**HW 4 - LOOCV and K-Fold**

Ian Douglas 3/3/2019

### MSpE-train = mean((mod1$fitted.values - train$y)^2)  
### MSpEtest = mean((predict(mod1, newdata = test) - test$y)^2)  
data <-  
read.table("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/KutnerData/Chapter%20%209%20Data%20Sets/CH09PR15.txt", header = FALSE)  
names(data) <- c("y","creat","age","lbs")

### Part A

#Randomly sample 20 rows (observations) for the training dataset  
set.seed(90210)  
index <- sample(1:nrow(data), size = 20, replace = FALSE)  
train <- data[index,]  
test <- data[-index,]  
# Create every possible linear model using a function  
# (use <<- to produce objects that appear in the global environment)  
all.models <- function(y, x1, x2, x3, data) {  
 mod1 <<- lm(y ~ x1, data = data)  
 mod2 <<- lm(y ~ x2, data = data)  
 mod3 <<- lm(y ~ x3, data = data)  
 mod4 <<- lm(y ~ x1 + x2, data = data)  
 mod5 <<- lm(y ~ x1 + x3, data = data)  
 mod6 <<- lm(y ~ x2 + x3, data = data)  
 mod7 <<- lm(y ~ x1 + x2 + x3, data = data)  
}  
#Run function on train dataset  
all.models(y=train$y, x1=train$creat, x2=train$age, x3=train$lbs, data=train)  
  
#Generation of Mean Square Prediction Error of test prediction  
#A named-list would be useful here...but just as much typing  
MSpEts <- as.data.frame(rbind(mean((predict(mod1, newdata = test) - test$y)^2),  
 mean((predict(mod2, newdata = test) - test$y)^2),  
 mean((predict(mod3, newdata = test) - test$y)^2),  
 mean((predict(mod4, newdata = test) - test$y)^2),  
 mean((predict(mod5, newdata = test) - test$y)^2),  
 mean((predict(mod6, newdata = test) - test$y)^2),  
 mean((predict(mod7, newdata =test) - test$y)^2)),  
 row.names = c("mod1","mod2","mod3","mod4",  
 "mod5", "mod6", "mod7"))

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod1, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod2, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod3, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod4, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod5, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod6, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

## Warning: 'newdata' had 13 rows but variables found have 20 rows

## Warning in predict(mod7, newdata = test) - test$y: longer object length is  
## not a multiple of shorter object length

MSpEts

## V1  
## mod1 1428.538  
## mod2 1261.052  
## mod3 1018.332  
## mod4 1524.430  
## mod5 1483.988  
## mod6 1469.416  
## mod7 1627.698

# V1  
# mod1 1428.538  
# mod2 1261.052  
# mod3 1018.332  
# mod4 1524.430  
# mod5 1483.988  
# mod6 1469.416  
# mod7 1627.698  
MSpEts$V1[which(MSpEts$V1 == min(MSpEts$V1))]

## [1] 1018.332

#[1] 1018.332

The third model with weight as the predictor showed the lowest prediction error.

### Part B: LOOCV

#library(boot)  
#redefine the models using the full dataset (use glm() for compatability  
#with LOOCV)  
model1 <- glm(y ~ creat, family = "gaussian", data = data)  
model2 <- glm(y ~ age, family = "gaussian", data = data)  
model3 <- glm(y ~ lbs, family = "gaussian", data = data)  
model4 <- glm(y ~ creat + age, family = "gaussian", data = data)  
model5 <- glm(y ~ creat + lbs, family = "gaussian", data = data)  
model6 <- glm(y ~ age + lbs, family = "gaussian", data = data)  
model7 <- glm(y ~ creat + age + lbs, family = "gaussian", data = data)  
  
#Output MSE generated from LOOCV of each model:  
MSEloocv <- rbind(boot::cv.glm(data = data, glmfit=model1)$delta[1],  
 boot::cv.glm(data = data, glmfit=model2)$delta[1],  
 boot::cv.glm(data = data, glmfit=model3)$delta[1],  
 boot::cv.glm(data = data, glmfit=model4)$delta[1],  
 boot::cv.glm(data = data, glmfit=model5)$delta[1],  
 boot::cv.glm(data = data, glmfit=model6)$delta[1],  
 boot::cv.glm(data = data, glmfit=model7)$delta[1])  
MSEloocv

## [,1]  
## [1,] 375.8365  
## [2,] 596.2457  
## [3,] 915.1003  
## [4,] 283.3002  
## [5,] 312.9195  
## [6,] 450.4356  
## [7,] 180.7228

# [,1]  
# [1,] 375.8365  
# [2,] 596.2457  
# [3,] 915.1003  
# [4,] 283.3002  
# [5,] 312.9195  
# [6,] 450.4356  
# [7,] 180.7228  
min(MSEloocv)

## [1] 180.7228

#[1] 180.7228

Unsurprisingly, the “saturated” model had the best predictive validity. This is probably evidence of overfitting.

### Part C: K-Fold

#K-fold cross validation where K = 3, using the models built on the full data  
MSEkfold <- NULL  
MSEkfold[1] <- (boot::cv.glm(data = data, glmfit = model1, K = 3))$delta[1]  
MSEkfold[2] <- (boot::cv.glm(data = data, glmfit = model2, K = 3))$delta[1]  
MSEkfold[3] <- (boot::cv.glm(data = data, glmfit = model3, K = 3))$delta[1]  
MSEkfold[4] <- (boot::cv.glm(data = data, glmfit = model4, K = 3))$delta[1]  
MSEkfold[5] <- (boot::cv.glm(data = data, glmfit = model5, K = 3))$delta[1]  
MSEkfold[6] <- (boot::cv.glm(data = data, glmfit = model6, K = 3))$delta[1]  
MSEkfold[7] <- (boot::cv.glm(data = data, glmfit = model7, K = 3))$delta[1]  
MSEkfold

## [1] 353.4859 633.0958 925.8961 321.8047 275.6888 416.2705 194.0680

#[1] 379.0268 581.0623 845.1880 374.2564 385.9217 503.6629 201.3240

Again, the saturated model showed the lowest MSE, however this time, the model with just creatinine concentration and age predicted fairly well.