Lecture 5: Model selection and assessment

Hector Corrada Bravo and Rafael A. Irizarry

February, 2010

Preamble

Before we begin this section, we introduce subset selection for linear regression models.

Subset Selection

Although the least squares estimate is the linear unbiased estimate with minimum variance, it is possible that a biased estimate will give us a better mean squared error.

Consider a case where the true model is

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

and that X_1 and X_2 are almost perfectly correlated (statisticians say X_1 and X_2 are co-linear). What happens if we leave X_2 out?

Then the model is very well approximated by

$$Y = \beta_0 + (\beta_1 + \beta_2) X_1 + \epsilon$$

and we may get a good estimate of Y estimating 2 parameters instead of 3. Our estimate will be a bit biased but we may lower our variance considerably creating an estimate with smaller **expected prediciton error** than the least squares estimate.

We won't be able to interpret the estimated parameter, but our prediction may be good.

In subset selection regression we select a number of covariates to include in the model. Then we look at all possible combinations of covariates and pick the one with the smallest RSS.

Consider the prostate cancer data set presented in the HTF book, available in the `ElemStatLearn` R package (Figure 1). Notice that residual sum of squares

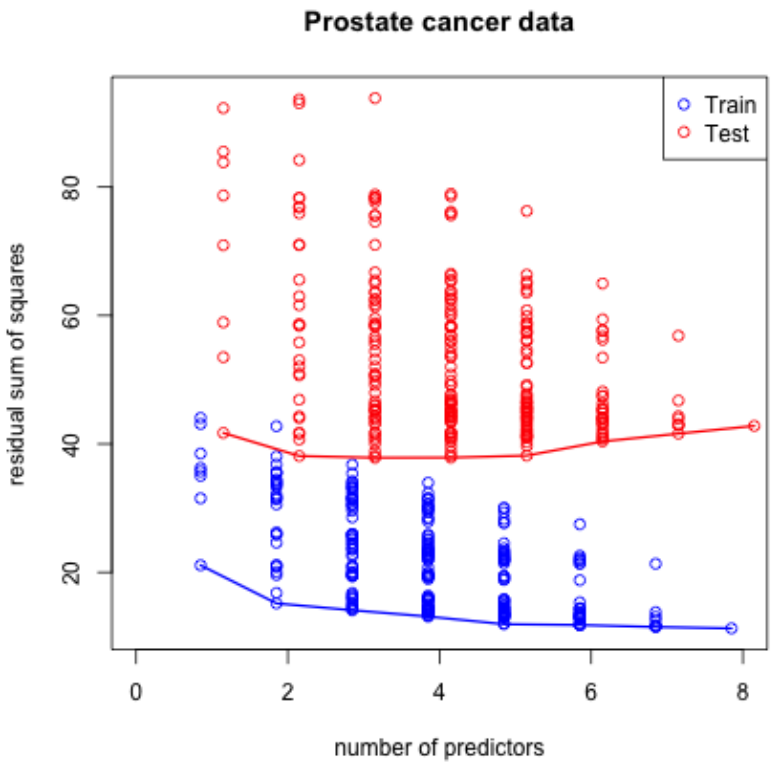

Figure 1: Prediction error (RSS) for all possible subset models for training and test sets for prostate cancer data. The solid lines denote the minimums.

consistently drops for the training set as larger models are used, whereas smaller models tend to do better for test set residual sum of squares.

For a given number of predictors, how do we find the model that gives the smallest RSS? There are algorithms that do this, but you do not really want to use this. We will describe penalty and shrinking methods that work better later on.

How do we choose the number of covariates to include? That's a bit harder, and the subject of this Section.

Overview

Each of the methods we have described so far can be parameterized in a way corresponding to *model complexity*. For instance, k-nearest neighbors is parameterized by k; subset selection in regression is parameterized by the number of predictors in the model; trees can be parameterized by the size penalty parameter α.

In classical statistical theory we usually assume that the underlying model generating the data is in the family of models we are considering. In this case bias is not an issue and efficiency (low variance) is all that matters. Much of the theory in classical statistics is geared toward finding efficient estimators.

In this course we try not make the above assumptions. Furthermore, for the techniques we have shown (and will show) asymptotic and finite sample bias and variance estimates are not always easy (many times impossible) to find in closed form. In this Chapter we discuss in-sample approximation, and resampling methods that are commonly used to get estimates of prediction error.

Remember that the main difficulty with model assessment and selection is that the observed prediction error for training data becomes smaller with model complexity regardless of the prediction ability on the test data. See figure 1 again.

In this Chapter we will look at choosing the complexity parameter with the hope of improving *expected prediction error*. For all methods we can think of an estimate $\hat{f}(x)$ (for example, in regression for k-nn it is the average outcome in the neighborhood of size k around x, in linear regression it is given by the linear function $\beta_0 + \beta' x$, and in regression trees by the average outcome in the partition where x falls). Furthermore, for all of these methods we will index the estimate with parameter λ, (e.g. $\hat{f}_\lambda$) and assume from the context what λ refers to (k in k-nn, tree size in tree methods, subset size in linear regression).

Model Selection and Assessment

Typically there are two parts to solving a prediction problem: model selection and model assessment. In model selection we estimate the performance of var-

ious competing models with the hope of choosing the best one. Having chosen the final model, we assess the model by estimating the prediction error on new data.

Remember that the best model is defined as the one with the lowest EPE:

$$\text{EPE}(\lambda) = E[L\{Y - \hat{f}_\lambda(X)\}]$$

Where Y and X are drawn at random from the population and the expectation averages anything that is random.

Typical loss function are squared error, $L(Y, \hat{f}(X)) = (Y - \hat{f}(X))^2$, and absolute error, $L(Y, \hat{f}(X)) = |Y - \hat{f}(X)|$.

We define training error as the observed average loss

$$\frac{1}{N}\sum_{i=1}^{N} L\{y_i, \hat{f}(x_i)\}$$

With squared error loss this is the residual sum of squares divided by N, which we will call the Average Squared Error (ASE).

For categorical data, using square loss doesn't make much sense. Typical loss functions are 0–1 loss, $L(G, \hat{G}(X)) = 0$ if $G = \hat{G}(X)$, 0 otherwise, and the log-likelihood: $L(G, \hat{G}(X)) = -2\sum_{k=1}^{K} I(G = k)\log \hat{p}_k(X) = -2\log \hat{p}_G(X)$. The latter is also called *cross-entropy*. Notice the –2 is used so that for normal error it is becomes equivalent to the loss function.

The training errors are obtained as in the continuous example. For 0–1 loss it is simple the percentage of times we are wrong in the training data. For the likelihood loss we simply use the observed log-likelihood times –2/N:

$$-\frac{2}{N}\sum_{i=1}^{N} \log \hat{p}_{g_i}(x_i)$$

As we have discussed various times, the training error underestimates the test error or EPE. In today's lectures we describe ways of getting better estimates of EPE.

Split Samples

When the amount of data and computation time permits it, there is no method better than data splitting. The idea is simple: Divide the data in three parts: train, validation, and test. We use the train and validation data to select the best model and the test data to assess the chosen model.

The recipe is the following:

1. In the first part, model selection, the validation model is treated as the test data. We train all competing model on the train data and define the best model as the one that predicts best in the validation set. We could re-split the train/validation data, do this many times, and select the method that, on average, best performs.

2. Because we chose the best model among many competitors, the observed performance will be a bit biased. Therefore, to appropriately assess performance on independent data we look at the performance on the test set.

3. Finally, we can re-split everything many times and obtain average results from steps 1) and 2).

There is no obvious choice on how to split the data. It depends on the signal to noise ratio which we, of course, do not know. A common choice is 1/2, 1/4, and 1/4.

There are two common problems:

1. When the amount of data is limited, the results from fitting a model to 1/2 the data can be substantially different to fitting to all the data. An extreme example: We have 12 data points and want to consider a regression model with 7 parameters.

2. Model fitting might have high computational requirements.

In this Chapter we describe some *in-sample* methods for model selection as well as less biased split sample methods.

Bias-Variance trade-off

We want to estimate f and assume our data comes from the following model:

$$Y_i = f(X_i) + \epsilon_i$$

with the ϵ IID, independent of X, and variance σ^2.

Suppose we are using k-nearest neighbors and want to decide what is the best neighborhood size λ (we've been calling this k, but in this section we use λ to index complexity parameters).

To quantify "best", we say it is the λ that minimizes the expected prediction error:

$$\text{EPE}(\lambda) = E[\{Y - \hat{f}_\lambda(X)\}^2] \tag{1}$$

Where, as mentioned, Y and X are drawn at random from the population and the expectation averages anything that is random.

The above is better understood in the following way. Let $\hat{f}_\lambda$ be the estimate obtained with the training data. Now, imagine that we get a completely independent data point. Let's simplify by assuming $X = x^*$ is fixed. So what we are looking to minimize is simply

$$\mathrm{E}[\{Y^* - \hat{f}_\lambda(x^*)\}^2]$$

This can be broken up into the following pieces.

$$\begin{aligned} \mathrm{Err}(x_0) &= \sigma^2 + \{\mathrm{E}[\hat{f}_\lambda(x^*)] - f(x^*)\}^2 + \mathrm{var}[\hat{f}_\lambda(x_0)] && (2) \\ &= \text{Irreducible error} + \text{Bias}^2 + \text{Variance} && (3) \end{aligned}$$

The first term is due to unpredictable measurement error. There is nothing we can do about it. The second term is bias of the estimator (squared) and the last term is the estimator's variance.

Notice that the above calculation can be done because the Y_i^*s are independent of the estimates $\hat{f}_\lambda(x_i)$s, the same can't be said about the Y_is.

In general, we want to pick a λ that performs well for all x. If instead of just one new point we obtain N then we would have

$$\frac{1}{N}\sum_{i=1}^{N} \mathrm{E}[Y_i^* - \hat{f}_\lambda(x_i^*)] = \sigma^2 + \{\mathrm{E}[\hat{f}_\lambda(x_0)] - f(x_0)\}^2 + \mathrm{var}[\hat{f}_\lambda]$$

If we instead assume X is random we can use expectations instead of averages and we are back to our original equation (1).

For k-nearest neighbors, these expressions have a simple form

$$\begin{aligned} \mathrm{Err}(x_0) &= \mathrm{E}[(Y - \hat{f}_\lambda)^2 | X = x_0] && (4) \\ &= \sigma^2 + \left[f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_l) \right]^2 + \frac{\sigma^2}{k} && (5) \end{aligned}$$

If k is small, then the estimated function can best adapt to the underlying true function. On the other hand, as k increases, bias can increase, but there is a reduction in variance.

For subset selection in linear regression $\hat{f}_\lambda(x_0) = \beta' x$ where the parameter vector $\hat{\beta}$ with $\lambda = p$ components is fit with least-squares, we have

$$\mathrm{Err}(x_0) = \mathrm{E}[(Y - \hat{f}_\lambda)^2 | X = x_0] \tag{6}$$

$$\sigma^2 + \left[f(x_0) - \mathrm{E}\hat{f}(x_i) \right]^2 + \|\mathbf{h}(x_0)\|^2 \sigma^2 \tag{7}$$

Here $\mathbf{h}(x_0) = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} x_0$, the N-vector of linear weights that produces the fit $\hat{f}_\lambda = x_0'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, and hence $\mathrm{var}[\hat{f}_\lambda(x_0)] = \|\mathbf{h}(x_0)\|^2\sigma^2$. While this variance changes with x_0, its average is $(p/N)\sigma^2$.

For N new points, we would have

$$\frac{1}{N}\sum_{i=1}^{N} \mathrm{Err}(\mathrm{x_i}) = \sigma^2 + \frac{1}{N}\sum_{i=1}^{N} [f(x_i) - \mathrm{E}\hat{f}(x_i)]^2 + \frac{p}{N}\sigma^2$$

Here model complexity is directly related to the number of parameters p.

Sidebar: Ridge Regression

By only considering some of the covariates we were able to improve our prediction error. However, the choice of one covariate over an another can sometimes be a very arbitrary decision as including either works well but both together do no work as well (this happens often with correlated predictors).

Notice that in subset-selection for linear regression, we are estimating models of the form $\hat{f}_\lambda(x) = x'\beta$ where β is constrained to have exactly (say λ) non-zero βs. The selection problem is, having chosen λ, select which $p - \lambda$ coefficients β will be exactly zero.

Thus, we can think of the subset selection procedure as one that *shrinks* some of the coefficients to 0. But what if we do this in a smoother way? Instead of either keeping it (multiply by 1) or not (multiply by 0), let's permit smoother shrinkage (multiply by a number between 0 and 1).

For ridge regression instead of minimizing least squares we *penalize* for having too many β that are big by considering the following minimization criteria:

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2.$$

We will denote the parameter vector that minimizes this $\hat{\boldsymbol{\beta}}^{\mathrm{ridge}}$. Here *complexity parameter* λ is a penalty. We saw a similar penalty parameter in tree-based methods.

One can demonstrate mathematically that minimizing the above expression is equivalent to minimizing the regular RSS

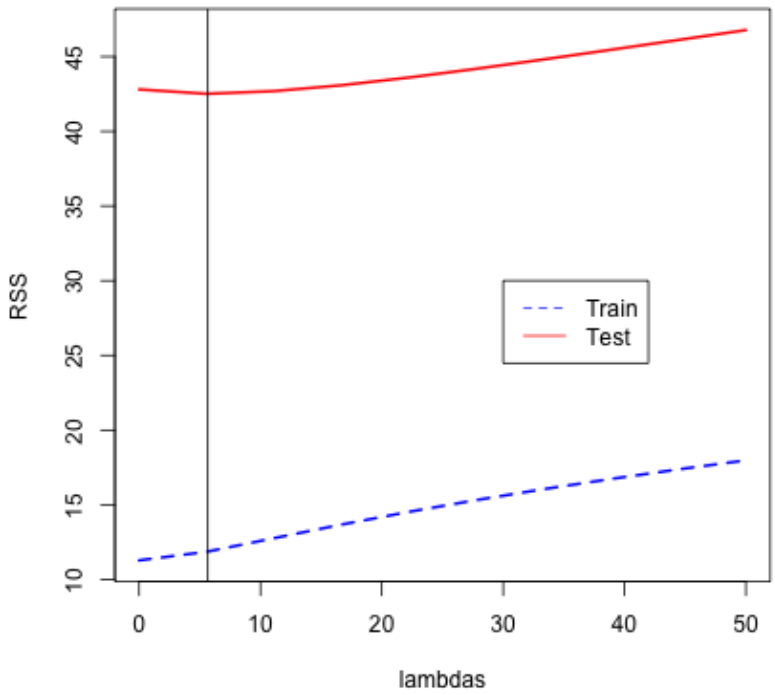

Figure 2: Prediction error (RSS) for ridge regression with varying penalty parameters

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{ subject to } \sum_{j=1}^{p}\beta_j^2 \leq s$$

where s is inversely proportional to lambda.

Notice that when λ is 0, we get the least squares estimate. However, as λ gets bigger, over fitting gets more expensive as larger values of β penalize the criterion more. The smallest penalty occurs when all the βs are 0. This gives us an estimate with small variance but likely large bias.

Although this problems looks complicated it turns out the resulting predictor is a linear estimate!

One can show that the solution is (in linear algebra notation)

$$\hat{\boldsymbol{\beta}}^{\text{ridge}} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y}$$

As with subset-selection, we can write the estimated $\hat{f}_\lambda = \mathbf{S}_\lambda\mathbf{y}$, using again a *hat* matrix. Later on we'll start calling these *smoother* matrices.

In Figure we see the RSS in a test and training set for the prostate cancer data for various values of λ.

As expected the RSS in the training set is best when $\lambda = 0$ (no shrinkage, nothing stopping us from over-fitting). However, for the training set the smallest RSS occurs for $\lambda \approx 5$

The least squares estimates are given below. Notice age has a significant protective effect. This is at odds with out intuition.

	Est	SEt	Pr(>|t|)	
(Intercept)	-0.10	1.42	0.9434	
lcavol	0.59	0.10	9.58e-07	***
lweight	0.73	0.28	0.0160	*
age	-0.03	0.01	0.0257	*
lbph	0.18	0.07	0.0244	*
svi	0.50	0.32	0.1329	
lcp	-0.16	0.10	0.1299	
gleason	0.07	0.17	0.6983	
pgg45	0.01	0.004	0.1199	

Ridge regression shrinks the regression coefficients toward 0. Notice what happens to these coefficients as λ grows. Notice in particular what happens to age.

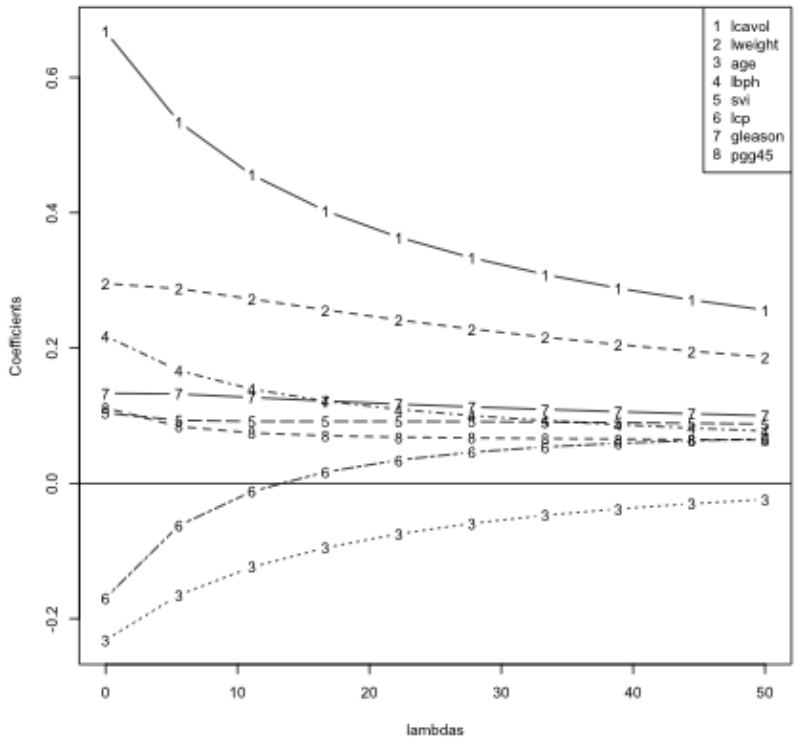

Figure 3: Estimated coefficients using ridge regression with various penalty parameters.

SVD, PCA and ridge regression

The singular value decomposition (SVD) of the centered input matrix $\mathbf{X}$ gives us insight into the nature of ridge regression.

This decomposition is extremely useful in many statistical analysis methods. We will see it again later.

The SVD of an $N \times p$ matrix $\mathbf{X}$ is

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}'$$

with $\mathbf{U}$ and $\mathbf{V}$ $N \times p$ and $p \times p$ orthogonal matrices and $\mathbf{D}$ a $p \times p$ diagonal matrix with entries $d_1 \geq d_2 \geq \ldots d_p \geq 0$ called the singular values of $\mathbf{X}$.

Technical Note: $\mathbf{U}$ is an orthogonal basis for the space defined by the columns of $\mathbf{X}$ and $\mathbf{V}$ is an orthogonal basis for the space defined by the rows of $\mathbf{X}$.

We can show that the least squares predictor for linear regression is

$$\begin{aligned} \hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}^{ls} &= \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \\ &= \mathbf{U}\mathbf{U}'\mathbf{y} \end{aligned}$$

Technical Note: $\mathbf{U}'\mathbf{y}$ are the coordinates of $\mathbf{y}$ with respect to the orthogonal basis $\mathbf{U}$

The ridge solution can be expressed as

$$\begin{aligned} \mathbf{X}\hat{\boldsymbol{\beta}}^{\text{ridge}} &= \mathbf{X}(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'y \\ &= \mathbf{U}\mathbf{D}(\mathbf{D}^2 + \lambda I)^{-1}\mathbf{D}\mathbf{U}'y \\ &= \sum_{j=1}^{p} \mathbf{u}_j \frac{d_j}{d_j + \lambda}\mathbf{u}_j'\mathbf{y} \end{aligned}$$

Notice that because $\lambda > 0$, $\frac{d_j}{d_j+\lambda} \leq 1$. Like linear regression, ridge regression computes the coordinates of $\mathbf{y}$ with respect to the orthogonal basis $\mathbf{U}$. It then shrinks these coordinates by the factors $\frac{d_j}{d_j+\lambda}$. This means that a greater amount of shrinkage occurs when λ is big and for smaller d_js.

What does having a small d_j represent? A way to understand this is by looking at the *principal components* of the variables in $\mathbf{X}$.

If the $\mathbf{X}$ are centered, the sample covariance matrix is given by $\mathbf{X}'\mathbf{X}/N$ and $\mathbf{X}'\mathbf{X}$ can be written as

$$\mathbf{X}'\mathbf{X} = \mathbf{V}\mathbf{D}^2\mathbf{V}.$$

Technical note: this is the eigen decomposition of $\mathbf{X}'\mathbf{X}$.

The v_js are called the eigen values and also the principal components directions of $\mathbf{X}$. Figure 4 shows a scatterplot of $\mathbf{X}$ and the directions as red (solid) and blue (dashed) lines.

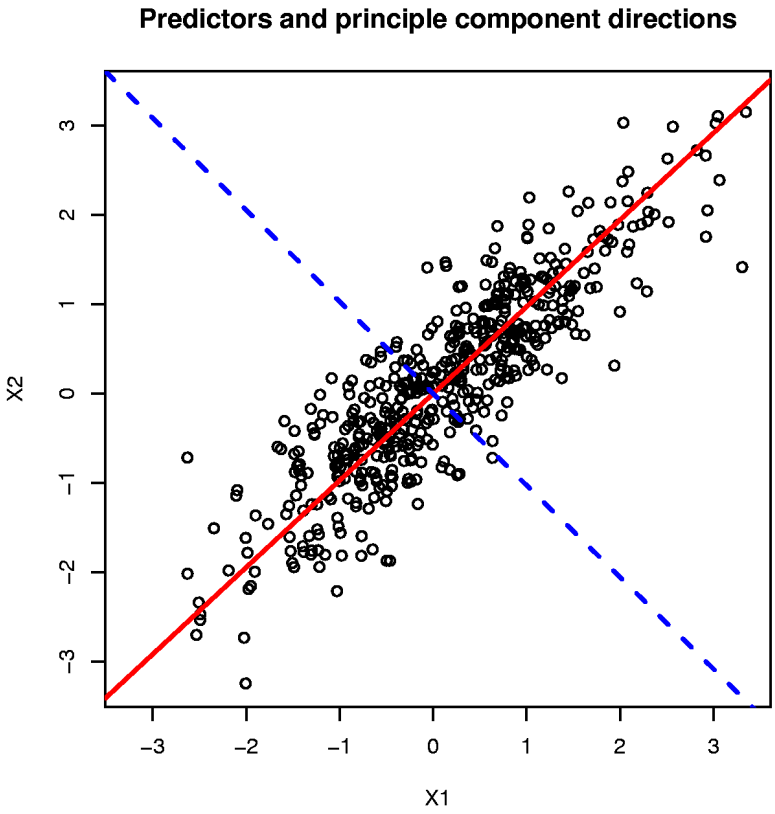

Figure 4: Plot of two predictors, X_2 versus X_1, and the principal component directions

The first principal component $\mathbf{z}_1 = \mathbf{X}v_1$ has the property that it has the largest sample covariance among all normalized (coefficients squared add up to 1) linear combinations of $\mathbf{X}$. The sample variance is d_1^2/N.

The derived variable $\mathbf{z}_1 = \mathbf{X}v_1 = \mathbf{u}_1 d_1$ is called the first principal component. Similarly $\mathbf{z}_j = \mathbf{X}v_1$ is called the jth principal component. $\mathbf{XV} = \mathbf{UD}$ is a matrix with principal components in the columns. Figure 5 shows these.

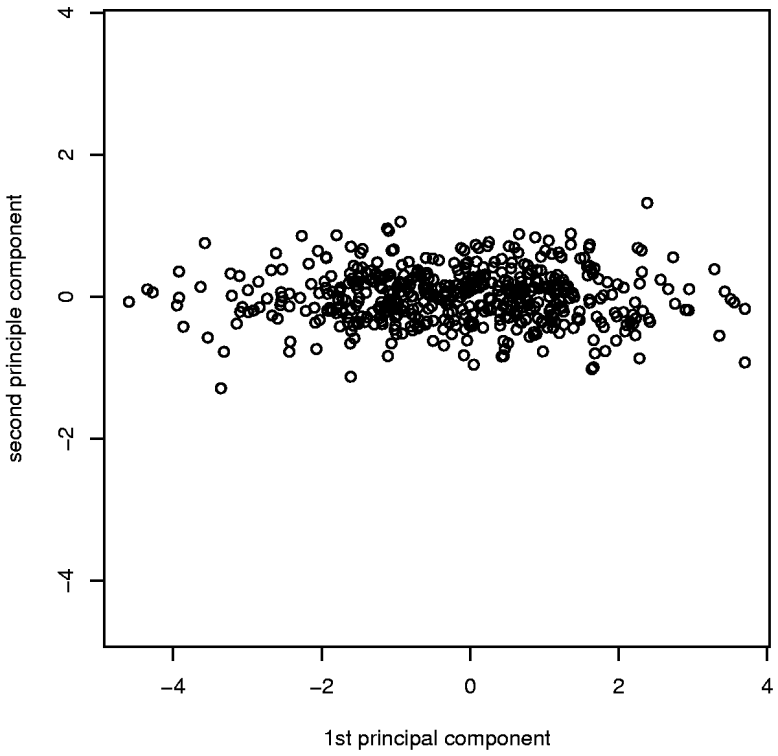

Figure 5: Plot of two principal components of $\mathbf{X}$.

We now see that ridge regression shrinks coefficients related to principal components with small variance. This makes sense because we have less information about them.

In the case of Figure 4, we can think of it as weight and height, we are saying predict with the sum and ignore the difference. In this case, the sum give much more info than the difference.

Bias-variance trade-off linear models

Define $\mathbf{S}_\lambda$ as the hat matrix for a particular linear model parameterized by λ. The estimated $\hat{f}$s will be written as $\hat{\mathbf{f}}_\lambda = \mathbf{S}_\lambda \mathbf{y}$.

Define

$$\mathbf{v}_\lambda = \mathbf{f} - \mathrm{E}(\mathbf{S}_\lambda \mathbf{y})$$

as the *bias* vector.

Define $\mathrm{ave}(\mathbf{x}^2) = n^{-1} \sum_{i=1}^n x_i^2$ for any vector $\mathbf{x}$. We can derive the following formulas:

$$\begin{aligned} \mathrm{MSE}(\lambda) &= n^{-1} \sum_{i=1}^n \mathrm{var}\{\hat{f}_\lambda(x_i)\} + \mathrm{ave}(\mathbf{v}_\lambda^2) \\ &= n^{-1} \mathrm{tr}(\mathbf{S}_\lambda \mathbf{S}_\lambda') \sigma^2 + n^{-1} \mathbf{v}_\lambda' \mathbf{v}_\lambda \\ \mathrm{EPE}(\lambda) &= \{1 + n^{-1} \mathrm{tr}(\mathbf{S}_\lambda \mathbf{S}_\lambda')\} \sigma^2 + n^{-1} \mathbf{v}_\lambda' \mathbf{v}_\lambda. \end{aligned}$$

Notice for least-squares regression $\mathbf{S}_\lambda$ is idempotent so that $\mathrm{tr}(\mathbf{S}_\lambda \mathbf{S}_\lambda') = \mathrm{tr}(\mathbf{S}_\lambda) = \mathrm{rank}(\mathbf{S}_\lambda)$ which is usually the number of parameters in the model. Later on, we will sometimes refer to $\mathrm{tr}(\mathbf{S}_\lambda \mathbf{S}_\lambda')$ as the *equivalent number of parameters* or degrees of freedom of our estimator.

Cross-validation: choosing complexity parameters

In the section, and the rest of the class, we will denote with $\hat{f}_\lambda$ the estimate obtained using complexity parameter λ.

In practice it is not common to have a new set of data $y_i^*, i = 1, \ldots, n$. Cross-validation tries to imitate this by leaving out points (x_i, y_i) one at a time and estimating the smooth at x_i based on the remaining $n - 1$ points. The cross-validation sum of squares is

$$\mathrm{CV}(\lambda) = n^{-1} \sum_{i=1}^n \{y_i - \hat{f}_\lambda^{-i}(x_i)\}^2$$

where $\hat{f}_\lambda^{-i}(x_i)$ indicates the fit at x_i computed by leaving out the $i - th$ point.

We can now use CV to choose λ by considering a wide span of values of λ, computing $\mathrm{CV}(\lambda)$ for each one, and choosing the λ that minimizes it. Plots of $\mathrm{CV}(\lambda)$ vs. λ may be useful.

Why do we think this is good? First notice that

$$\begin{aligned} \mathrm{E}\{y_i - \hat{f}_\lambda^{-i}(x_i)\}^2 &= \mathrm{E}\{y_i - f(x_i) + f(x_i) - \hat{f}_\lambda^{-i}(x_i)\}^2 \\ &= \sigma^2 + \mathrm{E}\{\hat{f}_\lambda^{-i}(x_i) - f(x_i)\}^2. \end{aligned}$$

Using the assumption that $\hat{f}_\lambda^{-i}(x_i) \approx \hat{f}_\lambda(x_i)$ we see that

$$\mathrm{E}\{\mathrm{CV}(\lambda)\} \approx \mathrm{EPE}(\lambda)$$

However, what we really want is

$$\min_\lambda \mathrm{E}\{\mathrm{CV}(\lambda)\} \approx \min_\lambda \mathrm{EPE}(\lambda)$$

but the law of large numbers says the above will do.

Why not simply use the averaged squared residuals

$$\mathrm{ASR}(\lambda) = n^{-1} \sum_{i=1}^{n} \{y_i - \hat{f}_\lambda(x_i)\}^2?$$

It turns out this under-estimates the EPE. Notice in particular that the estimate $\hat{f}(x_i) = y_i$ always has ASR equal to 0! But we know the EPE will not be small.

Later we will learn of a couple of ways we can adjust the ASR to form "good" estimates of the MSE.

CV for linear models

Now we will see some of the practical advantages of linear models. Soon, we will see classes of models that are much more flexible, but have these same advantages.

For linear smoothers in general it is not obvious what is meant by $\hat{f}_\lambda^{-i}(x_i)$. Let's give a definition...

Notice that any reasonable smoother will smooth constants into constants, i.e. $\mathbf{S1} = \mathbf{1}$. If we think of the rows $\mathbf{S}_{i\cdot}$ of $\mathbf{S}$ as weight vectors, this condition is requiring that all the n weights in each of the n vectors add up to 1. We can define $\hat{f}_\lambda^{-i}(x_i)$ as the "weighted average"

$$\mathbf{S}_{i\cdot}\mathbf{y} = \sum_{j=1}^{n} S_{ij} y_j$$

but giving zero weight to the ith entry, i.e.

$$\hat{f}_\lambda^{-i}(x_i) = \frac{1}{1 - S_{ii}} \sum_{j \neq i} S_{ij} y_j.$$

From this definition we can find CV without actually making all the computations again. Lets see how:

Notice that

$$\hat{f}_\lambda^{-i}(x_i) = \sum_{j \neq i} S_{ij} y_j + S_{ii} \hat{f}_\lambda^{-i}(x_i).$$

The quantities we add up to obtain CV are the squares of

$$y_i - \hat{f}_\lambda^{-i}(x_i) = y_i - \sum_{j \neq i} S_{ij} y_j - S_{ii} \hat{f}_\lambda^{-i}(x_i).$$

Adding and subtracting $S_{ii} y_i$ we get

$$y_i - \hat{f}_\lambda^{-i}(x_i) = y_i - \hat{f}_\lambda(x_i) + S_{ii}(y_i - \hat{f}_\lambda^{-i}(x_i))$$

which implies

$$y_i - \hat{f}_\lambda^{-i}(x_i) = \frac{y_i - \hat{f}_\lambda(x_i)}{1 - S_{ii}}$$

and we can write

$$\mathrm{CV}(\lambda) = n^{-1} \sum_{i=1}^{n} \left\{ \frac{y_i - \hat{f}_\lambda(x_i)}{1 - S_{ii}} \right\}^2$$

so we don't have to compute $\hat{f}_\lambda^{-i}(x_i)$!

Lets see how this definition of CV may be useful in finding the MSE.

Notice that the above defined CV is similar to the ASR except for the division by $1 - S_{ii}$. To see what this is doing we notice that in many situations $S_{ii} \approx [\mathbf{S}_\lambda \mathbf{S}_\lambda]_{ii}$ and $1/(1 - S_{ii})^2 \approx 1 + 2S_{ii}$ which implies

$$\mathrm{E}[\mathrm{CV}(\lambda)] \approx \mathrm{EPE}(\lambda) + 2\mathrm{ave}[\mathrm{diag}(\mathbf{S}_\lambda)\mathbf{v}^2].$$

Thus CV adjusts ASR so that in expectation the variance term is correct but in doing so induces an error of $2S_{ii}$ into each of the bias components.

K-fold cross validation

We saw that for linear smoothers, we can get an estimate of EPE using (an approximation) cross-validation, where a single training point is set aside and tested with a model trained on the remaining points. For methods where no such nice approximations exist, one has to do the cross-validation procedure directly.

K-fold approximates this by testing on more than one point at a time. In particular, the training set is divided in K equal-sized parts, and each is used in turn as a validation set, testing with a model trained on the remaining $K - 1$ parts.

Lasso

The lasso's definition is similar to that of ridge regression. However, we obtain very different results.

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{ subject to } \sum_{j=1}^{p}|\beta_j| \leq s$$

Like ridge regression it can be written as a penalized loss problem (also known as *lagrangian form*):

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

Unlike ridge regression, the lasso estimate $\hat{\beta}^{\text{lasso}}$ does not have a closed-form solution. However, there has been an explosion of algorithms to efficiently solve the lasso problem for very large datasets (especially useful when $n << p$).

In practice one sees that the lasso makes more coefficients 0. This is sometimes nicer for interpret-ability. See the book and papers on lasso for more information. Figure 6 shows the path coefficients take as λ goes from infinity to zero. Notice coefficients move from *exactly* zero to non-zero for some λ value, and continue to grow as λ increases.

Since we don't have a nice closed form solution, cross-validation has to be done directly. Figure 7 shows the estimated expected prediction error from 10-fold cross validation.

Other in-sample estimates

Mallow's C_p

The following three sections describe the related ways of choosing the best model using only the training data. These are sometimes called in-sample methods. They were originally developed in the context of parametric models. For example, Mallow's C_p was developed for choosing the number of covariates in a regression model (Mallows 1973).

The basic idea is to start with to try estimate the expected difference between ASR and EPE. Remember ASR is a random quantity and EPE is not!

The larger the model, the more ASR underestimates EPE. For a linear model with p covariates, Mallow's C_p estimates this bias with $2 * d/N * \hat{\sigma}^2$. A problem here is that we need to estimate $\hat{\sigma}^2$. Which model do we use? Typically, a big model (small bias) is used. Below I include some notes on the calculations as presented by the Mallow.

The C_p statistic is defined as a criteria to assess fits when models with different numbers of parameters are being compared. It is given by

$$C_p = \frac{\text{RSS}(p)}{\sigma^2} - N + 2p \tag{8}$$

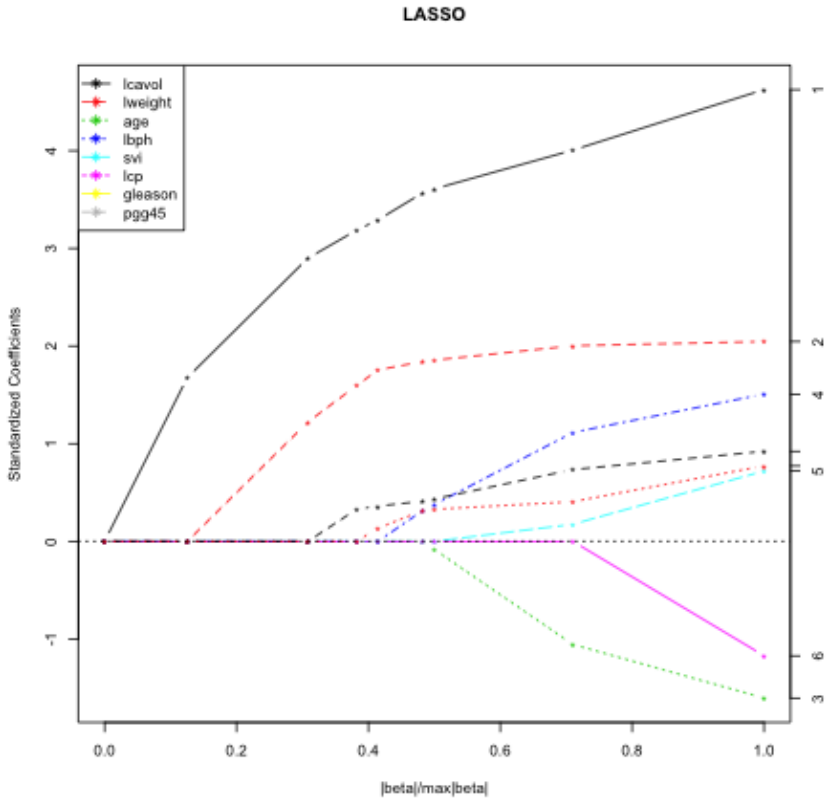

Figure 6: Path of coefficients in lasso estimate, prostate cancer data

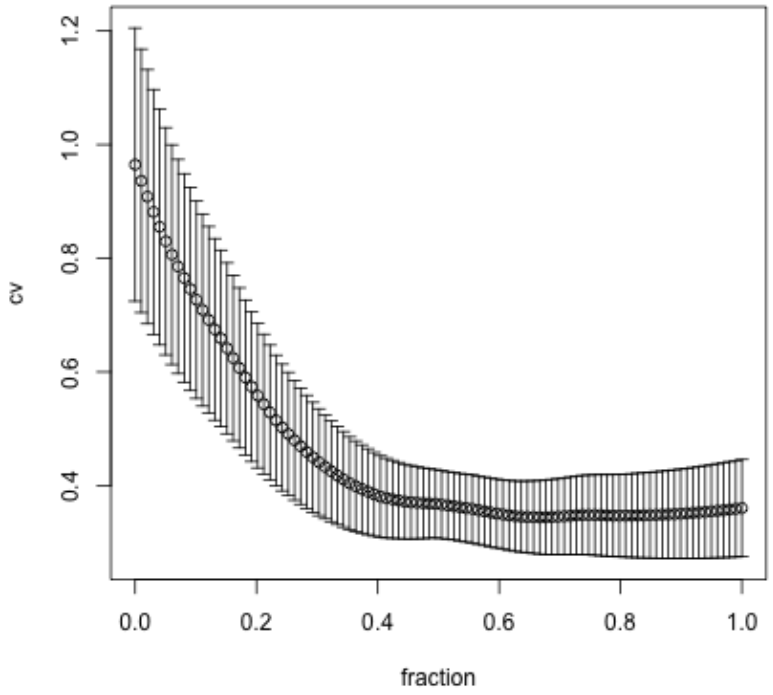

Figure 7: 10-fold cross-validation estimate of prediction error for lasso in prostate cancer data

If model(p) is correct then C_p will tend to be close to or smaller than p. Therefore a simple plot of C_p versus p can be used to decide among models.

In the case of ordinary linear regression, Mallow's method is based on estimating the mean squared error (MSE) of the estimator $\hat{\boldsymbol{\beta}}_p = (\mathbf{X}_p'\mathbf{X}_p)^{-1}\mathbf{X}_p'\mathbf{Y}$,

$$\mathrm{E}[\hat{\boldsymbol{\beta}}_p - \boldsymbol{\beta}]^2$$

via a quantity based on the residual sum of squares (RSS)

$$\begin{aligned}
\mathrm{RSS}(p) &= \sum_{n=1}^{N}(y_n - \mathbf{x}_n\hat{\boldsymbol{\beta}}_p)^2 \\
&= (\mathbf{Y} - \mathbf{X}_p\hat{\boldsymbol{\beta}}_p)'(\mathbf{Y} - \mathbf{X}_p\hat{\boldsymbol{\beta}}_p) \\
&= \mathbf{Y}'(\mathbf{I}_N - \mathbf{X}_p(\mathbf{X}_p'\mathbf{X}_p)^{-1}\mathbf{X}_p')\mathbf{Y}
\end{aligned}$$

Here $\mathbf{I}_N$ is an $N \times N$ identity matrix. By using a result for quadratic forms, presented for example as Theorem 1.17 in Seber's book, page 13, namely

$$\mathrm{E}[\mathbf{Y}'\mathbf{A}\mathbf{Y}] = \mathrm{E}[\mathbf{Y}']\mathbf{A}\mathrm{E}[\mathbf{Y}] + \mathrm{tr}[\boldsymbol{\Sigma}\mathbf{A}]$$

$\boldsymbol{\Sigma}$ being the variance matrix of $\mathbf{Y}$, we find that

$$\begin{aligned}
\mathrm{E}[\mathrm{RSS}(p)] &= \mathrm{E}[\mathbf{Y}'(\mathbf{I}_N - \mathbf{X}_p(\mathbf{X}_p'\mathbf{X}_p)^{-1}\mathbf{X}_p')\mathbf{Y}] \\
&= \mathrm{E}[\hat{\boldsymbol{\beta}}_p - \boldsymbol{\beta}]^2 + \mathrm{tr}\left[\mathbf{I}_N - \mathbf{X}_p(\mathbf{X}_p'\mathbf{X}_p)^{-1}\mathbf{X}_p'\right]\sigma^2 \\
&= \mathrm{E}[\hat{\boldsymbol{\beta}}_p - \boldsymbol{\beta}]^2 + \sigma^2\left(N - \mathrm{tr}\left[(\mathbf{X}_p'\mathbf{X}_p)(\mathbf{X}_p'\mathbf{X}_p)^{-1}\right]\right) \\
&= \mathrm{E}[\hat{\boldsymbol{\beta}}_p - \boldsymbol{\beta}]^2 + \sigma^2(N - p)
\end{aligned}$$

where N is the number of observations and p is the number of parameters. Notice that when the true model has p parameters $\mathrm{E}[\mathrm{C}_p] = p$.
This shows why, if model(p) is correct, C_p will tend to be close to p.

One problem with the C_p criterion is that we have to find an appropriate estimate of σ^2 to use for all values of p.

C_p for smoothers

A more direct way of constructing an estimate of EPE is to correct the ASR. It is easy to show that

$$\mathrm{E}\{\mathrm{ASR}(\lambda)\} = \left\{1 - n^{-1}\mathrm{tr}(2\mathbf{S}_\lambda - \mathbf{S}_\lambda\mathbf{S}_\lambda')\right\}\sigma^2 + n^{-1}\mathbf{v}_\lambda'\mathbf{v}_\lambda$$

notice that

$$\mathrm{EPE}(\lambda) - \mathrm{E}\{\mathrm{ASR}(\lambda)\} = n^{-1}2\mathrm{tr}(\mathbf{S}_\lambda)\sigma^2$$

This means that if we knew σ^2 we could find a "corrected" ASR

$$\text{ASR}(\lambda) + 2\text{tr}(\mathbf{S}_\lambda)\sigma^2$$

with the right expected value.

For linear regression $\text{tr}(\mathbf{S}_\lambda)$ is the number of parameters so we could think of $2\text{tr}(\mathbf{S}_\lambda)\sigma^2$ as a penalty for large number of parameters or for un-smooth estimates.

How do we obtain an estimate for σ^2? If we had a λ^* for which the bias is 0, then the usual unbiased estimate is

$$\frac{\sum_{i=1}^n \{y_i - f_{\lambda^*}(x_i)\}^2}{n - \text{tr}(2\mathbf{S}_{\lambda^*} - \mathbf{S}_{\lambda^*}\mathbf{S}'_{\lambda^*})}$$

The usual trick is to chose one a λ^* that does little smoothing and consider the above estimate. Another estimate that has been proposed it the first order difference estimate

$$\frac{1}{2(n-1)}\sum_{i=1}^{n-1}(y_{i+1} - y_i)^2$$

Once we have an estimate $\hat{\sigma}^2$ then we can define

$$C_p = \text{ASR}(\lambda) + n^{-1}2\text{tr}(S_\lambda)\hat{\sigma}^2$$

Notice that the p usually means number of parameters so it should be C_λ.

Notice this motivates a definition for degrees of freedoms.

AIC

Akaike (1977) developed a correction for more general situations, i.e. not just the squared error case. The AIC derives a correction for the training error with the more general likelihood loss. To do this

The AIC is simply:

$$AIC = -\frac{2}{N}\text{loglik} + 2d/N$$

This reduces to Mallow's C_p in the case of Gaussian likelihood. Below is the derivation as shown by Akaike (1977).

Suppose we observe a realization of a random variable Y, with distribution defined by a parameter $\boldsymbol{\beta}$

$$\prod_{\mathbf{x}_i \in N_0} f(y_i; \mathbf{x}_i, \boldsymbol{\beta}) \equiv f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}) \tag{9}$$

where $\mathbf{y}$ is the observed response associated with the covariates $\mathbf{X}$ and $\boldsymbol{\beta} \in \mathbb{R}^P$ is a $P \times 1$ parameter vector.

We are interested in estimating $\boldsymbol{\beta}$. Suppose that before doing so, we need to choose from among P competing models, generated by simply restricting the general parameter space R^P in which $\boldsymbol{\beta}$ lies.

In terms of the parameters, we represent *the full model* with P parameters as:

$$\text{Model(P): } f_{\mathbf{Y}}(\mathbf{y}; \mathbf{x}, \boldsymbol{\beta}_P), \boldsymbol{\beta}_P = (\beta_1, \ldots, \beta_p, \beta_{p+1}, \ldots, \beta_P)'.$$

We denote the "true value" of the parameter vector $\boldsymbol{\beta}$ with $\boldsymbol{\beta}^*$.

Akaike (1977) formulates the problem of statistical model identification as one of selecting a model $f_{\mathbf{Y}}(\mathbf{y}; \mathbf{x}, \boldsymbol{\beta}_p)$ based on the
observations from that distribution, where the particular restricted model is defined by the constraint $\beta_{p+1} = \beta_{p+2} = \cdots = \beta_P = 0$, so that

$$\text{Model(p): } f_{\mathbf{Y}}(\mathbf{y}; \mathbf{x}, \boldsymbol{\beta}_p), \boldsymbol{\beta}_p = (\beta_1, \ldots, \beta_p, 0, \ldots, 0)' \tag{10}$$

We will refer to p as the *number of parameters* and to Ω_p as the sub-space of $\mathbb{R}^P$ defined by restriction (10). For each $p = 1, \ldots, P$, we may assume model(p) to estimate the non-zero components of the vector $\boldsymbol{\beta}^*$. We are interested in a criterion that helps us chose among these P competing estimates.

Akaike's original work is for IID data, however it is extended to a regression type setting in a straight forward way. Suppose that the conditional distribution of Y given $\mathbf{x}$ is know except for a P-dimensional parameter $\boldsymbol{\beta}$. In this case, the probability density function of $\mathbf{Y} = (Y_1, \ldots, Y_n)$ can be written as

$$f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}) \equiv \prod_{i=1}^{n} f(y_i; \mathbf{x}_i, \boldsymbol{\beta}) \tag{11}$$

with $\mathbf{X}$ the design matrix with rows $\mathbf{x}_i$.

Assume that there exists a true parameter vector $\boldsymbol{\beta}^*$ defining a true probability density denoted by $f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*)$. Given these assumptions, we wish to select $\boldsymbol{\beta}$, from one of the models defined as in (10), "nearest" to the true parameter $\boldsymbol{\beta}^*$ based on the observed data $\mathbf{y}$. The principle behind Akaike's criterion is to define "nearest" as the model that minimizes the Kullback-Leibler Information Quantity

$$\Delta(\boldsymbol{\beta}^*; \mathbf{X}, \boldsymbol{\beta}) = \int \{\log f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*) - \log f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta})\} f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*) \, d\mathbf{y}. \tag{12}$$

The analytical properties of the Kullback-Leibler Information Quantity are discussed in detail by Kullback (1959) . Two important properties for Akaike's criterion are

1. $\Delta(\boldsymbol{\beta}^*; \mathbf{X}, \boldsymbol{\beta}) > 0$ if $f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*) \neq f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta})$

2. $\Delta(\boldsymbol{\beta}^*; \mathbf{X}, \boldsymbol{\beta}) = 0$ if and only if $f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*) = f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta})$

almost everywhere on the range of $\mathbf{Y}$. The properties mentioned suggest that finding the model that minimizes the Kullback-Leibler Information Quantity is an appropriate way to choose the "nearest" model.

Since the first term on the right hand side of (12) is constant over all models we consider, we may instead maximize

$$\begin{aligned} H(\boldsymbol{\beta}) &= \int \log f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}) f_{\mathbf{Y}}(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}^*)\, d\mathbf{y} \\ &= \sum_{i=1}^{n} \int \log f(y_i; \mathbf{X}, \boldsymbol{\beta})\, f(y_i; \mathbf{x}_i, \boldsymbol{\beta}^*)\, dy_i. \end{aligned} \tag{13}$$

Let $\hat{\boldsymbol{\beta}}_p$ be the maximum likelihood estimate under Model(p). Akaike's procedure for model selection is based on choosing the model which produces the estimate that maximizes $\mathrm{E}_{\boldsymbol{\beta}^*}\left[H(\hat{\boldsymbol{\beta}}_p)\right]$ among all competing models. Akaike then derives a criterion by constructing an asymptotically unbiased estimate of $\mathrm{E}_{\boldsymbol{\beta}^*}\left[H(\hat{\boldsymbol{\beta}}_p)\right]$ based on the observed data.

Notice that $H(\hat{\boldsymbol{\beta}}_p)$ is a function, defined by (13), of the maximum likelihood estimate $\hat{\boldsymbol{\beta}}_p$, which is a random variable obtained from the observed data. A natural estimator of its expected value (under the true distribution of the data) is obtained by substituting the empirical distribution of the data into (13) resulting in the log likelihood equation evaluated at the maximum likelihood estimate under model(p)

$$l(\hat{\boldsymbol{\beta}}_p) = \sum_{i=1}^{n} \log f(y_i; \mathbf{x}_i, \hat{\boldsymbol{\beta}}_p).$$

Akaike noticed that in general $l(\hat{\boldsymbol{\beta}}_p)$ will overestimate $\mathrm{E}_{\boldsymbol{\beta}^*}\left[H(\hat{\boldsymbol{\beta}})\right]$. In particular Akaike found that under some regularity conditions

$$E_{\boldsymbol{\beta}^*}\left[l(\hat{\boldsymbol{\beta}}_p) - H(\hat{\boldsymbol{\beta}}_p)\right] \approx p.$$

This suggests that larger values of p will result in smaller values of $l(\hat{\boldsymbol{\beta}}_p)$, which may be incorrectly interpreted as a "better" fit, regardless of the true model. We need to "penalize" for larger values of p in order to obtain an unbiased estimate of the "closeness" of the model. This fact leads to the Akaike Information Criteria which is a bias-corrected estimate given by

$$\mathrm{AIC}(p) = -2l(\hat{\boldsymbol{\beta}}_p) + 2p. \tag{14}$$

See, for example, Akaike (1973) and Bozdogan (1987) for the details.

BIC

Objections have been raised that minimizing Akaike's criterion does not produce asymptotically consistent estimates of the correct model Notice that if we consider Model(p^*) as the correct model then we have for any $p > p*$

$$\Pr\left[AIC(p) < AIC(p^*)\right] = \Pr\left[2\{l(\hat{\boldsymbol{\beta}}_p) - l(\hat{\boldsymbol{\beta}}_{p^*})\} > 2(p - p^*)\right]. \qquad (15)$$

Notice that, in this case, the random variable $2\{l(\hat{\boldsymbol{\beta}}_p) - l(\hat{\boldsymbol{\beta}}_{p^*})\}$ is
the logarithm of the likelihood ratio of two competing models which, under certain regularity conditions, is known to converge in distribution to $\chi^2_{p-p^*}$, and thus it follows that the probability in Equation (15) is not 0 asymptotically. Some have suggested multiplying the penalty term in the AIC by some increasing function of n, say $a(n)$, that makes the probability

$$\Pr\left[2\{l(\hat{\boldsymbol{\beta}}_p) - l(\hat{\boldsymbol{\beta}}_{p^*})\} > 2a(n)(p - p^*)\right]$$

asymptotically equal to 0. There are many choices of $a(n)$ that would work in this context. However, some of the choices made in the literature seem arbitrary.

Schwarz (1978) and Kashyap (1982) suggest using a Bayesian approach to the problem of model selection which, in the IID case, results in a criterion that is similar to AIC in that it is based on a penalized log-likelihood function evaluated at the maximum likelihood estimate for the model in question. The penalty term in the Bayesian Information Criteria (BIC) obtained by Schwarz (1978) is the AIC penalty term p multiplied by the function $a(n) = \frac{1}{2}\log(N)$.

The Bayesian approach to model selection is based on maximizing the posterior probabilities of the alternative models, given the observations. To do this we must define a strictly positive prior probability $\pi_p = \Pr[\text{Model}(p)]$ for each model and a conditional prior $d\mu_p(\boldsymbol{\beta})$ for the parameter given it is in Ω_p, the subspace defined by Model(p). Let $\mathbf{Y} = (Y_1, \ldots, Y_n)$ be the response variable and define the distribution given $\boldsymbol{\beta}$ following (11)

$$f_{\mathbf{Y}}(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) \equiv \prod_{i=1}^{n} f(y_i; \mathbf{x}_i, \boldsymbol{\beta})$$

The posterior probability that we look to maximize is

$$\Pr\left[\text{Model}(p)|\mathbf{Y} = \mathbf{y}\right] = \frac{\int_{\Omega_p} \pi_p f_{\mathbf{Y}}(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) d\mu_p(\boldsymbol{\beta})}{\sum_{q=1}^{P} \int_{\Omega_q} \pi_q f_{\mathbf{Y}}(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) d\mu_q(\boldsymbol{\beta})}$$

Notice that the denominator depends neither on the model nor the data, so we need only to maximize the numerator when choosing models.

Schwarz (1978) and Kashyap (1982) suggest criteria derived by taking a Taylor expansion of the log posterior probabilities of the alternative models. Schwarz

(1978) presents the following approximation for the IID case

$$\log \int_{\Omega_p} \pi_p f_{\mathbf{Y}}(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) d\mu_p(\boldsymbol{\beta}) \approx l(\hat{\boldsymbol{\beta}}_p) - \frac{1}{2} p \log n$$

with $\hat{\boldsymbol{\beta}}_p$ the maximum likelihood estimate obtained under Model(p).
This fact leads to the Bayesian Information Criteria (BIC) which is

$$\text{BIC}(p) = -2l(\hat{\boldsymbol{\beta}}_p) + p \log n \tag{16}$$