Random Forest examples

Abhirup Sen

02/06/2021

IRIS datasets

```
library(caTools)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
# install.packages('randomForest')
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
data(iris)
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Step 1:Split data in train and test data

```
split <- sample.split(iris, SplitRatio = 0.7)
split
```

```
## [1]  TRUE FALSE  TRUE FALSE  TRUE
```

```
training_set <- subset(iris, split== "TRUE")
test_set <- subset(iris, split== "FALSE")
```

Step 2: Fitting Random Forest Classification to the Training set

```
set.seed(123)
?randomForest
```

```
## starting httpd help server ... done
```

```
classifier = randomForest(x = training_set[-5],
                          y = training_set$Species,
                          ntree = 500)
```

Step 3: Predicting the Test set results

```
y_pred = predict(classifier, newdata = test_set[-5])
```

Step 4: Making the Confusion Matrix

```
table(test_set[, 5], y_pred)
```

```
##             y_pred
##              setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0         19         1
##   virginica       0          3        17
```

```
confusionMatrix(table(test_set[, 5], y_pred))
```

```
## Confusion Matrix and Statistics
##
##             y_pred
##              setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0         19         1
##   virginica       0          3        17
```

```
##
## Overall Statistics
##
##                Accuracy : 0.9333
##                  95% CI : (0.838, 0.9815)
##     No Information Rate : 0.3667
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.8636           0.9444
## Specificity                 1.0000            0.9737           0.9286
## Pos Pred Value              1.0000            0.9500           0.8500
## Neg Pred Value              1.0000            0.9250           0.9750
## Prevalence                  0.3333            0.3667           0.3000
## Detection Rate              0.3333            0.3167           0.2833
## Detection Prevalence        0.3333            0.3333           0.3333
## Balanced Accuracy           1.0000            0.9187           0.9365
```

Loan application

```
dataset <- read.csv("Loan Default.csv")
str(dataset)
```

```
## 'data.frame':    1000 obs. of  14 variables:
##  $ Default        : int  0 1 0 0 1 0 0 0 0 1 ...
##  $ checkingstatus1: chr  "A11" "A12" "A14" "A11" ...
##  $ duration       : int  6 48 12 42 24 36 24 36 12 30 ...
##  $ history        : chr  "A34" "A32" "A34" "A32" ...
##  $ purpose        : chr  "A43" "A43" "A46" "A42" ...
##  $ amount         : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
##  $ savings        : chr  "A65" "A61" "A61" "A61" ...
##  $ employ         : chr  "A75" "A73" "A74" "A74" ...
##  $ installment    : int  4 2 2 2 3 2 3 2 2 4 ...
##  $ residence      : int  4 2 3 4 4 4 4 2 4 2 ...
##  $ age            : int  67 22 49 45 53 35 53 35 61 28 ...
##  $ otherplans     : chr  "A143" "A143" "A143" "A143" ...
##  $ cards          : int  2 1 1 1 2 1 1 1 1 2 ...
##  $ tele           : chr  "A192" "A191" "A191" "A191" ...
```

```
dataset$Default <- factor(dataset$Default)
```

#Step 1:Split data in train and test data

```
split <- sample.split(dataset, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
```

Fitting Random Forest to the Training set

```
library(randomForest)
classifier <- randomForest(x = training_set[-1],
                           y = training_set$Default,
                           ntree = 520)
```

#Predict on your test Data using trained model

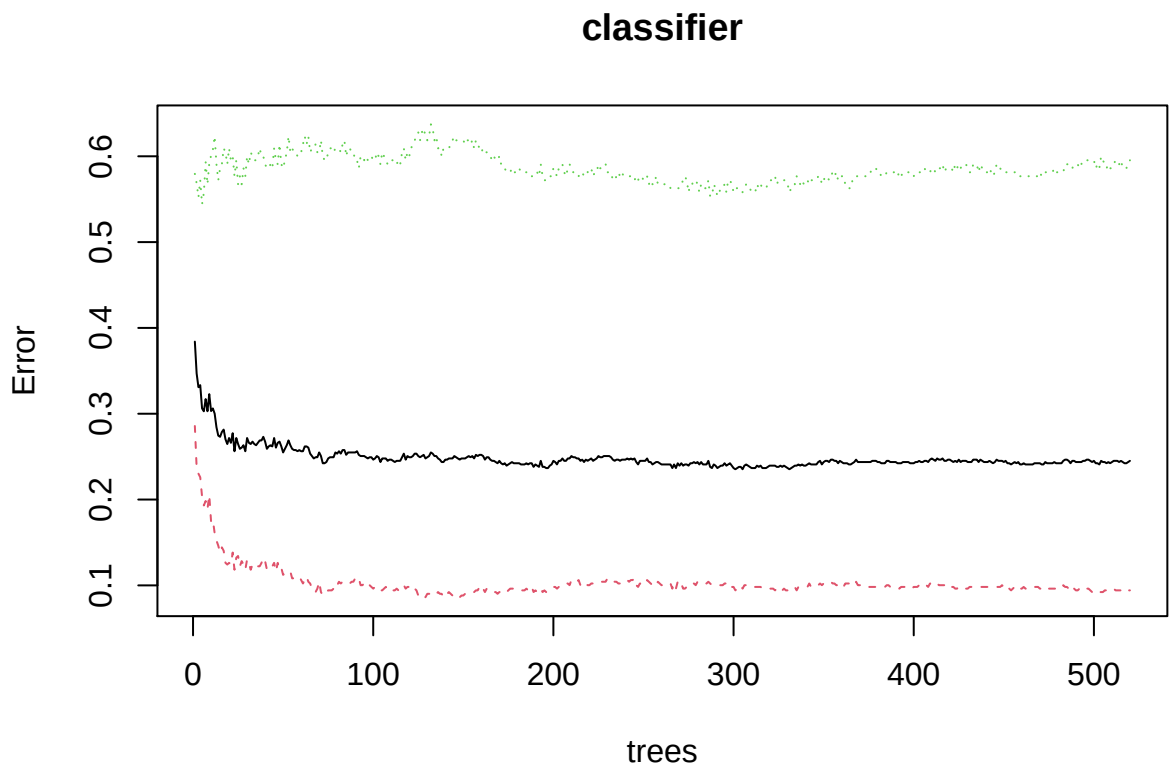
```
y_pred = predict(classifier, newdata = test_set[-1])
```

#Model Evaluation

```
cm <- table(test_set$Default,y_pred)
confusionMatrix(cm)
```

```
## Confusion Matrix and Statistics
##
##    y_pred
##       0   1
##   0 180  21
##   1  43  42
##
##                Accuracy : 0.7762
##                  95% CI : (0.7234, 0.8232)
##     No Information Rate : 0.7797
##     P-Value [Acc > NIR] : 0.589652
##
##                   Kappa : 0.4211
##
##  Mcnemar's Test P-Value : 0.008665
##
##             Sensitivity : 0.8072
##             Specificity : 0.6667
##          Pos Pred Value : 0.8955
##          Neg Pred Value : 0.4941
##              Prevalence : 0.7797
##          Detection Rate : 0.6294
##    Detection Prevalence : 0.7028
##       Balanced Accuracy : 0.7369
##
##        'Positive' Class : 0
##
```

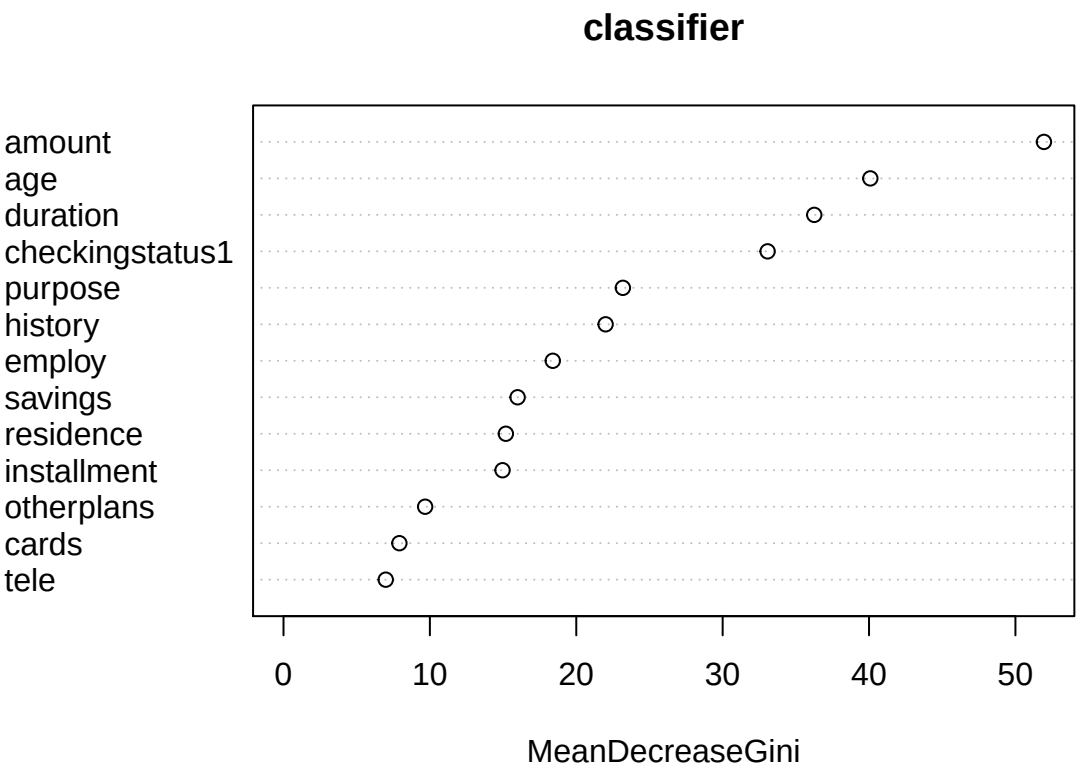
```
plot(classifier)
```




```
importance(classifier)
```

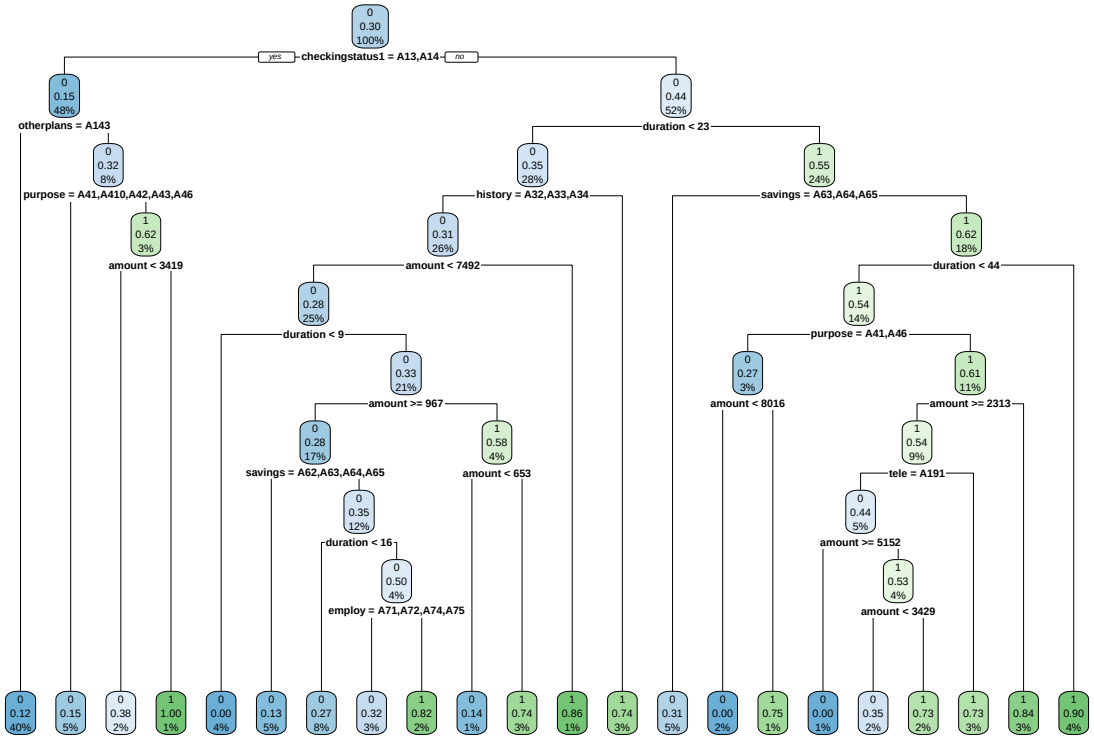
```
##                  MeanDecreaseGini
## checkingstatus1         33.070730
## duration                36.258046
## history                 22.005558
## purpose                 23.182571
## amount                  51.948272
## savings                 15.992434
## employ                  18.395977
## installment             14.963458
## residence               15.189473
## age                     40.086323
## otherplans               9.676125
## cards                    7.915945
## tele                     6.984610
```

```
varImpPlot(classifier)
```



#Compare with Decision Tree # Step 2:Train model with random forest using rpart function

```
library(rpart)
library(rpart.plot)
decision_tree_model<-rpart(Default ~ ., data = training_set, method = "class")
library(rpart.plot)
rpart.plot(decision_tree_model)
```



Step 3:Predict test data based on trained model

```
y_pred <-predict(decision_tree_model,newdata=test_set,type="class")
```

Step 4: Evaluate Model Accuracy using Confusion matrix

```
table(test_set$Default,y_pred)
```

```
##    y_pred
##       0   1
##   0 173  28
##   1  41  44
```

```
library(caret)
confusionMatrix(table(test_set$Default,y_pred))
```

```
## Confusion Matrix and Statistics
##
##    y_pred
##       0   1
##   0 173  28
##   1  41  44
```

```
##
##                Accuracy : 0.7587
##                  95% CI : (0.7049, 0.8072)
##     No Information Rate : 0.7483
##     P-Value [Acc > NIR] : 0.3705
##
##                   Kappa : 0.3958
##
##  Mcnemar's Test P-Value : 0.1486
##
##             Sensitivity : 0.8084
##             Specificity : 0.6111
##          Pos Pred Value : 0.8607
##          Neg Pred Value : 0.5176
##              Prevalence : 0.7483
##          Detection Rate : 0.6049
##    Detection Prevalence : 0.7028
##       Balanced Accuracy : 0.7098
##
##        'Positive' Class : 0
##
```