the optimum λ ?

Let $H(\lambda) = A\left(A^T A + \lambda I\right)^{-1} A^T$

Prediction $\hat{y} = Hy$

Exercise: Compute the cross validation score:

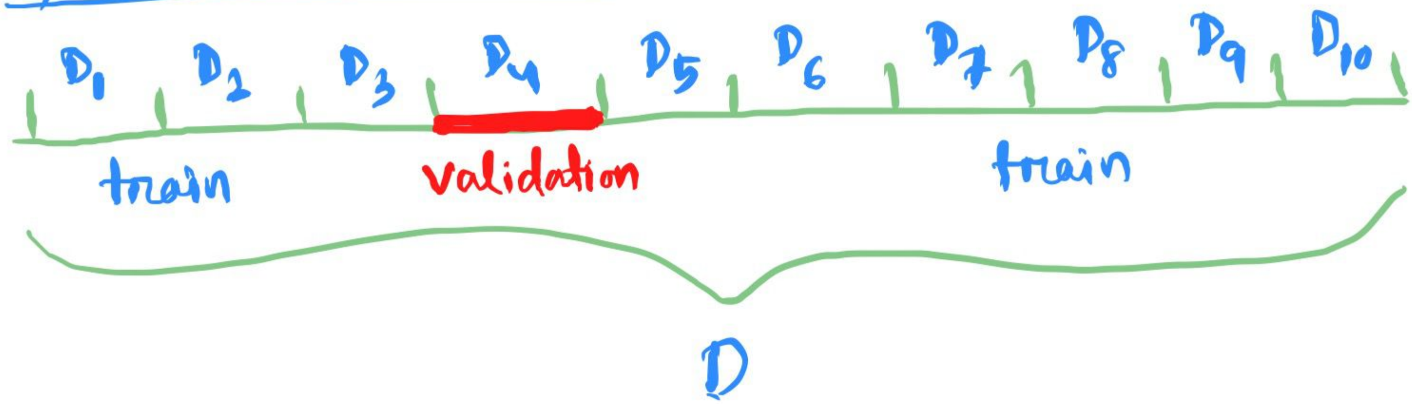
$$\boxed{E_{CV}} = \frac{1}{N} \sum_{n=1}^{N} \left(\frac{\hat{y}_n - y_n}{1 - H_{n,n}(\lambda)} \right)^2$$

diagonal element of matrix H

$$H_{n,n}(\lambda) = x_n^T \left(A^T A + \lambda I\right)^{-1} x_n .$$

and pick λ that minimizes E_{CV}.

2) V-fold validation



→ Leave-one-out : N training sessions.
Each session has $N-1$ pts.

→ V-fold : Partition the data-set into V sessions, then each session has N/V pts.

Train using $D \setminus D_V$

(then you can use a different combination of all these validation and training set)

Advantage : This is computationall efficient.

Rule of thumb: $\boxed{V = 10, \text{ 10-fold CV.}}$