C1M6_peer_reviewed

March 5, 2025

1 Module 6: Peer Reviewed Assignment

1.0.1 Outline:

The objectives for this assignment:

1. Apply the processes of model selection with real datasets.
2. Understand why and how some problems are simpler to solve with some forms of model selection, and others are more difficult.
3. Be able to explain the balance between model power and simplicity.
4. Observe the difference between different model selection criterion.

General tips:

1. Read the questions carefully to understand what is being asked.
2. This work will be reviewed by another human, so make sure that you are clear and concise in what your explanations and answers.

[1]:
```
# This cell loads in the necesary packages
library(tidyverse)
library(leaps)
library(ggplot2)
```

```
  Attaching packages                                       tidyverse
1.3.0

 ggplot2 3.3.0       purrr   0.3.4
 tibble  3.0.1       dplyr   0.8.5
 tidyr   1.0.2       stringr 1.4.0
 readr   1.3.1       forcats 0.5.0

  Conflicts
tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()
```

1.1 Problem 1: We Need Concrete Evidence!

Ralphie is studying to become a civil engineer. That means she has to know everything about concrete, including what ingredients go in it and how they affect the concrete's properties. She's currently writting up a project about concrete flow, and has asked you to help her figure out which ingredients are the most important. Let's use our new model selection techniques to help Ralphie out!

Data Source: Yeh, I-Cheng, "Modeling slump flow of concrete using second-order regressions and artificial neural networks," Cement and Concrete Composites, Vol.29, No. 6, 474-480, 2007.

[2]:
```
concrete.data = read.csv("Concrete.data")

concrete.data = concrete.data[, c(-1, -9, -11)]
names(concrete.data) = c("cement", "slag", "ash", "water", "sp", "course.agg",
↪"fine.agg", "flow")

head(concrete.data)
```

A data.frame: 6 × 8

	cement <dbl>	slag <dbl>	ash <dbl>	water <dbl>	sp <dbl>	course.agg <dbl>	fine.agg <dbl>	flow <dbl>
1	273	82	105	210	9	904	680	62.0
2	163	149	191	180	12	843	746	20.0
3	162	148	191	179	16	840	743	20.0
4	162	148	190	179	19	838	741	21.5
5	154	112	144	220	10	923	658	64.0
6	147	89	115	202	9	860	829	55.0

1.1.1 1. (a) Initial Inspections

Sometimes, the best way to start is to just jump in and mess around with the model. So let's do that. Create a linear model with `flow` as the response and all other columns as predictors.

Just by looking at the summary for your model, is there reason to believe that our model could be simpler?

[3]:
```
lm.full <- lm(flow ~ cement + slag + ash + water + sp + course.agg + fine.agg,
↪data = concrete.data)
summary(lm.full)

print("Most p-values are greater than alpha = 0.05.")

# Your Code Here
```

```
Call:
lm(formula = flow ~ cement + slag + ash + water + sp + course.agg +
    fine.agg, data = concrete.data)

Residuals:
```

```
    Min      1Q  Median      3Q     Max
-30.880 -10.428   1.815   9.601  22.953

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -252.87467  350.06649  -0.722   0.4718
cement         0.05364    0.11236   0.477   0.6342
slag          -0.00569    0.15638  -0.036   0.9710
ash            0.06115    0.11402   0.536   0.5930
water          0.73180    0.35282   2.074   0.0408 *
sp             0.29833    0.66263   0.450   0.6536
course.agg     0.07366    0.13510   0.545   0.5869
fine.agg       0.09402    0.14191   0.663   0.5092
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.84 on 95 degrees of freedom
Multiple R-squared:  0.5022,Adjusted R-squared:  0.4656
F-statistic: 13.69 on 7 and 95 DF,  p-value: 3.915e-12



[1] "Most p-values are greater than alpha = 0.05."
```

1.1.2 1. (b) Backwards Selection

Our model has 7 predictors. That is not too many, so we can use backwards selection to narrow them down to the most impactful.

Perform backwards selection on your model. You don't have to automate the backwards selection process.

[4]:
```
lm.full <- lm(flow ~ cement + slag + ash + water + sp + course.agg + fine.agg,
 ↪data = concrete.data)
summary(lm(flow ~ cement + slag + ash + water + sp + course.agg + fine.agg,
 ↪data = concrete.data))
summary(lm(flow ~ cement + slag + ash + water + sp + course.agg, data =
 ↪concrete.data))
summary(lm(flow ~ cement + slag + ash + water + sp, data = concrete.data))
summary(lm(flow ~ cement + slag + ash + water, data = concrete.data))
summary(lm(flow ~ cement + slag + ash, data = concrete.data))
summary(lm(flow ~ cement + slag, data = concrete.data))
summary(lm(flow ~ cement, data = concrete.data))

# Your Code Here
```

```
Call:
lm(formula = flow ~ cement + slag + ash + water + sp + course.agg +
```

```
    fine.agg, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-30.880 -10.428   1.815   9.601  22.953 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)  
(Intercept) -252.87467  350.06649  -0.722   0.4718  
cement         0.05364    0.11236   0.477   0.6342  
slag          -0.00569    0.15638  -0.036   0.9710  
ash            0.06115    0.11402   0.536   0.5930  
water          0.73180    0.35282   2.074   0.0408 *
sp             0.29833    0.66263   0.450   0.6536  
course.agg     0.07366    0.13510   0.545   0.5869  
fine.agg       0.09402    0.14191   0.663   0.5092  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 12.84 on 95 degrees of freedom
Multiple R-squared:  0.5022,Adjusted R-squared:  0.4656 
F-statistic: 13.69 on 7 and 95 DF,  p-value: 3.915e-12 



Call:
lm(formula = flow ~ cement + slag + ash + water + sp + course.agg, 
    data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-31.788 -10.183   1.821   9.422  23.252 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -22.54730   40.87442  -0.552 0.582488    
cement       -0.01905    0.02412  -0.790 0.431520    
slag         -0.10746    0.02921  -3.679 0.000386 ***
ash          -0.01296    0.02198  -0.590 0.556781    
water         0.50572    0.08934   5.660 1.56e-07 ***
sp            0.01029    0.49859   0.021 0.983585    
course.agg   -0.01465    0.02192  -0.668 0.505530    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 12.81 on 96 degrees of freedom
Multiple R-squared:  0.4999,Adjusted R-squared:  0.4687 
F-statistic:    16 on 6 and 96 DF,  p-value: 1.141e-12 
```

```
Call:
lm(formula = flow ~ cement + slag + ash + water + sp, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-30.973 -10.567   1.813   8.794  24.087 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -47.410133  16.885337  -2.808 0.006032 ** 
cement       -0.011299   0.021086  -0.536 0.593287    
slag         -0.098448   0.025833  -3.811 0.000243 ***
ash          -0.007367   0.020266  -0.364 0.717019    
water         0.545793   0.066039   8.265 7.32e-13 ***
sp            0.091285   0.482252   0.189 0.850262    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 12.77 on 97 degrees of freedom
Multiple R-squared:  0.4976,Adjusted R-squared:  0.4717 
F-statistic: 19.21 on 5 and 97 DF,  p-value: 3.026e-13


Call:
lm(formula = flow ~ cement + slag + ash + water, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-31.192 -10.559   1.722   8.965  24.448 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -46.159182  15.461704  -2.985 0.003577 ** 
cement       -0.011580   0.020931  -0.553 0.581362    
slag         -0.097463   0.025178  -3.871 0.000196 ***
ash          -0.007819   0.020025  -0.390 0.697053    
water         0.543682   0.064769   8.394 3.63e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 12.71 on 98 degrees of freedom
Multiple R-squared:  0.4974,Adjusted R-squared:  0.4769 
F-statistic: 24.25 on 4 and 98 DF,  p-value: 5.801e-14
```

```
Call:
lm(formula = flow ~ cement + slag + ash, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max
-37.700 -13.841   2.107  12.102  32.762

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 63.126066  10.880165   5.802 7.89e-08 ***
cement       0.001988   0.027222   0.073  0.94194
slag        -0.110772   0.032779  -3.379  0.00104 **
ash         -0.035802   0.025758  -1.390  0.16766
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.58 on 99 degrees of freedom
Multiple R-squared:  0.1361,Adjusted R-squared:  0.1099
F-statistic: 5.197 on 3 and 99 DF,  p-value: 0.002244



Call:
lm(formula = flow ~ cement + slag, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max
-38.980 -15.453   2.269  13.384  30.488

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 50.58702    6.11074   8.278  5.7e-13 ***
cement       0.02528    0.02155   1.173  0.24362
slag        -0.08705    0.02812  -3.096  0.00254 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.65 on 100 degrees of freedom
Multiple R-squared:  0.1192,Adjusted R-squared:  0.1016
F-statistic: 6.766 on 2 and 100 DF,  p-value: 0.001754



Call:
lm(formula = flow ~ cement, data = concrete.data)

Residuals:
    Min      1Q  Median      3Q     Max
-33.851 -11.510   6.606  12.506  29.868
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 40.06293    5.28948   7.574 1.77e-11 ***
cement       0.04153    0.02177   1.907   0.0593 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 17.35 on 101 degrees of freedom
Multiple R-squared:  0.03477,Adjusted R-squared:  0.02521
F-statistic: 3.638 on 1 and 101 DF,  p-value: 0.05932
```

1.1.3 1. (c) Objection!

Stop right there! Think about what you just did. You just removed the "worst" features from your model. But we know that a model will become less powerful when we remove features so we should check that it's still just as powerful as the original model. Use a test to check whether the model at the end of backward selection is significantly different than the model with all the features.

Describe why we want to balance explanatory power with simplicity.

[5]:
```
n = floor(0.8*nrow(concrete.data))
index = sample(seq_len(nrow(concrete.data)), size = n)
train = concrete.data[index, ]
train_sum <- summary(train)


lm.model <- lm(flow ~ cement + ash + water + course.agg + fine.agg, data =␣
 ↪concrete.data)

anova(lm.full, lm.model)


print("The p-value is greater than the significance level alpha = 0.05 so we␣
 ↪fail to reject the null hypothesis.  There is not significant difference␣
 ↪between the models.")




# Your Code Here
```

A anova: 2 × 6

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	95	15671.26	NA	NA	NA	NA
2	97	15733.53	-2	-62.27123	0.1887457	0.8283068

```
[1] "The p-value is greater than the significance level alpha = 0.05 so we fail
to reject the null hypothesis.  There is not significant difference between the
models."
```

1.1.4 1. (d) Checking our Model

Ralphie is nervous about her project and wants to make sure our model is correct. She's found a function called `regsubsets()` in the leaps package which allows us to see which subsets of arguments produce the best combinations. Ralphie wrote up the code for you and the documentation for the function can be found here. For each of the subsets of features, calculate the AIC, BIC and adjusted R^2. Plot the results of each criterion, with the score on the y-axis and the number of features on the x-axis.

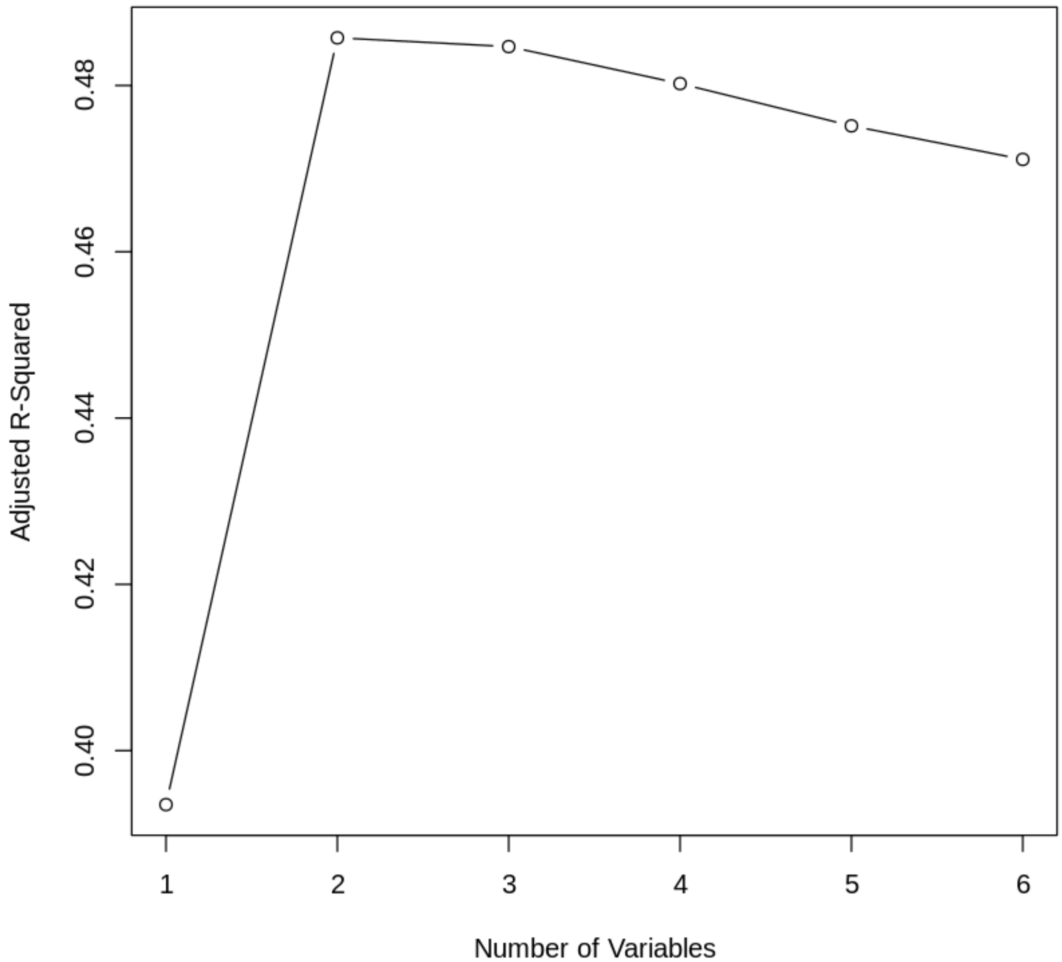
Do all of the criterion agree on how many features make the best model? Explain why the criterion will or will not always agree on the best model.

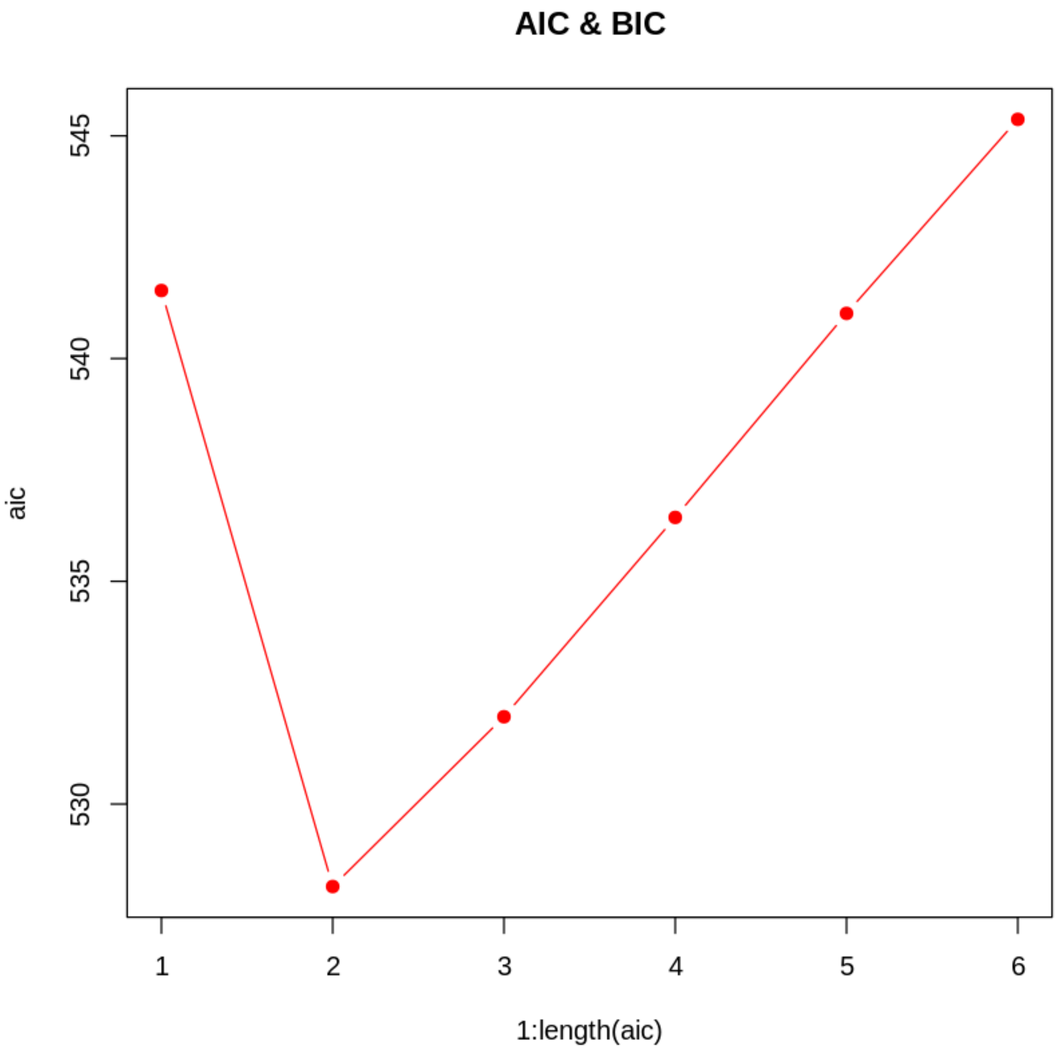
Hint: It may help to look at the attributes stored within the regsubsets summary using `names(rs)`.

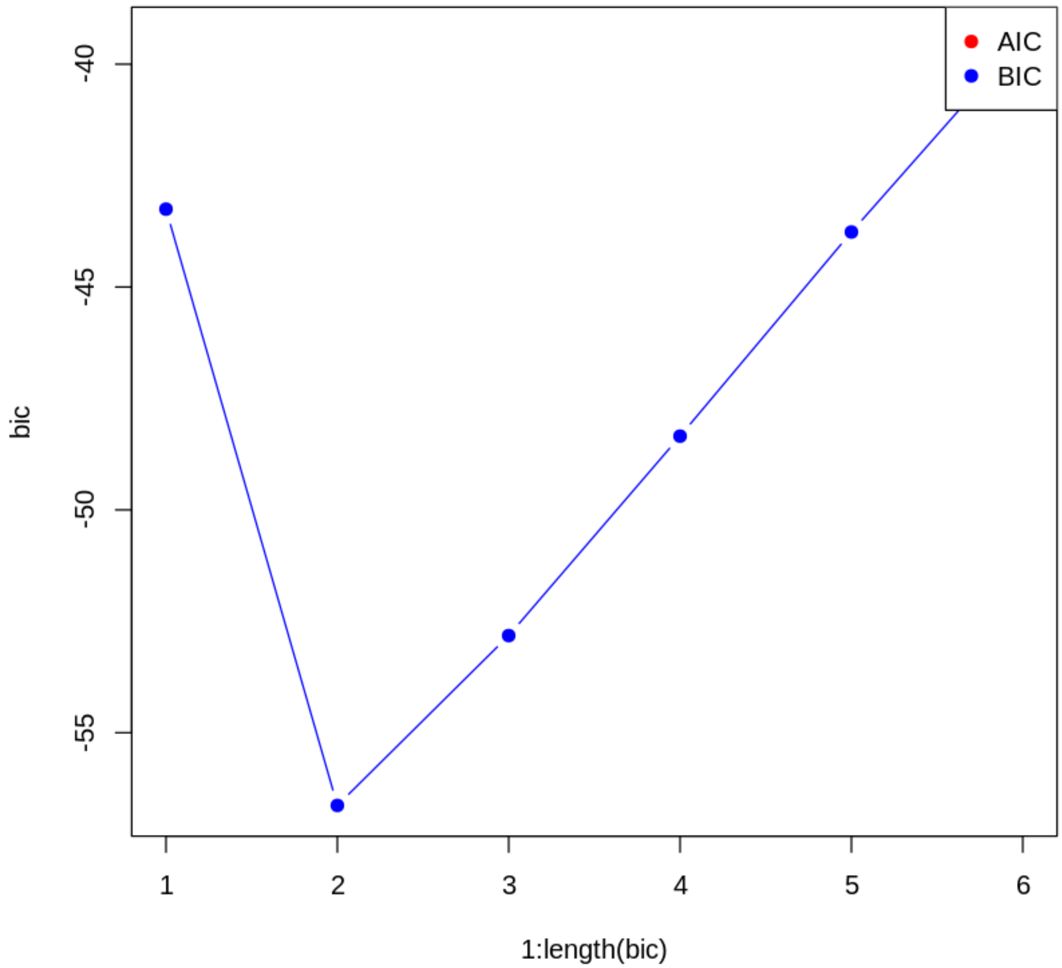
[6]:
```
reg = regsubsets(flow ~ cement+slag+ash+water+sp+course.agg+fine.agg+flow,
 ↪data=concrete.data, nvmax=6)
rs = summary(reg)
names(rs)
plot(rs$adjr2, xlab = "Number of Variables", ylab = "Adjusted R-Squared", type
 ↪= "b")

n <- nrow(concrete.data)
aic <- sapply(1:length(rs$rss), function(k){
    rss <- rs$rss[k]
    n * log(rss/n) + k * log(n)
})

bic <- rs$bic


plot(1:length(aic), aic, type = "b", pch = 19, col = "red", main = "AIC & BIC")
 ↪+ plot(1:length(bic), bic, type = "b", pch = 19, col = "blue")
legend("topright", legend = c("AIC", "BIC"), col = c("red", "blue"), pch = 19)



# Your Code Here
```

```
Warning message in model.matrix.default(terms(formula, data = data), mm):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(terms(formula, data = data), mm):
"problem with term 8 in model.matrix: no columns are assigned"
```

1. 'which' 2. 'rsq' 3. 'rss' 4. 'adjr2' 5. 'cp' 6. 'bic' 7. 'outmat' 8. 'obj'


0.40
0.42
0.44
0.46
0.48
Adjusted R-Squared
1
2
3
4
5
6
Number of Variables


AIC & BIC
aic
545
540
535
530
1
2
3
4
5
6
1:length(aic)



[7]:
```
print("The model with the higher adjusted R2 would most likely be a better fit␣
↪and have more predictors. The best model according to AIC and BIC depend on␣
↪the fit and complexity of each of the models.")
```

```
[1] "The model with the higher adjusted R2 would most likely be a better fit and
have more predictors. The best model according to AIC and BIC depend on the fit
and complexity of each of the models."
```

[ ]: