Model Selection in Logistic Regression

Summary of Main Points

Recall that the two main objectives of regression modeling are:

- **Estimate the effect of one or more covariates** while adjusting for the possible confounding effects of other variables.

  Here, usually no single "final" model need be selected, one is free to examine as many models as one wishes, and draw overall conclusions from all results.

- **Prediction** of the outcome for the next set of similar subjects.

  Here model selection can be useful, but "best" predictions are usually obtained by averaging over a number of "top" models.

  Recall that "Bayesian Model Averaging" did a particularly good job of this, and that the BIC programs allow one to:

  - See what covariates are in the best models, with a model list provided that is ordered from best to worst.
  - Line up the coefficient estimates from the different models so that one can easily see what estimates change as variables enter end exit the models, good for investigating confounding.
  - Automatically perform model averaging.

  Thus, regardless of whether one is interested in prediction or assessing the effects of variables adjusted for confounding, these programs are very useful.

  We have seen the program bicreg for linear regression, and we will today see bic.glm for generalized linear models, including logistic regression.

In most sciences, the aim is to seek the most parsimonious model that still explains the data. Smaller models tend to be more generalizable, and more numerically stable when fit to a data set of finite size.

Many researchers recommend including clinically and intuitively relevant variables, especially for control of confounding. Sometimes a single variable by itself will not exhibit strong confounding, but a collection of variables might.

Remember that a model selected entirely by statistical means may not be intuitively reasonable.

Remember that one still needs to consider possible interactions terms, and issues such as linearity of the covariates in terms of the logit of the probability of the outcome. Polynomial terms, transformations of the data, etc are possible, just as in linear regression. This can usually be investigated at the univariate modeling stage.

In logistic regression cells with small numbers of outcomes can cause numerical estimation problems, so it is usually a good idea not to skip the preliminary steps we have outlined, including cross-tables of categorical outcomes.

As always, keep in mind the famous quote from George Box:

"All models are wrong, but some are useful."

We are not finding the "correct" model, but rather trying to solve our practical problems using a model reasonable for our purposes.

Model Selection for Future Predictions in Logistic Regression

While there are many similarities, there are also some differences between model selection in linear versus logistic regression. In particular, not all criteria we saw for linear regression apply to logistic regression. In addition, there are some new criteria that can be used.

Problem: We wish to predict $\pi(x)$ using potential predictor variables $X_1, X_2, \ldots, X_p$, using a logistic regression model.

Challenge: Which subset of the $X_1, X_2, \ldots, X_p$ potential predictors should be included in the model for "best" predictions?

Frequentist Approaches:

- Backwards or forwards or backwards/forwards selection - still applies to logistic regression, but has all of the same problems we saw there: no theoretical basis, p-values do not retain their usual meaning, tends to pick models that are much too large, etc. We do not consider these methods any further.
- All subsets selection - A generic term, where a criterion (such as AIC, BIC, R^2, Adjusted R^2, C_p, etc) can be used. We will mainly compare results from AIC and BIC today.

- AIC criterion - calculated as $AIC = n\ln(SSE) - n\ln(n) + 2p$. Recall that it tends to be good for complex models, less good in finding simple models. Tends to overfit.
- R^2 criterion - Does not apply to logistic regression models, as we do not have the same kind of residuals as in linear models. [In fact, there is a "trick" whereby one can use a linear regression program to fit a logistic regression model, ending up with the same fit as had a logistic regression model been used. This is not too surprising, as except for the logit function, we are really dealing with a (generalized) linear model. So, using these programs, an R^2 measure can in fact be defined for logistic regression models, but it does not work well, and is seldom used in practice.]
- Adjusted R^2 criterion - Same comments as above.
- PRESS$_p$ criterion - Same as above, as based on residuals.
- Mallow's C_p - as in linear regression, based on standardized residuals, and is the method preferred by Hosmer and Lemeshow. See that book for details (formula on page 131).

Bayesian Approaches:

- Bayes Factors - General method that applies to all models, we will continue to use them for logistic regression, but usually in the form of the BIC approximation.
- BIC criterion - As an approximation to a Bayes Factor, this will be our main method for model ordering. We will use the bic.glm software available for R.
- DIC criterion - Available WinBUGS, used mainly for hierarchical models. We will see an example in the next lecture in comparing two hierarchical logistic regression models.

We will now look at a series of examples where we will compare the two main techniques, the AIC and BIC, through several examples. In both cases, we will use routines from R to find the best model or best few models.

The first two examples will use simulated data, where we know the true model. This will allow us to compare methods where we know the truth. The third and last example will use a real data set we have seen before.

Example 1: Large number of covariates, null model is true

As a first example, we will create a large data set with 1000 cases and 30 independent variables, but where no variable in fact is related to the outcome. This will allow us to compare the AIC to the BIC in cases where the true model is small (simple).

```
#  Create 15 dichotomous random independent variables, each with a
#  different rate of "positive" outcomes (i.e., 1's).

> rates <- round(seq(.1, .9, length.out=15), 2)
> rates
 [1] 0.10 0.16 0.21 0.27 0.33 0.39 0.44 0.50 0.56
      0.61 0.67 0.73 0.79 0.84 0.90

#  For each rate, generate a random 0/1 data with that rate.

> x1  <- rbinom(1000, 1, rates[1])
> x2  <- rbinom(1000, 1, rates[2])
> x3  <- rbinom(1000, 1, rates[3])
> x4  <- rbinom(1000, 1, rates[4])
> x5  <- rbinom(1000, 1, rates[5])
> x6  <- rbinom(1000, 1, rates[6])
> x7  <- rbinom(1000, 1, rates[7])
> x8  <- rbinom(1000, 1, rates[8])
> x9  <- rbinom(1000, 1, rates[9])
> x10 <- rbinom(1000, 1, rates[10])
> x11 <- rbinom(1000, 1, rates[11])
> x12 <- rbinom(1000, 1, rates[12])
> x13 <- rbinom(1000, 1, rates[13])
> x14 <- rbinom(1000, 1, rates[14])
> x15 <- rbinom(1000, 1, rates[15])

#  Similarly, create 15 normally distributed random variables.
#  Without loss of generality, mean = 0 and sd = 1 throughout.

> x16 <- rnorm(1000)
> x17 <- rnorm(1000)
> x18 <- rnorm(1000)
> x19 <- rnorm(1000)
> x20 <- rnorm(1000)
> x21 <- rnorm(1000)
```

```
> x22 <- rnorm(1000)
> x23 <- rnorm(1000)
> x24 <- rnorm(1000)
> x25 <- rnorm(1000)
> x26 <- rnorm(1000)
> x27 <- rnorm(1000)
> x28 <- rnorm(1000)
> x29 <- rnorm(1000)
> x30 <- rnorm(1000)

#  [Yes, there may be quicker ways to do this, but the goal
#  here is clarity, not programming tricks.]

#  Now create a random logistic regression variable
#  to use as the outcome or dependent variable, rate = 0.5, say.

y <- rbinom(1000, 1, 0.5)

#  Note that y is NOT related to ANY of the x's

#  Now put all data together into one large data frame:

> example1.dat <- data.frame(y, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10,
                      x11, x12, x13, x14, x15, x16, x17, x18, x19, x20,
                      x21, x22, x23, x24, x25, x26, x27, x28, x29, x30)

#  Check that all data are there

> names(example1.dat)
 [1] "y"   "x1"  "x2"  "x3"  "x4"  "x5"  "x6"  "x7"  "x8"  "x9"  "x10" "x11" "x12"
[14] "x13" "x14" "x15" "x16" "x17" "x18" "x19" "x20" "x21" "x22" "x23" "x24" "x25"
[27] "x26" "x27" "x28" "x29" "x30"

#  Now use the bic.glm program to analyse these data
#  (remembering that the "null" model is in fact correct).

#  Instructions for finding and using bic.glm

#  If you have not already done so, using a computer connected
#  to the internet, download the BMA package from a CRAN site.

#  To do this, open R, and open the menu item titled:
#  "packages --> install package(s)"

#  Pick any site to download from (geographically closer sites may
```

```
#  be faster), and select "BMA".

#  Once it is downloaded, you need to load it for this session
#  (and any future session where you want to use it) by going to the
#  "packages --> load package..." menu item and clicking on BMA.

#  BMA should now be installed and loaded, ready to use.

#  To see the help for BMA, once it is loaded you
#  can always go to the menu item "help --> html help",
#  go to packages on the html page, and find BMA, which lists
#  all BMA functions, including bic.glm and bicreg.

#  Now ready to use bic.glm, whose command line looks like this:

#  bic.glm(f, data, glm.family, wt = rep(1, nrow(data)), strict = FALSE,
#      prior.param = c(rep(0.5, ncol(x))), OR = 20, maxCol = 30, OR.fix = 2,
#      nbest = 150, dispersion = , factor.type = TRUE, factor.prior.adjust = FALSE,
#          occam.window = TRUE, ...)

#  where the main items of interest are:

#  f = a formula

#  data = a data frame containing the variables in the model.

#  glm.family = a description of the error distribution and link function
#     to be used in the model.  For logistic regression, family = binomial
#     link = logit (link implied by default when family = binomial is chosen)

#  prior.param  = a vector of values specifying the prior weights for
#  each variable.  Default is ‘‘noninformative’’, each variable with a 50%
#  chance of being in the model.  Setting different numbers can nudge a
#  variable into or out of the model, as dictated by prior information.
#  Setting a value = 1 forces that variable into the model, as we may
#  want to do when checking confounding for a main variable.


#   OR = a number specifying the maximum ratio for excluding models in
#   Occam's window.  Set higher to include more models, lower to include
#   fewer models.

#  maxCol a number specifying the maximum number of columns in design
#  matrix (including intercept) to be kept.  Note that here we need 31,
#  as we have 30 variables plus an intercept.
```

```
#  So for our example, we type:

> output <- bic.glm(y ~ x1 + x2 + x3 + x4 + x5 + x6 +
   x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 +
   x15 + x16 + x17+ x18 + x19 + x20 + x21 +
   x22 + x23 + x24 + x25 + x26 +
   x27 + x28 + x29 + x30, glm.family="binomial",
   data=example1.dat, maxCol = 31)

#  We can now look at various parts of the output,
#  starting with the default output via the usual
#  summary command:

> summary(output)

Call:
bic.glm.formula(f = y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 +
     x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 +
     x19 + x20 + x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 +
     x29 + x30, data = example1.dat, glm.family = "binomial",
     maxCol = 31)


  10  models were selected
 Best  5  models (cumulative posterior probability =  0.8353 ):

     p!=0   EV        SD       model 1     model 2     model 3     model 4     model 5
Int  100   -0.077661  0.16872  -1.200e-02  -5.288e-02  -3.887e-01  -1.823e-01   8.444e-02
x1   15.1   0.067837  0.18259        .       4.493e-01        .           .           .
x2    0.0   0.000000  0.00000        .            .           .           .           .
x3    0.0   0.000000  0.00000        .            .           .           .           .
x4    0.0   0.000000  0.00000        .            .           .           .           .
x5    0.0   0.000000  0.00000        .            .           .           .           .
x6    0.0   0.000000  0.00000        .            .           .           .           .
x7    0.0   0.000000  0.00000        .            .           .           .           .
x8    4.5  -0.008263  0.04660        .            .           .           .      -1.834e-01
x9    0.0   0.000000  0.00000        .            .           .           .           .
x10   0.0   0.000000  0.00000        .            .           .           .           .
x11   0.0   0.000000  0.00000        .            .           .           .           .
x12   5.8   0.013347  0.06406        .            .           .       2.312e-01       .
x13   3.0   0.005145  0.03955        .            .           .           .           .
x14   0.0   0.000000  0.00000        .            .           .           .           .
x15  13.3   0.055232  0.16162        .            .       4.175e-01       .           .
x16   0.0   0.000000  0.00000        .            .           .           .           .
```

```
x17   0.0   0.000000  0.00000       .          .          .          .          .
x18   4.4   0.004015  0.02288       .          .          .          .          .
x19   0.0   0.000000  0.00000       .          .          .          .          .
x20   0.0   0.000000  0.00000       .          .          .          .          .
x21   0.0   0.000000  0.00000       .          .          .          .          .
x22   0.0   0.000000  0.00000       .          .          .          .          .
x23   0.0   0.000000  0.00000       .          .          .          .          .
x24   3.2   0.002490  0.01787       .          .          .          .          .
x25   0.0   0.000000  0.00000       .          .          .          .          .
x26   0.0   0.000000  0.00000       .          .          .          .          .
x27   0.0   0.000000  0.00000       .          .          .          .          .
x28   0.0   0.000000  0.00000       .          .          .          .          .
x29   3.2  -0.002455  0.01757       .          .          .          .          .
x30   0.0   0.000000  0.00000       .          .          .          .          .

nVar                              0          1          1          1          1
BIC                        -5.515e+03 -5.512e+03 -5.512e+03 -5.510e+03 -5.510e+03
post prob                       0.500      0.126      0.107      0.058      0.045

#  From this, we can see that the null model indeed comes up as best
#  according to the BIC, with posterior probability of 0.5 or 50%.

#  Second best model had about 4 times less probability, about 12.6%
#  and had just a single variable in it, x1.  None of the five best models
#  even had two variables in it.

#  Can look at several other available items:

#  Posterior probability of each of 10 best models (rest very small by
#   comparison, so are omitted, change value of OR to see them)

> output$postprob
 [1] 0.50014815 0.12552735 0.10684804 0.05771828 0.04505754
     0.04437032 0.03248863 0.03235586 0.02980426 0.02568158

#  What variables were in each of above 10 models

> output$label
 [1] "NULL"   "x1"     "x15"    "x12"    "x8"
      "x18"    "x29"    "x24"    "x13"    "x1,x15"

#  For each of 30 variables, probability they should be in the model
#  Note that largest is 15.1%, quite small.
#  Note straightforward interpretation compared to p-values
```

```
> output$probne0
 [1]  15.1  0.0  0.0  0.0  0.0  0.0  0.0  4.5  0.0  0.0  0.0  5.8  3.0
 [14] 0.0 13.3  0.0  0.0  4.4  0.0  0.0  0.0  0.0  0.0  3.2  0.0  0.0
 [27] 0.0 0.0  3.2  0.0

#  Bayesian model averaged means for each variable.  All very near 0 except intercept.

> output$postmean
[1] -0.077661308 0.067836687 0.000000000 0.000000000 0.000000000
    0.000000000 0.000000000 0.000000000 -0.008262573 0.000000000 0.000000000
    0.000000000 0.013347027 0.005145117 0.000000000 0.055232349 0.000000000
    0.000000000 0.004014608 0.000000000 0.000000000 0.000000000 0.000000000
    0.000000000 0.002489726 0.000000000 0.000000000 0.000000000 0.000000000
    -0.002454815 0.000000000

#  Bayesian model averaged SDs for each variable.

> output$postsd
[1] 0.16871507 0.18258761 0.00000000 0.00000000 0.00000000 0.00000000
    0.00000000 0.00000000 0.04659719 0.00000000 0.00000000 0.00000000
    0.06406213 0.03954787 0.00000000 0.16161682 0.00000000 0.00000000
    0.02288460 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
    0.01787378 0.00000000 0.00000000 0.00000000 0.00000000 0.01756609
    0.00000000

#  For each of top 10 models (in this case), model by model estimates
#  This is where you can check for confounding, very conveniently.

> output$mle
              [,1]      [,2] [,3] [,4] [,5] [,6] [,7] [,8]       [,9]
 [1,] -0.01200014 0.0000000    0    0    0    0    0    0  0.0000000
 [2,] -0.05287575 0.4492910    0    0    0    0    0    0  0.0000000
 [3,] -0.38865795 0.0000000    0    0    0    0    0    0  0.0000000
 [4,] -0.18232156 0.0000000    0    0    0    0    0    0  0.0000000
 [5,]  0.08443832 0.0000000    0    0    0    0    0    0 -0.1833783
 [6,] -0.01497873 0.0000000    0    0    0    0    0    0  0.0000000
 [7,] -0.01151393 0.0000000    0    0    0    0    0    0  0.0000000
 [8,] -0.01309456 0.0000000    0    0    0    0    0    0  0.0000000
 [9,] -0.14705342 0.0000000    0    0    0    0    0    0  0.0000000
[10,] -0.42563099 0.4453925    0    0    0    0    0    0  0.0000000

[,10] [,11] [,12]     [,13]     [,14] [,15]     [,16] [,17] [,18]
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.4175168     0     0
```

```
    0     0     0 0.2312444 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
    0     0     0 0.0000000 0.1726303     0 0.0000000     0     0
    0     0     0 0.0000000 0.0000000     0 0.4135844     0     0

     [,19] [,20] [,21] [,22] [,23] [,24]      [,25] [,26] [,27] [,28]
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.09047959     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.07694822     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0

 [,29]       [,30] [,31]
     0  0.00000000     0
     0  0.00000000     0
     0  0.00000000     0
     0  0.00000000     0
     0  0.00000000     0
     0  0.00000000     0
     0 -0.07555923     0
     0  0.00000000     0
     0  0.00000000     0
     0  0.00000000     0

> output$se
            [,1]      [,2] [,3] [,4] [,5] [,6] [,7] [,8]
 [1,] 0.06324669 0.0000000    0    0    0    0    0    0 0
 [2,] 0.06639553 0.2227353    0    0    0    0    0    0 0
 [3,] 0.20480187 0.0000000    0    0    0    0    0    0 0
 [4,] 0.12360287 0.0000000    0    0    0    0    0    0 0
 [5,] 0.09194491 0.0000000    0    0    0    0    0    0 0
 [6,] 0.06334641 0.0000000    0    0    0    0    0    0 0
 [7,] 0.06329347 0.0000000    0    0    0    0    0    0 0
 [8,] 0.06329881 0.0000000    0    0    0    0    0    0 0
 [9,] 0.13582320 0.0000000    0    0    0    0    0    0 0
[10,] 0.20605504 0.2231372    0    0    0    0    0    0 0
```

```
    [,9] [,10] [,11] [,12]     [,13]     [,14] [,15]     [,16] [,17] [,18]
.0000000     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.2153700     0     0
.0000000     0     0     0 0.1439295 0.0000000     0 0.0000000     0     0
.1267953     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.1535053     0 0.0000000     0     0
.0000000     0     0     0 0.0000000 0.0000000     0 0.2157906     0     0

     [,19] [,20] [,21] [,22] [,23] [,24]      [,25] [,26] [,27] [,28]
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.06308512     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.06437585     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0
0.00000000     0     0     0     0     0 0.00000000     0     0     0

 [,29]      [,30] [,31]
 [1,]     0 0.00000000     0
 [2,]     0 0.00000000     0
 [3,]     0 0.00000000     0
 [4,]     0 0.00000000     0
 [5,]     0 0.00000000     0
 [6,]     0 0.00000000     0
 [7,]     0 0.06303965     0
 [8,]     0 0.00000000     0
 [9,]     0 0.00000000     0
[10,]     0 0.00000000     0

#  Overall, we can see that the BIC does very well, picking out the correct model
#  with high probability.

#  Let's see how AIC compares here.

> output.aic <- glm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 +
+      x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 +
+      x19 + x20 + x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 +
+      x29 + x30, data = example1.dat, family = "binomial")
```

```
> summary(output.aic)

Call:
glm(formula = y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 +
    x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 + x19 +
    x20 + x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 + x29 +
    x30, family = "binomial", data = example1.dat)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5605  -1.1480  -0.8262   1.1566   1.5755

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.475393   0.360206  -1.320   0.1869
x1           0.489555   0.228073   2.146   0.0318 *
x2          -0.129563   0.176738  -0.733   0.4635
x3           0.132800   0.161390   0.823   0.4106
x4          -0.087652   0.149033  -0.588   0.5564
x5          -0.019543   0.139165  -0.140   0.8883
x6           0.026090   0.134244   0.194   0.8459
x7          -0.091074   0.130886  -0.696   0.4865
x8          -0.199179   0.130822  -1.523   0.1279
x9          -0.028450   0.130797  -0.218   0.8278
x10         -0.077993   0.131511  -0.593   0.5531
x11          0.001851   0.138874   0.013   0.9894
x12          0.211354   0.147272   1.435   0.1513
x13          0.199699   0.157654   1.267   0.2053
x14         -0.071509   0.171926  -0.416   0.6775
x15          0.416499   0.220038   1.893   0.0584 .
x16          0.025160   0.064798   0.388   0.6978
x17          0.057351   0.064573   0.888   0.3745
x18          0.078949   0.064791   1.219   0.2230
x19          0.002473   0.064916   0.038   0.9696
x20          0.012197   0.064973   0.188   0.8511
x21         -0.060903   0.064230  -0.948   0.3430
x22         -0.032764   0.067307  -0.487   0.6264
x23          0.035026   0.061918   0.566   0.5716
x24          0.091855   0.066616   1.379   0.1679
x25         -0.022096   0.062603  -0.353   0.7241
x26         -0.037016   0.063202  -0.586   0.5581
x27          0.029595   0.065374   0.453   0.6508
x28         -0.048848   0.065923  -0.741   0.4587
x29         -0.069790   0.064941  -1.075   0.2825
x30         -0.028556   0.063286  -0.451   0.6518
```

```
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1386.3  on 999  degrees of freedom
Residual deviance: 1360.6  on 969  degrees of freedom
AIC: 1422.6

Number of Fisher Scoring iterations: 4

> step.aic <- step(output.aic)
Start:  AIC= 1422.56
 y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 +
    x12 + x13 + x14 + x15 + x16 + x17 + x18 + x19 + x20 + x21 +
    x22 + x23 + x24 + x25 + x26 + x27 + x28 + x29 + x30

       Df Deviance    AIC
- x11   1   1360.6 1420.6
- x19   1   1360.6 1420.6
- x5    1   1360.6 1420.6
- x20   1   1360.6 1420.6
- x6    1   1360.6 1420.6
- x9    1   1360.6 1420.6
- x25   1   1360.7 1420.7
- x16   1   1360.7 1420.7
- x14   1   1360.7 1420.7
- x30   1   1360.8 1420.8
- x27   1   1360.8 1420.8
- x22   1   1360.8 1420.8
- x23   1   1360.9 1420.9
- x26   1   1360.9 1420.9
- x4    1   1360.9 1420.9
- x10   1   1360.9 1420.9
- x7    1   1361.0 1421.0
- x2    1   1361.1 1421.1
- x28   1   1361.1 1421.1
- x3    1   1361.2 1421.2
- x17   1   1361.3 1421.3
- x21   1   1361.5 1421.5
- x29   1   1361.7 1421.7
- x18   1   1362.0 1422.0
- x13   1   1362.2 1422.2
- x24   1   1362.5 1422.5
<none>      1360.6 1422.6
```

```
- x12   1   1362.6 1422.6
- x8    1   1362.9 1422.9
- x15   1   1364.2 1424.2
- x1    1   1365.3 1425.3

Step:  AIC= 1420.56
 y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x12 +
    x13 + x14 + x15 + x16 + x17 + x18 + x19 + x20 + x21 + x22 +
    x23 + x24 + x25 + x26 + x27 + x28 + x29 + x30

# ..........etc..... few hundred lines deleted here

#  Last couple of steps

Step:  AIC= 1383.78
 y ~ x1 + x8 + x12 + x15

       Df Deviance    AIC
- x8    1   1375.7 1383.7
<none>      1373.8 1383.8
- x12   1   1376.2 1384.2
- x15   1   1377.1 1385.1
- x1    1   1378.4 1386.4

Step:  AIC= 1383.74
 y ~ x1 + x12 + x15

       Df Deviance    AIC
<none>      1375.7 1383.7
- x12   1   1378.4 1384.4
- x15   1   1379.2 1385.2
- x1    1   1380.1 1386.1

#  So three variables make the final model
#  according to the AIC

#  As expected, the model is too large.

#  Can also just ask for a summary of the AIC
#  output

> summary(step.aic)

Call:
glm(formula = y ~ x1 + x12 + x15, family = "binomial", data = example1.dat)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4002  -1.1973  -0.9403   1.1576   1.4347

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.5871     0.2295  -2.559   0.0105 *
x1            0.4631     0.2238   2.069   0.0385 *
x12           0.2348     0.1449   1.621   0.1050
x15           0.3990     0.2162   1.846   0.0649 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1386.3  on 999  degrees of freedom
Residual deviance: 1375.7  on 996  degrees of freedom
AIC: 1383.7

Number of Fisher Scoring iterations: 4
```

Overall conclusion: As expected, when small models are "correct", the BIC does better compared to the AIC, the latter tending to provide too many variables in the "best" model.

Example 2: Large number of covariates, some covariates are important

For the second example, we will create a large data set with 500 cases and 20 independent variables, but where only 10 of the 20 variables are in fact related to the outcome. This will allow us to compare the AIC to the BIC in cases where the true model is large (complex).

```
#  Create 10 dichotomous random independent variables, each with a
#  different rate of ``positive'' outcomes (i.e., 1's).

>  rates <- round(seq(.1, .9, length.out=10), 2)
>  rates
```

```
#  For each rate, generate a random 0/1 data with that rate.

> x1  <- rbinom(500,  1, rates[1])
> x2  <- rbinom(500,  1, rates[2])
> x3  <- rbinom(500,  1, rates[3])
> x4  <- rbinom(500,  1, rates[4])
> x5  <- rbinom(500,  1, rates[5])
> x6  <- rbinom(500,  1, rates[6])
> x7  <- rbinom(500,  1, rates[7])
> x8  <- rbinom(500,  1, rates[8])
> x9  <- rbinom(500,  1, rates[9])
> x10 <- rbinom(500,  1, rates[10])

#  Similarly, create 10 normally distributed random variables.
#  Without loss of generality, mean = 0 and sd = 1 throughout.

> x11 <- rnorm(500)
> x12 <- rnorm(500)
> x13 <- rnorm(500)
> x14 <- rnorm(500)
> x15 <- rnorm(500)
> x16 <- rnorm(500)
> x17 <- rnorm(500)
> x18 <- rnorm(500)
> x19 <- rnorm(500)
> x20 <- rnorm(500)

#  Now create a logistic regression variable
#  to use as the outcome or dependent variable,
#  where half but not the other half are related.

> inv.logit.rate <- exp(x1 + x2 + x3 + x4 + x5 + x11 + x12 + x13 + x14 +x15)/
                (1+ exp(x1 + x2 + x3 + x4 + x5 + x11 + x12 + x13 + x14 +x15))

> y <- rbinom(500, 1, inv.logit.rate)

#  Now put all data together into one large data frame:

> example2.dat <- data.frame(y, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10,
                      x11, x12, x13, x14, x15, x16, x17, x18, x19, x20)

#  Run the BIC program for these data

output <- bic.glm(y ~ x1 + x2 + x3 + x4 + x5 + x6 +
```

```
   x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 +
   x15 + x16 + x17+ x18 + x19 + x20, glm.family="binomial",
   data=example2.dat)

#  We can now look at various parts of the output,
#  starting with the default output via the usual
#  summary command:

>
Call:
bic.glm.formula(f = y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 +
     x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 +
     x19 + x20, data = example2.dat, glm.family = "binomial")

  41  models were selected
 Best  5  models (cumulative posterior probability =  0.4373 ):

    p!=0    EV        SD       model 1   model 2   model 3   model 4    model 5
Int  100   0.200618  0.40358   0.5211    0.1618    0.5426    0.1754    -0.1101
x1   1.9   0.008797  0.09268     .         .         .         .         .
x2  96.5   1.009675  0.37546   1.0604    0.9965    1.1111    1.0397     0.9965
x3  11.3   0.056628  0.18570     .         .         .         .         .
x4  94.3   0.789229  0.32499   0.7999    0.8544    0.8069    0.8633     0.8465
x5  25.5   0.135280  0.26403     .         .         .         .        0.5331
x6   0.0   0.000000  0.00000     .         .         .         .         .
x7  46.5   0.293646  0.36097     .       0.6155      .       0.6361     0.6469
x8   0.0   0.000000  0.00000     .         .         .         .         .
x9  17.8   0.110463  0.27226     .         .         .         .         .
x10  1.7   0.005929  0.06912     .         .         .         .         .
x11 100.0  0.809662  0.14696   0.7879    0.8194    0.8145    0.8488     0.8286
x12 100.0  0.955404  0.16193   0.9271    0.9673    0.9692    1.0138     0.9663
x13 100.0  0.864077  0.14863   0.8434    0.8614    0.8794    0.8993     0.8841
x14 100.0  1.146932  0.14792   1.1164    1.1535    1.1610    1.1998     1.1601
x15 100.0  0.786048  0.13905   0.7647    0.7816    0.8039    0.8217     0.7996
x16  23.7 -0.062255  0.12736     .         .       -0.2605   -0.2709      .
x17   1.9 -0.002357  0.02426     .         .         .         .         .
x18   0.8 -0.000736  0.01345     .         .         .         .         .
x19   1.6 -0.001559  0.01968     .         .         .         .         .
x20   1.9  0.002309  0.02359     .         .         .         .         .

nVar                            7         8         8         9         9
BIC                         -2639.5804 -2639.2389 -2637.7986 -2637.7884 -2637.6608
post prob                    0.144      0.121      0.059      0.059      0.055

#  Can look at several other available items:
```

```
#  Posterior probability of each of 41 best models (rest very small by
#   comparison, so are omitted, change value of OR to see them)

> output$postprob
 [1] 0.143619429 0.121076255 0.058925211 0.058627170 0.055002999 0.050963272 0.039575647
 [8] 0.033376075 0.029701648 0.026427144 0.024799700 0.023619779 0.023611427 0.020953606
[15] 0.017934960 0.017533391 0.016849682 0.016508442 0.013109733 0.013083984 0.012198185
[22] 0.011082183 0.010717331 0.010390403 0.010116726 0.010025525 0.009970218 0.009880258
[29] 0.009318278 0.008870916 0.008829660 0.008672594 0.008658077 0.008494251 0.008451433
[36] 0.008344194 0.008260385 0.008193036 0.008141221 0.008116599 0.007968973

#  What variables were in each of above 41 models

> output$label
 [1] "x2,x4,x11,x12,x13,x14,x15"              "x2,x4,x7,x11,x12,x13,x14,x15"
 [3] "x2,x4,x11,x12,x13,x14,x15,x16"          "x2,x4,x7,x11,x12,x13,x14,x15,x16"
 [5] "x2,x4,x5,x7,x11,x12,x13,x14,x15"        "x2,x4,x5,x11,x12,x13,x14,x15"
 [7] "x2,x4,x9,x11,x12,x13,x14,x15"           "x2,x3,x4,x11,x12,x13,x14,x15"
 [9] "x2,x4,x7,x9,x11,x12,x13,x14,x15"        "x2,x3,x4,x7,x11,x12,x13,x14,x15"
[11] "x2,x11,x12,x13,x14,x15"                 "x2,x4,x5,x9,x11,x12,x13,x14,x15"
[13] "x2,x4,x5,x7,x9,x11,x12,x13,x14,x15"     "x2,x4,x5,x7,x11,x12,x13,x14,x15,x16"
[15] "x2,x4,x9,x11,x12,x13,x14,x15,x16"       "x4,x7,x11,x12,x13,x14,x15"
[17] "x2,x4,x5,x11,x12,x13,x14,x15,x16"       "x2,x4,x7,x9,x11,x12,x13,x14,x15,x16"
[19] "x2,x3,x4,x5,x11,x12,x13,x14,x15"        "x2,x3,x4,x5,x7,x11,x12,x13,x14,x15"
[21] "x2,x7,x11,x12,x13,x14,x15"              "x2,x4,x11,x12,x13,x14,x15,x20"
[23] "x1,x2,x4,x11,x12,x13,x14,x15"           "x2,x5,x11,x12,x13,x14,x15"
[25] "x2,x4,x5,x7,x9,x11,x12,x13,x14,x15,x16" "x2,x11,x12,x13,x14,x15,x16"
[27] "x2,x3,x4,x11,x12,x13,x14,x15,x16"       "x2,x4,x7,x11,x12,x13,x14,x15,x17"
[29] "x2,x4,x11,x12,x13,x14,x15,x17"          "x4,x11,x12,x13,x14,x15"
[31] "x2,x3,x4,x7,x11,x12,x13,x14,x15,x16"    "x2,x4,x7,x10,x11,x12,x13,x14,x15"
[33] "x4,x5,x7,x11,x12,x13,x14,x15"           "x2,x4,x10,x11,x12,x13,x14,x15"
[35] "x2,x4,x11,x12,x13,x14,x15,x18"          "x2,x4,x5,x9,x11,x12,x13,x14,x15,x16"
[37] "x2,x4,x11,x12,x13,x14,x15,x19"          "x2,x4,x7,x11,x12,x13,x14,x15,x20"
[39] "x1,x2,x4,x7,x11,x12,x13,x14,x15"        "x2,x3,x4,x9,x11,x12,x13,x14,x15"
[41] "x2,x4,x7,x11,x12,x13,x14,x15,x19"

#  Note that best model is close to correct, but missing
#  x1, x3, and x5.  Not surprising, as these all had relatively small positive rates,
#  so while they did have an effect, OR = exp(1) = 2.7, there were few subjects with
#  these covariates = 1, so hard to detect.

#  For each of 20 variables, probability they should be in the model
#  Note that largest is 100, for all continuous variables, dichotomous
```

```
#  Note straightforward interpretation compared to p-values

> output$probne0
 [1]   1.9  96.5  11.3  94.3  25.5   0.0  46.5   0.0  17.8
         1.7 100.0 100.0 100.0 100.0
[15] 100.0  23.7   1.9   0.8   1.6   1.9

#  On an individual level, only x1 and x3 do poorly.
#  Note that continuous variables do better than dichotomous
#  as they carry more information per "individual" point.

#  Bayesian model averaged means for each variable.

>  output$postmean
 [1]  0.2006184649  0.0087971269  1.0096750218  0.0566284656  0.7892289254  0.1352797851
 [7]  0.0000000000  0.2936460982  0.0000000000  0.1104627990  0.0059291998  0.8096621422
[13]  0.9554038942  0.8640772982  1.1469315456  0.7860478118 -0.0622552374 -0.0023572089
[19] -0.0007360552 -0.0015587321  0.0023092062


#  Can also Bayesian model averaged SDs for each variable (omitted here).

#  For each of top 41 models (in this case), model by model estimates
#  This is where you can check for confounding, very conveniently.

>  output$mle
             [,1]      [,2]      [,3]      [,4]      [,5]      [,6] [,7]      [,8] [,9]
 [1,]  0.52109736 0.0000000 1.0603966 0.0000000 0.7998570 0.0000000    0 0.0000000    0
 [2,]  0.16184360 0.0000000 0.9965460 0.0000000 0.8544038 0.0000000    0 0.6155354    0
 [3,]  0.54257964 0.0000000 1.1110882 0.0000000 0.8068503 0.0000000    0 0.0000000    0
 [4,]  0.17540152 0.0000000 1.0397053 0.0000000 0.8633237 0.0000000    0 0.6360559    0
 [5,] -0.11013345 0.0000000 0.9965406 0.0000000 0.8464877 0.5330818    0 0.6469364    0
 [6,]  0.28612647 0.0000000 1.0655008 0.0000000 0.7893717 0.4994360    0 0.0000000    0
 [7,]  0.03502827 0.0000000 1.0681268 0.0000000 0.8437805 0.0000000    0 0.0000000    0
 [8,]  0.38561535 0.0000000 1.0538024 0.5110045 0.8016314 0.0000000    0 0.0000000    0
 [9,] -0.30715263 0.0000000 1.0036449 0.0000000 0.8975954 0.0000000    0 0.6065672    0
[10,]  0.03035734 0.0000000 0.9875649 0.5035636 0.8572636 0.0000000    0 0.6126204    0
[11,]  0.78900593 0.0000000 1.0932387 0.0000000 0.0000000 0.0000000    0 0.0000000    0
[12,] -0.30353526 0.0000000 1.0771339 0.0000000 0.8452567 0.5668040    0 0.0000000    0
[13,] -0.69406124 0.0000000 1.0069348 0.0000000 0.9020166 0.6029347    0 0.6433542    0
[14,] -0.08595767 0.0000000 1.0430923 0.0000000 0.8627812 0.5083318    0 0.6641209    0
[15,]  0.03548613 0.0000000 1.1185256 0.0000000 0.8580677 0.0000000    0 0.0000000    0
[16,]  0.29730456 0.0000000 0.0000000 0.0000000 0.8825100 0.0000000    0 0.6891885    0
[17,]  0.31633089 0.0000000 1.1193577 0.0000000 0.8040250 0.4758058    0 0.0000000    0
[18,] -0.32163438 0.0000000 1.0465007 0.0000000 0.9146920 0.0000000    0 0.6315418    0
```

```
[19,]  0.13599841 0.0000000 1.0575502 0.5281764 0.7962513 0.5134563    0 0.0000000    0
[20,] -0.25354577 0.0000000 0.9851336 0.5186411 0.8530354 0.5448057    0 0.6428679    0
[21,]  0.47995446 0.0000000 1.0331320 0.0000000 0.0000000 0.0000000    0 0.5486796    0
[22,]  0.51595563 0.0000000 1.0805057 0.0000000 0.8188289 0.0000000    0 0.0000000    0
[23,]  0.48563248 0.4859330 1.0695793 0.0000000 0.8026192 0.0000000    0 0.0000000    0
[24,]  0.54747841 0.0000000 1.0936326 0.0000000 0.0000000 0.5131421    0 0.0000000    0
[25,] -0.69378801 0.0000000 1.0550640 0.0000000 0.9276357 0.5794530    0 0.6661299    0
[26,]  0.81136765 0.0000000 1.1365860 0.0000000 0.0000000 0.0000000    0 0.0000000    0
[27,]  0.41841634 0.0000000 1.1032690 0.4636722 0.8064904 0.0000000    0 0.0000000    0
[28,]  0.14034984 0.0000000 0.9751167 0.0000000 0.8679500 0.0000000    0 0.6441564    0
[29,]  0.51850191 0.0000000 1.0480891 0.0000000 0.8076552 0.0000000    0 0.0000000    0
[30,]  0.71342967 0.0000000 0.0000000 0.0000000 0.8287357 0.0000000    0 0.0000000    0
[31,]  0.06035904 0.0000000 1.0307759 0.4456663 0.8635071 0.0000000    0 0.6272904    0
[32,] -0.19431723 0.0000000 1.0101375 0.0000000 0.8399607 0.0000000    0 0.6379304    0
[33,]  0.02350100 0.0000000 0.0000000 0.0000000 0.8746574 0.5349497    0 0.7200006    0
[34,]  0.25772285 0.0000000 1.0709885 0.0000000 0.7876556 0.0000000    0 0.0000000    0
[35,]  0.53135737 0.0000000 1.0457663 0.0000000 0.7974878 0.0000000    0 0.0000000    0
[36,] -0.28820073 0.0000000 1.1325994 0.0000000 0.8677098 0.5426817    0 0.0000000    0
[37,]  0.52114665 0.0000000 1.0497010 0.0000000 0.7964433 0.0000000    0 0.0000000    0
[38,]  0.16333229 0.0000000 1.0148708 0.0000000 0.8725646 0.0000000    0 0.6036898    0
[39,]  0.13628676 0.4408701 1.0042009 0.0000000 0.8558310 0.0000000    0 0.6056520    0
[40,] -0.08244702 0.0000000 1.0572847 0.4924792 0.8435062 0.0000000    0 0.0000000    0
[41,]  0.15243395 0.0000000 0.9827362 0.0000000 0.8526890 0.0000000    0 0.6317700    0
          [,10]     [,11]     [,12]     [,13]     [,14]     [,15]     [,16]      [,17]
 [1,] 0.0000000 0.0000000 0.7879140 0.9271193 0.8433824 1.116424 0.7647157  0.0000000
 [2,] 0.0000000 0.0000000 0.8194133 0.9672917 0.8614397 1.153546 0.7815934  0.0000000
 [3,] 0.0000000 0.0000000 0.8145336 0.9692156 0.8794265 1.161034 0.8038507 -0.2605132
 [4,] 0.0000000 0.0000000 0.8487759 1.0137508 0.8993280 1.199832 0.8217401 -0.2709165
 [5,] 0.0000000 0.0000000 0.8286481 0.9662654 0.8841281 1.160054 0.7995718  0.0000000
 [6,] 0.0000000 0.0000000 0.7932019 0.9247414 0.8649361 1.121975 0.7787222  0.0000000
 [7,] 0.5882341 0.0000000 0.7913227 0.9464491 0.8462609 1.119599 0.7700681  0.0000000
 [8,] 0.0000000 0.0000000 0.8050408 0.9101998 0.8347850 1.135287 0.7520082  0.0000000
 [9,] 0.5741696 0.0000000 0.8218361 0.9859837 0.8666542 1.155330 0.7884871  0.0000000
[10,] 0.0000000 0.0000000 0.8362665 0.9542092 0.8556382 1.172856 0.7680951  0.0000000
[11,] 0.0000000 0.0000000 0.7415368 0.8761449 0.8068846 1.095023 0.7588531  0.0000000
[12,] 0.6752666 0.0000000 0.8004574 0.9487587 0.8750768 1.129815 0.7871209  0.0000000
[13,] 0.6702897 0.0000000 0.8349418 0.9903347 0.8981233 1.166679 0.8102970  0.0000000
[14,] 0.0000000 0.0000000 0.8592108 1.0129464 0.9224491 1.206402 0.8390262 -0.2590345
[15,] 0.6112889 0.0000000 0.8198042 0.9904282 0.8860572 1.164661 0.8113136 -0.2665460
[16,] 0.0000000 0.0000000 0.7675848 0.9679936 0.8360673 1.126310 0.7833738  0.0000000
[17,] 0.0000000 0.0000000 0.8210481 0.9673953 0.9011180 1.167638 0.8171496 -0.2491791
[18,] 0.6037300 0.0000000 0.8536340 1.0357415 0.9087589 1.202305 0.8314123 -0.2796497
[19,] 0.0000000 0.0000000 0.8117870 0.9080732 0.8539211 1.144109 0.7665482  0.0000000
[20,] 0.0000000 0.0000000 0.8466588 0.9537951 0.8753825 1.182530 0.7866014  0.0000000
[21,] 0.0000000 0.0000000 0.7655172 0.9058468 0.8174293 1.125036 0.7692881  0.0000000
```

```
[22,] 0.0000000 0.0000000 0.8003989 0.9355278 0.8457208 1.121442 0.7506775  0.0000000
[23,] 0.0000000 0.0000000 0.7960178 0.9198185 0.8454121 1.117049 0.7779392  0.0000000
[24,] 0.0000000 0.0000000 0.7467552 0.8723057 0.8307546 1.101895 0.7736424  0.0000000
[25,] 0.6933078 0.0000000 0.8688874 1.0411504 0.9417534 1.213913 0.8537180 -0.2671697
[26,] 0.0000000 0.0000000 0.7655193 0.9135205 0.8389166 1.136587 0.7992395 -0.2534832
[27,] 0.0000000 0.0000000 0.8286000 0.9495861 0.8695115 1.172966 0.7900886 -0.2430084
[28,] 0.0000000 0.0000000 0.8271769 0.9798484 0.8724192 1.169355 0.7924736  0.0000000
[29,] 0.0000000 0.0000000 0.7931632 0.9338992 0.8523930 1.126453 0.7731496  0.0000000
[30,] 0.0000000 0.0000000 0.7282424 0.9212929 0.8128451 1.079769 0.7702705  0.0000000
[31,] 0.0000000 0.0000000 0.8613398 0.9969333 0.8910985 1.211064 0.8067948 -0.2509693
[32,] 0.0000000 0.3905781 0.8200452 0.9777264 0.8679187 1.155935 0.7903280  0.0000000
[33,] 0.0000000 0.0000000 0.7753438 0.9668440 0.8579034 1.130248 0.7996948  0.0000000
[34,] 0.0000000 0.2992465 0.7885326 0.9332238 0.8479668 1.117382 0.7705015  0.0000000
[35,] 0.0000000 0.0000000 0.7841718 0.9344848 0.8456148 1.116978 0.7634036  0.0000000
[36,] 0.6909714 0.0000000 0.8306965 0.9935066 0.9156689 1.175868 0.8283640 -0.2534698
[37,] 0.0000000 0.0000000 0.7847852 0.9361261 0.8500205 1.117796 0.7702201  0.0000000
[38,] 0.0000000 0.0000000 0.8300034 0.9743316 0.8639278 1.157946 0.7685762  0.0000000
[39,] 0.0000000 0.0000000 0.8275094 0.9594162 0.8640868 1.152759 0.7928672  0.0000000
[40,] 0.5720914 0.0000000 0.8068582 0.9295995 0.8371197 1.137117 0.7578981  0.0000000
[41,] 0.0000000 0.0000000 0.8166366 0.9813470 0.8699659 1.156972 0.7894999  0.0000000
          [,18]       [,19]      [,20]     [,21]
 [1,]  0.0000000  0.00000000  0.0000000 0.0000000
 [2,]  0.0000000  0.00000000  0.0000000 0.0000000
 [3,]  0.0000000  0.00000000  0.0000000 0.0000000
 [4,]  0.0000000  0.00000000  0.0000000 0.0000000
 [5,]  0.0000000  0.00000000  0.0000000 0.0000000
 [6,]  0.0000000  0.00000000  0.0000000 0.0000000
 [7,]  0.0000000  0.00000000  0.0000000 0.0000000
 [8,]  0.0000000  0.00000000  0.0000000 0.0000000
 [9,]  0.0000000  0.00000000  0.0000000 0.0000000
[10,]  0.0000000  0.00000000  0.0000000 0.0000000
[11,]  0.0000000  0.00000000  0.0000000 0.0000000
[12,]  0.0000000  0.00000000  0.0000000 0.0000000
[13,]  0.0000000  0.00000000  0.0000000 0.0000000
[14,]  0.0000000  0.00000000  0.0000000 0.0000000
[15,]  0.0000000  0.00000000  0.0000000 0.0000000
[16,]  0.0000000  0.00000000  0.0000000 0.0000000
[17,]  0.0000000  0.00000000  0.0000000 0.0000000
[18,]  0.0000000  0.00000000  0.0000000 0.0000000
[19,]  0.0000000  0.00000000  0.0000000 0.0000000
[20,]  0.0000000  0.00000000  0.0000000 0.0000000
[21,]  0.0000000  0.00000000  0.0000000 0.0000000
[22,]  0.0000000  0.00000000  0.0000000 0.1260138
[23,]  0.0000000  0.00000000  0.0000000 0.0000000
[24,]  0.0000000  0.00000000  0.0000000 0.0000000
```

```
[25,]  0.0000000  0.00000000  0.0000000 0.0000000
[26,]  0.0000000  0.00000000  0.0000000 0.0000000
[27,]  0.0000000  0.00000000  0.0000000 0.0000000
[28,] -0.1375536  0.00000000  0.0000000 0.0000000
[29,] -0.1071168  0.00000000  0.0000000 0.0000000
[30,]  0.0000000  0.00000000  0.0000000 0.0000000
[31,]  0.0000000  0.00000000  0.0000000 0.0000000
[32,]  0.0000000  0.00000000  0.0000000 0.0000000
[33,]  0.0000000  0.00000000  0.0000000 0.0000000
[34,]  0.0000000  0.00000000  0.0000000 0.0000000
[35,]  0.0000000 -0.08709235  0.0000000 0.0000000
[36,]  0.0000000  0.00000000  0.0000000 0.0000000
[37,]  0.0000000  0.00000000 -0.0851032 0.0000000
[38,]  0.0000000  0.00000000  0.0000000 0.1113993
[39,]  0.0000000  0.00000000  0.0000000 0.0000000
[40,]  0.0000000  0.00000000  0.0000000 0.0000000
[41,]  0.0000000  0.00000000 -0.1073848 0.0000000
>
>  output$se
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6] [,7]      [,8] [,9]
 [1,] 0.1638423 0.0000000 0.3217373 0.0000000 0.2619917 0.0000000    0 0.0000000    0
 [2,] 0.2194563 0.0000000 0.3252583 0.0000000 0.2663878 0.0000000    0 0.2560554    0
 [3,] 0.1650554 0.0000000 0.3260774 0.0000000 0.2639929 0.0000000    0 0.0000000    0
 [4,] 0.2197586 0.0000000 0.3286996 0.0000000 0.2688504 0.0000000    0 0.2575632    0
 [5,] 0.2552459 0.0000000 0.3280535 0.0000000 0.2677174 0.2492834    0 0.2586904    0
 [6,] 0.2001709 0.0000000 0.3248002 0.0000000 0.2629098 0.2468977    0 0.0000000    0
 [7,] 0.3005016 0.0000000 0.3246144 0.0000000 0.2652360 0.0000000    0 0.0000000    0
 [8,] 0.1797843 0.0000000 0.3235852 0.2847070 0.2630664 0.0000000    0 0.0000000    0
 [9,] 0.3358282 0.0000000 0.3279124 0.0000000 0.2694688 0.0000000    0 0.2574010    0
[10,] 0.2323603 0.0000000 0.3268795 0.2859490 0.2674569 0.0000000    0 0.2576323    0
[11,] 0.1406262 0.0000000 0.3182206 0.0000000 0.0000000 0.0000000    0 0.0000000    0
[12,] 0.3368508 0.0000000 0.3281898 0.0000000 0.2670878 0.2512026    0 0.0000000    0
[13,] 0.3761905 0.0000000 0.3312193 0.0000000 0.2717834 0.2538164    0 0.2603919    0
[14,] 0.2564269 0.0000000 0.3318229 0.0000000 0.2700736 0.2509840    0 0.2599665    0
[15,] 0.3045388 0.0000000 0.3285533 0.0000000 0.2677972 0.0000000    0 0.0000000    0
[16,] 0.2132546 0.0000000 0.0000000 0.0000000 0.2630884 0.0000000    0 0.2530495    0
[17,] 0.2021873 0.0000000 0.3294331 0.0000000 0.2648418 0.2484629    0 0.0000000    0
[18,] 0.3395349 0.0000000 0.3309704 0.0000000 0.2726343 0.0000000    0 0.2589785    0
[19,] 0.2168766 0.0000000 0.3263879 0.2857268 0.2641226 0.2478942    0 0.0000000    0
[20,] 0.2686005 0.0000000 0.3293727 0.2869554 0.2688989 0.2502194    0 0.2602655    0
[21,] 0.1969434 0.0000000 0.3214510 0.0000000 0.0000000 0.0000000    0 0.2520729    0
[22,] 0.1642284 0.0000000 0.3229559 0.0000000 0.2629595 0.0000000    0 0.0000000    0
[23,] 0.1674596 0.4886148 0.3223195 0.0000000 0.2628254 0.0000000    0 0.0000000    0
[24,] 0.1794029 0.0000000 0.3199624 0.0000000 0.0000000 0.2441996    0 0.0000000    0
[25,] 0.3795124 0.0000000 0.3348408 0.0000000 0.2749151 0.2556977    0 0.2617681    0
```

```
[26,] 0.1420040 0.0000000 0.3213470 0.0000000 0.0000000 0.0000000    0 0.0000000    0
[27,] 0.1813641 0.0000000 0.3275396 0.2870325 0.2649616 0.0000000    0 0.0000000    0
[28,] 0.2212369 0.0000000 0.3274268 0.0000000 0.2673308 0.0000000    0 0.2583924    0
[29,] 0.1641598 0.0000000 0.3231353 0.0000000 0.2623502 0.0000000    0 0.0000000    0
[30,] 0.1534830 0.0000000 0.0000000 0.0000000 0.2582915 0.0000000    0 0.0000000    0
[31,] 0.2326767 0.0000000 0.3301463 0.2886248 0.2697966 0.0000000    0 0.2588871    0
[32,] 0.4245595 0.0000000 0.3264681 0.0000000 0.2668967 0.0000000    0 0.2572278    0
[33,] 0.2487368 0.0000000 0.0000000 0.0000000 0.2643105 0.2457350    0 0.2554592    0
[34,] 0.3855254 0.0000000 0.3225547 0.0000000 0.2625641 0.0000000    0 0.0000000    0
[35,] 0.1646684 0.0000000 0.3220590 0.0000000 0.2621473 0.0000000    0 0.0000000    0
[36,] 0.3402094 0.0000000 0.3327401 0.0000000 0.2696555 0.2529243    0 0.0000000    0
[37,] 0.1641379 0.0000000 0.3221928 0.0000000 0.2618872 0.0000000    0 0.0000000    0
[38,] 0.2201430 0.0000000 0.3261915 0.0000000 0.2674670 0.0000000    0 0.2568124    0
[39,] 0.2212636 0.4951363 0.3255895 0.0000000 0.2670750 0.0000000    0 0.2564206    0
[40,] 0.3099172 0.0000000 0.3262475 0.2851998 0.2661309 0.0000000    0 0.0000000    0
[41,] 0.2200354 0.0000000 0.3257689 0.0000000 0.2663692 0.0000000    0 0.2571120    0
          [,10]     [,11]     [,12]     [,13]     [,14]     [,15]     [,16]     [,17]
 [1,] 0.0000000 0.0000000 0.1417842 0.1550445 0.1444477 0.1424891 0.1355445 0.0000000
 [2,] 0.0000000 0.0000000 0.1452934 0.1594956 0.1460773 0.1451020 0.1364441 0.0000000
 [3,] 0.0000000 0.0000000 0.1441339 0.1578794 0.1475440 0.1459140 0.1380671 0.1249770
 [4,] 0.0000000 0.0000000 0.1479532 0.1628460 0.1491198 0.1486840 0.1389153 0.1254630
 [5,] 0.0000000 0.0000000 0.1470101 0.1602062 0.1490175 0.1463884 0.1381369 0.0000000
 [6,] 0.0000000 0.0000000 0.1433493 0.1555507 0.1470678 0.1437354 0.1367603 0.0000000
 [7,] 0.3077873 0.0000000 0.1423723 0.1575486 0.1441043 0.1429393 0.1370630 0.0000000
 [8,] 0.0000000 0.0000000 0.1430383 0.1554023 0.1447269 0.1444553 0.1362812 0.0000000
 [9,] 0.3106223 0.0000000 0.1454821 0.1619299 0.1458909 0.1455468 0.1381126 0.0000000
[10,] 0.0000000 0.0000000 0.1465497 0.1601763 0.1464556 0.1470156 0.1372666 0.0000000
[11,] 0.0000000 0.0000000 0.1387908 0.1474840 0.1405516 0.1395901 0.1341036 0.0000000
[12,] 0.3115953 0.0000000 0.1442328 0.1584003 0.1471167 0.1447663 0.1386025 0.0000000
[13,] 0.3145978 0.0000000 0.1474115 0.1630602 0.1494315 0.1474709 0.1402611 0.0000000
[14,] 0.0000000 0.0000000 0.1496434 0.1637523 0.1521707 0.1499879 0.1406947 0.1263836
[15,] 0.3115712 0.0000000 0.1448216 0.1604008 0.1475929 0.1461614 0.1396570 0.1251615
[16,] 0.0000000 0.0000000 0.1401588 0.1580417 0.1425828 0.1414274 0.1340305 0.0000000
[17,] 0.0000000 0.0000000 0.1456557 0.1585407 0.1502286 0.1474003 0.1393794 0.1256997
[18,] 0.3142970 0.0000000 0.1482921 0.1655110 0.1493750 0.1489513 0.1407482 0.1260045
[19,] 0.0000000 0.0000000 0.1447747 0.1561501 0.1472352 0.1461893 0.1375614 0.0000000
[20,] 0.0000000 0.0000000 0.1483546 0.1611542 0.1492774 0.1487653 0.1389826 0.0000000
[21,] 0.0000000 0.0000000 0.1418738 0.1509348 0.1414132 0.1417773 0.1346379 0.0000000
[22,] 0.0000000 0.0000000 0.1422700 0.1562186 0.1441792 0.1430281 0.1356558 0.0000000
[23,] 0.0000000 0.0000000 0.1425550 0.1550563 0.1444522 0.1425521 0.1365949 0.0000000
[24,] 0.0000000 0.0000000 0.1406333 0.1478231 0.1433399 0.1410684 0.1356505 0.0000000
[25,] 0.3173185 0.0000000 0.1502903 0.1669447 0.1532400 0.1508485 0.1431535 0.1269964
[26,] 0.0000000 0.0000000 0.1410677 0.1500007 0.1432261 0.1423534 0.1369024 0.1220017
[27,] 0.0000000 0.0000000 0.1451919 0.1582767 0.1479861 0.1471893 0.1387278 0.1257731
[28,] 0.0000000 0.0000000 0.1455685 0.1609045 0.1468386 0.1464010 0.1367668 0.0000000
```

```
[29,] 0.0000000 0.0000000 0.1418030 0.1557600 0.1452543 0.1433296 0.1359089 0.0000000
[30,] 0.0000000 0.0000000 0.1359380 0.1528228 0.1406258 0.1381124 0.1334980 0.0000000
[31,] 0.0000000 0.0000000 0.1489158 0.1634597 0.1495786 0.1498053 0.1396600 0.1263366
[32,] 0.0000000 0.3995700 0.1453761 0.1602449 0.1470621 0.1454516 0.1371996 0.0000000
[33,] 0.0000000 0.0000000 0.1415438 0.1588266 0.1453491 0.1426507 0.1357158 0.0000000
[34,] 0.0000000 0.3979244 0.1418376 0.1553927 0.1451761 0.1426986 0.1361227 0.0000000
[35,] 0.0000000 0.0000000 0.1418325 0.1559630 0.1450581 0.1429190 0.1356533 0.0000000
[36,] 0.3144942 0.0000000 0.1467525 0.1614658 0.1508810 0.1482514 0.1414146 0.1258577
[37,] 0.0000000 0.0000000 0.1418017 0.1561534 0.1449845 0.1423568 0.1361472 0.0000000
[38,] 0.0000000 0.0000000 0.1457681 0.1605174 0.1458358 0.1456033 0.1366914 0.0000000
[39,] 0.0000000 0.0000000 0.1460681 0.1594705 0.1461579 0.1450610 0.1374726 0.0000000
[40,] 0.3101797 0.0000000 0.1435739 0.1578991 0.1444334 0.1448138 0.1377918 0.0000000
[41,] 0.0000000 0.0000000 0.1453859 0.1613104 0.1467407 0.1451381 0.1372073 0.0000000
          [,18]     [,19]     [,20]     [,21]
 [1,] 0.0000000 0.0000000 0.0000000 0.0000000
 [2,] 0.0000000 0.0000000 0.0000000 0.0000000
 [3,] 0.0000000 0.0000000 0.0000000 0.0000000
 [4,] 0.0000000 0.0000000 0.0000000 0.0000000
 [5,] 0.0000000 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.0000000 0.0000000 0.0000000
 [9,] 0.0000000 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000 0.0000000
[11,] 0.0000000 0.0000000 0.0000000 0.0000000
[12,] 0.0000000 0.0000000 0.0000000 0.0000000
[13,] 0.0000000 0.0000000 0.0000000 0.0000000
[14,] 0.0000000 0.0000000 0.0000000 0.0000000
[15,] 0.0000000 0.0000000 0.0000000 0.0000000
[16,] 0.0000000 0.0000000 0.0000000 0.0000000
[17,] 0.0000000 0.0000000 0.0000000 0.0000000
[18,] 0.0000000 0.0000000 0.0000000 0.0000000
[19,] 0.0000000 0.0000000 0.0000000 0.0000000
[20,] 0.0000000 0.0000000 0.0000000 0.0000000
[21,] 0.0000000 0.0000000 0.0000000 0.0000000
[22,] 0.0000000 0.0000000 0.0000000 0.1206730
[23,] 0.0000000 0.0000000 0.0000000 0.0000000
[24,] 0.0000000 0.0000000 0.0000000 0.0000000
[25,] 0.0000000 0.0000000 0.0000000 0.0000000
[26,] 0.0000000 0.0000000 0.0000000 0.0000000
[27,] 0.0000000 0.0000000 0.0000000 0.0000000
[28,] 0.1256969 0.0000000 0.0000000 0.0000000
[29,] 0.1243528 0.0000000 0.0000000 0.0000000
[30,] 0.0000000 0.0000000 0.0000000 0.0000000
[31,] 0.0000000 0.0000000 0.0000000 0.0000000
```

```
[32,] 0.0000000 0.0000000 0.0000000 0.0000000
[33,] 0.0000000 0.0000000 0.0000000 0.0000000
[34,] 0.0000000 0.0000000 0.0000000 0.0000000
[35,] 0.0000000 0.1178438 0.0000000 0.0000000
[36,] 0.0000000 0.0000000 0.0000000 0.0000000
[37,] 0.0000000 0.0000000 0.1200531 0.0000000
[38,] 0.0000000 0.0000000 0.0000000 0.1224002
[39,] 0.0000000 0.0000000 0.0000000 0.0000000
[40,] 0.0000000 0.0000000 0.0000000 0.0000000
[41,] 0.0000000 0.0000000 0.1222791 0.0000000

#  Note that neither mle estimates nor their SDs change much
#  from model to model, so no evidence of confounding. This
#  is as expected, as all variables independent here,
#  so in fact there was no confounding.

#  Overall, we can see that the BIC does reasonably well, but misses the
#  true correct model.  So, evidence that BIC models sometimes too small.

#  Let's see if AIC does better here:

> output.aic <- glm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 +
       x9 + x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 +
       x19 + x20, data = example2.dat, family = "binomial")
> summary(output.aic)

Call:
glm(formula = y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 +
    x10 + x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 + x19 +
    x20, family = "binomial", data = example2.dat)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.0587  -0.5233   0.2353   0.6474   2.2684

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.270954   0.621399  -2.045 0.040824 *
x1           0.667552   0.523681   1.275 0.202405
x2           1.021604   0.345234   2.959 0.003085 **
x3           0.466288   0.297242   1.569 0.116715
x4           0.939412   0.279999   3.355 0.000793 ***
x5           0.613206   0.260130   2.357 0.018408 *
x6          -0.079718   0.260187  -0.306 0.759308
x7           0.706103   0.273113   2.585 0.009727 **
```

```
x8           0.002336   0.299903   0.008 0.993786
x9           0.677549   0.325826   2.079 0.037573 *
x10          0.475501   0.405723   1.172 0.241203
x11          0.908021   0.154481   5.878 4.16e-09 ***
x12          1.050050   0.171871   6.110 9.99e-10 ***
x13          0.959881   0.156664   6.127 8.96e-10 ***
x14          1.252995   0.155319   8.067 7.19e-16 ***
x15          0.868445   0.147162   5.901 3.61e-09 ***
x16         -0.226770   0.130360  -1.740 0.081935 .
x17         -0.160615   0.130644  -1.229 0.218918
x18         -0.076304   0.125784  -0.607 0.544100
x19         -0.038795   0.128031  -0.303 0.761877
x20          0.064614   0.125445   0.515 0.606499
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 648.68  on 499  degrees of freedom
Residual deviance: 391.28  on 479  degrees of freedom
AIC: 433.28

Number of Fisher Scoring iterations: 6


> step.aic <- step(output.aic)
Start:  AIC= 433.28
 y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 +
    x12 + x13 + x14 + x15 + x16 + x17 + x18 + x19 + x20

       Df Deviance    AIC
- x8    1   391.28 431.28
- x19   1   391.37 431.37
- x6    1   391.38 431.38
- x20   1   391.55 431.55
- x18   1   391.65 431.65
- x10   1   392.64 432.64
- x17   1   392.80 432.80
- x1    1   392.97 432.97
<none>      391.28 433.28
- x3    1   393.79 433.79
- x16   1   394.35 434.35
- x9    1   395.58 435.58
- x5    1   396.96 436.96
- x7    1   398.11 438.11
```

```
- x2    1   400.70 440.70
- x4    1   403.24 443.24
- x15   1   432.09 472.09
- x11   1   433.54 473.54
- x13   1   437.52 477.52
- x12   1   439.97 479.97
- x14   1   481.48 521.48

Step:  AIC= 431.28
 y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x9 + x10 + x11 + x12 +
    x13 + x14 + x15 + x16 + x17 + x18 + x19 + x20

..............etc.........few hundred lines deleted

#  Final model according to AIC

Step:  AIC= 422.02
 y ~ x2 + x3 + x4 + x5 + x7 + x9 + x11 + x12 + x13 + x14 + x15 +
    x16

       Df Deviance    AIC
<none>      396.02 422.02
- x3    1   398.45 422.45
- x16   1   399.87 423.87
- x9    1   400.51 424.51
- x5    1   401.49 425.49
- x7    1   402.44 426.44
- x2    1   406.37 430.37
- x4    1   408.03 432.03
- x15   1   435.51 459.51
- x11   1   437.30 461.30
- x13   1   440.78 464.78
- x12   1   444.45 468.45
- x14   1   485.45 509.45

#  So 12 variables make the final model
#  according to the AIC, including 9
#  that should be there, but also 3 that should not be there.

#  As expected with the AIC, the model is too large.

#  Can also just ask for a summary of the AIC
#  output

> summary(step.aic)
```

```
Call:
glm(formula = y ~ x2 + x3 + x4 + x5 + x7 + x9 + x11 + x12 + x13 +
    x14 + x15 + x16, family = "binomial", data = example2.dat)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.1365  -0.5448   0.2363   0.6306   2.2374

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.8071     0.3892  -2.074 0.038102 *
x2            1.0395     0.3360   3.094 0.001975 **
x3            0.4492     0.2905   1.546 0.122094
x4            0.9278     0.2757   3.365 0.000767 ***
x5            0.5937     0.2563   2.316 0.020552 *
x7            0.6605     0.2631   2.510 0.012067 *
x9            0.6772     0.3190   2.123 0.033768 *
x11           0.8828     0.1516   5.824 5.74e-09 ***
x12           1.0249     0.1678   6.109 1.00e-09 ***
x13           0.9295     0.1535   6.054 1.41e-09 ***
x14           1.2275     0.1524   8.056 7.90e-16 ***
x15           0.8397     0.1439   5.835 5.39e-09 ***
x16          -0.2485     0.1278  -1.944 0.051882 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 648.68  on 499  degrees of freedom
Residual deviance: 396.02  on 487  degrees of freedom
AIC: 422.02

Number of Fisher Scoring iterations: 6

#  So BIC best model a bit too small, AIC much too large, and still
#  also missed x1.

#  Overall, not clear cut which was better here, but BIC
#  better for predictions, and allows easy checking of confounding.
```

Example 3: Low birth weight

We have already analyzed these data twice before, once using a frequentist approach with R's glm function, and again in WinBUGS using a Bayesian approach. Results were similar, but here we will see the BIC results for in model selection (and how the particular BIC program we will use also greatly facilitates investigation of confounding).

Once again, recall the following data were collected:

Variable	Coding
Low Birth Weight (0 = Birth Weight $\geq$ 2500g, 1 = Birth Weight $<$ 2500g)	low
Age of the Mother in Years	age
Weight in Pounds at the Last Menstrual Period	lwt
Race (1 = White, 2 = Black, 3 = Other)	race
Smoking Status During Pregnancy (1 = Yes, 0 = No)	smoke
History of Premature Labor (0 = None 1 = One, etc.)	ptl
History of Hypertension (1 = Yes, 0 = No)	ht
Presence of Uterine Irritability (1 = Yes, 0 = No)	ui
Number of Physician Visits During the First Trimester (0 = None, 1 = One, 2 = Two, etc.)	ftv
Birth Weight in Grams	bwt

What model will the BIC select here?

```
#  Read in the data set
#  (Variable bwt already dropped)

> lbw2.dat <- read.table(file="g:\\lbw2.txt", header=T)

# Declare some variables as factors
# bic.glm recognizes factor variables automatically

 lbw2.dat$smoke <- as.factor(lbw.dat$smoke)
```

```
 lbw2.dat$race <- as.factor(lbw.dat$race)
 lbw2.dat$ptl <- as.factor(lbw.dat$ptl)
 lbw2.dat$ht <- as.factor(lbw.dat$ht)
 lbw2.dat$ui <- as.factor(lbw.dat$ui)

#  Summarize the data

> summary(lbw2.dat)

      low              age              lwt         smoke   ptl     ht      ui
 Min.   :0.0000   Min.   :14.00   Min.   : 80.0   0:115   0:159   0:177   0:161
 1st Qu.:0.0000   1st Qu.:19.00   1st Qu.:110.0   1: 74   1: 24   1: 12   1: 28
 Median :0.0000   Median :23.00   Median :121.0           2:  5
 Mean   :0.3122   Mean   :23.24   Mean   :129.8           3:  1
 3rd Qu.:1.0000   3rd Qu.:26.00   3rd Qu.:140.0
 Max.   :1.0000   Max.   :45.00   Max.   :250.0
      ftv         race
 Min.   :0.0000   1:96
 1st Qu.:0.0000   2:26
 Median :0.0000   3:67
 Mean   :0.7937
 3rd Qu.:1.0000
 Max.   :6.0000

#  Run the BIC

> output <- bic.glm(low ~ age + lwt + smoke + ptl + ht + ui +
          ftv + race, glm.family = binomial, data = lbw2.dat)

> summary(output)

Call:
bic.glm.formula(f = low ~ age + lwt + smoke + ptl + ht + ui +
           ftv + race, data = lbw2.dat, glm.family = binomial)


  49  models were selected


 Best  5  models (cumulative posterior probability =  0.3676 ):

           p!=0      model 1     model 2     model 3     model 4     model 5
Intercept  100       1.45068     1.06795     1.20695     1.08354     0.72186
age        12.6        .           .           .           .           .
lwt        68.7     -0.01865    -0.01692    -0.01881    -0.01805    -0.01631
```

```
smoke      33.2
     .1                .           .           .         0.68391     0.65300
ptl        35.1
   .1                  .           .         1.74321       .           .
   .2                  .           .         0.50107       .           .
   .3                  .           .       -13.98561       .           .
ht         65.4
  .1                 1.85551     1.96157     1.92368     1.82202     1.92213
ui         33.8
  .1                  .         0.93000       .           .         0.89627
ftv         1.5        .           .           .           .           .
race        9.6
    .2                .           .           .           .           .
    .3                .           .           .           .           .

nVar                    2           3           3           3           4
BIC                -753.82285  -753.11035  -752.99778  -752.86548  -751.65571
post prob            0.111       0.078       0.073       0.069       0.037



#  As always, can look at several other items:

#  Posterior probability of each of 41 best models (rest very small by
#   comparison, so are omitted, change value of OR to see them)

> output$postprob
 [1] 0.110708824 0.077529065 0.073285936 0.068594759 0.037462193 0.034934970 0.033014119
 [8] 0.026886406 0.026275368 0.025975383 0.024135850 0.023395418 0.022718911 0.022592906
[15] 0.022367836 0.022217345 0.020015823 0.018093889 0.015330605 0.015111145 0.014675001
[22] 0.014476191 0.014435935 0.014213978 0.013791381 0.013337665 0.013116770 0.012539374
[29] 0.011802513 0.011757133 0.010826335 0.010659212 0.008751301 0.008403062 0.008373099
[36] 0.008183875 0.008096363 0.007982928 0.007935787 0.007679670 0.007018423 0.006978567
[43] 0.006709482 0.006611304 0.006574884 0.006431099 0.006248551 0.006179709 0.005563654

#  Large number of models with closely equivalent posterior probabilities

#  What variables were in each of above 41 models

> output$label
 [1] "lwt,ht"               "lwt,ht,ui"          "lwt,ptl,ht"        "lwt,smoke,ht"
 [5] "lwt,smoke,ht,ui"      "lwt"                "lwt,ptl,ht,ui"     "lwt,smoke,ht,race"
 [9] "lwt,ptl"              "ptl"                "NULL"              "age,ptl"
[13] "age,lwt,ptl,ht"       "ht,ui"              "lwt,smoke"         "ui"
[17] "smoke"                "lwt,smoke,ptl,ht"   "lwt,ui"           "age,lwt,ht"
```

```
[21] "lwt,smoke,ht,ui,race" "smoke,ui"          "smoke,race"       "smoke,ht,ui"
[25] "ptl,ht"               "ptl,ui"            "ht"               "lwt,smoke,race"
[29] "age,ptl,ht"           "ptl,ht,ui"         "smoke,ht"         "age,lwt,ptl"
[33] "lwt,ht,ftv"           "age,ptl,ui"        "age,lwt,ptl,ht,ui""smoke,ui,race"
[37] "smoke,ptl"            "lwt,smoke,ui"      "lwt,ptl,ui"       "lwt,smoke,ptl,ht,ui"
[41] "lwt,smoke,ptl"        "age"               "age,ptl,ht,ui"    "lwt,ht,race"
[45] "lwt,ptl,ht,ftv"       "smoke,ht,race"     "smoke,ht,ui,race" "age,smoke,ptl"
[49] "age,lwt"

#  For each of 8 variables, probability they should be in the model


>  output$names
[1] "age"   "lwt"   "smoke" "ptl"   "ht"    "ui"    "ftv"   "race"

> output$probne0
[1] 12.6 68.7 33.2 35.1 65.4 33.8  1.5  9.6

#  Note that best variable only has only 69% probability (lwt),
#  not very good evidence for any of them.

#  Bayesian model averaged means for each variable.  All very near 0 except intercept.

>  output$postmean
 [1]  0.471633805 -0.007765361 -0.011557775  0.255357346  0.617359889  0.168588430 -4.910
 [8]  1.166892815  0.310537495 -0.001338774  0.115278891  0.092740180

#  For each of top 49 models (in this case), model by model estimates
#  This is where you can check for confounding, very conveniently.

#  For space reasons, I have cut this off at top 20 models

>
>  output$mle
          [,1]        [,2]        [,3]      [,4]     [,5]      [,6]      [,7]     [,8]
 [1,] 1.45067939  0.00000000 -0.01865264 0.0000000 0.000000 0.0000000   0.00000 1.85551
 [2,] 1.06795067  0.00000000 -0.01691950 0.0000000 0.000000 0.0000000   0.00000 1.96157
 [3,] 1.20694583  0.00000000 -0.01881475 0.0000000 1.743215 0.5010743 -13.98561 1.92368
 [4,] 1.08353776  0.00000000 -0.01804644 0.6839097 0.000000 0.0000000   0.00000 1.82202
 [5,] 0.72185532  0.00000000 -0.01630585 0.6529976 0.000000 0.0000000   0.00000 1.92212
 [6,] 0.99831432  0.00000000 -0.01405826 0.0000000 0.000000 0.0000000   0.00000 0.00000
 [7,] 0.89024810  0.00000000 -0.01737956 0.0000000 1.657921 0.2964293 -14.68756 2.01695
 [8,] 0.35204934  0.00000000 -0.01790657 1.0715663 0.000000 0.0000000   0.00000 1.74916
 [9,] 0.74611373  0.00000000 -0.01408871 0.0000000 1.717056 0.4341668 -13.97375 0.00000
[10,]-1.05711256  0.00000000  0.00000000 0.0000000 1.750260 0.6516474 -13.50896 0.00000
```

```
[11,]-0.78999701  0.00000000  0.00000000 0.0000000 0.000000 0.0000000   0.00000 0.00000
[12,] 0.62617705 -0.07480350  0.00000000 0.0000000 1.949469 0.6397230 -13.32216 0.00000
[13,] 2.29198573 -0.05944305 -0.01681870 0.0000000 1.899783 0.5221180 -13.77420 1.84896
[14,]-1.07194404  0.00000000  0.00000000 0.0000000 0.000000 0.0000000   0.00000 1.40841
[15,] 0.62199682  0.00000000 -0.01332433 0.6766732 0.000000 0.0000000   0.00000 0.00000
[16,]-0.94692770  0.00000000  0.00000000 0.0000000 0.000000 0.0000000   0.00000 0.00000
[17,]-1.08705147  0.00000000  0.00000000 0.7040592 0.000000 0.0000000   0.00000 0.00000
[18,] 0.93018305  0.00000000 -0.01826532 0.5465250 1.627536 0.3765970 -14.30757 1.85648
[19,] 0.64770772  0.00000000 -0.01234677 0.0000000 0.000000 0.0000000   0.00000 0.00000
[20,] 2.13505153 -0.03637716 -0.01745358 0.0000000 0.000000 0.0000000   0.00000 1.81556


          [,9]       [,10]    [,11]     [,12]
 [1,] 0.0000000  0.00000000 0.000000 0.0000000
 [2,] 0.9299976  0.00000000 0.000000 0.0000000
 [3,] 0.0000000  0.00000000 0.000000 0.0000000
 [4,] 0.0000000  0.00000000 0.000000 0.0000000
 [5,] 0.8962654  0.00000000 0.000000 0.0000000
 [6,] 0.0000000  0.00000000 0.000000 0.0000000
 [7,] 0.8823068  0.00000000 0.000000 0.0000000
 [8,] 0.0000000  0.00000000 1.287662 0.9436448
 [9,] 0.0000000  0.00000000 0.000000 0.0000000
[10,] 0.0000000  0.00000000 0.000000 0.0000000
[11,] 0.0000000  0.00000000 0.000000 0.0000000
[12,] 0.0000000  0.00000000 0.000000 0.0000000
[13,] 0.0000000  0.00000000 0.000000 0.0000000
[14,] 1.0719440  0.00000000 0.000000 0.0000000
[15,] 0.0000000  0.00000000 0.000000 0.0000000
[16,] 0.9469277  0.00000000 0.000000 0.0000000
[17,] 0.0000000  0.00000000 0.000000 0.0000000
[18,] 0.0000000  0.00000000 0.000000 0.0000000
[19,] 0.8132763  0.00000000 0.000000 0.0000000
[20,] 0.0000000  0.00000000 0.000000 0.0000000

#  Can look over columns, see if any coefficients
#  Change values as other variables enter and exit the model.

#  As all effects rather weak, not so much evidence for confounding.

#  Top 20 model SDs

>  output$se
          [,1]       [,2]        [,3]      [,4]      [,5]      [,6]     [,7]      [,8]
 [1,] 0.8209655 0.00000000 0.006594143 0.0000000 0.0000000 0.0000000   0.0000 0.7009758 0
 [2,] 0.8329329 0.00000000 0.006564433 0.0000000 0.0000000 0.0000000   0.0000 0.6957015 0
```

```
 [3,] 0.8734975 0.00000000 0.006984423 0.0000000 0.4868026 0.9529977 882.7434 0.7365123 0
 [4,] 0.8342194 0.00000000 0.006564702 0.3309536 0.0000000 0.0000000   0.0000 0.6860388 0
 [5,] 0.8490740 0.00000000 0.006546259 0.3356765 0.0000000 0.0000000   0.0000 0.6826654 0
 [6,] 0.7852909 0.00000000 0.006169588 0.0000000 0.0000000 0.0000000   0.0000 0.0000000 0
 [7,] 0.8803800 0.00000000 0.006934261 0.0000000 0.4906691 0.9733347 882.7435 0.7308590 0
 [8,] 0.9244383 0.00000000 0.006798875 0.3875173 0.0000000 0.0000000   0.0000 0.6908200 0
 [9,] 0.8420540 0.00000000 0.006582726 0.0000000 0.4751827 0.9445108 882.7434 0.0000000 0
[10,] 0.1812866 0.00000000 0.000000000 0.0000000 0.4694303 0.9306977 882.7434 0.0000000 0
[11,] 0.1569759 0.00000000 0.000000000 0.0000000 0.0000000 0.0000000   0.0000 0.0000000 0
[12,] 0.7852534 0.03468830 0.000000000 0.0000000 0.4921417 0.9356429 882.7434 0.0000000 0
[13,] 1.0976945 0.03590566 0.007043310 0.0000000 0.5048840 0.9536281 882.7434 0.7366460 0
[14,] 0.1879488 0.00000000 0.000000000 0.0000000 0.0000000 0.0000000   0.0000 0.6149649 0
[15,] 0.7959165 0.00000000 0.006089569 0.3246986 0.0000000 0.0000000   0.0000 0.0000000 0
[16,] 0.1756214 0.00000000 0.000000000 0.0000000 0.0000000 0.0000000   0.0000 0.0000000 0
[17,] 0.2147338 0.00000000 0.000000000 0.3196423 0.0000000 0.0000000   0.0000 0.0000000 0
[18,] 0.8821160 0.00000000 0.006917469 0.3483753 0.4938117 0.9625647 882.7434 0.7168562 0
[19,] 0.8015931 0.00000000 0.006173316 0.0000000 0.0000000 0.0000000   0.0000 0.0000000 0
[20,] 1.0312359 0.03288763 0.006642326 0.0000000 0.0000000 0.0000000   0.0000 0.6999546 0


         [,10]     [,11]     [,12]
 [1,] 0.0000000 0.0000000 0.0000000
 [2,] 0.0000000 0.0000000 0.0000000
 [3,] 0.0000000 0.0000000 0.0000000
 [4,] 0.0000000 0.0000000 0.0000000
 [5,] 0.0000000 0.0000000 0.0000000
 [6,] 0.0000000 0.0000000 0.0000000
 [7,] 0.0000000 0.0000000 0.0000000
 [8,] 0.0000000 0.5216483 0.4233816
 [9,] 0.0000000 0.0000000 0.0000000
[10,] 0.0000000 0.0000000 0.0000000
[11,] 0.0000000 0.0000000 0.0000000
[12,] 0.0000000 0.0000000 0.0000000
[13,] 0.0000000 0.0000000 0.0000000
[14,] 0.0000000 0.0000000 0.0000000
[15,] 0.0000000 0.0000000 0.0000000
[16,] 0.0000000 0.0000000 0.0000000
[17,] 0.0000000 0.0000000 0.0000000
[18,] 0.0000000 0.0000000 0.0000000
[19,] 0.0000000 0.0000000 0.0000000
[20,] 0.0000000 0.0000000 0.0000000

#  Overall, not much to predict outcomes from this data set.
```

Final Notes

Do not forget, amidst all these statistical ideas, that substantive knowledge and knowledge about a study design can and should play a role in thinking about a model, and how well it suits a given purpose.

If you have good *a priori* reasons to believe a variable should be in a model then simply include it, unless the evidence against it is very strong. This is easy to do in the BIC functions (set prior = 1).

If the main point of a model is prediction, you might not care too much about which independent variables are included, as long as the model "fits well". But if the purpose of your model is to see which variables are important, then much attention needs to be paid to this issue.