Model Selection and Assessment

DS-6030 | Spring 2026

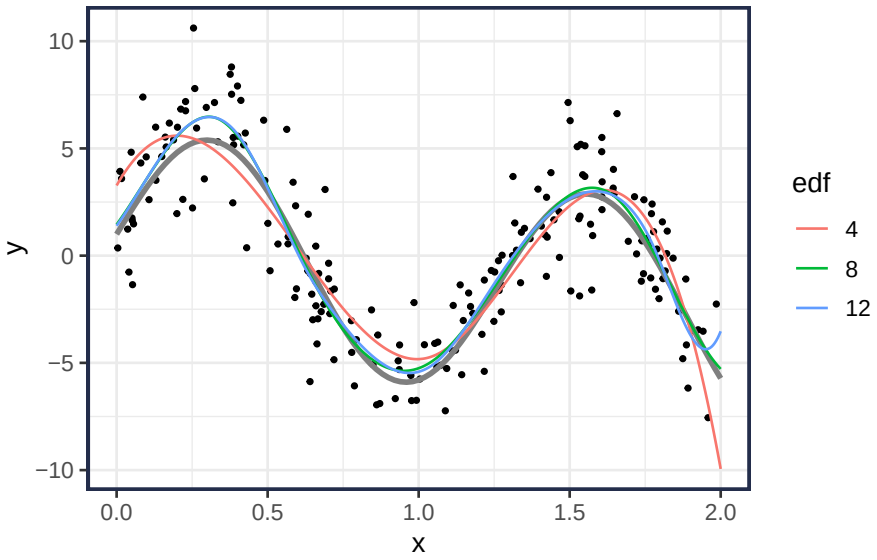
modeling.pdf

Table of contents

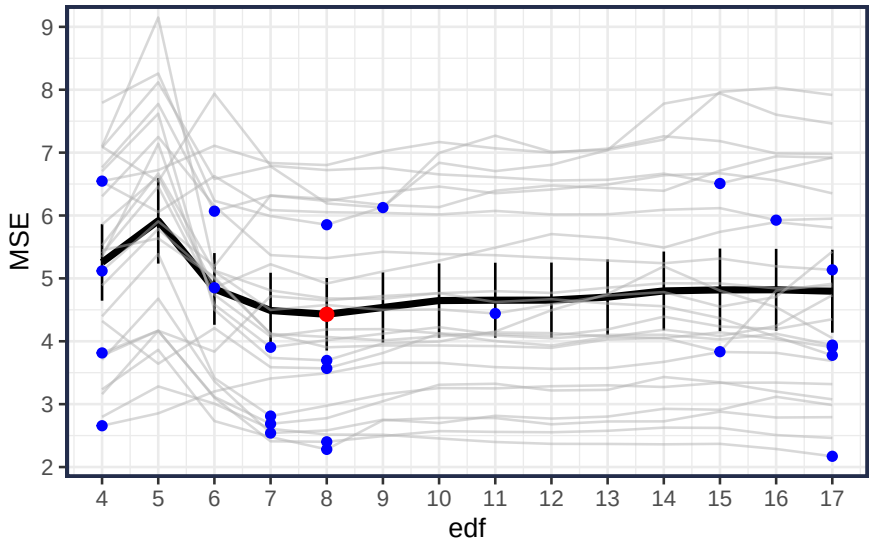
1 Modeling CV error

Cross-validation gives us noisy measurements of performance; modeling those measurements is the difference between simple ranking and understanding for decision-making.

I've generated `n_train = 200` observations and plotted a few b-spline fits.



The tuning parameter is *edf*, the effective degree of freedom, which is the number of spline parameters that get estimated. I'll use 25 iterations of monte-carlo cross-validation with 20 hold-out and 180 training samples (90-10 split) each iteration.



A few things to notice:

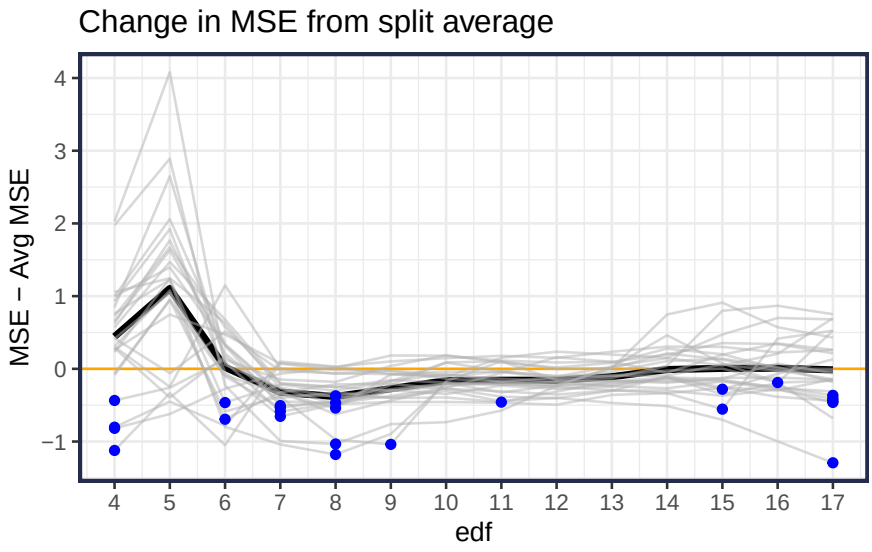
1. The solid black line is the usual cross-validation average. This would select edf = 8 as the optimal model.
2. Several edfs look to have comparable MSE to the cross-validation selection.

3. Some splits select a very low edf (simpler model) while other splits select a very high edf (complex model).
4. The estimate MSE varies from 2.2 to 6.5.

1.1 Accounting for resample iteration (fold/split)

Notice that the performance estimates vary substantially by the split. Some splits produce MSE estimates as low as 2.2 while others as high as 6.5. **The variability due to split is greater than the variability due to edf**.

To help account for split, I'm going to look at the difference between the MSE at each edf and the average MSE for the split. In the plot below, each grey line shows the MSE(i, edf) - AvgMSE(i) where AvgMSE(i) is the average MSE for split i.



This helps us see how the edf varies after adjusting for split effect. Let's put this into a linear model formulation:

$$\begin{aligned}\text{MSE(edf, split)} &= \text{Intercept} + \text{Model Effects} + \text{Split Effects} + \epsilon(\text{edf, split}) \\ &= \alpha_0 + \sum_{d=4}^{17} \beta_d \, \mathbb{1}(\text{edf} = d) + \sum_{i=1}^{k} \gamma_i \, \mathbb{1}(\text{split} = i) + \epsilon(\text{edf, split})\end{aligned}$$

Fitting a least squares (linear regression) model, we get the following estimates. Note: this is repeated-measures ANOVA.

```
lm(mse ~ factor(edf) + iter)
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.34	0.16	32.50	0.00
edf: 5	0.66	0.14	4.70	0.00
edf: 6	-0.42	0.14	-2.99	0.00
edf: 7	-0.76	0.14	-5.42	0.00
edf: 8	-0.82	0.14	-5.84	0.00
edf: 9	-0.72	0.14	-5.07	0.00
edf: 10	-0.61	0.14	-4.30	0.00
edf: 11	-0.60	0.14	-4.26	0.00
edf: 12	-0.60	0.14	-4.25	0.00
edf: 13	-0.55	0.14	-3.91	0.00
edf: 14	-0.45	0.14	-3.21	0.00
edf: 15	-0.44	0.14	-3.10	0.00
edf: 16	-0.43	0.14	-3.08	0.00
edf: 17	-0.46	0.14	-3.24	0.00
iter: 2	0.17	0.19	0.90	0.37
iter: 3	-2.36	0.19	-12.54	0.00
iter: 4	1.63	0.19	8.66	0.00
iter: 5	2.08	0.19	11.04	0.00
iter: 6	2.13	0.19	11.30	0.00
iter: 7	-0.17	0.19	-0.91	0.37
iter: 8	-1.68	0.19	-8.91	0.00
iter: 9	2.27	0.19	12.01	0.00
iter: 10	-0.58	0.19	-3.09	0.00
iter: 11	-0.51	0.19	-2.71	0.01
iter: 12	-0.41	0.19	-2.16	0.03
iter: 13	-0.86	0.19	-4.58	0.00
iter: 14	0.64	0.19	3.42	0.00
iter: 15	-0.28	0.19	-1.50	0.13
iter: 16	1.34	0.19	7.11	0.00
iter: 17	-0.50	0.19	-2.65	0.01
iter: 18	-2.08	0.19	-11.03	0.00
iter: 19	-1.44	0.19	-7.61	0.00
iter: 20	1.21	0.19	6.44	0.00
iter: 21	-2.13	0.19	-11.28	0.00
iter: 22	0.68	0.19	3.59	0.00
iter: 23	1.89	0.19	10.02	0.00
iter: 24	-1.42	0.19	-7.56	0.00
iter: 25	-1.85	0.19	-9.83	0.00

Model Summary:

metric	value
sample size	350
number of predictors	38
Adjusted R^2	0.90
sigma_hat	0.50
sigma_hat^2	0.25

Your Turn #1 : Model Summary

1. What is the size of the data used to fit the model? Are all observations independent?
2. How many parameters are estimated?
3. Is it enough data?
4. What can we do to get better estimates?

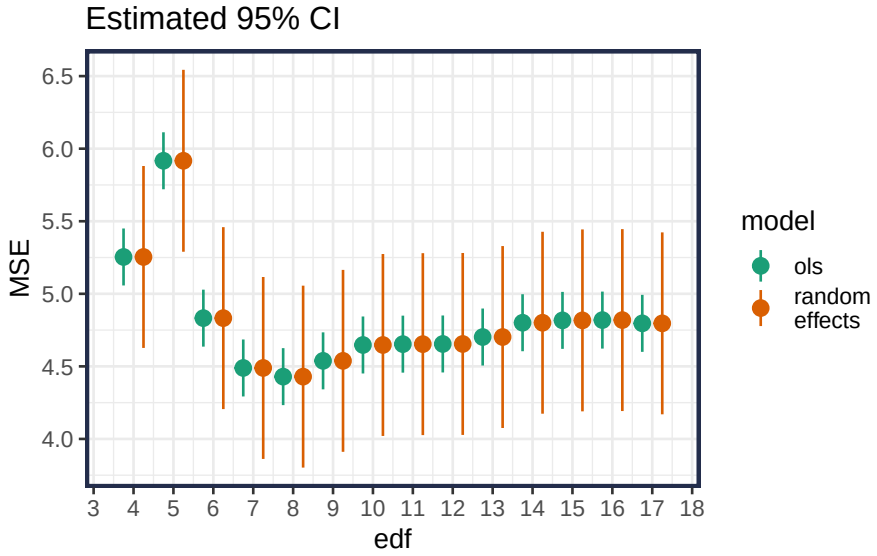
1.1.1 Random Effects

The edf coefficients are what we care about. These are supposed to reflect the true EPE. However, our estimates of edf are impacted by the iteration effects. In other words, all MSE's from iteration i share a common random "shock". So iteration i carries about 1 piece of information instead of many. Thus the uncertainty in the edf effects is too low if we don't correctly handle the iteration shocks. Note: behind the scenes, this is very similar to adding a ridge penalty on the iteration parameters.

```
library(lme4)
lme4::lmer(mse ~ factor(edf) + (1 | iter), data = cv_mse )
```

Source of variability	What it represents	Variance	Std dev	Meaning
Iteration	Differences between resamples/iterations	2.10	1.45	Each resample shifts the whole MSE curve up or down by a random amount with std dev of 1.45
Noise (epsilon)	Remaining noise within an iteration	0.25	0.50	Unexplained randomness. After accounting for edf and iteration, the MSE still varies by a random amount with std dev of 0.50.

Here is a graphic showing the estimated coefficients and 95% confidence intervals under each modeling approach.

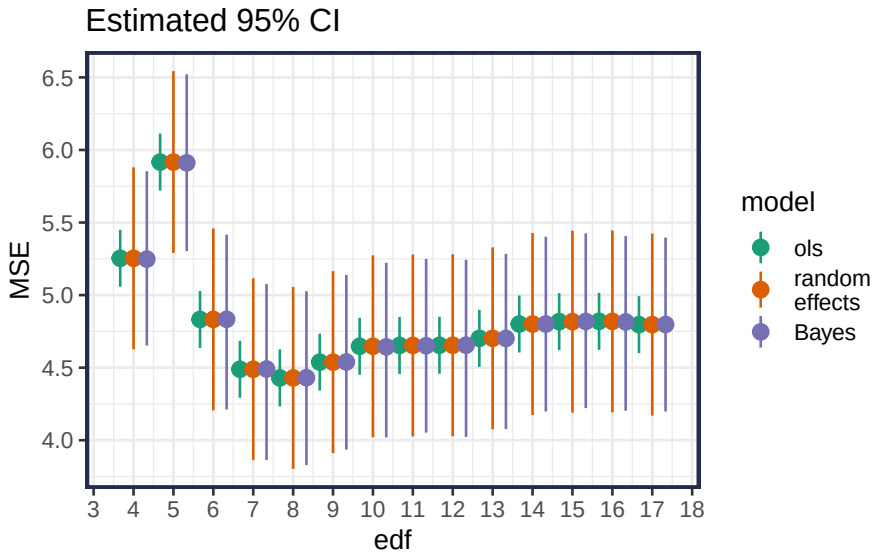


1.1.2 Bayesian Estimation

Without much additional effort we can get a fully Bayesian estimation. This is an example of a hierarchical Bayes model.

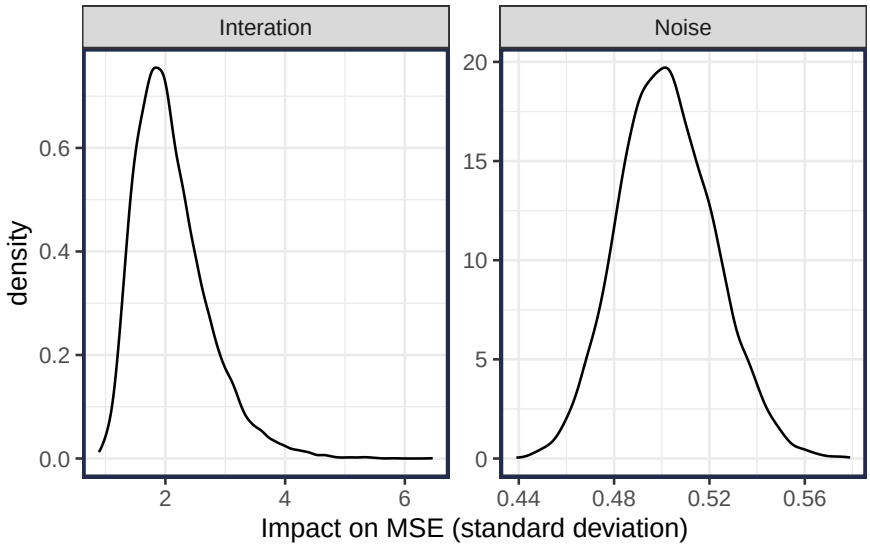
```
library(rstanarm)
stan_lmer(mse ~ factor(edf) + (1 | iter), data = cv_mse)
```

Details



From the posterior samples, it is straightforward to estimate the distribution of the best edf:

edf	n	p
edf: 4	0	0.0%
edf: 5	0	0.0%
edf: 6	3	0.0%
edf: 7	3222	26.8%
edf: 8	6514	54.3%
edf: 9	1572	13.1%
edf: 10	212	1.8%
edf: 11	214	1.8%
edf: 12	188	1.6%
edf: 13	62	0.5%
edf: 14	4	0.0%
edf: 15	3	0.0%
edf: 16	2	0.0%
edf: 17	4	0.0%



1.1.3 Smooth

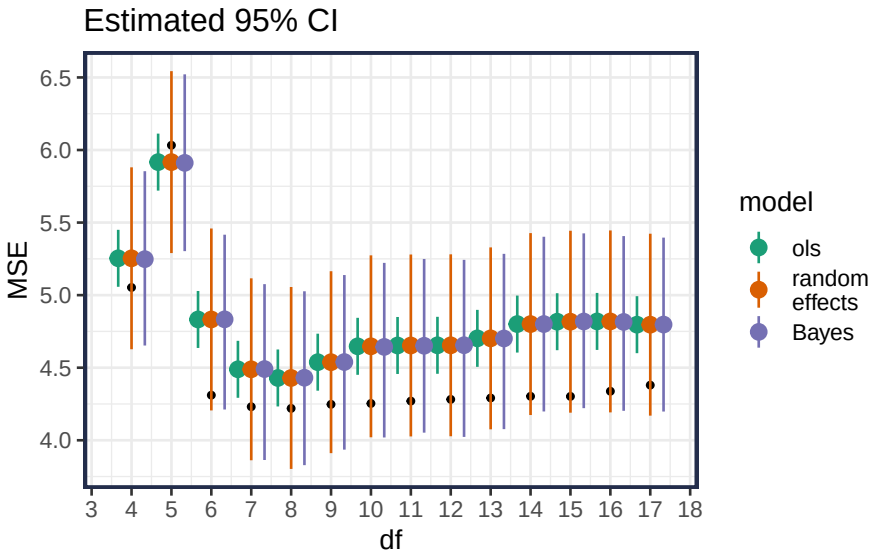
Because the edf should be relatively smooth, we could enforce the smoothness in the edf effects. For example by adding a spline:

```
library(rstanarm)
stan_lmer(mse ~ s(edf) + (1 | iter), data = cv_mse)
```

1.1.4 How did we do?

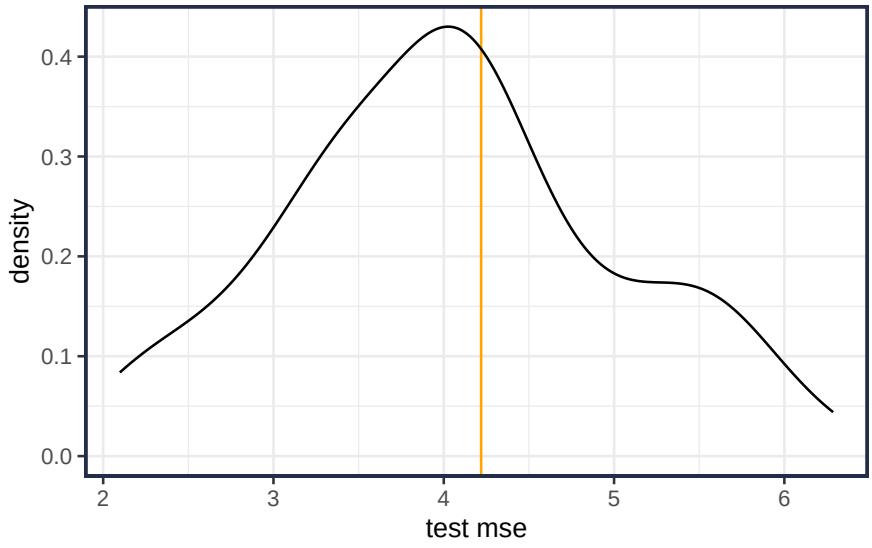
Using a test data of n_test = 100,000 we should get a better approximation of the true performance.

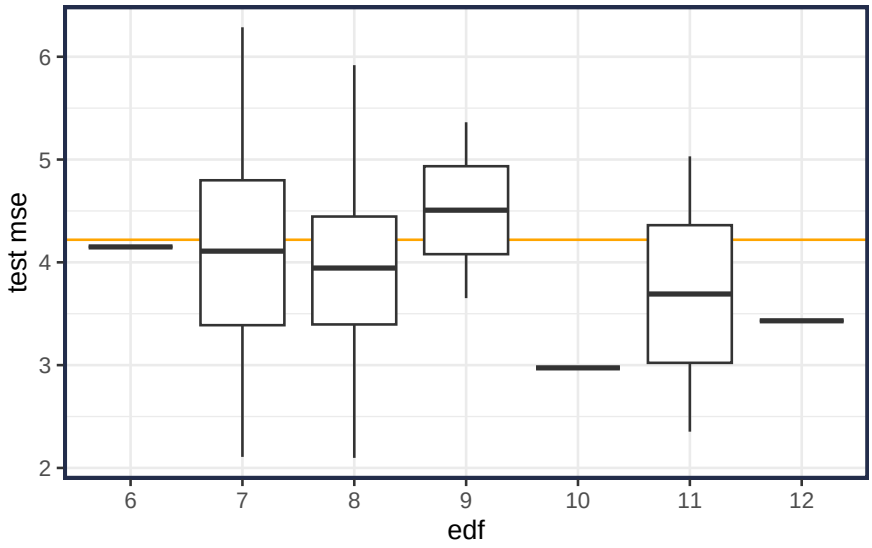
edf	mse
4	5.05
5	6.03
6	4.31
7	4.23
8	4.22
9	4.25
10	4.25
11	4.27
12	4.28
13	4.29
14	4.30
15	4.30
16	4.34
17	4.38



1.2 Nested Cross-Validation

Setting up nested cross-valiation. I'm using an outer loop of 100 iterations with 20 monte-carlo hold-out observations (test). The inner loop is 25 iterations of 20 monte-carlo hold-out observations. The inner-loop is for selecting the optimal edf and fitting a final model using all the available data with the optimal edf.


density
0.0
0.1
0.2
0.3
0.4
2
3
4
5
6
test mse


test mse
2
3
4
5
6
6
7
8
9
10
11
12
edf