C1M6_autograded

January 15, 2022

1 Module 6: Autograded Assignment

1.0.1 Outline:

Here are the objectives of this assignment:

1. Apply model selection techniques to various data sets.
2. Learn how to calculate and interpret different model selection criterion.
3. Prove to yourself that you have learned how to apply, interpret and optimize statistical models.
4. Apply variance inflation factors to analyze multicollinearity issues.

Here are some general tips:

1. Read the questions carefully to understand what is being asked.
2. When you feel that your work is completed, feel free to hit the `Validate` button to see your results on the *visible* unit tests. If you have questions about unit testing, please refer to the "Module 0: Introduction" notebook provided as an optional resource for this course. In this assignment, there are hidden unit tests that check your code. You will not recieve any feedback for failed hidden unit tests until the assignment is submitted. **Do not misinterpret the feedback from visible unit tests as all possible tests for a given question–write your code carefully!**
3. Before submitting, we recommend restarting the kernel and running all the cells in order that they appear to make sure that there are no additional bugs in your code.
4. There are 70 total points in this assignment.

[39]:
```
# This cell loads the required packages
library(testthat)
library(tidyverse)
library(ggplot2)
library(leaps)
library(MASS)
library(regclass)
library(faraway)
```

2 Problem 1: Model Selection Criterion

In this lesson, we will perform both the full and partial F-tests in R.

Recall again, the Amazon book data. The data consists of data on $n = 325$ books and includes measurements of:

- `aprice`: The price listed on Amazon (dollars)
- `lprice`: The book's list price (dollars)
- `weight`: The book's weight (ounces)
- `pages`: The number of pages in the book
- `height`: The book's height (inches)
- `width`: The book's width (inches)
- `thick`: The thickness of the book (inches)
- `cover`: Whether the book is a hard cover of paperback.
- And other variables...

Before we do any model selection, we'll repeat the data cleaning methods from the previous lesson on this dataset. For all tests in this lesson, let $\alpha = 0.05$.

[40]:
```
amazon = read.csv("amazon.txt", sep="\t")
df = data.frame(aprice = amazon$Amazon.Price, lprice = as.numeric(amazon$List.
↪Price),
                pages = amazon$NumPages, width = amazon$Width, weight =␣
↪amazon$Weight..oz,
                height = amazon$Height, thick = amazon$Thick, cover =␣
↪amazon$Hard..Paper)

df$lprice[which(is.na(df$lprice))] = mean(df$lprice, na.rm = TRUE)
df$weight[which(is.na(df$weight))] = mean(df$weight, na.rm = TRUE)
df$pages[which(is.na(df$pages))] = mean(df$pages, na.rm = TRUE)
df$height[which(is.na(df$height))] = mean(df$height, na.rm = TRUE)
df$width[which(is.na(df$width))] = mean(df$width, na.rm = TRUE)
df$thick[which(is.na(df$thick))] = mean(df$thick, na.rm = TRUE)
df = df[-205,]
summary(df)
```

```
     aprice            lprice            pages            width      
 Min.   :  0.770   Min.   :  1.50   Min.   : 24.0   Min.   :4.100  
 1st Qu.:  8.598   1st Qu.: 13.95   1st Qu.:208.0   1st Qu.:5.200  
 Median : 10.200   Median : 15.00   Median :320.0   Median :5.400  
 Mean   : 13.010   Mean   : 18.58   Mean   :335.8   Mean   :5.584  
 3rd Qu.: 13.033   3rd Qu.: 19.95   3rd Qu.:416.0   3rd Qu.:5.900  
 Max.   :139.950   Max.   :139.95   Max.   :896.0   Max.   :9.500  
     weight          height           thick         cover  
 Min.   : 1.20   Min.   : 5.100   Min.   :0.100   H: 89  
 1st Qu.: 7.80   1st Qu.: 7.900   1st Qu.:0.600   P:235  
 Median :11.20   Median : 8.100   Median :0.900          
 Mean   :12.48   Mean   : 8.161   Mean   :0.908          
```

```
 3rd Qu.:16.00   3rd Qu.: 8.500   3rd Qu.:1.100  
 Max.   :35.20   Max.   :12.100   Max.   :2.100  
```

2.0.1 1. (a) The Model (15 points)

We want to determine which predictors impact the Amazon list price. Begin by fitting the full model.

Fit a model named `lmod.full` to the data with `aprice` as the response and all other columns as predictors. Then calculate the AIC, BIC and adjusted R^2 for this model. Store these values in `AIC.full`, `BIC.full` and `adj.R2.full` respectively.

[41]:
```
AIC.full = NA
BIC.full = NA
adj.R2.full = NA

# your code here
lmod.full = lm(aprice~., data =df)
reg_sum = summary(lmod.full)
n = dim(df)[1]
pred_count = length(coef(lmod.full))
rss = sum(resid(lmod.full)^2)
reg_sum
rss
```

```
Call:
lm(formula = aprice ~ ., data = df)

Residuals:
     Min       1Q   Median       3Q      Max 
-20.3927  -1.8341  -0.3855   1.4233  22.2885 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.790394   2.483623  -0.721  0.47152    
lprice       0.854647   0.018251  46.828  < 2e-16 ***
pages       -0.001128   0.002471  -0.456  0.64840    
width        0.158748   0.307261   0.517  0.60576    
weight      -0.071535   0.048775  -1.467  0.14347    
height      -0.030707   0.285790  -0.107  0.91450    
thick       -1.677617   1.132703  -1.481  0.13958    
coverP       1.489428   0.569327   2.616  0.00932 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.639 on 316 degrees of freedom
Multiple R-squared:  0.9163,Adjusted R-squared:  0.9145
```

```
F-statistic: 494.4 on 7 and 316 DF,  p-value: < 2.2e-16
```

4185.26316840452

[42]:
```
AIC.full = 2 * pred_count + n * log(rss/n)
```

[43]:
```
BIC.full = log(n) * pred_count + n * log(rss/n)
```

[44]:
```
adj.R2.full = reg_sum$adj.r.squared
```

[45]:
```
# Test Cell
# Check that the correct number of predictors were used in the model.
if(test_that("Check number of model parameters.", expect_equal(length(lmod.
↪full$coefficients), 8))){
    print("Correct number of parameters in the model.")
}else{
    print("Make sure you're not using the Port column!")
}
# This cell has hidden test cases that will run after submission.
```

```
[1] "Correct number of parameters in the model."
```

[46]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[47]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.0.2 1. (b) A Partial Model (15 points)

Fit a partial model to the data, with `aprice` as the response and `lprice`, and `pages` as predictors. Calculate the AIC, BIC and adjusted R^2 for this partial model. Store their values in `AIC.part`, `BIC.part` and `adj.R2.part` respectively.

[48]:
```
AIC.part = NA
BIC.part = NA
adj.R2.part = NA

# your code here
lmod.part = lm(aprice~lprice+pages, data=df)
reg_sum.part = summary(lmod.part)

pred_count.part = length(coef(lmod.part))
rss.part = sum(resid(lmod.part)^2)
```

[49]:
```
AIC.part = 2 * pred_count.part + n * log(rss.part/n)
```

[50]:
```
BIC.part = log(n) * pred_count.part + n * log(rss.part/n)
```

[51]:
```
adj.R2.part = reg_sum.part$adj.r.squared
```

[52]:
```
AIC.full
BIC.full
adj.R2.full
print("---------------")
AIC.part
BIC.part
adj.R2.part
```

844.980357038035

875.226305164374

0.914482534685603

```
[1] "---------------"
```

863.768401306743

875.11063185412

0.907992167738236

[53]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[54]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[55]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.0.3 1. (c) Model Selection (9 points)

Which model is better, `lmod.full` or `lmod.part` according to AIC, BIC, and R_a^2? Note that the answer may or may not be different across the different criteria. Save your selections as `selected.model.AIC`, `selected.model.BIC`, and `selected.model.adj.R2`.

[56]:
```
selected.model.AIC = lmod.full
selected.model.BIC = lmod.part
selected.model.adj.R2 = lmod.full
# your code here
```

[57]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[58]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[59]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.0.4 1. (d) Model Validation (6 points)

Recall that a simpler model may perform statistically worse than a larger model. Test whether there is a statistically significant difference between `lmod.part` and `lmod.full`. Based on the result of this test, what model should you use? Save your answer as `validated.model`.

[60]:
```
anova(lmod.part, lmod.full)
```

A anova: 2 × 6

	Res.Df <dbl>	RSS <dbl>	Df <dbl>	Sum of Sq <dbl>	F <dbl>	Pr(>F) <dbl>
1	321	4574.153	NA	NA	NA	NA
2	316	4185.263	5	388.89	5.872473	3.312548e-05

[61]:
```
validated.model = NA

# your code here
validated.model = lmod.full
```

[62]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.1 Problem 2

`divorce` is a data frame with 77 observations on the following 7 variables.

1. `year`: the year from 1920-1996
2. `divorce`: divorce per 1000 women aged 15 or more
3. `unemployed` unemployment rate
4. `femlab`: percent female participation in labor force aged 16+
5. `marriage`: marriages per 1000 unmarried women aged 16+
6. `birth`: births per 1000 women aged 15-44
7. `military`: military personnel per 1000 population

Here's the data:

[63]:
```
# Load in the data
divorce = read.csv("divusa.txt", sep="\t")
summary(divorce)
head(divorce)
```

```
      year         divorce        unemployed         femlab
 Min.   :1920   Min.   : 6.10   Min.   : 1.200   Min.   :22.70
 1st Qu.:1939   1st Qu.: 8.70   1st Qu.: 4.200   1st Qu.:27.47
 Median :1958   Median :10.60   Median : 5.600   Median :37.10
 Mean   :1958   Mean   :13.27   Mean   : 7.173   Mean   :38.58
 3rd Qu.:1977   3rd Qu.:20.30   3rd Qu.: 7.500   3rd Qu.:47.80
 Max.   :1996   Max.   :22.80   Max.   :24.900   Max.   :59.30
    marriage          birth           military
 Min.   : 49.70   Min.   : 65.30   Min.   : 1.940
 1st Qu.: 61.90   1st Qu.: 68.90   1st Qu.: 3.469
 Median : 74.10   Median : 85.90   Median : 9.102
 Mean   : 72.97   Mean   : 88.89   Mean   :12.365
 3rd Qu.: 80.00   3rd Qu.:107.30   3rd Qu.:14.266
 Max.   :118.10   Max.   :122.90   Max.   :86.641
```

A data.frame: 6 × 7

	year <int>	divorce <dbl>	unemployed <dbl>	femlab <dbl>	marriage <dbl>	birth <dbl>	military <dbl>
1	1920	8.0	5.2	22.70	92.0	117.9	3.2247
2	1921	7.2	11.7	22.79	83.0	119.8	3.5614
3	1922	6.6	6.7	22.88	79.7	111.2	2.4553
4	1923	7.1	2.4	22.97	85.2	110.5	2.2065
5	1924	7.2	5.0	23.06	80.3	110.9	2.2889
6	1925	7.2	3.2	23.15	79.2	106.6	2.1735

2.1.1 2 (a) (10 points)

Using the **divorce** data, with **divorce** as the response and all other variables as predictors, select the "best" regression model, where "best" is defined using AIC. Save your final model as lm_divorce.**

[72]:
```
n = dim(divorce)[1]
divorce.reg = regsubsets(divorce~., data=divorce)
divorce.rs = summary(divorce.reg)
divorce.rs$which
```

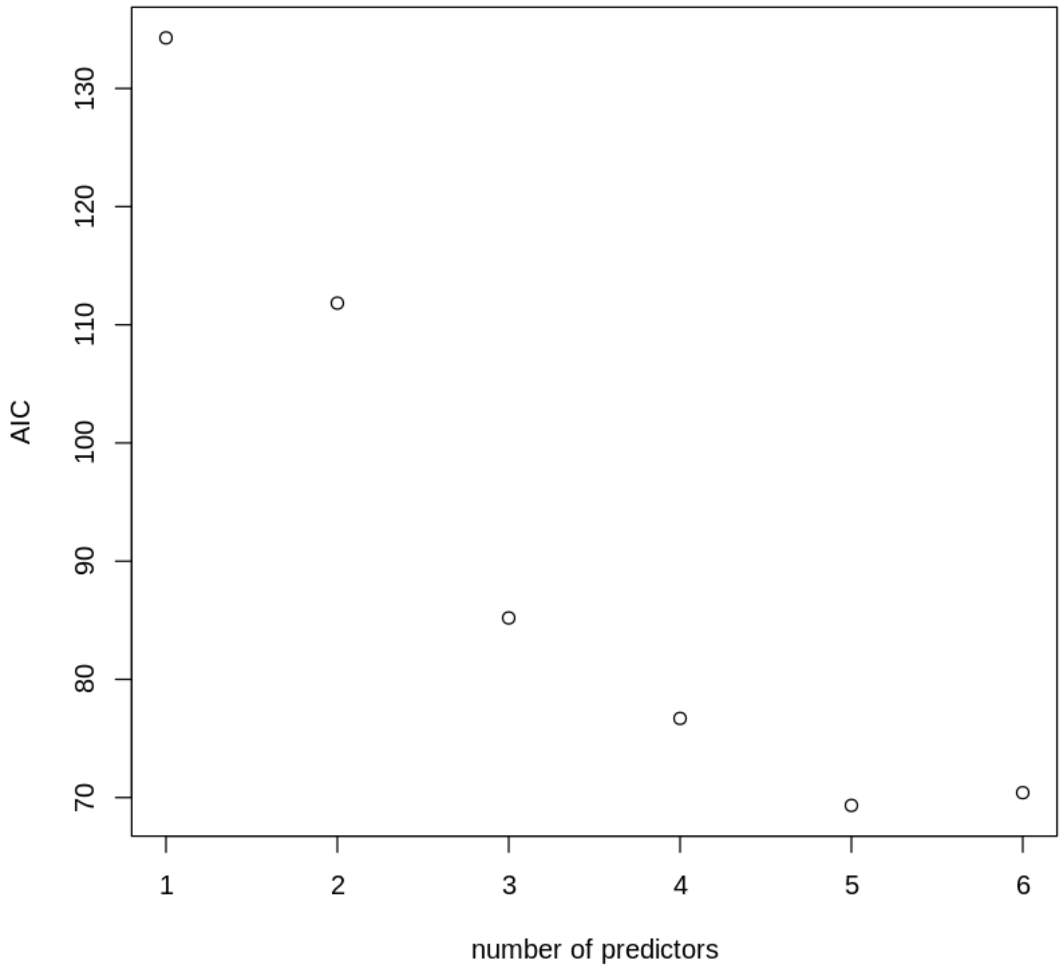
A matrix: 6 × 7 of type lgl

	(Intercept)	year	unemployed	femlab	marriage	birth	military
1	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE
2	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE
3	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE
4	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE
5	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
6	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

[75]:

```r
lm_divorce = NA

# your code here

AIC = 2*(2:7) + n*log(divorce.rs$rss/n)
plot(AIC ~ I(1:6), xlab = "number of predictors", ylab = "AIC")

lm_divorce = lm(divorce~. - unemployed, data=divorce)
summary(lm_divorce)
```

```
Call:
lm(formula = divorce ~ . - unemployed, data = divorce)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.7586 -1.0494 -0.0424  0.7201  3.3075 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 405.61670   95.13189   4.264 6.09e-05 ***
year         -0.21790    0.05078  -4.291 5.52e-05 ***
femlab        0.85480    0.10276   8.318 4.29e-12 ***
marriage      0.15934    0.02140   7.447 1.76e-10 ***
birth        -0.11012    0.01266  -8.700 8.43e-13 ***
military     -0.04120    0.01360  -3.030  0.00341 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.511 on 71 degrees of freedom
Multiple R-squared:  0.9336,Adjusted R-squared:  0.929 
F-statistic: 199.7 on 5 and 71 DF,  p-value: < 2.2e-16
```



[65]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.1.2 2 (b) (10 points)

Using your model from part (a), compute the variance inflation factors VIFs for each $\widehat{\beta}_j$, $j = 1, ..., p$. Store them in the variable `v`. Also, compute the condition number for the design matrix, stored in `k`. If using the `kappa()` function, you might need to specify `exact = TRUE`. Is there evidence that collinearity causes some predictors not to be significant?

[79]:

```
X = as.matrix(model.matrix(~year + femlab + marriage + birth + military, 
 ↪divorce))
X
```

A matrix: 77 × 6 of type dbl

	(Intercept)	year	femlab	marriage	birth	military
1	1	1920	22.70	92.0	117.9	3.2247
2	1	1921	22.79	83.0	119.8	3.5614
3	1	1922	22.88	79.7	111.2	2.4553
4	1	1923	22.97	85.2	110.5	2.2065
5	1	1924	23.06	80.3	110.9	2.2889
6	1	1925	23.15	79.2	106.6	2.1735
7	1	1926	23.24	78.7	102.6	2.1073
8	1	1927	23.33	77.0	99.8	2.0913
9	1	1928	23.42	74.1	93.8	2.0821
10	1	1929	23.51	75.5	89.3	2.0944
11	1	1930	23.60	67.6	89.2	2.0753
12	1	1931	24.03	61.9	84.6	2.0347
13	1	1932	24.46	56.0	81.7	1.9600
14	1	1933	24.89	61.3	76.3	1.9401
15	1	1934	25.32	71.8	78.5	1.9539
16	1	1935	25.75	72.5	77.2	1.9770
17	1	1936	26.18	74.0	75.8	2.2730
18	1	1937	26.61	78.0	77.1	2.4178
19	1	1938	27.04	69.9	79.1	2.4847
20	1	1939	27.47	73.0	77.6	2.5527
21	1	1940	27.90	82.8	79.9	3.4693
22	1	1941	28.50	88.5	83.4	13.5013
23	1	1942	30.90	93.0	91.5	28.6133
24	1	1943	35.70	83.0	94.3	66.1461
25	1	1944	36.30	76.5	88.8	82.7454
26	1	1945	35.80	83.6	85.9	86.6407
27	1	1946	30.80	118.1	101.9	21.4309
28	1	1947	31.80	106.2	113.3	10.9834
29	1	1948	32.70	98.5	107.3	9.8609
30	1	1949	33.20	86.7	107.1	10.8277
48	1	1967	41.1	76.4	87.6	16.9938
49	1	1968	41.6	79.1	85.7	17.6771
50	1	1969	42.7	80.0	86.5	17.0723
51	1	1970	43.4	76.5	87.9	14.9537
52	1	1971	42.8	76.2	81.8	13.0648
53	1	1972	43.4	77.9	73.4	11.0624
54	1	1973	44.2	76.0	69.2	10.6269
55	1	1974	45.1	72.0	68.4	10.1097
56	1	1975	46.3	66.9	66.7	9.8536
57	1	1976	46.8	65.2	65.8	9.5485
58	1	1977	47.8	63.6	66.8	9.4195
59	1	1978	49.3	64.1	66.6	9.2626
60	1	1979	50.3	63.6	67.2	9.0062
61	1	1980	51.5	61.4	68.4	9.0247
62	1	1981	52.1	61.7	67.4	9.0757
63	1	1982	52.6	61.4	67.3	9.1020
64	1	1983	52.9	59.9	65.8	9.0822
65	1	1984	53.6	59.5	65.4	9.0667
66	1	1985	54.5	57.0	66.2	9.0408
67	1	1986	55.3	56.2	65.4	9.0330
68	1	1987	56.0	55.7	65.7	8.9737

[80]:
```
# your code here
v = vif(lm_divorce)
k = kappa(X, exact=TRUE)
v
k
```

year 42.9482669472054 **femlab** 48.6509347631077 **marriage** 2.62453070971822 **birth** 2.03167683563951 **military** 1.35800179637071

1083823.31588984

[81]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

2.1.3 2. (c) (5 points)

Remove the predictor with the highest VIF. Is multicollinearity still present in the model? If yes, store `TRUE` in `prob.2.c`, and `FALSE` otherwise.

[82]:
```
lm.new.mod = lm(divorce~year + marriage + birth + military, data=divorce)
X2 = as.matrix(model.matrix(~year + marriage + birth + military, divorce))

vif(lm.new.mod)
kappa(X2, exacct=TRUE)
```

year 1.83370639925005 **marriage** 2.33192135504654 **birth** 1.9825411070497 **military** 1.12580667212605

135342.421396956

[83]:
```
prob.2.c =TRUE

# your code here
```

[84]:
```
# Test Cell
# This cell has hidden test cases that will run after submission.
```

[ ]:
```
```