In-class activity 7

You are expected to complete this activity by yourself and everybody should submit your worksheet. Discussions with classmates are encouraged. Some codes of this activity are provided. Please implement them in R and then answer the questions.

1. (**Menu pricing**) Recall **Example 5.3.1** in the lecture notes. You have been asked to produce a regression model to predict the price of dinner. Data from surveys of customers of 168 Italian restaurants in the target area are available.

The data "nyc.csv" is available on Blackboard. The data are in the form of the average of customer views on

Y = Price = the price (in \$US) of dinner (including 1 drink & a tip)

X_1 = Food = customer rating of the food (out of 30)

X_2 = Décor = customer rating of the decor (out of 30)

X_3 = Service = customer rating of the service (out of 30)

X_4 = East = dummy variable = 1 (0) if the restaurant is east (west) of Fifth Avenue

(a) Exploratory data analysis. Scatter plots and boxplots would be a nice visual display for the data.

```
nyc <- read.csv("nyc.csv", header=TRUE)
attach(nyc)
pairs(~Price+Food+Decor+Service)
```

```
par(mfrow=c(1,1))
boxplot(Price~East, xlab = "East", ylab = "Price")
```

(b) The full model we can consider here is

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \epsilon.$$

The purpose of this problem is seeking for a "best" model to do prediction. Given the full model above, find the "best" model applying the all possible subsets selection method and do model validation.

i. We first randomly split the data to two groups. One is called "training data" for variable selection. The other one is called "test data" for model validation.

```
#the number of predictors in full model
  m <- 4
#sample size
  n <- length(Price)
#split the data
set.seed(3)
train.indx <- sample(1:n, n/2)

#Select training data
y.train <- nyc[train.indx,3]
x.train <- as.matrix(nyc[train.indx,4:(3+m)])

#Select the test data
y.test <- nyc[-train.indx,3]
x.test <- as.matrix(nyc[-train.indx,4:(3+m)])
```

ii. Based on the R output by running the R-script below, what will be your "best" model and why?

```r
#All possible subsets selection
library(leaps)
b <- regsubsets(x.train, y.train)
rs <- summary(b)
rs$outmat

#calculate adjusted R^2, AIC, corrected AIC, and BIC
om1 <- lm(y.train~x.train[,2])
om2 <- lm(y.train~x.train[,2]+x.train[,1])
om3 <- lm(y.train~x.train[,2]+x.train[,1]+x.train[,4])
om4 <- lm(y.train~x.train[,2]+x.train[,1]+x.train[,4]+x.train[,3])

om <- list(om1,om2,om3,om4)
n.train <- length(y.train)

#number of parameters in each model in "om"
npar <- 3:(2+m)
#adjusted R^2
Rsq.adj <- round(rs$adjr2,3)
Rsq.adj
#AIC
AIC<- sapply(1:m, function(x) round(extractAIC(om[[x]],k=2)[2],2))
AIC
#corrected AIC
AICc <- sapply(1:m, function(x) round(extractAIC(om[[x]],k=2)[2]+
                                      2*npar[x]*(npar[x]+1)/(n-npar[x]+1),2))
AICc
#BIC
BIC<- sapply(1:m, function(x) round(extractAIC(om[[x]],k=log(n))[2],2))
BIC

#all possible subsets selection with PRESS
myPRESS <- function(x,y,indx){
  m1 <- lm(y~x[,indx])
  press <- sum((m1$residuals/(1-hatvalues(m1)))^2)
  return(press)
}

PRESS.indx <- matrix("", nrow = m, ncol = m)
colnames(PRESS.indx) <- c("Food", "Decor", "Service", "East")
PRESS <- rep(0,m)
for(i in 1:m)
{
  indx <- combn(m,i)
  n.indx <- ncol(indx)
  tmp <- sapply(1:n.indx, function(m) myPRESS(x.train, y.train, indx[,m]))
  PRESS[i] <- round(min(tmp),2)
  PRESS.indx[i, indx[,which.min(tmp)]] <- "*"
}

PRESS.indx
PRESS
```

iii. Draw the diagnostic plots for the "best" model. Based on your results, is it a valid model?

iv. Calibrate the predictive capability of the selected model using test data (Calculate the **mean squared prediction errors (MSPR)**).

```
#predicted value for the test data

yPred <- sapply(1:length(y.test), function(m) sum(om3$coefficients*c(1,x.test[m,c(2,1,4)])))
plot(y.test, yPred, main="Predicted Price vs Observed Price", xlab ="observed Price in test data",
     ylab = "predicted Price")
```

```
#MSPE
MSPE <- mean((y.test-yPred)^2)
MSPE
```

v. We could also try "lasso" method. Given the "lasso" output below, does it improve the prediction performance?

```
#lasso fitting
library(glmnet)
fit.lasso <- glmnet(x.train,y.train, family = 'gaussian', alpha = 1)
par(mar=c(5.1,4.1,4.1,4.1), mfrow=c(1,1))
plot(fit.lasso)
vnat <- coef(fit.lasso)
vnat2 <- vnat[-1, ncol(vnat)]
axis(4, at=vnat2, line=-.5, label=names(vnat2), las=1, tick=FALSE, cex.axis=1)
```

```
#5-fold cross validation for the training dataset
cv <- cv.glmnet(x.train,y.train, type.measure = "mse", nfolds = 5, alpha = 1, family = "gaussian")

#plot
plot(cv,xlab=expression(log(lambda)),  ylab= "Cross validation error")
text(log(cv$lambda.min), 80, substitute(atop(paste("usual rule:"), paste(hat(lambda),
         "=", lam.min,sep="")), list(lam.min=round(cv$lambda.min,4))))
text(log(cv$lambda.1se), 80, substitute(atop(paste("one standard deivation rule:"),
   paste(hat(lambda), "=", lam.1se,sep="")), list(lam.1se=round(cv$lambda.1se,2))))
```

```
#Selected model by the one standard deviation rule
coef(cv, cv$lambda.1se)

#the MSPE
yPred <- predict(cv, s = cv$lambda.1se, new = x.test)

plot(y.test, yPred, main="Predicted Price vs Observed Price", xlab ="observed Price in test data",
     ylab = "predicted Price")
```

```
#MSPE
MSPE <- mean((y.test-yPred)^2)
MSPE


detach(nyc)
```