Lab on Decision Trees & Random Forests

1 Decision Trees

Decision trees (also referred to as classification and regression trees) are the traditional building blocks of data mining and one of the classic machine learning algorithms. Their appeal lies in their transparency (or interpretability) and the usual simplicity of the resulting model. A decision tree (at least one that is not too large) is quite easy to view, to understand, and, indeed, to explain. However, decision trees do not always deliver the best performance and represent a trade-off between performance and simplicity of explanation.

Preparation:

1. Work in a group of size three to four (no more than four students are to work together).
2. Boot the computer into Windows mode.
3. Download the file **datasets.zip** and save it to an arbitrary folder, say `C:/Users/yourname/Desktop`
4. Uncompress the zip file into this folder
5. Start **R**
6. Change the working directory by entering:

   ```
   setwd("C:/Users/yourname/Desktop")
   ```

   (Note that R expects forward slashes rather than backwars slashes as used by Windows.)
7. Install the packages `party`, `tree`, `rpart` and `randomForest` by entering the commands:

   ```
   install.packages("party")
   install.packages("tree")
   install.packages("rpart")
   install.packages("randomForest")
   ```

Your task: Your group is to submit a PDF document which contains your answers of the questions in this laboratory exercise. One document is to be submitted by each group. The header of the document must list the name and student number of all students in the group. Clearly indicate which question you have answered.

Getting started: Let's start by re-producing some of the Decision Trees that were shown in the lecture notes: Execute the following sequence of commands:

```
library(party)
library(tree)
library(rpart)
library(randomForest)

#Create a decision tree with rpart
iris.rpart = rpart(Species ~ ., data = iris)
#Lets have a look at the result
print(iris.rpart)
plot(iris.rpart)  #Plot the tree
text(iris.rpart)  #Show the labels

#Create a decision tree with ctree
iris.ctree = ctree(Species ~ ., data = iris)
print(iris.ctree)
plot(iris.ctree)
```

Note: The lecture notes explained the output of ctree and rpart.

Also notice that ctree and rpart produce two different Descision Trees.

Both ctree and rpart expect at least two parameters:

Format: The first parameter defines the target attribute and the set of attributes to use when building the Descision Tree. In the example above, the format "Species ∼ ." specifies that the attribute Species contains the targets (the classification labels) and that all other attributes (denoted by the dot '.') are to be considered when building the tree. If, for example, we only wanted to use the attributes Petal.Length and Petal.Width then we would have specified `iris.rpart = rpart(Species ∼ Petal.Length + Petal.Width, data = iris)`

Data: Name of the dataset to use.

We have learnt that it can be beneficial to use some of the training data for testing and/or validation. For example as in the following:

```
#Chose a subset for training
sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
fit <- rpart(Species ~ ., data = iris, subset = sub)

#Compute the confusion matrix on the training set
table(predict(fit, iris[sub,], type = "class"), iris[sub, "Species"])
#Compute the confusion matrix on the test set
table(predict(fit, iris[-sub,], type = "class"), iris[-sub, "Species"])
```

We have also learnt that R uses a default value for the minimum number of records (or minimum weight is a weighting variable is specified) which must be present in order to allow nodes to split. The parameter can be controlled as in the following example:

```
play=read.csv("Play.csv") # load the records
print(play)               # look at the records
#Create decision tree with default parameters
play.ctree = ctree(Class ~ ., data=play)
print(play.ctree)         # Look at the tree produced by ctree
play.rpart = rpart(Class ~ ., data=play)
print(play.rpart)         # Look at the tree produced by rpart

#Create decision tree by changing the default minimum split condition
play.ctree = ctree(Class ~ ., data=play, weights = rep(14, 14))
print(play.ctree)         # Tree produced by ctree
play.rpart = rpart(Class ~ ., data=play, minsplit=1)
print(play.rpart)         # Look at the tree produced by rpart
```

Question 1: Information Gain

Q1.1: Use the information gain algorithm (see lecture notes) to create the Decision Tree for the "Play" dataset. Main aim of this task is to obtain the Decision Tree for the "Play" dataset when using the information gain method. Assume the following:

```
Outlook={sunny, overcast, rain}
Tmp={>=80F, <80F}
Humidity={>70%, <=70%}
Windy={True, False}
```

Q1.2: Compare the Decision Tree which you obtained in Q1.1 with those produced by ctree and rpart.

Question 2: Evaluation

Q2.1 Use ctree to ctreate the Decision Tree for the Audit dataset as follows:

```
audit = read.csv("Audit.csv")
audit.ctree = ctree(TARGET_Adjusted ~ ., data = audit)
```

Plot the Decision Tree and show the confusion matrix. Explain what can be seen.
Note: Default values used in R may lead to R believing that the Audit dataset is a regression problem. You may have to change the default behaviour of R so that the Audit dataset is being dealt with correctly (as a classification problem).

Q2.2: Repeat the above by using a randomly selected half of the samples in the dataset for training and the rest for testing:

```
set.seed(7)
#Randomly select half of the samples
set=sample(1:nrow(audit),nrow(audit)/2, replace=FALSE);
audit.train=audit[set,];  # Create training set
audit.test=audit[-set,];  # Create test set
```

Create a Decision Tree for the training set, plot the tree, and create the confusion matrices for the training set and for the test set. Compare the Decision tree and confusion matrices with those obtained in Q2.1.

Question 3: Random Forests

A random forest is an ensemble of randomly constructed trees. Using random forests in R on the example of the "Play" dataset:

```
play.forest <- randomForest(Class ~ .,   data=play)
print(play.forest)
plot(play.forest)
```

Repeat these two commands several times and observe the changes in the printed result (confusion matrix) and in the plot.

By default, R uses 500 random trees and a minimum size for a leaf node of 5. These default values can be changed as follows:

```
play.forest <- randomForest(Class ~ .,   data=play, nodesize=1, ntree=10)
plot(play.forest)
```

Q3.1: Explain what is shown by the function `plot`. Also explain why the results seem to change every time when the function randomForest is called.

Q3.2: Use the voting dataset:

```
vote=read.csv("Voting.csv")
```

Classify the samples by using random forrests of size 2. Repeat by doubling the number of trees to 4. Repeat by doubling the number of trees to 8, and so on, doubling every time until ntree=$32,000$. Explain the results.

Submit your answers as a PDF file. Only one member of each group needs to submit. Make sure that the name and student number of all group members is listed on the first page of your answer sheet. Your answer must be properly typeset (do not submit a scanned version of a handwritten document).

– End of tasksheet –