# Decision Tree

*Ravi Kumar Tiwari*

*13 July 2016*

## Classification

### 1: Decision Tree

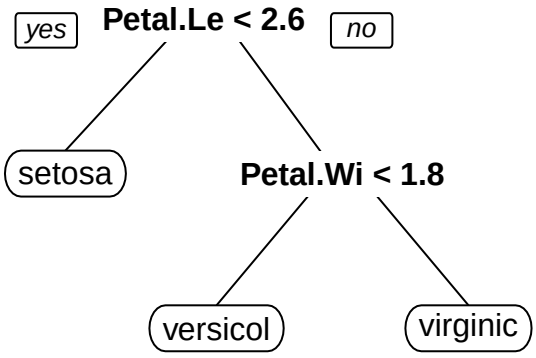
This approach involves dividing the predictor variables into smaller regions (using decision rules that combine to form a decision tree) such that response within these regions are (nearly) homogeneous. The response corresponding to a given observation is determined based on the region it falls into. For qualitative response, it is assigned the most dominant class of the region. For quantitative response, it is assigned the mean of the response values in the region

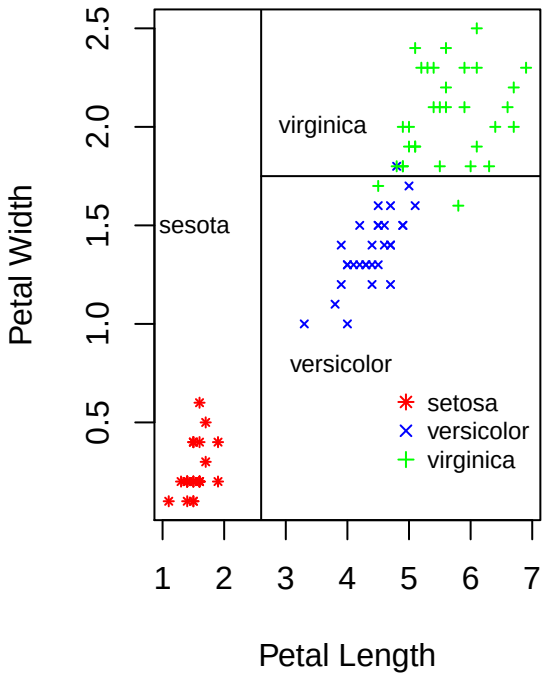
Example:

Decision tree showing decision rules to determine the species of a flower based on its Sepal and Petal measurments. On the right, the regions that results from those rules are shown. We look at the data, first

```
iris[c(1,100,150),]
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1            5.1         3.5          1.4         0.2     setosa
## 100          5.7         2.8          4.1         1.3 versicolor
## 150          5.9         3.0          5.1         1.8  virginica
```




**1.1: Code to create decision tree model**

```
## Load the required libraries
library(rpart)
library(rpart.plot)

## create data partition
set.seed(1)
inTrain <- sample(c(TRUE, FALSE), size = nrow(iris), replace = TRUE, prob = c(0.6,0.4))
trainData <- iris[inTrain,]
testData <- iris[!inTrain,1:4]
testClass <- iris[!inTrain,5]

## create the tree model and make prediction using the tree model
treeModel <- rpart(Species ~ ., data = trainData)
predClass <- predict(treeModel, newdata = testData, type = "class")

# Plot the tree
#rpart.plot(treeModel, type = 0)
```

### 1.2: Code for making prediction using the decision tree

```
predTrainClass <- predict(treeModel, newdata = trainData, type = "class")
predTestClass <- predict(treeModel, newdata = testData, type = "class")
```

### 1.3: Evaluating the performance of the decision tree

```
## Training Data
table(predTrainClass, trainData$Species)  # Confusion Matrix
```

```
##
## predTrainClass setosa versicolor virginica
##     setosa         27          0         0
##     versicolor      0         29         2
##     virginica       0          1        30
```

```
mean(predTrainClass == trainData$Species) # Prediction Accuracy
```

```
## [1] 0.9662921
```

```
## Test Data
table(predTestClass, testClass)           # Confusion Matrix
```

```
##              testClass
## predTestClass setosa versicolor virginica
##    setosa         23          0         0
##    versicolor      0         20         3
##    virginica       0          0        15
```

```
mean(predTestClass == testClass)
```

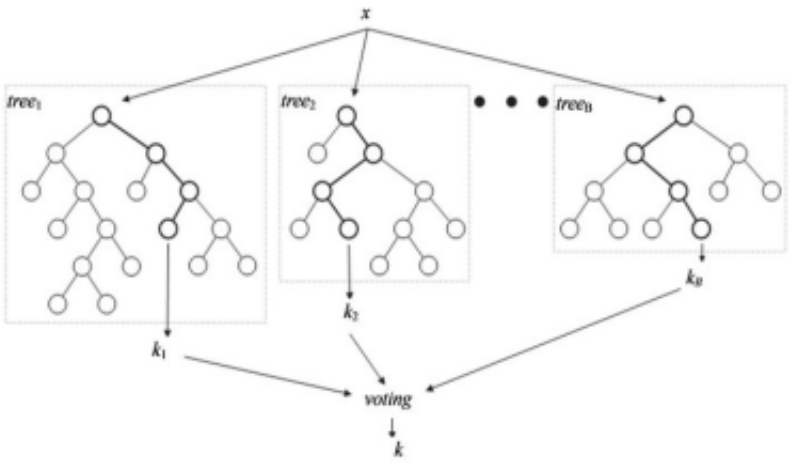
```
## [1] 0.9508197
```

## 2: Random Forest

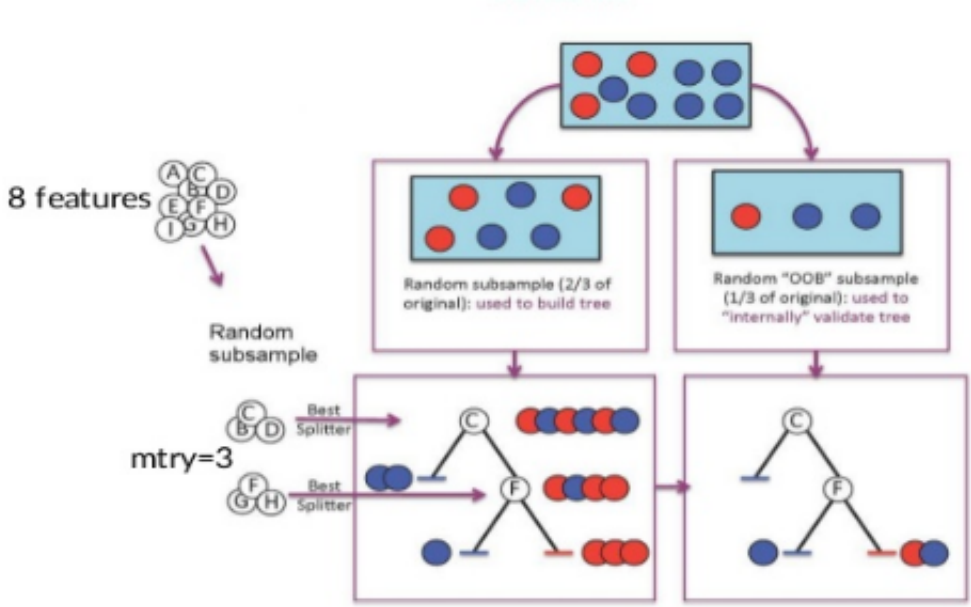
It fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging for prediction.

Important parameters of the random forest are: 1) ntree, 2) mtry

1. ntree




2. mtry



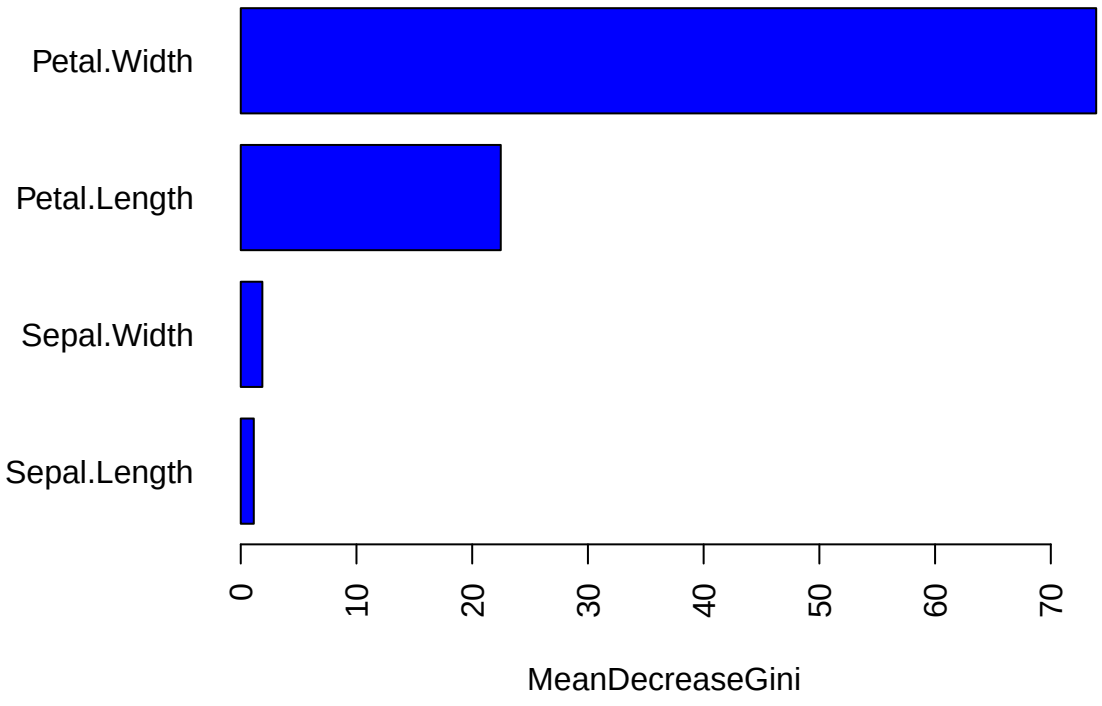
### 2.1: Code to build random forest model

```
library(randomForest)
set.seed(1)
rfModel <- randomForest(Species ~ ., data=iris, mtry=4, ntree=20)
predClass <- predict(rfModel, newdata = iris)
table(predClass, iris$Species)
rfModel$importance
```

### 2.2 Prediction Accuracy

```
##
## predClass    setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         49         0
##   virginica       0          1        50
```

### 2.3 Variable Importance

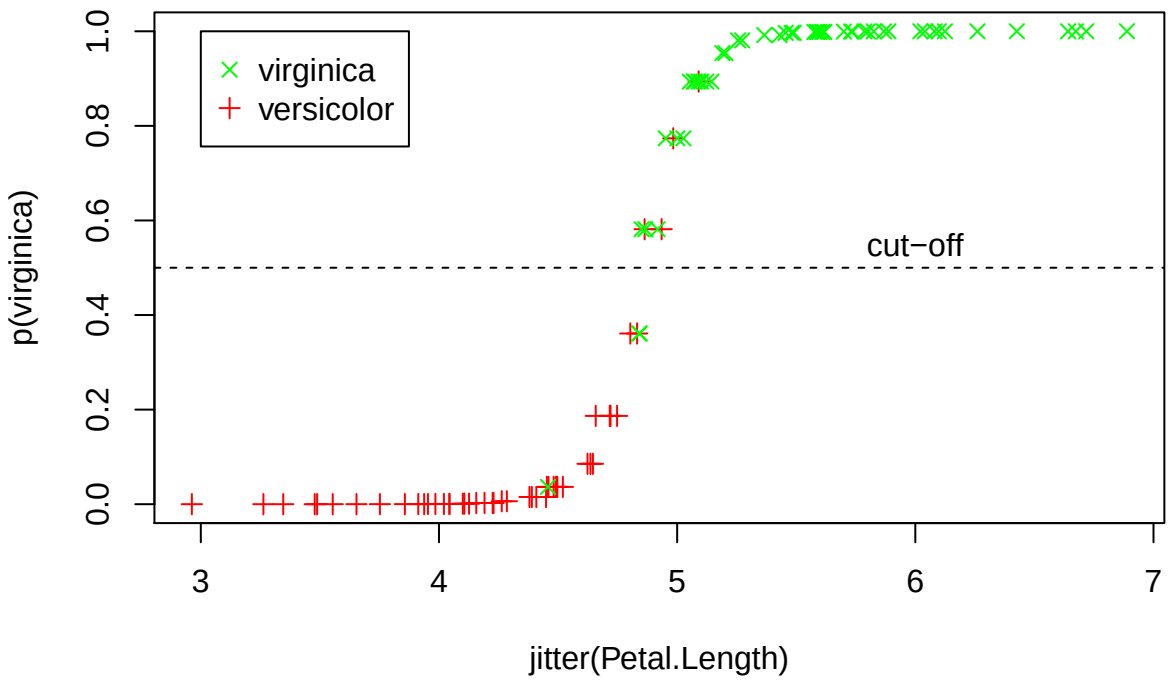


## 3. Logistic Regression

Logistic Regression is used for modeling qualitative Response. This approach involves fitting data to a logistic function whose output represents the class probability of an observation. A new observation is assigned a class based on its class probability and the chosen cut-off value.

### 3.1: Example

Logistic regression model for flower Species prediction based on its Petal.Length. The cut-off value for the class separation has been chosen to be 0.5.




### 3.2: Code to create logistic regression model

```
## In order to have only two classes, observations corresponding
## to Species, setosa, has been removed
inSetosa <- iris$Species == "setosa"
myIris <- iris[!inSetosa,]
myIris$Species <- factor(myIris$Species, levels = c("versicolor", "virginica"))

## Model training, prediction, and validation
glmModel <- glm(Species ~ Petal.Length, data = myIris, family = binomial(link="logit"))
predValue <- predict(glmModel, myIris, type = "response")
prediction <- ifelse(predValue > 0.5, "virginica", "versicolor")
table(prediction, myIris$Species)
```

### 3.3: Model Assessment

```
prediction <- ifelse(predValue > 0.5, "virginica", "versicolor")
table(prediction, myIris$Species)
```

```
##
## prediction   versicolor virginica
##   versicolor         46         3
##   virginica           4        47
```

### 3.4: Adding more predictor variables

```
## Adding more predictor variable
glmModel <- glm(Species ~ Petal.Length + Petal.Width, data = myIris,
                family = binomial(link="logit"))
predValue <- predict(glmModel, myIris, type = "response")
prediction <- ifelse(predValue > 0.5, "virginica", "versicolor")
table(prediction, myIris$Species)
```