Godavari college of Engineering , Jalgaon

Subject Name: Machine Learning

Practical No: 03 Date:

Title: Random Forest and Parameter Tuning in R

Aim: Study and implementation of Random Forest and Parameter Tuning in R

Theory:

In the random forest approach, a large number of decision trees are created. Every observation is fed into every decision tree. The most common outcome for each observation is used as the final output. A new observation is fed into all the trees and taking a majority vote for each classification model.

An error estimate is made for the cases which were not used while building the tree. That is called an **OOB (Out-of-bag)** error estimate which is mentioned as a percentage.

The R package "**randomForest**" is used to create random forests.

Install R Package:

Use the below command in R console to install the package. You also have to install the dependent packages if any.

```
install.packages(‘randomForest’)
```

The package "randomForest" has the function **randomForest()** which is used to create and analyze random forests.

Syntax:

The basic syntax for creating a random forest in R is −

```
randomForest(formula, data)
```

Following is the description of the parameters used −

- **formula** is a formula describing the predictor and response variables.
- **data** is the name of the data set used.

Input Data:

We will use the R in-built data set named readingSkills to create a decision tree. It describes the score of someone's readingSkills if we know the variables "age","shoesize","score" and whether the person is a native speaker.

Here is the sample data.

```
# Load the party package. It will automatically load other
# required packages.
library(party)

# Print some records from data set readingSkills.
print(head(readingSkills))
```

When we execute the above code, it produces the following result and chart –

Output:-

```
nativeSpeaker  age  shoeSize    score
1       yes    5  24.83189  32.29385
2        yes    6  25.95238  36.63105
3         no   11  30.42170   49.60593
4        yes    7  28.66450   40.28456
5        yes   11  31.88207  55.46085
6        yes   10  30.07843  52.83124
Loading required package: methods
Loading required package: grid
...............................
...............................
```

Example:

We will use the **randomForest()** function to create the decision tree and see it's graph.

```
# Load the party package. It will automatically load other
# required packages.
library(party)
library(randomForest)

# Create the forest.
output.forest < - randomForest(nativeSpeaker ~ age + shoeSize + score,
        data = readingSkills)

# View the forest results.
print(output.forest)
```

When we execute the above code, it produces the following result –

Output:-

```
Call:
 randomForest(formula = nativeSpeaker ~ age + shoeSize + score,
             data = readingSkills)
            Type of random forest: classification
                  Number of trees: 500
No. of variables tried at each split: 1

       OOB estimate of  error rate: 1%
Confusion matrix:
   no yes class.error
no  99   1        0.01
yes  1 99        0.01
```

Conclusion:-

From the random forest shown above we can conclude that the shoesize and score are the important factors deciding if someone is a native speaker or not. Also the model has only 1% error which means we can predict with 99% accuracy.