

What makes a good Deval?

Really want: $\hat{f} \perp D_{\text{eval}}$.

↳ broadly construe
whatever process produces
final pred fn. ↗

Generally: means use data not used
for training.*

Reason: $D_{\text{eval}} \perp D_{\text{train}} \Rightarrow D_{\text{eval}} \perp \hat{f}_{D_{\text{train}}}$
(approx.)

Why is D_{train} not a good candidate for
Deval?

After all, D_{train} is real thot tells us
what (1 - fix)

about $y = f(x)$.

Often \hat{f} is highly dependent on D_{train}
and so then if $D_{\text{eval}} = D_{\text{train}}$ we
wouldn't have $\hat{f} \perp D_{\text{eval}}$.

Trable: if \hat{f} follows D_{train} too closely,
(interpolate/memorize) it will "learn"
apparent patterns that don't generalize
called: over-fitting

Problem: - \mathcal{X} may be diff from $\mathcal{X}_{\text{train}}$
- given $\mathcal{X} = \mathcal{X}_{\text{train}}$, y might be
different

D_{train} ain't whole picture.

Reasonable rule of thumb:

Find $D_{\text{eval}} \perp D_{\text{train}}$ so that $\hat{f} \perp D_{\text{eval}}$.

Note: converse is false

$D_{\text{train}} \not\perp D_{\text{eval}} \not\Rightarrow \hat{f} \not\perp D_{\text{eval}}$.

Only really have a problem when \hat{f} overfits

e.g. let $\hat{f}(x)$ be a indep draw from $U(0,1)$.

Then $\hat{f} \perp D_{\text{train}}$ so can use $D_{\text{eval}} = D_{\text{train}}$.

more realistic: \hat{f} only lightly uses training data
(lots of data, fit really simple model)

Using D_{train} to eval isn't worst sin.

Problem when use a very flex method that
can very closely follow D_{train} .

problem even use a very flex method that
can very closely follow D_{train} .

e.g. If $D_{\text{eval}} = D_{\text{train}}$ and choose K
to minimize MSE

then basically will always choose $K=1$
to get $MSE=0$.


e.g. Fit regression w/ polynomial degree

$$y = \beta_0 + \beta_1 X + \beta_2 X^2 + \dots + \beta_q X^q$$

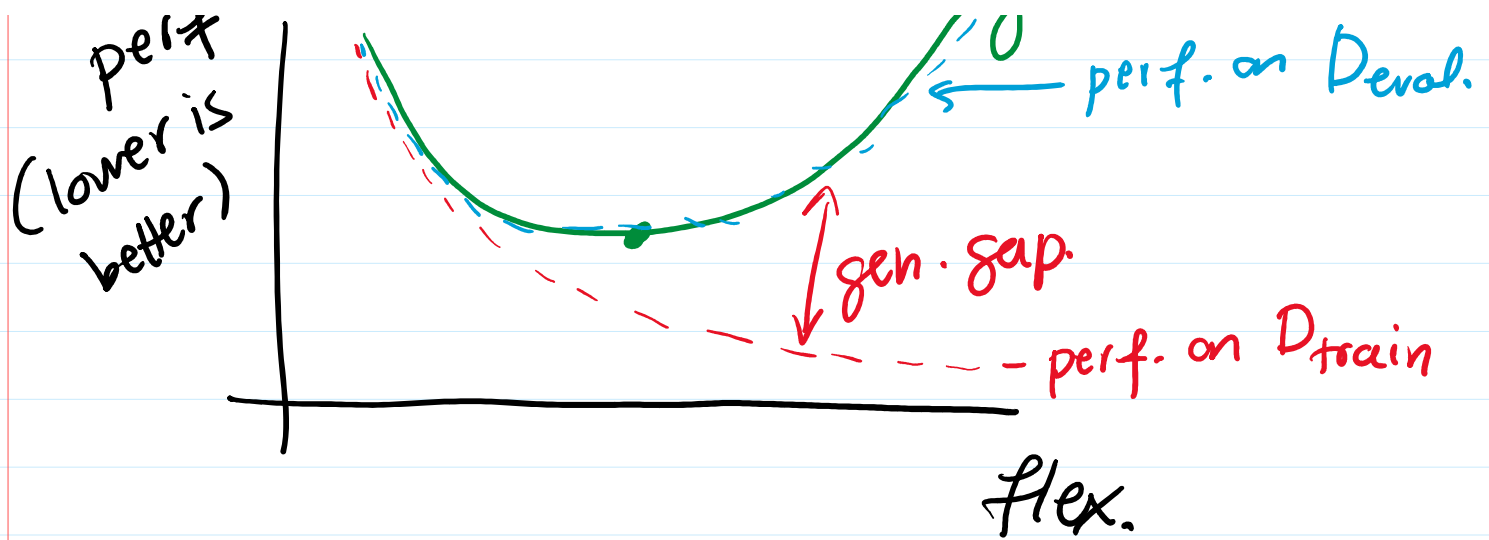
and choose q based on training MSE.

Will choose $q = N-1$ to interp training
data and get $MSE=0$.

General picture:

perf
:c | 

gen. perf.
← perf. on D_{eval} .



Heuristic: choose Deval \perp D_{train} .

Ways to do:

① Split data: $p\%$ into D_{train}

$(1-p)\%$ into Deval

e.g. $p \approx 95\%, 90\%, 50\%$

potential problems:

- might be sensitive to particular split
- training on less data so my model may be overfitted

eval. perf. is pessimistic.

② resampling: do split multiple times and average the est. perf.

e.g. X-validation

- Split data into B equally sized "folds"

- For $b = 1, \dots, B$

\hat{f}_{-b} = train on all but b^{th} fold

m_b = perf. of \hat{f}_{-b} on other folds

$$- m = \frac{1}{B} \sum_{b=1}^B m_b.$$

↑ overall perf. metric

OK, after CV which of B models do I use?

Ans: None of them.

... all ...

I should use all my data to train a final model.

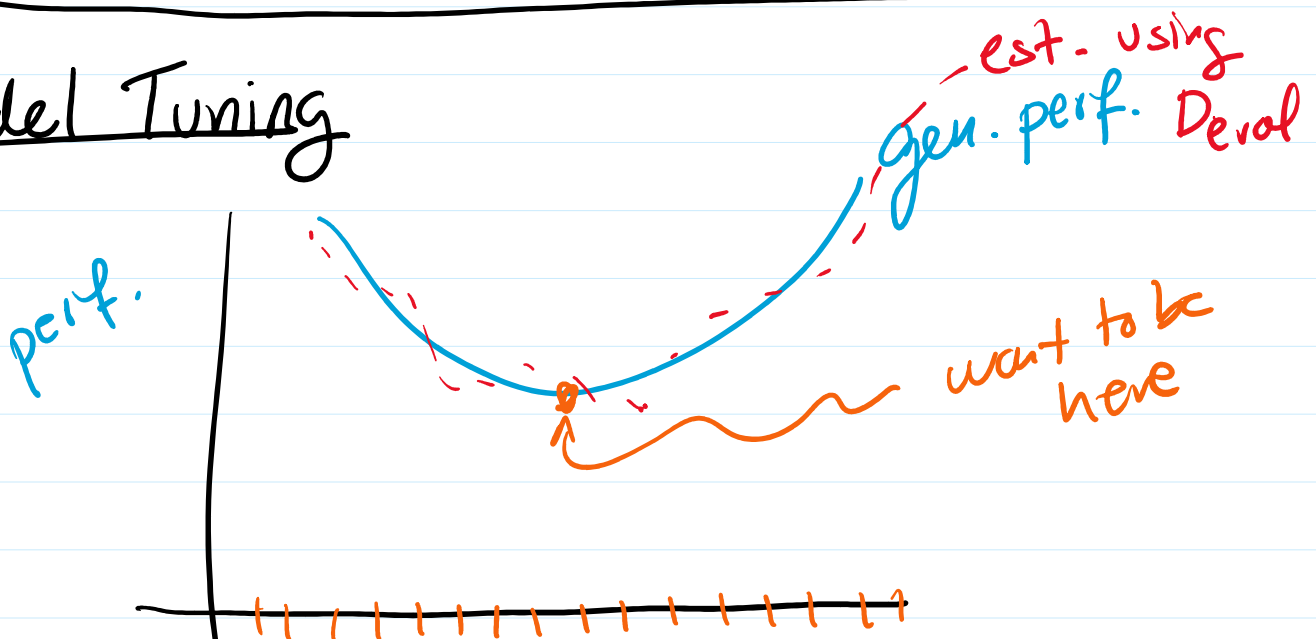
Advantages of resampling: less sensitive to particular split

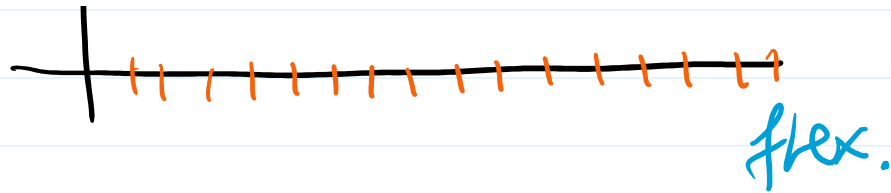
Problem: still pessimistic b/c only use $\frac{B-1}{B}$ % to train.

Can do CV w/ $B=N$.

Called leave-one-out CV (LOOCV).

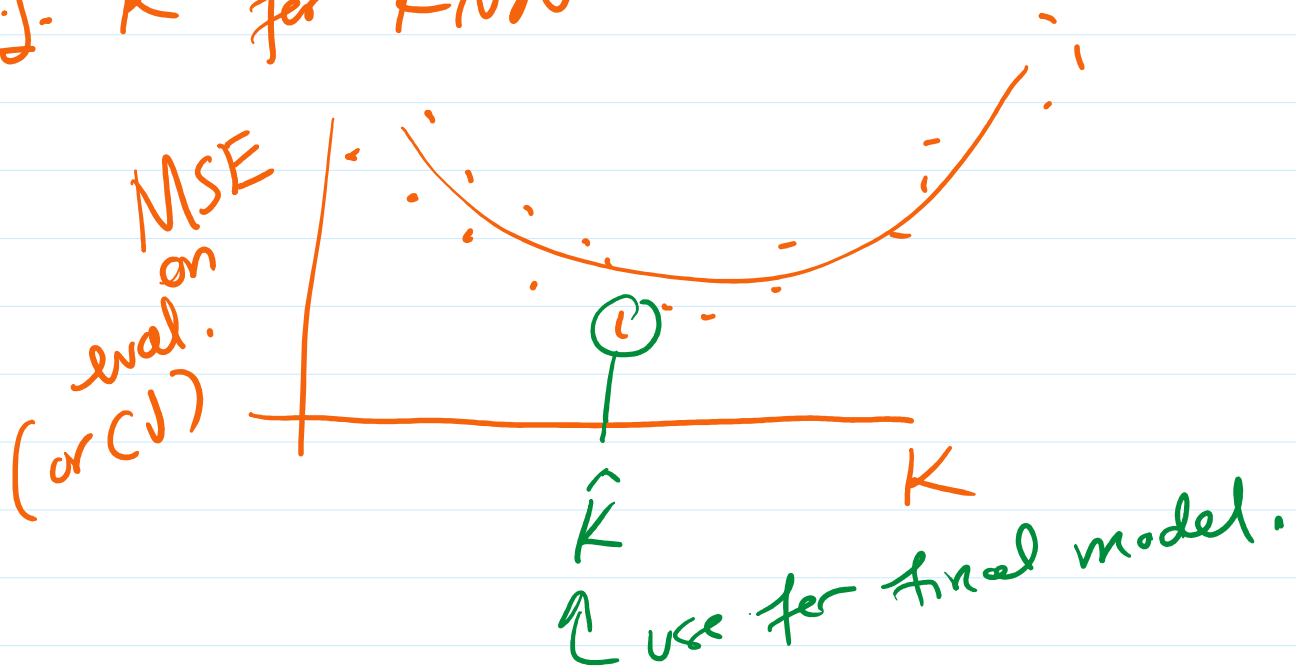
Model Tuning





Idea! Estimate gen. perf. over possible flex. settings, choose value of flex. to minimize error.

e.g. K for KNN



$$\hat{f} = \hat{f}_{\hat{K}}$$

Beware: can overfit to D_{eval} , too.

Known as model selection bias

Known as model selection bias
or " " overfitting.

Rule: $\text{Devel} \perp \hat{f}$.
↑ *constitute this broadly.*

Q: Above, is $\hat{f}_k \perp \text{Devel}$? No 😞

Is this an issue?

Depends on how hard I try to tune the model.

Ways to avoid

① don't try too hard to over-optimize meta-params

② hold out some data used to choose params:

cross partition

train/val/test

③ use re-sampling like x-val.
