Bias-Variance, Cross Validation, Regularization

Raguvir Kunani

Discussion 10

October 30, 2019

Bias-Variance Overview

Some models are better than others. But how do we quantify that?

1. We could say the model with the lowest training error is the best
2. We could say the model with the lowest test error is the best

> We can't choose the model with the lowest test error. Why?

We can only look at the test set once! We can't retrain the model after looking at the test set

We characterize every model by its **bias-variance decomposition**:

$$\underbrace{\mathbb{E}\left[(Y - f_{\hat{\beta}}(x))^2\right]}_{\text{model risk}} = \underbrace{\sigma^2}_{\text{observation variance}} + \underbrace{(h(x) - \mathbb{E}\left[f_{\hat{\beta}}(x)\right])^2}_{(\text{model bias})^2} + \underbrace{\mathbb{E}\left[(\mathbb{E}\left[f_{\hat{\beta}}(x)\right] - f_{\hat{\beta}}(x))^2\right]}_{\text{model variance}}$$

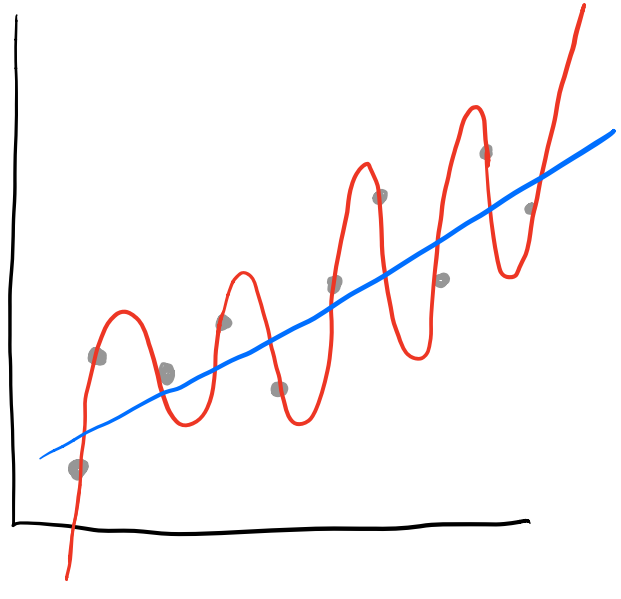
actual; "average" model; "avg" model; current model

Takeaway: every model has a bias and a variance, which we'll use to evaluate the model

low variance = model generalizes well outside of training data

Bias-Variance Tradeoff

Almost always, lower bias means higher variance and vice versa. This contradicts our goal of achieving a model with low bias AND low variance.



red model: high variance, low bias

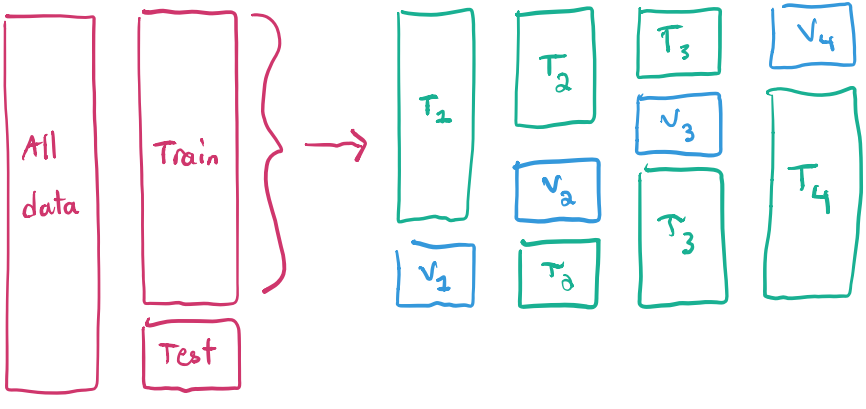
blue model: low variance, high-ish bias

We want models like the red model! We live with somewhat high bias in exchange for a low variance model that generalizes well

→ abbreviated CV

Cross-Validation Overview

In general, **cross-validation** is used to choose between a set of things. For now, those things are models. The goal of cross-validation is to *simulate* evaluating our model on the test data.

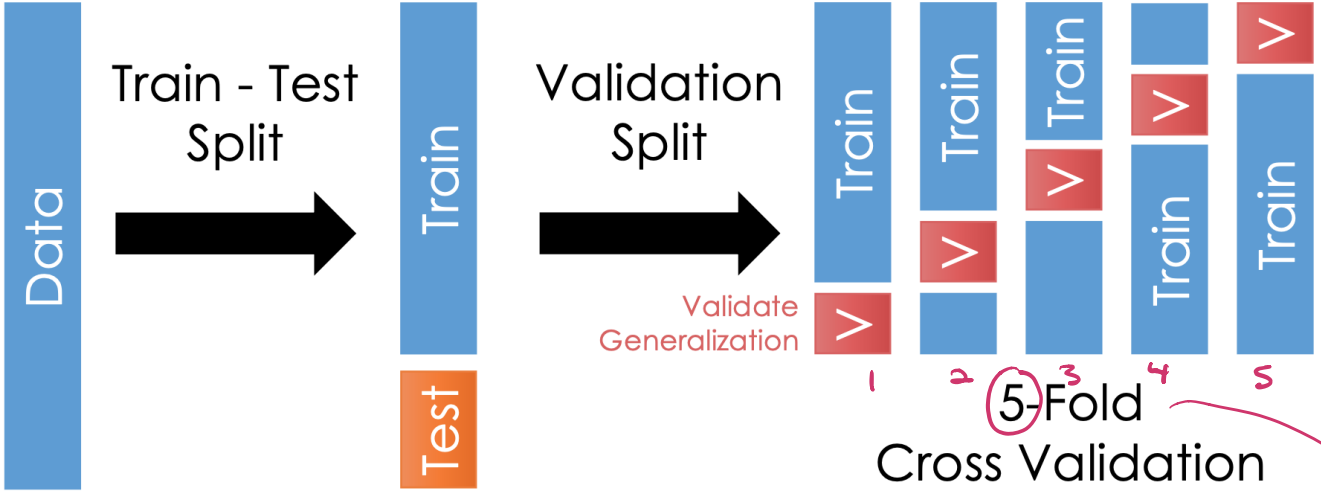


4 fold CV w/ 3 models M1, M2, M3

1. Train M1 on T1, compute error E_1 on V_1
2. Train M1 on T2, compute error E_2 on V_2
3. Train M1 on T3, compute error E_3 on V_3
4. Train M1 on T4, compute error E_4 on V_4
5. Validation error for M1: $\frac{V_1 + V_2 + V_3 + V_4}{4}$
6. Repeat 1-5 for M2 and M3

At the end of cross-validation, we find the model with the lowest validation error and choose that model as the model to train on the entire training set.

Cross-Validation in a Pretty Picture



You can think of cross-validation as using "practice tests" (the validation sets) to find out what the best model is for the "final exam" (the test set).

folds refer to # of validation sets

Why bother with cross-validation?

Remember that every model has a model bias and a model variance. We already know how to get a model with low model bias, that's what training a model does by definition!

But how do we get a model with low variance? Remember that a model with low variance generalizes well, or in other words does well at predicting data it has not seen before.

How does this relate to cross-validation?

Taking "practice tests" lets our model know how its doing without using the test set

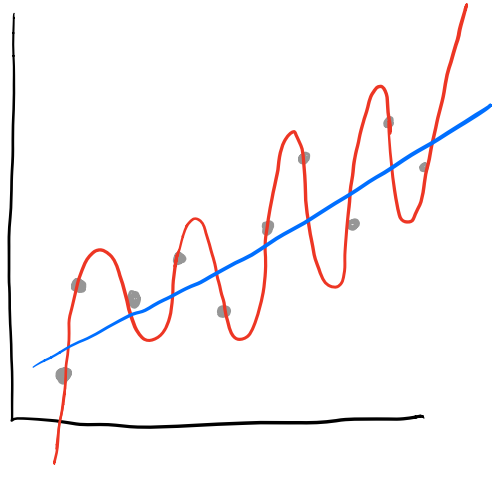
What are some problems with cross-validation?

Computationally expensive: have to train each model on each fold

Regularization Overview

We can do better than cross-validation when trying to achieve low model variance by using **regularization**.

Regularization penalizes models from having large weights on features. Penalizing large weights means discouraging the model from using large weights. Why does this help us with model variance?



Red model equation: $y = \beta_0 + \beta_1 x + \beta_2 \sin(x)$

β_i are the weights

If we penalize having large β_i, then we discourage the model from setting β_i to be large values. This means we are discouraging the model from putting weight on the $\sin(x)$ term! No $\sin(x)$ gives us a lower variance model.

Regularization in Math

"fooling" the model into thinking a given β is worse than it actually is

Penalizing models for having large weights is done by adding a term to the loss function our model trains on:

$$\arg\min_{\hat{\beta}} \frac{1}{n}||y - X\beta||_2^2 \implies \arg\min_{\hat{\beta}} \frac{1}{n}||y - X\beta||_2^2 + \lambda S(\beta)$$

L_2 loss

L_2 loss + regularization

The $\lambda S(\beta)$ term is the penalty. To penalize model weights, we can choose

$= \sum_i \beta_i^2$

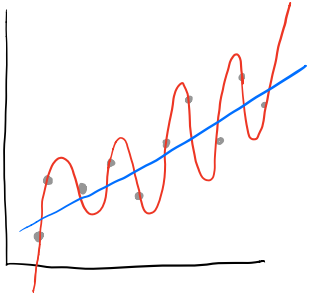
1. $S(\beta) = ||\beta||_2^2$ (ridge regression)
2. $S(\beta) = \sum_{i=1}^{n} |\beta_i|$ (LASSO regression)

> Check your understanding: What condition *must* hold for $\lambda S(\beta)$?

$\lambda S(\beta) \geq 0$!

How/why does regularization penalize model weights?

The process of training is trying to find the model with lowest bias. By definition, this means training is trying to find the model with zero bias. But that's not what we want:



If we're just trying to get the model with minimal bias, we get the red model. But we don't want that! We use regularization to influence the training process to choose the blue model.

When I penalize the model weights, I'm telling the model to try and achieve low bias without letting the β_i be large for any of the features. But since some of the β_i must be fairly large for us to have a good model, only the "unnecessary" β_i are affected.

this is the ultimate effect of regularization

Feedback Form

This *anonymous* form is for me to learn what I can do to ensure you all get the most of discussion and lab. This form will be open all semester, and I'll be checking it regularly. Be as ruthless as you want, I promise my feelings won't get hurt.

Feedback Form: tinyurl.com/raguvirTAfeedback