Cross Validation

Christopher Daigle

Oct. 23 2018

```
library(boot)
library(ISLR)
rm(list = ls())
set.seed(1)
```

Exercise 1 #### Cross validation can also be used to estimate the test error for a classification problem. Run a logit model with the Smarket data. The dependent variable is Direction

```
glm.fit <-
  glm(Direction ~ Lag1 + Lag2, family = binomial, data = Smarket)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Smarket)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.387  -1.203   1.073   1.147   1.346
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.07425    0.05667   1.310    0.190
## Lag1        -0.07151    0.05010  -1.427    0.153
## Lag2        -0.04450    0.05000  -0.890    0.374
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1731.2  on 1249  degrees of freedom
## Residual deviance: 1728.4  on 1247  degrees of freedom
## AIC: 1734.4
##
## Number of Fisher Scoring iterations: 3
```

Compare this model with the following models using K-Fold cross-validation with K=10 Direction ~Lag1+Lag2+Lag3, Direction ~Lag1+Lag2+Lag3+Lag4, Direction ~Lag1+Lag2+Lag3+Lag5

```
# Creating K-Fold 'bins'
n <- nrow(Smarket)
x <- 1:n
cv.error <- matrix(NA, 4, 11)
rownames(cv.error) <- c('Model1', 'Model2', 'Model3', 'Model4')
colnames(cv.error) <-
  c(
    'MSEK-Fold1',
    'MSEK-Fold2',
    'MSEK-Fold3',
    'MSEK-Fold4',
    'MSEK-Fold5',
```

```
    'MSEK-Fold6',
    'MSEK-Fold7',
    'MSEK-Fold8',
    'MSEK-Fold9',
    'MSEK-Fold10',
    'MeanMSE'
  )

for (i in 1:4) {
  for (j in 1:10) {
    glm.fit1 <-
      glm(Direction ~ Lag1 + Lag2, family = binomial, data = Smarket)
    glm.fit2  <-
      glm(Direction ~ Lag1 + Lag2 + Lag3,
          family = binomial,
          data = Smarket)
    glm.fit3  <-
      glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4,
          family = binomial,
          data = Smarket)
    glm.fit4  <-
      glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5,
          family = binomial,
          data = Smarket)
    cv.error[i, j] <- cv.glm(Smarket, glm.fit1, K = 10)$delta[1]
  }
  cv.error[i, 11] <- mean(cv.error[i,], na.rm = TRUE)
}
cv.error
```

```
##        MSEK-Fold1 MSEK-Fold2 MSEK-Fold3 MSEK-Fold4 MSEK-Fold5 MSEK-Fold6
## Model1  0.2510845  0.2507110  0.2502009  0.2505966  0.2506181  0.2501461
## Model2  0.2504039  0.2502799  0.2503015  0.2505412  0.2499754  0.2503921
## Model3  0.2499181  0.2499972  0.2502963  0.2503801  0.2503917  0.2503733
## Model4  0.2511568  0.2501327  0.2504992  0.2508246  0.2503305  0.2503038
##        MSEK-Fold7 MSEK-Fold8 MSEK-Fold9 MSEK-Fold10   MeanMSE
## Model1  0.2511562  0.2510622  0.2500436   0.2506138 0.2506233
## Model2  0.2503264  0.2501024  0.2506971   0.2502256 0.2503246
## Model3  0.2503203  0.2508028  0.2498398   0.2499493 0.2502269
## Model4  0.2502949  0.2502806  0.2504433   0.2508866 0.2505153
```

```
numbers = c('first', 'second', 'third', 'fourth')
for (i in 1:4) {
  print(paste('The MSE for the', numbers[i], 'model is:', cv.error[i,11]))
}
```

```
## [1] "The MSE for the first model is: 0.250623306933055"
## [1] "The MSE for the second model is: 0.250324563431918"
## [1] "The MSE for the third model is: 0.250226881189798"
## [1] "The MSE for the fourth model is: 0.250515307870842"
```

Exercise 2 #### Consider KNN Estimation to predict direction using Lag1 and Lag2. To choose the optimal number of neighbors, use the K=10. Fold cross validation. Use only 2004 and 2005 year data. Instructed to use all of the data set for divisibility.

```r
library(ISLR)
library(class)
rm(list = ls())
set.seed(1)

df <- Smarket
X <- df[, c('Lag1', 'Lag2')]
y <- df$Direction

n <- nrow(df)
ind <- 1:n

optKs <- rep(NA,10)

for (i in 1:10) {
  testSplit <- sample(ind, size = n/10, replace = FALSE)

  train <- df[-testSplit, ]
  test <- df[testSplit, ]

  trainX <- train[, c('Lag1', 'Lag2')]
  testX <- test[, c('Lag1', 'Lag2')]
  trainY <- train$Direction
  testY <- test$Direction

  ind <- ind[-testSplit]

  Acc <- rep(NA, 100)

  for (j in 1:100) {
    knnPred <- knn(trainX, testX, trainY, k = j)
    Acc[j] <- mean(testY == knnPred)
  }
  optKs <- which.max(Acc)
}
optKs
```

```
## [1] 38
```

```r
print(paste('The best K for this is', optKs))
```

```
## [1] "The best K for this is 38"
```