MTH 4230 R Notes 8

1 Model Selection in R

- We'll look at four criteria for selecting a model:

  1. The ***adjusted coefficient of determination*** (R^2_{adj})
  2. ***Akaike's information criterion*** (***AIC***)
  3. ***Bayesian information criterion*** (***BIC***) (also called Schwarz's Bayesian criterion, SBC)
  4. ***PRESS***

- Suppose we have a *data frame* called `my.data` as shown below.

```
my.data

##    response x1  x2 x3
## 1        21 10 2.4 94
## 2        22 11 1.9 75
## 3        24 13 2.6 66
## 4        20 12 1.8 77
## 5        26 13 1.3 75
## 6        26 16 0.9 81
## 7        25 15 1.2 72
## 8        27 20 1.0 55
## 9        31 18 1.3 44
## 10       34 19 1.0 69
## 11       29 19 0.8 73
## 12       33 22 0.7 90
```

1.1 Computing R^2_{adj}

- We've seen that after fitting a model using `lm()`, `summary()` will print the ***adjusted coefficient of determination***, R^2_{adj}.

1.2 Computing AIC and BIC for a Single Model

- If we want R to compute ***AIC*** or ***BIC*** for a **single** multiple regression model, we can use:

```
extractAIC()      # Computes the AIC or BIC for a given fitted model
```

The `extractAIC()` function takes as its main argument an *lm* object, and an optional argument `k` that can be used to indicate that ***BIC*** should be computed instead of ***AIC***.

1.2.1 Obtaining AIC Using `extractAIC()`

- As an example of obtaining ***AIC***, first we fit the model (using `my.data` from above):

```
my.reg <- lm(response ~ x1 + x2 + x3, data = my.data)
```

Then we call `extractAIC()` with the fitted *lm* object:

```
extractAIC(my.reg)

## [1]  4.00000 25.47726
```

The function `extractAIC()` returns a vector with two elements, the first of which is the **number of parameters** in the fitted model and the second of which is the ***AIC*** value. Thus above we see that the model has $\boldsymbol{p = 4}$ parameters and $\boldsymbol{AIC = 25.4773}$.

1.2.2 Obtaining BIC Using `extractAIC()`

- If we want R to compute ***BIC*** for a **single** model, we specify `k = log(n)` in the call to `extractAIC()`, where `n` is the number of observations in the data set:

```
n <- nrow(my.data)
extractAIC(my.reg, k = log(n))

## [1]  4.00000 27.41689
```

From the output, we see that the model has $\boldsymbol{p = 4}$ parameters and $\boldsymbol{AIC = 27.4169}$.

- The value `k = log(n)` refers to the constant $\log(n)$ appearing in front of p as part of the "penalty" term in the formula for BIC:

$$\text{BIC} = n\log(\text{SSE}) - n\log(n) + \log(n)p$$

The default value for `k` in `extractAIC()` is `k = 2`, corresponding to the constant 2 appearing in front of p as part of the "penalty" term in the formula for AIC:

$$\text{AIC} = n\log(\text{SSE}) - n\log(n) + 2p$$

1.3 Computing AIC and BIC Before and After Dropping or Adding a Predictor

- We may wish to compare the AIC or BIC values for a model before and after dropping or adding a single predictor. To do so, we use the functions:

```
drop1()        # Compute AIC or BIC before and after dropping a single
               # predictor from a model.
add1()         # Compute AIC or BIC before and after adding a single
               # predictor from a model.
```

Both functions take *lm* objects as their main argument.

1.3.1 Obtaining AIC and BIC using `drop1()`

- Suppose we want to compute ***AIC*** before and after **dropping** a single predictor from a model. We first fit a model containing **all** of the predictors (in this case using `my.data` from above):

```
my.reg <- lm(response ~ x1 + x2 + x3, data = my.data)
```

Next we pass our *lm* object `drop1()`:

```
drop1(my.reg)

## Single term deletions
##
## Model:
## response ~ x1 + x2 + x3
##        Df Sum of Sq     RSS    AIC
## <none>               51.488 25.477
## x1      1    49.626 101.113 31.576
## x2      1     0.003  51.490 23.478
## x3      1     0.096  51.584 23.500
```

Here's what's shown in the output:

- At the top, the **starting model** (that contains `x1`, `x2`, and `x3`) is shown.
- Below that, the first column indicates which predictor is **dropped** from the starting model. These are **not** dropped sequentially, but rather each line corresponds to the model with just that one predictor omitted from starting model.
- The second column gives the ***degrees of freedom*** for ***extra sum of squares*** associated with the predictor in the first column.
- The third column gives the ***extra sum of squares*** associated with the predictor in the first column.
- The fourth column gives the ***error sum of squares*** (or ***residual sum of squares***) for the model in which the predictor in the first column is omitted.
- Finally, the fifth column gives the ***AIC*** for the model in which the predictor in the first column is omitted.

From the output, we see that:

- The ***AIC*** for the **full model** containing all three predictors is **25.477**.
- If `x1` is **dropped** from the full model, the ***AIC*** increases to **31.576**.
- If `x2` is **dropped** from the full model, the ***AIC*** decreases to **23.478**.
- If `x3` is **dropped** from the full model, the ***AIC*** decreases to **23.5**.

Based on the ***AIC***, if we had to choose between the **full model** and one of the three models with only **two predictors**, we'd choose the model with the **two predictors** `x1` and `x2`.

To obtain the value of ***BIC*** before and after **dropping** a predictor from the model, we specify `k = log(n)` in `drop1()` (as described in Subsection 1.2.2 above):

```
n <- nrow(my.data)
drop1(my.reg, k=log(n))

## Single term deletions
##
## Model:
## response ~ x1 + x2 + x3
##        Df Sum of Sq     RSS    AIC
## <none>               51.488 27.417
## x1      1    49.626 101.113 33.031
## x2      1     0.003  51.490 24.933
## x3      1     0.096  51.584 24.954
```

Notice that ***BIC*** values are still labeled "`AIC`" in the output. Based on ***BIC***, if we had to choose between the **full model** and one of the three models with only **two predictors**, we'd choose the model with the **two predictors** `x1` and `x3`.

1.3.2 Obtaining AIC and BIC using `add1()`

- If we want to compute ***AIC*** before and after **adding** a predictor to a model, we start by fitting the model to which we're considering adding a predictor.

  Below we fit a model with just an intercept:

```
my.reg <- lm(response ~ 1, data = my.data)
```

  Now we pass the *lm* object to `add1()` along with a model *formula* containing the predictors we're considering adding, via the argument `scope`, and the *data frame* containing those predictors:

```
add1(my.reg, scope = response ~ x1 + x2 + x3, data = my.data)

## Single term additions
##
## Model:
## response ~ 1
##        Df Sum of Sq     RSS    AIC
## <none>              227.000 37.281
## x1      1   175.399  51.601 21.504
## x2      1   116.420 110.580 30.650
## x3      1    18.672 208.328 38.250
```

The output is as described in Subsection 1.3.1, except that now each row corresponds to **adding** that predictor to the model.

Thus based on the output above, among the models that have just one predictor or none at all, we'd choose the model with the one predictor `x1`.

- If instead we want to use ***BIC*** to compare models before and after adding a predictor, we simply include the argument `k = log(n)` in the call to `add1()` (as described in Subsection 1.2.2):

```
n <- nrow(my.data)
add1(my.reg, scope = response ~ x1 + x2 + x3, k=log(n), data=my.data)

## Single term additions
##
## Model:
## response ~ 1
##        Df Sum of Sq     RSS    AIC
## <none>              227.000 37.765
## x1      1   175.399  51.601 22.473
## x2      1   116.420 110.580 31.620
## x3      1    18.672 208.328 39.220
```

Note that ***BIC*** is still labeled `"AIC"` in the output. Based on the ***BIC*** values in the output, of the models that have just one predictor or none at all, we'd again choose the model with the one predictor `x1`.

1.4 Computing PRESS

- There are a few ways to compute ***PRESS*** in R. Here are two of them:

  1. Use the well known fact that

$$Y_i - \hat{Y}_{(i)i} = \frac{Y_i - \hat{Y}_i}{1 - h_{ii}}.$$

where $\hat{Y}_i$ is the ith fitted value for the model that was fitted to the full data set, $\hat{Y}_{(i)i}$ is the ith fitted value for the model that was fitted to the data set with the ith observation deleted, and h_{ii} is the ith diagonal element of the ***hat matrix*** (obtained using the full data set). Then ***PRESS*** can be computed as

$$\text{PRESS} \;=\; \sum_{i=1}^{n}(Y_i - \hat{Y}_{(i)i})^2 \;=\; \sum_{i=1}^{n}\left(\frac{Y_i - \hat{Y}_i}{1 - h_{ii}}\right)^2 .$$

Notice that this only requires fitting a regression model **once** (to the full data set), **not** n separate times (once for each delete-i data set). In R, the diagonal elements h_{ii} of the ***hat matrix*** are easily obtained by passing an *lm* object such as `my.reg` to the function `hatvalues()`, and the fitted values $\hat{Y}_i$ are easily obtained from an *lm* object as `my.reg$fitted.values`.

2. Use the `press()` function in the `DAAG` library. After installing `DAAG` and loading it into the current R session (by typing `library(DAAG)`), the function `press()` will accept an *lm* object as an argument, for example by typing:

```r
# This only needs to be done once:
install.packages("DAAG")
```

```r
# This needs to be done for each R session that press() is to be used:
library(DAAG)
```

```r
# Now press() should be available:
press(my.reg)
```

Then `press()` returns the value of ***PRESS***.