MSG500/MVE190
Linear Models - Lecture 9

Rebecka Jörnsten
Mathematical Statistics
University of Gothenburg/Chalmers University of Technology

November 19, 2015

1 RECAP

- Ultimate validation of a model is to test its predictive capacity
- Prediction performance is affected by both Bias and Estimation variance
- We combine the two into the criterion *prediction* $MSE = Bias^2$ + *Estimation Variance*
- We can't compute this in real life situations since we don't know the true model (needed to compute the bias)
- With training and test data we can separate model estimation from model validation (prediction)
- Training data: compute $MSE_{train} = RSS/n$ (the fit of the model on the data used to estimate model parameters)
- Test data: compute $pMSE = MSE_{test}$ (the fit of the model to new data *not* used for estimation).
- The $pMSE$ is a substitute for the real *prediction MSE* as defined above.
- We select the model that minimizes the $pMSE$
- We usually don't have separate training and test data, and use *cross-validation* to mimic this scenario, and to estimate the *prediction MSE*

2 Model Selection

Usually, parsimonious, or simple models work best for prediction. Why is that? Well, it's easier to estimate parameters of a simple model with limited amounts of data. We limit the risk of including a spurious relationships that do not generalize to future data. This preference for as simple an explanation of the data as possible is sometimes referred to as *Occam's Razor* - a simple model is 'safer' in terms of prediction performance and is also easier to interpret.

2.1 Caution

Most model selection procedures work with a global criterion based on e.g. RSS. Don't forget to check the model fit using diagnostic plots though! The selected model is not OK if you see trends or patterns in the residuals and the presence of outliers can have a huge impact on the RSS.

3 Optimism and Model selection criteria

We review the setup form previous lectures:

$$\text{Training data: } (X_i, y_i)_{i=1}^n, X_i = (x_{i1}, x_{i2}, \ldots, x_{i,p-1})$$

$$\text{Test data: } (X_i, y_i^{new})_{i=1}^n, X_i = (x_{i1}, x_{i2}, \ldots, x_{i,p-1})$$

The test data consist of new outcome data drawn from the same true model and at the same x-locations as the training data. (Note, this is a practical construction to derive the criterion, but not a necessity to use this for model selection. However, you should not use modelselection criteria for a specific range of x and assume you can predict well on a different range of x.)

The true model $\beta^* = (\beta_0^*, \beta_1^*, \ldots, \beta_{p-1}^*)$ is unknown to us. If the outcome is not related to some of the x-variables, x_j, the corresponding $\beta_j^* = 0$. We can thus write

$$\text{Training data: } y_i = X_i\beta^* + \epsilon_i,\ \epsilon \sim N(0, \sigma^2)$$

$$\text{Test data: } y_i^{new} = X_i\beta^* + \epsilon_i^{new},\ \epsilon^{new} \sim N(0, \sigma^2)$$

1. We enumerate all models $m = 1, \ldots, M$. If we have $p-1$ variables there are $M = 2^{p-1}$ possible subset models.
2. We fit each model m to the training data and obtain parameter estimates $\hat{\beta}(m)$ with corresponding fitted values $\hat{y}(m)_i, i = 1, \ldots, n$
3. $RSS(m) = \sum_{i=1}^n (y_i - \hat{y}(m)_i)^2$
4. $MSE(m)_{train} = \frac{1}{n} RSS(m)$
5. $pMSE(m) = MSE(m)_{test} = \frac{1}{n}\sum_{i=1}^n (y_i^{new} - \hat{y}(m)_i)^2$

$pMSE(m) - MSE(m)$ is called the *optimism* for model m. That is, because we fit the model m to match the training data as best possible using the least squares criterion, the $MSE(m)$ is an underestimate of how well the model would perform on future data. We can estimate this optimism directly using training and test data we create from our observed data set. We did this in lecture 8, using a technique called *cross-validation*.

In this lecture we will try to estimate the *expected value* of the optimism, or gap between $pMSE$ and MSE directly. This is the basis for the Mallow's Cp selection criterion.

A somewhat similar derivation (though there are approaches) leads to the AIC (Akaike's "An Information Criterion"), but it's motivated very differently. Here we look at the difference between the model likelihoods using a distance called Kullback-Leibler. Similarly to Cp, however, the derivation can end up looking at the difference between the expected likelihood (wrt the true model) on future data, and the expected maximized likelhood at the parameters values estimated from training data and estimate the likelihood optimism instead of the RSS (or MSE) optimism. When data are normally distributed, the likelihood approach means we are measuring the fit of the models on the scale $\log(RSS)$.

The BIC is a Bayesian criterion where we choose the model with the largest *posterior probability*. BIC is derived under the weakest prior on models and is a linear approximation of what the posterior looks like, which holds for large samples. Since BIC involves the likelihood it also works on the scale $\log(RSS)$.

More on these at the close of the lecture.

4 Deriving Mallow's Cp

Let us take a closer look at the gap between the $pMSE$-curve and the MSE-curve from training data.

The *Prediction Error* of model m is defined as

$$PE(m) = E[\frac{1}{n}\sum_{i=1}^n (y_i^{new} - \hat{y}(m)_j)^2] = \frac{1}{n}\sum_{i=1}^n \underbrace{E[y_i^{new} - \hat{y}(m)_i]^2}_{***}$$

The expectation is taken over the new test data and the training data. The prediction $\hat{y}(m)_i$ is a function of the training data only. We expand the term *** above as follows:

$$*** = E\big[y_i^{new} - E(y_i^{new}) + E(y_i^{new}) - E(\hat{y}(m)_i) + E(\hat{y}(m)_i) - \hat{y}(m)_i\big]^2 =$$

$$= \underbrace{E\big[y_i^{new} - E(y_i^{new})\big]^2}_{(1)} + \underbrace{E\big[E(y_i^{new}) - E(\hat{y}(m)_i)\big]^2}_{(2)} + \underbrace{E\big[E(\hat{y}(m)_i) - \hat{y}(m)_i\big]^2}_{(3)} +$$

$$+ \underbrace{2E\big[\big(y_i^{new} - E(y_i^{new})\big)\big(E(y_i^{new}) - E(\hat{y}(m)_i)\big)\big]}_{(4)} + \underbrace{2E\big[\big(y_i^{new} - E(y_i^{new})\big)\big(E(\hat{y}(m)_i) - \hat{y}(m)_i\big)\big]}_{(5)} +$$

$$+ \underbrace{2E\big[\big(E(y_i^{new}) - E(\hat{y}(m)_i)\big)\big(E(\hat{y}(m)_i) - \hat{y}(m)_i\big)\big]}_{(6)}$$

We will work through each of the 6 terms above.

(1)
* $E[y_i^{new} - E(y_i^{new})]^2 = \sigma^2$.
* The *irreducible error*.
* The noise or random scatter around the true model.

(2)
* $E\big[E(y_i^{new}) - E(\hat{y}(m)_i)\big]^2 = bias^2(i,m)$
* this is the local bias of model m at location x_i
* if model m is adequate the bias is 0

(3)
* $E\big[E(\hat{y}(m)_i) - \hat{y}(m)_i\big]^2 = V[\hat{y}(m)_i]$
* the estimation variance of model m
* increases with the complexity of the model

(4)
* This term is 0 since the constant $\big(E(y_i^{new}) - E(\hat{y}(m)_i)\big)$ can be pulled outside the outer expectation and $E\big[(y_i^{new} - E(y_i^{new})\big] = 0$

(5)
* This term is 0 since y_i^{new} and y_i are uncorrelated
* $(5) = E\big[y_i^{new} - E(y_i^{new})\big]E\big[E(\hat{y}(m)_i) - \hat{y}(m)_i\big] = 0$

(6)
* This term is 0 since the constant $\big(E(y_i^{new}) - E(\hat{y}(m)_i)\big)$ can be pulled outside the outer expectation and $E\big[E(\hat{y}(m)_i) - \hat{y}(m)_i\big] = 0$

We summarize our findings:

$$PE(m) = \frac{1}{n}\sum_{i=1}^{n}(\sigma^2 + bias^2(i,m) + V[\hat{y}(m)_k]) = \sigma^2 + \bar{bias}^2(m) + \frac{1}{n}\sum_{i=1}^{n} V[\hat{y}(m)_i]$$

Example - linear model

What if $E[y_i] = \sum_{j\in m}\beta_j x_{ij}$, where $j \in m$ means the sum includes the variables j in model m with $p(m)$ variables total?

- $\hat{\beta}(m) = (X'X)^{-1}X'y$, where X is the $n \times p(m)$ *design* matrix for model m
- If the model m with $p(m)$ variables is adequate there is no bias
- $V[\hat{y}] = \sigma^2 H$
- $V[\hat{y}_i] = \sigma^2 h_{ii} \rightarrow \frac{1}{n}V[\hat{y}_i] = \frac{\sigma^2}{n}\sum_i h_{ii} = \frac{\sigma^2}{n}Trace(H)$
- $Trace(H) = Trace(X(X'X)^{-1}X') = Trace((X'X)^{-1}(X'X)) = Trace(I_{p(m)}) = p(m)$

So, for a linear model with $E[y_i] = \sum_{j\in m}\beta_j x_{ij}$ we have

$$PE(m) = \sigma^2\big(1 + \frac{p(m)}{n}\big)$$

4.1 The training error

What about the training error? What can we *expect* the RSS (or the training MSE) to be across multiple data sets from the same underlying, true model?

$$TE(m) = E\big[\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}(m)_i)^2\big] = \frac{1}{n}\sum_{i=1}^{n} E\big[y_i - \hat{y}(m)_i\big]^2$$

Similar to the above, we expand the expression $E\big[y_i - \hat{y}(m)_i\big]^2$ into 6 terms but now term (5) is *not* zero:

$$\textbf{(5)} \quad = 2E\big[(y_i - E(y_i)(E(\hat{y}(m)_i) - \hat{y}(m)_i)\big] = 2E\big[y_i(E[\hat{y}(m)_i] - \hat{y}(m)_i)\big] =$$
$$= 2E[y_i]E[\hat{y}(m)_i] - 2E[y_i\hat{y}(m)_i] = -2Cov(y_i, \hat{y}(m)_i)$$

So, taken together we have that the *expected* training error is

$$TE(m) = PE(m) - \frac{2}{n}\sum_{i=1}^{n} Cov(y_i, \hat{y}(m)_i).$$

We have thus learnt that the training error is always less than the prediction error ($TE(m) < PE(m)$). In addition, the training error (TE) is much smaller than the prediction error (PE) if y_i and $\hat{y}_i$ are highly correlated. This is exactly what happens when we fit complex models to data: we match data to the model closely by adding parameters to our model. That means that the gap between the training error (TE) (expected MSE) and the prediction error (PE) (expected prediction MSE) increases with the complexity, or size, of the model we fit. As you can see from the above

$$gap(PE, TE) = \frac{2}{n}\sum_{i=1}^{n} Cov(y_i, \hat{y}(m)_i).$$

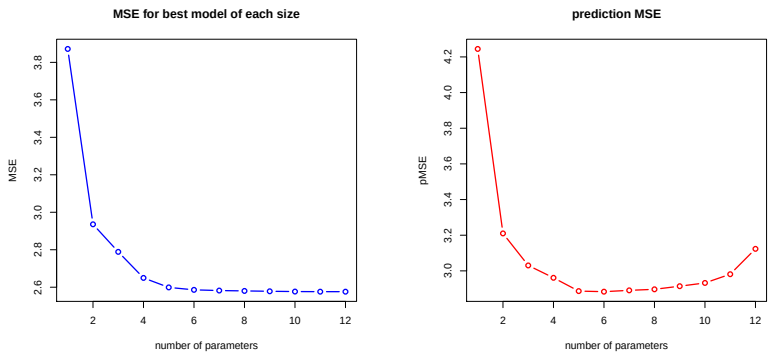

Figure 1: Left: RSS-curve. MSE on training data. Middle: pMSE-curve. MSE on test data.

In Figure 1 we depict the MSE and pMSE for a data example. For large models (where the bias is small) we see that the gap between the two curves increases with the size of the model.

Example: For a Least Squares linear models fit we can write $\hat{y} = Hy$ and $Cov(y_i, \hat{y}_i) = \sigma^2 h_{ii}$. We can thus write

$$gap(PE, TE) = \frac{2}{n}\sum_{i=1}^{n} Cov(y_i, \hat{y}(m)_i) = 2\frac{\sigma^2}{n}p(m),$$

since

$$H(m) = X(X'X)^{-1}X' \rightarrow \frac{1}{n}\sum_{i} h_{ii} Trace(H) \rightarrow$$

$$Trace(H) = Trace(X(X'X)^{-1}X') = Trace((X'X)(X'X)^{-1}) = Trace(I_{p(m)}) = p(m)$$

4.2 Conclusion

Why did we bother with the above derivation? Well, in most cases we don't actually have an independent test data set so we can't estimate the expected pMSE (the PE). However, we need it for model selection since we have already established that the RSS does not work. Thus, we need to get at an estimate of the PE. Since we have established that the gap between the *expected* MSE and pMSE is $\frac{2\sigma^2}{n}p(m)$ and we have a natural estimate for the expected training MSE in the MSE (RSS/(n-p)). Thus, our selection criterion is

$$p\widehat{MSE}(m) = \widehat{MSE}(m) + \frac{2\sigma^2}{n}p(m) \hat{=} \frac{RSS(m)}{n - p(m)} + \frac{2\sigma^2}{n}p(m).$$

Many model selection criteria have this format:

$$\text{Goodness-of-fit measure} + \text{modelsize-penalty}.$$

The one we just derived is called *Mallow's Cp* and takes the form

$$Cp(m) = MSE + \frac{2\sigma^2}{n}p(m),$$

or

$$RSS(m) + 2\sigma^2 p(m).$$

To use this in practise we need to know σ^2. We plug in the best (or safest) estimate for σ^2 we have, namely $\hat{\sigma}^2$ obtained from the largest model fit to the data.

1. Enumerate all models $m = 1, \ldots, M$
2. Evaluate the $MSE(m), RSS(m)$ for all models
3. Compute $Cp(m) = RSS(m) + 2\hat{\sigma}^2 p(m)$ where $\hat{\sigma}^2 = RSS(M)/(n - p(M))$
4. Pick the model that minimizes Cp

(Note, the Cp that `leaps()` provides is a scaled version of the above (divide by $\hat{\sigma}^2$).)

As mentioned above, there are other commonly used model selection criteria that take on a similar form (and can be derived somewhat similarly): the AIC and BIC, where

$$AIC(m) = n \log RSS(m) + 2p(m)$$

and

$$BIC(m) = n \log RSS(m) + p(m) \log(n).$$

Note that both these criteria are similar to Cp in that they compare the goodness of fit (a function of the RSS) and the complexity of the model (the number of parameters $p(m)$). The AIC is likelihood based, which for linear regression models makes Cp and AIC behave rather similarly. The BIC is a Bayesian take on model selection. What you should know is that, in general, the AIC is rather "generous", i.e. picks larger models whereas the BIC is quite conservative in comparison (picks smaller models), especially for small sample sizes. It has been shown that the AIC is overly generous asymptotically whereas the BIC is consistent. However, for finite samples it is more complicated. In addition, in linear regression Cp and AIC behave similary, often picking the same size models.

5 Demo 9

We will compare cross-validation, backward selection and all subset selection using Cp, AIC and BIC using the South African heart disease data as our demo data. We start by reading the data into R. We then use the R package `leaps()` to create an enumeration of all subset models (the matrix `Models` below).

```
> SA<-data.frame(read.table("SA.dat",header=T)) ## read in the data
> SAuse<-SA
> SAuse$ldl<-log(SA$ldl)
> SAuse$obesity<-log(SA$obesity)
```

```
> SAuse$age<-log(SA$age)
> yy<-SA[,12]
> xx<-SA[,-12]
> ##
> library(leaps)
> rleaps<-regsubsets(xx,yy,int=T,nbest=250,nvmax=250,really.big=T,method=c("ex"))
> ## all subset models
> cleaps<-summary(rleaps,matrix=T)
> ## True/False matrix. The r-th is a True/False statement about which
> Models<-cleaps$which
> Models<-rbind(c(T,rep(F,dim(xx)[2])),Models)
> ## adding the empty model (just an intercept) to the model matrix
```

Let's review 10-fold cross-validation. First, we create the 10 folds of data:

```
> K<-10
> ii<-sample(seq(1,length(yy)),length(yy)) ## random perturbation of observations first.
> foldsize<-floor(length(yy)/K)
> sizefold<-rep(foldsize,K)
> restdata<-length(yy)-K*foldsize
> if (restdata>0) {
+ sizefold[1:restdata]<-sizefold[1:restdata]+1 }
> ## creates the size for each fold
```

We cycle trough each fold of data, fit each of the models and compute the prediction errors:

```
> Prederrors<-matrix(0,dim(Models)[1],K)
> # a matrix to store the prediction errors in
> iused<-0
> Xmat<-as.matrix(cbind(rep(1,dim(xx)[1]),xx)) # the design matrix
> for (k in (1:K)) {
+   itest<-ii[(iused+1):(iused+sizefold[k])] ## the k-fold test set
+   itrain<-ii[-c((iused+1):(iused+sizefold[k]))] ## the k-fold training set
+   iused<-iused+length(itest)
+   for (mm in (1:dim(Models)[1])) {
+      betahat<-solve(t(Xmat[itrain,Models[mm,]])%*%
+                       Xmat[itrain,Models[mm,]])%*%t(Xmat[itrain,Models[mm,]])%*%yy[itrain]
+      ypred<-Xmat[itest,Models[mm,]]%*%betahat ## predictions
+      Prederrors[mm,k]<-sum((yy[itest]-ypred)^2) } }
> PE<-apply(Prederrors,1,sum)/length(yy)  ## final prediction errors, average across all folds.
```

There are more than 1400 models in the above comparison: here are the top 5

```
> jj<-sort.list(PE)[1:5]
> print(as.matrix(Models[jj,]))

  (Intercept)   age   sbp adiposity obesity typea alcohol alcind tobacco tobind  chd famhist
5        TRUE  TRUE FALSE      TRUE   FALSE FALSE    TRUE  FALSE   FALSE   TRUE TRUE   FALSE
6        TRUE  TRUE FALSE      TRUE   FALSE FALSE    TRUE  FALSE   FALSE   TRUE TRUE    TRUE
6        TRUE  TRUE FALSE      TRUE    TRUE FALSE    TRUE  FALSE   FALSE   TRUE TRUE   FALSE
4        TRUE FALSE FALSE      TRUE   FALSE FALSE    TRUE  FALSE   FALSE   TRUE TRUE   FALSE
6        TRUE  TRUE FALSE      TRUE   FALSE  TRUE    TRUE  FALSE   FALSE   TRUE TRUE   FALSE
```

I also use the `xtable()` command in R to create a nicer table that works with LaTeX (if you use another editor for your lab or project work, make sure to put output like this into a proper table with a caption).

```
> z<-data.frame(as.matrix(cbind(Prederrors[jj,],PE[jj])))
> colnames(z) <- c("Fold1","Fold2","Fold3","Fold4","Fold5","Fold6","Fold7",
+                  "Fold8","Fold9","Fold10","PE")
> row.names<-c(seq(1,5))
> library(xtable)
> xtable(z, digits=c(0, rep(0,10),3),
+        caption="Prederrors in different folds and total",label="tab:CVldl")
```

	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	Fold10	PE
1	70	48	101	73	94	154	105	46	84	102	2.815
2	66	45	107	82	89	155	102	44	88	101	2.821
3	78	49	99	72	92	155	106	45	86	100	2.825
4	66	48	102	69	89	158	107	49	87	105	2.826
5	73	46	99	73	96	156	103	51	83	101	2.826

Table 1: Prederrors in different folds and total

In Table 1 you can compare the fold results and total prediction error results for the different models (seen in the R output). The winning model is

```
> winmod<-Models[which.min(PE),]
> print(winmod)

(Intercept)         age         sbp   adiposity     obesity       typea     alcohol      alcind
       TRUE        TRUE       FALSE        TRUE       FALSE       FALSE        TRUE       FALSE
        chd     famhist
       TRUE       FALSE
```

Let's compare this to the backward model selection results:

```
> mm<-lm(log(ldl)~log(age)+sbp+adiposity+log(obesity)+typea+alcohol+alcind+
+          tobacco+tobind+as.factor(chd)+as.factor(famhist),data=SA)
> ss<-step(mm,trace=F) ## backward selection (using AIC)
> print(ss)

Call:
lm(formula = log(ldl) ~ adiposity + log(obesity) + alcohol +
    tobind + as.factor(chd) + as.factor(famhist), data = SA)

Coefficients:
        (Intercept)           adiposity        log(obesity)             alcohol               tob
          -0.295887            0.019417            0.342913           -0.003449             0.143
    as.factor(chd)1  as.factor(famhist)2
           0.154434            0.071745
```

Let's now try using all subset model selection criteria Cp, AIC and BIC instead. We use the results from `rleaps()` above (which computes Cp for us, as well as everything we need to compute AIC and BIC).

```
> Models<-cleaps$which
> Models<-rbind(c(T,rep(F,dim(xx)[2])),Models)
> nullrss<-sum((yy-mean(yy))^2)
> RSS<-cleaps$rss
> RSS<-c(nullrss,RSS)
> modsize<-apply(Models==T,1,sum)
> BIC<-length(yy)*log(RSS)+modsize*log(length(yy))
> mse<-min(RSS)/(length(yy)-max(modsize))
> CP<-(RSS+2*modsize*mse)/mse-length(yy)
> AIC<-length(yy)*log(RSS)+2*modsize

> plot(modsize,CP,xlab="modelsize",main="Cp")

> plot(modsize,AIC,xlab="modelsize",main="AIC")

> plot(modsize,BIC,xlab="modelsize", main="BIC")
```

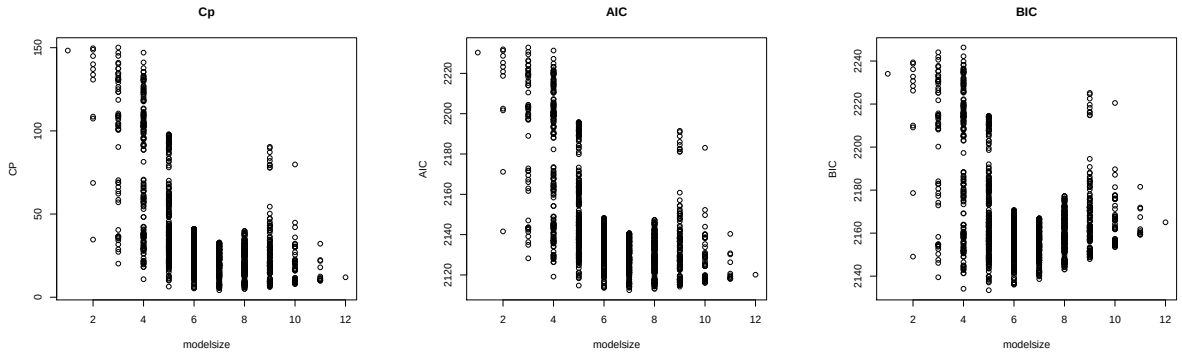

Figure 2: Cp, AIC and BIC selection criteria

In Figure 2 we depict Cp, AIC and BIC as a function of model size for all models we investigate. Notice how the lower envelope, which consists of the best models of each size, increases faster for large models using BIC compared with Cp and AIC. BIC penalizes model size more severely than Cp and AIC.

```
> rleaps<-regsubsets(xx,yy,int=T,nbest=1,nvmax=dim(xx)[2]+1,really.big=T,method=c("ex"))
> ## nbest=1: the best model of each size
> cleaps<-summary(rleaps,matrix=T)
> # just the best model of every size
> Models<-cleaps$which
> Models<-rbind(c(T,rep(F,dim(xx)[2])),Models)
> nullrss<-sum((yy-mean(yy))^2)
> RSS<-cleaps$rss
> RSS<-c(nullrss,RSS)
> modsize<-apply(Models==T,1,sum)
> BIC<-length(yy)*log(RSS)+modsize*log(length(yy))
> mse<-min(RSS)/(length(yy)-max(modsize))
> CP<-(RSS+2*modsize*mse)/mse-length(yy)
> AIC<-length(yy)*log(RSS)+2*modsize
> par(mfrow=c(1,1))
> plot(modsize,CP-min(CP),type="l",main="Cp-black,AIC-blue,BIC-red")
> lines(modsize,AIC-min(AIC),col="blue")
> lines(modsize,BIC-min(BIC),col="red")
> abline(h=0,lty=2)
```

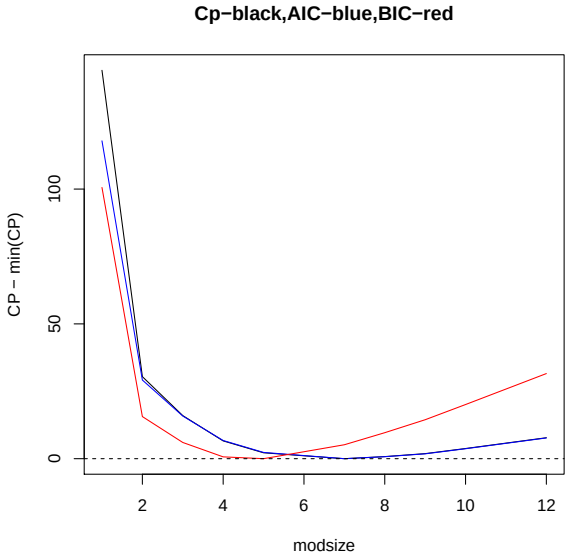

Figure 3: Cp, AIC and BIC selection criteria

It is easier to see this if we limit ourselves to the winning model of each size (see Figure 3).

We summarize the selected models

```
> cpmod<-Models[CP=min(CP),]
> aicmod<-Models[AIC==min(AIC),]
> bicmod<-Models[BIC==min(BIC),]
> print(cpmod)

(Intercept)         age         sbp   adiposity     obesity       typea     alcohol      alcind
       TRUE       FALSE       FALSE        TRUE       FALSE       FALSE        TRUE       FALSE
        chd     famhist
       TRUE       FALSE

> print(aicmod)

(Intercept)         age         sbp   adiposity     obesity       typea     alcohol      alcind
       TRUE        TRUE       FALSE        TRUE       FALSE       FALSE        TRUE       FALSE
        chd     famhist
       TRUE        TRUE

> print(bicmod)

(Intercept)         age         sbp   adiposity     obesity       typea     alcohol      alcind
       TRUE       FALSE       FALSE        TRUE       FALSE       FALSE        TRUE       FALSE
        chd     famhist
       TRUE       FALSE
```