Degrees of Freedom

Advanced Methods for Data Analysis (36-402/36-608)

Spring 2014

1 Degrees of freedom

1.1 Motivation

- So far we've seen several methods for estimating the underlying regression function $r(x) = \mathbb{E}(Y|X = x)$ (linear regression, k-nearest-neighbors, kernel smoothing), and next time we'll consider another one (smoothing splines). Often, in a predictive setting, we want to compare (estimates of) test error between such methods, in order choose between them
- We've learned how to do this: cross-validation. But in a sense, comparing cross-validation curves between methods is not straightforward, because each curve is parametrized differently. E.g., linear regression has no tuning parameters and so we would report just one error value; k-nearest-neighbors would give us an error curve over k; kernel regression would give us an error curve over the bandwidth h; smoothing splines would give us an error curve over the smoothing parameter λ
- So what does it actually mean to choose kernel regression with $h = 1.5$, over say, k-nearest-neighbors with $k = 10$ or smoothing slines with $\lambda = 0.417$? I.e., does the $h = 1.5$ model from kernel regression correspond to a more of less complex estimate than the $k = 10$ model from k-nearest-neighbors, or the $\lambda = 0.417$ model from smoothing splines?
- The notion of *degrees of freedom* gives us a way of precisely making this comparison. Roughly speaking, the degrees of freedom of a fitting procedure (like kernel regression with $h = 1.5$, or k-nearest-neighbors with $k = 10$) describes the *effective number of parameters* used by this procedure, and hence provides a quantitive measure of estimator complexity
- Keeping track of degrees of freedom therefore saves us from unsuspectingly comparing a procedure that uses say, 10 effective parameters to another that uses 100

1.2 Definition

- Even though the concept it represents is quite broad, degrees of freedom has a rigorous definition. Suppose that we observe

$$y_i = r(x_i) + \epsilon_i, \quad i = 1, \ldots n,$$

where the errors ϵ_i, $i = 1, \ldots n$ are uncorrelated with common variance $\sigma^2 > 0$ (note: this is weaker than assuming $\epsilon_i \sim N(0, \sigma^2)$, i.i.d. for $i = 1, \ldots n$). Here we will treat the predictor measurements x_i, $i = 1, \ldots n$ as fixed (equivalently: consider conditioning on the values of the random predictors). Now consider the fitted values $\hat{y}_i = \hat{r}(x_i)$, $i = 1, \ldots n$ from a regression estimator $\hat{r}$. We define the degrees of freedom of $\hat{y}$ (i.e., the degrees of freedom of $\hat{r}$) as

$$\mathrm{df}(\hat{y}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \mathrm{Cov}(\hat{y}_i, y_i). \tag{1}$$

To reiterate: this covariance treats only y_i, $i = 1, \ldots n$ as random (and not x_i, $i = 1, \ldots n$)

- The definition of degrees of freedom in (1) looks at the amount of covariance between each point y_i and its corresponding fitted values $\hat{y}_i$. We add these up over $i = 1, \ldots n$, and divide the result by σ^2 (dividing by σ^2 gets rid of the dependence of the sum on the marginal error variance)

- It is going to be helpful for some purposes to rewrite the definition of degrees of freedom in matrix notation. This is

$$\mathrm{df}(\hat{y}) = \frac{1}{\sigma^2}\mathrm{tr}\Big(\mathrm{Cov}(\hat{y}, y)\Big), \tag{2}$$

where we write $y = (y_1, \ldots y_n) \in \mathbb{R}^n$ and $\hat{y} = (\hat{y}_1, \ldots \hat{y}_n) \in \mathbb{R}^n$

1.3 Examples

- To get a sense for degrees of freedom, it helps to work through several basic examples

- Simple average estimator: consider $\hat{y}^{\mathrm{ave}} = (\bar{y}, \ldots \bar{y})$, where $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$. Then

$$\mathrm{df}(\hat{y}^{\mathrm{ave}}) = \frac{1}{\sigma^2}\sum_{i=1}^n \mathrm{Cov}(\bar{y}, y_i) = \frac{1}{\sigma^2}\sum_{i=1}^n \frac{\sigma^2}{n} = 1,$$

  i.e., the effective number of parameters used by $\mathrm{df}(\hat{y}^{\mathrm{ave}})$ is just 1, which makes sense

- Identity estimator: consider $\hat{y}^{\mathrm{id}} = (y_1, \ldots y_n)$. Then

$$\mathrm{df}(\hat{y}^{\mathrm{id}}) = \frac{1}{\sigma^2}\sum_{i=1}^n \mathrm{Cov}(y_i, y_i) = n,$$

  i.e., $\hat{y}^{\mathrm{id}}$ uses n effective parameters, which again makes sense

- Linear regression: consider the fitted values from linear regression of y on x,

$$\hat{y}^{\mathrm{linreg}} = x\hat{\beta} = x(x^T x)^{-1}x^T y.$$

  Here x is the $n \times p$ predictor matrix (with x_i along its ith row). Then, relying on the matrix form of degrees of freedom,

$$\begin{aligned}
\mathrm{df}(\hat{y}^{\mathrm{linreg}}) &= \frac{1}{\sigma^2}\mathrm{tr}\Big(\mathrm{Cov}\big(x(x^T x)^{-1}x^T y, y\big)\Big) \\
&= \frac{1}{\sigma^2}\mathrm{tr}\Big(x(x^T x)^{-1}x^T \mathrm{Cov}(y, y)\Big) \\
&= \mathrm{tr}\Big(x(x^T x)^{-1}x^T\Big) \\
&= \mathrm{tr}\Big(x^T x(x^T x)^{-1}\Big) \\
&= p.
\end{aligned}$$

  So we have shown that the effective number of parameters used by $\hat{y}^{\mathrm{linreg}}$ is p. This is highly intuitive, since we have estimated p regression coefficients

- Linear smoothers: recall that a linear smooth has the form

$$\hat{r}^{\mathrm{linsm}}(x) = \sum_{j=1}^n w(x, x_j) \cdot y_j.$$

This means that

$$\hat{y}_i^{\text{linsm}} = \hat{r}^{\text{linsm}}(x_i) = \sum_{j=1}^{n} w(x_i, x_j) \cdot y_j,$$

i.e., we can write

$$\hat{y}^{\text{linsm}} = Sy,$$

for the matrix $S \in \mathbb{R}^{n \times n}$ defined as $S_{ij} = w(x_i, x_j)$. Calculating degrees of freedom, again in matrix form,

$$\begin{aligned}
\text{df}(\hat{y}^{\text{linsm}}) &= \frac{1}{\sigma^2} \text{tr}\Big(\text{Cov}(Sy, y)\Big) \\
&= \frac{1}{\sigma^2} \text{tr}\Big(S\text{Cov}(y, y)\Big) \\
&= \text{tr}(S) \\
&= \sum_{i=1}^{n} w(x_i, x_i).
\end{aligned}$$

This is a very useful formula! Recall that k-nearest-neighbors and kernel regression and both linear smoothers, and we will see that smoothing splines are too, so we can calculate degrees of freedom for all of these simply by summing these weights

- As a concrete example: consider k-nearest-neighbors regression with some fixed value of $k \geq 1$. Recall that here

$$w(x, x_j) = \begin{cases} \frac{1}{k} & \text{if } x_j \text{ is one of the } k \text{ closest points to } x \\ 0 & \text{else.} \end{cases}$$

Therefore $w(x_i, x_i) = 1/k$, and

$$\text{df}(\hat{y}^{\text{knn}}) = \sum_{i=1}^{n} \frac{1}{k} = \frac{n}{k}.$$

Think: what happens for small k? Large k? Does this match your intuition for the complexity of the k-nearest-neighbors fit?

2 Estimating degrees of freedom

2.1 Naive approach: pairs bootstrap

- Degrees of freedom can't always be calculated analytically, as we did above. In fact, at large, it's rather uncommon for this to be the case. As an extreme example, if the fitting procedure $\hat{r}$ is just a black box (e.g., just an R function whose mechanism is unknown), then we would really have no way of analytically counting its degrees of freedom. However, the expression in (1) is still well-defined for any fitting procedure $\hat{r}$, and to get an estimate of its degrees of freedom, we can estimate the covariance terms $\text{Cov}(\hat{y}_i, y_i)$ via the boostrap

- A naive first approach would be to use the bootstrap, as we learned it in the last class. That is, for $b = 1$ to B (say $B = 1000$), we repeat the following steps:

  – draw bootstrap samples

$$(\tilde{x}_i^{(b)}, \tilde{y}_i^{(b)}) \overset{\text{i.i.d.}}{\sim} \text{Unif}\Big\{(x_1, y_1), \ldots (x_n, y_n)\Big\}, \quad i = 1, \ldots n;$$

– recompute the estimator $\hat{r}^{(b)}$ on the samples $(\tilde{x}_i^{(b)}, \tilde{y}_i^{(b)})$, $i = 1, \ldots n$;

– store $\tilde{y}^{(b)} = (\tilde{y}_1^{(b)}, \ldots \tilde{y}_n^{(b)})$, and the fitted values $\hat{y}^{(b)} = (\hat{r}^{(b)}(\tilde{x}_1^{(b)}), \ldots \hat{r}^{(b)}(\tilde{x}_n^{(b)}))$.

At the end, we approximate the covariance of $\hat{y}_i$ and y_i by the empirical covariance between $\hat{y}_i^{(b)}$ and $\tilde{y}_i^{(b)}$ over $b = 1, \ldots B$, i.e.,

$$\mathrm{Cov}(\hat{y}_i, y_i) \approx \frac{1}{B} \sum_{b=1}^{B} \Big(\hat{y}_i^{(b)} - \frac{1}{B} \sum_{r=1}^{B} \hat{y}_i^{(r)} \Big) \cdot \Big(\tilde{y}_i^{(b)} - \frac{1}{B} \sum_{r=1}^{B} \tilde{y}_i^{(r)} \Big),$$

and sum this up over $i = 1, \ldots n$ to yield our bootstrap estimate for degrees of freedom

$$\mathrm{df}(\hat{y}) \approx \frac{1}{\sigma^2} \sum_{i=1}^{n} \left(\frac{1}{B} \sum_{b=1}^{B} \Big(\hat{y}_i^{(b)} - \frac{1}{B} \sum_{r=1}^{B} \hat{y}_i^{(r)} \Big) \cdot \Big(\tilde{y}_i^{(b)} - \frac{1}{B} \sum_{r=1}^{B} \tilde{y}_i^{(r)} \Big) \right).$$

(For simplicity, you can assume that σ^2 is known; otherwise, we'd have to estimate it too)

- We'll refer to this sampling scheme as the *pairs bootstrap*, since we are bootstrapping (x_i, y_i) pairs. In this particular application, it actually doesn't yield a very good estimate of degrees of freedom ... why? (Hint: think about what is random and what is fixed in (1))

2.2 Informed approach: residual bootstrap

- A better approach for estimating degrees of freedom is to use the *residual bootstrap*. Here, after fitting $\hat{y}_i = \hat{r}(x_i)$, $i = 1, \ldots n$ using the original samples (x_i, y_i), $i = 1, \ldots n$, we record the (empirical) residuals

$$\hat{e}_i = y_i - \hat{y}_i, \quad i = 1, \ldots n.$$

Then for $b = 1, \ldots B$, we repeat:

– draw bootstramp samples $(x_i, \tilde{y}_i^{(b)})$, $i = 1, \ldots n$ according to

$$\tilde{y}_i^{(b)} = \hat{y}_i + \tilde{e}_i^{(b)}, \text{ where } \tilde{e}_i^{(b)} \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}\{\hat{e}_1, \ldots \hat{e}_n\}, \quad i = 1, \ldots n,$$

i.e., instead of resampling pairs with replacement, we're resampling residuals with replacement;

– proceed as before.

We use the same formula as above for estimating the covariance terms and degrees of freedom

- This ends up being more accurate in estimating degrees of freedom ... why? (Hint: again, think about what is being treated as random, and what is being treated as fixed)

3 Using degrees of freedom for error estimation

3.1 Optimism

- Degrees of freedom is directly related to a concept called optmism; once we see the precise relationship, we'll see how we can use it to construct estimates of expected test error (computationally efficient alternatives to cross-validation)

- Remember, as described above, we're assuming the model

$$y_i = r(x_i) + \epsilon_i, \quad i = 1, \ldots n,$$

where $\epsilon = (\epsilon_1, \ldots \epsilon_n)$ satisfy $\mathbb{E}(\epsilon) = 0$ and $\mathrm{Var}(\epsilon) = \sigma^2 I$ (this is a shorter way of writing that ϵ_i, $i = 1, \ldots n$ have mean zero, and are uncorrelated with marginal variance $\sigma^2 > 0$). Given an estimator $\hat{r}$ producing fitted values $\hat{y}_i = \hat{r}(x_i)$, $i = 1, \ldots n$, the expected training error of $\hat{r}$ is

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i - \hat{y}_i)^2\Big].$$

Meanwhile, if

$$y_i' = r(x_i) + \epsilon_i', \quad i = 1, \ldots n$$

is an independent test sample (important: note here that the predictors measurements x_i, $i = 1, \ldots n$ are the same—i.e., we are considering these fixed) from the same distribution as our training sample, then

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i' - \hat{y}_i)^2\Big]$$

is the expected test error

- Recall that these two quantities—training and test errors—behave very differently, and it is usually the test error that we seek for the purposes of model validation or model selection. Interestingly, it turns out that (in this simple setup, with x_i, $i = 1, \ldots n$ fixed) we have the relationship

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i' - \hat{y}_i)^2\Big] = \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i - \hat{y}_i)^2\Big] + \frac{2\sigma^2}{n}\mathrm{df}(\hat{y}).$$

In words, the expected test error is exactly the expected training error plus a constant factor $(2\sigma^2/n)$ times the degrees of freedom

- From this decomposition, we can immediately see that with a larger degrees of freedom, i.e., a more complex fitting method, the test error is going to be larger than the training error. Perhaps more evocative is to rewrite the relationship above as

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i' - \hat{y}_i)^2\Big] - \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i - \hat{y}_i)^2\Big] = \frac{2\sigma^2}{n}\mathrm{df}(\hat{y}).$$

We will call left-hand side, the difference between expected test and training errors, the *optimism* of the estimator $\hat{r}$. We can see that it is precisely equal to $2\sigma^2/n$ times its degrees of freedom; so again, the higher the degrees of freedom, i.e., the more complex the fitting procedure, the larger the gap between testing and training errors

3.2 Error estimation

- The relationship discussed in the last section actually leads to a very natural estimate for the expected test error. Consider

$$T = \frac{1}{n}\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \frac{2\sigma^2}{n}\mathrm{df}(\hat{y}),$$

i.e., the observed training error of $\hat{y}$ plus $2\sigma^2/n$ times its degrees of freedom. From the previous section, we know that

$$\mathbb{E}(T) = \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^n (y_i' - \hat{y}_i)^2\Big],$$

i.e., T is an unbiased estimate for the expected test error. Hence if we knew an estimator's degrees of of freedom, then we could use T to approximate its test error—note that this is a comptuationally efficient alternative to cross-validation (no extra computation really needed, beyond the training error)

- What happens when we don't know its degrees of freedom? If have a good estimate $\widehat{\mathrm{df}}(\hat{y})$ for the degrees of freedom of $\hat{y}$, then we can simply form the error estimate

$$T = \frac{1}{n}\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \frac{2\sigma^2}{n}\widehat{\mathrm{df}}(\hat{y}),$$

and by the same logic, $\mathbb{E}(T)$ will be close to the expected test error as long as $\mathbb{E}[\widehat{\mathrm{df}}(\hat{y})]$ is close to $\mathrm{df}(\hat{y})$. Such an estimate $\widehat{\mathrm{df}}(\hat{y})$ could have come from, say, the bootstrap, as discussed in Section 2 (but in this case, you should be aware of the fact that these bootstrap calculations themselves may be roughly as expensive as cross-validation). But the estimate $\widehat{\mathrm{df}}(\hat{y})$ can also come from other means, e.g., from an analytic form. We'll see an example of this later in the course (when we discuss ℓ_1 regularization, i.e., the lasso)

- Finally, what about σ^2? In general, if we don't know its true value (which is going to pretty much going to always be the case in practice), then we will have to estimate it too. Notice however that if we used the bootstrap to form the estimate $\widehat{\mathrm{df}}(\hat{y})$, then this already provides us with an estimate of $2\sigma^2\mathrm{df}(\hat{y})$, so we don't have to estimate σ^2 on its own

3.3 Model selection

- Finally, we can apply the above ideas to the model selection problem. Suppose our estimate $\hat{r} = \hat{r}_\theta$ depends on a tuning parameter θ; also write $\hat{y}_\theta$ for the fitted values at θ. Then over a grid of values, say $\theta \in \{\theta_1, \ldots \theta_m\}$, we compute the error estimate

$$T_\theta = \frac{1}{n}\sum_{i=1}^n \Big(y_i - (\hat{y}_\theta)_i\Big)^2 + \frac{2\sigma^2}{n}\mathrm{df}(\hat{y}_\theta)$$

(possibly replacing the degrees of freedom term and σ^2 in the above with estimates, if needed), and choose θ to minimize T_θ. That is, we select

$$\hat{\theta} = \underset{\theta \in \{\theta_1, \ldots \theta_m\}}{\mathrm{argmin}} \; \frac{1}{n}\sum_{i=1}^n \Big(y_i - (\hat{y}_\theta)_i\Big)^2 + \frac{2\sigma^2}{n}\mathrm{df}(\hat{y}_\theta)$$

- This may look familiar to you if we consider the case of linear regression on any number of predictor variables between 1 and p. Here, θ indexes the number of predictors used in a linear regression, and simply to make things look more familiar, we will rewrite this parameter as k. Hence $k \in \{1, \ldots p\}$, and the above model selection criterion becomes

$$\hat{k} = \underset{k \in \{1, \ldots p\}}{\mathrm{argmin}} \; \frac{1}{n}\sum_{i=1}^n \Big(y_i - (\hat{y}_k)_i\Big)^2 + \frac{2\sigma^2}{n}k.$$

You may recall this as the C_p criterion for model selection in linear regression (related to AIC, and then there's BIC, and RIC, ...)