

2) Consider the following data set for a binary class problem.

A	B	Class Label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−

Calculate the misclassification error rate when splitting on A and B to determine the best split. Which of these splits considered is the best according to misclassification error rate?

- i. As we can see from the above table, if we split on A, the misclassification error would be: $\frac{3}{10} = 0.3$. Because in rows 4, 9 and 10 we can see that three records of A are misclassified and 10 is the total number of records.
- ii. If we split on B, there are misclassifications in row 1 and 9 with respect to B, so the rate would be 0.2.
- iii. Since the misclassification rate is low when we split the data set on B, we need to induct our decision tree based on B split.

3) Consider the training examples shown below for a binary classification problem.

Instance	a_1	a_2	a_3	Target Class
1	T	T	1.0	+
2	T	T	6.0	+
3	T	F	5.0	−
4	F	F	4.0	+
5	F	T	7.0	−
6	F	T	3.0	−
7	F	F	8.0	−
8	T	F	7.0	+
9	F	T	5.0	−

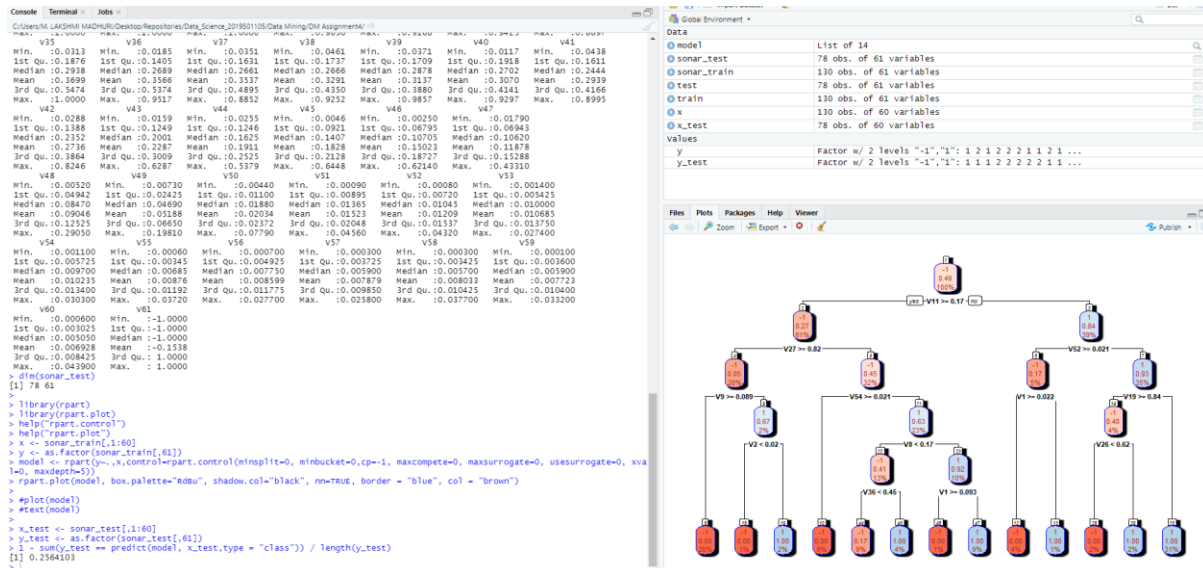
For a3, which is a continuous attribute compute misclassification error rate for every possible split to determine the best split. Which of these splits considered is the best according to misclassification error rate?

- i. Splitting on a1, the misclassification error rate = $\frac{2}{9} = 0.22$
- ii. Splitting on a2, the misclassification error rate = $\frac{5}{9} = 0.55$
- iii. Splitting on a3, [So, splitting on a3 will not be straight because it is not a nominal value or categorical value. Here, the a3 has discrete values and I decided to split on condition $a3 < 5.0$ as + $a3 \geq 5.0$ as -, the misclassification error rate would be = $\frac{3}{9} = 0.33$

4) The file http://www-stat.wharton.upenn.edu/~dmease/rpart_text_example.txt gives an example of text output for a tree fit using the rpart() function in R from the library rpart. Use this tree to predict the class labels for the 10 observations in the test data http://www-stat.wharton.upenn.edu/~dmease/test_data.csv linked here. Do this manually - do not use R or any software.

- i. Age = Middle, Number = 5 and Start = 10, the class label is present, as we traverse from 1 -> 2 -> 5 -> 11
- ii. Age = young, Number = 2, Start = 17, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8
- iii. Age = old, Number = 10, Start = 6, the class label is present, as we traverse from 1 -> 3 -> 7 -> 15
- iv. Age = young, Number = 2, Start = 17, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8
- v. Age = old, Number = 4, Start = 15, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8
- vi. Age = middle, Number = 5, Start = 15, the class label is absent, as we traverse from 1 -> 2 -> 5 -> 10
- vii. Age = young, Number = 3, Start = 13, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 9
- viii. Age = old, Number = 5, Start = 8, the class label is present, as we traverse from 1 -> 3 -> 7 -> 15
- ix. Age = young, Number = 7, Start = 9, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 9
- x. Age = middle, Number = 3, Start = 13, the class label is absent, as we traverse from 1 -> 2 -> 5 -> 10

5) I split the popular sonar data set into a training set (http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv) and a test set (http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv). Use R to compute the misclassification error rate on the test set when training on the training set for a tree of depth 5 using all the default values except control = rpart.control(minsplit = 0, minbucket = 0, cp = -1, maxcompete = 0, maxsurrogate = 0, usesurrogate = 0, xval = 0, maxdepth = 5). Remember that the 61st column is the response and the other 60 columns are the predictors.



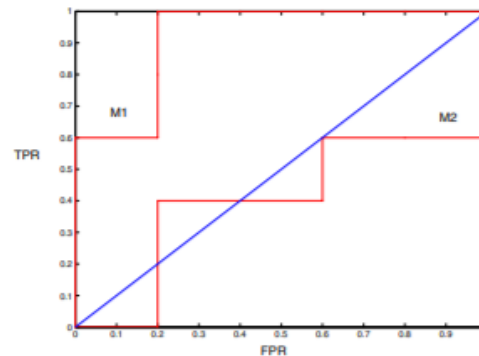
6) Do Chapter 5 textbook problem #17 (parts a and c only) on pages 322-323. Note that there is a typo in part c - it should read "Repeat the analysis for part (b)". We will do part b in class.

You are asked to evaluate the performance of two classification models, M1 and M2. The test set you have chosen contains 26 binary attributes, labeled as A through Z.

Table 5.14 shows the posterior probabilities obtained by applying the models to the test set. (Only the posterior probabilities for the positive class are shown). As this is a two-class problem, $P(-) = 1 - P(+)$ and $P(-|A, \dots, Z) = 1 - P(+|A, \dots, Z)$. Assume that we are mostly interested in detecting instances from the positive class.

Instance	True Class	$P(+ A, \dots, Z, M_1)$	$P(+ A, \dots, Z, M_2)$
1	+	0.73	0.61
2	+	0.69	0.03
3	-	0.44	0.68
4	-	0.55	0.31
5	+	0.67	0.45
6	+	0.47	0.09
7	-	0.08	0.38
8	-	0.15	0.05
9	+	0.45	0.01
10	-	0.35	0.04

- a) Plot the ROC curve for both M1 and M2. (You should plot them on the same graph.) Which model do you think is better? Explain your reasons.



From the above figure we can see that the M1 model is better as the TPR is more than that of the M2. M1 is better, since its area under the ROC curve is larger than the area under ROC curve for M2.

- c) Repeat the analysis for part (c) using the same cutoff threshold on model M2. Compare the F-measure results for both models. Which model is better? Are the results consistent with what you expect from the ROC curve?

$t = 0.5$, the confusion matrix for M2 is shown below:

		+	-
Actual	+	1	4
	-	1	4

For model M2: Precision ($\frac{TP}{TP+FP}$) = $\frac{1}{2} = 50\%$, Recall ($\frac{TP}{TP+FN}$) = $\frac{1}{5} = 20\%$
F-measure = $\frac{(2 \times .5 \times .2)}{(.5 + .2)} = 0.2857$

- 7) Compute the misclassification error on the training data for the Random Forest classifier to the last column of the sonar training data. Show your R code for doing this.

```

Console | Terminal | Jobs
C:\Users\W. LAKSHMI MADHURI\Desktop\Repositores\Data_Science_2019501105\Data Mining\DM Assignment4\
> setwd("C:\\Users\\W. LAKSHMI MADHURI\\Desktop\\Repositores\\Data_Science_2019501105\\Data Mining\\DM Assignment4")
> options(warn=1)
>
> #install.packages("randomForest")
> library("randomForest")
> sonar_train <- read.csv("sonar_train.csv", header = FALSE)
> sonar_test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = sonar_train[,1:60]
> y_train = as.factor(sonar_train[,61])
>
> x_test = sonar_test[,1:60]
> y_test = as.factor(sonar_test[,61])
>
> model<-randomForest(x_train, y_train)
> 1 - sum(y_train == predict(model, x_train)) / length(y_train)
[1] 0
>

```

Global Environment

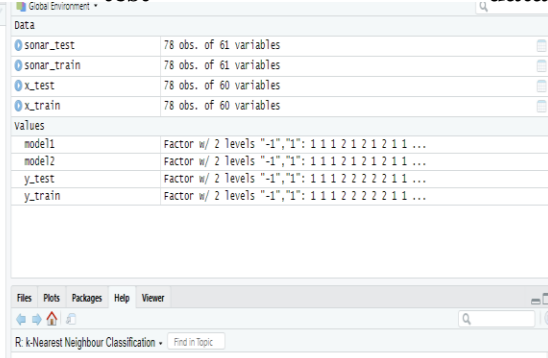
Data	
model	List of 18
sonar_test	78 obs. of 61 variables
sonar_train	78 obs. of 61 variables
x_test	78 obs. of 60 variables
x_train	78 obs. of 60 variables

Values	
y_test	Factor w/ 2 levels "1","0": 1 1 1 2 2 2 2 1 1 ...
y_train	Factor w/ 2 levels "1","0": 1 1 1 2 2 2 2 1 1 ...

8) This question deals with sonar data

a) Use `knn()` for the k-nearest neighbor classifier for $k=5$ and $k=6$ to the last column of the sonar training data. Compute the misclassification error on the training data and also on the test data.

```
C:\Users\M. LAKSHMI MADHURI\Desktop\Repositories\Data_Science_2019501105\Data Mining\DM Assignment4\
> options(warn=-1)
>
> #install.packages("class")
> library(class)
>
> sonar_train <- read.csv("sonar_train.csv", header = FALSE)
> sonar_test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = sonar_train[,1:60]
> y_train = as.factor(sonar_train[,61])
>
> x_test = sonar_test[,1:60]
> y_test = as.factor(sonar_test[,61])
>
> help("knn")
> model1<-knn(x_train, x_test, y_train, k = 5)
> 1 - sum(y_test == model1) / length(y_test)
[1] 0.2051282
>
> model2<-knn(x_train, x_test, y_train, k = 6)
> 1 - sum(y_test == model2) / length(y_test)
[1] 0.2820513
>
>
```



b) Repeat part a using the exact same R code a few times. Explain why both the training errors and the test errors often change for $k=6$ but not for $k=5$. Hint: Read the help on the `knn` function if you do not know.

```
C:\Users\M. LAKSHMI MADHURI\Desktop\Repositories\Data_Science_2019501105\Data Mining\DM Assignment4\
> options(warn=-1)
>
> #install.packages("class")
> library(class)
>
> sonar_train <- read.csv("sonar_train.csv", header = FALSE)
> sonar_test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = sonar_train[,1:60]
> y_train = as.factor(sonar_train[,61])
>
> x_test = sonar_test[,1:60]
> y_test = as.factor(sonar_test[,61])
>
> help("knn")
> model1<-knn(x_train, x_test, y_train, k = 5)
> 1 - sum(y_test == model1) / length(y_test)
[1] 0.2051282
>
> model2<-knn(x_train, x_test, y_train, k = 6)
> 1 - sum(y_test == model2) / length(y_test)
[1] 0.2820513
> options(warn=-1)
>
> #install.packages("class")
> library(class)
>
> sonar_train <- read.csv("sonar_train.csv", header = FALSE)
> sonar_test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = sonar_train[,1:60]
> y_train = as.factor(sonar_train[,61])
>
> x_test = sonar_test[,1:60]
> y_test = as.factor(sonar_test[,61])
>
> help("knn")
> model1<-knn(x_train, x_test, y_train, k = 5)
> 1 - sum(y_test == model1) / length(y_test)
[1] 0.2051282
>
> model2<-knn(x_train, x_test, y_train, k = 6)
> 1 - sum(y_test == model2) / length(y_test)
[1] 0.2948718
>
>
```

When there are two classes and the k value is even there might be a risk of a tie in the decision of which class to choose. So, the misclassification error rate often changes for $k=6$.