

Modeling Basics: Assessment, Selection, and Complexity

Statistical Machine Learning, Spring 2015

Ryan Tibshirani (with Larry Wasserman)

1 You (should) already know this stuff: statistical prediction and the bias-variance tradeoff

- Suppose that we observe (X, Y) from some unknown joint distribution, and we want to predict Y from X . Over all functions f , the expected prediction error, measured in terms of squared loss,

$$\mathbb{E}[(Y - f(X))^2]$$

is minimized at $f(x) = \mathbb{E}(Y|X = x)$. This is called the true regression function associated with the pair (X, Y) . Note that we can always write

$$Y = f(X) + \varepsilon,$$

for some noise term satisfying $\mathbb{E}(\varepsilon) = 0$. Why? Just take $\varepsilon = Y - f(X)$.

- Now suppose that we observe training samples $(x_1, y_1), \dots, (x_n, y_n)$ i.i.d. from the same joint distribution as (X, Y) . We use these training samples to construct a prediction function \hat{f} ; this mimics $f(x) = \mathbb{E}(Y|X = x)$, which is optimal but unknowable in practice. The expected prediction error, or expected test error, of \hat{f} is

$$\mathbb{E}[(Y - \hat{f}(X))^2], \tag{1}$$

where the expectation is over all that is random, namely, the training set (x_i, y_i) , $i = 1, \dots, n$ and the test point (X, Y)

- Why would we be interested in (1)? Here are two reasons:
 1. Model assessment: sometimes we want to know how well we can predict a future observation, in absolute terms
 2. Model selection: often we want to choose between different prediction functions; this might mean choosing between two different model classes entirely, or choosing an underlying tuning parameter for a single method. In principle, we could do this by comparing expected test errors
- The expected test error in (1) has an important property: it decomposes into informative parts. But before we show this, let's think about a few points:
 - Can we ever predict Y from X with zero error? Generally, no. Even the true regression function f cannot generically do this, and will incur some error due to noise. We call this *irreducible error*

- What happens if our fitted function \hat{f} belongs to a model class that is far from the true regression function f ? E.g., we choose to fit a linear model in a setting where the true relationship is far from linear? As a result, we encounter error, what we call *estimation bias*
- What happens if our fitted (random) function \hat{f} is itself quite variable? In other words, over different copies of the training set, we end up constructing substantially different functions \hat{f} ? This is another source of error, that we'll call *estimation variance*
- More so than just these intuitive descriptions, the expected test error mathematically decomposes into a sum of three corresponding parts. Begin by writing the model

$$Y = f(X) + \varepsilon,$$

where ε has mean zero, variance σ^2 , and is independent of X . Note that the independence condition is the an actual (nontrivial) assumption. Recall that (x_i, y_i) , $i = 1, \dots, n$ are independent of each other and of (X, Y) , all with the same distribution. We'll look at the expected test error, conditional on $X = x$ for some arbitrary input x . It follows that

$$\mathbb{E}[(Y - \hat{f}(x))^2 | X = x] = \sigma^2 + \underbrace{\mathbb{E}[(f(x) - \hat{f}(x))^2]}_{\text{Risk}(\hat{f}(x))}.$$

The first term σ^2 is the *irreducible error*, or sometimes referred to as the *Bayes error*, and the second term is called the risk, or mean squared error (MSE). The risk further decomposes into two parts, so that

$$\mathbb{E}[(Y - \hat{f}(x))^2 | X = x] = \sigma^2 + \underbrace{(f(x) - \mathbb{E}[\hat{f}(x)])^2}_{\text{Bias}^2(\hat{f}(x))} + \underbrace{\mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]}_{\text{Var}(\hat{f}(x))}, \quad (2)$$

the latter terms being the squared *estimation bias* or simply *bias*, and the *estimation variance* or simply *variance*, respectively. The decomposition (2) is called the *bias-variance decomposition* or *bias-variance tradeoff*

- From the bias-variance tradeoff (2), we can see that even if our prediction is unbiased, i.e., $\mathbb{E}[\hat{f}(x)] = f(x)$, we can still incur a large error if it is highly variable. Meanwhile, even when our prediction is stable and not variable, we can incur a large error if it is badly biased. There is a tradeoff here, but it need it is really never be one-to-one; e.g., in some cases, it can be worth sacrificing a little bit of bias to gain large decrease in variance
- Typical trend: underfitting means high bias and low variance, overfitting means low bias but high variance. E.g., think about k in k -nearest-neighbors regression: recall that, trained on (x_i, y_i) , $i = 1, \dots, n$, the k -nearest-neighbors regression fit at an input point x is

$$\hat{f}(x) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i, \quad (3)$$

where $\mathcal{N}_k(x)$ denotes the indices of the k training inputs closest to x . How do the bias and variance behave for small k , and for large k ?

Let's consider the input training points x_1, \dots, x_n all fixed (nonrandom) for simplicity. Then for the k -nearest-neighbors fit \hat{f} ,

$$\mathbb{E}[(Y - \hat{f}(X))^2 | X = x] = \sigma^2 + \left(f(x) - \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} f(x_i)\right)^2 + \frac{\sigma^2}{k}.$$

Hence we can explicitly see that as k increases, the variance decreases, and the bias (likely) increases. Note that, for fixed x_1, \dots, x_n , the k -nearest-neighbors fit $\hat{f}(x)$ in (3), at any input x , is just a linear function of y_1, \dots, y_n . This is an example of what is called a linear smoother, which we'll discuss more later in Section 5

- Finally, unconditionally over X , the bias-variance decomposition reads

$$\mathbb{E}[(Y - \hat{f}(X))^2] = \sigma^2 + \int \text{Bias}^2(\hat{f}(x)) P_X(dx) + \int \text{Var}(\hat{f}(x)) P_X(dx),$$

where P_X is the distribution of X (this follows from taking an expectation over X in (2)). In words, the expected test error is the Bayes error + average (squared) bias + average variance

2 Training error and optimism

- Now that we have convinced ourselves that the expected test error (1) is a useful quantity, how do we estimate it? Clearly if we had a test set $(x'_1, y'_1), \dots, (x'_m, y'_m)$, independent of the training set and from the same distribution, then the observed average test error

$$\frac{1}{m} \sum_{i=1}^m (y'_i - \hat{f}(x'_i))^2$$

would serve as an unbiased estimate for $\mathbb{E}[(Y - \hat{f}(X))^2]$. But we are often not this fortunate to have test data

- What's wrong with the average training error

$$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{f}(x_i))^2?$$

This measures the squared error of \hat{f} around the data we used to fit it—the problem is that this will be, generally speaking, far too optimistic. Why? Because we chose \hat{f} so that it fit well to the training data (x_i, y_i) , $i = 1, \dots, n$, of course its error is going to look too small

- This rules out using the training error for the first purpose described above, model assessment. But if the expected training error was systematically (always) different than the expected test error by the same amount, then we could use it for the second purpose, model selection. E.g., if the training error curve had the right shape with k (but just not the right height), then we could still use it to choose k in k -nearest-neighbors regression. Unfortunately, this is essentially never the case; in almost all real examples, the training error curve has both the wrong magnitude and the wrong shape. (What is the 1-nearest-neighbor training error?)
- The expected test and training errors can be linked in a useful way, if we again assume that fixed inputs, for simplicity. That is, let $(x_1, y'_1), \dots, (x_n, y'_n)$ denote a test set, independent of and having the same distribution as $(x_1, y_1), \dots, (x_n, y_n)$. The inputs x_1, \dots, x_n are fixed in both sets, and only the observations y_i, y'_i , $i = 1, \dots, n$ are random, drawn according to

$$\begin{aligned} y_i &= f(x_i) + \epsilon_i, & i &= 1, \dots, n, \\ y'_i &= f(x_i) + \epsilon'_i, & i &= 1, \dots, n, \end{aligned}$$

all errors ϵ_i, ϵ'_i , $i = 1, \dots, n$ i.i.d. with mean zero and variance σ^2

In vector notation, write $y, y' \in \mathbb{R}^n$ for the training and test observations, and write $\hat{f} = (\hat{f}(x_1), \dots, \hat{f}(x_n)) \in \mathbb{R}^n$ for the training fitted values. Then the expected test error is

$$\frac{1}{n} \mathbb{E} \|y' - \hat{f}\|_2^2,$$

where the expectation is taken over y, y' . Now write $f = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$, and decompose

$$\begin{aligned} \frac{1}{n} \mathbb{E} \|y' - \hat{f}\|_2^2 &= \frac{1}{n} \mathbb{E} \|y' - f + f - \hat{f}\|_2^2 \\ &= \sigma^2 + \frac{1}{n} \mathbb{E} \|f - \hat{f}\|_2^2 \\ &= 2\sigma^2 + \frac{1}{n} \mathbb{E} \|y - \hat{f}\|_2^2 + \frac{2}{n} \mathbb{E} (f - y)^T (y - \hat{f}) \\ &= \frac{1}{n} \mathbb{E} \|y - \hat{f}\|_2^2 + \frac{2}{n} \text{tr}(\text{Cov}(y, \hat{f})). \end{aligned}$$

In other words, the *optimism*—difference in expected test and expected training errors—can be expressed as

$$\frac{1}{n} \mathbb{E} \|y' - \hat{f}\|_2^2 - \frac{1}{n} \mathbb{E} \|y - \hat{f}\|_2^2 = \frac{2}{n} \text{tr}(\text{Cov}(y, \hat{f})) = \frac{2}{n} \sum_{i=1}^n \text{Cov}(y_i, \hat{f}(x_i)). \quad (4)$$

So the higher the correlation between y_i and its fitted value $\hat{f}(x_i)$, the greater the optimism. We'll see that in many cases we can either evaluate or estimate the right-hand side above, which can be extremely useful

3 Degrees of freedom and unbiased risk estimation

- For $\hat{f} = (\hat{f}(x_1), \dots, \hat{f}(x_n)) \in \mathbb{R}^n$ and the same setup as before—that is, x_1, \dots, x_n considered fixed, and \hat{f} trained on $(x_1, y_1), \dots, (x_n, y_n)$ —the *degrees of freedom* of \hat{f} is defined as

$$\text{df}(\hat{f}) = \frac{1}{\sigma^2} \sum_{i=1}^n \text{Cov}(y_i, \hat{f}(x_i)). \quad (5)$$

Hence the optimism in (4) can be more succinctly written as

$$\frac{1}{n} \mathbb{E} \|y' - \hat{f}\|_2^2 - \frac{1}{n} \mathbb{E} \|y - \hat{f}\|_2^2 = \frac{2\sigma^2}{n} \text{df}(\hat{f}). \quad (6)$$

Intuitively, we can think of $\text{df}(\hat{f})$ as the effective number of parameters used by the fit \hat{f} , a measure of complexity of \hat{f} . A few simple examples that support this intuition:

- If $\hat{f} = (y_1, \dots, y_n)$, then $\text{df}(\hat{f}) = n$
- If $\hat{f}(x_i) = \bar{y} = \frac{1}{n} \sum_{j=1}^n y_j$ for each $i = 1, \dots, n$, then $\text{df}(\hat{f}) = 1$
- If $\hat{f}(x_i) = \frac{1}{k} \sum_{j \in \mathcal{N}_k(x)} y_j$ for each $i = 1, \dots, n$, then $\text{df}(\hat{f}) = n/k$
- Suppose that we can compute the degrees of freedom of \hat{f} , or even more broadly, compute an unbiased estimate $\widehat{\text{df}}(\hat{f})$, with the property

$$\mathbb{E}[\widehat{\text{df}}(\hat{f})] = \text{df}(\hat{f}).$$

Then following from (6), the quantity

$$\hat{T} = \frac{1}{n} \|y - \hat{f}\|_2^2 + \frac{2\sigma^2}{n} \widehat{\text{df}}(\hat{f})$$

serves as unbiased estimate for the expected test error of \hat{f} , i.e.,

$$\mathbb{E}(\hat{T}) = \frac{1}{n} \mathbb{E} \|y' - \hat{f}\|_2^2$$

- Often times you will encounter this principle referred to in terms of unbiased risk estimation, since

$$\hat{R} = \hat{T} - \sigma^2 = \frac{1}{n} \|y - \hat{f}\|_2^2 + \frac{2\sigma^2}{n} \widehat{\text{df}}(\hat{f}) - \sigma^2$$

serves as an unbiased estimate for the risk,

$$\mathbb{E}(\hat{R}) = \frac{1}{n} \mathbb{E} \|f - \hat{f}\|_2^2$$

We'll see how to explicitly compute degrees of freedom for linear smoothers, in Section 5. This is a suprisingly broad class. Later in the course, we'll see how to use Stein's formula to compute unbiased estimates of degrees of freedom in an even more general setting, too

- Note: all of the above assumes that σ^2 is known, which is not realistic. In practice, we'd have to estimate it
- An example of how to use this for model selection: suppose that we were considering fitted models \hat{f}_θ indexed by a parameter $\theta \in \Theta$; e.g., this could represent the number of neighbors k in k -nearest-neighbors regression. Suppose also that had access to $\widehat{\text{df}}(\hat{f}_\theta)$ at each value of θ . Then we could choose the parameter by minimizing our test error estimate,

$$\hat{\theta} = \underset{\theta \in \Theta}{\text{argmin}} \frac{1}{n} \|y - \hat{f}_\theta\|_2^2 + \frac{2\sigma^2}{n} \widehat{\text{df}}(\hat{f}_\theta).$$

An important point to be aware of: while $\hat{T}(\hat{f}_\theta) = \frac{1}{n} \|y - \hat{f}_\theta\|_2^2 + \frac{2\sigma^2}{n} \widehat{\text{df}}(\hat{f}_\theta)$ gives an unbiased estimate for the expected test error at any fixed θ , we should anticipate that $\hat{T}(\hat{f}_{\hat{\theta}})$ is itself downward biased (optimistic) as an estimate for the expected test error at $\hat{\theta}$. Why?

4 Cross-validation

- Cross-validation (CV) is quite a general tool for estimating the expected test error (1), that makes minimal assumptions—i.e., it doesn't assume that $Y = f(X) + \varepsilon$ with ε independent of X , it doesn't assume that the training inputs x_1, \dots, x_n are fixed, all it really assumes is that the training samples $(x_1, y_1), \dots, (x_n, y_n)$ are i.i.d.

We split up our training set into K divisions or folds, for some number K ; usually this is done randomly. Write these as F_1, \dots, F_K , so $F_1 \cup \dots \cup F_K = \{1, \dots, n\}$. Now for each $k = 1, \dots, K$, we fit our prediction function on all points but those in the k th fold, denoted $\hat{f}^{-(k)}$, and evaluate squared errors on the points in the k th fold,

$$\text{CV}_k(\hat{f}^{-(k)}) = \frac{1}{n_k} \sum_{i \in F_k} (y_i - \hat{f}^{-(k)}(x_i))^2.$$

Here n_k denotes the number of points in the k th fold, $n_k = |F_k|$. We average these fold-based errors to yield an estimate of the expected test error,

$$\text{CV}(\hat{f}) = \frac{1}{K} \sum_{k=1}^K \text{CV}_k(\hat{f}^{-(k)}). \quad (7)$$

This is called K -fold cross-validation; the special case when $K = n$ is referred to as leave-one-out cross-validation

- What is the difference between choosing say $K = 5$ (a common choice) versus $K = n$?
 - When $K = 5$, the function $\hat{f}^{-(k)}$ in each fold k is fit on about $4/5 \cdot n$ samples, and so we are looking at the errors incurred by a procedure that is trained on less data than the full \hat{f} in (1). Therefore the mean of the CV estimate (7) could be off. When $K = n$, this is not really an issue, since each $\hat{f}^{-(k)}$ is trained on $n - 1$ samples
 - When $K = n$, the CV estimate (7) is an average of n extremely correlated quantities; this is because each $\hat{f}^{-(k)}$ and $\hat{f}^{-(\ell)}$ are fit on $n - 2$ common training points. Hence the CV estimate will likely have very high variance. When $K = 5$, the CV estimate will have lower variance, since it is the average of quantities that are less correlated, as the fits $\hat{f}^{-(k)}$, $k = 1, \dots, 5$ do not share as much overlapping training data

This is tradeoff (the bias-variance tradeoff, in fact!). Usually, a choice like $K = 5$ or $K = 10$ is more common in practice than $K = n$, but this is probably an issue of debate

- For K -fold CV, it's can be helpful to assign a notion of variability to the CV error estimate. We argue that

$$\text{Var}(\text{CV}(\hat{f})) = \text{Var}\left(\frac{1}{K} \sum_{k=1}^K \text{CV}_k(\hat{f}^{-(k)})\right) \approx \frac{1}{K} \text{Var}(\text{CV}_1(\hat{f}^{-(1)})). \quad (8)$$

Why is this an approximation? This would hold exactly if $\text{CV}_1(\hat{f}^{-(1)}), \dots, \text{CV}_K(\hat{f}^{-(K)})$ were i.i.d., but they're not. This approximation is valid for small K (e.g., $K = 5$ or 10) but not really for big K (e.g., $K = n$), because then the quantities $\text{CV}_1(\hat{f}^{-(1)}), \dots, \text{CV}_K(\hat{f}^{-(K)})$ are highly correlated

- For small K (e.g., $K = 5$ or 10), we can leverage (8) to get an estimate of the variance of the CV error estimate. We just use the sample variance appropriately, so that

$$\frac{1}{K} \sum_{k=1}^K \left(\text{CV}_k(\hat{f}^{-(k)}) - \text{CV}(\hat{f}) \right)^2$$

is our estimate of the variance of $\text{CV}(\hat{f})$

- We can use this variance estimate to draw approximate standard deviation bands around a CV error curve, and this leads to model selection heuristics like the *one-standard-error rule*. That is, the usual rule for selecting a parameter θ in a family of fitted models \hat{f}_θ , $\theta \in \Theta$ would be to minimize the CV error:

$$\hat{\theta} = \underset{\theta \in \Theta}{\text{argmin}} \text{CV}(\hat{f}_\theta).$$

The one-standard-error rule, instead, chooses the simplest model that is within one standard error (i.e., one standard deviation) of the minimum CV error. How to measure simplicity? Well, degrees of freedom is one way to do so! Often, the parametrization \hat{f}_θ will be such that $\text{df}(\hat{f}_\theta)$ behaves monotonically with θ ; e.g., in k -nearest-neighbors, the degrees of freedom

increases as k increases (in fact, it equals k). When $\text{df}(\hat{f}_\theta)$ is monotone increasing with θ , the one-standard-error rule can be written as

$$\tilde{\theta} = \min \left\{ \theta \in \Theta : \text{CV}(\hat{f}_\theta) \leq \text{CV}(\hat{f}_{\tilde{\theta}}) + \text{SE}(\hat{f}_{\tilde{\theta}}) \right\},$$

where $\text{SE}(\hat{f}_\theta)$ denotes our estimate of the standard deviation of $\text{CV}(\hat{f}_\theta)$, as explained above. Just to be perfectly clear, this is

$$\text{SE}(\hat{f}_\theta) = \frac{1}{\sqrt{K}} \left[\sum_{k=1}^K \left(\text{CV}_k(\hat{f}_\theta^{(k)}) - \text{CV}(\hat{f}_\theta) \right)^2 \right]^{1/2}$$

5 Linear smoothers

- Consider a prediction function \hat{f} trained on (x_i, y_i) , $i = 1, \dots, n$, that can be written as

$$\hat{f}(x) = \sum_{i=1}^n w_i(x) y_i = w(x)^T y.$$

Here the weights $w_i(x)$, $i = 1, \dots, n$ are allowed to depend on the query point x and x_1, \dots, x_n . With x_1, \dots, x_n considered fixed, this just a linear function of y_1, \dots, y_n , and therefore such a function \hat{f} is called a *linear smoother*

- We saw that the k -nearest-neighbor regression fit is a linear smoother, but so are many other interesting examples. We'll see a lot more in the the nonparametric regression lectures, but e.g., both linear regression and ridge regression are examples of linear smoothers. In these cases, each $x_i \in \mathbb{R}^p$, $i = 1, \dots, n$, and we write $X \in \mathbb{R}^{n \times p}$ as the matrix whose i th row is x_i . Then for linear regression of y on X ,

$$\hat{f}(x) = x^T \hat{\beta} = x^T (X^T X)^{-1} X^T y,$$

and for ridge regression of y on X , with tuning parameter $\lambda \geq 0$,

$$\hat{f}(x) = x^T \hat{\beta}_\lambda = x^T (X^T X + \lambda I)^{-1} X^T y$$

- A linear smoother produces fitted values $\hat{f} = (\hat{f}(x_1), \dots, \hat{f}(x_n))$ of the form

$$\hat{f} = S y$$

for a matrix $S \in \mathbb{R}^{n \times n}$, whose i th row is $w_i(x)$. Just to emphasize, the matrix S can depend on x_1, \dots, x_n , and also on external quantities like a tuning parameter, but it cannot depend on $y = (y_1, \dots, y_n)$. E.g., for linear regression, $S = X(X^T X)^{-1} X^T$, and for ridge regression, $S = X(X^T X + \lambda I)^{-1} X^T$

- It turns out that a lot of specific things can be said about linear smoothers. The degrees of freedom of a linear smoother is easily computed:

$$\text{df}(\hat{f}) = \frac{1}{\sigma^2} \text{tr}(\text{Cov}(y, \hat{y})) = \text{tr}(S),$$

with $\text{tr}(S)$ denoting the trace of the smoother matrix S . For linear regression, this turns out to be $\text{tr}(X(X^T X)^{-1} X^T) = p$, and for ridge regression, this is $\text{tr}(X(X^T X + \lambda I)^{-1} X^T) < p$ when $\lambda > 0$. We'll return to linear smoothers when we cover nonparametric regression, where we'll see that they admit many other nice properties