Exercises Week 4

Julian D. Karch

This week we will work with the `student-mat.csv` data set available on Brightspace. It consists of information related to the performance of 394 students on a mathematics exam. The dependent variable in this data set is whether a student scored above the median and is created in the code below. The three independent variables we will focus on are "studytime", "schoolsup", and "romantic".

The "studytime" variable represents the number of hours per week that the student studies. The "schoolsup" variable indicates whether the student received extra educational support from the school, while the "romantic" variable indicates whether the student is in a romantic relationship.

1 k-fold Cross-validation Code

Complete the below code such that it performs k-fold cross-validation of a logistic regression model. You have to complete lines 29, 32, and 33.

```
1  # Load data
2  library(readr)
3  student_mat <- read_delim("student-mat.csv",
4      delim = ";", escape_double = FALSE, trim_ws = TRUE)
5  student_mat$y <- as.factor(student_mat$G1>11)
6  # Set the number of folds
7  k <- 5
8
9  # Calculate the size of each fold
10 fold_size <- floor(nrow(student_mat) / k)
11
12 # Define the folds
13 folds <- rep(1:k, each = fold_size)
14 #in case sample size n is not dividable by number of folds k
15 if (length(folds) < nrow(student_mat)) {
16   folds[(length(folds) + 1):nrow(student_mat)] <- sample(1:k, nrow(student_mat) - length(folds))
17 }
18
19 # Initialize variables for storing results
20 predictions <- logical(nrow(student_mat))
21
22 # Shuffle the folds (equivalent to shuffling the data)
23 set.seed(123)
24 folds <- sample(folds)
25
26 # Perform k-fold cross-validation
27 for (i in 1:k) {
28   # Determine the test indices for the current fold
29   test_indices <- ...
```

```r
  # Create the training and testing sets
  train_set <- ...
  test_set <- ...


  # Fit the model and make predictions
  fitted_model <- glm(y ~ studytime +  schoolsup + romantic, data = train_set, family = "binomial")
  probs <- predict(fitted_model, test_set, type = "response")
  classes <- rep(FALSE, nrow(test_set))
  classes[probs > .5] <- TRUE
  predictions[test_indices] <- classes
}

# Calculate the overall mean squared error
accuracy <-  mean(predictions == student_mat$y)
print(accuracy)
```

2 k-fold Cross-validation + Test Set

We now want to try out a few different models to see which one performs best. Specifically, we want to select the best k for k-nearest-neighbor and estimate the accuracy of the kNN with the optimal k. We will use the cross-validation + test set approach discussed in the lecture to do this.

First, split the data into a training and test set using an 80-20 split, where 80% of the data goes into the training set. On the training set, use 10-fold cross-validation to find the best value of k among $1, 3, 5, 7, 9$. For the cross-validation, you don't need to copy the code above. Instead, you can use the cross-validation function included in the `caret` package.

The code below demonstrates how it works. Note that this code also standardizes all features, which is recommended for kNN (as discussed in last week's reading material). Do not yet test your selected model on the test set. We will do this in the next exercise!

```r
# Define the training control
train_control <- trainControl(method = "cv", number = 5)

# Define the knn model with candidate ks
knn_model <- train(y ~ studytime +  schoolsup + romantic, data = train_set,
                   method = "knn",
                   tuneGrid = expand.grid(k = c(1,3,5,7,9)),
                   trControl = train_control,
                   preProcess = c("center","scale"))
```

3 Multiple Different Models

We now want to see whether LDA might provide better predictions. Unfortunately, the `train` function does not support more than one method. However, a simple work-around was recommended in the reading material (see the third item on the reading list). Use this approach to compare the performance of the optimal kNN model with LDA. Select the model with the highest accuracy and estimate its accuracy using the test set.

4 Cross-Validation - The Wrong Way

We now focus on a fake regression data set, which we generate as follows

```
library(MASS)
set.seed(123)
n <- 200
p <- 10^4
X <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- rnorm(n)
data_set <- as.data.frame(cbind(y,X))
names(data_set) <- c("y", paste0("x",1:p))
```

There is thus no relationship between the predictors X and the outcome Y. Thus, no model will have a lower true MSE than 1, as this is the variance of Y.

- Calculate the correlation between the predictors X and the outcome y.
- Select the 5 variables most highly correlated with y
- Define a linear regression model with only those variables as predictors.
- Estimate its accuracy using cross-validation. What do you notice about the performance?
- What is causing this?