12:14 Friday 13th November, 2015
See updates and corrections at http://www.stat.cmu.edu/~cshalizi/mreg/

Lecture 21: Model Selection

36-401, Fall 2015, Section B

10 November 2015

Contents

1 Generalization and Optimism

We estimated our model by minimizing the mean squared error on our data:

$$\widehat{\beta} = \underset{\mathbf{b}}{\operatorname{argmin}} \frac{1}{n}(\mathbf{y} - \mathbf{x}\mathbf{b})^T(\mathbf{y} - \mathbf{x}\mathbf{b})$$

Different linear models will amount to different choices of the design matrix $\mathbf{x}$ — we add or drop variables, we add or drop interactions or polynomial terms, etc., and this adds or removes columns from the design matrix. We might consider doing selecting among models themselves by minimizing the MSE. This is a very bad idea, for a fundamental reason:

> Every model is too optimistic about how well it will actually predict.

Let's be very clear about what it would mean to predict well. The most challenging case would be that we see a new *random* point, with predictor values $X_1, \ldots X_p$ and response Y, and our *old* $\widehat{\beta}$ has a small expected squared error:

$$\mathbb{E}\left[\left(Y - \left(\hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j\right)\right)^2\right]$$

Here both Y and the X's are random (hence the capital letters), so we might be asking the model for a prediction at a point it never saw before. (Of course if we have multiple identically distributed (X, Y) pairs, say q of them, the expected MSE over those q points is just the same as the expected squared error at one point.)

An easier task would be to ask the model for predictions at the *same* values of the predictor variables as before, but with different random noises. That is, we fit the model to

$$\mathbf{Y} = \mathbf{x}\beta + \epsilon$$

and now Tyche[1] reach into her urn and gives us

$$\mathbf{Y}' = \mathbf{x}\beta + \epsilon'$$

where ϵ and ϵ' are independent but identically distributed. The design matrix is the same, the true parameters β are the same, but the noise is different[2]. We now want to see if the coefficients we estimated from $(\mathbf{x}, \mathbf{Y})$ can predict $(\mathbf{x}, \mathbf{Y}')$. Since the only thing that's changed is the noise, if the coefficients can't predict well any more, that means that they were really just memorizing the noise, and not actually doing anything useful.

[1] Look her up.

[2] If we really are in an experimental setting, we really could get a realization of $\mathbf{Y}'$ just by running the experiment a second time. With surveys or with observational data, it would be harder to actually realize $\mathbf{Y}'$, but mathematically at least it's unproblematic.

Our out-of-sample expected MSE, then, is

$$\mathbb{E}\left[n^{-1}(\mathbf{Y'} - \mathbf{x}\widehat{\beta})^T(\mathbf{Y'} - \mathbf{x}\widehat{\beta})\right]$$

It will be convenient to break this down into an average over data points, and to abbreviate $\mathbf{x}\widehat{\beta} = \widehat{\mathbf{m}}$, the vector of fitted values. Notice that since the predictor variables and the coefficients aren't changing, our predictions are the same both in and out of sample — at point i, we will predict $\widehat{m}_i$.

In this notation, then, the expected out-of-sample MSE is

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \widehat{m}_i)^2\right]$$

We'll compare this to the expected in-sample MSE,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2\right]$$

Notice that $\widehat{m}_i$ is a function of Y_i (among other things), so those are dependent random variables, while $\widehat{m}_i$ and Y'_i are completely statistically independent[3].

Break this down term by term. What's the expected value of the i^{th} in-sample squared error?

$$\begin{aligned}
\mathbb{E}\left[(Y_i - \hat{m}_i)^2\right] &= \mathrm{Var}\left[Y_i - \hat{m}_i\right] + \left(\mathbb{E}\left[Y_i - \hat{m}_i\right]\right)^2 && (1)\\
&= \mathrm{Var}\left[Y_i\right] + \mathrm{Var}\left[\hat{m}_i\right] - 2\mathrm{Cov}\left[Y_i, \hat{m}_i\right] + \left(\mathbb{E}\left[Y_i\right] - \mathbb{E}\left[\hat{m}_i\right]\right)^2 && (2)
\end{aligned}$$

The covariance term is not (usually) zero, because, as I just said, $\hat{m}_i$ is a function of, in part, Y_i.

On the other hand, what's the expected value of the i^{th} squared error on new data?

$$\begin{aligned}
\mathbb{E}\left[(Y'_i - \hat{m}_i)^2\right] &= \mathrm{Var}\left[Y_i\prime - \hat{m}_i\right] + \left(\mathbb{E}\left[Y'_i - \hat{m}_i\right]\right)^2 && (3)\\
&= \mathrm{Var}\left[Y'_i\right] + \mathrm{Var}\left[\hat{m}_i\right] - 2\mathrm{Cov}\left[Y'_i, \hat{m}_i\right] + \left(\mathbb{E}\left[Y'_i\right] - \mathbb{E}\left[\hat{m}_i\right]\right)^2 && (4)
\end{aligned}$$

Y'_i is independent of Y_i, but has the same distribution. This tells us that $\mathbb{E}\left[Y'_i\right] = \mathbb{E}\left[Y_i\right]$, $\mathrm{Var}\left[Y'_i\right] = \mathrm{Var}\left[Y_i\right]$, but $\mathrm{Cov}\left[Y'_i, \hat{m}_i\right] = 0$. So

$$\begin{aligned}
\mathbb{E}\left[(Y'_i - \hat{m}_i)^2\right] &= \mathrm{Var}\left[Y_i\right] + \mathrm{Var}\left[\hat{m}_i\right] + \left(\mathbb{E}\left[Y_i\right] - \mathbb{E}\left[\hat{m}_i\right]\right)^2 && (5)\\
&= \mathbb{E}\left[(Y_i - \hat{m}_i)^2\right] + 2\mathrm{Cov}\left[Y_i, \hat{m}_i\right] && (6)
\end{aligned}$$

Averaging over data points,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \widehat{m}_i)^2\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2\right] + \frac{2}{n}\sum_{i=1}^{n}\mathrm{Cov}\left[Y_i, \hat{m}_i\right]$$

[3] That might sound weird, but remember we're holding $\mathbf{x}$ fixed in this exercise, so what we mean is that knowing $\widehat{m}_i$ doesn't give us an extra information about Y'_i beyond what we'd get from knowing the values of the X variables.

Clearly, we need to get a handle on that sum of covariances.

For a linear model, though, $\mathrm{Cov}\left[Y_i, \hat{m}_i\right] = \sigma^2 H_{ii}$ (Exercise 1). So, for linear models,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \widehat{m}_i)^2\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2\right] + \frac{2}{n}\sigma^2 \operatorname{tr}\mathbf{H}$$

and we know that with p predictors and one intercept, $\operatorname{tr}\mathbf{H} = p+1$ (Homework 5). Thus, for linear models,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \widehat{m}_i)^2\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2\right] + \frac{2}{n}\sigma^2(p+1)$$

Of course, we don't actually know the *expectation* on the right-hand side, but we do have a sample estimate of it, which is the in-sample MSE. If the law of large numbers is still our friend,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(Y'_i - \widehat{m}_i)^2\right] \approx \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2 + \frac{2}{n}\sigma^2(p+1)$$

The second term on the right, $(2/n)\sigma^2(p+1)$, is the **optimism** of the model — the amount by which its in-sample MSE systematically under-estimates its true expected squared error. Notice that this:

- Grows with σ^2: more noise gives the model more opportunities to seem to fit well by capitalizing on chance.
- Shrinks with n: at any fixed level of noise, more data makes it harder to pretend the fit is better than it really is.
- Grows with p: every extra parameter is another control which can be adjusted to fit to the noise.

Minimizing the in-sample MSE completely ignores the bias from optimism, so it is guaranteed to pick models which are too large and predict poorly out of sample. If we could calculate the optimism term, we could at least use an unbiased estimate of the true MSE on new data.

Of course, we do not actually know σ^2.

2 Mallow's C_p Statistic

The Mallows C_p statistic just substitutes in a feasible estimator of σ^2, which is $\hat{\sigma}^2$ *from the largest model we consider*. This will be an unbiased estimator of σ^2 if the real model is smaller (contains a strict subset of the predictor variables), but not vice versa[4].

[4] This assumes the largest model must contain the truth!

That is, for a linear model with $p+1$ coefficients fit by OLS,

$$C_p \equiv = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{m}_i)^2 + \frac{2}{\hat{\sigma}^2} n(p+1) \tag{7}$$

The selection rule is to pick the model which minimizes C_p.

We can think of C_p as having two parts,

$$C_p = MSE + (\text{penalty})$$

From one point of view, the penalty is just an estimate of the bias. From another point of view, it's a cost we're imposing on models for having extra parameters. Every new parameter has got to pay that cost by reducing the MSE by at least a certain amount; if it doesn't, the extra parameter isn't worth it.

(Before this, we've only been dealing with *one* model, so we've not had to distinguish carefully between the in-sample MSE and the maximum likelihood estimate of σ^2. With multiple models floating around, though, each can have its own MSE, but there is only one true σ^2, and we need *an* estimate of it.)

For comparing models, we really care about differences:

$$\Delta C_p = MSE_1 - MSE_2 + \frac{2}{n}\hat{\sigma}^2(p_1 - p_2) \tag{8}$$

(The extra term for the intercept, being common to both models, doesn't contribute.)

Alternate form of C_p You will find many references which define C_p somewhat differently:

$$\frac{nMSE}{\hat{\sigma}^2} - n + 2p \tag{9}$$

and say that the optimal value is close to p, not close to 0. To see that this selects exactly the same models as the rule given above, take a difference between two models, with MSE's MSE_1, MSE_2 and p_1, p_2 predictors. We get

$$\frac{n(MSE_1 - MSE_2)}{\hat{\sigma}^2} + 2(p_1 - p_2)$$

Dividing by n and multiplying by $\hat{\sigma}^2$ gives us back Eq. 8. There are reasons to assert that Eq. 9 should indeed be close to p for the right model (if the Gaussian noise assumption holds), but Eq. 7 is a good estimate of the out-of-sample error, and a good model selection rule, much more broadly.

2.1 R^2 and Adjusted R^2

Recall that

$$R^2 = 1 - \frac{MSE}{s_Y^2}$$

Picking a model by maximizing R^2 is thus equivalent to picking a model by minimizing MSE. It is therefore stupid for exactly the same reasons that minimizing MSE across models is stupid.

Recall that the adjusted R^2 is

$$R^2_{adj} = 1 - \frac{MSE\frac{n}{n-p-1}}{s^2_Y}$$

That is, it's R^2 with the unbiased estimator of σ^2. Maximizing adjusted R^2 therefore corresponds to minimizing that unbiased estimator. What does that translate to?

$$\begin{aligned} MSE\frac{n}{n-p-1} &= MSE\frac{1}{1-(p+1)/n} && (10)\\ &\approx MSE\left(1+\frac{p+1}{n}\right) && (11)\\ &= MSE + MSE\frac{p+1}{n} && (12) \end{aligned}$$

where the approximation becomes exact as $n \to \infty$ with p fixed[5]. Even for the completely right model, where MSE is a consistent estimator of $\hat{\sigma}^2$, the correction or penalty is only half as big as we've seen it should be. Selecting models using adjusted R^2 is not completely stupid, as maximizing R^2 is, but it is still not going to work very well.

3 Akaike Information Criterion (AIC)

The great Japanese statistician Hirotugu Akaike proposed a famous model selection rule which also has the form of "in-sample performance plus penalty". What has come to be called the **Akaike information criterion** (AIC) is

$$AIC(S) \equiv L_S - \dim(S)$$

where L_S is the log likelihood of the model S, evaluated at the maximum likelihood estimate, and $\dim(S)$ is the dimension of S, the number of adjustable parameters it has. Akaike's rule is to pick the model which maximizes AIC[6].

The reason for this definition is that Akaike showed AIC/n is an unbiased estimate of the expected log-probability the estimated parameters will give to a new data point which it hasn't seen before, if the model is right. This is the natural counterpart of expected squared error for more general distributions

[5] Use the binomial theorem to expand $1/(1-u)$ as $1+u+u^2+\ldots$, and truncate the series at first order. (If u is small, u^2 is tiny, and the higher powers microscopic.)

[6] Actually, in his original paper (Akaike, 1973), he proposed using *twice* this, to simplify some calculations involving chi-squared distributions. Many subsequent authors have since kept the factor of 2, which of course will not change which model is selected. Also, some authors define AIC as negative of this, and then minimize it; again, clearly the same thing.

than the Gaussian. IF we do specialize to linear-Gaussian models, then we've seen (Lecture 10) that

$$L = -\frac{n}{2}(1 + \log 2\pi) - \frac{n}{2}\log MSE$$

and the dimension of the model is $p + 2$ (because σ^2 is also an adjustable parameter). Notice that $-\frac{n}{2}(1+\log 2\pi)$ doesn't involve the parameters at all. If we compare AICs for two models, with mean squared errors in-sample of MSE_1 and MSE_2, and one with p_1 predictors and the other with p_2, the difference in AICs will be

$$\Delta AIC = -\frac{n}{2}\log MSE_1 + \frac{n}{2}\log MSE_2 - (p_1 - p_2)$$

To relate this to C_p, let's write $MSE_2 = MSE_1 + \Delta MSE$. Then

$$\begin{aligned} \Delta AIC &= -\frac{n}{2}\log MSE_1 + \frac{n}{2}\log MSE_1\left(1 + \frac{\Delta MSE}{MSE_1}\right) - (p_1 - p_2) && (13) \\ &= -\frac{n}{2}\log\left(1 + \frac{\Delta MSE}{MSE_1}\right) - (p_1 - p_2) && (14) \end{aligned}$$

Now let's suppose that model 1 is actually the correct model, so $MSE_1 = \hat{\sigma}^2$, and that ΔMSE is small compared to $\hat{\sigma}^2$, so[7]

$$\begin{aligned} \Delta AIC &\approx -\frac{n}{2}\frac{\Delta MSE}{\hat{\sigma}^2} - (p_1 - p_2) && (15) \\ \frac{-2\hat{\sigma}^2}{n}\Delta AIC &\approx \Delta MSE + \frac{2}{n}\hat{\sigma^2}(p_1 - p_2) = \Delta C_p && (16) \end{aligned}$$

So, if one of the models we're looking at is actually the correct model, and the others aren't too different from it, picking by maximizing AIC will give the same answer as picking by minimizing C_p.

Other Uses of AIC AIC can be applied whenever we have a likelihood. It is therefore used for tasks like comparing models of probability distributions, or predictive models where the whole distribution is important. C_p, by contrast, really only makes sense if we're trying to do regression and want to use squared error.

3.1 Why $-\dim(S)$?

Akaike had a truly brilliant argument for subtracting a penalty equal to the number of parameters from the log-likelihood, which is too pretty not to at least sketch here.[8]

Generically, say that the parameter vector is θ, and its true value is θ^*. (For linear regression with Gaussian noise, θ consists of all $p+1$ coefficients plus σ^2.)

[7] Taylor expand $\log 1 + u$ around 1 to get $\log 1 + u \approx u$, for u close to 0.

[8] Nonetheless, this subsection is optional.

The length of this vector, which is $\dim(S)$, is let's say d. (For linear regression with Gaussian noise, $d = p + 2$.) The maximum likelihood estimate is $\hat{\theta}$. We know that the derivative of the likelihood is zero at the MLE:

$$\nabla L(\hat{\theta}) = 0$$

Let's do a Taylor series expansion of $\nabla L(\theta)$ around the true parameter value θ^*:

$$\nabla L(\theta) = \nabla L(\theta^*) + (\theta - \theta^*) \nabla\nabla L(\theta^*)$$

Here $\nabla\nabla L(\theta^*)$ is the $d \times d$ matrix of second partial derivatives of L, evaluated at θ^*. This is called the **Hessian**, and would traditionally be written $\mathbf{H}$, but that would lead to confusion with the hat matrix, so I'll call it $\mathbf{K}$. Therefore the Taylor expansion for the gradient of the log-likelihood is

$$\nabla L(\theta) = \nabla L(\theta^*) + (\theta - \theta^*)\mathbf{K}$$

Applied to the MLE,

$$\mathbf{0} = \nabla L(\theta^*) + (\hat{\theta} - \theta^*)\mathbf{K}$$

or

$$\hat{\theta} = \theta^* - K^{-1} \nabla L(\theta^*)$$

What is the *expected* log-likelihood, on new data, of $\hat{\theta}$? Call this expected log-likelihood ℓ (using a lower-case letter to indicate that it is non-random). Doing another Taylor series,

$$\ell(\theta) \approx \ell(\theta^*) + (\theta - \theta^*)^T \nabla \ell(\theta^*) + \frac{1}{2}(\theta - \theta^*)^T \nabla\nabla \ell(\theta^*)(\theta - \theta^*)$$

However, it's not hard to show that the expected log-likelihood is always[9] maximized by the true parameters, so $\nabla \ell(\theta^*) = 0$. (The same argument also shows $\mathbb{E}\left[\nabla L(\theta^*)\right] = 0$.) Call the Hessian in this Taylor expansion $\mathbf{k}$. (Again, notice the lower-case letter for a non-random quantity.) We have

$$\ell(\theta) \approx \ell(\theta^*) + \frac{1}{2}(\theta - \theta^*)^T \mathbf{k} (\theta - \theta^*)$$

Apply this to the MLE:

$$\ell(\hat{\theta}) \approx \ell(\theta^*) + \frac{1}{2} \nabla L(\theta^*) \mathbf{K}^{-1} \mathbf{k} \mathbf{K}^{-1} \nabla L(\theta^*)$$

Taking expectations,

$$\mathbb{E}\left[\ell(\hat{\theta})\right] \approx \ell(\theta^*) + \frac{1}{2} \operatorname{tr} \mathbf{K}^{-1} \mathbf{k} \mathbf{K}^{-1} \mathbf{J}$$

where $\operatorname{Var}\left[\nabla L(\theta^*)\right] = \mathbf{J}$. For large n, $\mathbf{K}$ converges on $\mathbf{k}$, so this simplifies to

$$\mathbb{E}\left[\ell(\hat{\theta})\right] \approx \ell(\theta^*) + \frac{1}{2} \operatorname{tr} \mathbf{k}^{-1} \mathbf{J}$$

[9] Except for quite weird models.

This still leaves things in terms of $\ell(\theta^*)$, which of course we don't know, but now we do another Taylor expansion, this time of L around $\hat{\theta}$:

$$L(\theta^*) \approx L(\hat{\theta}) + \frac{1}{2}(\theta^* - \hat{\theta})^T \nabla\nabla L(\hat{\theta})(\theta^* - \hat{\theta})$$

so

$$L(\theta^*) \approx L(\hat{\theta}) + \frac{1}{2}(\mathbf{K}^{-1}\nabla L(\theta^*))^T \nabla\nabla L(\hat{\theta})(\mathbf{K}^{-1}\nabla L(\theta^*))$$

For large n, $\nabla\nabla L(\hat{\theta}) \rightarrow \nabla\nabla L(\theta^*) \rightarrow \mathbf{k}$. So, again taking expectations,

$$\ell(\theta^*) \approx \mathbb{E}\left[L(\hat{\theta}\right] + \frac{1}{2}\operatorname{tr}\mathbf{k}^{-1}\mathbf{J}$$

Putting these together,

$$\mathbb{E}\left[\ell(\hat{\theta})\right] \approx \mathbb{E}\left[L(\hat{\theta}\right] + \operatorname{tr}\mathbf{k}^{-1}\mathbf{J}$$

An unbiased estimate is therefore

$$L(\hat{\theta}) + \operatorname{tr}\mathbf{k}^{-1}\mathbf{J}$$

Finally, a fundamental result (the "Fisher identity") says that for well-behaved models, *if* the model is correct, then

$$\operatorname{Var}\left[\nabla L(\theta^*)\right] = -\nabla\nabla\ell(\theta^*)$$

or $\mathbf{J} = -\mathbf{k}$. Hence, if the model is correct, our unbiased estimate is just

$$L(\hat{\theta}) - \operatorname{tr}\mathbf{I}$$

and of course $\operatorname{tr}\mathbf{I} = d$.

There, as you'll notice, several steps where we're making a bunch of approximations. Some of these approximations (especially those involving the Taylor expansions) can be shown to be OK asymptotically (i.e., as $n \rightarrow \infty$) by more careful math. The last steps, however, where we invoke the Fisher identity, are rather more dubious. (After all, all of the models we're working with can hardly contain the true distribution.) A somewhat more robust version of AIC is therefore to use as the criterion

$$L(\hat{\theta}) + \operatorname{tr}\mathbf{KJ}$$

4 Leave-one-out Cross-Validation (LOOCV)

When looking at influential points and outliers, we considered omitting one point from the data set, estimating the model, and then trying to predict that one data point. The **leave-one-out** fitted value for data point i is $\hat{m}_i^{(-i)}$, where the subscript $(-i)$ indicates that point i was left out in calculating this fit. The **leave-one-out cross-validation score** of the model is

$$LOOCV = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_i^{(-i)})^2$$

(Many more old-fashioned regression textbooks look at $nLOOCV$, and call it PRESS, "predictive residual sum of squares".)

The story for cross-validation is pretty compelling: we want to know if our model can generalize to new data, so *see* how well it generalizes to new data. Leaving out each point in turn ensures that that the set of points on which we try to make predictions is just as representative of the whole population as the original sample was. Fortunately, this is one of those cases where a compelling story is actually true: LOOCV is an unbiased estimate of the generalization error.

4.1 Short-cut Based on Leverage

Re-estimating the model n times would be seriously time-consuming, but there is fortunately a short-cut:

$$LOOCV = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{m}_i}{1 - H_{ii}}\right)^2$$

The numerator inside the square is just the residual of the model fit to the full data. This gets divided by $1 - H_{ii}$, which is also something we can calculate with just one fit to the model. (The denominator says that the residuals for high-leverage points count more, and those for low-leverage points count less. If the model is going out of its way to match Y_i (high leverage H_{ii}) and it still can't fit it, that's worse than the same sized residual at a point the model doesn't really care about (low leverage).)

The gap between LOOCV and the MSE can be thought of as a penalty, just like with C_p or AIC. The penalty doesn't have such a nice mathematical expression, but it's well-defined and easy for us to calculate.

It also converges to the penalty C_p applies as n grows. To help see this, first observe that the H_{ii} must be getting small. (We know that $\sum_i H_{ii} = p + 1$.) Then[10] $(1 - H_{ii})^{-2} \approx 1 - 2H_{ii}$, and

$$LOOCV \approx \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_i)^2(1 - 2H_{ii}) \approx MSE + 2\sigma^2 \operatorname{tr}\mathbf{H}$$

Cross-validation with log-likelihood The leave-one-out idea can also be applied for any model where we make a probabilistic prediction. Instead of measuring mean squared error, we measure the negative log probability density

[10] Use the binomial theorem again.

the model assigns to the actual left-out point. (Negative, so that a lower score is still better.) With Gaussian noise, this comes to the same thing as the MSE, of course.

4.2 Summing Up C_p, AIC, LOOCV

Under a very broad range of circumstances, there are theorems which say, roughly, the following:

> As $n \to \infty$, the expected out-of-sample MSE of the model picked by leave-one-out cross-validation is close to that of the best model considered.

The condition for these results do *not* require that any of the models considered be true, or that the true model have Gaussian noise or even be linear.

As we've seen, for large n leave-one-out and Mallow's C_p become extremely similar, and will pick the same model, and so will AIC, if one of the models is right. So they will also pick models which predict almost as well as the best of the models we're working with. Since C_p and AIC involve less calculation than leave-one-out, they have advantages when n is large. Against this, there don't seem to be any situations where C_p or AIC pick models with good predictive performance but leave-one-out does not. The best way to think about C_p and AIC is that they are fast approximations to the more fundamental quantity, which is leave-one-out.

On the other hand, one can *also* prove the following:

> As $n \to \infty$, if the true model is among those being compared, LOOCV, C_p and AIC will all tend to pick a *strictly larger* model than the truth.

That is, all three criteria tend to prefer models which are bigger than the true model, even when the true model is available to them. They are "not consistent for model selection".

The problem is that while these methods give unbiased estimates of the generalization error, that doesn't say anything about the variance of the estimates. Models with more parameters have higher variance, and the penalty applied by these methods isn't strong enough to overcome the chance of capitalizing on that variance.

5 Other Model Selection Criteria

While many, many other model selection criteria have been proposed, two are particularly important.

5.1 k-Fold Cross-Validation

In leave-one-out cross-validation, we omitted each data point in turn, and tried to predict it. K-fold cross-validation is somewhat different, and goes as follows.

- Randomly divide the data into k equally-sized parts, or "folds".
- For each fold
  - Temporarily hold back that fold, calling it the "testing set".
  - Call the other $k - 1$ folds, taken together, the "training set".
  - Estimate each model on the training set.
  - Calculate the MSE of each model on the testing set.
- Average MSEs over folds.

We then pick the model with the lowest MSE, averaged across testing sets.

The point of this is just like the point of leave-one-out: the models are compared only on data which they didn't get to see during estimation. Indeed, leave-one-out is the special case of k-fold cross-validation where $k = n$. The disadvantage of doing that is that in leave-one-out, all of the training sets are very similar (they share $n - 2$ data points), so averaging over folds does very little to reduce variance. For moderate k — people typically use 5 or 10 — k-fold CV tends to produce very good model selection results.

Like leave-one-out CV, k-fold cross-validation can be applied to any loss function, such as the proportion of cases mis-classified, or negative log-likelihood.

5.2 BIC

A more AIC-like criterion is the "Bayesian[11] information criterion" introduced by Schwarz (1978). The name is quite misleading[12], but irrelevant; it's got the exact same idea of penalizing the log-likelihood with the number of parameters, but using a penalty which gets bigger with n:

$$BIC(S) = L_S - \frac{\log n}{2}\dim(S)$$

This is a stronger penalty than AIC applies, and this has consequences:

> As $n \rightarrow \infty$, if the true model is among those BIC can select among, BIC will tend to pick the true model.

Of course there are various conditions attached to this, some of them quite technical, but it's generally true for IID samples, for regression modeling, for many sorts of time series model, etc. Unfortunately, the model selected by BIC will tend to predict less well than the one selected by leave-one-out cross-validation or AIC.

[11] Bayesianism is the idea that we ought to have probabilities for parameter values and for models, and not just for random variables (or, said another way, to treat parameters and models as also random variables), and update those probabilities as we see more events using Bayes's rule. It is a controversial position within statistics and philosophy of science, with many able and learned supporters, and equally able and learned opponents. (It is also the only position in statistics and philosophy of science I know of which has an online cult dedicated to promoting it, alongside reading certain works of Harry Potter fanfic, and trying not to think about the possibility a future superintelligent computer will simulate your being tortured.)

[12] The truly Bayesian position is not to *select* a model at all, but rather to maintain a probability distribution over all models you think possible.

6 Stepwise Model Selection

One way to automatically select a model is to begin with the largest model you can, and then prune it, which can be done in several ways:

- Eliminate the least-significant coefficient.
- Pick your favorite model selection criterion, consider deleting each coefficient in turn, and pick the sub-model with the best value of the criterion.

Having eliminated a variable, one then re-estimates the model, and repeats the procedure. Stop when either all the remaining coefficients are significant (under the first option), or nothing can be eliminated without worsening the criterion.

(What I've described is **backwards** stepwise model selection. **Forward** stepwise model selection starts with the intercept-only model and adds variables in the same fashion. There are, naturally, forward-backward hybrids.)

Stepwise model selection is a **greedy** procedure: it takes the move which does the most to immediately improve the criterion, without considering the consequences down the line. There are very, very few situations where it is consistent for model selection, or (in its significance-testing version) where it even does a particularly good job of coming up with predictive models, but it's surprisingly popular.

7 Inference after Selection

All of the inferential statistics we have done in earlier lectures presumed that our choice of model was completely fixed, and not at all dependent on the data. If different data sets would lead us to use different models, and our data are (partly) random, then which model we're using is also random. This leads to some extra uncertainty in, say, our estimate of the slope on X_1, which is *not* accounted for by our formulas for the sampling distributions, hypothesis tests, confidence sets, etc.

A very common response to this problem, among practitioners, is to ignore it, or at least hope it doesn't matter. This can be OK, if the data-generating distribution forces us to pick one model with very high probability, or if all of the models we might pick are very similar to each other. Otherwise, ignoring it leads to nonsense.

Here, for instance, I simulate 200 data points where the Y variable is a standard Gaussian, and there are 100 independent predictor variables, all also standard Gaussians, independent of each other *and of* Y:

```
n <- 200; p <- 100
y <- rnorm(n)
x <- matrix(rnorm(n*p),nrow=n)
df <- data.frame(y=y,x)
mdl <- lm(y~., data=df)
```

Of the 100 predictors, 5 have t-statistics which are significant at the 0.05 level or less. (The expected number would be 5.) If I select the model using just those variables[13], I get the following:

```
stars <- 1+which(coefficients(summary(mdl))[-1,4]<0.05) # Why 1+?
mdl.2 <- lm(y~., data=df[,c(1,stars)])
summary(mdl.2)

##
## Call:
## lm(formula = y ~ ., data = df[, c(1, stars)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.53035 -0.75081  0.03042  0.58347  2.63677
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.03084    0.07092   0.435   0.6641
## X21         -0.13821    0.07432  -1.860   0.0644
## X25          0.12472    0.06945   1.796   0.0741
## X41          0.13696    0.07279   1.882   0.0614
## X83         -0.03067    0.07239  -0.424   0.6722
## X88          0.14585    0.07040   2.072   0.0396
##
## Residual standard error: 0.9926 on 194 degrees of freedom
## Multiple R-squared:  0.06209,Adjusted R-squared:  0.03792
## F-statistic: 2.569 on 5 and 194 DF,  p-value: 0.02818
```

Notice that final over-all F statistic: it's testing whether including those variables fits better than an intercept-only model, and saying it thinks it does, with a definitely significant p-value. This is the case even though, by construction, the response is *completely independent* of *all* predictors. This is not a fluke: if you re-run my simulation many times, your p-values in the full F test will not be uniformly distributed (as they would be on all 100 predictors), but rather will have a distribution strongly shifted over to the left. Similarly, if we looked at the confidence intervals, they would be much too narrow.

These issues do not go away if the true model isn't "everything is independent of everything else", but rather has some structure. Because we picked the model to predict well on this data, if we then run hypothesis tests on that same data, they'll be too likely to tell us everything is significant, and our confidence intervals will be too narrow. Doing statistical inference on the same data we used to select our model is just broken. It may not always be as spectacularly broken as in my demo above, but it's still broken.

There are three ways around this. One is to pretend the issue doesn't exist;

[13] Exercise: Explain all the ways in which this is a bad idea. Now imagine explaining the same thing to your boss, who took econometrics 20 years ago, and wants to know why he can't just follow the stars.

as I said, this is popular, but it's got nothing else to recommend it. Another, which is an area of very active research currently in statistics, is to try to come up with clever technical adjustments to the inferential statistics[14]. The third approach, which is in many ways the simplest, is to use *different data sets* to select a model and to do inference within the selected model[15].

7.1 Data Splitting

Data splitting is (for regression) a very simple procedure:

- Randomly divide your data set into two parts.
- Calculate your favorite model selection criterion for all your candidate models using only the first part of the data. Pick one model as the winner.
- Re-estimate the winner, and calculate all your inferential statistics, using only the other half of the data.

(Division into two equal halves is optional, but usual.)

Because the winning model is statistically independent of the second half of the data, the confidence intervals, hypothesis tests, etc., can treat it as though that model were fixed *a priori*. Since we're only using $n/2$ data points to calculate confidence intervals (or whatever), they will be somewhat wider than if we really had fixed the model in advance and used all n data points, but that's the price we pay for having to select a model based on data.

8 R Practicalities

R^2 and adjusted R^2 are calculated by the `summary` function for `lm` objects, if — Heaven forbid – you should ever need them. So, more practically, is the in-sample root mean squared error, using the unbiased estimator:

```
mdl <- lm(something ~ other_things, data=df)
summary(mdl)$r.squared
summary(mdl)$adj.r.squared
summary(mdl)$sigma
```

The un-adjusted MSE is also easily calculated:

```
mean(residuals(mdl)^2)
```

The `AIC` function knows how to work with models produced by `lm`; it uses an alternate definition of AIC which is $-2\times$ the one I gave above (so smaller AIC is preferred). Similarly for the `BIC` function.

[14] If you're curious, ask Profs. Tibshirani or G'Sell about this.

[15] Technically, there is a fourth possible approach, which is to select the model completely at random, and then do inference within it. This may sound like a joke, but there are actually situations, like testing for a difference in means between high-dimensional vectors, where it's perfectly reasonable.

The `step` function will do stepwise model selection based on AIC, either forward or backward. Manipulating the arguments also allows for doing BIC. (See the help file.) *Warning:* By default this prints out a lot of information about every model it looks at; consider setting `trace=0`.

For leave-one-out cross-validation, the most straightforward approach is to use the following function:

```
# Calculate LOOCV score for a linear model
# Input: a model as fit by lm()
# Output: leave-one-out CV score
cv.lm <- function(mdl) {
    return(mean((residuals(mdl)/(1-hatvalues(mdl)))^2))
}
```

For k-fold cross-validation, the easiest option at this stage is to use the `cv.glm` function in the package `boot`[16]. Note that this requires you to fit your model with the `glm` function, not with `lm`, and that you will really only be interested in the `delta` component of what `cv.glm` returns. (See the help file, especially the examples at the end.)

Nobody seems to have written a function for calculating C_p. Here is one.

```
# Calculate Mallow's Cp for a list of linear models
# Input: List of models, all fit by lm
# Output: Vector of Cp statistics
# Presumes: All models are nested inside the largest model; all models
  # fit on a common data set
Cp.lm <- function(mdl.list) {
    # How many samples do we have?
      # Presumes all models fit to the same data
    n <- nobs(mdl.list[[1]])
    # Extract the number of degrees of freedom for each model
    DoFs <- sapply(mdl.list, function(mdl) { sum(hatvalues(mdl)) })
    # Extract the MSEs of each model
    MSEs <- sapply(mdl.list, function(mdl) { mean(residuals(mdl)^2) })
    # Which model had the most parameters?
      # Presuming that model includes all the others as special cases
    biggest <- which.max(DoFs)
    # Use the nesting model's MSE to estimate sigma^2
    sigma2.hat <- MSEs[[biggest]]*n/(n-DoFs[[biggest]])
    Cp <- MSEs + 2*sigma2.hat*DoFs/n
    return(Cp)
}


# Example of usage:
Cp.lm(list(mdl1, mdl2, mdl3))
```

[16] Later in this course and in 402, we will write our own CV code, partly as character building and partly because there's nothing quite like doing this to actually get how it works.

8.1 A Demo

We'll do polynomial regression with just one X variable; this way we can keep throwing in as many terms as we need to, in order to make the point. X will be uniformly distributed on the interval $[-2, 2]$, and when we use a q^{th} order polynomial, we'll set

$$Y = \sum_{i=1}^{q} (-1)^q X^q + \epsilon$$

with ϵ having our usual Gaussian distribution with mean 0 and standard deviation 0.1.

Here's code to simulate from the model:

```r
# Simulate variable-degree polynomial with fixed X and coefficients
# Inputs: Number of points to simulate; degree of polynomial
# Output: Data from with x and y columns
sim.poly <- function(n, degree) {
    x <- runif(n, min=-2, max=2)
    poly.x <- poly(x, degree=degree, raw=TRUE)
    alternating.signs <- rep(c(-1,1),length.out=degree)
    sum.poly <- poly.x %*% alternating.signs
    y <- x+rnorm(n,0,0.1)
    return(data.frame(x=x,y=y))
}
```

And here is code to fit many polynomials to it:

```r
# Fit multiple univariate polynomials to the same data
# Input: data frame; maximum degree of polynomial
# Output: Liist of estimated models
# Presumes: data frame has columns called x and y; y is response; maximum
  # degree is an integer >= 1.
poly.fit <- function(df, max.degree) {
    lapply(1:max.degree, function(deg) { lm(y ~ poly(x, degree=deg), data=df) })
}
```

And to apply multiple selection criteria to a list of models:

```r
# Apply multiply model selection criteria to a list of models
# Inputs: list of models
# Outputs: Vector, indicating which model from the list was picked by
  # each criterion
# Presumes: all models are set up to work with all criteria functions applied
    # True if all models were fit by lm()
  # All models fit on same data set (otherwise, weird)
selectors <- function(mdl.list) {
    Rsq <- which.max(sapply(mdl.list, function(mdl) { summary(mdl)$r.sq }))
    Rsq.adj <-  which.max(sapply(mdl.list, function(mdl) { summary(mdl)$adj.r.sq }))
    Cp <- which.min(Cp.lm(mdl.list))
    LOOCV <- which.min(sapply(mdl.list, cv.lm))
    AIC <- which.min(sapply(mdl.list, AIC))
    BIC <- which.min(sapply(mdl.list, BIC))
    choices <- c(Rsq = Rsq, Rsq.adj = Rsq.adj, Cp=Cp, LOOCV=LOOCV,
                 AIC = AIC, BIC=BIC)
    return(choices)
}
```

To put this all together, let's see what gets picked if we simulate 20 data points from the quadratic, and allow models of up to order 10:

```
selectors(poly.fit(sim.poly(n=20, degree=2), max.degree=10))
```

```
##     Rsq Rsq.adj      Cp   LOOCV     AIC     BIC
##      10       9       1       1       1       1
```

Of course, one run doesn't mean much, so let's do this a bunch of times:

```
summary(t(replicate(1000,
                    selectors(poly.fit(sim.poly(n=20, degree=2),
                                       max.degree=10)))))
```

```
##       Rsq        Rsq.adj             Cp             LOOCV
##  Min.   :10   Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
##  1st Qu.:10   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 1.000
##  Median :10   Median : 6.000   Median : 1.000   Median : 1.000
##  Mean   :10   Mean   : 5.483   Mean   : 2.656   Mean   : 1.873
##  3rd Qu.:10   3rd Qu.: 9.000   3rd Qu.: 3.000   3rd Qu.: 2.000
##  Max.   :10   Max.   :10.000   Max.   :10.000   Max.   :10.000
##       AIC              BIC
##  Min.   : 1.000   Min.   : 1.000
##  1st Qu.: 1.000   1st Qu.: 1.000
##  Median : 2.000   Median : 1.000
##  Mean   : 3.912   Mean   : 2.044
##  3rd Qu.: 7.000   3rd Qu.: 2.000
##  Max.   :10.000   Max.   :10.000
```

This is showing us the summary statistics for the degree of the polynomial model selected according to each criteria. (Why do I put in the transpose?) Remember that the right degree here is 2, so R^2 is (as usual) useless, and adjusted R^2 little better. The others all at least do something roughly right, though AIC is worse than the other three.

Of course, $n = 20$ isn't very much information[17], so let's increase that to $n = 1000$.

```
summary(t(replicate(1000,
                    selectors(poly.fit(sim.poly(n=1000, degree=2),
                                       max.degree=10)))))
```

[17] Though it seems to be enough for leave-one-out or BIC.

```
##       Rsq        Rsq.adj           Cp             LOOCV
##  Min.   :10   Min.   : 1.0   Min.   : 1.000   Min.   : 1.000
##  1st Qu.:10   1st Qu.: 1.0   1st Qu.: 1.000   1st Qu.: 1.000
##  Median :10   Median : 4.0   Median : 1.000   Median : 1.000
##  Mean   :10   Mean   : 4.6   Mean   : 1.713   Mean   : 1.719
##  3rd Qu.:10   3rd Qu.: 8.0   3rd Qu.: 2.000   3rd Qu.: 2.000
##  Max.   :10   Max.   :10.0   Max.   :10.000   Max.   :10.000
##       AIC              BIC
##  Min.   : 1.000   Min.   :1.000
##  1st Qu.: 1.000   1st Qu.:1.000
##  Median : 1.000   Median :1.000
##  Mean   : 1.719   Mean   :1.015
##  3rd Qu.: 2.000   3rd Qu.:1.000
##  Max.   :10.000   Max.   :2.000
```

Adjusted R^2 is hopeless, leave-one-out does essentially the same as AIC or C_p (and all have a 25% probability of picking a model which is too big), and BIC is, as expected, much more conservative.

You can experiment with seeing what happens if you change the true order of the model, or the range of orders compared by the model selectors, or make some of the higher-order polynomial terms close to but not quite zero, etc., etc.

9 Further Reading

The best reference on model selection I know of, by far, is Claeskens and Hjort (2008); unfortunately, much of the theory is beyond the level of this course, but some of the earlier chapters should not be. Hansen (2005) provides interesting perspectives based on extensive experience in econometrics.

Cross-validation goes back in statistics into the 1950s, if not earlier, but did not become formalized as a tool until the 1970s, with the work of Stone (1974), Geisser (1975) and Geisser and Eddy (1979). (The last paper, written in 1977, made it perfectly clear the approach could be used on log-likelihood, mis-classification rates, etc., as well as squared error.) It was adopted, along with many other statistical ideas, by computer scientists during the period in the late 1980s–early 1990s when the modern area of "machine learning" emerged from (parts of) earlier areas called "artificial intelligence", "pattern recognition", "connectionism", "neural networks", or indeed "machine learning". Subsequently, many of the scientific descendants of the early machine learners forgot where their ideas came from, to the point where many people now think cross-validation is something computer science contributed to data analysis. For a recent survey of cross-validation techniques and their uses, see Arlot and Celisse (2010).

For theoretical results on model selection by cross-validation, and on data splitting, see Györfi *et al.* (2002).

This lecture has emphasized model selection criteria which could be applied automatically. Of course, doing anything automatically is usually somewhat dubious. An alternative, with a lot to recommend it, is to very carefully construct models, test their implications, and gradually move towards more complicated models as improvements in data (volume, precision of measurement, range of variables, etc.) show definite problems with simpler models (Gelman and Shalizi, 2013).

10 Exercises

To think through or practice on, not to hand in.

1. Show that $\mathrm{Cov}[Y_i, \hat{m}_i] = \sigma^2 H_{ii}$. (*Hint:* Write $\hat{m}_i$ as a weighted sum of Y_j.)
2. Using the `step` function, repeat the simulation from §7 and report the number of selected coefficients, their median p-value (in Wald tests of the slope being zero), and the p-value of the full F-test. Repeat the simulation many times, and plot a histogram of the F-test p-values.

References

Akaike, Hirotugu (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In *Proceedings of the Scond International Symposium on Information Theory* (B. N. Petrov and F. Caski, eds.), pp. 267–281. Budapest: Akademiai Kiado. Reprinted in (Akaike, 1998, pp. 199–213).

— (1998). *Selected Papers of Hirotugu Akaike*. Berlin: Springer-Verlag. Edited by Emanuel Parzen, Kunio Tanabe and Genshiro Kitagawa.

Arlot, Sylvain and Alain Celisse (2010). "A survey of cross-validation procedures for model selection." *Statistics Surveys*, **4**: 40–79. URL `http://projecteuclid.org/euclid.ssu/1268143839`.

Claeskens, Gerda and Nils Lid Hjort (2008). *Model Selection and Model Averaging*. Cambridge, England: Cambridge University Press.

Geisser, Seymour (1975). "The Predictive Sample Reuse Method with Applications." *Journal of the American Statistical Association*, **70**: 320–328.

Geisser, Seymour and William F. Eddy (1979). "A Predictive Approach to Model Selection." *Journal of the American Statistical Association*, **74**: 153–160. doi:10.1080/01621459.1979.10481632.

Gelman, Andrew and Cosma Rohilla Shalizi (2013). "Philosophy and the Practice of Bayesian Statistics." *British Journal of Mathematical and Statistical Psychology*, **66**: 8–38. URL `http://arxiv.org/abs/1006.3868`. doi:10.1111/j.2044-8317.2011.02037.x.

Györfi, László, Michael Kohler, Adam Krzyżak and Harro Walk (2002). *A Distribution-Free Theory of Nonparametric Regression*. Springer Series in Statistics. New York: Springer-Verlag.

Hansen, Bruce E. (2005). "Challenges for Econometric Model Selection." *Econometric Theory*, **21**: 60–68. URL `http://www.ssc.wisc.edu/~bhansen/papers/et_05.pdf`. doi:10.10170/S0266466605050048.

Schwarz, Gideon (1978). "Estimating the Dimension of a Model." *Annals of Statistics*, **6**: 461–464. URL `http://projecteuclid.org/euclid.aos/1176344136`.

Stone, M. (1974). "Cross-validatory choice and assessment of statistical predictions." *Journal of the Royal Statistical Society B*, **36**: 111–147. URL `http://www.jstor.org/stable/2984809`.