Chapter 5 HW

Ulices Gonzalez

April 15th, 2025

## Conceptual Questions

### 3. We now review k-fold cross-validation.

#### (a) Explain how k-fold cross-validation is implemented.

Set of observations are divided in to k groups of equal size. Each group/fold is treated as a validation set and the method is fit on the remaining folds. The MSE is used then found using the observations in the held out groups/folds. This is repeated k times with a different group being used as a validation each time. At the end averaging the k test errors will result in the k-fold cv estimate.

#### (b) What are the advantages and disadvantages of k-fold cross validation relative to:

##### i. The validation set approach?

* K-Fold gives data on the variability of the models performance through the use of using k amount of error estimates.
* There is less bias in the model since it is trained and validation using all of the data its provided in all of its iterations.
* Due to each iteration using of the data set its more efficient in how it handles data.

##### ii. LOOCV?

* Given that there is less bias present in the k-fold method this also means that it has higher variance than normal even with a small k.
* The use of k requires the model to be fit k times which can be inefficient especially if the data set being used is large.

## Applied Questions

### 5. In Chapter 4, we used logistic regression to predict the probability of default using income and balance on the Default data set. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

library(caTools)  
default <- read.csv("DataSets/Default.csv")  
default$default <- ifelse(default$default == "Yes", 1, 0)  
default$student <- ifelse(default$student == "Yes", 1, 0)  
set.seed(0)

#### (a) Fit a logistic regression model that uses income and balance to predict default.

defaultFit <- glm(default ~ income + balance, data = default, family = binomial)  
summary(defaultFit)

##   
## Call:  
## glm(formula = default ~ income + balance, family = binomial,   
## data = default)  
##   
## 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) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

##### i. Split the sample set into a training set and a validation set.

defaultSplit = sample.split(default$default, SplitRatio = 0.625)  
trainingSet <- subset(default, defaultSplit == TRUE)  
validSet <- subset(default, defaultSplit == FALSE)

##### ii. Fit a multiple logistic regression model using only the training observations.

defaultLRModel <- glm(default ~ income + balance, data = trainingSet, family = "binomial")

##### 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.

defaultProbability=predict(defaultLRModel, newdata = validSet, type="response")  
defaultPrediction <- ifelse(defaultProbability > 0.5, 1, 0)  
table(defaultPrediction,validSet$default)

##   
## defaultPrediction 0 1  
## 0 3607 80  
## 1 18 45

##### iv. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassifed.

mean(defaultPrediction!=validSet$default) \* 100

## [1] 2.613333

#### (c) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

defaultSplit = sample.split(default$default, SplitRatio = 0.25)  
trainingSet <- subset(default, defaultSplit == TRUE)  
validSet <- subset(default, defaultSplit == FALSE)  
defaultLRModel <- glm(default ~ income + balance, data = trainingSet, family = "binomial")  
defaultProbability=predict(defaultLRModel, newdata = validSet, type="response")  
defaultPrediction <- ifelse(defaultProbability > 0.5, 1, 0)  
table(defaultPrediction,validSet$default)

##   
## defaultPrediction 0 1  
## 0 7226 172  
## 1 24 78

mean(defaultPrediction!=validSet$default) \* 100

## [1] 2.613333

defaultSplit = sample.split(default$default, SplitRatio = 0.5)  
trainingSet <- subset(default, defaultSplit == TRUE)  
validSet <- subset(default, defaultSplit == FALSE)  
defaultLRModel <- glm(default ~ income + balance, data = trainingSet, family = "binomial")  
defaultProbability=predict(defaultLRModel, newdata = validSet, type="response")  
defaultPrediction <- ifelse(defaultProbability > 0.5, 1, 0)  
table(defaultPrediction,validSet$default)

##   
## defaultPrediction 0 1  
## 0 4815 119  
## 1 18 48

mean(defaultPrediction!=validSet$default) \* 100

## [1] 2.74

defaultSplit = sample.split(default$default, SplitRatio = 0.75)  
trainingSet <- subset(default, defaultSplit == TRUE)  
validSet <- subset(default, defaultSplit == FALSE)  
defaultLRModel <- glm(default ~ income + balance, data = trainingSet, family = "binomial")  
defaultProbability=predict(defaultLRModel, newdata = validSet, type="response")  
defaultPrediction <- ifelse(defaultProbability > 0.5, 1, 0)  
table(defaultPrediction,validSet$default)

##   
## defaultPrediction 0 1  
## 0 2405 62  
## 1 12 21

mean(defaultPrediction!=validSet$default) \* 100

## [1] 2.96

The misclassification rates among the different splits were overall small and never went past 3% showing that the model is effective at predicting defaults but as the split of data is changed from a small training set and large validation set to a large training set to a small validation set the rate increases overtime.

#### (d) Now consider a logistic regression model that predicts the probability of default using income, balance, and a dummy variable for student. Estimate the test error for this model using the validation set approach. Comment on whether or not including a dummy variable for student leads to a reduction in the test error rate.

defaultSplit = sample.split(default$default, SplitRatio = 0.625)  
trainingSet <- subset(default, defaultSplit == TRUE)  
validSet <- subset(default, defaultSplit == FALSE)  
defaultLRModel <- glm(default ~ income + balance + student, data = trainingSet, family = "binomial")  
defaultProbability=predict(defaultLRModel, newdata = validSet, type="response")  
defaultPrediction <- ifelse(defaultProbability > 0.5, 1, 0)  
table(defaultPrediction,validSet$default)

##   
## defaultPrediction 0 1  
## 0 3612 84  
## 1 13 41

mean(defaultPrediction!=validSet$default) \* 100

## [1] 2.586667

Including the student coefficient does reduce the misclassification rate, without student the rate is approximately 2.613% and decreases to 2.587% with student included

### 6. We continue to consider the use of a logistic regression model to predict the probability of default using income and balance on the Default data set. In particular, we will now compute estimates for the standard errors of the income and balance logistic regression coefficients in two different ways: (1) using the bootstrap, and (2) using the standard formula for computing the standard errors in the glm() function. Do not forget to set a random seed before beginning your analysis.

#### (a) Using the summary() and glm() functions, determine the estimated standard errors for the coefficients associated with income and balance in a multiple logistic regression model that uses both predictors.

library(boot)  
set.seed(1)  
defaultSDE <- glm(default ~ income + balance, data = default, family = binomial)  
summary(defaultSDE)$coefficients[2:3,]

## Estimate Std. Error z value Pr(>|z|)  
## income 2.080898e-05 4.985167e-06 4.174178 2.990638e-05  
## balance 5.647103e-03 2.273731e-04 24.836280 3.638120e-136

#### (b) Write a function, boot.fn(), that takes as input the Default data set as well as an index of the observations, and that outputs the coefficient estimates for income and balance in the multiple logistic regression model.

boot.fn <- function(data, index)   
{  
 bootData <- data[index, ]  
 bootFit <- glm(default ~ income + balance, data = bootData, family = binomial)  
 return(summary(bootFit)$coefficients[2:3,1])  
}

#### (c) Use the boot() function together with your boot.fn() function to estimate the standard errors of the logistic regression coefficients for income and balance.

boot(default, boot.fn, R=100)

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = default, statistic = boot.fn, R = 100)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 2.080898e-05 -3.993598e-07 4.186088e-06  
## t2\* 5.647103e-03 -4.116657e-06 2.226242e-04

#### (d) Comment on the estimated standard errors obtained using the glm() function and using your bootstrap function.

The standard error for the income is smaller when using bootstrap compared to the glm standard error. The same could also be said when looking at the standard error for balance between the two methods with the bootstrap standard error being smaller than that of the glm.

### 7. In Sections 5.3.2 and 5.3.3, we saw that the cv.glm() function can be used in order to compute the LOOCV test error estimate. Alternatively, one could compute those quantities using just the glm() and predict.glm() functions, and a for loop. You will now take this approach in order to compute the LOOCV error for a simple logistic regression model on the Weekly data set. Recall that in the context of classification problems, the LOOCV error is given in (5.4).

#### (a) Fit a logistic regression model that predicts Direction using Lag1 and Lag2.

weekly <- read.csv("DataSets/Weekly.csv")  
weekly$Direction <- as.factor(weekly$Direction)  
weeklyFit <- glm(Direction ~ Lag1 + Lag2, data = weekly, family = binomial)  
summary(weeklyFit)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = weekly)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.22122 0.06147 3.599 0.000319 \*\*\*  
## Lag1 -0.03872 0.02622 -1.477 0.139672   
## Lag2 0.06025 0.02655 2.270 0.023232 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1496.2 on 1088 degrees of freedom  
## Residual deviance: 1488.2 on 1086 degrees of freedom  
## AIC: 1494.2  
##   
## Number of Fisher Scoring iterations: 4

#### (b) Fit a logistic regression model that predicts Direction using Lag1 and Lag2 using all but the first observation.

weeklyFitEx <- glm(Direction ~ Lag1 + Lag2, data = weekly[-1, ], family = binomial)  
summary(weeklyFitEx)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = weekly[-1,   
## ])  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.22324 0.06150 3.630 0.000283 \*\*\*  
## Lag1 -0.03843 0.02622 -1.466 0.142683   
## Lag2 0.06085 0.02656 2.291 0.021971 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1494.6 on 1087 degrees of freedom  
## Residual deviance: 1486.5 on 1085 degrees of freedom  
## AIC: 1492.5  
##   
## Number of Fisher Scoring iterations: 4

#### (c) Use the model from (b) to predict the direction of the first observation. You can do this by predicting that the first observation will go up if P(Direction = “Up”|Lag1, Lag2) > 0.5. Was this observation correctly classified?

weeklyProbability <- predict(weeklyFitEx, data = weekly[1, ], type = "response")  
directionPrediction <- ifelse(weeklyProbability > 0.5, "Up", "Down")  
actualDirection <- weekly$Direction[1]  
classification <- (directionPrediction == actualDirection)

#### (d) Write a for loop from i = 1 to i = n, where n is the number of observations in the data set, that performs each of the following steps:

n <- nrow(weekly)  
error <- numeric(n)  
  
for (i in 1:n)  
{  
 # i. Fit a logistic regression model using all but the ith observation to predict Direction using Lag1 and Lag2.  
 weeklyForFit <- glm(Direction ~ Lag1 + Lag2, data = weekly[-i, ], family = binomial)  
 # ii. Compute the posterior probability of the market moving up for the ith observation.  
 forFitProb <- predict(weeklyForFit, newdata = weekly[i, ], type = "response")  
 # iii. Use the posterior probability for the ith observation in order to predict whether or not the market moves up.  
 forFitPredict <- ifelse(forFitProb > 0.5, TRUE, FALSE)   
 # iv. Determine whether or not an error was made in predicting the direction for the ith observation. If an error was made,then indicate this as a 1, and indicate it as a 0.  
 forFitDirection <- weekly$Direction[i] == "Up"  
 error[i] <- ifelse(forFitPredict != forFitDirection, 1, 0)  
}

#### (e) Take the average of the n numbers obtained in (d)iv in order to obtain the LOOCV estimate for the test error. Comment on the results.

mean(error)

## [1] 0.4499541

After taking the average from the results found in (d) the estimated test error for the LOOCV is approximately 44.99% or 45%