Lecture 9

Peter Shaffery

2/11/2021

Mea Culpa

Quick apology for today- I've been indicating the wrong R^2 in the regression table for a few weeks. While this hasn't really mattered so far, today it will make a difference so I wanted to clarify:

```
x = rnorm(100)
y = rnorm(100,2*x)

summary(lm(y~x))
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.18386 -0.60095 -0.07828  0.53019  2.61524
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.19787    0.09599   2.061   0.0419 *
## x            1.85453    0.09690  19.139   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9585 on 98 degrees of freedom
## Multiple R-squared:  0.7889, Adjusted R-squared:  0.7868
## F-statistic: 366.3 on 1 and 98 DF,  p-value: < 2.2e-16
```

In this table the **Multiple R-squared** is the R^2 we have been working with so far (the squared correlation coefficient) and the **Adjusted R-squared** is something we will define today. Apologies for any confusion.

Variable Selection

Variable selection is a *very* big topic, and a lot of the currently accepted "best" methods are actually beyond the scope of this course. Techniques like LASSO for example are very powerful variable selection tools, but fall more under "statistical learning" than statistical modeling *per se*.

Variable selection falls under the more general heading of "model" selection, and for linear regression the two problems are basically equivalent, since a regression model is basically defined by the choice of variables. In this lecture I will use the terms interchangably, however in some future cases they may be technically distinct terms.

Variable (or model) selection has two major types. There is **automated** variable selection, where we use some procedure to select variables to include automatically (ex. stepwise regression, LASSO), and then there is **pre-specified** variable selection, where we some pre-existing list of candidate models, and we would like to choose the "best" one from among them. In this lecture we will ignore automated variable selection.

Example: Motor Trends Car Data

In 1974 Motor Trends magazine collected data on 32 cars. We would like to look at how different car features relate to fuel efficiency.

```
library(tidyverse)
library(magrittr)
library(broom)

mtcars %>% head
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Variable	Definition
MPG	Miles per gallon
Cyl	Number of cylinders
Disp	Displacement (cubic in)
HP	Horsepower
Drat	Rear axle ratio
WT	Weight (tons)
QSEC	Quarter mile time
VS	Engine shape (v-shaped or straight)
AM	Transmission (auto or manual)
Gear	Number of gears
Carb	Number of carburetors

New Package: `broom`

A useful package for sophisticated model analysis in R is `broom`. `broom` is a member of the tidyverse, and covert's R's native `lm` model structure to a tidy dataset:

```
mod = lm (mpg ~ cyl + gear + disp, data=mtcars)
summary(mod)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + gear + disp, data = mtcars)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -4.4156 -2.1554 -0.6433  1.2438  7.0854
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.96602    4.76257   7.132 9.25e-08 ***
## cyl         -1.59024    0.72425  -2.196   0.0366 *
## gear         0.15828    0.91015   0.174   0.8632
## disp        -0.02002    0.01092  -1.833   0.0774 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.108 on 28 degrees of freedom
## Multiple R-squared:  0.7598, Adjusted R-squared:  0.7341
## F-statistic: 29.53 on 3 and 28 DF,  p-value: 8.112e-09
```

```
broom::tidy(mod)
```

```
## # A tibble: 4 x 5
##   term        estimate std.error statistic      p.value
##   <chr>          <dbl>     <dbl>     <dbl>        <dbl>
## 1 (Intercept)  34.0       4.76       7.13  0.0000000925
## 2 cyl          -1.59      0.724     -2.20  0.0366
## 3 gear          0.158     0.910      0.174 0.863
## 4 disp         -0.0200    0.0109    -1.83  0.0774
```

```
broom::glance(mod)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.760         0.734  3.11      29.5 8.11e-9     3  -79.6  169.  176.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

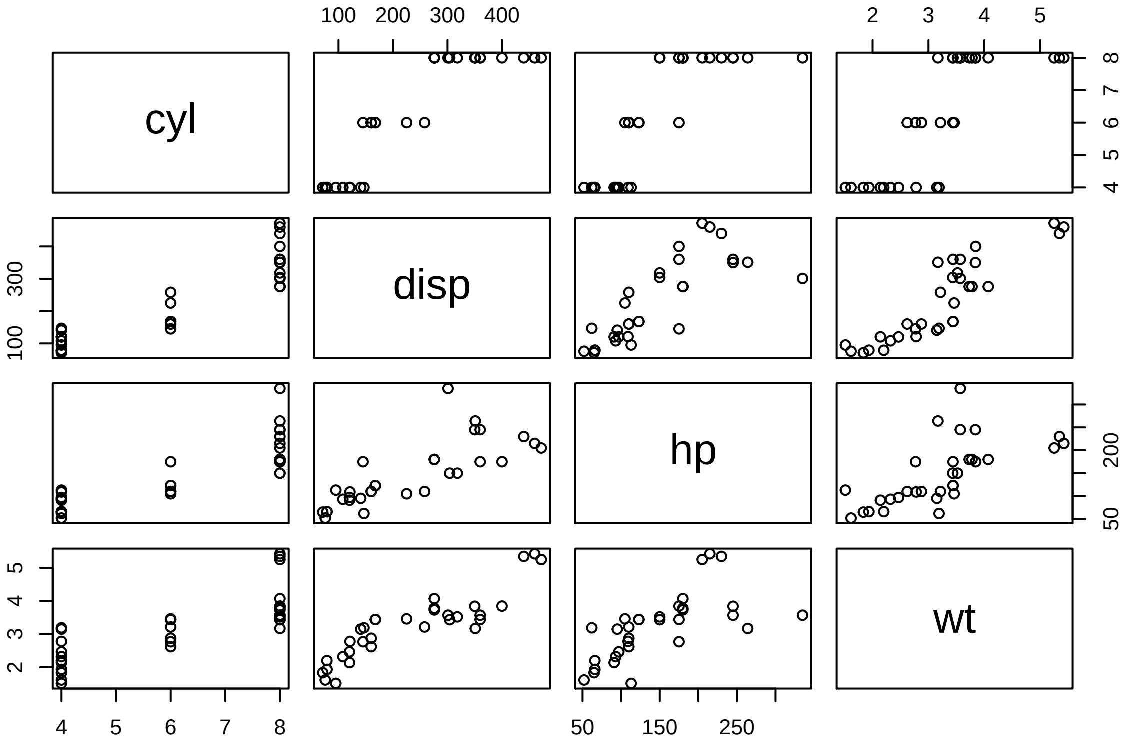
Checking for Collinearity

Since there are a large number of potential variables in this model, we should first check for collinearity:

```
car::vif(lm(mpg ~ ., data=mtcars))
```

```
##       cyl      disp        hp      drat        wt      qsec        vs        am
## 15.373833 21.620241  9.832037  3.374620 15.164887  7.527958  4.965873  4.648487
##      gear      carb
##  5.357452  7.908747
```

```
pairs(mtcars %>% select(c(cyl,disp,hp,wt)))
```



Using a VIF cutoff of 10, it seems that CYL, DISP, HP, and WT are good candidates for possible exclusion. But should we exclude all four? Are any of them useful in our model? How can we pick?

To some extent, it depends on what our aims are, ie. *prediction* vs *inference*. Do we want a model to predict out of sample data well? Does it matter if our p-values are interpretable?

Using p-values

One common choice for adding a variable to a model is to check if the variable's coefficient is statistically significant or not.

At first glance, this might seem like a reasonable choice. Recall that (for a single coefficient β_j) the p-value compares between two hypotheses:

- H_0: $\beta_j = 0$ *given the other model coefficients are nonzero*
- H_A: $\beta_j \neq 0$ *given the other model coefficients are nonzero*

Indeed given these definitions, it seems that we are exactly testing whether or not to add a single variable to the model.

Unfortunately, there are a number of problems that can result here. For one, when choosing between more than two models, our answer may not be well-defined

Say that, through some domain knowledge we've already chosen to include GEAR, CARB, and AM in our model. We now want to use the p-value to determine which of our four variables to add, with the standard cutoff of $\alpha = .05$:

```
# none of the four
mod.base = lm(mpg~gear+am+carb,data=mtcars)
mod.base %>% summary
```

```
##
## Call:
## lm(formula = mpg ~ gear + am + carb, data = mtcars)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.020 -2.754  0.707  1.440  5.945
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.7420     4.0689   3.132  0.00405 **
## gear          3.5580     1.3322   2.671  0.01246 *
## am            3.5451     1.8975   1.868  0.07221 .
## carb         -2.5642     0.3705  -6.921  1.6e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.082 on 28 degrees of freedom
## Multiple R-squared:  0.7638, Adjusted R-squared:  0.7385
## F-statistic: 30.19 on 3 and 28 DF,  p-value: 6.423e-09
```

```
# wt
mod.wt = lm(mpg~gear+am+carb+wt,data=mtcars)
p.wt = tidy(mod.wt) %>% filter(term=='wt') %>% select(p.value)

# hp
mod.hp = lm(mpg~gear+am+carb+hp,data=mtcars)
p.hp = tidy(mod.hp) %>% filter(term=='hp') %>% select(p.value)

# disp
mod.disp = lm(mpg~gear+am+carb+disp,data=mtcars)
p.disp = tidy(mod.disp) %>% filter(term=='disp') %>% select(p.value) ## SIGNIFICANT

# cyl
mod.cyl = lm(mpg~gear+am+carb+cyl,data=mtcars)
p.cyl = tidy(mod.cyl) %>% filter(term=='cyl') %>% select(p.value)

print("Which variables are significant?")
```

```
## [1] "Which variables are significant?"
```

```
alpha = .05
c(p.wt,p.hp,p.disp,p.cyl) < alpha
```

```
## p.value p.value p.value p.value
##    TRUE    TRUE    TRUE    TRUE
```

They're all significant! How do we interpret this?

Well, we've already seen that these variables have some collinearity between each other. That is, they introduce similar information into the model. We therefore *shouldn't* be surprised that they're all significant. This result is telling us that at least one of these variables does in fact belong in the model. But which?

You might be tempted at this point to just pick the variable with the *lowest* p-value.

```
c(p.wt,p.hp,p.disp,p.cyl)
```

```
## $p.value
## [1] 0.0069139
##
## $p.value
## [1] 0.006836583
```

```
##
## $p.value
## [1] 0.001909267
##
## $p.value
## [1] 0.009661501
```

THIS IS A BAD IDEA.

There is one question you can answer using p-values: "is this coefficient non-zero?". As we have just seen, this is a very different type of question from "is this variable relevant to the response?"

Indeed, in early lectures we saw that variables with very small β_j could still have small p-values. Let's look a little more closely at why that could be.

First recall the definition of the t-statistic, which we use to calculate a p-value (when $|t_j|$ is large then p is small):

$$t_j = \frac{\hat{\beta}_j}{\mathrm{se}[\hat{\beta}_j]}$$

Now, what features of β_j or the data could produce large $|t_j|$ (and thus small p)?

1. **Large coefficient values β_j.** In variable selection this is (roughly speaking) the case that we want.
2. Smaller $\mathrm{se}[\hat{\beta}_j]$, ie. more confidence in our estimate. This can come about in a few ways (more variance in x_j, lower VIF_j)

So by selecting for small p-values, we may just be including only the variables whose coefficients we are most certain about. While this *may* be justifiable in some use-cases, if your overall goal is produce a model that *explains* the dependent variable y you're going to be misled.

Prediction Error

So far we have talked about the *inference* side of variable selection (and it's pitfalls). However, as I've mentioned the core goal of model selection is typically **prediction**, improving the performance of our model for unobserved data points.

For illustration purposes, let's now limit our discussion to the variables in two of our models:

```
tidy(mod.wt)$term
```

```
## [1] "(Intercept)" "gear"        "am"          "carb"        "wt"
```

```
tidy(mod.base)$estimate
```

```
## [1] 12.742026  3.558004  3.545063 -2.564168
```

```
tidy(mod.wt)$estimate
```

```
## [1] 26.885527  1.793315  1.345680 -1.526221 -3.003166
```

Notice that the coefficients in the smaller model are larger than their counterparts in the bigger models.

To understand this, let's imagine that we were starting from the larger model, and deciding to move to the smaller model (by eliminating the WT):

$$\begin{aligned} \text{WT (Full) Model}: \mathrm{MPG}_i &= \hat{\beta}_0 + \hat{\beta}_1\mathrm{GEAR}_i + \hat{\beta}_2\mathrm{AM}_i + \hat{\beta}_3\mathrm{CARB}_i + \hat{\beta}_4\mathrm{WT}_i \\ \text{Base (Reduced) Model}: \mathrm{MPG}_i &= \hat{\beta}_0^* + \hat{\beta}_1^*\mathrm{GEAR}_i + \hat{\beta}_2^*\mathrm{AM}_i + \hat{\beta}_3^*\mathrm{CARB}_i + 0\mathrm{WT}_i \end{aligned}$$

If β_4 is truly nonzero (and in this case the p-value suggests it is not) the $\hat{\beta}_j^*$ are then **biased**.

Since the way to get unbiased estimates of the β_j was using the LSE, it shouldn't be surprising that choosing something other than those LSEs (by *forcing* $\beta_4 = 0$) introduces **upward bias** into our estimates $\hat{\beta}_j^*$. We have given up the property of biasedness to obtain a smaller model. The estimates obtained this way are not the BLUE.

This will have consequences for our predictions as well. If we let MPG_i denote the true expected value of some currently unobserved data point, and let $\hat{\text{MPG}}^*_{\text{new}}$ denote the prediction obtained from the reduced model, then $E[\hat{\text{MPG}}^*_{\text{new}}] \neq E[MPG_{\text{new}}]$, ie. our predictions will also be biased (although the direction of the bias no longer consistent).

Interestingly, this bias maybe be "worth it". Even though on average our predictions will be incorrect, it turns out that $\text{Var}[\hat{\text{MPG}}^*_{\text{new}}]$ will be lower than $\text{Var}[\hat{\text{MPG}}_{\text{new}}]$ (the prediction made with the full variable set).

Let's make this more explicit. For a prediction $\hat{y}_i$ of the dependent variable y_i Define Mean Square Error as:

$$MSE(\hat{y}_i) = E[(\hat{y}_i - y_i)^2]$$

As it turns out, it can be shown that:

$$MSE(\hat{y}_i) = \text{Bias}(\hat{y}_i)^2 + \text{Var}[\hat{y}_i]$$

Where $\text{Bias}(\hat{y}_i) = E[\hat{y}_i] - E[y_i]$

Going back to our example, if the reduction in variance $\text{Var}[\hat{\text{MPG}}^*_{\text{new}}] < \text{Var}[\hat{\text{MPG}}_{\text{new}}]$ is bigger than the gain in bias (from 0 to $E[\hat{\text{MPG}}^*_{\text{new}}] - E[MPG_{\text{new}}]$), then our overall *prediction error* $MSE(\hat{\text{MPG}}^*_{\text{new}})$ may be lower than $MSE(\hat{\text{MPG}}_{\text{new}})$. If this is the case, then removing WT_i is "worth it".

Using Mallow's C_p

Mallow's C_p attempts to estimate the the *overall* prediction error of the existing dataset: $J = \frac{1}{\sigma^2} \sum_i MSE(\hat{y}_i)$. Mallow's C_p is calculated:

$$C_p = \frac{SSE_p}{\hat{\sigma}_{\text{Full Model}}} + 2p - n$$

Where p here is the number of variables in the reduced model, SSE_p is the sum of square error for the reduced model, and n is the number of datapoints.

As you might expect, we would like to minimize the overall error J by minimizing C_p. Notice that we can do so by either:

a. Shrinking $SSE_{\text{Reduced Model}}$
b. Shrinking p

Due to the bias-variance tradeoff, C_p will initially decrease as we add variables to the model and reduce the bias. Eventually, however, the reduction in bias will be overtaken by the growth in variance, and so C_p will grow:

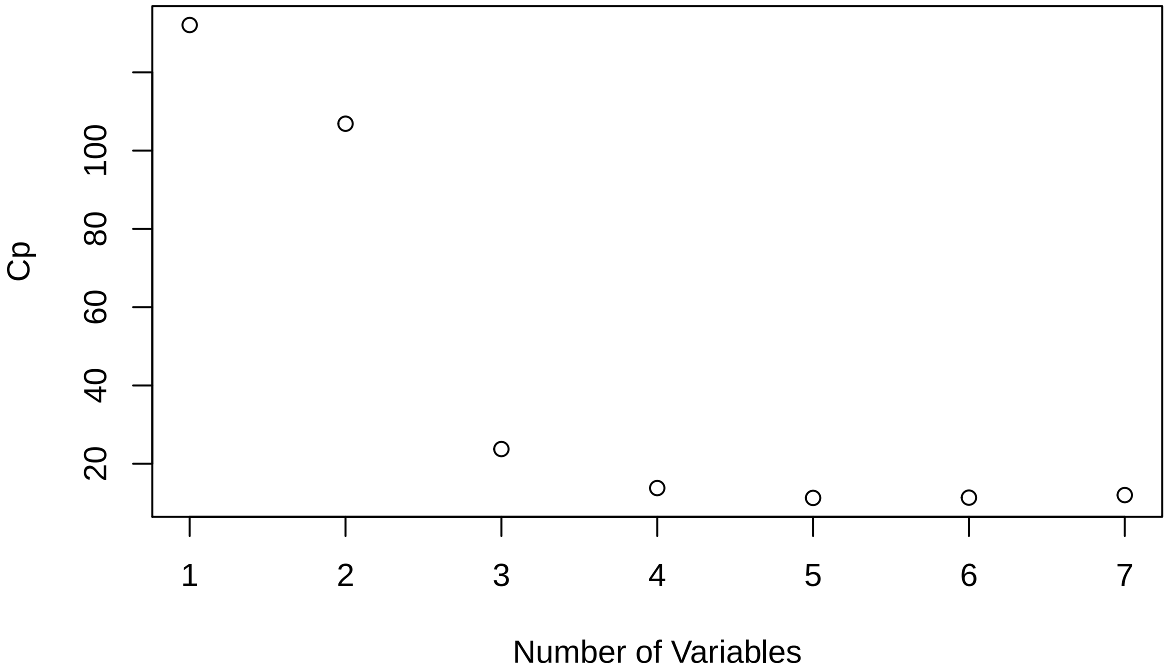
```
varset = c('gear','am','carb','wt','disp','hp','cyl')
dat = mtcars %>% select(c('mpg',varset))
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(varset)` instead of `varset` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
n = nrow(dat)
mod = lm(mpg~., dat)
hat.sigma2 = sum(resid(mod)^2)/(n-2)
```

```
cp = function(varset){
        dat = mtcars %>% select(c('mpg',varset))
        mod = lm(mpg~., dat)
        p = length(varset)
        SSE = sum(resid(mod)^2)
        return((SSE/hat.sigma2) + 2*p - n)
}


cp.vals = c()
for (p in 1:length(varset)){
        new.cp = cp(varset[1:p])
        cp.vals = c(cp.vals, new.cp)
}
plot(1:length(varset),cp.vals, xlab='Number of Variables',ylab='Cp')
```



Another way that we can use Mallow's C_p is to select between our list of prespecified models:

```
vars = c('gear','am','carb','wt')
cp(vars)
```

```
## [1] 13.7879
```

```
vars = c('gear','am','carb','hp')
cp(vars)
```

```
## [1] 13.75912
```

```
vars = c('gear','am','carb','disp')
cp(vars)
```

```
## [1] 10.59837
```

```
vars = c('gear','am','carb','cyl')
cp(vars)
```

```
## [1] 14.6508
```

Using Adjusted R^2

Notice that Mallow's C_P was basically a measure of model fit $SSE_{\text{Reduced Model}}$, penalized by the number of variables in the model $2p$. Bigger models needed to produce increasingly large reductions in SSE to be "worth it".

The idea of balancing model fit with complexity leads us to the Adjusted R^2, R_a^2. Unlike Mallow's C_p, there's not really any fancy theory here:

$$R_a^2 = 1 - \frac{n-1}{n-p-1}(1 - R^2)$$

Here R^2 is for the reduced model, p is the number of covariates, and n is the number of observations. Notice that, for fixed R^2, increasing p decreases R_a^2.

Whereas C_p was *minimized*, R_a^2 should be *maximized*.

```
glance(mod.wt) %>% select(adj.r.squared)
```

```
## # A tibble: 1 x 1
##   adj.r.squared
##           <dbl>
## 1         0.794
```

```
glance(mod.hp) %>% select(adj.r.squared)
```

```
## # A tibble: 1 x 1
##   adj.r.squared
##           <dbl>
## 1         0.794
```

```
glance(mod.disp) %>% select(adj.r.squared)
```

```
## # A tibble: 1 x 1
##   adj.r.squared
##           <dbl>
## 1         0.811
```

```
glance(mod.cyl) %>% select(adj.r.squared)
```

```
## # A tibble: 1 x 1
##   adj.r.squared
##           <dbl>
## 1         0.789
```

Using AIC

One final quantity that we can use for variable selection (which we will see a lot more of when we get to GLMs) is the **Akaike Information Criteria** (AIC):

$$AIC_p = n \ln\left(\frac{SSE_p}{n}\right) + 2p$$

We won't go through the derivation or interpretation of this quantity today, however in the case of MLR it turns out to be equivalent to C_p.

```r
glance(mod.wt) %>% select(AIC)
```

```
## # A tibble: 1 x 1
##     AIC
##   <dbl>
## 1  162.
```

```r
glance(mod.hp) %>% select(AIC)
```

```
## # A tibble: 1 x 1
##     AIC
##   <dbl>
## 1  162.
```

```r
glance(mod.disp) %>% select(AIC)
```

```
## # A tibble: 1 x 1
##     AIC
##   <dbl>
## 1  159.
```

```r
glance(mod.cyl) %>% select(AIC)
```

```
## # A tibble: 1 x 1
##     AIC
##   <dbl>
## 1  162.
```

More Model Comparisons

```r
get.aic = function(mod){
        mod %>% glance %>% select(AIC) %>% return
}
mod1 = lm(mpg~gear+am+carb,data=mtcars)
mod2 = lm(mpg~gear+am+carb+disp,data=mtcars)
mod3 = lm(mpg~gear+am+carb+disp+wt,data=mtcars)
mod4 = lm(mpg~gear+am+carb+disp+hp,data=mtcars)
mod5 = lm(mpg~gear+am+carb+disp+cyl,data=mtcars)

mod1.aic = mod1 %>% get.aic
mod2.aic = mod2 %>% get.aic
mod3.aic = mod3 %>% get.aic
mod4.aic = mod4 %>% get.aic
mod5.aic = mod5 %>% get.aic

c(mod1.aic, mod2.aic, mod3.aic, mod4.aic, mod5.aic)
```

```
## $AIC
## [1] 168.5719
##
## $AIC
## [1] 158.9462
##
## $AIC
## [1] 159.6936
##
```

```
## $AIC
## [1] 160.3593
##
## $AIC
## [1] 160.4956
```