Homework: Ch 04

STAT 4510/7510

Due Tuesday, March 1, 11:59 pm

Instructions: Please list your name and student number clearly. In order to receive credit for a problem, your solution must show sufficient detail so that the grader can determine how you obtained your answer.

Submit a single pdf file for your final outcome. All `R` code should be included, as well as all output produced. Upload your work to the Canvas course site.

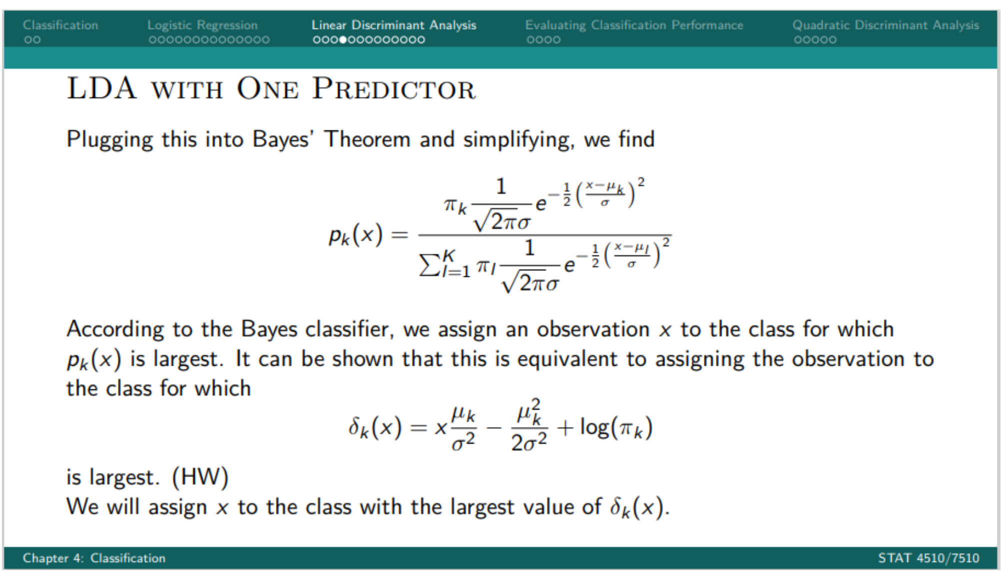
Chapter 4 Lab Exercise

I strongly suggest you to read the textbook 4.6 Lab: Logistic Regression, LDA, QDA, and KNN, found on pages 154 - 167. The data set files `Smarket.csv`, `Caravan.csv` and the `R` code used in this Lab can be found in Canvas. Just run each line of the code and see what happens.

- You don't have to submit your work for this, but to solve the following homework problems, you may want to do this first.

Problem 1

From the following description of LDA, show that classifying an observation to the class for which $p_k(x)$ is largest is equivalent to classifying an observation to the class for which $\delta_k(x)$ is largest.



Problem 2

(a) An odds of occurring fraud credit card transactions is known as 0.12. What is the probability that a credit card transaction is fraud?

(b) The probability that the tomorrow's stock price of a company increases is known as 0.52. What are the odds of tomorrow's stock price of the company increasing?

Problem 3

In this question, we will re-use the `Auto` data set used in the regression problem from the previous homework assignment, but now for the classification problem. We use classification models with `origin` as the response variable (1: American vs. 3: Japanese) and `mpg`, `cylinders`, `horsepower`, `acceleration`, and `year` as predictor variables. Read the data set and manipulate it as follows.

```
Auto <- read.csv("Auto.csv", na.strings ="?") # With the option, R recognizes ? as NA.
Auto <- na.omit(Auto) # Remove data rows including NA.
Auto.class <- Auto[,c(1,2,4,6,7,8)] # Keep only mpg, cylinders, horsepower,
# acceleration, year, and origin
Auto.class <- Auto.class[Auto.class$origin != '2',] # Only keep data points where
# origin is either 1 (American) or 3 (Japanese).
Auto.class$origin <- as.factor(Auto.class$origin) # Coerce the type of origin into factor
summary(Auto.class)
```

```
##       mpg          cylinders       horsepower      acceleration
##  Min.   : 9.00   Min.   :3.000   Min.   : 52.00   Min.   : 8.00
##  1st Qu.:16.00   1st Qu.:4.000   1st Qu.: 82.75   1st Qu.:13.50
##  Median :20.90   Median :6.000   Median : 97.00   Median :15.40
##  Mean   :22.57   Mean   :5.747   Mean   :109.49   Mean   :15.28
##  3rd Qu.:28.00   3rd Qu.:8.000   3rd Qu.:140.00   3rd Qu.:17.00
##  Max.   :46.60   Max.   :8.000   Max.   :230.00   Max.   :22.20
##       year       origin
##  Min.   :70.00   1:245
##  1st Qu.:73.00   3: 79
##  Median :76.00
##  Mean   :76.04
##  3rd Qu.:79.00
##  Max.   :82.00
```

(a) Use the logistic regression with `origin` as the response and the other five variables as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

(b) Interpret the estimated regression coefficient of `cylinders` and `year`.

(c) Predict the class labels of the observations in the training data by using the default threshold 0.5. Display the confusion matrix and overall fraction of correct predictions.

(d) Perform LDA on the same data `Auto.class`, and show the fitted object to print the results. When you perform LDA, let the model estimate priors of each class from the training data set.

(e) The `plot()` function produces plots of the linear discriminant scores for each of the training observations. Using this function, investigate how two classes are distinguished in terms of the discriminant score.

(f) Predict the class labels of the observations in the training data by using the fitted LDA model. Produce the confusion matrix from LDA prediction.

(g) Perform QDA on the same data `Auto.class`, and show the fitted object to print the results. When you perform QDA, let the model estimate priors of each class from the training data set.

(h) Predict the class labels of the observations in the training data by using the fitted QDA model. Produce the confusion matrix from QDA prediction.

(i) Compare prediction outcomes of LDA and QDA with respect to the following metrics.

   1. Accuracy (overall fraction of correct predictions)
   2. Sensitivity
   3. Specificity

4. Precision

(j) (7510 students only) Using the predicted probability outcomes from the QDA model, construct the ROC curve. Follow the description below.

1. The following function is created to compute the true positive rate (sensitivity) and false positive rate (1 - specificity) from the fitted QDA model, given a threshold value. Complete lines of `sensitivity <-` and `specificity <-`.

```
roc.metric <- function (trueLabel, probPositive, threshold) {
  # Description of input arguments:
  # trueLabel: a vector (factor) containing the true class labels of data points
  # probPositive: a vector (numeric) containing the predicted class labels of
  #               data points
  # threshold: a threshold value (scalar) to determine the predicted class labels

  class <- levels(trueLabel)
  # Extract two labels used in trueLabel. The first element is treated as
  # the negative class, and the second is treated as the positive class.

  predLabel <- ifelse(probPositive > threshold, class[2], class[1])
  # For each element of probPositive, if it is greater than threshold value,
  # assign positive class. Otherwise, assign negative class.

  sensitivity <-
  specificity <-

  return(c(sensitivity, 1-specificity))
  # Provide a vector containing sensitivity and 1-specificity as an output
}
```

2. Validate the above function by running it with `threshold = 0.5` and other input arguments being appropriately specified (`trueLabel` should be the class label from the training data, and `probPositive` should be extracted from QDA output). You should be able to obtain the same values of sensitivity and 1-specificity that you obtained from the outcome of (h).

3. Create a vector `thresholdValues` that contains different threshold values as follows. Print the `thresholdValues`.

```
thresholdValues <- seq(0, 1, by=0.1)
```

- We want to save the output of `roc.metric()` evaluated with different threshold values included in `thresholdValues`. To do this, we need to prepare the object in which the outcomes are stored. Create a matrix `roc.metric.out` with 3 columns as follows.

```
roc.metric.out <- matrix(NA, nrow=length(thresholdValues), ncol=3)
colnames(roc.metric.out) <- c("threshold", "sensitivity", "1-specificity")
```

- As being indicated by the column names, we will save a threshold value and corresponding sensitivity and 1-specificity at the first, second, and third column, respectively.

4. Using a `for` loop, we will run `roc.metric()` function as many times as the number of threshold values. For each run, we will save appropriate outcomes to `roc.metric.out`. Complete the line of `roc.metric.out[i,2:3] <-` from the following code, and show the result of `roc.metric.out`.

```
for (i in 1:length(thresholdValues)) {
  roc.metric.out[i,1] <- thresholdValues[i]
  roc.metric.out[i,2:3] <-
}
```

5. Using the function `plot()`, draw the ROC curve. To do this, you can specify the 1-specificity as the x-axis and sensitivity as the y-axis. If you use the option `type = "o"`, you can obtain the curve connected through each other as well as the points.