Data Mining Assignment 4

1) Read Chapter 4 (all sections) and Chapter 5 (Sections 5.2, 5.5, 5.6 and 5.7).

2) Repeat In Class Exercise #38 using the misclassification error rate instead of information gain to determine the best split. Which of these splits considered is the best according to misclassification error rate?

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

- As we can see from the above table, if we split on A, the misclassification error would be: 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.
- If we split on B, there are misclassifications in row 1 and 9 with respect to B, so the rate would be 0.2.
- 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) Repeat In Class Exercise #39 using the misclassification error rate instead of information gain to determine the best split. Which of these splits considered is the best according to misclassification error rate?

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	−

- Splitting on a1, the misclassification error rate = 2 / 9 = 0.22
- Splitting on a2, the misclassification error rate = 5 / 9 = 0.55
- 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 >= 5.0 as -], the misclassification error rate would be = 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.

- Age = Middle, Number = 5 and Start = 10, the class label is present, as we traverse from 1 -> 2 -> 5 -> 11
- Age = young, Number = 2, Start = 17, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8
- Age = old, Number = 10, Start = 6 , the class label is present, as we traverse from 1 -> 3 -> 7 -> 15
- Age = young, Number = 2, Start = 17, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8

- Age = old, Number = 4, Start = 15, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 8
- Age = middle, Number = 5, Start = 15, the class label is absent, as we traverse from 1 -> 2 -> 5 -> 10
- Age = young, Number = 3, Start = 13, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 9
- Age = old, Number = 5, Start = 8, the class label is present, as we traverse from 1 -> 3 -> 7 -> 15
- Age = young, Number = 7, Start = 9, the class label is absent, as we traverse from 1 -> 2 -> 4 -> 9
- 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 predictor

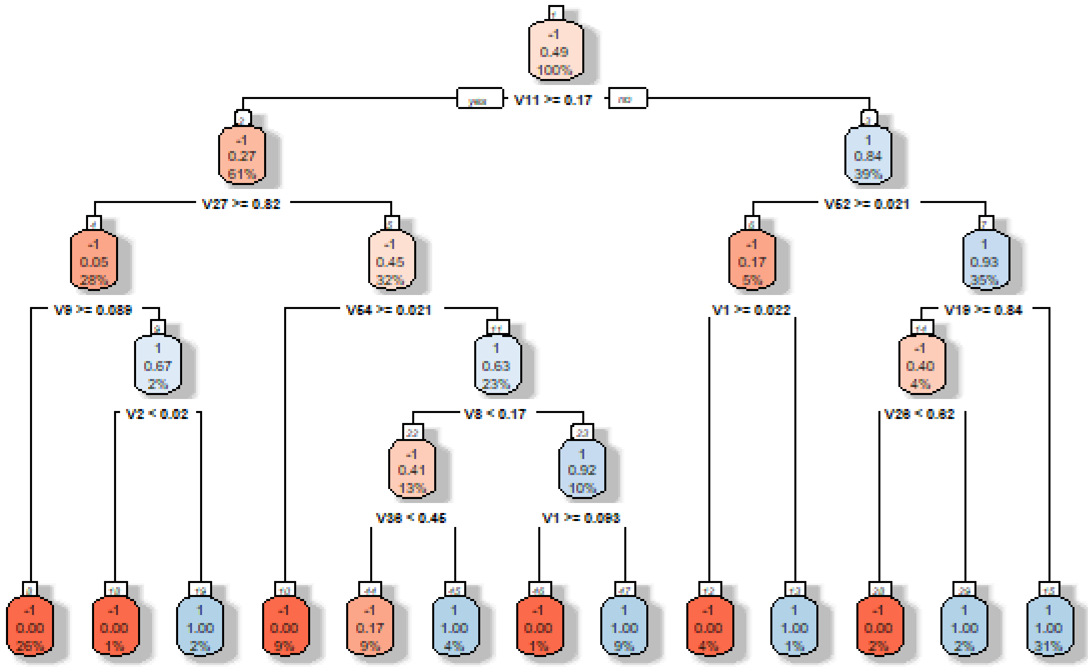
Console   Terminal ×   Jobs ×

F:/Data Science/DataScience_2019501111/Data Mining/DM Assignment4/

```
> setwd("F:\\Data Science\\DataScience_2019501111\\Data Mining\\DM Assignmen
t4")
> getwd()
[1] "F:/Data Science/DataScience_2019501111/Data Mining/DM Assignment4"
>
> train = read.csv("sonar_train.csv",header = FALSE)
> test = read.csv("sonar_test.csv",header = FALSE)
> summary(train)
       V1                V2                V3                V4
 Min.   :0.00150   Min.   :0.00170   Min.   :0.00150   Min.   :0.00580
 1st Qu.:0.01523   1st Qu.:0.01675   1st Qu.:0.01903   1st Qu.:0.02390
 Median :0.02285   Median :0.03040   Median :0.03415   Median :0.03905
 Mean   :0.02868   Mean   :0.03677   Mean   :0.04250   Mean   :0.05127
 3rd Qu.:0.03445   3rd Qu.:0.04735   3rd Qu.:0.05375   3rd Qu.:0.06270
 Max.   :0.13710   Max.   :0.16320   Max.   :0.19970   Max.   :0.26040
       V5                V6                V7               V8
 Min.   :0.00670   Min.   :0.01160   Min.   :0.0033   Min.   :0.00980
 1st Qu.:0.03955   1st Qu.:0.06310   1st Qu.:0.0808   1st Qu.:0.07485
 Median :0.06090   Median :0.08695   Median :0.1055   Median :0.11405
 Mean   :0.07355   Mean   :0.10467   Mean   :0.1222   Mean   :0.13749
 3rd Qu.:0.09625   3rd Qu.:0.13378   3rd Qu.:0.1568   3rd Qu.:0.17095
 Max.   :0.32250   Max.   :0.38230   Max.   :0.3729   Max.   :0.45900
       V9               V10              V11              V12
 Min.   :0.0155   Min.   :0.0242   Min.   :0.0327   Min.   :0.0269
 1st Qu.:0.0953   1st Qu.:0.1057   1st Qu.:0.1217   1st Qu.:0.1211
 Median :0.1463   Median :0.1736   Median :0.2204   Median :0.2493
 Mean   :0.1839   Mean   :0.2123   Mean   :0.2399   Mean   :0.2542
 3rd Qu.:0.2455   3rd Qu.:0.2774   3rd Qu.:0.3128   3rd Qu.:0.3444
 Max.   :0.6828   Max.   :0.7106   Max.   :0.7342   Max.   :0.7060
      V13              V14              V15              V16
 Min.   :0.0252   Min.   :0.0336   Min.   :0.0031   Min.   :0.0162
 1st Qu.:0.1548   1st Qu.:0.1568   1st Qu.:0.1459   1st Qu.:0.1805
 Median :0.2451   Median :0.2614   Median :0.2594   Median :0.2903
 Mean   :0.2723   Mean   :0.2809   Mean   :0.2959   Mean   :0.3549
 3rd Qu.:0.3679   3rd Qu.:0.3877   3rd Qu.:0.4291   3rd Qu.:0.5059
 Max.   :0.7131   Max.   :0.7842   Max.   :0.8392   Max.   :0.9444
      V17              V18              V19              V20
 Min.   :0.0349   Min.   :0.0375   Min.   :0.0494   Min.   :0.0740
 1st Qu.:0.1940   1st Qu.:0.2261   1st Qu.:0.3119   1st Qu.:0.3430
 Median :0.3036   Median :0.3654   Median :0.4433   Median :0.5736
```

```
> dim(train)
[1] 130  61
> summary(test)
       V1                V2                V3                V4
 Min.   :0.00250   Min.   :0.00060   Min.   :0.00510   Min.   :0.00610
 1st Qu.:0.01268   1st Qu.:0.01560   1st Qu.:0.01843   1st Qu.:0.02898
 Median :0.02245   Median :0.03085   Median :0.03450   Median :0.04830
 Mean   :0.02996   Mean   :0.04121   Mean   :0.04606   Mean   :0.05826
 3rd Qu.:0.03717   3rd Qu.:0.04893   3rd Qu.:0.06137   3rd Qu.:0.07093
 Max.   :0.13130   Max.   :0.23390   Max.   :0.30590   Max.   :0.42640
       V5                V6                V7                V8
 Min.   :0.00760   Min.   :0.01020   Min.   :0.01820   Min.   :0.0055
 1st Qu.:0.03535   1st Qu.:0.07445   1st Qu.:0.08192   1st Qu.:0.0888
 Median :0.06615   Median :0.09435   Median :0.11090   Median :0.1086
 Mean   :0.07796   Mean   :0.10440   Mean   :0.12103   Mean   :0.1303
 3rd Qu.:0.10743   3rd Qu.:0.13350   3rd Qu.:0.14822   3rd Qu.:0.1659
 Max.   :0.40100   Max.   :0.25870   Max.   :0.30160   Max.   :0.4223
       V9               V10               V11               V12
 Min.   :0.0075   Min.   :0.0113   Min.   :0.0289   Min.   :0.0236
 1st Qu.:0.1032   1st Qu.:0.1251   1st Qu.:0.1475   1st Qu.:0.1565
 Median :0.1560   Median :0.1888   Median :0.2354   Median :0.2490
 Mean   :0.1682   Mean   :0.2016   Mean   :0.2295   Mean   :0.2436
 3rd Qu.:0.2099   3rd Qu.:0.2541   3rd Qu.:0.2935   3rd Qu.:0.3105
 Max.   :0.5744   Max.   :0.5378   Max.   :0.5533   Max.   :0.5771
      V13               V14               V15               V16
 Min.   :0.0184   Min.   :0.0273   Min.   :0.0456   Min.   :0.0906
 1st Qu.:0.1798   1st Qu.:0.2273   1st Qu.:0.1882   1st Qu.:0.2062
 Median :0.2671   Median :0.2995   Median :0.3230   Median :0.3533
```

```
 Max.   :0.033200   Max.   :0.043900   Max.   : 1.0000
> dim(test)
[1] 78 61
>
> library(rpart)
> library(rpart.plot)
> help("rpart.control")
> help("rpart.plot")
> x <- train[,1:60]
> y <- as.factor(train[,61])
> model <- rpart(y~.,x,control=rpart.control(minsplit=0,minbucket=0,cp=-1, m
axcompete=0, maxsurrogate=0, usesurrogate=0, xval=0,maxdepth=5))
> rpart.plot(model, box.palette="RdBu", shadow.col="gray", nn=TRUE)
>
> x_test <- test[,1:60]
> y_test <- as.factor(test[,61])
> 1 - sum(y_test == predict(model,x_test,type = "class")) / length(y_test)
[1] 0.2564103
>
```



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, . . . , Z) = 1 - P(+|A, . . . , Z). Assume that we are mostly interested in detecting instances from the positive class.

Table 5.14. Posterior probabilities for Exercise 17.

Instance	True Class	$P(+\|A,\ldots,Z,M_1)$	$P(+\|A,\ldots,Z,M_2)$
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.

A) From the above figure we can see that the M1 model is better as the TPR is more than that of the M2.

(c)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.

A) For model M2: Precision = 1/2 = 50%. Recall = 1/5 = 20%. F-measure = (2 × .5 × .2)/(.5 + .2) = 0.2857.

7) Compute the misclassification error on the training data for the Random Forest classifier from In Class Exercise #47. Show your R code for doing this.

```
Console   Terminal ×   Jobs ×
F:/Data Science/DataScience_2019501111/Data Mining/DM Assignment4/
> install.packages("randomForest")
WARNING: Rtools is required to build R packages but is not currently installed. Please do
wnload and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/Documents/R/win-library/4.0’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.0/randomForest_4.6-14.zip'
Content type 'application/zip' length 249367 bytes (243 KB)
downloaded 243 KB

package ‘randomForest’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\hp\AppData\Local\Temp\RtmpiiHeOT\downloaded_packages
> library("randomForest")
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Warning message:
package ‘randomForest’ was built under R version 4.0.3
>
> train <- read.csv("sonar_test.csv", header = FALSE)
> test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = train[,1:60]
> y_train = as.factor(train[,61])
>
> x_test = test[,1:60]
> y_test = as.factor(test[,61])
>
> model<-randomForest(x_train, y_train)
> 1 - sum(y_train == predict(model, x_train)) / length(y_train)
[1] 0
>
```

8) This question deals with In Class Exercise #42.

a) Repeat In Class Exercise #42 for the k-nearest neighbor classifier for k=5 and k=6.

Console Terminal Jobs

F:/Data Science/DataScience_2019501111/Data Mining/DM Assignment4/

```
> library(class)
>
> train <- read.csv("sonar_test.csv", header = FALSE)
> test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = train[,1:60]
> y_train = as.factor(train[,61])
>
> x_test = test[,1:60]
> y_test = as.factor(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] 0.2692308
```

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.

```
> setwd("F:\\Data Science\\DataScience_2019501111\\Data Mining\\DM Assignment4")
> library(class)
>
> train <- read.csv("sonar_test.csv", header = FALSE)
> test <- read.csv("sonar_test.csv", header = FALSE)
>
> x_train = train[,1:60]
> y_train = as.factor(train[,61])
>
> x_test = test[,1:60]
> y_test = as.factor(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] 0.3461538
> 
```