S&DS 265 / 565

Introductory Machine Learning

Bias-Variance Tradeoff and Cross Validation

September 29

Yale

Outline

- Bias/variance (redux)
- Cross validation
- Leave-one-out CV

Tutorial: `https://medium.com/@catriona_52586/loocv-for-evaluating-machine-learning-algorithms-a979cec82cc9`

Bias and variance

Bias: $\theta - \mathbb{E}\widehat{\theta}$

Variance: $\mathbb{E}(\widehat{\theta} - \mathbb{E}\widehat{\theta})^2$

Bias and variance

Bias: $\theta - \mathbb{E}\widehat{\theta}$

Variance: $\mathbb{E}(\widehat{\theta} - \mathbb{E}\widehat{\theta})^2$

- $\widehat{\theta}$ is an estimate from a sample
- $\mathbb{E}$ is the expectation (average) with respect to the sample
- So $\mathbb{E}\widehat{\theta}$ is the average estimate
- We can only directly compute $\widehat{\theta}$ for the sample we have
- *We don't know* θ

Bias and variance

Bias and variance are two sides of the same coin: As squared bias goes up, variance goes down

Bias-variance tradeoff

$$\text{Risk} = \text{Bias}^2 + \text{Variance}$$

Bias-variance tradeoff

$$\mathbb{E}(\theta - \widehat{\theta})^2 = \text{Bias}(\widehat{\theta})^2 + \text{Variance}(\theta)$$

Bias-variance tradeoff

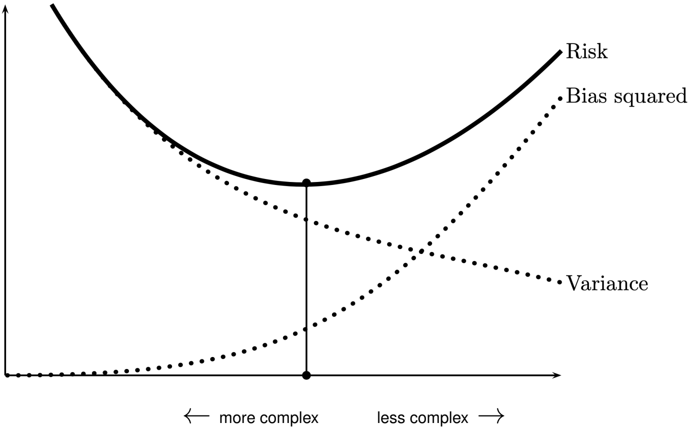
$$\mathbb{E}(\theta - \widehat{\theta})^2 = (\theta - \mathbb{E}\widehat{\theta})^2 + \mathbb{E}(\widehat{\theta} - \mathbb{E}\widehat{\theta})^2$$

Bias-variance tradeoff

Proof:

$$
\begin{aligned}
\mathbb{E}(\theta-\widehat{\theta})^2 &= \mathbb{E}(\theta-\mathbb{E}\widehat{\theta}+\mathbb{E}\widehat{\theta}-\widehat{\theta})^2 \\
&= \mathbb{E}(\theta-\mathbb{E}\widehat{\theta})^2 - 2\mathbb{E}\left\{(\theta-\mathbb{E}\widehat{\theta})(\widehat{\theta}-\mathbb{E}\widehat{\theta})\right\} + \mathbb{E}(\widehat{\theta}-\mathbb{E}\widehat{\theta})^2 \\
&= \mathbb{E}(\theta-\mathbb{E}\widehat{\theta})^2 - 2(\theta-\mathbb{E}\widehat{\theta})\mathbb{E}(\widehat{\theta}-\mathbb{E}\widehat{\theta}) + \mathbb{E}(\widehat{\theta}-\mathbb{E}\widehat{\theta})^2 \\
&= \mathbb{E}(\theta-\mathbb{E}\widehat{\theta})^2 + \mathbb{E}(\widehat{\theta}-\mathbb{E}\widehat{\theta})^2 \\
&= \text{Bias}(\widehat{\theta})^2 + \text{Variance}(\widehat{\theta})
\end{aligned}
$$

Bias-variance tradeoff



Example: Regularization

Suppose that $\mathbb{E}(Y) = \theta^*$ and we estimate

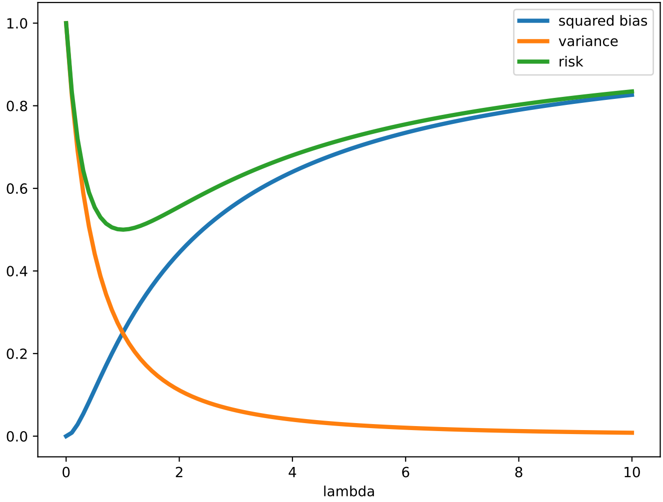
$$\widehat{\theta} = \arg\min_{\theta} (Y - \theta)^2 + \lambda\theta^2$$

Then $\widehat{\theta} = \frac{Y}{1+\lambda}$. What are the squared bias and variance?

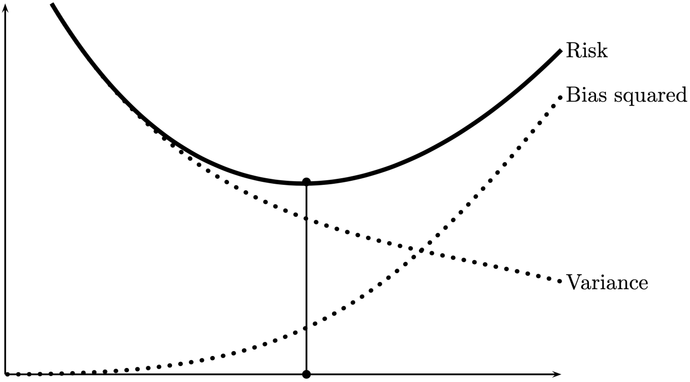
$$\text{Bias}^2 = \theta^{*2} \left(\frac{\lambda}{1+\lambda}\right)^2$$

$$\text{Variance} = \left(\frac{1}{1+\lambda}\right)^2 \text{Variance}(Y)$$

Example: Regularization



Bias-variance tradeoff



$\longleftarrow$ less regularization more regularization $\longrightarrow$

$\longleftarrow$ less shrinkage more shrinkage $\longrightarrow$

$\longleftarrow$ smaller penalty bigger penalty $\longrightarrow$

Next Topic: Model selection

For purposes of prediction, **minimizing test error** is priority.

Recall our two error metrics for evaluating predictions $\widehat{f}(x_i)$:

- Regression:
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{f}(x_i))^2$$
- Classification:
$$Err = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\left\{\widehat{f}(x_i) \neq y_i\right\}$$

Bias-Variance Tradeoff: Regression case

Given $Y = f(X) + \varepsilon$, where $\mathbb{E}(\varepsilon) = 0$ and $Var(\varepsilon) = \sigma^2$, consider a predictor $\widehat{f}$.

Expected MSE for predicting a new Y at $X = x$ can be decomposed into:

$$\mathbb{E}[(Y - \widehat{f}(x))^2] = Var(\widehat{f}(x)) + [Bias(\widehat{f}(x))]^2 + \sigma^2$$

Bias-Variance Tradeoff: Regression case

$$\mathbb{E}[(Y - \widehat{f}(x))^2] = Var(\widehat{f}(x)) + [Bias(\widehat{f}(x))]^2 + \sigma^2$$

- $Var(\widehat{f})$ is the amount of variability in our predictor with different training set.
- $Bias(\widehat{f})$ is the systematic error introduced by model approximation.
- σ^2 is *irreducible error*, inherent in the error term ε.

Bias-Variance Tradeoff: Regression case

$$\mathbb{E}[(Y - \widehat{f}(x))^2] = Var(\widehat{f}(x)) + [Bias(\widehat{f}(x))]^2 + \sigma^2$$

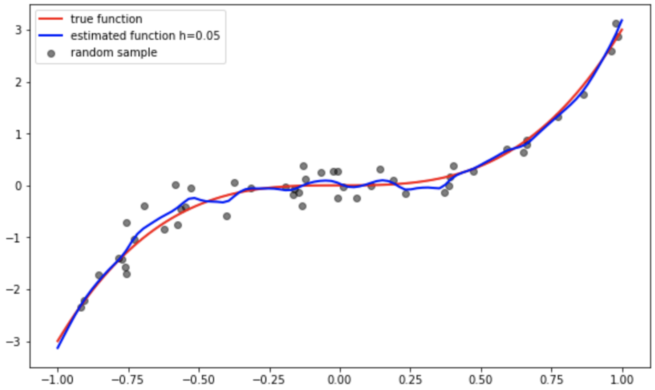
- $Var(\widehat{f})$ is the amount of variability in our predictor with different training set. Increases with increasing model flexibility.
- $Bias(\widehat{f})$ is the systematic error introduced by model approximation. Decreases with increasing model flexibility.
- σ^2 is *irreducible error*, inherent in the error term ε. Cannot get rid of this!

Need to balance bias and variance.

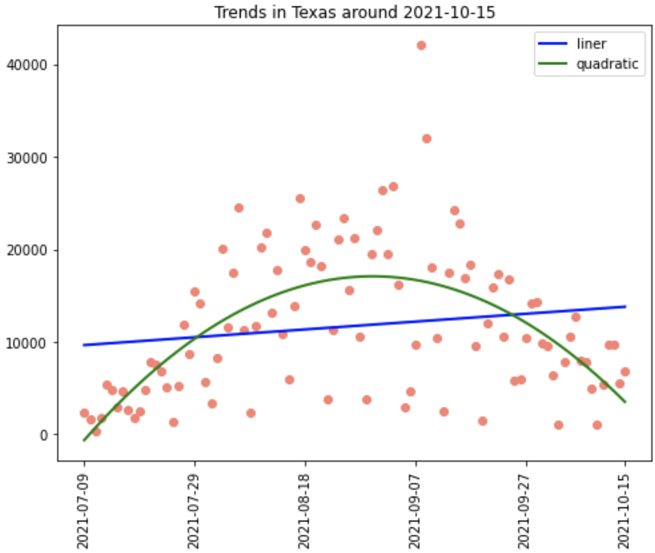
Classification

- For classification, we replace mean squared error by probability of making a mistake.
- There is no direct decomposition of misclassification error into (squared) bias and variance
- *But the situation is conceptually the same*
- We can break down the error into *approximation error* (like squared bias) and *estimation error* (like variance)
- Approximation error large if classifier is too simple
- Estimation error large if training classifier on too little data

Let’s go to the first notebook



Let’s go to the second notebook



Cross-Validation

Cross-validation is an intuitive, widely-applicable approach for:

- model assessment
- model selection

Example: Election Forecasting

The Economist

Today
Weekly edition
Menu

Forecasting the US elections

The Economist is analysing polling, economic and demographic data to predict America's elections in 2020

→ **Read more of our election coverage**

President Senate House

National forecast
How this works

COMPETITIVE STATES

- Arizona
- Florida
- Georgia
- Iowa
- Michigan
- Nevada
- New Hampshire
- North Carolina
- Ohio
- Pennsylvania
- Texas
- Wisconsin

ALL STATES

- Alabama

How The Economist presidential forecast works

THIS YEAR, *The Economist* is publishing its first-ever statistical forecast of an American presidential election. Developed with the assistance of Andrew Gelman and Merlin Heidemanns, political scientists at Columbia University, our model calculates Joe Biden's and Donald Trump's probabilities of winning each individual state and the election overall. Its projections will be updated every day at https://projects.economist.com/us-2020-forecast/president.

In another first, we are publishing the source code for what we believe to be the most innovative section of the model. All readers are welcome to download it, explore how it works, tweak its parameters and run it

Example: Election Forecasting

On the regular

A common criticism of fundamentals models is that they are extremely easy to "over-fit"—the statistical term for deriving equations that provide a close match to historical data, but break down when used to predict the future. To avoid this risk, we borrow two techniques from the world of machine learning, with appropriately inscrutable names: "elastic-net regularisation" and "leave-one-out cross-validation".

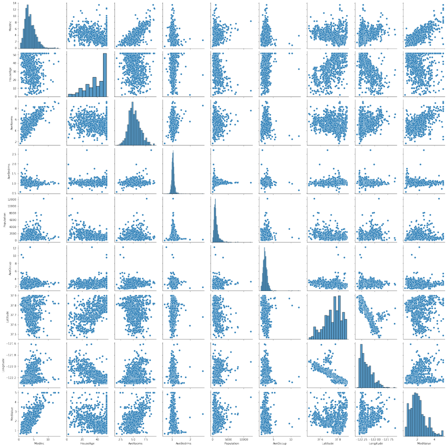
Example: Election Forecasting

Elastic-net regularisation is a method of reducing the complexity of a model. In general, equations that are simpler—or more “parsimonious”, in statisticians’ lingo—tend to do a better job of predicting unseen data than convoluted ones do. “Regularisation” makes models less complicated, either by shrinking the impact of the variables used as predictors, or by removing weak ones entirely.

Example: Election Forecasting

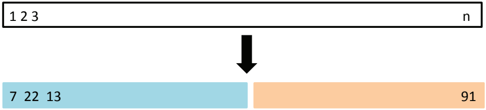
Next, in order to determine how much of this “shrinkage” to use, we deploy “leave-one-out cross-validation”. This technique involves chopping up a dataset into lots of pieces, training models on some chunks, and testing their performance on others. In this case, each chunk is one election year.

Example: California Housing



Validation Sets

We've been doing this:



1. Divide dataset randomly into a training set and a validation set.
2. Fit the model on the training set.
3. Use the validation set to obtain estimated test error.
4. Repeat!

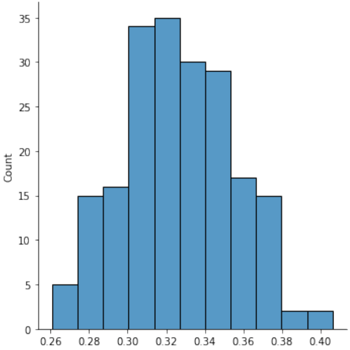
Validation Sets

Example:

$$\widehat{MedValue} = \widehat{\beta}_0 + \widehat{\beta}_1 MedInc$$

Histogram of errors

Highly variable



Validation Sets

- highly variable validation error
- only uses a fraction of the training set

Leave-One-Out Cross-Validation

How do we use more data to train with?

- Use a tiny validation set (e.g. (x_1, y_1))
- Train with the rest (e.g. $\{(x_2, y_2), \ldots, (x_n, y_n)\}$)

We're only evaluating the error using a single observation

Leave-One-Out Cross-Validation

How do we use more data to train with?

- Use a tiny validation set (e.g. (x_1, y_1))
- Train with the rest (e.g. $\{(x_2, y_2), \ldots, (x_n, y_n)\}$)

We're only evaluating the error using a single observation

But we can iterate through the dataset, each time using a different (x_i, y_i) as the validation set and obtaining an error MSE_i.

Leave-One-Out Cross-Validation

	Iteration					
Obs	1	2	3	4	...	n
1	valid	train	train	train	...	train
2	train	valid	train	train	...	train
3	train	train	valid	train	...	train
4	train	train	train	valid	...	train
...	...	...	...	...	...	...
n	train	train	...	...	...	valid
MSE	MSE_1	MSE_2	MSE_3	MSE_4	...	MSE_n

Leave-One-Out Cross-Validation

LOOCV estimate of test error is given by:

$$CV_{(n)} = \frac{1}{n} \sum_{i} MSE_i$$

A single number, no randomness.

k-fold Cross-Validation

A potentially faster approach:

- Randomly divide the dataset into k *folds*.
- For $b = 1, \ldots, k$:
  - Use b-th fold ("batch") as validation set.
  - Use everything else as training set.
  - Compute validation error on b-th fold.
- Estimate test error using:

$$CV_{(k)} = \sum_b \frac{n_b}{n} MSE_b,$$

where n_b is the total # observations in the b-th fold, and n is the total # observations in the entire dataset.

k-fold Cross-Validation

	Iteration						
Obs	1	2	3	4	...	k	
1	valid	train	train	train	...	train	
2	valid	train	train	train	...	train	fold 1
3	valid	train	train	train	...	train	
4	train	valid	train	train	...	train	
...	...	...	...	...	...	...	
$n-2$	train	train	...	...	...	valid	
$n-1$	train	train	...	...	...	valid	fold k
n	train	train	...	...	...	valid	
MSE	MSE_1	MSE_2	MSE_3	MSE_4	...	MSE_k	

k-fold Cross-Validation

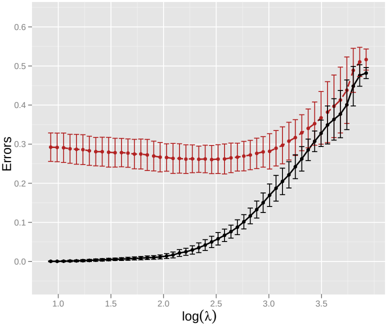
	Iteration					
Obs	1	2	3	4	...	k
1	valid	train	train	train	...	train
2	valid	train	train	train	...	train
3	valid	train	train	train	...	train
4	train	valid	train	train	...	train
...	...	...	...	...	...	...
$n-2$	train	train	...	...	...	valid
$n-1$	train	train	...	...	...	valid
n	train	train	...	...	...	valid
MSE	MSE_1	MSE_2	MSE_3	MSE_4	...	MSE_k

fold 1

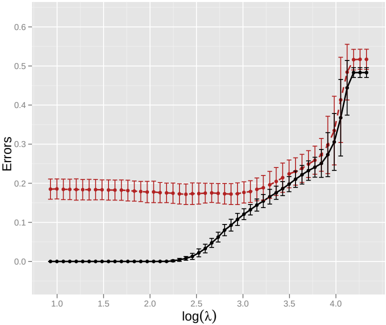
fold k

n-fold CV is just LOOCV.

Recall: Political blog classification results



without links



with links

A Slick Shortcut

Suppose the fitted values can be written $\widehat{Y} = HY$ where H is an $n \times n$ matrix.

A Slick Shortcut

Suppose the fitted values can be written $\widehat{Y} = HY$ where H is an $n \times n$ matrix.

This is the case for least squares and regularized regression

A Slick Shortcut

Suppose the fitted values can be written $\widehat{Y} = HY$ where H is an $n \times n$ matrix.

This is the case for least squares and regularized regression

Then the leave-one-out-cross-validation error is

$$R_{LOOCV} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(-i)})^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \widehat{Y}_i}{1 - H_{ii}}\right)^2$$

where H_{ii} is the ith diagonal entry.

A Slick Shortcut

Suppose the fitted values can be written $\widehat{Y} = HY$ where H is an $n \times n$ matrix.

This is the case for least squares and regularized regression

Then the leave-one-out-cross-validation error is

$$
\begin{aligned}
R_{LOOCV} &= \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(-i)})^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \widehat{Y}_i}{1 - H_{ii}}\right)^2
\end{aligned}
$$

where H_{ii} is the ith diagonal entry.

So, no need to fit n regressions!

Let's go to the notebook

Open up the notebook `california-housing.ipynb` and follow along...

Summary

- Cross validation is a practical way of estimating the variability of test error. Used for model selection.
- Leave-one-out CV is the most important version of CV. Has a shortcut formula.