Analytics 512 Homework 4

Arif Ali
03/10/2015

Exercise 5.4 #3

Part A

K fold validation is implimented by first randomly dividing the sets of observations into k groups that should be about equal sizes. The first group (or fold) is treated as the validation set and the method is fit on the remaining k-1 folds. Then the mean-squared error (MSE) is computed for the fold, this method is then repeated on the remaining k-1 folds.

The CV estimate is computed by averaging the various MSEs

$$CV_k = \frac{1}{k} \sum_{i=1}^{k} MSE_i$$

Part B

i

The advantages of k fold over the validation set approach are:

- the validation estimate of the test error rate can be highly variable
- There's only one subset used to test and train, which makes it easy to implement

The disadvantages of k fold over the validation set approach are:

- The validation method is significiantly easier to impliment since we are splitting only two data sets
- The validation method has significiantly less bias compared to the k-fold method.

ii

The disadvantages of k fold over LOOCV approach are:

The LOOCV is significantly more comprehensive compared to the k-fold method

has a low bias

The advantages of k fold over LOOCV approach are:

While LOOCV has a low bias, the test error can be highly variable as well

LOOCV requires going trhough every point, so it requires more time to implement than k-fold

Exercsise 5.4 #5

In [35]:
```
library(ISLR)
attach(Default)
```

```
The following objects are masked from Default (pos = 3):

    balance, default, income, student

The following objects are masked from Default (pos = 6):

    balance, default, income, student
```

Part A

In [36]:
```
glm.default = glm(default~balance+income, data = Default, family = binomial)
```

Part B

i

In [37]:
```
train = sample(nrow(Default), .5*nrow(Default), replace = F)
training = Default[train,]
validation = Default[-train,]
```

ii

In [38]:
```
glm.default = glm(default~., data = training, family = binomial)
```

iii

In [39]:
```
pred = predict(glm.default, validation, type = "response")
pred.default = rep(x = "No", times = length(pred))
pred.default[pred>0.5] = "Yes"
```

iv

In [40]:
```
mean(pred.default != validation$default)
```

Out[40]: 0.0292

I attempted CV by applying K = 10 instead of the validation method as specified by the homework.

In [41]:
```
glm.default = glm(default~income+balance, data = Default, family = binomial)
cost <- function(r, pi = 0){
    mean(abs(r-pi) > 0.5)
    }
library(boot)
cv.default = cv.glm(Default, glm.default, K = 10)
cv.default$delta
```

Out[41]: 0.0214718792280596 0.0214682053025623

Part C

In [42]:
```
glm.default = glm(default~income+balance, data = Default, family = binomial)
cost <- function(r, pi = 0){
    mean(abs(r-pi) > 0.5)
    }
library(boot)
cv.default = cv.glm(Default, glm.default, cost, K = 10)
cv.default$delta

glm.default = glm(default~income+balance, data = Default, family = binomial)
cost <- function(r, pi = 0){
    mean(abs(r-pi) > 0.5)
    }
library(boot)
cv.default = cv.glm(Default, glm.default, cost, K = 10)
cv.default$delta

glm.default = glm(default~income+balance, data = Default, family = binomial)
cost <- function(r, pi = 0){
    mean(abs(r-pi) > 0.5)
    }
library(boot)
cv.default = cv.glm(Default, glm.default, cost, K = 10)
cv.default$delta
```

Out[42]: 0.0263 0.02634

Out[42]: 0.0264 0.02642

Out[42]: 0.0265 0.02645

Based on the three runs of the, it seems that the CV error holds between 0.026 to 0.0265.

Part D

In [43]:
```
glm.default = glm(default~income+balance+student, data = Default, family = bin
omial)
cost <- function(r, pi = 0){
    mean(abs(r-pi) > 0.5)
    }
library(boot)
cv.default = cv.glm(Default, glm.default, cost, K = 10)
cv.default$delta
```

Out[43]: 0.0268 0.02694

The difference in error seems to by minimal, but lower without the student dummy variable. However, I would argue that the differences between the CV errors are small enough that it wouldn't matter. As to the specific question, adding the student dummy variable does not lead to a reduction in the test errors.

Exercise 6.8 #1

Part A

In the case of Best Subset selection, "This task must be performed with care, because the RSS of these p + 1 models decreases monotonically, and the R^2 increases monotonically, as the number of features included in the models increases." Thus Best Subset selection has the smallest training RSS.

Part B

Best Subset Selection would most likely have the smallest test RSS, but it's possible for forward stepwise or backwards stepwise to have a smaller test RSS, so the model with the smallest test RSS cannot be determined.

Part C

i

True, forward selection adds predictors to a model until it goes through each predictor, adding them if they meet the specified criteria.

ii

True, Backward selection starts by looking at all $k + 1$ predictors, removing ones that don't fit a criteria.

iii

False, the methods could lead to different models thus one cannot be a subset of the other.

iv

False,the methods could lead to different models thus one cannot be a subset of the other.

v

False, since best subset selection looks at all possible combinationa, there are multiple models with k+1 predictors, so it doesn't the same values as the lower subsets.

Exercise 6.8 #8

Part A

In [44]:
```
X = rnorm(100)
e = rnorm(100)
```

Part B

In [45]:
```
Y = 1 +2*X+3*X^2+4*X^3+e
```

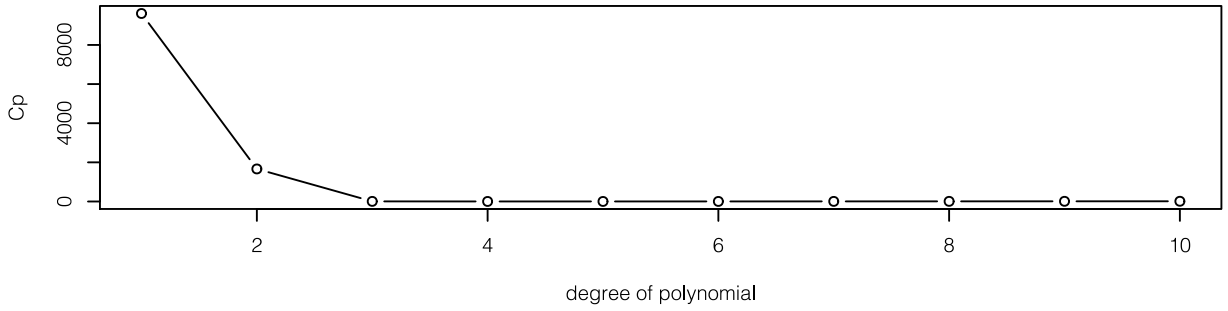
Part C

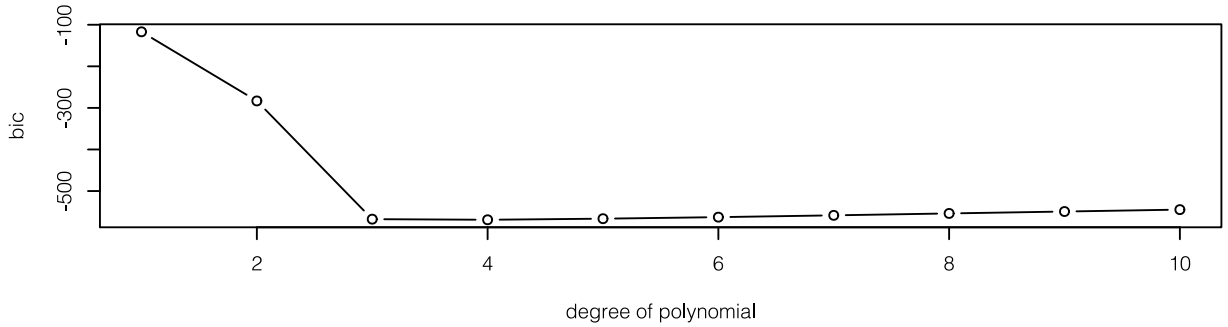
In [46]:
```
XY = data.frame(Y,X)
names(XY)
library(leaps)
best_selection = regsubsets(Y~poly(X, 10), data = XY,nvmax=10)
```

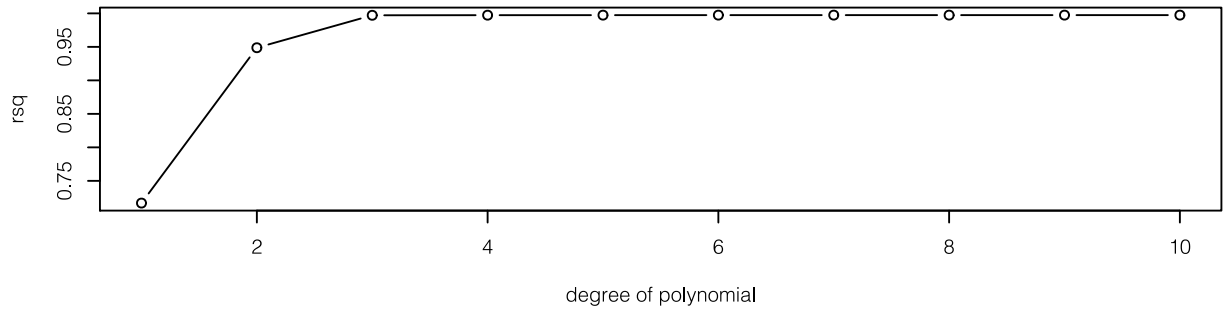
Out[46]: 'Y' 'X'

In [47]:
```
best_selection_summary = summary(best_selection)
```

In [48]:
```
par(mfrow = c(3,1))
plot(best_selection_summary$cp, type = "b", xlab = "degree of polynomial" , yl
ab = "Cp")
plot(best_selection_summary$bic,type = "b", xlab = "degree of polynomial" , yl
ab = "bic")
plot(best_selection_summary$rsq, type = "b", xlab = "degree of polynomial" , y
lab = "rsq")
```







In [49]:
```
print("Based on Cp, the best model is:")
coef(best_selection,which.min(best_selection_summary$cp))
print("Based on BIC, the best model is:")
coef(best_selection,which.min(best_selection_summary$bic))
print("Based on R squared, the best model is:")
coef(best_selection,which.max(best_selection_summary$rsq))
```

```
[1] "Based on Cp, the best model is:"
```

Out[49]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

```
[1] "Based on BIC, the best model is:"
```

Out[49]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

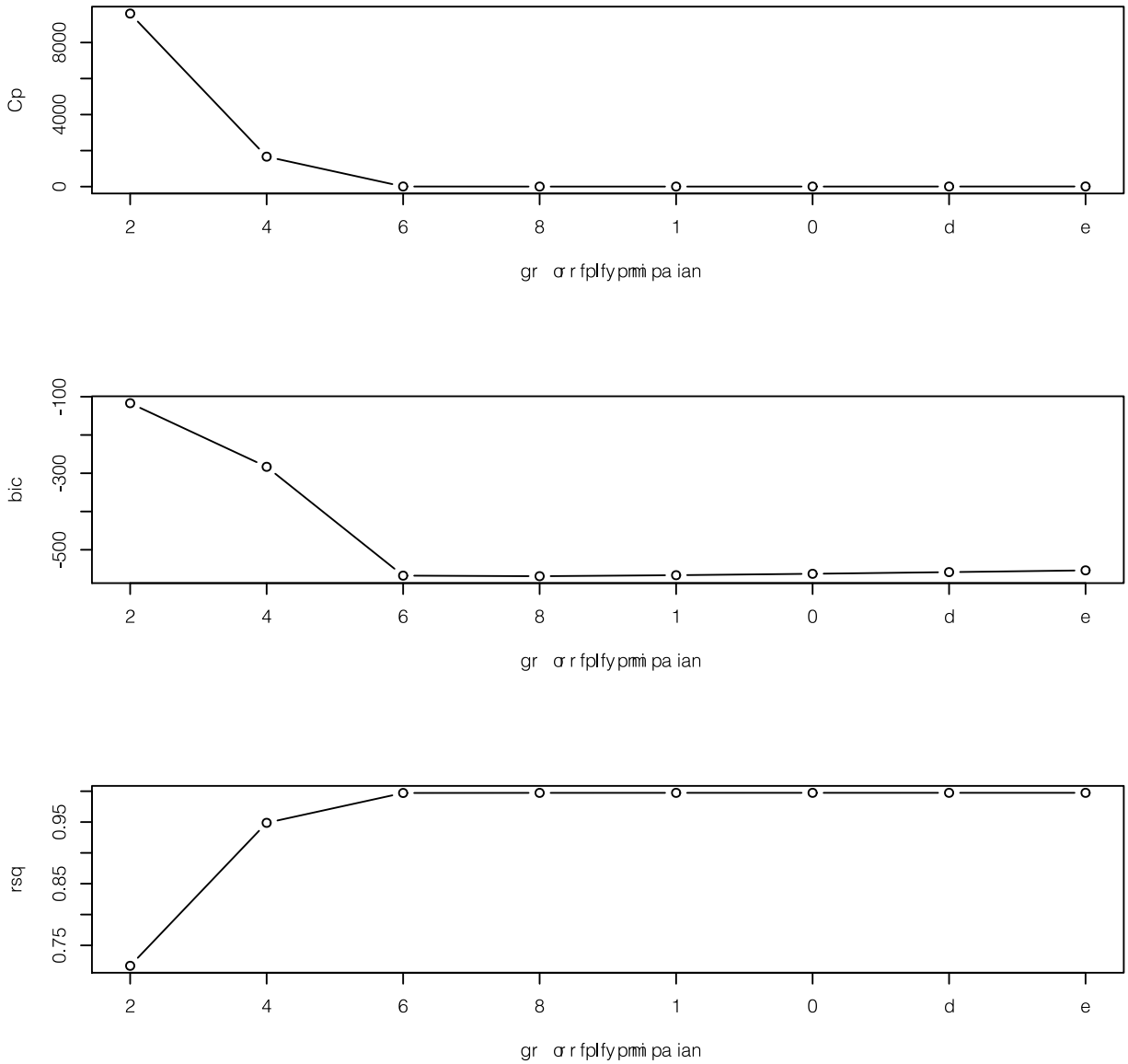
```
[1] "Based on R squared, the best model is:"
```

Out[49]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183
poly(X, 10)5	0.457699187806449
poly(X, 10)6	1.28566571981932
poly(X, 10)7	-0.107867813523064
poly(X, 10)8	0.863027733541585
poly(X, 10)9	0.113331849816744
poly(X, 10)10	0.149897834460875

Part D

In [50]:
```
forward_selection = regsubsets(Y~poly(X, 10), data = XY, method ="forward")
forward_selection_summary = summary(forward_selection)
par(mfrow = c(3,1))
plot(forward_selection_summary$cp,type = "b", xlab = "degree of polynomial" ,
ylab = "Cp")
plot(forward_selection_summary$bic,type = "b", xlab = "degree of polynomial" ,
ylab = "bic")
plot(forward_selection_summary$rsq,type = "b", xlab = "degree of polynomial" ,
ylab = "rsq")
```



In [51]:
```
print("Based on Cp, the best model is:")
coef(forward_selection,which.min(forward_selection_summary$cp))
print("Based on BIC, the best model is:")
coef(forward_selection,which.min(forward_selection_summary$bic))
print("Based on R squared, the best model is:")
coef(forward_selection,which.max(forward_selection_summary$rsq))
```

```
[1] "Based on Cp, the best model is:"
```

Out[51]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

```
[1] "Based on BIC, the best model is:"
```

Out[51]:

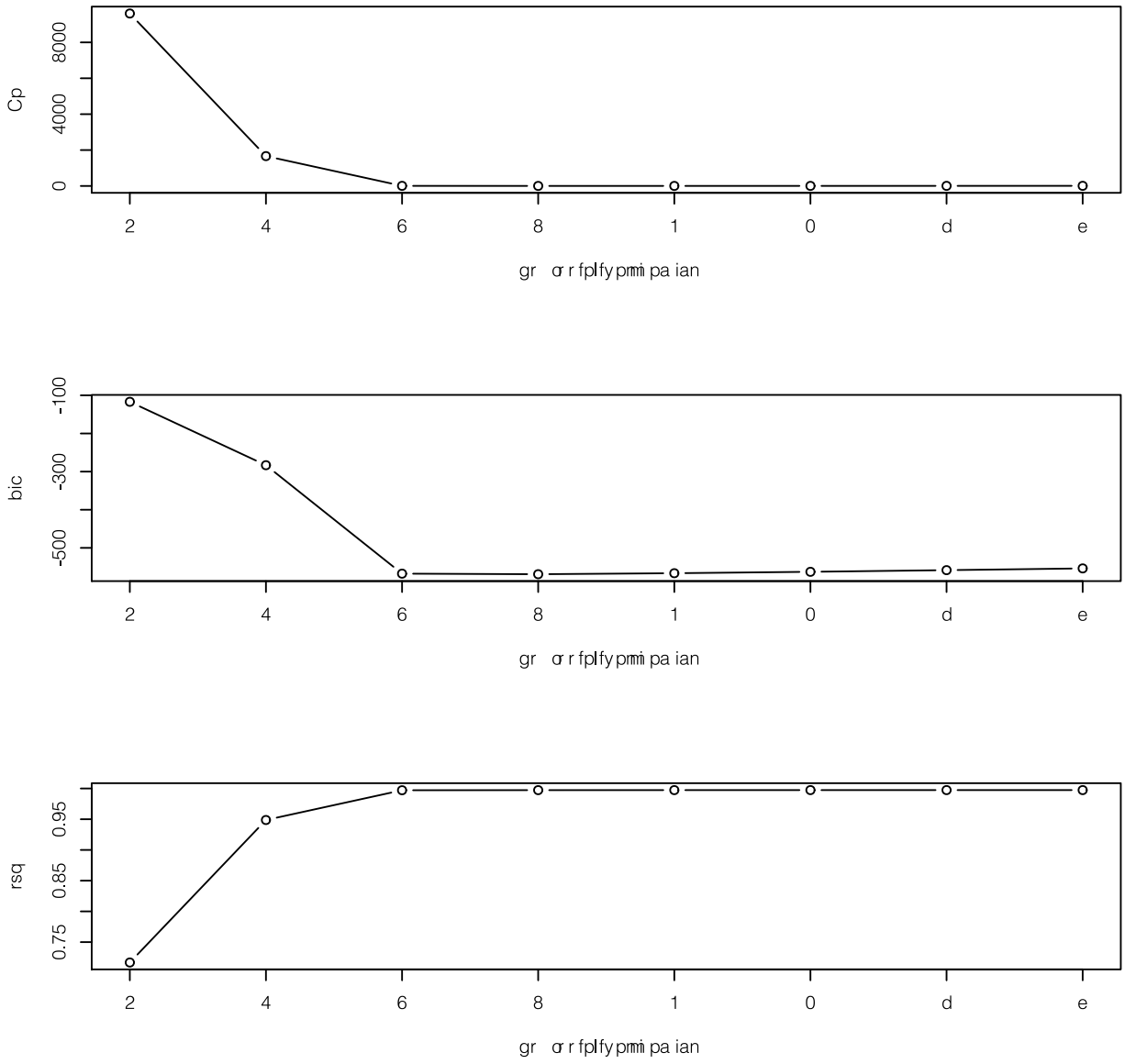
(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

```
[1] "Based on R squared, the best model is:"
```

Out[51]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183
poly(X, 10)5	0.457699187806449
poly(X, 10)6	1.28566571981932
poly(X, 10)8	0.863027733541585
poly(X, 10)10	0.149897834460875

In [52]:
```
backward_selection = regsubsets(Y~poly(X, 10), data = XY, method ="backward")
backward_selection_summary = summary(backward_selection)
par(mfrow = c(3,1))
plot(backward_selection_summary$cp,type = "b", xlab = "degree of polynomial" ,
ylab = "Cp")
plot(backward_selection_summary$bic,type = "b", xlab = "degree of polynomial"
, ylab = "bic")
plot(backward_selection_summary$rsq,type = "b", xlab = "degree of polynomial"
, ylab = "rsq")
```



In [53]:
```
print("Based on Cp, the best model is:")
coef(backward_selection,which.min(backward_selection_summary$cp))
print("Based on BIC, the best model is:")

coef(backward_selection,which.min(backward_selection_summary$bic))
print("Based on R squared, the best model is:")

coef(backward_selection,which.max(backward_selection_summary$rsq))
```

```
[1] "Based on Cp, the best model is:"
```

Out[53]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

```
[1] "Based on BIC, the best model is:"
```

Out[53]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183

```
[1] "Based on R squared, the best model is:"
```

Out[53]:

(Intercept)	4.56470622794346
poly(X, 10)1	149.309802114251
poly(X, 10)2	38.8087847420602
poly(X, 10)3	84.9219405421287
poly(X, 10)4	-2.28234848572183
poly(X, 10)5	0.457699187806449
poly(X, 10)6	1.28566571981932
poly(X, 10)8	0.863027733541585
poly(X, 10)10	0.149897834460875

All the models based on R^2, Cp, and BIC seem to conclude the same model types. Both Cp and BIC pick models with a polynomial degree of 4 whereas R^2 picks a 10th degree polynomial model.