Lab / HW 1

Lab 1: Olive Oils

Get the olive oils data sets (train and test) from Blackboard. Read the training data into R using the read.table function. You will need to use the "header = F" argument and set the "sep" argument correctly for the read.table function. To get help on the read.table function, call help(read.table).

When working in R it is helpful to write all your code in a script (.R) file, and run the code by copying and pasting to the command line. This saves having to retype code, and keeps a record of what you have done.

The resulting data set (e.g., "oliveOils") should be a data.frame object (check this using is.data.frame(oliveOils)). We want oliveOils to have the appropriate column names, so we will need to change them by calling

> colnames(oliveOils) = c("region", "area", …)

Choose appropriate names based on the information in the olive oils metadata file, also on Blackboard.

How many samples are in the training data? Use the dim command.

Look at the first several records in your imported data set to check whether the data has been corrupted when reading it into R. The command to do this is e.g.

> olive[1:10,]

at the R command prompt. Just make sure the data looks the same as the data in the training data text file (if it was read in incorrectly then there might be too few columns, columns with all "NA" values, etc.). Our goal is to estimate the region or area of origin for new samples of olive oil (e.g. in the test data set), by learning from the data set we have been given (the training data set). Is this supervised or unsupervised learning?

Check whether the data has been read in as numeric, character, or factor data. This may be different for different variables (columns) in your data.frame object. The variables can be selected using the syntax "olive$region". Useful functions for obtaining the format of the variables are "is.numeric", "is.character", and "is.factor". Region and area clearly are intended to be factor (categorical) variables, so coerce them to factor variables by calling e.g.

> olive$region = as.factor(olive$region)

How many values does the region variable take in the data set? Obtain this by calling:

> levels(olive$region)

Are there any missing values for the region in the data set? Missing values can be encoded in many ways: by a period, dash, or question mark, by a text string such as "unknown", or using a number that does not correspond to any of the possible values for the variable, for instance a 9 when the variable should be between 1 and 4. The "levels" function will show you all the values that the variable takes in the data set, except the special NA value which can be used for missing data. Check for NA values in

olive$region using the is.na function. Note that applying is.na to olive$region will result in a long vector of true / false values. You want to summarize this vector rather than visually inspecting the whole thing, so use the sum function, which will give you the number of true values.

Exploratory data analysis

Obtain the distribution (frequencies of each value) of the region and area variables using the "summary" function. What is the region with the most samples? Look at the distribution of several of the fatty acid variables by using the "density" function and creating a density plot. This plot should show a continuous curve for the density (not a bunch of separate points), and should have informative x and y labels. For help on the density function and its arguments type "help(density)". For help on using the plot function to create a line plot such as this one, call "help(plot.default)". The main help page for this function says "Graphical parameters may also be supplied as arguments to this function (see par)." Go to the "par" help file by clicking, and you will see the additional graphical arguments. The ones that you will need are the "xlab", "ylab", and "type" arguments.

Now take the "palmitic" variable, and create separate density plots for the three different regions. You will need to take the appropriate subsets of the olive$palmitic vector (try olive$palmitic[olive$region = = 1]). Put the three plots on a single page using the command:
> par(mfrow = c(3, 1))
before creating the plots. Make sure that the three plots have the same x-axis scale so that they are easy to compare. This can be done using the "xlim" argument, which is also described in the "par" help file. What do you notice?

Creating a Decision Rule

Repeat this process for each of the fatty acid variables. Which fatty acids could be used to most accurately distinguish between the regions? Given the fatty acid composition for a new sample, what simple decision rule would you use to guess the region of origin for the sample? Here I am looking for a classification rule looking something like the following:

if (fatty_acid_name_1 > level_1) then predicted_region = region_num;
else if (fatty_acid_name_2 < level_2) then predicted_region = other_region_num;
else predicted_region = last_region_num;

For now, I want each part of this decision rule to involve just one of the fatty acids. Not everyone will choose the same decision rule here, but there are a few clearly best decision rules and you should be able to figure out one of them.

Homework 1

PLEASE PUT YOUR NAME AND YOUR TA'S NAME ON YOUR HW (or we'll deduct a point).

1. Accuracy on the Test Data

Write a function that takes a data frame that has the fatty acid levels as variables (columns) and a set of samples as the rows, and outputs a vector of predicted regions, predicted according to your rule. Write another function that takes a (test) data frame that has the fatty acid levels and the true region as variables and a set of samples as the rows, and outputs a table of the counts of predicted vs. true regions, i.e.

	1	2	3
1	5	1	0
2	0	3	0
3	0	0	6

meaning that there were 5 samples with both true and predicted region equal to 1, 1 sample with true region 1 and predicted region 2, etc. The function should also output the overall error rate, i.e. the percentage of test samples that were misclassified. This function should call the first function (the prediction function) in order to do the prediction, rather than having copies of the same prediction code in both functions.

Describe your classification rule IN WORDS. Run your function on the olive oils test data. What was your error rate? What types of errors were made? Print out your code and hand it in.

2. Refining the Decision Rule

Recall the olive oils data set and the prediction function that you wrote. With a decision rule of the form used in Lab I, it is possible to get 100% accuracy on the training dataset, but there are many data points very close to this decision boundary. We will find a pair of fatty acids that distinguish the regions more decisively, creating a decision rule that involves a linear combination of these two fatty acid levels that also attains 100% accuracy on the training data. Then we will apply to the test data, and report the accuracy.

You should have discovered in the previous lab that one region can be very effectively separated from the other two using just one fatty acid. We can ignore this region for now since it can already be distinguished, and try to find a good separation for the remaining two regions. Work for now with a data set that has data from the easily separable region removed. The code to remove this region is

```
> olivesRegRem = oliveOils[ oliveOils$region != regionNum, ]
```

Take the 2-3 types of fatty acids that you found in Lab I to be most helpful in distinguishing the two remaining regions. Create pairwise scatterplots of these variables (using the plot function), assigning different colors for the two regions. For help on using the plot function to create a scatterplot, call "help(plot.default)". The main help page for this function says "Graphical parameters may also be supplied as arguments to this function (see par)." Go to the "par" help file by clicking, and you will see the additional graphical arguments. The one that you will need is the "col" argument.

These scatterplots give you an idea of whether you will be able to use two of these variables together in order to create a good classification boundary. If the two colors are well separated on the plot, then you can create a good classification rule based on the separating boundary.

Also create pairwise scatterplots of each of these 2-3 variables with each of the other variables in the data set. What we are looking for is a combination of two variables that yields a clear linear separation between the two regions. When you find two such variables, find the formula for a line that effectively separates the two regions. It may be that a curve would fit the data even better, but for now just use a line. You can add your line to the plot to check whether it is a good separator using the "lines" function.

Update your function from Lab 1 to use your new prediction rule. Then report the % of observations correctly classified in the test data.

Describe your classification rule IN WORDS. Hand in the printed code for your classification function.

3. Say that after training naïve Bayes the estimated marginal probability that Y= + is 0.4 and the estimated marginal probability that Y= - is 0.6. The estimated conditional distributions of the predictors X_1 and X_2 are:

	$X_1 = 1$	$X_1 = 2$
Y= +	0.27	0.73
Y= -	0.42	0.58

	$X_2 = a$	$X_2 = b$
Y= +	0.89	0.11
Y= -	0.83	0.17

a. Predict the value of Y if $X_1 = 2$ and $X_2 = b$, using a classification threshold of 0.5 (predict Y=+ if $Pr(Y=+ \mid X_1 = 2 \text{ and } X_2 = b) > 0.5$). Show all work.
b. Predict the value of Y if $X_1 = 2$ and X_2 is missing, using a classification threshold of 0.5. Show all work.