Computer Assignment 4 - Classification

Machine Learning, Fall 2021

YOUR NAME

Bayes Rule and Univariate Normal Simulations

As mentioned in our first few Computing Assignments, `R` can simulate a number of distributions, including the normal distribution:

$$\mathbb{W} \sim \mathcal{N}(-2, 1.2) \qquad \& \qquad \mathbb{V} \sim \mathcal{N}(2, 1.2).$$

For this next exercise, we are going to simulate 300 observations from $\mathbb{W}$ and 200 observations from $\mathbb{V}$, and create another variable $\mathbb{Y}$ that classifies from which normal distribution each observation came. Indeed,

```r
W_obs = rnorm(300, mean = -2, sd = 1.2)
V_obs = rnorm(200, mean = 2, sd = 1.2)
Y_class = c(rep(0, times = length(W_obs)), rep(1, times = length(V_obs)))
train_data = data.frame(X = c(W_obs, V_obs), Y = Y_class)
```

Since we have specified ourselves the model for our data (here: two different normals), we can assess the performance of any classification technique we use on this data. We will illustrate this by calculating the Bayes rule for our `train_data`.

Bayes Rule

Recall that the Bayes rule provides a prediction of the value of Y when only the value of X is available. To do this, it is necessary to find how likely it is that $Y = 1$ (and $Y = 0$) given $X = x$. In this case, since X is continuous, this is represented by:

$$\eta(x) = P(Y = 1 | X = x) = \frac{f_{x,y}(x, 1)}{f_x(x)} = \frac{\pi_1 f_{x|y}(x|1)}{f_x(x)},$$

and, analogously:

$$1 - \eta(x) = P(Y = 0 | X = x) = \frac{\pi_0 f_{x|y}(x|0)}{f_x(x)}.$$

Once we know both conditional probabilities, we can find the Bayes rule. It consists of always predicting the value of Y that is the most likely. In general, we say that:

$$\phi^*(x) = \begin{cases} 1 & \text{if } \eta(x) > 1 - \eta(x) \\ 0 & \text{otherwise} \end{cases} = \begin{cases} 1 & \text{if } \eta(x) > \frac{1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

Since both values are being divided by the same constant $f_x(x)$, we can reduce our decision as what follows:

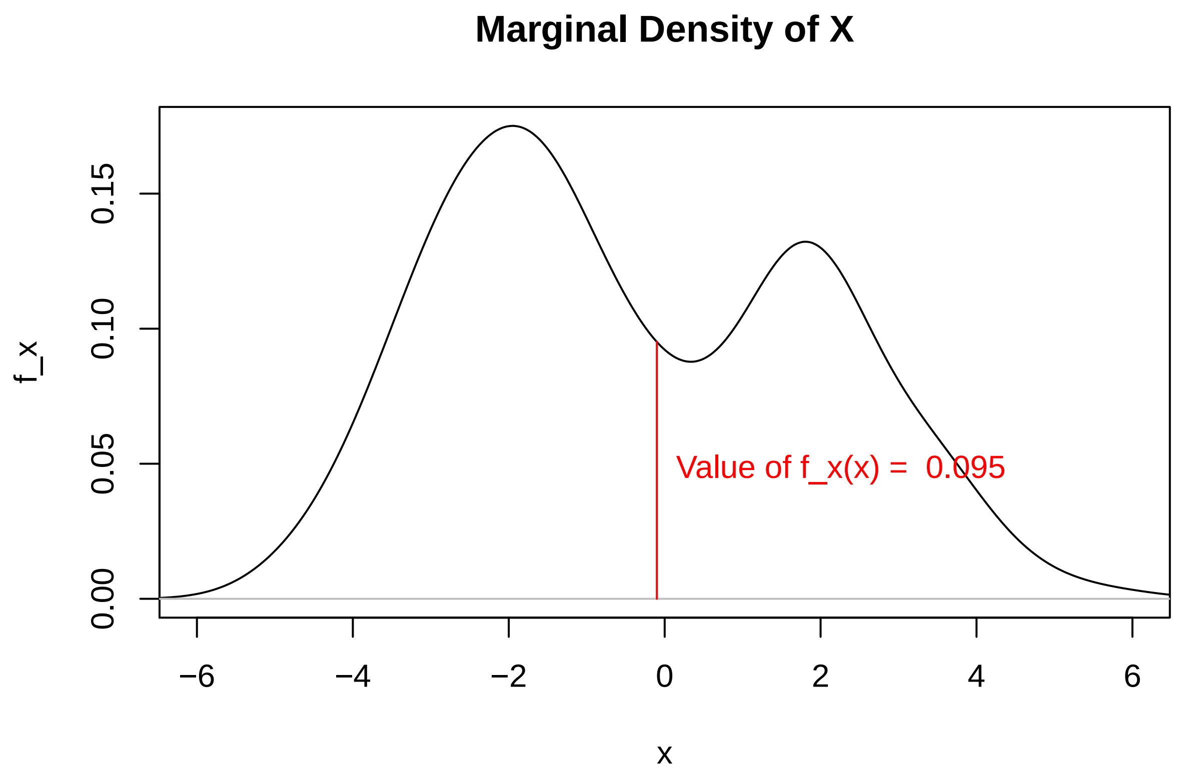
$$\phi^*(x) = \begin{cases} 1 & \text{if } \pi_1 f_{x|y}(x|1) > \pi_0 f_{x|y}(x|0) \\ 0 & \text{otherwise.} \end{cases}$$

In the following chunk, we provide an example of how to find an estimate of the density $f_x(x)$ on the value $x = -0.1$ based on our training data.

```r
# Example of PDF estimation
x_obs = -0.1
full_density = density(train_data$X)     ## Find an estimate of the entire density function f_x.
index_of_density = sum(full_density$x <= x_obs)

if(index_of_density == 0){
  pdf_value_of_x_obs = 0                 ## If all observations are above x_obs density = 0
} else {
  pdf_value_of_x_obs = full_density$y[index_of_density]
}

## To visualize what was done.
plot(x = full_density$x,                 ## Plot of the density f_x.
     y = full_density$y,
     type = "l",
     xlim = c(-6,6),
     main = "Marginal Density of X",
     xlab = "x",
     ylab = "f_x")
abline(h = 0, col = "grey")              ## Drawing the x axis.
segments(x0 = x_obs, y0 = 0,             ## Drawing the density on x = -0.1
         x1 = x_obs, y1 = pdf_value_of_x_obs,
         col = "red")
text(x = x_obs, y = pdf_value_of_x_obs/2, ## Adding nice label.
     labels = paste("Value of f_x(x) = ", round(pdf_value_of_x_obs, digits = 3)),
     col = "red", pos = 4)
```



Question 1:

1. What are the values of π_1 and π_0?
2. Based on the code previously provided, find an estimation of the conditional densities $f_{x|y}(x|1)$ and $f_{x|y}(x|0)$ for the value $x = 1.3$.

3. With the newly found values of $f_{x|y}(x|1)$ and $f_{x|y}(x|0)$ on $x = 1.3$, find the value of $\pi_1 f_{x|y}(x|1)$ and $\pi_0 f_{x|y}(x|0)$.
4. Which of the two weighted densities is larger? What would be the Bayes rule on $x = 1.3$?

```
## Find pi1 and pi0.

## Use this value of X:
x_obs = 1.3

## Find the density of X given Y = 0

## Find the density of X given Y = 1

## Do comparisson of the products.
```

ANSWERS

1. HERE.

2. HERE.
3. HERE.
4. HERE.

Finding the Bayes Rule

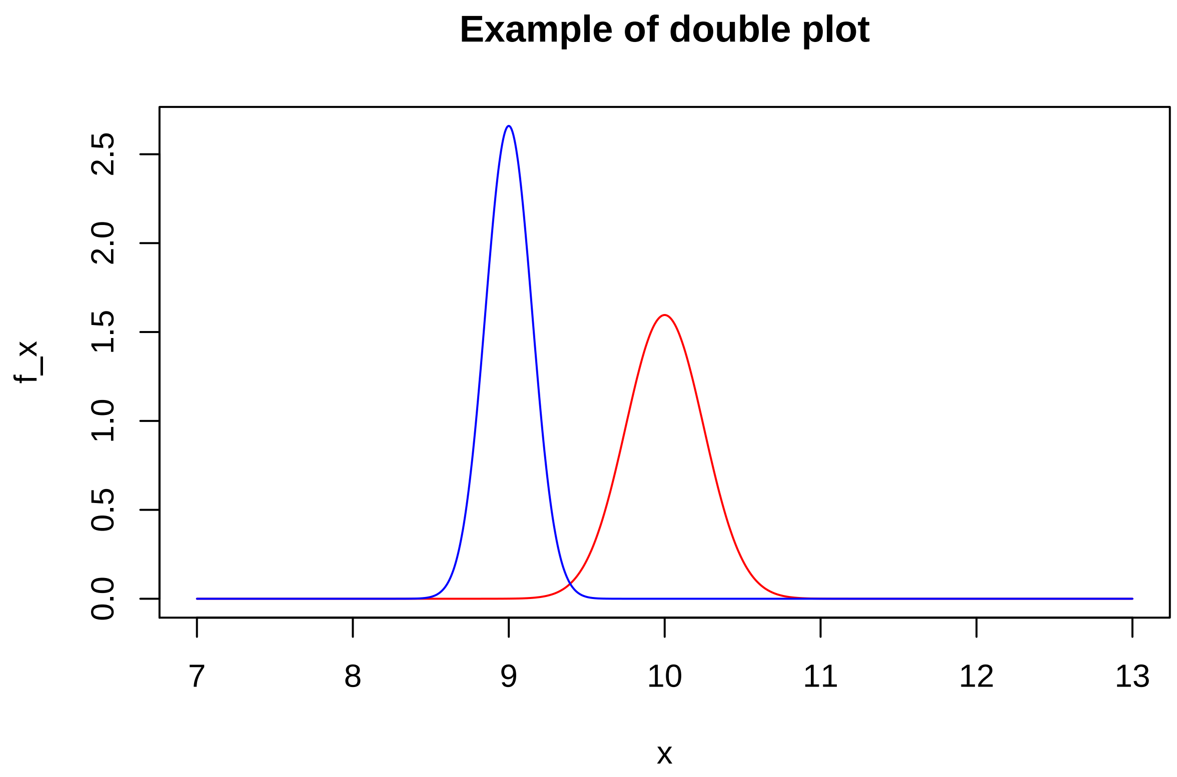
Once we know how to find the Bayes rule for a single value of $X = x$, we can utilize a `for` loop to find the Bayes rule for all of the values of x for a certain range of x. Since the distributions of $X|Y = 1$ and $X|Y = 0$ are normal, the decision boundary will consist of a single point. In the next question, the goal is to find the Bayes rule, and find the decision boundary.

Recall that the Bayes rule is determined by the values of the weighted densities $\pi_1 f_{x|y}(x|1)$ and $\pi_0 f_{x|y}(x|0)$. More specifically, we know that $\phi^*(x) = 1$ if $\pi_1 f_{x|y}(x|1) > \pi_0 f_{x|y}(x|0)$ and $\phi^*(x) = 0$ if $\pi_0 f_{x|y}(x|0) > \pi_1 f_{x|y}(x|1)$. Therefore, the a useful plot for discussing the Bayes rule is to simultaneously visualize the two weigthed densities.

The following chunk shows how to create a plot of 2 functions over a given range of values. Use this as a basis for your answer in the next question.

```r
## The functions we will consider are normal densities:
x = seq(from = 7, to = 13, length.out = 1000)
y1 = dnorm(x =x, mean = 10, sd = 0.25)
y2 = dnorm(x =x, mean = 9, sd = 0.15)

plot(x = x,
     y = y1,
     type = "l",
     xlim = c(7,13),
     ylim = c(0, max(y1,y2)),
     main = "Example of double plot",
     xlab = "x",
     ylab = "f_x",
     col = "red")
lines(x = x,
      y = y2,
      col = "blue")
```



Question 2:

1. Create a vector of uniformly spaced values between -6 and 6 of size 1000. (Hint: use `seq`)
2. Modify your code from Q1 to find the weighted densities $\pi_1 f_{x|y}(x|1)$ and $\pi_0 f_{x|y}(x|0)$ for all the values of x between -6 and 6.
3. Find the Bayes rule for all values of x between -6 and 6. What is the decision boundary?
4. Create a plot that contains both weighted densities. Use a vertical line to mark where the decision boundary is.
5. Comment on the position of the decision boundary. Is it located in a special location? If so, provide an intuitive reason for why is it located there.

```
## Create fine grid of values between -6 and 6.

## Find the weigthed conditional densities for all values between -6 and 6.

## Create a plot with both weighted densities and represent the Bayes rule
```

ANSWERS

1. HERE.
2. HERE.
3. HERE.
4. HERE.
5. HERE.

Bayes Rule and Estimation Error.

The Bayes rule can be used as an "educated guess" of what the value of Y is when only the value of $X = x$ is available. Since ultimately the value of Y is random, there will be an intrinsic error in our predictions. To measure the quality of our estimation, we can find the "risk" of our estimate. This is the percentage of missclassified observations. In general, the smaller the risk is, the better a classifier is.

Question 3:

1. Modify the code used in Q2, to find the Bayes rule on the X values from our data: $x_1, x_2, ..., x_{500}$. The result must be a vector $\phi^* = (\phi_1^*, \phi_2^*, ..., \phi_{500}^*)$.
2. Create a contingency table, where you contrast the actual values of $y_1, ..., y_{500}$ from our data, with the predictions of the Bayes rule. What is the percentage of missclassified observations (the risk)?
3. Is there any misclassified observation with $x_i \geqslant 1$? Is there any missclassified observation with $x_i \leqslant -1$?
4. Find the maximum and minimum values of x that were misclassified. Are most missclassified observations near the decision boundary? Why?

```
YOUR CODE HERE
```

ANSWERS:

1.

2.

3.

4.

k-Nearest Neighbors

Question 4:

1. Fit a k nearest neighbors model for all $k = 1, 2, ..., 10$. For each model find the risk of each model. Are there any trends?
2. Generate a plot where the x-axis is the numbers $k = 1, 2, ..., 10$, and the y-axis is the risk of each model. Comment on the results.
3. Why does `k = 1` do so well? What is it doing that gives it such great performance?

```
library(class)

# Example for k = 1
train_data_classifiers = as.factor(train_data$Y)
train_data_observations = data.frame(train_data$X)
knn.1 <-  knn(train_data_observations, train_data_observations, cl = train_data_classifiers, k = 5)
R_knn_1 = 100 * sum(train_data_classifiers == knn.1)/length(knn.1)
R_knn_1

YOUR CODE, AND ANALYSIS, HERE.

## Hint:
```

```r
## Error = rep(0, 10)
## for(k in 1:10){
##    "fit model with k-NN"
##    Error[k] = "error of the model"
## }
```

ANSWERS:

1.

2.

3.

Linear Discriminant Analysis

Now let's do the same thing with Fisher's Linear Discriminate Analysis (LDA). Don't forget that you can type `?lda` or `?predict` in the console to see the documentation for these functions.

Question 5

1. Run Fisher's LDA on the training data from the previous exercises.
2. Use the recently fitted model and `predict` to create the predictions of LDA on the training data.
3. Find the risk. Comment on how it compares to the risk of the Bayes rule.

4. Create a fine grid of values between -6 to 6. Using the `predict` function, find the prediction of the LDA algorithm on the values of x between -6 and 6.
5. Use this to determine the decision boundary of the LDA model. Is it similar to that of the Bayes rule?

```r
library(MASS)
library(dplyr)

?lda
?predict.lda
# Fit the model using the 'lda' function
YOUR CODE HERE

# Make predictions using the 'predict' function
YOUR CODE HERE

# Compute the percentage of missclassified observations.
YOUR CODE HERE
```

ANSWERS:

1.

2.

3.

4.

5.

Method Evaluation on New Data

Now that we have explored Bayes' Rule, k-nearest neighbors, and LDA, we will see how each method performs on new data that was NOT used to fit them. Consider the following new data set drawn from the same distributions.

```r
W_obs = rnorm(150, mean = -2, sd = 1.2)
V_obs = rnorm(100, mean = 2, sd = 1.2)
Y_class_test = c(rep(0, times = length(W_obs)), rep(1, times = length(V_obs)))
test_data = data.frame(X = c(W_obs, V_obs), Y = Y_class_test)
```

Question 6:

1. Using the decision boundary of the Bayes rule obtained in Q2, classify the new available data. Find the risk on these new observations.
2. Using the decision boundary of the LDA procedure obtained in Q5, classify the new available data. Find the risk on these new observations.
3. Use the function **knn** to determine the predicted class of the new data given the training data with $k = 1, 5, 10$. Find the risk on these new observations.

```r
## Find the risk of the estimated Bayes rule.

## Find the risk of the estimated LDA classification rule.

## Find the risk of the KNN algorithm for k = 1.

## Find the risk of the KNN algorithm for k = 5.

## Find the risk of the KNN algorithm for k = 10.
```

1.

2.

3.

Now, let's do this 1000 more times!

Question 7:

1. Use the code we provide below, and your code from Q6 to find the risk of the Bayes rule, LDA and KNN with $k = 1, 5, 10$ on 1000 different replications of new data. Save the risk on the corresponding vectors.
2. Create a plot with multiple boxplots, with which you can compare how the errors of Bayes, LDA and the KNN models behave.
3. Which model has the lowest average risk? Is the difference large?
4. Which model has the lowest variability in the risk? Are the result of different models highly variable?

```r
set.seed(13)
all_bayes_risks = c()
all_knn_risks = c()
```

```r
all_lda_risks = c()

for(iteration in 1:1000){
  W_obs = rnorm(150, mean = -2)
  V_obs = rnorm(50, mean = 2)
  Y_class_test = c(rep(0, times = length(W_obs)), rep(1, times = length(V_obs)))
  test_data = data.frame(X = c(W_obs, V_obs), Y = Y_class_test)

  YOUR MODEL CODE HERE

  bayes_risk = YOUR CODE HERE
  knn_risk1 = YOUR CODE HERE
  knn_risk5 = YOUR CODE HERE
  knn_risk10 = YOUR CODE HERE
  lda_risk = YOUR CODE HERE

  all_bayes_risks = c(all_bayes_risks, bayes_risk)
  all_knn_risks = c(all_knn_risks, knn_risk)
  all_lda_risks = c(all_lda_risks, lda_risk)

}

## Create boxplot here.
```

ANSWERS:

1.
2.
3.
4.

1994 Census Data

Let's walk through real example. First, load the `adults.csv` data (downloaded originally from here).

```r
adult = read.csv("adults.csv")
adult$X = NULL
```

Next, we will run the logistic regression model to predict the class `income`, which marks whether a given adult makes $\leqslant \$50k$ (coded as a 0) or $\geqslant \$50k$ (coded as a 1). To assess the performance of our model, we will only build the model on 75% of our data so that we can later use the remaining 25% as a testing data set.

```r
set.seed(13)
training_size <- round(.75 * nrow(adult))  # training set size
indices = sample(1:nrow(adult), training_size)
training_set <- adult[indices,]
testing_set <- adult[-(indices),]
m1 <- glm(income ~ ., data = training_set, family = binomial('logit'))
```

Question 8:

1. Examine the `summary` of our logistic regression model. Comment on the significance of each of our predictors.

2. Do any of the significant predictors surprise you?

3. Provide an interpretation of what the `Estimate` value is for the predictor `age`. Your answer should say something about this value's relation to the log-odds.

```
YOUR CODE HERE
```

ANSWERS:

1.
2.
3.

Question 9:

1. Now that you have created a model on the `training_data`, use the `predict` function in **R** to use your model to classify the data in the `testing_data`.

2. What proportion of values were classified correctly? Find the risk on the testing data.

```
YOUR CODE, AND ANALYSIS, HERE
```

ANSWERS:

1.
2.

Next we would like to fit an LDA model for comparison. Unfortunately, there is a slight problem: most of the predictors in the dataset are categorical. Recall that LDA assumes that the class conditional densities are Gaussian. Obviously this does not make sense for categorical variables. Luckily, **R** has a way around this. For a categorical variable that takes on d different values, **R** will automatically replace that variable with $d-1$ binary variables representing the levels of the original categorical variable. When all of these binary variables are equal to 0, the implied category is the one that was left out. For example, in the `adults` dataset, the variable `sex` takes on two values: `Female` and `Male`. As described above, `sex` can be recoded as a single binary variable, `sexMale`, which is equal to 1 when `sex == Male` and 0 when `sex == Female`. With these new binary variables in place, we can run LDA. Thus, when you run LDA on the `adults` dataset, you will not see a binary variable for `Female`.

Question 10:

1. Build an LDA model on the `training_data`, and see how well it performs classifying the observations in the `testing_data`.
2. Compare the risk (percentage of missclassified observations) of the LDA and the logistic regression. Which method gives the best performance?

```
YOUR CODE, AND ANALYSIS, HERE
```

ANSWERS:

1.
2.