**데이터마이닝이론**

**S T A 6 6 0 0**

**김현중 교수님**

**Homework 11**

2018314030

응용통계학과 김단아

**Note** : Consider only a classification problem. That is, there is a variable which indicates classes. The location of the class variable is not fixed. Make your program to handle more than two classes. You can assume that values of the class variable are integers starting with 1. Assume your data has both numerical and categorical variables. Further assume that the categorical variables are coded as integers starting with 1. You may use one of two (R or Python) language for this assignment.

1. Prompt the user to type in the filename of the training data.

2. Prompt the user to enter the locations of the categorical variables and the class variables.

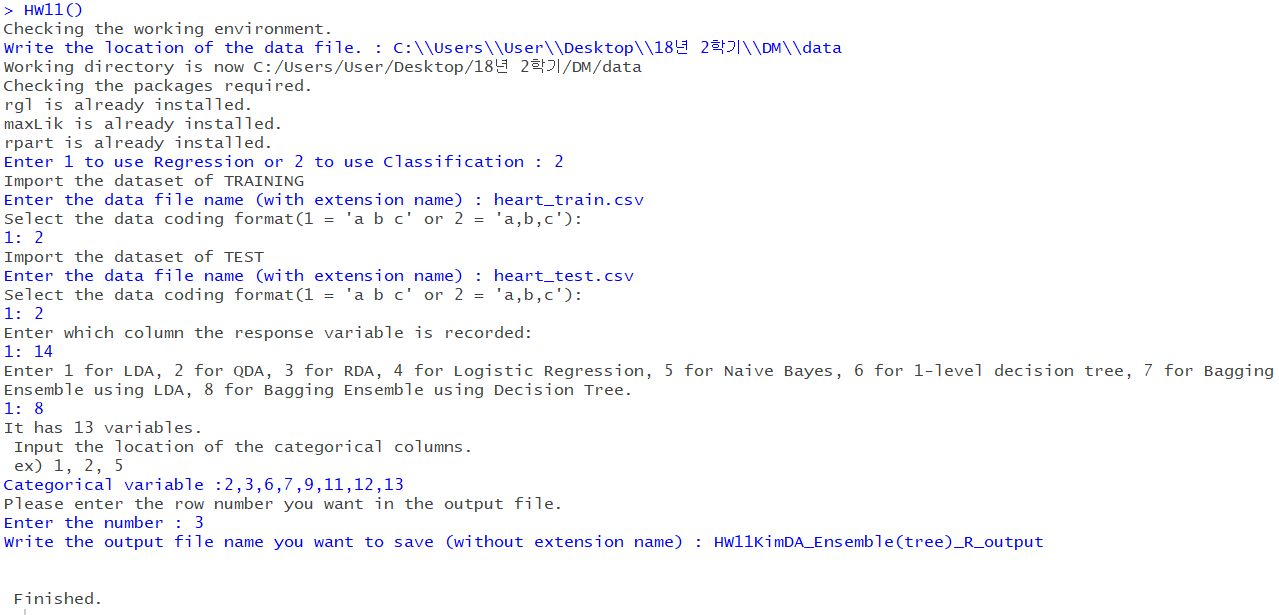
3. Prompt the user to enter the filename of the test dataset. (Assume the column location of the class variable is as same as that of the training dataset.)

4. Perform Bagging depending on the choice by the user.

5. For Bagging Ensemble method, use (1) decision trees with depth 2 and (2) decision trees with depth 4 as the classifier and 51 bootstraps as the number of re-sampled data.

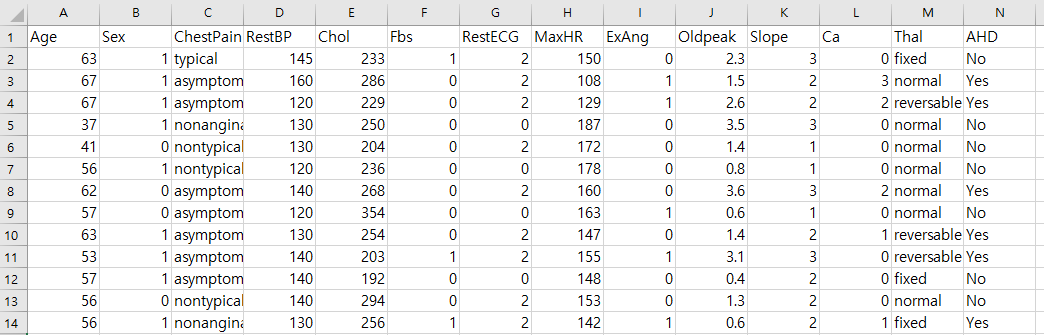
|  |
| --- |
| **( R code )** – 필요부분만 발췌  #############################################################  ##### 0. Checking the working environment & make importing function #####  #############################################################  # check working library  **mylib** = function() {  mylib = readline('Write the location of the data file. : ')  setwd(mylib)  cat('Working directory is now', getwd(),'\n')  }  # check installed package  **is.install** = function(package) {  if(!is.element(package, installed.packages()[,1])) {install.packages(package)}  else {cat(package,"is already installed. \n")}  }  # importing dataset function  **read** = function() {  data = readline("Enter the data file name (with extension name) : " )  cat("Select the data coding format(1 = 'a b c' or 2 = 'a,b,c'): ")  fm = scan(n=1, quiet=TRUE)  if(fm == 1){form=""} else {form=","}  read.table(data, sep=form)  }  ##############################  ##### 2. classification function #####  ##############################  **classification** <- function() {  ###################### General background #######################  # import training & test data file  cat('Import the dataset of TRAINING','\n')  train = read()  cat('Import the dataset of TEST','\n')  test = read()    # enter the Column number  cat("Enter which column the response variable is recorded: ")  num = scan(n=1, quiet=TRUE)    # nclass of response variable  k = length(unique(train[,num])) # Assume that values of the class variable are integers starting with 1    # choose the classification method  cat("Enter 1 for LDA, 2 for QDA, 3 for RDA, 4 for Logistic Regression, 5 for Naive Bayes, 6 for 1-level decision tree, 7 for Bagging Ensemble using LDA, 8 for Bagging Ensemble using Decision Tree.")  choice = scan(n=1, quiet=TRUE)    # do not prompt (iv) (v) (vi) when the data has more than 2 classes.  if(k>2&choice==4|k>2&choice==5|k>2&choice==6) {stop('Cannot run chosen method since data has more than 2 classes.') }  #############################################  ### (viii) **Bagging Ensemble using Decision Tree** ###  #############################################    if(choice==8) {  library(rpart)  n = nrow(train)  n.t = nrow(test)  y = train[,num]  y.t = test[,num]  x = train[,-num]  p = ncol(train)-1  k = length(unique(train[,num]))  cat('It has', p,'independent variables.','\n',  'Input the location of the categorical columns.','\n',  'ex) 1, 2, 5','\n')  cate = as.numeric(strsplit(readline('Categorical variable :'),split=",")[[1]])  for (i in cate) {  train[,i] <- as.factor(train[,i])  test[,i] <- as.factor(test[,i])  }  ##### **(1) Tree with depth 2** #####  # 51 bootstraps  id = c(1:n)  result\_d2 = matrix(nrow=n.t, ncol=51)  bootstrap = c()  names(train)[num]<-paste('y')  names(test)[num]<-paste('y')    for(i in 1:51) {  set.seed(i\*2)  bootstrap = train[sample(id, n, replace=TRUE),]  m = rpart(y ~ ., data=bootstrap, method='class', control = list(maxdepth=2))  p = predict(m, test, type='class')  result\_d2[,i] = p  }  c\_d2 = apply( result\_d2, 1, function(x) names(which.max(table(x))) )    # output  cat('Please enter the row number you want in the output file.')  out\_num = as.numeric(readline('Enter the number : '))  predict\_d2 = head(cbind(c(1:n.t), y.t, c\_d2), n=out\_num)  table\_d2 = table(y.t, c\_d2, dnn=c("Actual Class","Predicted Class"))  accuracy\_d2 = sum(diag(table\_d2))/sum(table\_d2)    ##### **(2) Tree with depth 4** #####  # 51 bootstraps  id = c(1:n)  result\_d4 = matrix(nrow=n.t, ncol=51)  bootstrap = c()  for(i in 1:51) {  set.seed(i\*4)  bootstrap = train[sample(id, n, replace=TRUE),]  m = rpart(y ~ . , data=bootstrap, method='class', control = list(maxdepth=4))  p = predict(m, test, type='class')  result\_d4[,i] = p  }  c\_d4 = apply( result\_d4, 1, function(x) names(which.max(table(x))) )    # output  predict\_d4 = head(cbind(c(1:n.t), y.t, c\_d4), n=out\_num)  table\_d4 = table(y.t, c\_d4, dnn=c("Actual Class","Predicted Class"))  accuracy\_d4 = sum(diag(table\_d4))/sum(table\_d4)    # make output file  outputname = readline("Write the output file name you want to save (without extension name) : ")  outputname = paste(outputname,".txt",sep="")  cat(' (1) Tree with depth 2', '\n', file = outputname, sep="")  cat("ID, Actual class, tree-depth2 pred", "\n", "-----------------------------", "\n", file = outputname, sep="", append=TRUE)  write.table(predict\_d2, outputname, sep= ", ", row.names=FALSE, col.names=FALSE, append=TRUE, quote=FALSE)  cat('(continue)','\n','\n', file = outputname, sep="", append = TRUE)  cat('Confusion Matrix (tree-depth2)', "\n", "----------------------------------", "\n",file = outputname,sep="", append=TRUE)  capture.output(print(table\_d2), file=outputname, append=TRUE)  cat("\n", "Model Summary (tree-depth2)", "\n", "------------------------------", "\n",file = outputname, sep="", append=TRUE)  cat("Overall accuracy: ", round(accuracy\_d2, 3), "\n\n",file = outputname, sep="", append=TRUE)    cat(' (2) Tree with depth 4', '\n', file = outputname, sep="", append=TRUE)  cat("ID, Actual class, tree-depth4 pred", "\n", "-----------------------------", "\n",file = outputname,sep="", append=TRUE)  write.table(predict\_d4, file=outputname, sep= ", ", row.names=FALSE, col.names=FALSE, append=TRUE, quote=FALSE)  cat('(continue)',"\n",'\n', file = outputname, sep="", append = TRUE)  cat('Confusion Matrix (tree-depth4)', "\n", "----------------------------------", "\n",file = outputname,sep="", append=TRUE)  capture.output(print(table\_d4), file=outputname,append=TRUE)  cat("\n", "Model Summary (tree-depth4)", "\n", "------------------------------", "\n",file = outputname,sep="", append=TRUE)  cat("Overall accuracy: ", round(accuracy\_d4, 3), "\n" ,file = outputname,sep="", append=TRUE)  }}  **HW11** = function(){  cat('Checking the working environment. \n')  mylib()  cat('Checking the packages required. \n')  is.install('rgl')  is.install('maxLik')  is.install('rpart')  ans = readline('Enter 1 to use Regression or 2 to use Classification : ')  if (ans==1) { regression() }  if (ans==2) {**classification()**}  cat('\n', '\n', 'Finished.')  } |

**( R console )**



|  |
| --- |
| **( HW11KimDA\_Ensemble(tree)\_R\_output )**  (1) Tree with depth 2  ID, Actual class, tree-depth2 pred  -----------------------------  1, 1, 1  2, 1, 1  3, 1, 2  (continue)  Confusion Matrix (tree-depth2)  ----------------------------------  Predicted Class  Actual Class 1 2  1 35 11  2 6 38  Model Summary (tree-depth2)  ------------------------------  Overall accuracy: 0.811  (2) Tree with depth 4  ID, Actual class, tree-depth4 pred  -----------------------------  1, 1, 1  2, 1, 1  3, 1, 2  (continue)  Confusion Matrix (tree-depth4)  ----------------------------------  Predicted Class  Actual Class 1 2  1 37 9  2 5 39  Model Summary (tree-depth4)  ------------------------------  Overall accuracy: 0.844 |

위 분석에 사용한 데이터는 아래와 같다. 심장병 진단과 관련된 13개의 변수가 있는 heart.csv 데이터를 웹에서 다운받아 사용하였다.[[1]](#endnote-1)



*Heart.csv*

Heart.csv 데이터를 Train dataset과 Test dataset으로 분할하기 위해 R의 ‘caret’ package를 사용하여 종속변수인 AHD를 기준으로 train:test = 7:3의 비율로 층화추출을 시행하였다. 213개의 row인 Train dataset과 90개의 row인 Test dataset을 heart\_train, heart\_test로 저장하였다. 코드는 아래와 같다.

|  |
| --- |
| library(caret)  intrain = createDataPartition(y=data$AHD, p=0.7, list=FALSE)  train = data[intrain, ]  test = data[-intrain, ]  write.csv(train, 'heart\_train.csv')  write.csv(test, 'heart\_test.csv') |

HW11() 함수를 실행하면 유저가 Prompt 창에 working directory를 지정하고 Regression은 1을, Classification은 2를 입력하고, Decision Tree를 이용한 Bagging Ensemble을 직접 실행할 수 있도록 하였다. heart\_train과 heart\_test를 이용한 Bagging Ensemble의 실행코드는 아래와 같다.

|  |
| --- |
| HW11()  C:\\Users\\User\\Desktop\\18년 2학기\\DM\\data  2  heart\_train.csv  2  heart\_test.csv  2  14  8  2,3,6,7,9,11,12,13  3  HW11KimDA\_Ensemble(tree)\_R\_output |

이 코드의 수행 결과는 앞 쪽에 첨부되어 있다.

1. http://www-bcf.usc.edu/~gareth/ISL/data.html [↑](#endnote-ref-1)