Homework 3

Matthew, Ryan, Dani

2017-11-01

Question 1 (Chapter 6, #8, parts (a)-(e), 10 marks)

a. (1 mark) (Note: You should set your random seed, for reproducibility.)

```
set.seed(1234)
n = 100
X = rnorm(n)
error = rnorm(n)
```

b. (1 mark)

We selected Beta values by the following rule:

$$\beta_i = i + 1$$

```
Y = 1 + 2*X + 3*X^2 +4*X^3 + error
```

c. (3 marks) For the "best model obtained", you should use one that is parsimonious and close to the consensus best according tht the three selection criteria.

You don't **have** to create a data frame. `regsubsets()` can take a design matrix and response vector, just like `lm.fit()` and `glmnet()`. If you do decide to create a data frame, the following hint may be of use:

```
pmax <- 10
Xmat <- matrix(NA,nrow=n,ncol=pmax)
for(i in 1:pmax) {
  Xmat[,i] <- X^i
}
colnames(Xmat) <- paste0("X.",1:pmax)
dat <- data.frame(Y,Xmat)
```

Exhaustive Method

```
models = regsubsets(Y ~ ., data = dat, nvmax = 10)
models.sum = summary(models)

cp.best = which.min(models.sum$cp)
bic.best = which.min(models.sum$bic)
rsq.best = which.max(models.sum$adjr2)

print(paste("Best model for Cp is model with", cp.best, "variables"))
```

```
## [1] "Best model for Cp is model with 3 variables"
```

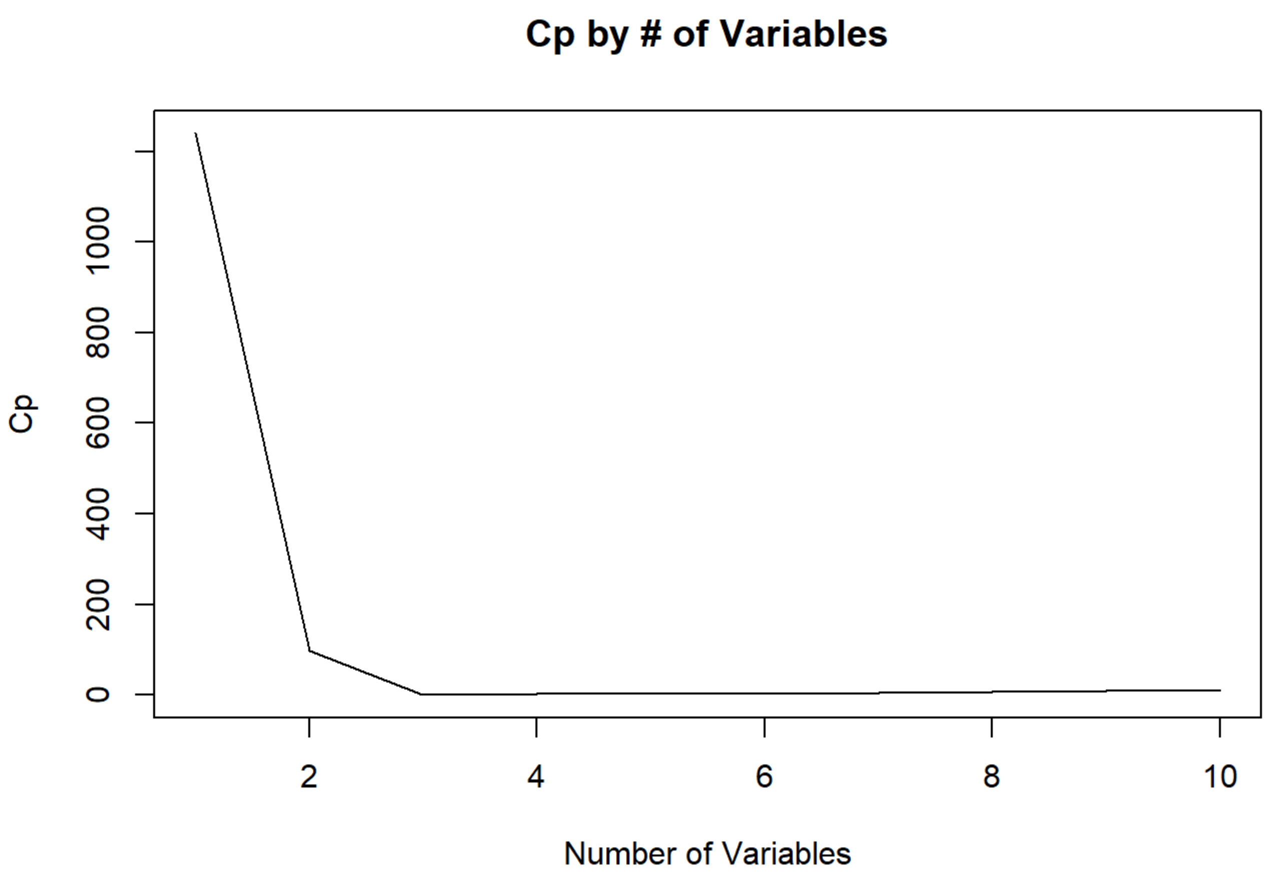
```
print(paste("Best model for BIC is model with", bic.best, "variables"))
```

```
## [1] "Best model for BIC is model with 3 variables"
```

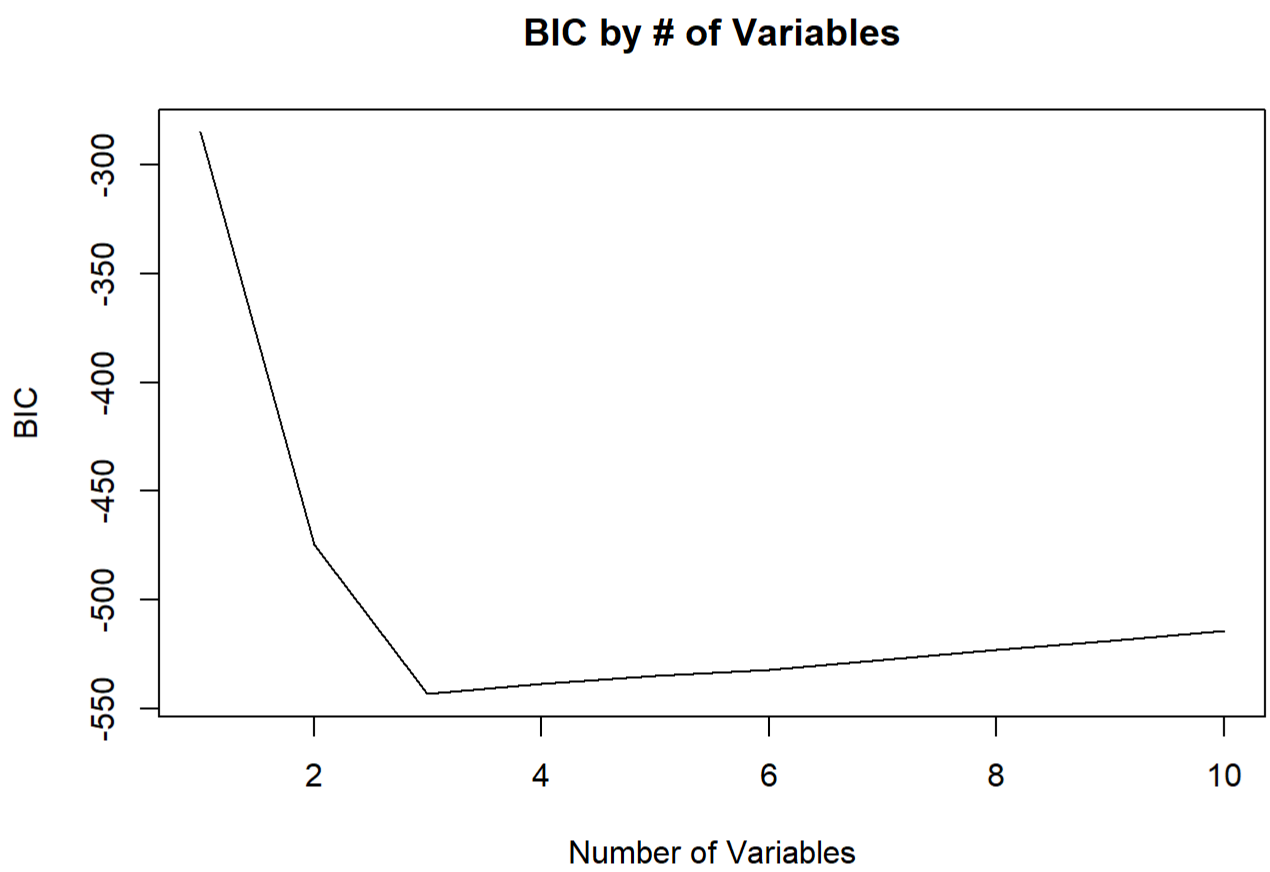
```
print(paste("Best model for Adjusted R^2 is model with", rsq.best, "variables"))
```

```
## [1] "Best model for Adjusted R^2 is model with 3 variables"
```

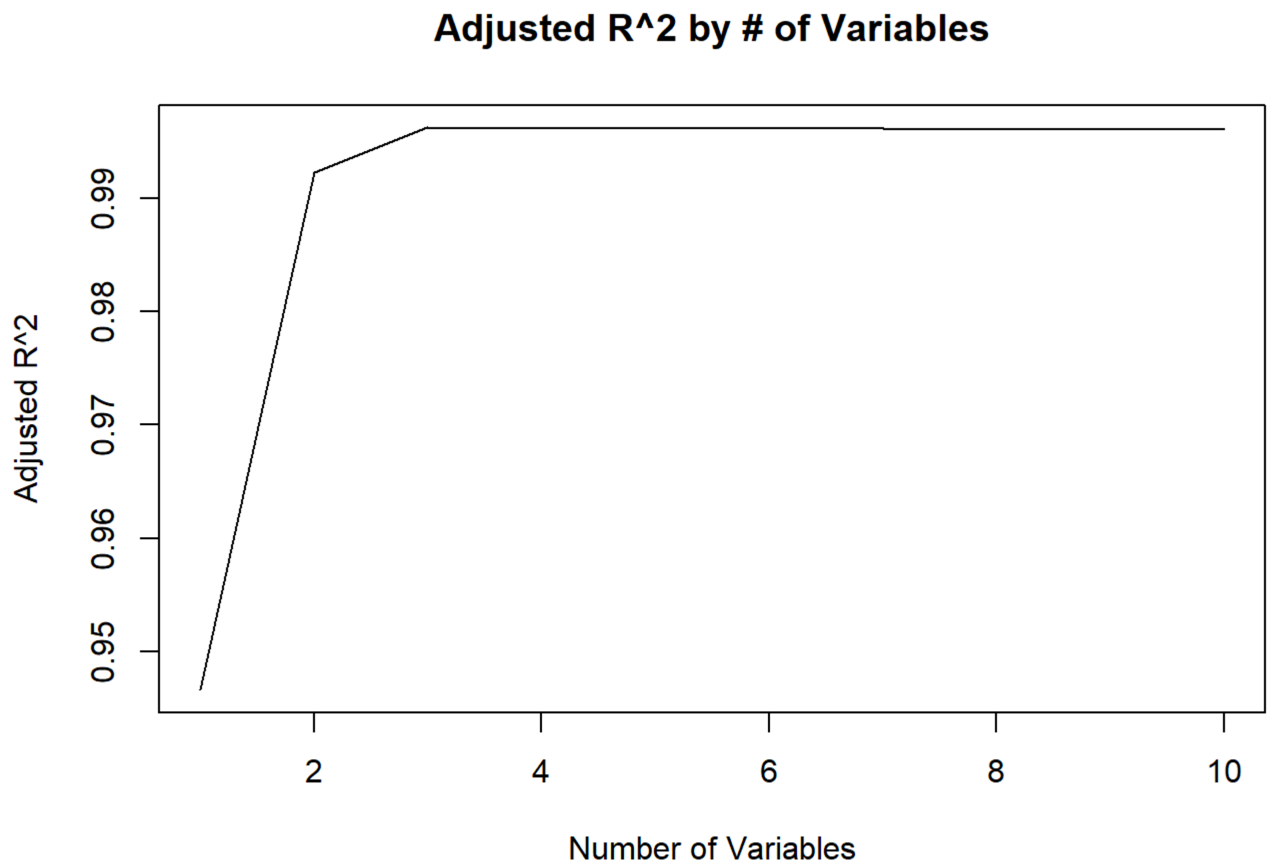
```
plot(models.sum$cp, xlab = "Number of Variables", ylab = "Cp", main = "Cp by # of Variables", ty
pe="l")
```




```
plot(models.sum$bic, xlab = "Number of Variables", ylab = "BIC", main = "BIC by # of Variables",
 type="l")
```




```
plot(models.sum$adjr2, xlab = "Number of Variables", ylab = "Adjusted R^2", main = "Adjusted R^2
 by # of Variables", type="l")
```




```
if(cp.best == bic.best & bic.best == rsq.best){
  print("Best model for all variables has coeffiecients")
  coef(models, cp.best)
} else{
  print("Best model for Cp has coeffiecients")
  coef(models, cp.best)

  print("Best model for BIC  has coeffiecients")
  coef(models, bic.best)

  print("Best model for Adjusted R^2  has coeffiecients")
  coef(models, rsq.best)
}
```

```
## [1] "Best model for all variables has coeffiecients"
```

```
## (Intercept)         X.1         X.2         X.3
##    1.132470    1.912586    2.893627    4.032305
```

d. (2 marks)

Forward Selection

```
models = regsubsets(Y ~ ., method = "forward", data = dat, nvmax = 10)
models.sum = summary(models)

cp.best = which.min(models.sum$cp)
bic.best = which.min(models.sum$bic)
rsq.best = which.max(models.sum$adjr2)

print(paste("Best model for Cp is model with", cp.best, "variables"))
```

```
## [1] "Best model for Cp is model with 3 variables"
```

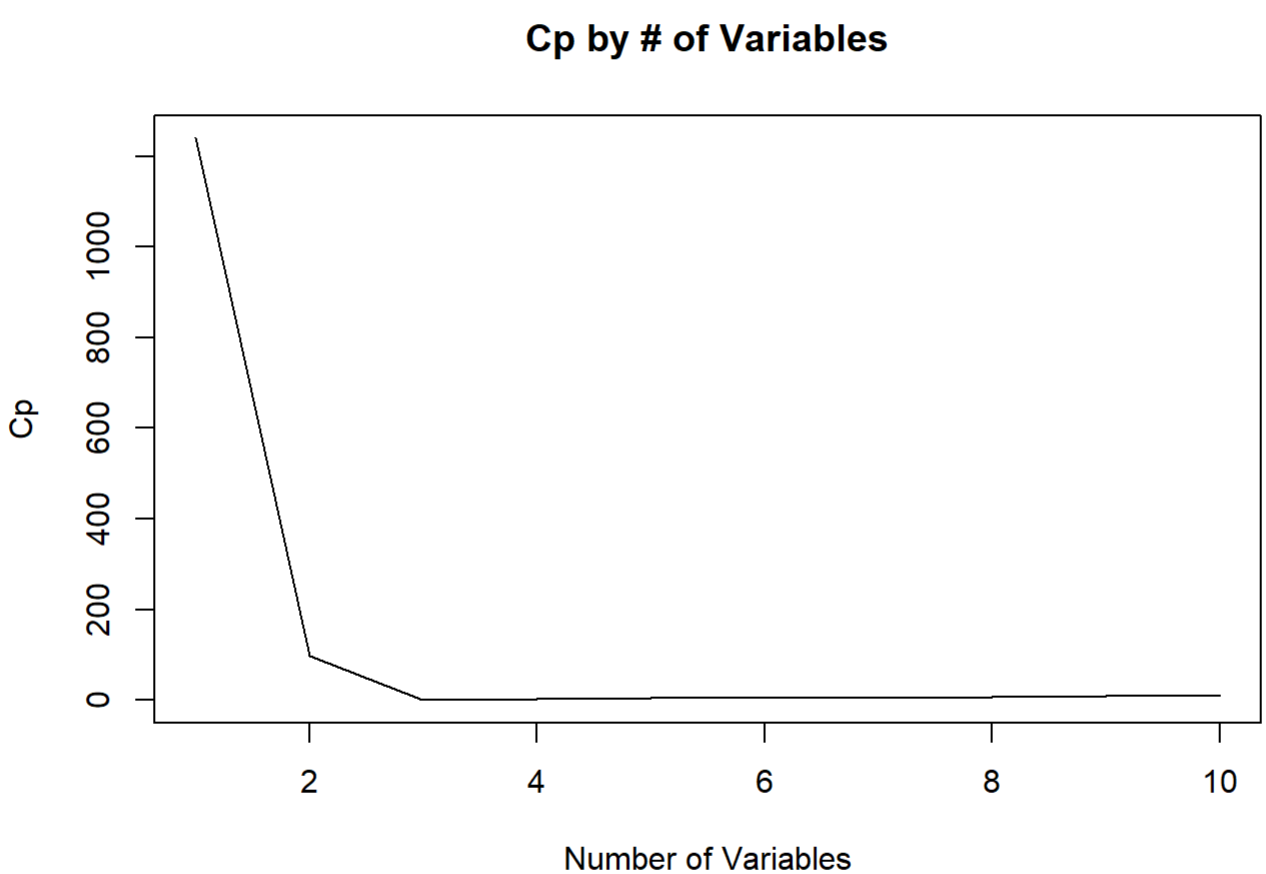
```
print(paste("Best model for BIC is model with", bic.best, "variables"))
```

```
## [1] "Best model for BIC is model with 3 variables"
```

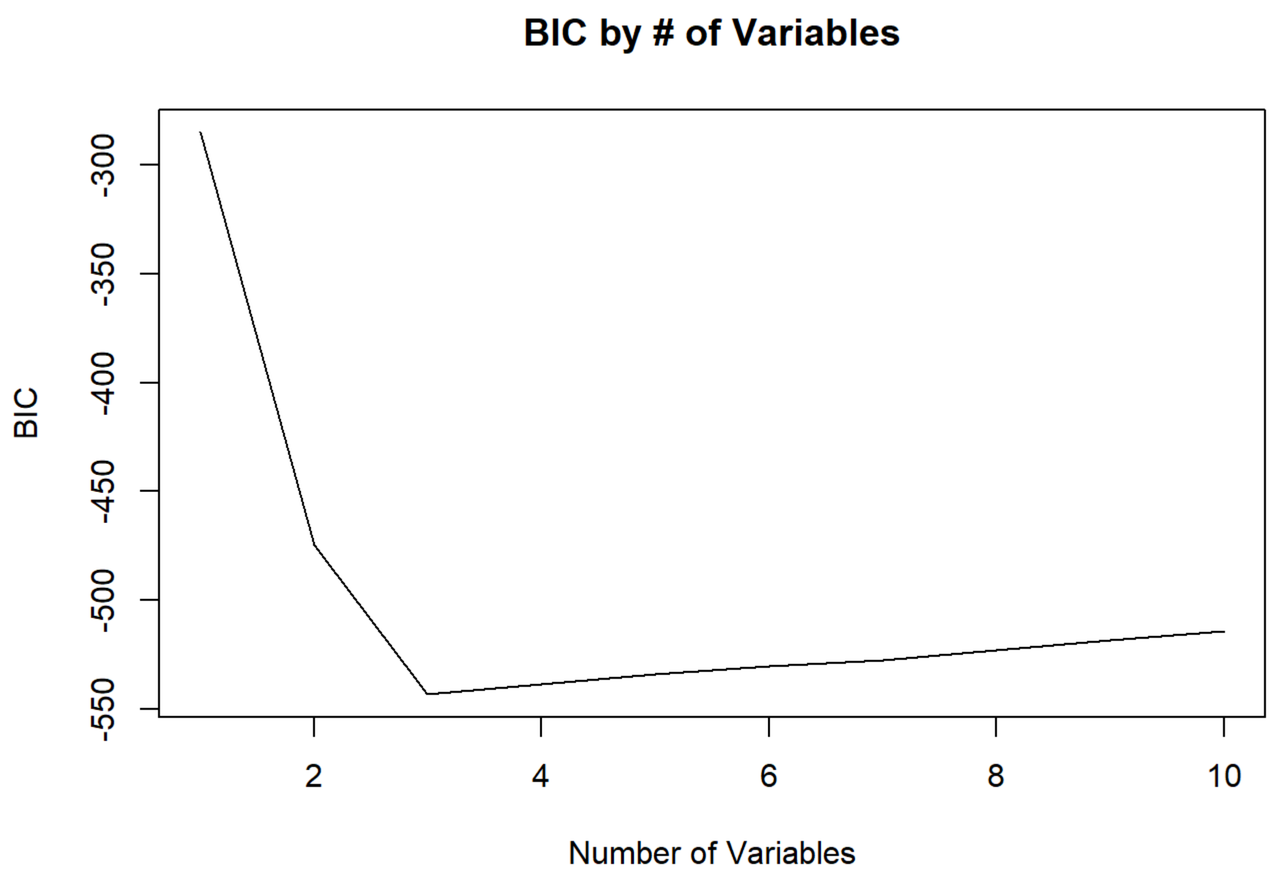
```
print(paste("Best model for Adjusted R^2 is model with", rsq.best, "variables"))
```

```
## [1] "Best model for Adjusted R^2 is model with 3 variables"
```

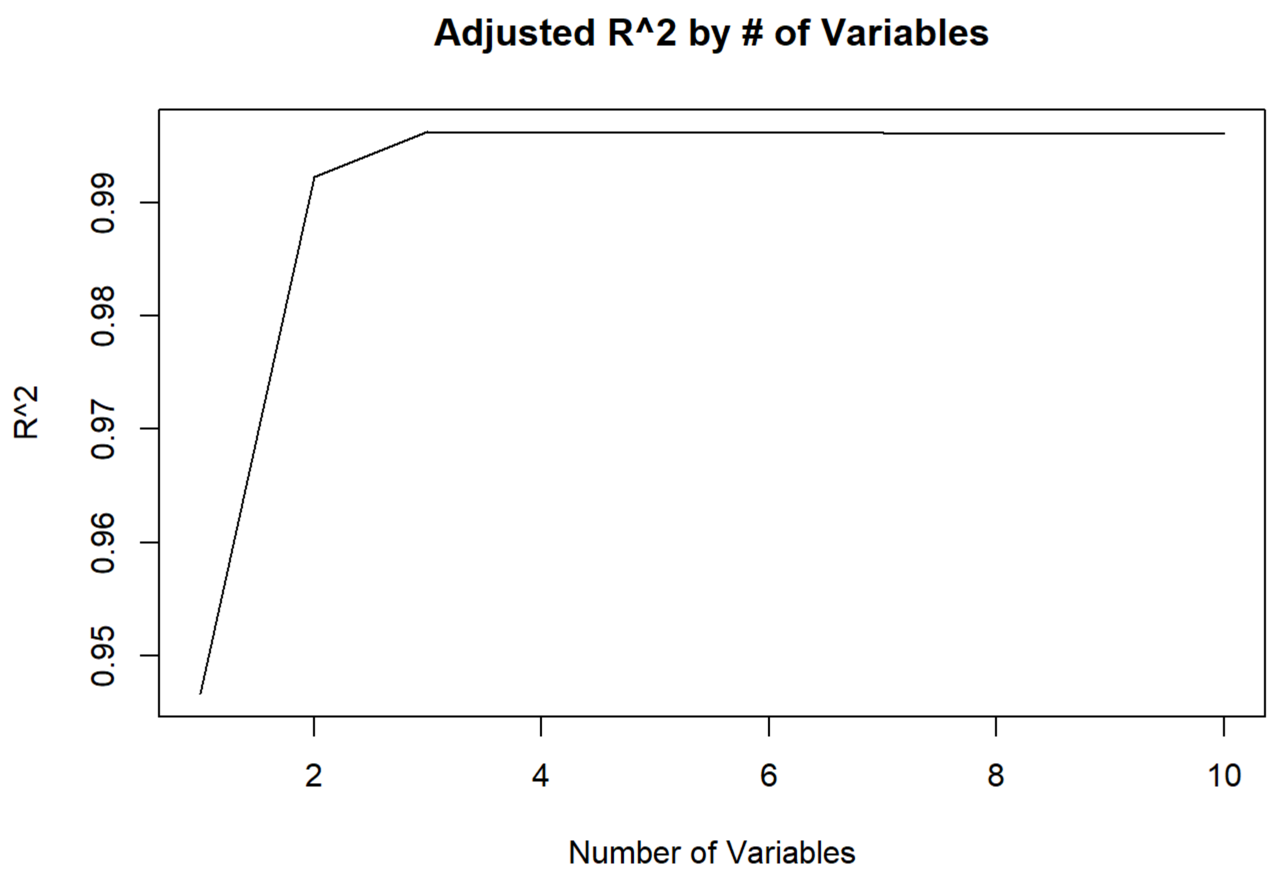
```
plot(models.sum$cp, xlab = "Number of Variables", ylab = "Cp", main = "Cp by # of Variables", ty
pe="l")
```



```
plot(models.sum$bic, xlab = "Number of Variables", ylab = "BIC", main = "BIC by # of Variables",
 type="l")
```




```
plot(models.sum$adjr2, xlab = "Number of Variables", ylab = "R^2", main = "Adjusted R^2 by # of
 Variables", type="l")
```




```
if(cp.best == bic.best & bic.best == rsq.best){
  print("Best model for all variables has coeffiecients")
  coef(models, cp.best)
} else{
  print("Best model for Cp has coeffiecients")
  coef(models, cp.best)

  print("Best model for BIC  has coeffiecients")
  coef(models, bic.best)

  print("Best model for Adjusted R^2  has coeffiecients")
  coef(models, rsq.best)
}
```

```
## [1] "Best model for all variables has coeffiecients"
```

```
## (Intercept)         X.1         X.2         X.3
##    1.132470    1.912586    2.893627    4.032305
```

Backward Selection

```
models = regsubsets(Y ~ ., method = "backward", data = dat, nvmax = 10)
models.sum = summary(models)

cp.best = which.min(models.sum$cp)
bic.best = which.min(models.sum$bic)
rsq.best = which.max(models.sum$adjr2)

print(paste("Best model for Cp is model with", cp.best, "variables"))
```

```
## [1] "Best model for Cp is model with 3 variables"
```

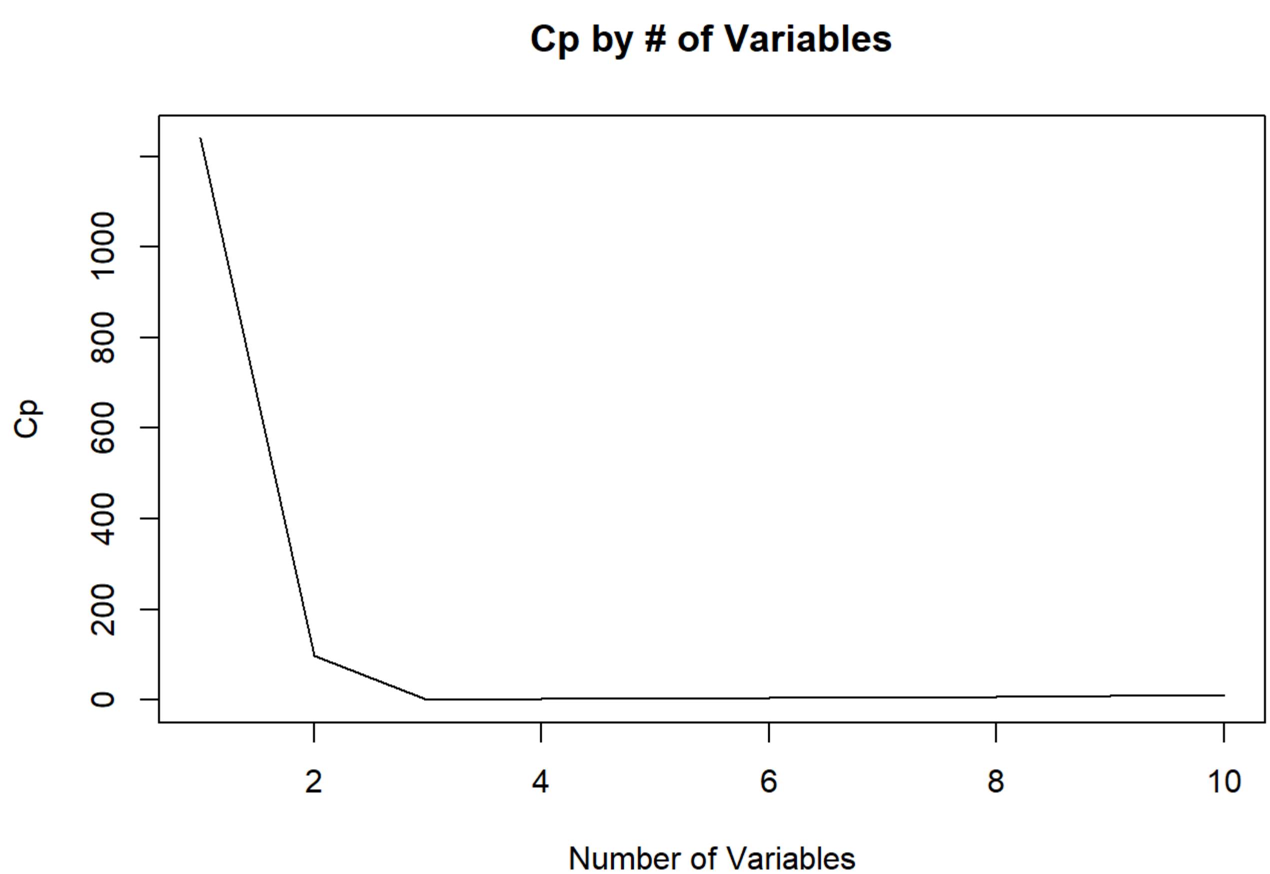
```
print(paste("Best model for BIC is model with", bic.best, "variables"))
```

```
## [1] "Best model for BIC is model with 3 variables"
```

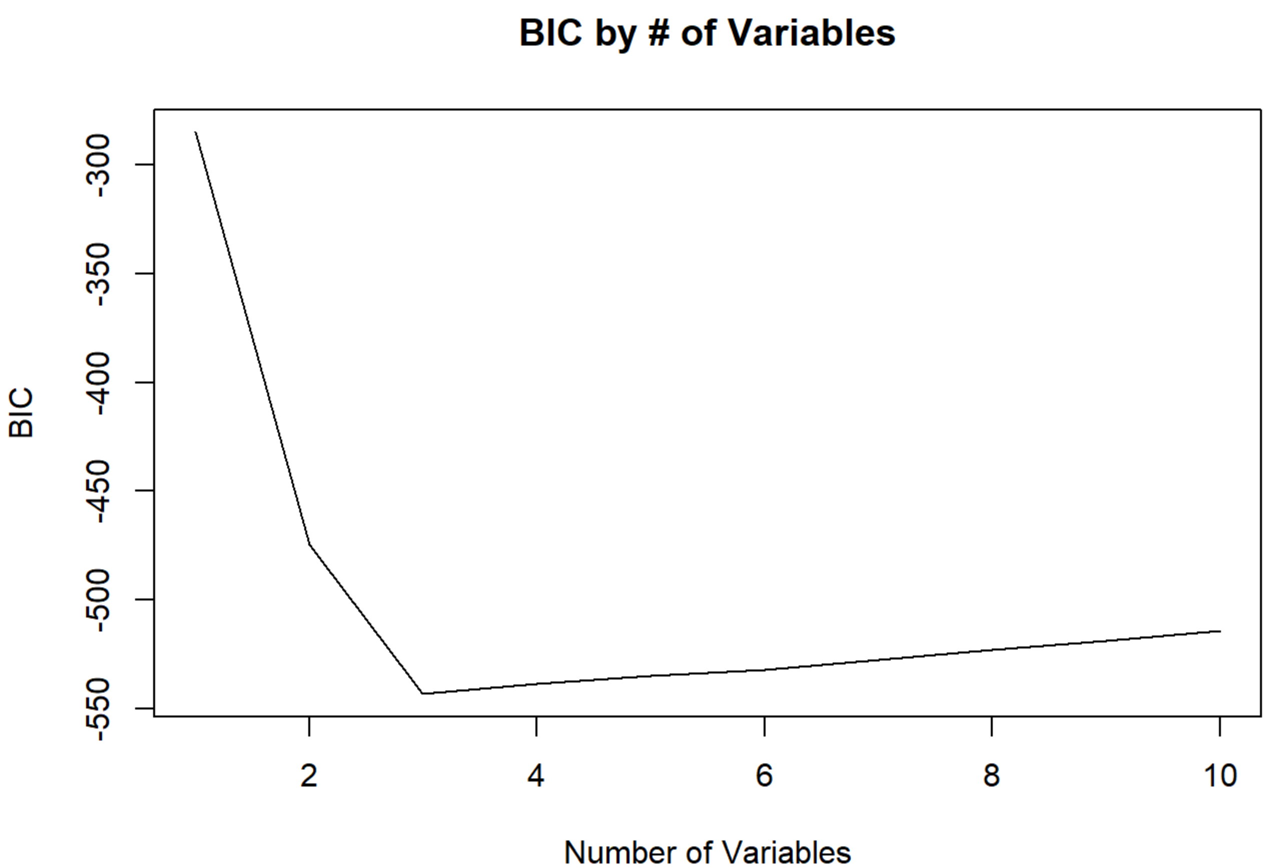
```
print(paste("Best model for Adjusted R^2 is model with", rsq.best, "variables"))
```

```
## [1] "Best model for Adjusted R^2 is model with 3 variables"
```

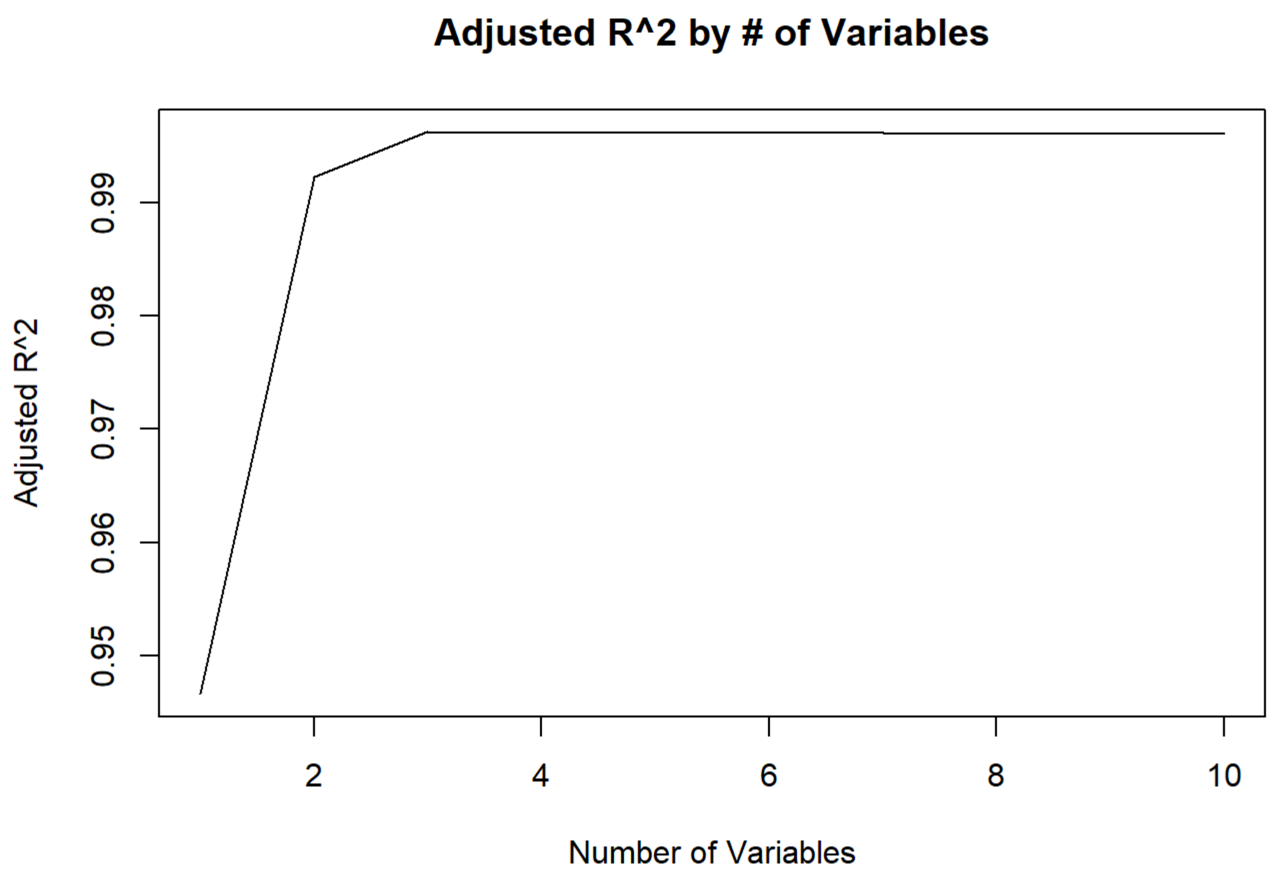
```
plot(models.sum$cp, xlab = "Number of Variables", ylab = "Cp", main = "Cp by # of Variables", ty
pe="l")
```



```
plot(models.sum$bic, xlab = "Number of Variables", ylab = "BIC", main = "BIC by # of Variables",
 type="l")
```




```
plot(models.sum$adjr2, xlab = "Number of Variables", ylab = "Adjusted R^2", main = "Adjusted R^2
 by # of Variables", type="l")
```




```
if(cp.best == bic.best & bic.best == rsq.best){
  print("Best model for all variables has coeffiecients")
  coef(models, cp.best)
} else{
  print("Best model for Cp has coeffiecients")
  coef(models, cp.best)

  print("Best model for BIC  has coeffiecients")
  coef(models, bic.best)

  print("Best model for Adjusted R^2  has coeffiecients")
  coef(models, rsq.best)
}
```

```
## [1] "Best model for all variables has coeffiecients"
```

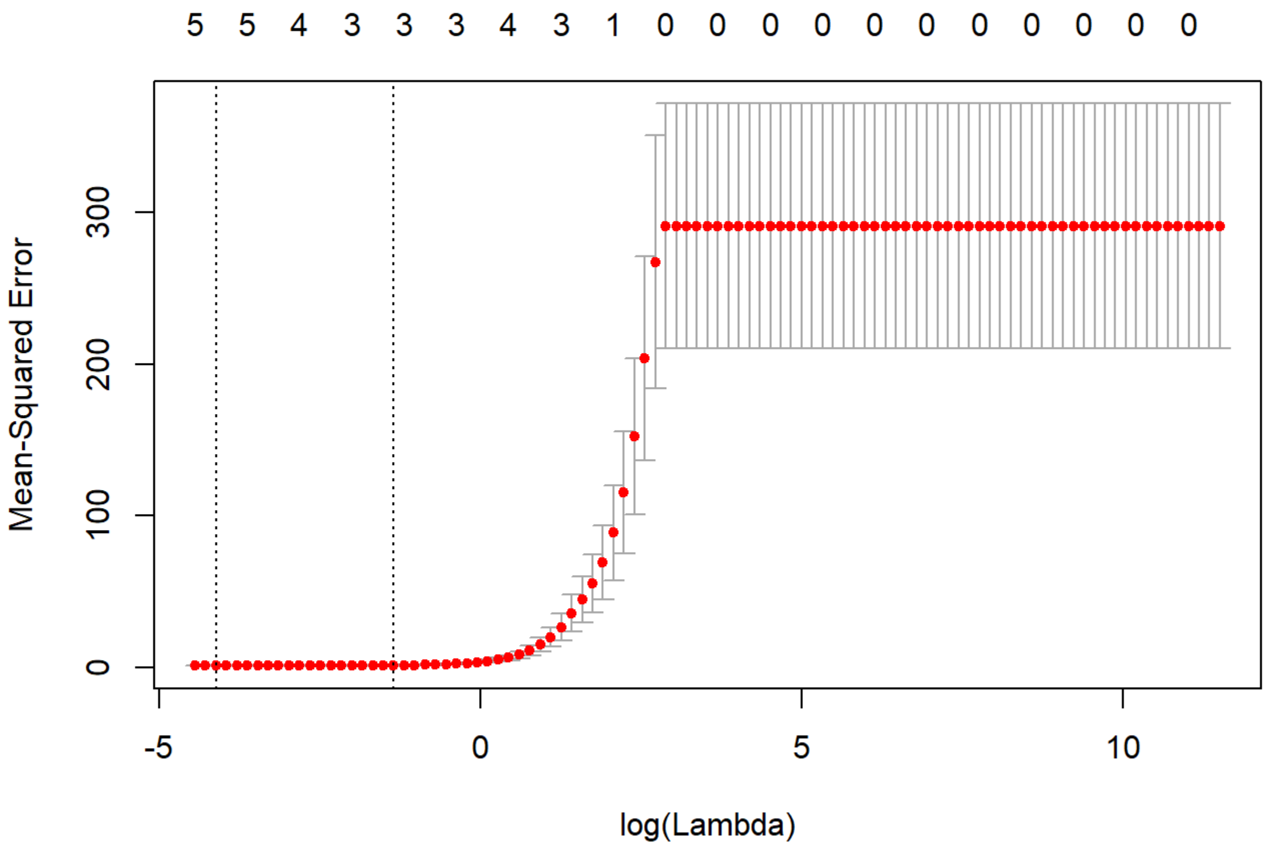
```
## (Intercept)         X.1         X.2         X.3 
##    1.132470    1.912586    2.893627    4.032305
```

Compare to results in 8c

The results are excatly the same for all 3 selection criteria. The same model was selected by all criteria in all directions.

e. (3 marks)

```
lambdas <- 10^{seq(from=-2,to=5,length=100)}
cv.lafit <- cv.glmnet(Xmat,Y,alpha=1,lambda=lambdas)
plot(cv.lafit)
```




```
la.best.lam <- cv.lafit$lambda.1se
la.best.lam
```

```
## [1] 0.2595024
```

```
la.best <- glmnet(Xmat,Y,alpha=1,lambda=la.best.lam)
coef(la.best)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                   s0
## (Intercept) 1.268441
## X.1         1.754901
## X.2         2.738949
## X.3         4.012495
## X.4         .
## X.5         .
## X.6         .
## X.7         .
## X.8         .
## X.9         .
## X.10        .
```

Discuss results obtained from CV and lasso model

The results are almost identical to the model selected above. The coefficients of the predictors in the lasso model have seen a very slight shrinkage the intercept has seen a small increase.

Question 2 (Ch6, #9, 12 marks)

a. (0 marks) To make everyone's results comparable, please select your test set with the following.

```
data(College)
# Standardize columns
College <- mutate(College,Private = as.numeric(Private=="Yes"))
College <- data.frame(lapply(College,scale))
dim(College) # 777 rows, use 111 as test
```

```
## [1] 777  18
```

```
set.seed(1)
testset <- sample(1:777,size=111)
College.test <- College[testset,]
College.train <- College[-testset,]
```

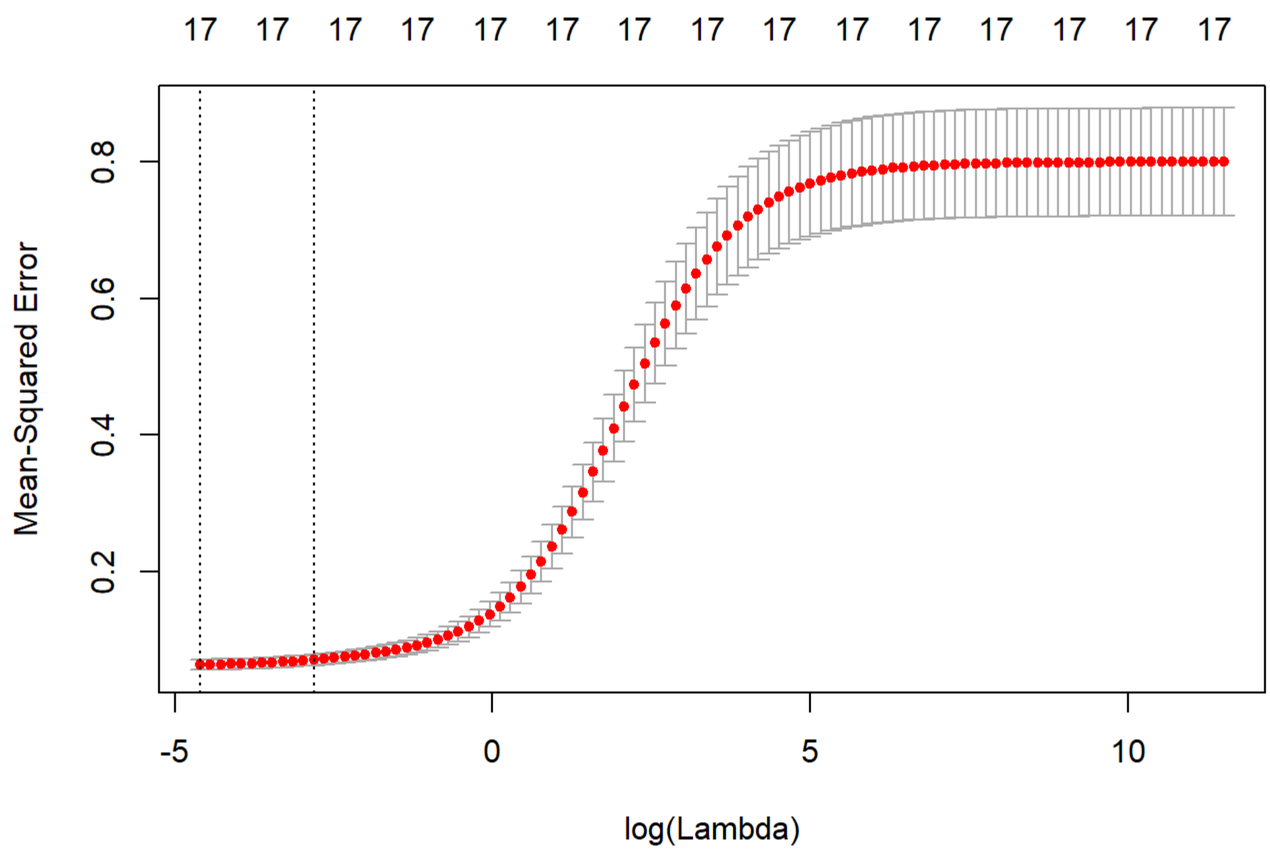
b. (2 marks)

```
linear = lm(Apps ~ . , data = College.train)

preds = predict(linear, College.test)
e1 = mean((College.test$Apps - preds)^2)
e1
```

```
## [1] 0.1866961
```

c. (2 marks)

```
lambdas <- 10^{seq(from=-2,to=5,length=100)}
X = College.train %>% select(-Apps) %>% data.matrix()
y = College.train$Apps
cv.lafit <- cv.glmnet(X, y, alpha=0, lambda=lambdas)
plot(cv.lafit)
```

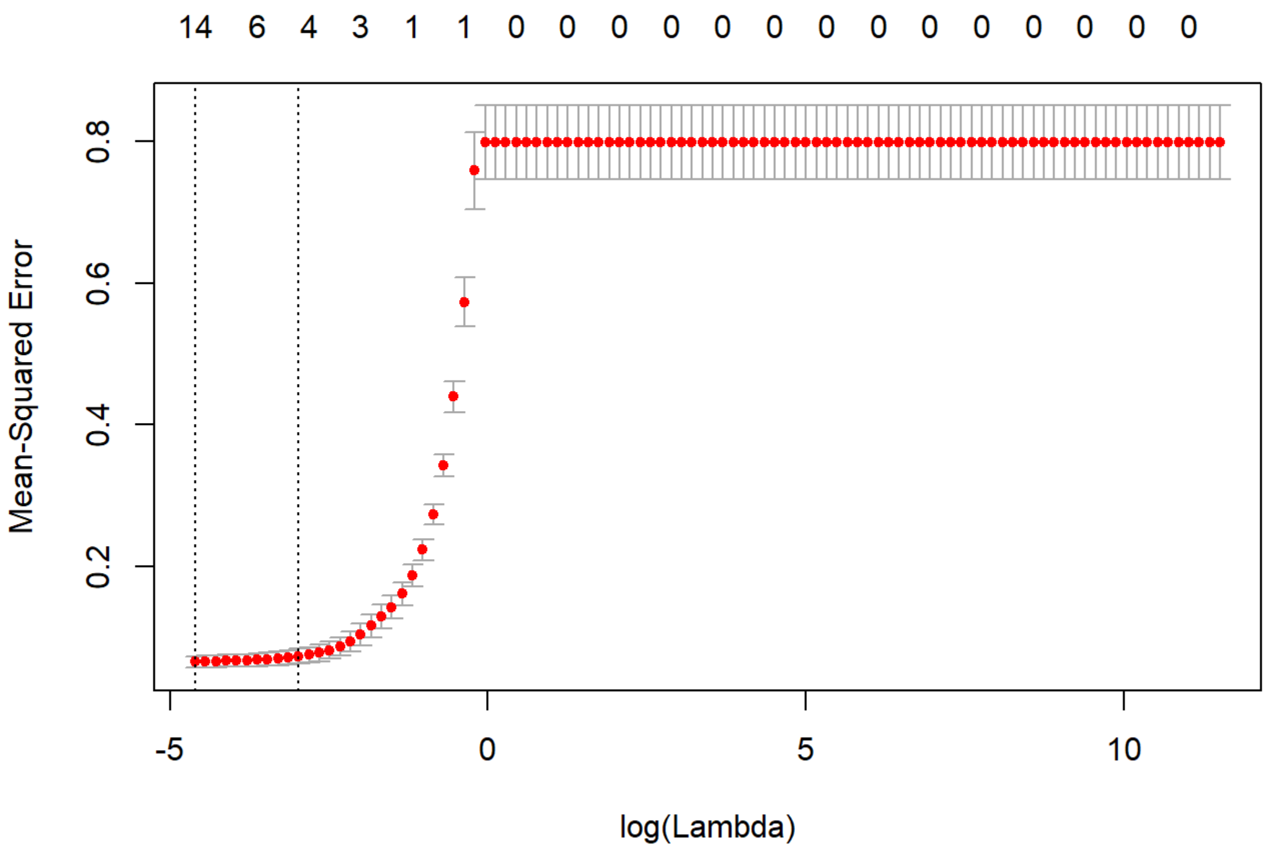



```
la.best.lam <- cv.lafit$lambda.1se
la.best <- glmnet(X, y, alpha=0, lambda = la.best.lam)

x.pred = College.test %>% select(-Apps) %>% data.matrix()
preds = predict(la.best, x.pred)
e2 = mean((College.test$Apps - preds)^2)
e2
```

```
## [1] 0.3363783
```

d. (2 marks)

```
lambdas <- 10^{seq(from=-2,to=5,length=100)}
X = College.train %>% select(-Apps) %>% data.matrix()
y = College.train$Apps
cv.lafit <- cv.glmnet(X, y, alpha = 1, lambda = lambdas)
plot(cv.lafit)
```




```
la.best.lam <- cv.lafit$lambda.1se
la.best <- glmnet(X, y, alpha = 1, lambda = la.best.lam)

x.pred = College.test %>% select(-Apps) %>% data.matrix()
preds = predict(la.best, x.pred)
e3 = mean((College.test$Apps - preds)^2)

print(paste("number of coefs larger than 0:", sum(coef(la.best) > 0)))
```

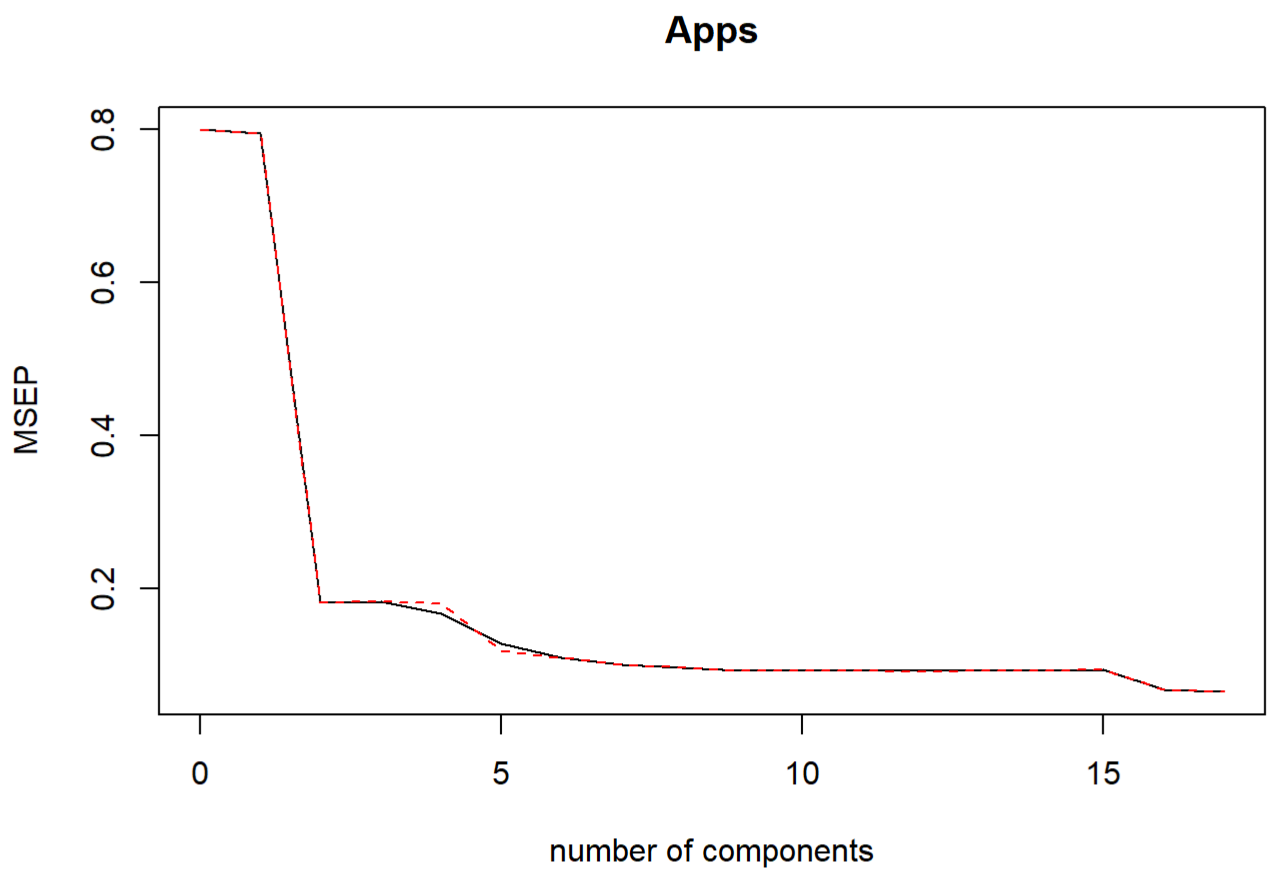
```
## [1] "number of coefs larger than 0: 4"
```

```
e3
```

```
## [1] 0.2490143
```

e. (2 marks)

```
# data already scaled above
pcr.fit = pcr(Apps ~ ., data = College.train, validation = "CV")

# Choose M by the graph
validationplot(pcr.fit, val.type = "MSEP")
```




```
print("We choose a value of 5 for M is a simple model that has a very good % variance
explained")
```

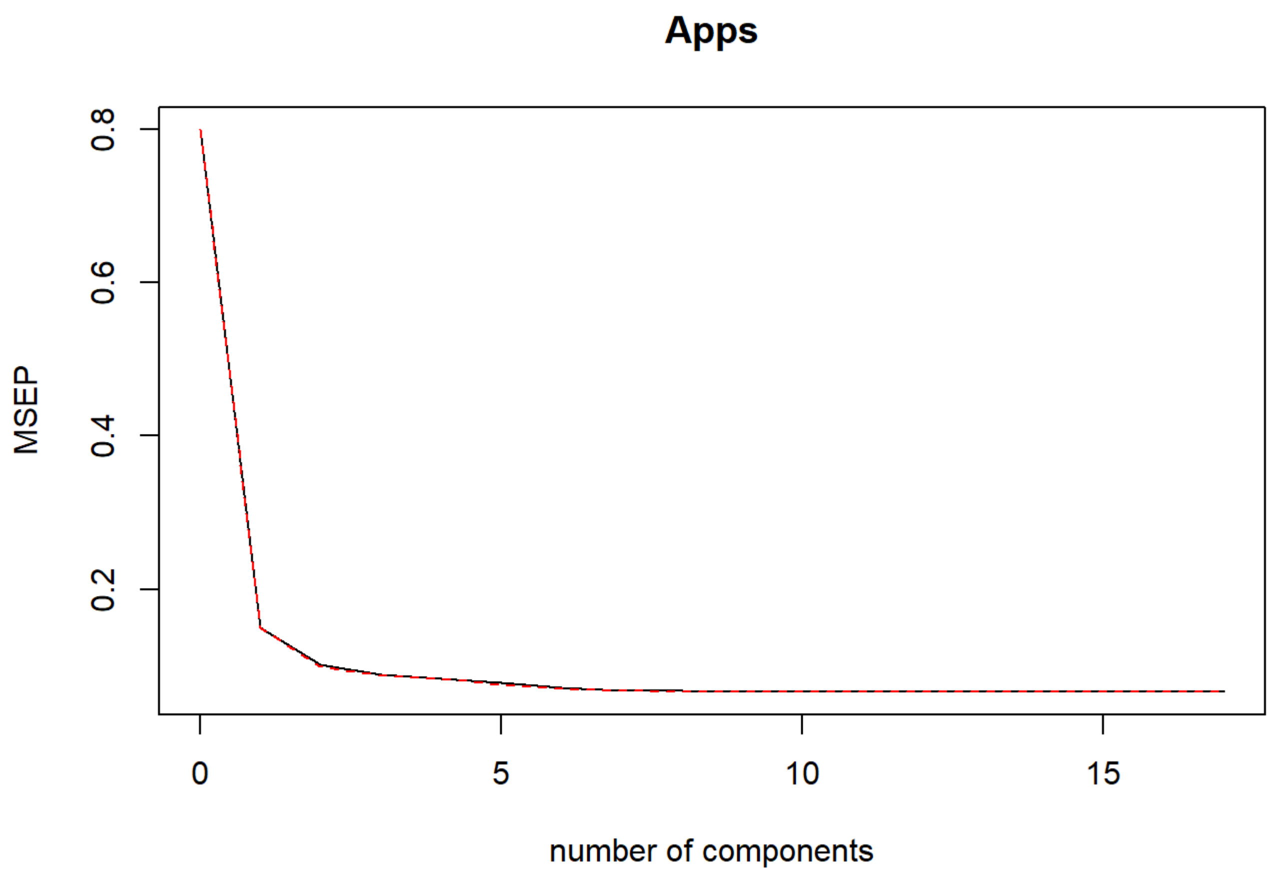
```
## [1] "We choose a value of 5 for M is a simple model that has a very good % variance explaine
d"
```

```
preds = predict(pcr.fit, College.test, ncomp = 5)
e4 = mean((preds - College.test$Apps)^2)
e4
```

```
## [1] 0.5938384
```

f. (2 marks)

```r
pls.fit = plsr(Apps ~ ., data = College.train, validation = "CV")
# Choose M by the graph and summary
validationplot(pls.fit, val.type = "MSEP")
```




```r
summary(pls.fit)
```

```
## Data:    X dimension: 666 17
##  Y dimension: 666 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.8943   0.3874   0.3181   0.2964   0.2897   0.2798   0.2669
## adjCV       0.8943   0.3867   0.3147   0.2959   0.2882   0.2752   0.2653
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.2611   0.2591   0.2585    0.2582    0.2579    0.2583    0.2579
## adjCV   0.2601   0.2583   0.2577    0.2573    0.2570    0.2574    0.2570
##        14 comps  15 comps  16 comps  17 comps
## CV       0.2580    0.2580     0.258    0.2580
## adjCV    0.2571    0.2571     0.257    0.2571
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X       25.12    36.86    62.35    65.59    67.94    73.39    76.29
## Apps    81.70    88.15    89.58    90.66    92.05    92.42    92.57
##       8 comps  9 comps  10 comps  11 comps  12 comps  13 comps  14 comps
## X       80.57    82.82     84.84     88.40     91.49     93.03     95.54
## Apps    92.60    92.63     92.67     92.69     92.69     92.70     92.70
##       15 comps  16 comps  17 comps
## X        97.18     98.96     100.0
## Apps     92.70     92.70      92.7
```

```
print("We choose a value of 2 for M as it is the simplest model that has a acceptable error rat
e")
```

```
## [1] "We choose a value of 2 for M as it is the simplest model that has a acceptable error rat
e"
```

```
preds = predict(pls.fit, College.test, ncomp = 2)
e5 = mean((preds - College.test$Apps)^2)
e5
```

```
## [1] 0.5186612
```

g. (2 marks)

How accurately can we predict the number of college applications received?

Is there much difference among the test errors resulting from these five approaches?

```
e1 #lm
```

```
## [1] 0.1866961
```

```
e3 # lasso
```

```
## [1] 0.2490143
```

```
e2 # ridge
```

```
## [1] 0.3363783
```

```
e5 # pls
```

```
## [1] 0.5186612
```

```
e4 # pcr
```

```
## [1] 0.5938384
```

We can predict the number of college applications very closely. The MSE is very small compared to the magnitude of the response variable. Our in sample Adjusted R-squared is 0.92.

The linear model has the smallest MSE for the testing data. It is significantly smaller than that of pcr. There is a spread in MSE across the 5 methods. This indicates that there is a distinct ranking between the 5 methods.

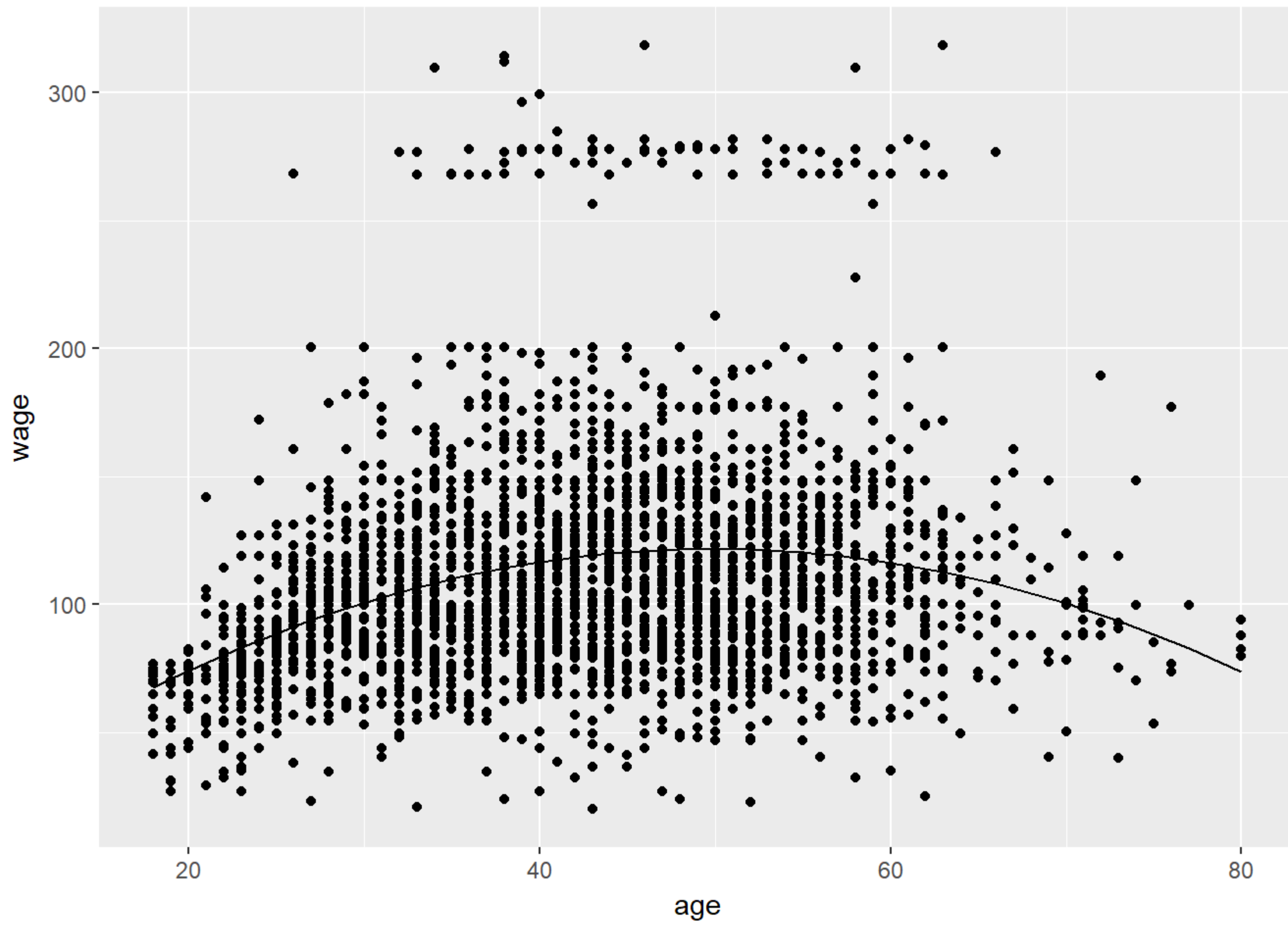
Question 3 (Ch7, #6, 8 marks)

a. (5 marks)

```r
attach(Wage)
set.seed(444)

# Make sure train set has full range of age values
Wage <- Wage %>% arrange(age)
Wage.train = Wage[c(1,nrow(Wage)), ]
Wage2 <- Wage[-c(1, nrow(Wage)), ]

# make test set and train set
testset <- sample(1:2998, size = 300)
Wage.test <- Wage2[testset,]
Wage.train <- rbind(Wage2[-testset,], Wage.train)

errors = rep(NA, 8)
for(i in 1:8){
  fit = lm(wage ~ poly(age, i), data = Wage.test)
  preds = predict(fit, Wage.train)
  errors[i] = mean((Wage.train$wage - preds)^2)
}

d = which.min(errors)


fit.1=lm(wage~age,data=Wage)
fit.2=lm(wage~poly(age, 2), data=Wage)
fit.3=lm(wage~poly(age, 3), data=Wage)
fit.4=lm(wage~poly(age, 4), data=Wage)
fit.5=lm(wage~poly(age, 5), data=Wage)
fit.6=lm(wage~poly(age, 6), data=Wage)
fit.7=lm(wage~poly(age, 7), data=Wage)
fit.8=lm(wage~poly(age, 8), data=Wage)
anova(fit.1, fit.2, fit.3, fit.4, fit.5, fit.6, fit.7, fit.8)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ age
## Model 2: wage ~ poly(age, 2)
## Model 3: wage ~ poly(age, 3)
## Model 4: wage ~ poly(age, 4)
## Model 5: wage ~ poly(age, 5)
## Model 6: wage ~ poly(age, 6)
## Model 7: wage ~ poly(age, 7)
## Model 8: wage ~ poly(age, 8)
##   Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1   2998 5022216
## 2   2997 4793430  1    228786 143.6484 < 2.2e-16 ***
## 3   2996 4777674  1     15756   9.8926  0.001676 **
## 4   2995 4771604  1      6070   3.8113  0.051002 .
## 5   2994 4770322  1      1283   0.8053  0.369590
## 6   2993 4766389  1      3932   2.4690  0.116221
## 7   2992 4763834  1      2555   1.6044  0.205381
## 8   2991 4763707  1       127   0.0795  0.777952
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

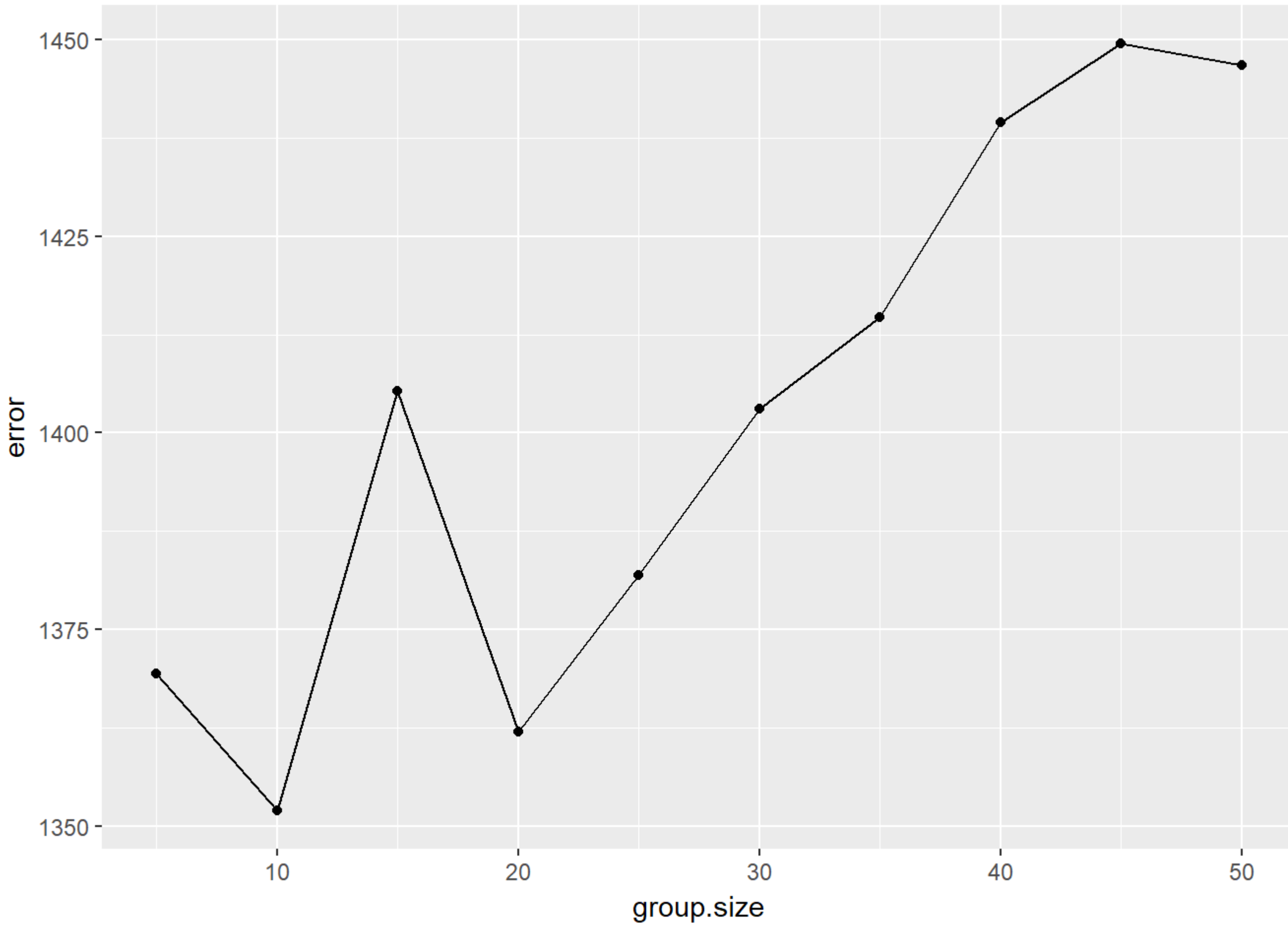
```
dat = data.frame(Wage$age, fitted(fit.2))


ggplot() +
  geom_point(data = Wage, aes(x = age, y = wage)) +
  geom_line(data = dat, aes(x = Wage.age, y = fitted.fit.2.))
```



The degree chose was a polynomial of degree 2. This is confirmed by our anova analysis.

b. (3 marks)

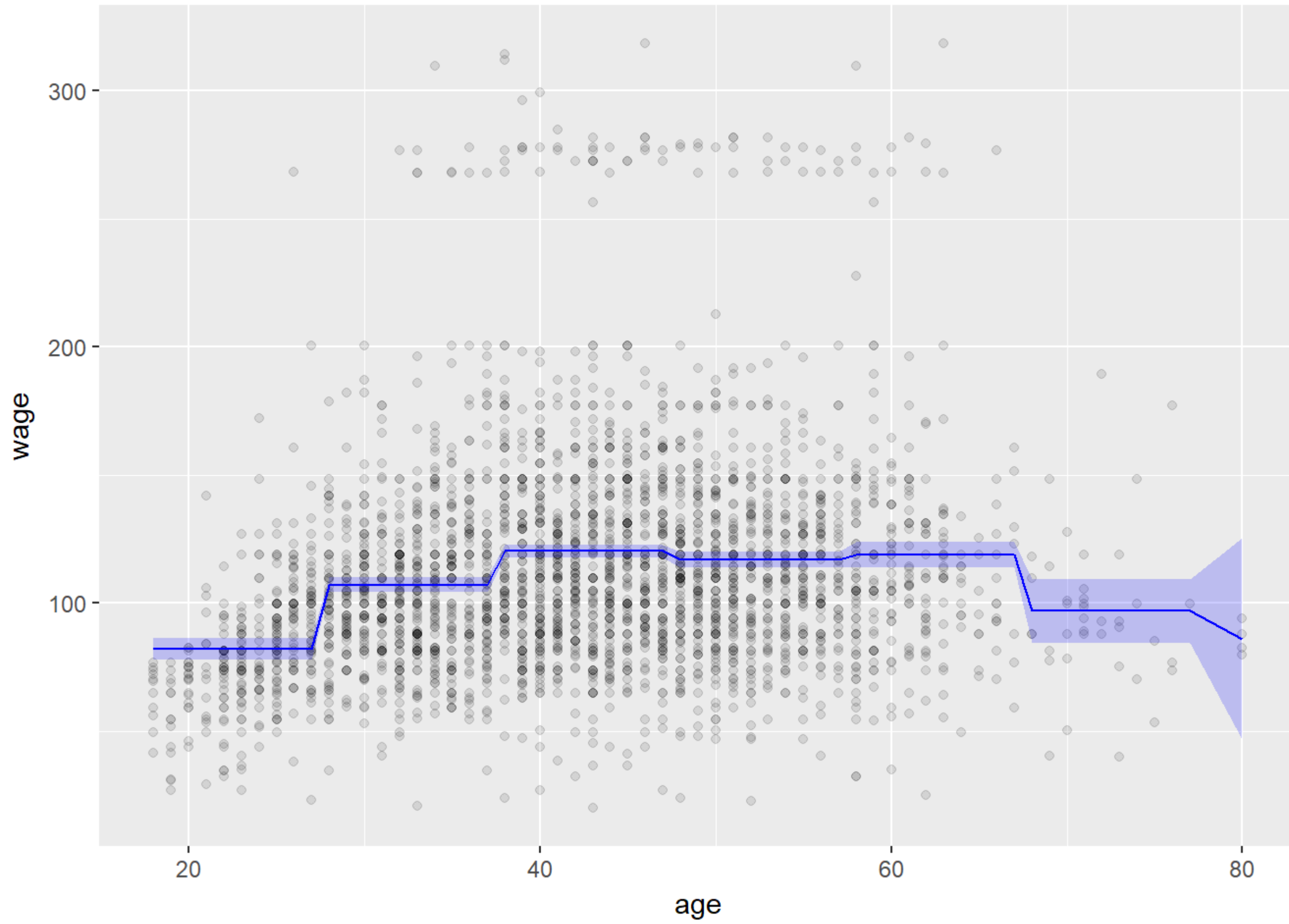
```
errors = rep(NA, 10)
for(i in 1:10){
  j = i*5
  agebreaks <- seq(17, 80+j, by = j)
  fit = lm(wage ~ cut(age, breaks = agebreaks), data = Wage.train)
  preds <-  predict(fit, Wage.test)
  errors[i] <-  mean((Wage.test$wage - preds)^2)
}


df.errors = data.frame(seq(5,50, by = 5), errors)
colnames(df.errors) = c("group.size","error")

ggplot(df.errors, aes(x = group.size, y =error))+
  geom_point()+
  geom_line()
```




```
j = which.min(errors)*5

# age groups should be of size 10
j
```

```
## [1] 10
```

```
agebreaks <- seq(17, 80+j, by = j)
fit = lm(wage ~ cut(age, breaks = agebreaks), data = Wage)
uniqueAge <- data.frame(age = sort(unique(Wage$age)))
preds <- data.frame(uniqueAge,predict(fit,newdata = uniqueAge,interval="confidence"))

ggplot(Wage,aes(x=age,y=wage)) +
  geom_point(alpha=0.1) +
  geom_ribbon(aes(x=age,y=fit,ymin=lwr,ymax=upr), data=preds, fill="blue", alpha=.2) +
  geom_line(aes(y = fit), data = preds, color="blue")
```



300
200
100
wage
20
40
60
80
age