12 Statistical Justifications; the Bias-Variance Decomposition

STATISTICAL JUSTIFICATIONS FOR REGRESSION

[So far, I've talked about regression as a way to fit curves to points. Recall that early in the semester I divided machine learning into 4 levels: the application, the model, the optimization problem, and the optimization algorithm. My last two lectures about regression were at the bottom two levels: optimization. The cost functions that we optimize seem somewhat arbitrary. Today, let's take a step up to the second level, the model. I will describe some models, how they lead to those optimization problems, and how they contribute to underfitting or overfitting.]

Typical model of reality:

- sample points come from unknown prob. distribution: $X_i \sim D$
- y-values are sum of unknown, non-random fn + random noise:

$$\forall X_i, \quad y_i = g(X_i) + \epsilon_i, \qquad \epsilon_i \sim D', \quad D' \text{ has mean zero}$$

[We are positing that reality is described by a function g. We don't know g, but g is not random; it represents a consistent relationship between X and y that we can estimate. We add to g a random variable ϵ, which represents measurement errors and all the other sources of statistical error when we measure real-world phenomena. Notice that the noise is independent of X. That's a pretty big assumption, and often it does not apply in practice, but that's all we'll have time to deal with this semester. Also notice that this model leaves out systematic errors, like when your measuring device adds one to every measurement, because we usually can't diagnose systematic errors from data alone.]

Goal of regression: find h that estimates g.
Ideal approach: choose $h(x) = \underbrace{\mathrm{E}_Y[Y|X = x]} = g(x) + \mathrm{E}[\epsilon] = g(x)$

[If this expectation exists at all, it partly justifies our model of reality. We can retroactively define g to be this expectation.]

Least-Squares Cost Function from Maximum Likelihood

Suppose $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$; then $y_i \sim \mathcal{N}(g(X_i), \sigma^2)$

Recall that log of normal PDF is

$$\ln f(y_i) = -\frac{(y_i - \mu_i)^2}{2\sigma^2} - \text{constant} \qquad \Leftarrow \mu_i = g(X_i)$$

& log likelihood is

$$\ell(g; X, y) = \ln\left(f(y_1)\, f(y_2) \cdots f(y_n)\right) = \ln f(y_1) + \ldots + \ln f(y_n) = -\frac{1}{2\sigma^2} \sum (y_i - g(X_i))^2 - \text{constant}$$

Takeaway: Max likelihood on "parameter" $g \Rightarrow$ estimate g by least-squares regression

[We treat g as a "distribution parameter." MLE tells us to choose a g that minimizes $\sum (y_i - g(X_i))^2$.]

[So if the noise is normally distributed, maximum likelihood justifies using the least-squares cost function.] [However, I've told you in previous lectures that least-squares is very sensitive to outliers. If the error is truly normally distributed, that's not a big deal, especially when you have a lot of sample points. But in the real world, the distribution of outliers often isn't normal. Outliers might come from wrongly measured measurements, data entry errors, anomalous events, or just not having a normal distribution. When you have a heavy-tailed distribution, for example, least-squares isn't a good choice.]

Empirical Risk

The risk for hypothesis h is expected loss $R(h) = \mathrm{E}[L]$ over all (X, Y) in some joint distribution.
Discriminative model: we don't know X's dist. D. How can we minimize risk?
[If we have a generative model, we can estimate the joint probability distribution for X and Y and derive the expected loss. That's what we did for Gaussian discriminant analysis. But today I'm assuming we don't have a generative model, so we don't know those probabilities. Instead, we approximate the distribution in a very crude way: we pretend that the sample points *are* the distribution.]

Empirical distribution: the **discrete** uniform distribution over the sample pts

Empirical risk: expected loss under empirical distribution

$$\hat{R}(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(X_i), y_i)$$

[The hat on the R indicates it's only a cheap approximation of the true, unknown statistical risk we really want to minimize. Often, this is the best we can do. For many but not all distributions, the empirical risk converges to the true risk in the limit as $n \to \infty$. Choosing h that minimizes $\hat{R}$ is called empirical risk minimization.]

Takeaway: this is why we [usually] minimize the sum of loss fns.

Logistic Loss from Maximum Likelihood

What cost fn should we use for probabilities?

Actual probability pt X_i is in the class is y_i; predicted prob. is $h(X_i)$.

Imagine β duplicate copies of X_i: $y_i\beta$ are in the class, $(1 - y_i)\beta$ are not.
[The size of β isn't very important, but imagine that $y_i\beta$ and $(1 - y_i)\beta$ are both integers for all i.]

[If we use maximum likelihood estimation to choose the hypothesis most likely to generate this sequence of sample points and labels, we get the following likelihood.]

Likelihood is $\mathcal{L}(h; X, y) = \prod_{i=1}^{n} h(X_i)^{y_i\beta}(1 - h(X_i))^{(1-y_i)\beta}$

$$\begin{aligned}
\text{Log likelihood is } \ell(h) &= \ln \mathcal{L}(h) \\
&= \beta \sum_i \Big(y_i \ln h(X_i) + (1 - y_i) \ln(1 - h(X_i)) \Big) \\
&= -\beta \sum \text{logistic loss fn } L(h(X_i), y_i).
\end{aligned}$$

Takeaway: Max likelihood ⇒ minimize $\sum$ logistic losses.
[So the principle of maximum likelihood explains where the weird logistic loss function comes from.]

THE BIAS-VARIANCE DECOMPOSITION

There are 2 sources of error in a hypothesis h:

bias: error due to inability of hypothesis h to fit g perfectly
e.g., fitting quadratic g with a linear h

variance: error due to fitting random noise in data
e.g., we fit linear g with a linear h, yet $h \neq g$.

Model: $X_i \sim D$, $\epsilon_i \sim D'$, $y_i = g(X_i) + \epsilon_i$ [remember that D' has mean zero]
fit hypothesis h to X, y
Now h is a random variable; i.e., its weights are random

Consider arbitrary pt $z \in \mathbb{R}^d$ (not necessarily a sample pt!) & $\gamma = g(z) + \epsilon$, $\epsilon \sim D'$
[So z is *arbitrary*, whereas γ is *random*.]
Note: $\mathrm{E}[\gamma] = g(z)$; $\mathrm{Var}(\gamma) = \mathrm{Var}(\epsilon)$ [the mean comes from g, and the variance comes from ϵ]

Risk fn when loss = squared error:

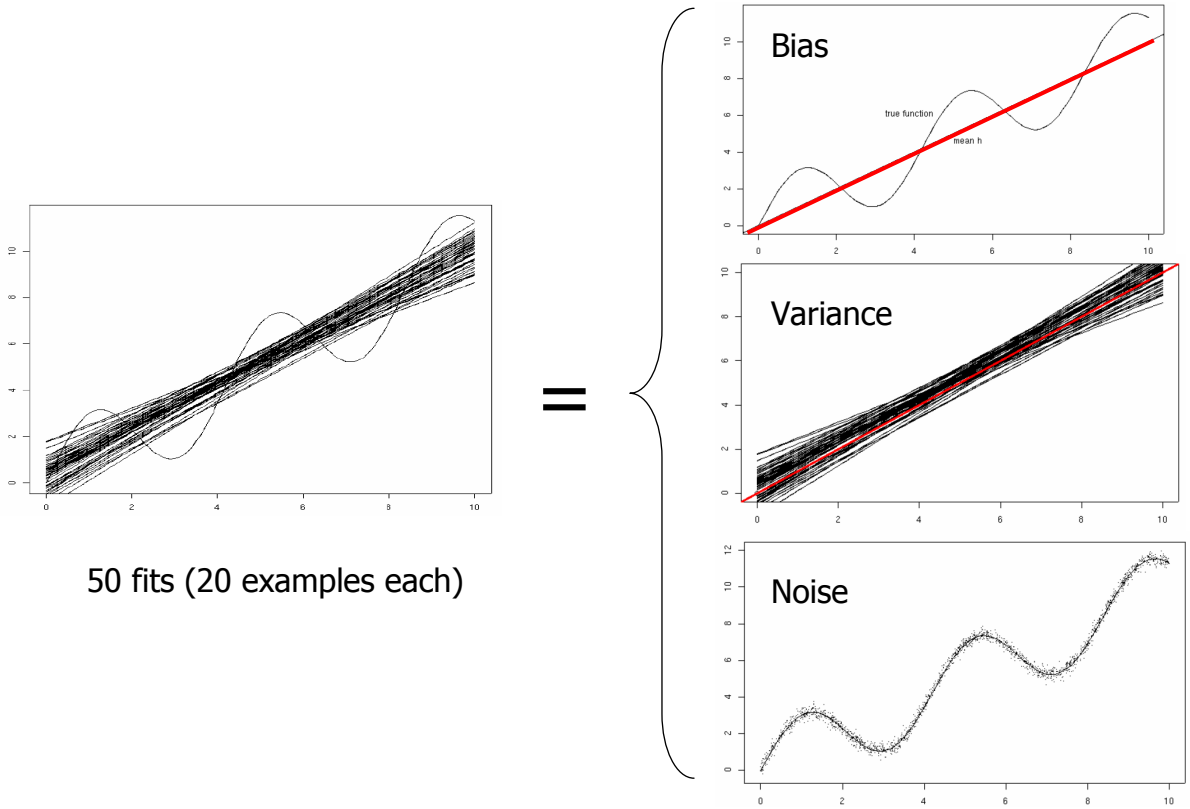
$$R(h) = \mathrm{E}[L(h(z), \gamma)]$$

↑ take expectation over possible training sets X, y & values of γ

[Stop and take a close look at this expectation. Remember that the hypothesis h is a random variable. We are taking a mean over the probability distribution of hypotheses. That seems pretty weird if you've never seen it before. But remember, the training data X and y come from a joint probability distribution. We use the training data to choose weights, so the weights that define h also come from some probability distribution. It might be hard to work out what that distribution is, but it exists. This "E[·]" is integrating the loss over all possible values of the weights.]

$$\begin{aligned}
&= \mathrm{E}[(h(z) - \gamma)^2] \\
&= \mathrm{E}[h(z)^2] + \mathrm{E}[\gamma^2] - 2\,\mathrm{E}[\gamma\, h(z)] \qquad \text{[Observe that } \gamma \text{ and } h(z) \text{ are independent]} \\
&= \mathrm{Var}(h(z)) + \mathrm{E}[h(z)]^2 + \mathrm{Var}(\gamma) + \mathrm{E}[\gamma]^2 - 2\mathrm{E}[\gamma]\,\mathrm{E}[h(z)] \\
&= (\mathrm{E}[h(z)] - \mathrm{E}[\gamma])^2 + \mathrm{Var}(h(z)) + \mathrm{Var}(\gamma) \\
&= \underbrace{(\mathrm{E}[h(z)] - g(z))^2}_{\text{bias}^2 \text{ of method}} + \underbrace{\mathrm{Var}(h(z))}_{\text{variance of method}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{irreducible error}}
\end{aligned}$$

[This is called the *bias-variance decomposition* of the risk function. Let's look at an intuitive interpretation of these three parts.]



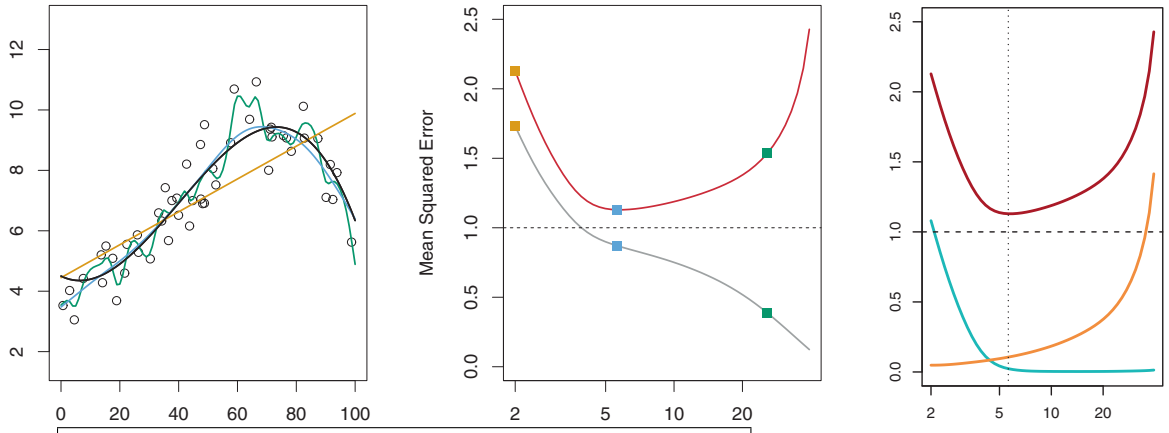
bvn.pdf [In this example, we're trying to fit a sine wave with lines, which obviously aren't going to be accurate. At left, we have generated 50 different hypotheses (lines). Each line was generated from 20 random training points by least-squares linear regression. At upper right, the red line is the *expected hypothesis*—an average over infinitely many hypotheses. The black curve illustrates test points on the true function g. We see that most test points have a large bias (difference between the black and red curves), because lines don't fit sine waves well. However, some of the test points happen to have a small bias—where the sine wave crosses the red line. At center right, the variance is the expected squared difference between a random black line and the red line (at a test point z). At lower right, the irreducible error is the expected squared difference between a random test point and the sine wave.]

This is pointwise version [of the bias-variance decomposition.]
Mean version: let $z \sim D$ be random variable; take mean over D of bias^2, variance.

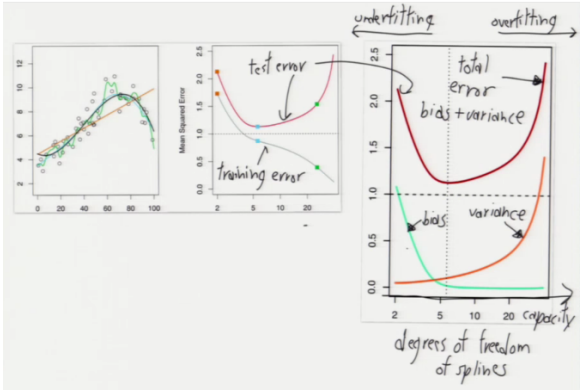
[So you can decompose one test point's error into these three numbers, or you can decompose the error of the hypothesis over its entire range into three numbers, which tells you roughly how big they'll be on a large test set.]

[Now I will write down a list of consequences of what we've just learned.]

- Underfitting = too much bias
- Most overfitting caused by too much variance
- Training error reflects bias but not variance; test error reflects both
  [which is why low training error can fool you when you've overfitted]
- For many distributions, variance $\to 0$ as $n \to \infty$
- If h can fit g exactly, for many distributions bias $\to 0$ as $n \to \infty$
- If h cannot fit g well, bias is large at "most" points
- Adding a good feature reduces bias; adding a bad feature rarely increases it
- Adding a feature usually increases variance [don't add a feature unless it reduces bias more]
- Can't reduce irreducible error [hence its name]
- Noise in test set affects only Var(ϵ);
  noise in training set affects only bias & Var(h)
- We can't precisely measure bias or variance of real-world data
  [because we cannot know g exactly and our noise model might be wrong]
- But we can test learning algs by choosing g & making synthetic data



splinefit.pdf, biasvarspline.pdf (ISL, Figures 2.9 and 2.12) [At left, a data set is fit with splines having various degrees of freedom. The synthetic data is taken from the black curve with added noise. At center, we plot training error (gray) and test error (red) as a function of the number of degrees of freedom. At right, we plot the squared test error as a sum of squared bias (blue) and variance (orange). As the number of degrees of freedom increases, the training and test errors both decrease up to degree 6 because the bias decreases, but for higher degrees the test error increases because the variance increases.]



Example: Least-Squares Linear Reg.

For simplicity, no fictitious dimension.
[This implies that our linear regression function has to be zero at the origin.]

Model: $g(z) = v^\top z$ (ground truth is linear)
[So we could fit g perfectly with a linear h if not for the noise in the training set.]
Let e be noise n-vector, $e_i \sim \mathcal{N}(0, \sigma^2)$
Training labels: $y = Xv + e$
[X & y are the inputs to linear regression. We don't know v or e.]

Lin. reg. computes weights

$$w = X^+ y = X^+(Xv + e) = v + \underbrace{X^+ e}_{\text{noise in weights}}$$ [We want $w = v$, but the noise in y becomes noise in w.]

BIAS is $\mathrm{E}[h(z)] - g(z) = \mathrm{E}[w^\top z] - v^\top z = z^\top \mathrm{E}[w - v] = z^\top \mathrm{E}[X^+ e] = z^\top \mathrm{E}[X^+]\mathrm{E}[e] = 0$

Warning: This does not mean $h(z) - g(z)$ is always 0!
Sometimes +ve, sometimes −ve, mean over training sets is 0.
[Those deviations from the mean are captured in the variance.]

[When the bias is zero, a perfect fit is possible. But when a perfect fit is possible, not all learning methods give you a bias of zero; here it's a benefit of the squared error loss function. With a different noise or a different loss function, we might have a nonzero bias even fitting a linear h to a linear g.]

VARIANCE is $\mathrm{Var}(h(z)) = \mathrm{Var}(w^\top z) = \mathrm{Var}(z^\top v + z^\top X^+ e) = \mathrm{Var}(z^\top X^+ e)$

[This is the dot product of a vector $z^\top X^+$ with an isotropic, normally distributed vector e. The dot product reduces it to a one-dimensional Gaussian along the direction $z^\top X^+$, so this variance is just the variance of the 1D Gaussian times the squared length of the vector $z^\top X^+$.]

$$\begin{aligned} &= \sigma^2 \left\| z^\top X^+ \right\|^2 = \sigma^2 z^\top (X^\top X)^{-1} X^\top X (X^\top X)^{-1} z \\ &= \sigma^2 z^\top (X^\top X)^{-1} z \end{aligned}$$

If we choose coordinate system so D has mean zero, then $X^\top X \to n\,\mathrm{Cov}(D)$ as $n \to \infty$, so for $z \sim D$,

$$\mathrm{Var}(h(z)) \approx \sigma^2 \frac{d}{n}$$

[where d is the dimension—the number of features per sample point.]

[If anyone asks: With the eigendecomposition $\mathrm{Cov}(D) = V \Lambda V^\top$, we have $\mathrm{E}[z^\top \mathrm{Cov}(D)^{-1} z] = \mathrm{E}[\|\Lambda^{-1/2} V^\top z\|^2] = \sum_{i=1}^d \mathrm{E}[(v_i \cdot z)^2]/\lambda_i$. But as $z \sim D$, $\mathrm{E}[(v_i \cdot z)^2] = \mathrm{Var}[v_i \cdot z] = \lambda_i$, so $\mathrm{E}[z^\top \mathrm{Cov}(D)^{-1} z] = d$. Hence $\mathrm{E}[z^\top (X^\top X)^{-1} z] \approx d/n$.]

Takeaways: Bias can be zero when hypothesis function can fit the real one!
[This is a nice property of the squared error loss function.]
Variance portion of RSS (overfitting) decreases as $1/n$ (sample points),
increases as d (features)
or $O(d^p)$ if you use degree-p polynomials.

[I've used linear regression because it's a relatively simple example. But the bias-variance trade-off applies to many learning algorithms, including classification as well as regression. But for most learning algorithms, the math gets a lot more complicated than this, if you can do it at all. Sometimes there are multiple competing bias-variance models and no consensus on which is the right one.]