Machine Learning

•••

29 March 2018
PLSC 309

Review

- In previous weeks, we have learned OLS
- This is a procedure to minimize the sum-of-squared errors for a linear model
- However, OLS requires the sample to be perfectly representative of the population, with a large enough size to capture complex relationships
- If this is not met we have a problem with *external validity*
- It means that we can have a low *in-sample error*, but a high *out-of-sample error*

Review: bias-variance trade-off

- If a model is wrong, it is biased
  - g(x) does not approximate f(x)
  - It isn't *complex* enough
- If a model is inconsistent, it has high variance
  - E(g(x)) has a wide range of possible estimates
  - It isn't *simple* enough
- We want to find the MVUE, or the minimum-variance unbiased estimator

Calculating in-sample model error

- The best square fit wants to find the smallest residuals in either direction
- We can do this by first squaring each of our residuals
  - $e_1^2, e_2^2, e_3^2, \ldots$
- And then summing the residuals
  - $e_1^2 + e_2^2 + e_3^2 + \ldots$
- This is known as the *sum of least squares*
- A regression model calculated with the sum of least squares is known as *Ordinary Least Squares (OLS) Regression*

But what about calculating out-of-sample error?

- We calculated in-sample error by summing the squared differences between $\hat{Y}$ and Y
- In this case, our Y's came from in-sample, so this is *in-sample error*
- If we had Y's from outside of our sample, we could then compute out-of-sample error
- This is contradictory: if you have the data, then it's in your sample!

Review: bootstrap logic

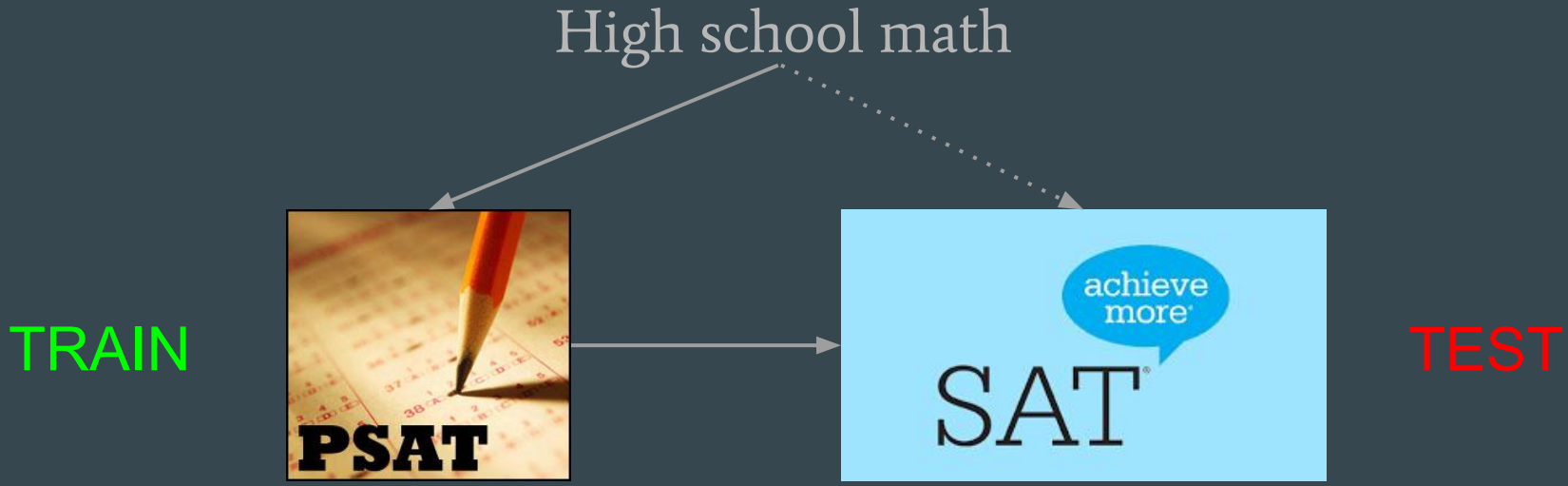
- We faced a similar problem in the bootstrap
- We wanted to know if our sample was from an alternative distribution, but there was no way to figure out what that distribution was
- So we used the data we had to make a fake one!

Fake it ‘til you make it

- We’re going to use a similar approach to finding the out-of-sample error
- To do this, we will take our data and split it into two data-sets
  - One set is called the training set
  - The other is called the test set
- Now we have two datasets
- The training set is just like our regular sample
- But the test is our phony “unobserved data” that we will use to compute out-of-sample error

Training and test sets

We can simulate the process of learning from the real world, by dividing our data into training (our dataset) and test set (“real world”). We assume that if g(x) predicts well on the test set, then it will predict f(x) well.



Training and test sets

- Training sets
  - This is the data that you actually calculate your model parameters on
  - The Sum of Squared Errors is in-sample error
- Test set
  - No information from this is used to determine model parameters
  - The Sum of Squared Errors is out-of-sample error

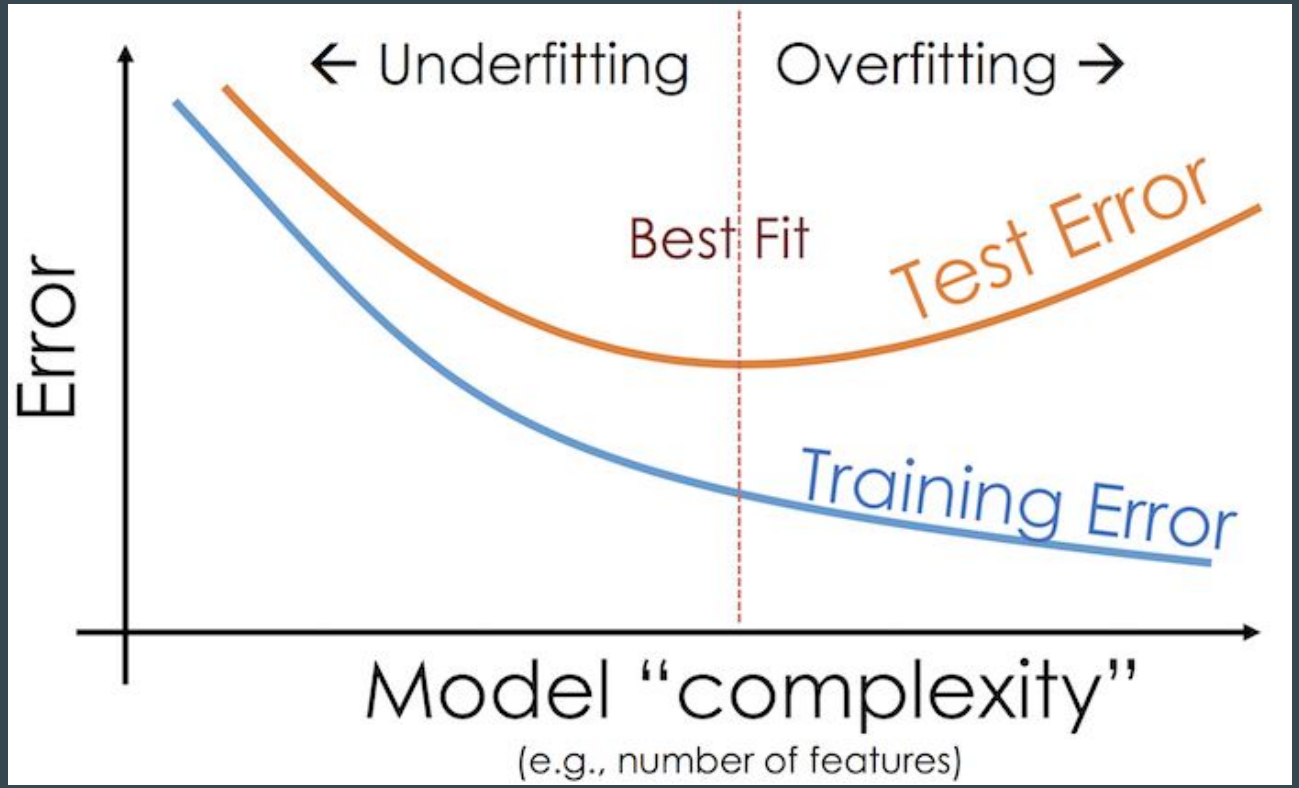
Machine Learning

- We have just described the process of machine learning
- It is called machine learning, because your computer “learns” from the data in the training set
- But it is evaluated by its performance on unobserved data, or the test set

Calculating out-of-sample error

1. Take your sample and split it into two sets at random
2. Find the OLS fit on the training set
    a. Calculate β and α, which gives you the parameters for your estimator
3. Using the estimator you just generated, calculate $\hat{Y}$ for your *TEST set*
4. Find the sum of squared error, or mean of squared errors with the new $\hat{Y}$ you collected

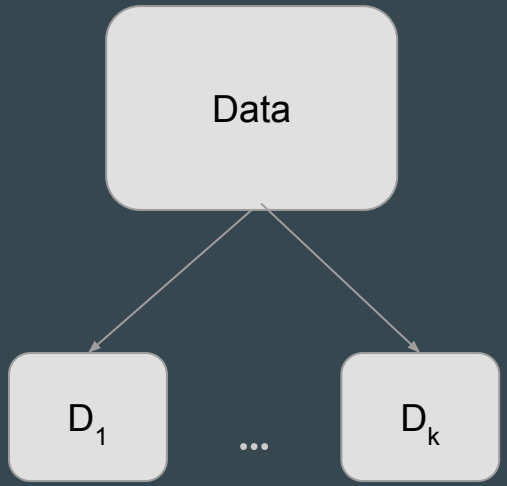
Training and test error diverge



Problem with training-test split

- Large datasets are required to learn complex relationships
- Training-test split requires lots of data
- Too small of a sample size will prevent us from finding the best function on our training set

Solution: *k*-fold cross-validation



- Split data into k samples
- *Test*: one sub-sample
- *Training*: k-1 sub-samples
- Pool model estimates for each test set

Example: three fold cross-validation

1. First, split your data up into three sets
2. Then use all possible combinations (3) of those three sets to split into training and test
3. Take the average of your error on all test sets

Iteration	Training	Test
1	1 and 2	3
2	1 and 3	2
3	2 and 3	1

Review

- We learned that out-of-sample error, or how well or model performs on unobserved data is necessary to prevent overfitting
- To truly calculate out-of-sample error, we would need to repeat our process multiple times
- Instead, we fake this by splitting our data into training and test sets