HW6

Jingshi

3/9/2017

5.4 Question 5

a)

```
library(ISLR)
attach(Default)
set.seed(6)
model.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
summary(model.glm)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

b)

```
set.seed(6)
# i) Split the sample set into a training set and a validation set with 80% training and 20%
# validation.
train <- sample(dim(Default)[1], dim(Default)[1] * 0.8)

# ii) Fit a multiple logistic regression model using only the training observations.
model.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
#summary(model.glm)
```

```r
# iii) Obtain a prediction of default status for each individual in the validation set by
# computing the posterior probability of default for that individual, and classifying the
# individual to the default category if the posterior probability is greater than 0.5.
probabilities <- predict(model.glm, newdata = Default[-train, ], type = "response")
pred <- rep("No", length(probabilities))
pred[probabilities > 0.5] <- "Yes"

# iv) Compute the validation set error, which is the fraction of the observations in the
# validation set that are misclassified.
cat ("The test error rate is", mean(pred != Default[-train, ]$default))
```

```
## The test error rate is 0.027
```

c)

```r
set.seed(10)
# i) Split the sample set into a training set and a validation set with 80% training and 20%
# validation.
train <- sample(dim(Default)[1], dim(Default)[1] * 0.8)

# ii) Fit a multiple logistic regression model using only the training observations.
model.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)


# iii) Obtain a prediction of default status for each individual in the validation set by
# computing the posterior probability of default for that individual, and classifying the
# individual to the default category if the posterior probability is greater than 0.5.
probabilities <- predict(model.glm, newdata = Default[-train, ], type = "response")
pred <- rep("No", length(probabilities))
pred[probabilities > 0.5] <- "Yes"

# iv) Compute the validation set error, which is the fraction of the observations in the
# validation set that are misclassified.
cat ("The test error rate is", mean(pred != Default[-train, ]$default))
```

```
## The test error rate is 0.0315
```

```r
set.seed(20)
# i) Split the sample set into a training set and a validation set with 80% training and 20%
# validation.
train <- sample(dim(Default)[1], dim(Default)[1] * 0.8)

# ii) Fit a multiple logistic regression model using only the training observations.
model.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)

# iii) Obtain a prediction of default status for each individual in the validation set by
# computing the posterior probability of default for that individual, and classifying the
# individual to the default category if the posterior probability is greater than 0.5.
probabilities <- predict(model.glm, newdata = Default[-train, ], type = "response")
pred <- rep("No", length(probabilities))
pred[probabilities > 0.5] <- "Yes"

# iv) Compute the validation set error, which is the fraction of the observations in the
```

```r
# validation set that are misclassified.
cat ("The test error rate is", mean(pred != Default[-train, ]$default))
```

```
## The test error rate is 0.0275
```

```r
set.seed(30)
# i) Split the sample set into a training set and a validation set with 80% training and 20%
# validation.
train <- sample(dim(Default)[1], dim(Default)[1] * 0.8)

# ii) Fit a multiple logistic regression model using only the training observations.
model.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)

# iii) Obtain a prediction of default status for each individual in the validation set by
# computing the posterior probability of default for that individual, and classifying the
# individual to the default category if the posterior probability is greater than 0.5.
probabilities <- predict(model.glm, newdata = Default[-train, ], type = "response")
pred <- rep("No", length(probabilities))
pred[probabilities > 0.5] <- "Yes"

# iv) Compute the validation set error, which is the fraction of the observations in the
# validation set that are misclassified.
cat ("The test error rate is", mean(pred != Default[-train, ]$default))
```

```
## The test error rate is 0.0205
```

As the three outputs shown, the test error rate changes between 0.0205 and 0.0315. The test error rate depends on the different splits of training set and validation set.

d)

```r
# Adding a dummy variable for students to the model and repeat similar steps as previous
# question to calculate test error rate.
set.seed(6)
train <- sample(dim(Default)[1], dim(Default)[1] * 0.8)
model.glm <- glm(default ~ income + balance+ student, data = Default, family = "binomial",
                 subset = train)
probabilities <- predict(model.glm, newdata = Default[-train, ], type = "response")
pred <- rep("No", length(probabilities))
pred[probabilities > 0.5] <- "Yes"
cat ("The test error rate is", mean(pred != Default[-train, ]$default))
```

```
## The test error rate is 0.027
```

Since, as the output shown, the test error rate is 0.027 which is at the same level as the model without adding a dummy variable for student. Thus, adding a dummy variable for student does not lead to a reduction in the test error rate.

6.8 Question 1

a)

The model with k predictors from best subset selection has the smallest training RSS because it selects the model with the smallest training RSS among all C_k^p possible models with k predictors. Whereas, forward/backward stepwise selection only select the best model with the smallest training RSS among p-k or k models and based on the previous selected models (either the model with k-1 predictors or the model with k+1 predictors), so the "best" models from forward/backward stepwise selections are not guaranteed to have universal smallest training RSS among all models with k predictors.

b)

It is hard to determine. Best subset selection is likely to have the smallest test RSS because it selects the best model among all possible models. However, due to different splits of testing and training data, forward/backward stepwise selection may luckily to have the smallest test RSS.

c)

i)

True, since the predictors in (k+1)-variable model is the augmentation of k-variable model with one more variable in forward stepwise selection.

ii)

True, since the predictors in k-variable model comes from (k+1)-variable model with one predictor dropped in backward stepwise selection.

iii)

False, the predictors of the k-variable model from backward stepwise selection and the predictors of the (k+1)-variable model from forward stepwise selection are not related with each other. In other words, the models' selected predictors in consecutive steps from different methods are unrelated.

iv)

False, the predictors of the k-variable model from forward stepwise selection and the predictors of the (k+1)-variable model from backward stepwise selection are not related with each other. In other words, the models' selected predictors in consecutive steps from different methods are unrelated.

v)

False, in best subset selection, the (k+1)-variable model is selected from all possible (k+1)-variable models which may not necessarily include every predictor in the selected k-variable model. In other words, the models' selected predictors in consecutive steps of best subset selection may not be related.

6.8 Question 8

a)

```
set.seed(6)
X<-rnorm(100)
e<-rnorm(100)
```

b)

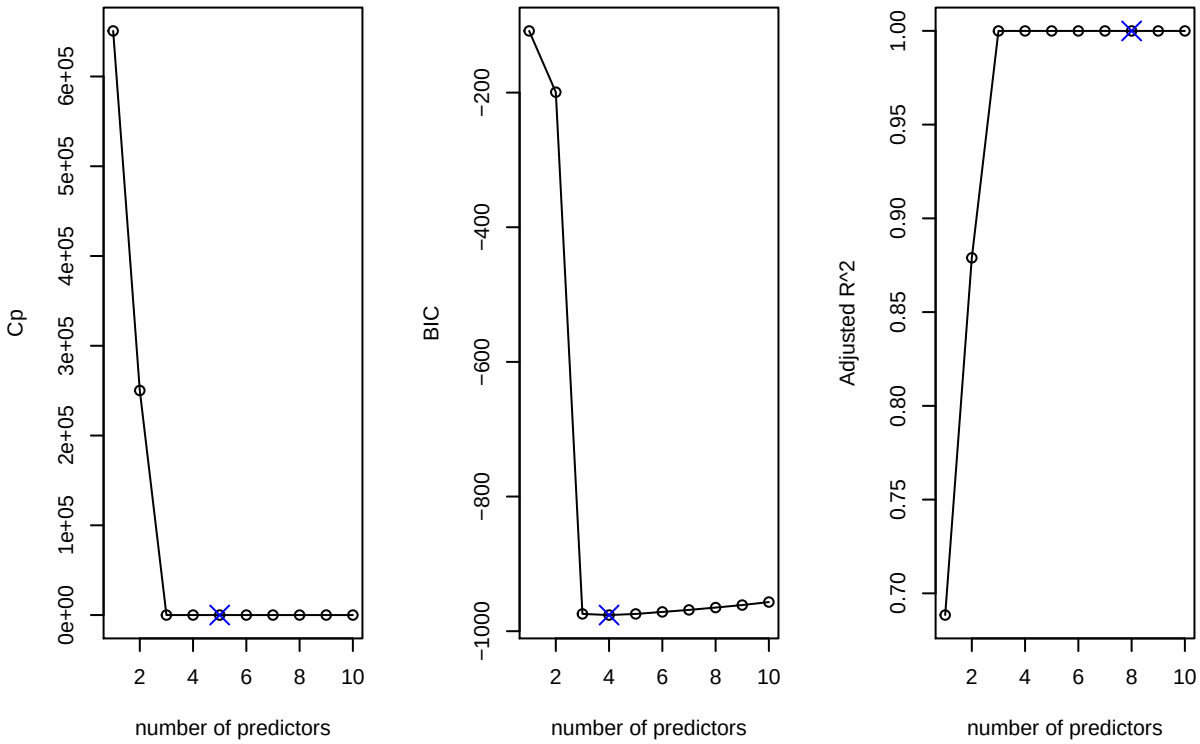
```
Y<- 10 + 10*X+20*(X^2)+30*(X^3)+e
```

c)

```
#install.packages("leaps")
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.3.2
```

```
mydf <- data.frame(X,Y)
bestSubset <- regsubsets(Y~poly(X, 10), data = mydf, nvmax = 10)
mySummary<-summary(bestSubset)
par(mfrow = c(1, 3))
plot(mySummary$cp, xlab = "number of predictors", ylab = "Cp", type = "o")
points(which.min(mySummary$cp), mySummary$cp[which.min(mySummary$cp)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$bic, xlab = "number of predictors", ylab = "BIC", type = "o")
points(which.min(mySummary$bic), mySummary$bic[which.min(mySummary$bic)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$adjr2, xlab = "number of predictors", ylab = "Adjusted R^2", type = "o")
points(which.max(mySummary$adjr2), mySummary$adjr2[which.max(mySummary$adjr2)], col = "blue"
       , cex = 2, pch = 4)
```



As the output shows, Cp is lowest at 5-variable model, BIC is lowest at 4-variable model, adjusted R^2 is highest at 8-variable model.

```
# best model according to Cp.
coef(bestSubset, which.min(mySummary$cp))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)7
##    45.299292  1057.570844   554.066790   437.880296    -1.470791
## poly(X, 10)9
##    -2.194827
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 1.47 \times x^7 - 2.19 \times x^9$$

```
# best model according to BIC.
coef(bestSubset, which.min(mySummary$bic))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)9
##    45.299292  1057.570844   554.066790   437.880296    -2.194827
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 2.19 \times x^9$$

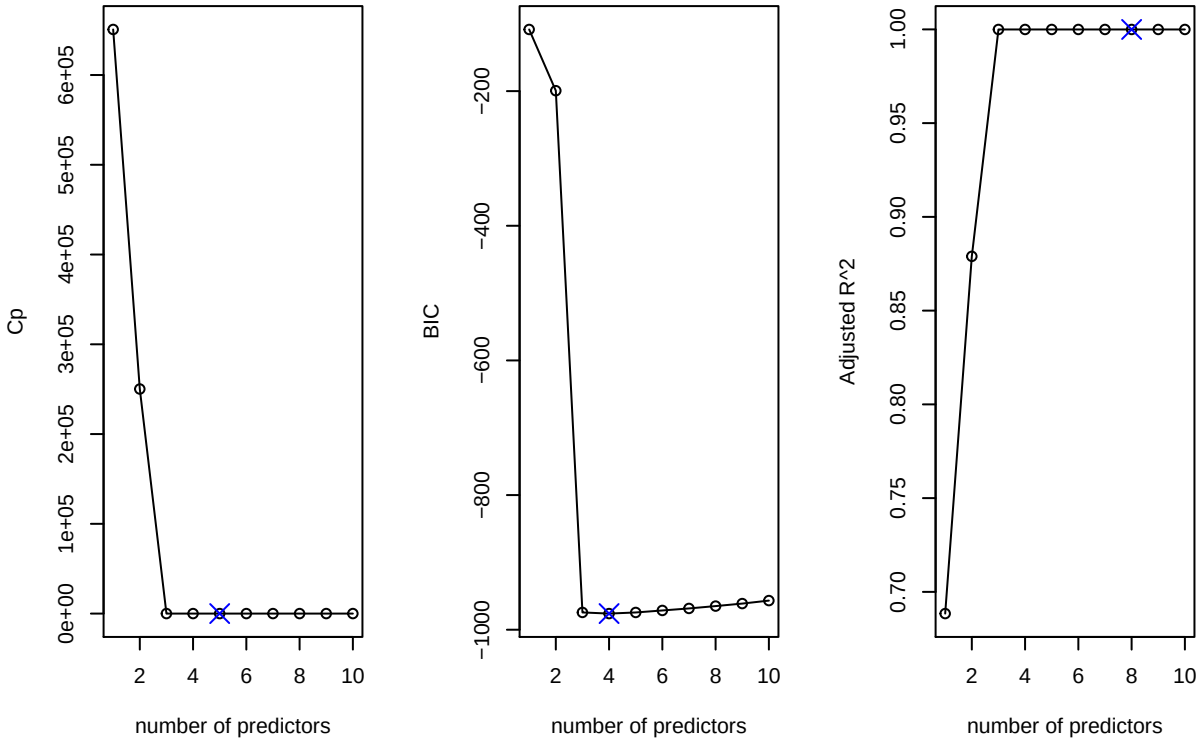
```
# best model according to adjusted rsquared.
coef(bestSubset, which.max(mySummary$adjr2))
```

```
##   (Intercept)  poly(X, 10)1  poly(X, 10)2  poly(X, 10)3  poly(X, 10)4
##    45.2992916  1057.5708437   554.0667903   437.8802960     1.0651987
##  poly(X, 10)6  poly(X, 10)7  poly(X, 10)9 poly(X, 10)10
##     0.8801486    -1.4707912    -2.1948269     1.0709511
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 + 1.07 \times x^4 + 0.88 \times x^6 - 1.47 \times x^7 - 2.19 \times x^9 + 1.07 \times x^{10}$$

d)

```
# Forward stepwise selection
forwardStepwise <- regsubsets(Y~poly(X, 10), data = mydf, nvmax = 10, method = "forward")
mySummary<-summary(forwardStepwise)
par(mfrow = c(1, 3))
plot(mySummary$cp, xlab = "number of predictors", ylab = "Cp", type = "o")
points(which.min(mySummary$cp), mySummary$cp[which.min(mySummary$cp)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$bic, xlab = "number of predictors", ylab = "BIC", type = "o")
points(which.min(mySummary$bic), mySummary$bic[which.min(mySummary$bic)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$adjr2, xlab = "number of predictors", ylab = "Adjusted R^2", type = "o")
points(which.max(mySummary$adjr2), mySummary$adjr2[which.max(mySummary$adjr2)], col = "blue"
       , cex = 2, pch = 4)
```



As the output shows, Cp is lowest at 5-variable model, BIC is lowest at 4-variable model, adjusted R^2 is highest at 8-variable model.

```
# best model according to Cp.
coef(bestSubset, which.min(mySummary$cp))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)7
##    45.299292  1057.570844   554.066790   437.880296    -1.470791
## poly(X, 10)9
##    -2.194827
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 1.47 \times x^7 - 2.19 \times x^9$$

```r
# best model according to BIC.
coef(bestSubset, which.min(mySummary$bic))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)9
##    45.299292  1057.570844   554.066790   437.880296    -2.194827
```

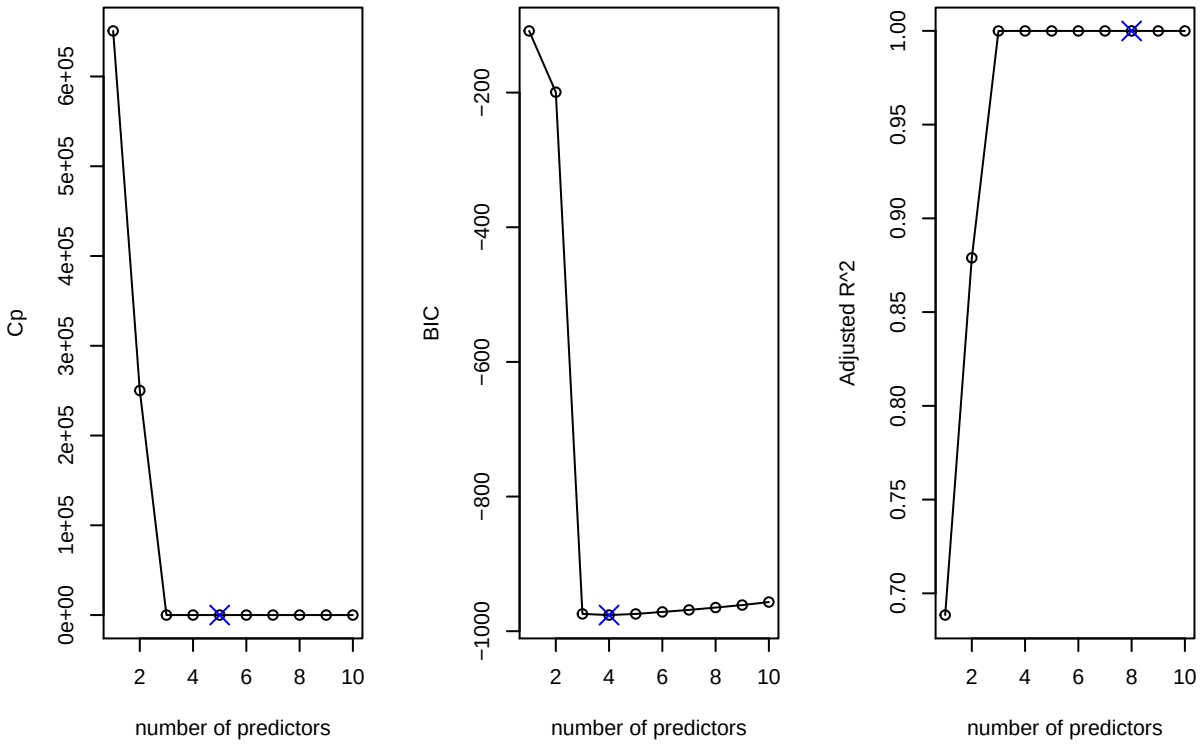
$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 2.19 \times x^9$$

```r
# best model according to adjusted rsquared.
coef(bestSubset, which.max(mySummary$adjr2))
```

```
##   (Intercept)  poly(X, 10)1  poly(X, 10)2  poly(X, 10)3  poly(X, 10)4
##    45.2992916  1057.5708437   554.0667903   437.8802960     1.0651987
##  poly(X, 10)6  poly(X, 10)7  poly(X, 10)9 poly(X, 10)10
##     0.8801486    -1.4707912    -2.1948269     1.0709511
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 + 1.07 \times x^4 + 0.88 \times x^6 - 1.47 \times x^7 - 2.19 \times x^9 + 1.07 \times x^{10}$$

```r
# Backward stepwise selection
backwardStepwise <- regsubsets(Y~poly(X, 10), data = mydf, nvmax = 10, method = "backward")
mySummary<-summary(backwardStepwise)
par(mfrow = c(1, 3))
plot(mySummary$cp, xlab = "number of predictors", ylab = "Cp", type = "o")
points(which.min(mySummary$cp), mySummary$cp[which.min(mySummary$cp)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$bic, xlab = "number of predictors", ylab = "BIC", type = "o")
points(which.min(mySummary$bic), mySummary$bic[which.min(mySummary$bic)], col = "blue",
       cex = 2, pch = 4)
plot(mySummary$adjr2, xlab = "number of predictors", ylab = "Adjusted R^2", type = "o")
points(which.max(mySummary$adjr2), mySummary$adjr2[which.max(mySummary$adjr2)], col = "blue"
       , cex = 2, pch = 4)
```



As the output shows, Cp is lowest at 5-variable model, BIC is lowest at 4-variable model, adjusted R^2 is highest at 8-variable model.

```
# best model according to Cp.
coef(bestSubset, which.min(mySummary$cp))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)7
##    45.299292  1057.570844   554.066790   437.880296    -1.470791
## poly(X, 10)9
##    -2.194827
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 1.47 \times x^7 - 2.19 \times x^9$$

```
# best model according to BIC.
coef(bestSubset, which.min(mySummary$bic))
```

```
##  (Intercept) poly(X, 10)1 poly(X, 10)2 poly(X, 10)3 poly(X, 10)9
##    45.299292  1057.570844   554.066790   437.880296    -2.194827
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 - 2.19 \times x^9$$

```
# best model according to adjusted rsquared.
coef(bestSubset, which.max(mySummary$adjr2))
```

```
##   (Intercept)  poly(X, 10)1  poly(X, 10)2  poly(X, 10)3  poly(X, 10)4
##    45.2992916  1057.5708437   554.0667903   437.8802960     1.0651987
##  poly(X, 10)6  poly(X, 10)7  poly(X, 10)9 poly(X, 10)10
##     0.8801486    -1.4707912    -2.1948269     1.0709511
```

$$\hat{y} = 45.30 + 1057.57 \times x + 554.07 \times x^2 + 437.88 \times x^3 + 1.07 \times x^4 + 0.88 \times x^6 - 1.47 \times x^7 - 2.19 \times x^9 + 1.07 \times x^{10}$$

According to the outputs, the three methods (best subset, foward/backward stepwise selection) have the same results (Cp is lowest at 5-variable model, BIC is lowest at 4-variable model, adjusted R^2 is highest at 8-variable model).

5.4 Question 5 (Repeat b d w/ 10-fold cross validation)

b)

```
library(ISLR)

model.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
#summary(model.glm)
library(boot)
set.seed(6)
# Since the response is a binary variable an appropriate cost function is
cost <- function(r, pi = 0){
  mean(abs(r-pi) > 0.5)
}
# 10-fold cross validation
cv.default = cv.glm(Default, model.glm, cost = cost,K = 10)

cat("The error/misclassification rate for 10 fold CV is ", cv.default$delta[1])
```

```
## The error/misclassification rate for 10 fold CV is  0.0261
```

d)

```
model.glm <- glm(default ~ income + balance+ student, data = Default, family = "binomial")
set.seed(6)
# Since the response is a binary variable an appropriate cost function is
cost <- function(r, pi = 0){
  mean(abs(r-pi) > 0.5)
}
# 10-fold cross validation
cv.default = cv.glm(Default, model.glm, cost = cost,K = 10)
cv.default$delta[1]
```

```
## [1] 0.0265
```

Since, as the output shown, the test error rate is about 0.027 which is at the same level as or slightly larger than the model without adding a dummy variable for student. Thus, adding a dummy variable for student does not lead to a reduction in the test error rate.

5.4 Question 5 (Repeat b d w/ LOOCV)

b)

```
library(ISLR)

model.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
```

```r
#summary(model.glm)
library(boot)
set.seed(6)
# Since the response is a binary variable an appropriate cost function is
cost <- function(r, pi = 0){
  mean(abs(r-pi) > 0.5)
}
# LOOCV
cv.default = cv.glm(Default, model.glm, cost = cost)
cat("The error/misclassification rate for Leave one out cross validation is ", cv.default$delta[1])
```

```
## The error/misclassification rate for Leave one out cross validation is  0.0263
```

d)

```r
model.glm <- glm(default ~ income + balance+ student, data = Default, family = "binomial")
set.seed(6)
# Since the response is a binary variable an appropriate cost function is
cost <- function(r, pi = 0){
  mean(abs(r-pi) > 0.5)
}
# LOOCV
cv.default = cv.glm(Default, model.glm, cost = cost)
cv.default$delta[1]
```

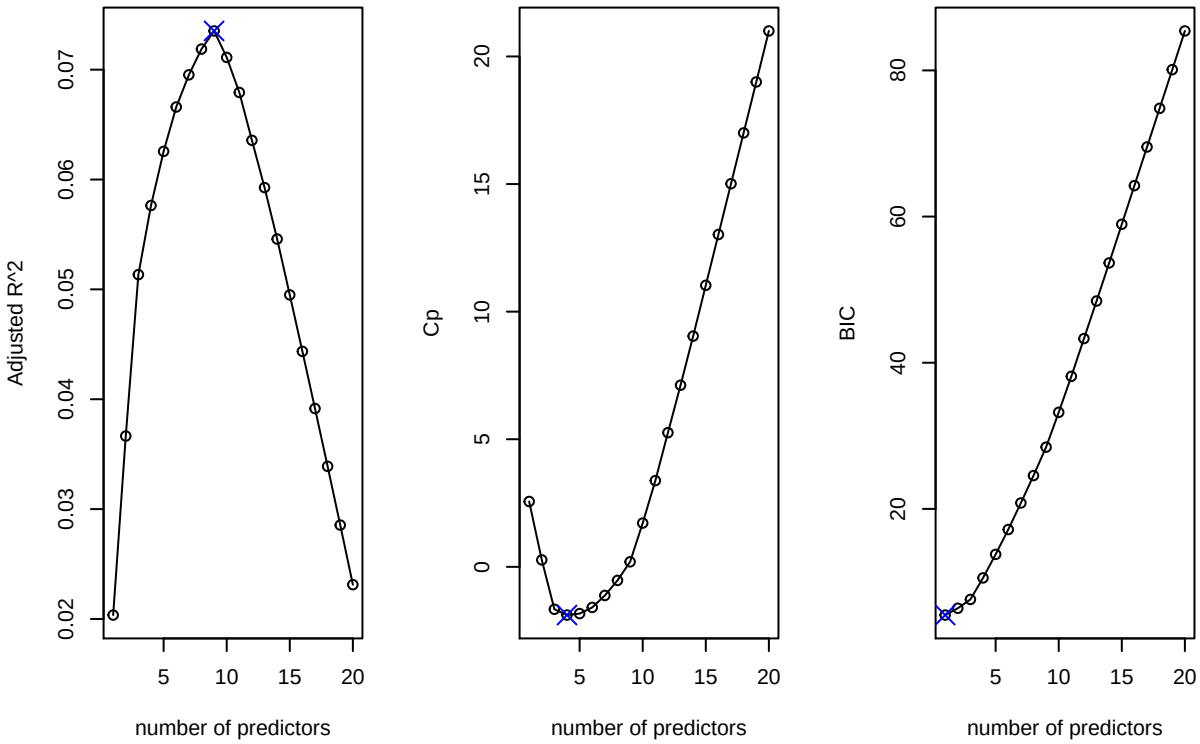
```
## [1] 0.0268
```

Since, as the output shown, the test error rate is about 0.027 which is at the same level as or slightly larger than the model without adding a dummy variable for student. Thus, adding a dummy variable for student does not lead to a reduction in the test error rate.

Best Subset Selection Question

a)

```r
library(leaps)
set.seed(107)
X = as.data.frame(matrix(rnorm(4200), ncol = 21))
names(X)[1] <- "y"
bestSubset <- regsubsets(y~., data = X, nvmax = 20)
mySummary<-summary(bestSubset)
par(mfrow = c(1, 3))
plot(mySummary$adjr2, xlab = "number of predictors", ylab = "Adjusted R^2", type = "o")
points(which.max(mySummary$adjr2), mySummary$adjr2[which.max(mySummary$adjr2)], col = "blue"
       , cex = 2, pch = 4)
plot(mySummary$cp, xlab = "number of predictors", ylab = "Cp", type = "o")
points(which.min(mySummary$cp), mySummary$cp[which.min(mySummary$cp)], col = "blue",
cex = 2, pch = 4)
plot(mySummary$bic, xlab = "number of predictors", ylab = "BIC", type = "o")
points(which.min(mySummary$bic), mySummary$bic[which.min(mySummary$bic)], col = "blue",
cex = 2, pch = 4)
```



As the plots shown, the best model size for adjusted R^2 criteria is 9 because adjusted R^2 is highest at model size 9.

The best model size for Mallows Cp is 4 because Mallows Cp is lowest at model size 4.

The best model size for Bayes Information Criterion is 1 because BIC is lowest at model size 1.

b)

```
library(boot)
#install.packages("gmp")
#install.packages("HH")
library(HH)
```

```
## Warning: package 'HH' was built under R version 3.3.2
## Loading required package: lattice
##
## Attaching package: 'lattice'
## The following object is masked from 'package:boot':
##
##     melanoma
## Loading required package: grid
## Loading required package: latticeExtra
## Loading required package: RColorBrewer
## Loading required package: multcomp
## Loading required package: mvtnorm
```

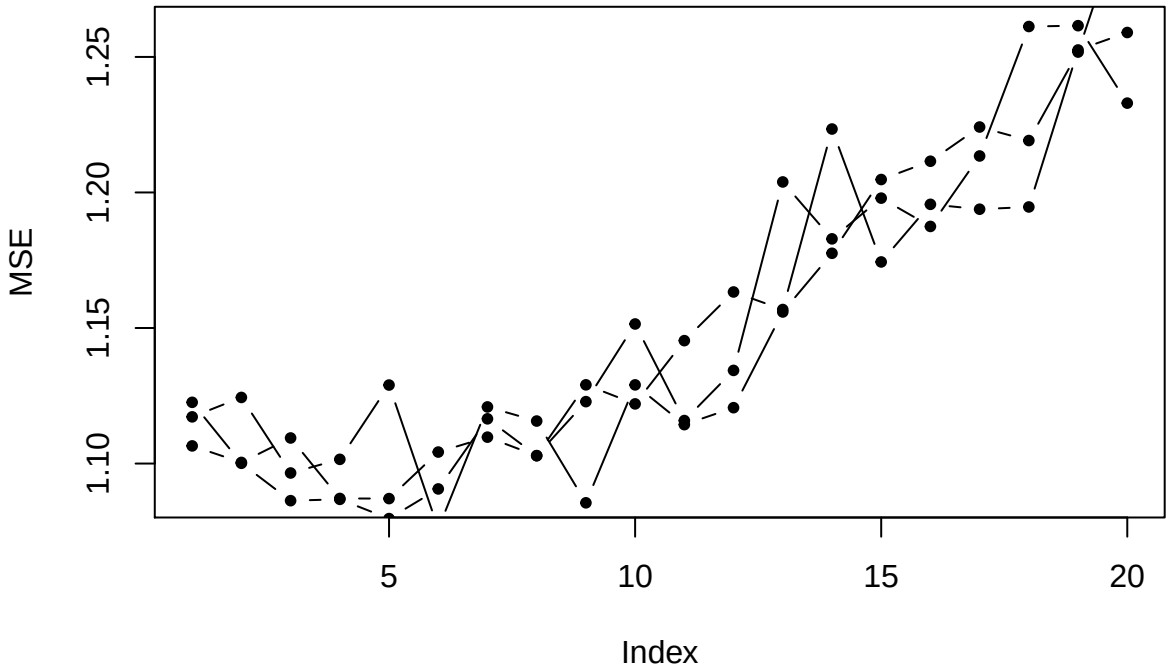
```
## Warning: package 'mvtnorm' was built under R version 3.3.2
## Loading required package: survival
##
## Attaching package: 'survival'
## The following object is masked from 'package:boot':
##
##     aml
## Loading required package: TH.data
## Warning: package 'TH.data' was built under R version 3.3.2
## Loading required package: MASS
##
## Attaching package: 'TH.data'
## The following object is masked from 'package:MASS':
##
##     geyser
## Loading required package: gridExtra
##
## Attaching package: 'HH'
## The following object is masked from 'package:boot':
##
##     logit
```

```
myDelta<-function(num_variables){
  model<-lm.regsubsets(bestSubset, num_variables)
  r1<-cv.glm(X, glm(model), K = 10)$delta[1]
  r2<-cv.glm(X, glm(model), K = 10)$delta[1]
  r3<-cv.glm(X, glm(model), K = 10)$delta[1]
  range(r1,r2,r3)
}

myDelta1<-function(num_variables){
  model<-lm.regsubsets(bestSubset, num_variables)
  cv.glm(X, glm(model), K = 10)$delta[1]

}

set.seed(11)
for(i in c(1:20)){
  cat ("Error for model size of",i,"is between",myDelta(i)[1],"and",myDelta(i)[2], "\n")
}
```

```
## Error for model size of 1 is between 1.106506 and 1.121885
## Error for model size of 2 is between 1.094966 and 1.110056
## Error for model size of 3 is between 1.092583 and 1.102988
## Error for model size of 4 is between 1.098967 and 1.11681
## Error for model size of 5 is between 1.079332 and 1.106135
## Error for model size of 6 is between 1.077003 and 1.100056
## Error for model size of 7 is between 1.09847 and 1.1405
## Error for model size of 8 is between 1.09937 and 1.122391
```

```
## Error for model size of 9 is between 1.105353 and 1.154517
## Error for model size of 10 is between 1.119566 and 1.145204
## Error for model size of 11 is between 1.134963 and 1.143928
## Error for model size of 12 is between 1.135141 and 1.156166
## Error for model size of 13 is between 1.128785 and 1.164939
## Error for model size of 14 is between 1.176015 and 1.181191
## Error for model size of 15 is between 1.143862 and 1.208781
## Error for model size of 16 is between 1.135758 and 1.22054
## Error for model size of 17 is between 1.203654 and 1.222236
## Error for model size of 18 is between 1.199302 and 1.231342
## Error for model size of 19 is between 1.21385 and 1.272799
## Error for model size of 20 is between 1.223498 and 1.274105
```

```r
cv.errors <- matrix(NA, 3, 20) # 3 iterations = 3 rows;  20 variables = 20 columns
set.seed(11)
for(i in c(1:20)){
  cv.errors[1,i] = myDelta1(i)
  cv.errors[2,i] = myDelta1(i)
  cv.errors[3,i] = myDelta1(i)
}
plot (cv.errors[1,], pch=20, type="b", ylab = "MSE")
lines (cv.errors[2,], pch=20, type="b")
lines (cv.errors[3,], pch=20, type="b")
```



As the output shows, the model achives lowest MSE between 5 to 9. Models with size less than or equal to 9 has comparative lower MSE.

c)

Compare all four approaches, I would choose 9 as the best model size since it has the highest adjusted R^2, comparatively low MSE, comparatively low Cp and comparatively low BIC. Thus, adjusted R^2 makes sense.

```r
cat ("The best model is:")
```

```
## The best model is:
```

```r
coef(bestSubset, 9)
```

```
## (Intercept)          V4          V7          V9         V10         V13
##  0.01560445 -0.16245658  0.11554916 -0.16144077 -0.10240625 -0.12926352
##         V15         V16         V18         V20
## -0.08580679 -0.08462870  0.15317684  0.09810784
```