Machine Learning-I (CS/DS 706)

Assignment 1

Chetna Jain (MT2016044)
Kanika Narang (MT2016069)
Tehreem Ansari (MT2016145)

Question 2: R

Background for the problem:

- First we check how data behaves without any corruption. 'Training' set consisting of 40 samples each from all the three classes setosa, versicolor, virginica. 'Testing' set consists of the separate 10 samples.
- For training the data we are using Random Forest. 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.
- The metrics used: Accuracy and Confusion Matrix
- Our class label is 'Species'. For the dataset we are using we have 3 different Species: Iris-setosa, Iris-versicolor and Iris-virginica. We train our random forest(rf) on this label and then test the rf on the same.
- The data also has 4 other labels: SepalLengthCm, SepalWidthCm, PetalLengthCm and PetalWidthCm.
- The number of trees to be generated is kept default of 500.
- Mtry: Is the Number of variables available for splitting at each tree node.

Step 1: Importing libraries.

```
library(party)
library(randomForest)
library(caret)
library(e1071)
```

Step 2: Loading the test and train data in R to be used.

```r
trainData <- read.csv(file='./IrisTraining.csv')
testData <- read.csv(file='./IrisTesting.csv')

print(head(trainData))
print(head(testData))
```

Output:

```
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.1          3.5           1.4          0.2 Iris-setosa
2           4.9          3.0           1.4          0.2 Iris-setosa
3           4.7          3.2           1.3          0.2 Iris-setosa
4           4.6          3.1           1.5          0.2 Iris-setosa
5           5.0          3.6           1.4          0.2 Iris-setosa
6           5.4          3.9           1.7          0.4 Iris-setosa
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.0          3.5           1.3          0.3 Iris-setosa
2           4.5          2.3           1.3          0.3 Iris-setosa
3           4.4          3.2           1.3          0.2 Iris-setosa
4           5.0          3.5           1.6          0.6 Iris-setosa
5           5.1          3.8           1.9          0.4 Iris-setosa
6           4.8          3.0           1.4          0.3 Iris-setosa
```

Step 3: Train the randomForest on uncorrupted training data.

```r
#Training random forest on actual uncorrupted data
iris_rf <- randomForest(Species~.,data=trainData)
```

Step 4: Testing the random forest error on test data

```r
#Testing the dataset after training it on uncorrupted
irisPred<-predict(iris_rf,newdata=testData)
print(irisPred)
```

Output:

```
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.1          3.5           1.4          0.2 Iris-setosa
2           4.9          3.0           1.4          0.2 Iris-setosa
3           4.7          3.2           1.3          0.2 Iris-setosa
4           4.6          3.1           1.5          0.2 Iris-setosa
5           5.0          3.6           1.4          0.2 Iris-setosa
6           5.4          3.9           1.7          0.4 Iris-setosa
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.0          3.5           1.3          0.3 Iris-setosa
2           4.5          2.3           1.3          0.3 Iris-setosa
3           4.4          3.2           1.3          0.2 Iris-setosa
4           5.0          3.5           1.6          0.6 Iris-setosa
5           5.1          3.8           1.9          0.4 Iris-setosa
6           4.8          3.0           1.4          0.3 Iris-setosa
```

Step 5: Confusion Matrix and Accuracy

```
print ("WITHOUT CORRUPTION:")
confusionMatrix(testData$Species, irisPred)
```

Output:

```
Confusion Matrix and Statistics

                 Reference
Prediction        Iris-setosa Iris-versicolor Iris-virginica
  Iris-setosa              10               0              0
  Iris-versicolor           0              10              0
  Iris-virginica            0               0             10

Overall Statistics

               Accuracy : 1
                 95% CI : (0.8843, 1)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 4.857e-15

                  Kappa : 1
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Iris-setosa Class: Iris-versicolor
Sensitivity                      1.0000                 1.0000
Specificity                      1.0000                 1.0000
Pos Pred Value                   1.0000                 1.0000
Neg Pred Value                   1.0000                 1.0000
Prevalence                       0.3333                 0.3333
Detection Rate                   0.3333                 0.3333
Detection Prevalence             0.3333                 0.3333
Balanced Accuracy                1.0000                 1.0000
                     Class: Iris-virginica
Sensitivity                         1.0000
Specificity                         1.0000
Pos Pred Value                      1.0000
Neg Pred Value                      1.0000
Prevalence                          0.3333
Detection Rate                      0.3333
Detection Prevalence                0.3333
```

Step 6: Interpretation

From the above metric we can see that accuracy is 100%. Also the data used for training is uncorrupted.

Part 2.a: Training data with the 4 explanatory attributes corrupted with some noise, no change in class(Species) attribute

Currently the corruption is only in 4 data points. If more data points are corrupted, the answer may vary.

Step 1 and 2 is same as above. The only difference being, instead of IrisTraining.csv we are using IrisTraining_corrupted1.csv.

Ie

```
#Solving now on corrupted data
trainDataCorrupted <- read.csv(file='./IrisTraining_corrupted1.csv')
print(head(trainDataCorrupted))
```

Now, we are training our Random Forest algorithm on corrupted data instead of uncorrupted as before.

Step 3: Training on corrupted dataset

```
#training on corrupted data
iris_rf_corrupted1 <- randomForest(Species~.,data=trainDataCorrupted)
```

Step 4: Testing the test dataset after training on corrupted data

```
#testing dataset after training on corrupted
irisPred_corrupted1<-predict(iris_rf_corrupted1,newdata=testData)
print(irisPred_corrupted1)
```

Output:

```
              1               2               3               4               5 
    Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa 
              6               7               8               9              10 
    Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa 
             11              12              13              14              15 
Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor 
             16              17              18              19              20 
Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor 
             21              22              23              24              25 
 Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica 
             26              27              28              29              30 
 Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica 
Levels: Iris-setosa Iris-versicolor Iris-virginica
```

Step 5: Determining the accuracy and confusion metrics

```
print ("WITH CORRUPTION PART 1:")
confusionMatrix(testData$Species, irisPred_corrupted1)
```

Output:

```
[1] "WITH CORRUPTION PART 1:"

Confusion Matrix and Statistics

                 Reference
Prediction        Iris-setosa Iris-versicolor Iris-virginica
  Iris-setosa              10               0              0
  Iris-versicolor           0              10              0
  Iris-virginica            0               0             10

Overall Statistics

               Accuracy : 1
                 95% CI : (0.8843, 1)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 4.857e-15

                  Kappa : 1
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Iris-setosa Class: Iris-versicolor
Sensitivity                      1.0000                 1.0000
Specificity                      1.0000                 1.0000
Pos Pred Value                   1.0000                 1.0000
Neg Pred Value                   1.0000                 1.0000
Prevalence                       0.3333                 0.3333
Detection Rate                   0.3333                 0.3333
Detection Prevalence             0.3333                 0.3333
Balanced Accuracy                1.0000                 1.0000
                     Class: Iris-virginica
Sensitivity                         1.0000
Specificity                         1.0000
Pos Pred Value                      1.0000
Neg Pred Value                      1.0000
Prevalence                          0.3333
```

Step 6: Interpretation

From above, we can see that accuracy is still 100% even after adding some noise in the 4 features(Without adding noise in the class label). This is so because after adding noise, the random forest algorithm becomes more generalized and can recognize noise better. Hence on testing on test data, the random forest still gives 100% accuracy.

Part 2.b: Training data with some of the class labels corrupted (changed to something else) without changing any of the explanatory attributes.

6 data points have been corrupted in their class label. I have changed from the original value to IamCorrupted value.
Step 1 and 2 is same as above. The only difference being, instead of IrisTraining.csv we are using IrisTraining_corrupted2.csv.

```
trainDataCorrupted2 <- read.csv(file='./IrisTraining_corrupted2.csv')
testData <- read.csv(file='./IrisTesting.csv')

print(head(trainDataCorrupted2))
print(head(testData))
```

```
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.1          3.5           1.4          0.2 Iris-setosa
2           4.9          3.0           1.4          0.2 Iris-setosa
3           4.7          3.2           1.3          0.2 Iris-setosa
4           4.6          3.1           1.5          0.2 Iris-setosa
5           5.0          3.6           1.4          0.2 Iris-setosa
6           5.4          3.9           1.7          0.4 Iris-setosa
  SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
1           5.0          3.5           1.3          0.3 Iris-setosa
2           4.5          2.3           1.3          0.3 Iris-setosa
3           4.4          3.2           1.3          0.2 Iris-setosa
4           5.0          3.5           1.6          0.6 Iris-setosa
5           5.1          3.8           1.9          0.4 Iris-setosa
6           4.8          3.0           1.4          0.3 Iris-setosa
```

Step 3: Training on corrupted dataset

```
#training on corrupted data
iris_rf_corrupted2 <- randomForest(Species~.,data=trainDataCorrupted2)
```

Step 4: Testing the test dataset after training on corrupted data

```
#testing dataset after training on corrupted
irisPred_corrupted2<-predict(iris_rf_corrupted2,newdata=testData)
print(irisPred_corrupted2)
```

Output:

```
             1               2               3               4               5
   Iris-setosa     IamCorrupted     Iris-setosa     Iris-setosa     Iris-setosa
             6               7               8               9              10
   Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa     Iris-setosa
            11              12              13              14              15
Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor
            16              17              18              19              20
Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor
            21              22              23              24              25
 Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica
            26              27              28              29              30
 Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica  Iris-virginica
Levels: IamCorrupted Iris-setosa Iris-versicolor Iris-virginica
```

Step 5: Determining the accuracy and confusion metrics

```
print ("WITH CORRUPTION PART 2:")
confusionMatrix(testData$Species, irisPred_corrupted2)
```

Output:

```
Confusion Matrix and Statistics

                 Reference
Prediction        IamCorrupted Iris-setosa Iris-versicolor Iris-virginica
  IamCorrupted               0           0               0              0
  Iris-setosa                1           9               0              0
  Iris-versicolor            0           0              10              0
  Iris-virginica             0           0               0             10

Overall Statistics

               Accuracy : 0.9667
                 95% CI : (0.8278, 0.9992)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 2.963e-13

                  Kappa : 0.9508
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: IamCorrupted Class: Iris-setosa
Sensitivity                      0.00000             1.0000
Specificity                      1.00000             0.9524
Pos Pred Value                       NaN             0.9000
Neg Pred Value                   0.96667             1.0000
Prevalence                       0.03333             0.3000
Detection Rate                   0.00000             0.3000
Detection Prevalence             0.00000             0.3333
Balanced Accuracy                0.50000             0.9762
                     Class: Iris-versicolor Class: Iris-virginica
Sensitivity                          1.0000                1.0000
Specificity                          1.0000                1.0000
Pos Pred Value                       1.0000                1.0000
Neg Pred Value                       1.0000                1.0000
Prevalence                           0.3333                0.3333
Detection Rate                       0.3333                0.3333
Detection Prevalence                 0.3333                0.3333
```

Step 6: Interpretation

Here, we see that accuracy has reduced from 100% to 96.67%. This is because we are corrupting the training data in the class label itself and not the features. Hence the training random forest algorithm becomes overfitted and thus accuracy reduces when testing on the test data.