Chapter 5 HW

Fabiani Rafael

Conceptual Questions

Exercise 3: We now review k-fold cross-validation.

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

- Given a data set of say n observations the set would then be randomly partitioned into k equal-sized folds. For each fold therein, the model is trained on the remaining k-1 folds and subsequently validated on the current fold. The process is repeated k times, with each fold being used as the validation set once. The validation error, such as the mean squared error MSE, is then computed using the held-out set. The average of these error values yields the overall cross-validation error.

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

1. the validation set approach?

   - Advantages:
     - More data is used for training, as each observation is used in the training set k-1 times.

     - can provide a more reliable estimate of the model's performance by averaging over multiple folds.

   - Disadvantages:
     - More computationally expensive, as the model needs to be trained k times. That is to say as the data set increases the computational costs increase.

     - The choice of k can affect the results; too small or too large k can lead to biased estimates.

2. LOOCV (leave-one-out cross-validation)?

   - Advantages:
     - Each training set is almost the same size as the original data which can yield less biased estimates of the model's performance.

     - Useful for smaller data sets where every observation is valuable.

   - Disadvantages:
     - computationally expensive, as the model needs to be trained n times.

     - High variance in the estimates, especially for small data sets, as each training set is very similar.

Applied Questions

Exercise 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.

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

```r
# fit logistic regression model & summary
log_mdl <- glm(default ~ income + balance, data = Default, family = binomial)
summary(log_mdl)
```

```
##
## 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) 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.

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

(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.

(iv) Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

```r
set.seed(216)

# split 70 - 30
train_index <- sample(1:nrow(Default), nrow(Default) * 0.7)

# create the training and validation sets
train_set <- Default[train_index, ]
validtn_set <- Default[-train_index, ]

# fit the logistic regression model & summary
log_mdl <- glm(default ~ income + balance, data = train_set, family = binomial)

# obtain the predicted probabilities
predictd_probs <- predict(log_mdl, newdata = validtn_set, type = "response")
```

```r
# classify the individuals based on the predicted probabilities
predictd_classes <- ifelse(predictd_probs > 0.5, "Yes", "No")

# compute the validatn set error
validtn_error <- mean(predictd_classes != validtn_set$default)
validtn_error
```

```
## [1] 0.02566667
```

(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.

```r
set.seed(216)
# init vector to store validation errors
validtn_errs <- numeric(3)

for (i in 1:3) {
  # split 70-30
  train_indx <- sample(1:nrow(Default), nrow(Default) * 0.7)
  train_set <- Default[train_indx, ]
  validtn_set <- Default[-train_indx, ]

  # fit logistic reg model
  log_mdl <- glm(default ~ income + balance, data = train_set, family = binomial)

  # get pred prob & classes
  pred_prob <- predict(log_mdl, newdata = validtn_set, type = "response")
  pred_clss <- ifelse(pred_prob > 0.5, "Yes", "No")

  # compute validtn set error
  validtn_errs[i] <- mean(pred_clss != validtn_set$default)
}

# show valid errors
validtn_errs
```

```
## [1] 0.02566667 0.02933333 0.03066667
```

(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.

```r
# fit the logistic regression model & summary
log_mdl_student <- glm(default ~ income + balance + student, data = train_set, family = binomial)

# get predicted probabilities
predictd_probs_student <- predict(log_mdl_student, newdata = validtn_set, type = "response")

# classify the individuals on the predicted probabilities
predictd_classes_student <- ifelse(predictd_probs_student > 0.5, "Yes", "No")


# get validation set error
validtn_error_student <- mean(predictd_classes_student != validtn_set$default)
validtn_error_student
```

```
## [1] 0.03066667
```

- The test error for the model with the dummy variable for student (2.5

Exercise 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.

```r
# fit the logistic regression model & summary
log_mdl <- glm(default ~ income + balance, data = Default, family = binomial)
summary(log_mdl)
```

```
##
## 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
```

```r
# extract the standard errors
std_errors <- summary(log_mdl)$coefficients[, "Std. Error"]
std_errors
```

```
##  (Intercept)       income      balance
## 4.347564e-01 4.985167e-06 2.273731e-04
```

(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.

```r
boot.fn <- function(data, index) {# return income & balance
  log_mdl <- glm(default ~ income + balance, data = data[index, ], family = binomial)
  return(coef(log_mdl)[c("income", "balance")])
}
```

(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.

```
set.seed(216)
# bootstrap with 1k reps income & balance
boot_results <- boot(
  data = Default,
  statistic = boot.fn,
  R = 1000
)

# std err for income & balance
boot_se_income <- sd(boot_results$t[, 1])
boot_se_balance <- sd(boot_results$t[, 2])

cat("Bootstrap Standard Errors:\n",
    "Income:", boot_se_income, "\n",
    "Balance:", boot_se_balance)
```

```
## Bootstrap Standard Errors:
##  Income: 4.772267e-06
##  Balance: 0.0002335733
```

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

- The standard errors obtained from the glm() function are 0.000005 and 0.000234 for income and balance, respectively. The bootstrap standard errors are 4.772267e-06 and 0.0002335733, which are very similar to the glm() estimates. This suggests that the bootstrap method provides a reliable estimate of the standard errors for the coefficients in this logistic regression model.

Exercise 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.

```
# fit the logistic regression model & summary
log_mdl_weekly <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial)
summary(log_mdl_weekly)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.623  -1.261   1.001   1.083   1.506
##
## 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.

```
# fit the logistic regression model & summary
log_mdl_weekly_1 <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = binomial)
summary(log_mdl_weekly_1)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly[-1,
##     ])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6258  -1.2617   0.9999   1.0819   1.5071
##
## 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 $P(Direction = \text{"Up"} \,|\, Lag1, Lag2) > 0.5$. Was this observation correctly classified?

```
# get the predicted probabilities
predictd_probs_weekly_1 <- predict(log_mdl_weekly_1, newdata = Weekly[1, ], type = "response")
# classify the individual based on the predicted probabilities
predictd_class_weekly_1 <- ifelse(predictd_probs_weekly_1 > 0.5, "Up", "Down")

# check if the prediction is correct
correct_class_weekly_1 <- ifelse(predictd_class_weekly_1 == Weekly$Direction[1], 1, 0)
cat("Predicted class for the first observation:", predictd_class_weekly_1, "\n")
```

```
## Predicted class for the first observation: Up
```

```r
cat("Correct class for the first observation:", Weekly$Direction[1], "\n")
```

```
## Correct class for the first observation: 1
```

```r
cat("Was the prediction correct?", ifelse(correct_class_weekly_1 == 1, "Yes", "No"), "\n")
```

```
## Was the prediction correct? No
```

- The first observation was predicted to be "Up" with a probability of 0.5, but the actual direction was "Down" hemce the prediction was incorrect.

(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:

(i) Fit a logistic regression model using all but the ith observation to predict Direction using Lag1 and Lag2.

(ii) Compute the posterior probability of the market moving up for the ith observation.

(iii) Use the posterior probability for the ith observation in order to predict whether or not the market moves up.

(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 otherwise indicate it as a 0.

```r
n <- nrow(Weekly)
# init vector to store th errs
errors <- numeric(n)

for (i in 1:n) {# iterating through each obs
  # fit log reg model leaving outt the ith obs
  log_mdl_weekly_i <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = binomial)

  # get the posterior probability of the market moving up for the ith observation
  predictd_probs_weekly_i <- predict(log_mdl_weekly_i, newdata = Weekly[i, ], type = "response")

  # classfy on the pred prob
  predictd_class_weekly_i <- ifelse(predictd_probs_weekly_i > 0.5, "Up", "Down")

  #  whether or not an error was made in predicting the direction for the ith observation
  errors[i] <- ifelse(predictd_class_weekly_i != Weekly$Direction[i], 1, 0)
}
#  LOOCV error
loocv_error <- mean(errors)
cat("LOOCV Error Rate:", loocv_error, "\n")
```

```
## LOOCV Error Rate: 0.4499541
```

(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.

```r
#   LOOCV err
loocv_error <- mean(errors)
cat("LOOCV Error Rate:", loocv_error, "\n")
```

```
## LOOCV Error Rate: 0.4499541
```

- The LOOCV error rate is $0.449 \approx 0.45$ i.e the model misclassifies 45% of the observations in the data set. This is a relatively high error rate, indicating that the model may not be performing well in predicting the direction of the market based on Lag1 and Lag2. Moreover the error rate makes it clear that the

model is not very effective in predicting the direction of the market, as it misclassifies nearly half of the observations. It is technically better than pure by chance but not by much.