Machine Learning

Ravi Kumar Tiwari

14 June 2016

Introduction

1. What is machine learning
2. Why is it important (some applications)
3. Some good motivation examples

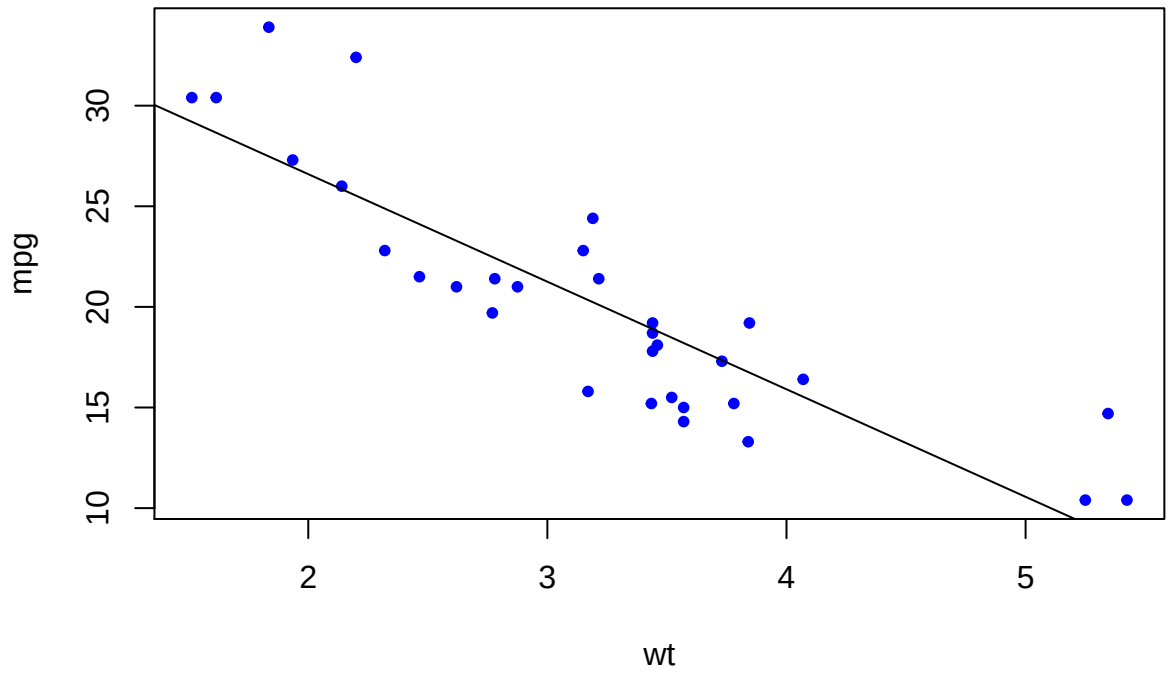
Regression

Working principle

We find a line that best describes the data.

Example

Build a linear model to describe the relationship between mpg (miles per gallon) and wt (weight of the car) in the mtcars dataset

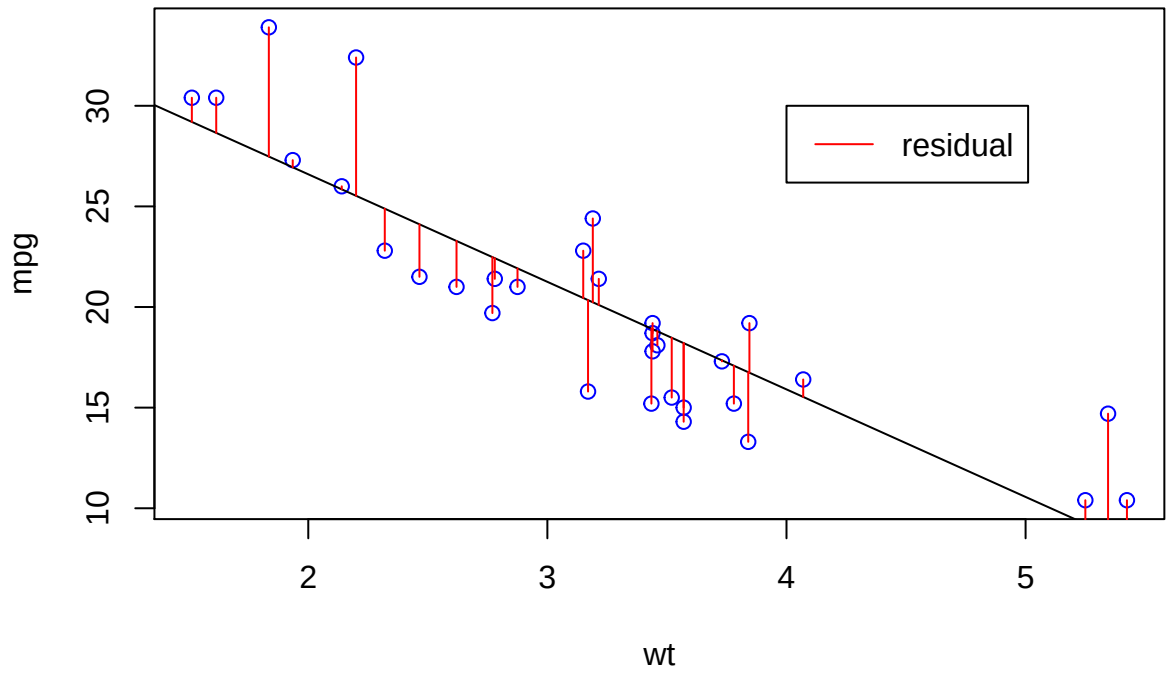


Codes

```
## Build the linear model object
lmModel <- lm(mpg ~ wt, data = mtcars)

## Obtain the model parameters
sumModel <- summary(lmModel)
sumModel$coefficients

## Prediction using the model
predValue <- predict(lmModel, data.frame(wt = mtcars$wt))
```

Model Assessment

1. Visual Inspection



2. R-squared value

```
sumModel <- summary(lmModel)
sumModel$r.squared
```

```
## [1] 0.7528328
```

3. F-statistics

```
sumModel$fstatistic
```

```
##    value    numdf    dendf
## 91.37533  1.00000 30.00000
```

Extension of linear model

```
## More than one predictors
lmModel2 <- lm(mpg ~ wt+hp+disp, data = mtcars) # wt, hp, and disp will be used as predictor
lmModel3  <- lm(mpg~ ., data = mtcars)   # All the variable will be used

## subset selection
## Identify the best model that contains a given number of predictor, where best is
## quantified using RSS

library(leaps)
fwdSelection <- regsubsets(mpg ~ ., data = mtcars, method = "forward")
sumFwdSel <- summary(fwdSelection)
names(sumFwdSel)
sumFwdSel$outmat
sumFwdSel$rsq
which.max(sumFwdSel$adjr2)
coef(fwdSelection,6)
```

Challenge

Use backward selection model to find the best model for mpg

Tree based algorithm

Used both for classification and regression

Workign principle

Divide the data set into several small regions such that the response variables are homogeneous in those regions. The predictd value of a new observation is the most dominant class of the region to which the observation belongs.

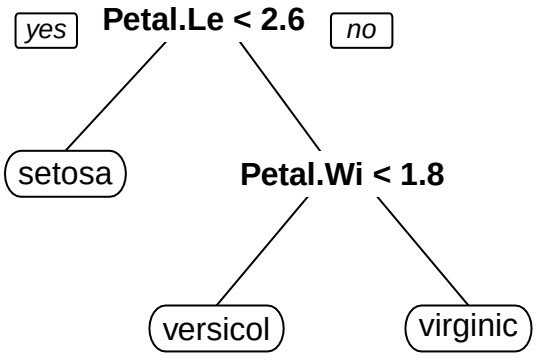
Example

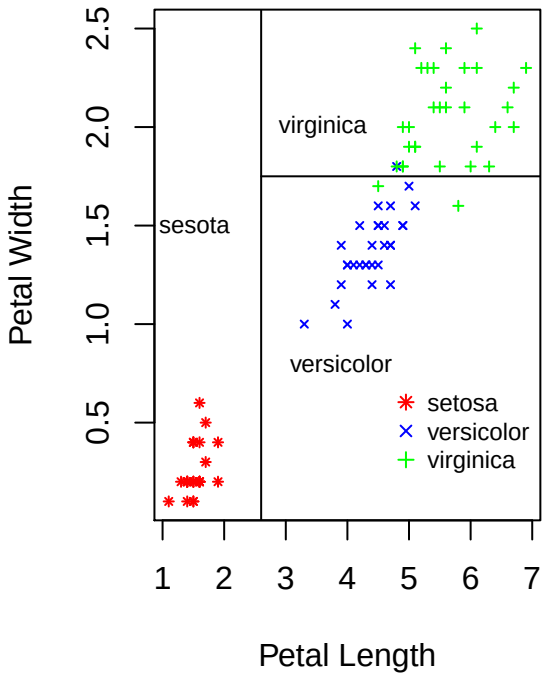
Find the decision rule to predict the species of iris dataset based on Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width

```
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
```

Decision Tree visualization

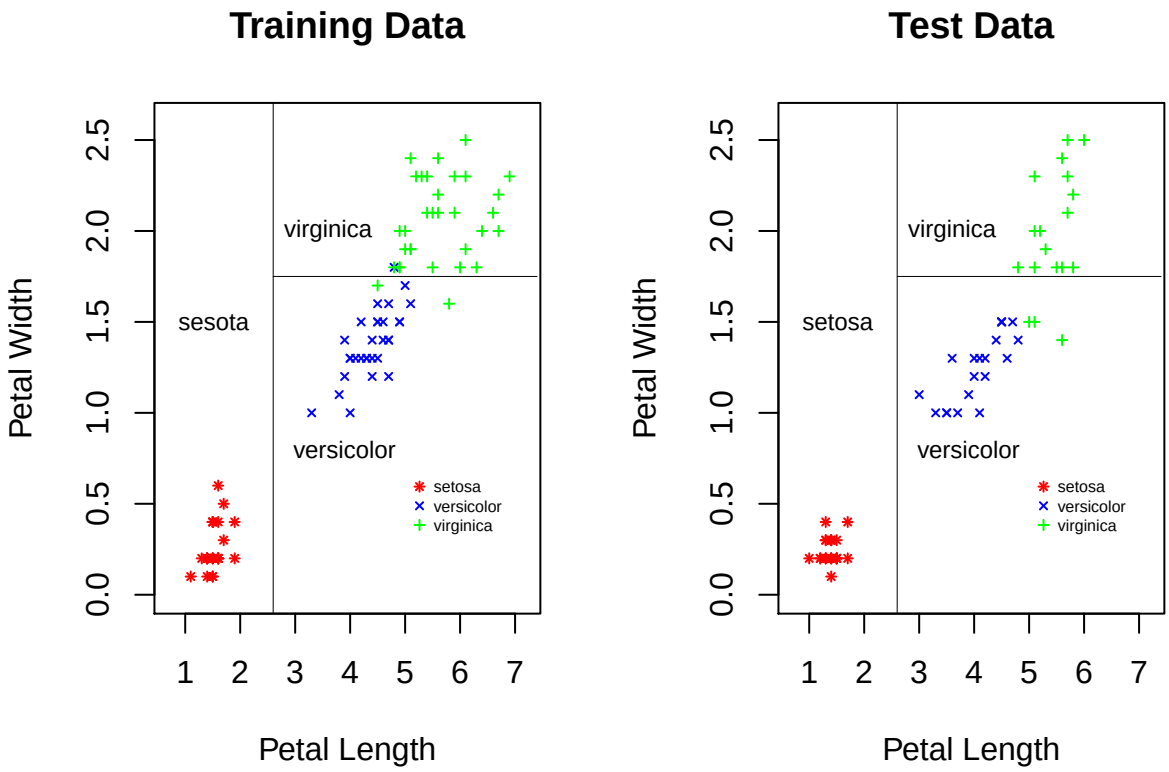




Codes

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

## create the 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
treeModel <- rpart(Species ~ ., data = trainData)

## Use the tree model to predict the class of the test data
predTrainClass <- predict(treeModel, newdata = trainData, type = "class")
predTestClass <- predict(treeModel, newdata = testData, type = "class")

## Find out the performance of the decision tree
table(predTrainClass, trainData$Species)
table(predTestClass, testClass)
```

Decision tree prediction visualization



Add some challenge

Advantages of decision tree

Easy to interpret

Problem with the decision tree

Lower prediction accuracy

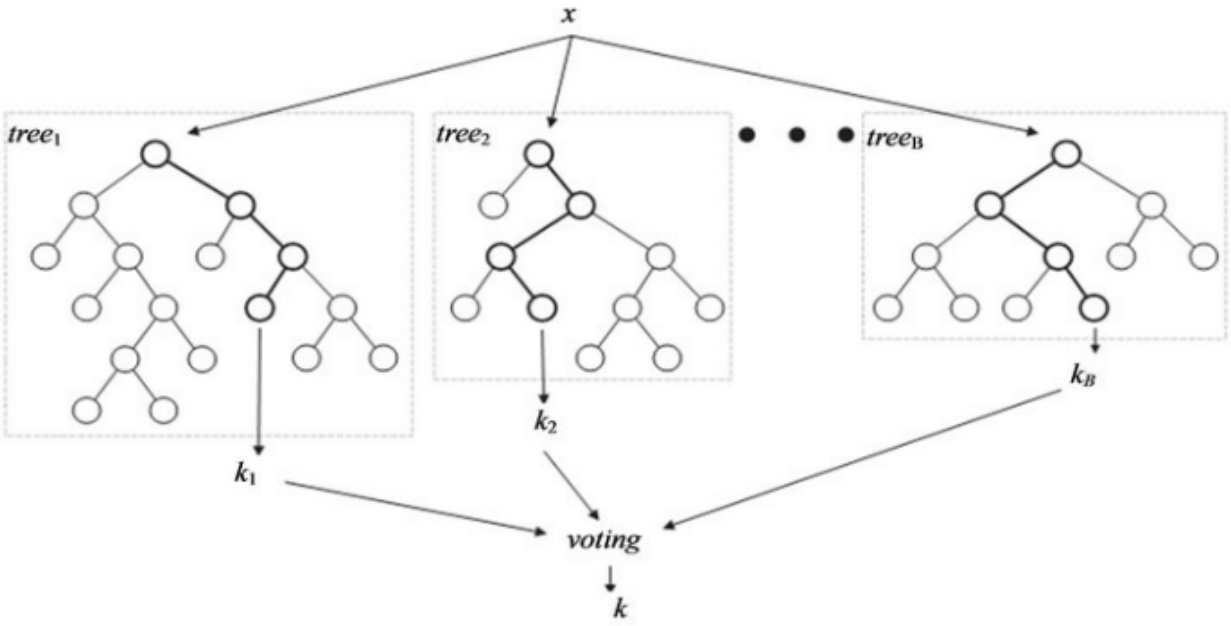
Solution

Aggregate many decision trees (bagging, random forest, boosting)

random Forest

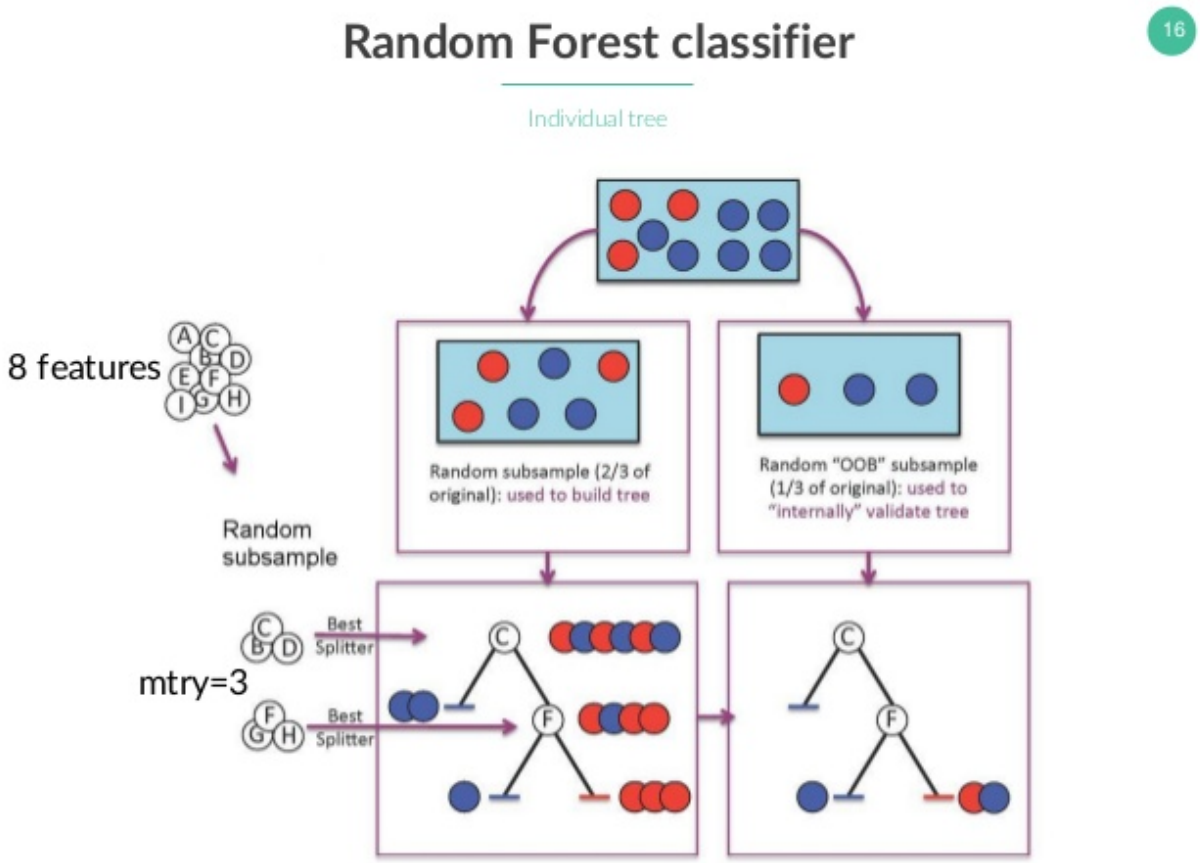
Need to decorrelate the trees. Making it more accurate

ntree



mtry

Decorrelate the trees a random sample of m predictors is chosen as split candidates from the full set of p predictors.

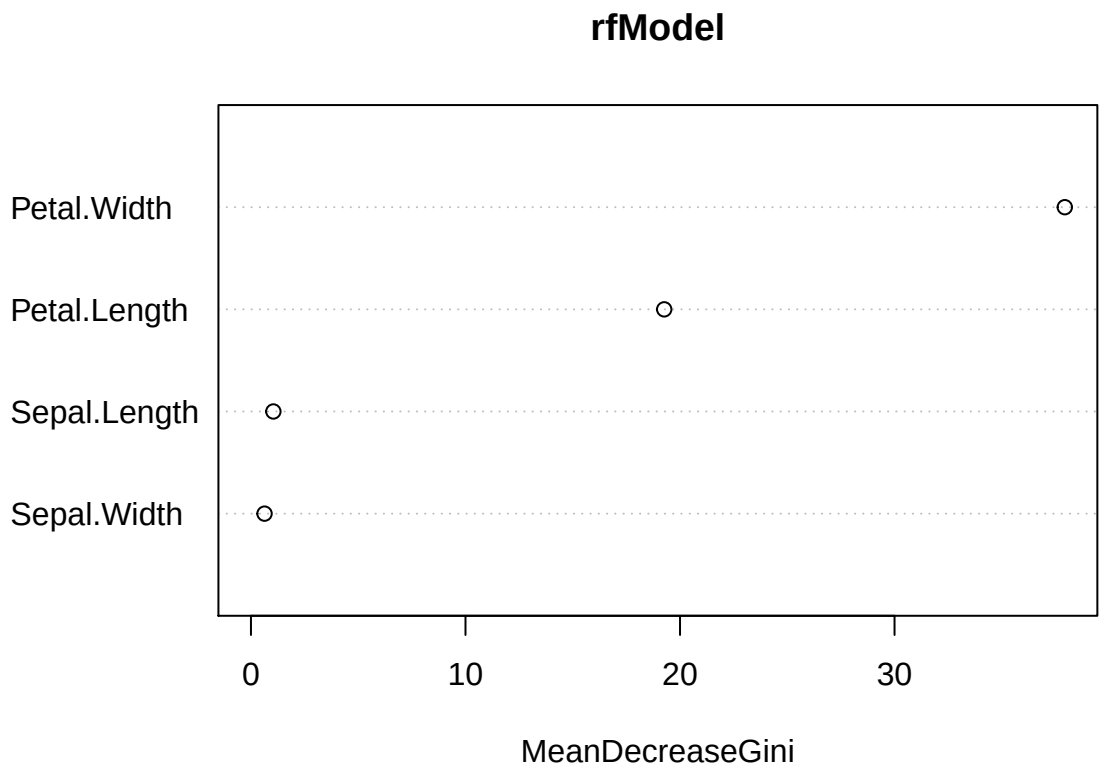


random forest example

```
library(randomForest)
rfModel <- randomForest(Species ~ ., data=trainData, mtry=3, ntree=15)
predClass <- predict(rfModel, newdata = testData)
table(predClass, testClass)
```

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

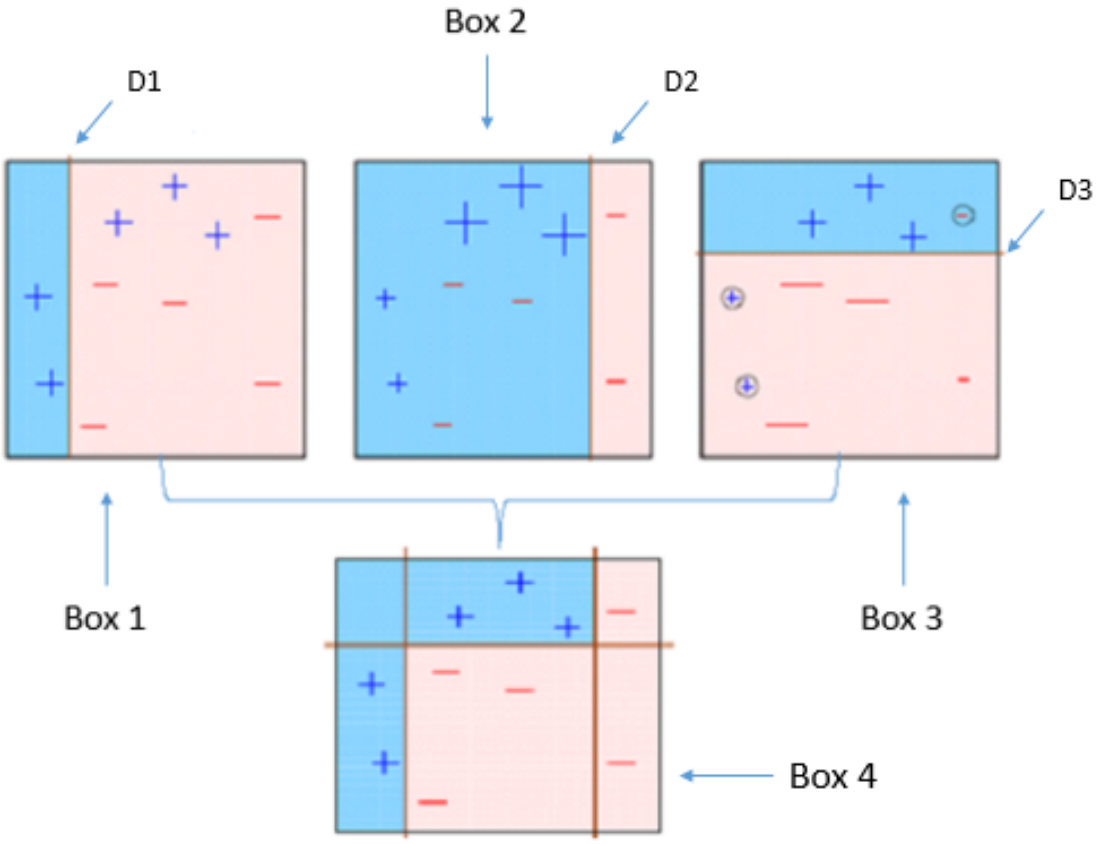
```
varImpPlot(rfModel)
```



Add some challenge

Boosting

Illustration



example

```
#gbmModel <- gbm(Species ~ ., data = trainData, distribution = "multinomial",
#                n.trees = 20)
#predict(gbmModel, newdata = testData, single.tree = TRUE, n.trees = 20, type = "response")


# boost.boston=gbm(medv~.,data=Boston[train,],distribution= # #"gaussian",n.trees=5000,interaction.dept
```

The argument n.teees = 5000 indicates that we want 5000 trees, and the option interaction.depth = 4 limits the depth of each tree

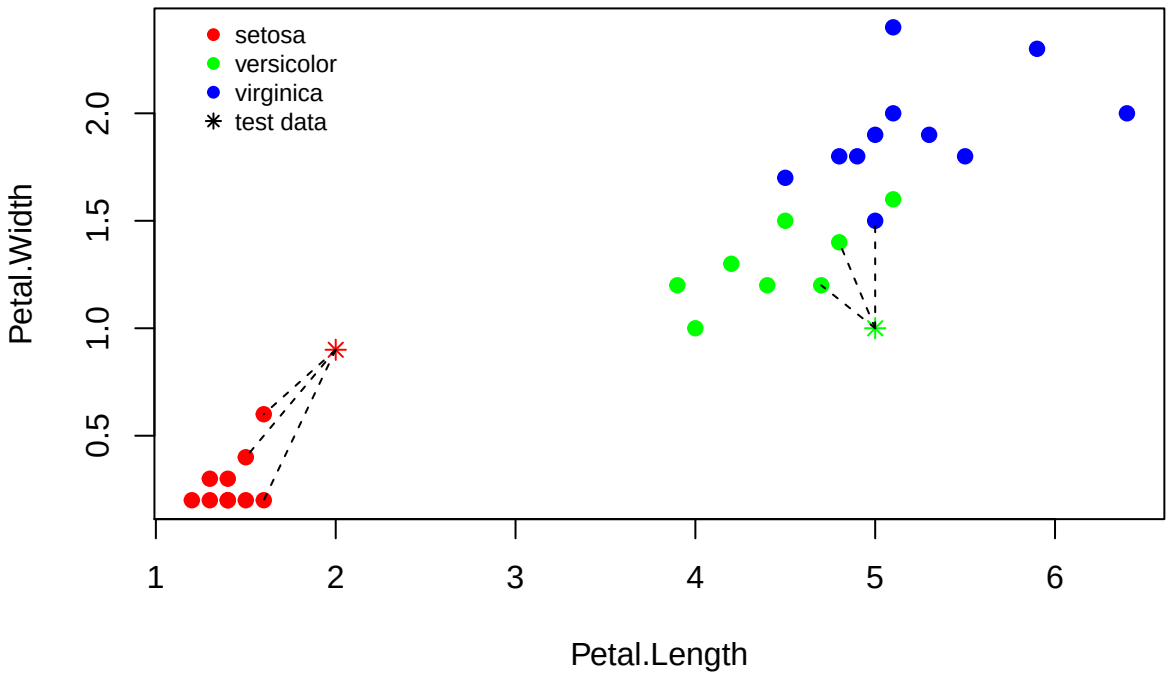
knn

make sure, you scale the data and also with an example tell why it is important to scale the data

Working principle

Members of a given class have similar characteristics. So, a given observation is assigned the class of its nearest neighbours (number of nearest neighbour to be decided by the user)

Example



code

```
library(class)
myIris <- iris[,3:5]

inTrain <- sample(c(TRUE, FALSE), size = nrow(myIris), replace = TRUE, prob = c(0.8,0.2))
trainData <- myIris[inTrain,1:2]
trainClass <- myIris[inTrain,3]
testData <- myIris[!inTrain,1:2]
testClass <- myIris[!inTrain,3]

predClass <- knn(trainData, testData, cl = trainClass, k = 3)
table(predClass, testClass)
```

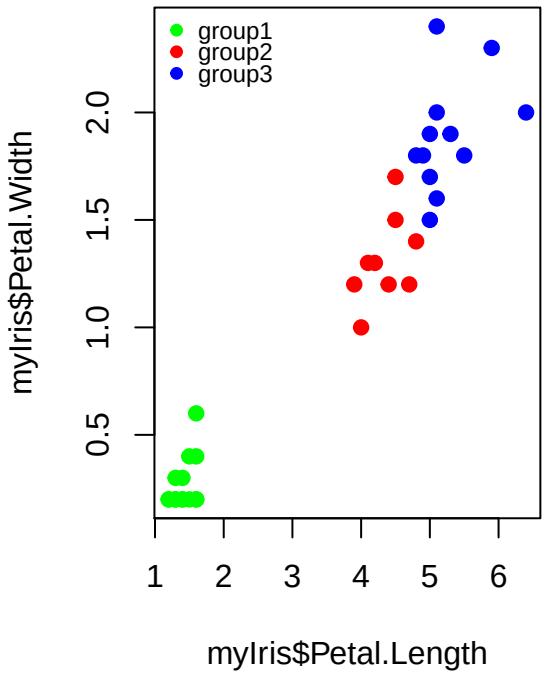
```
##             testClass
```

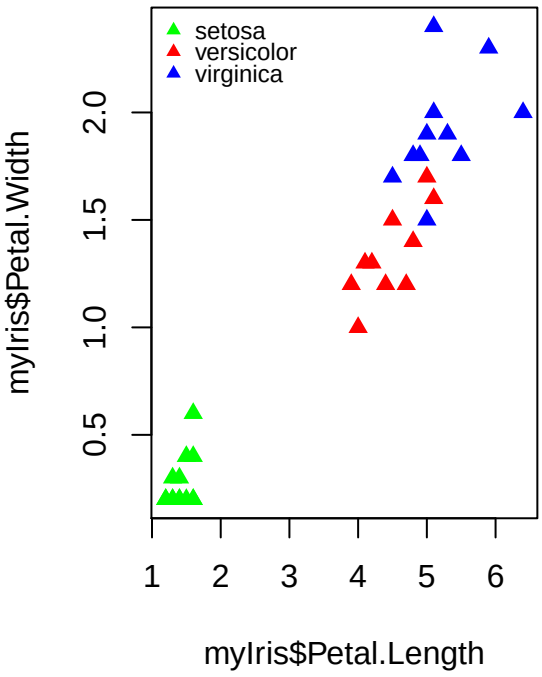
```
## predClass    setosa versicolor virginica
##   setosa         11          0         0
##   versicolor      0          8         0
##   virginica       0          1         9
```

clustering example

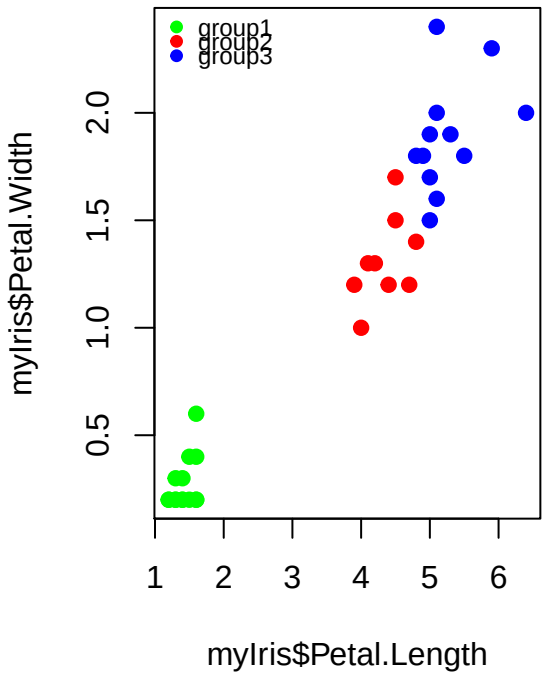
kmeans clustering

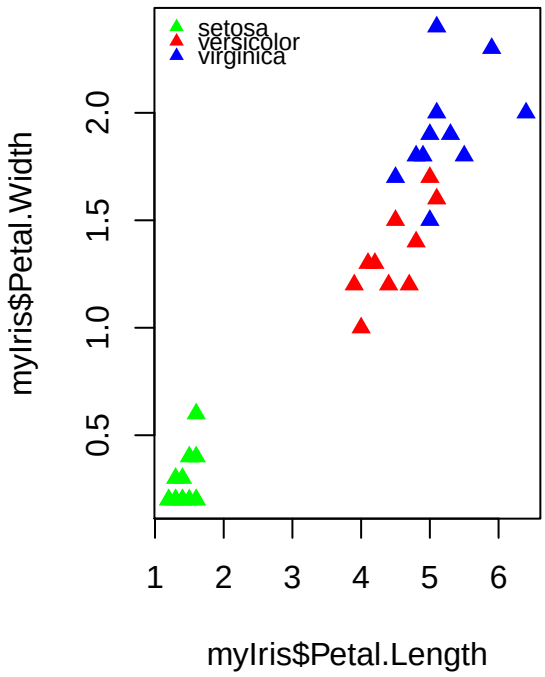
```
##              group
## predGroupC    setosa versicolor virginica
##    setosa         14          0         0
##    versicolor      0          8         1
##    virginnica      0          2        10
```

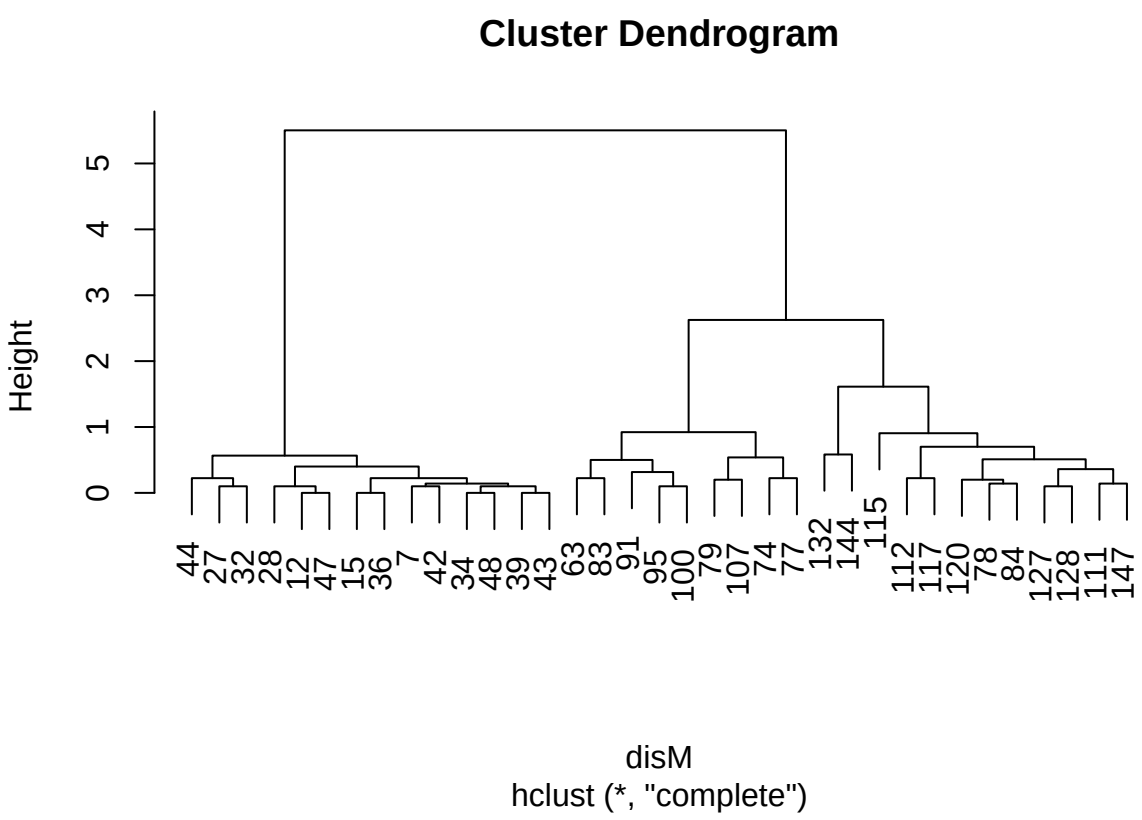


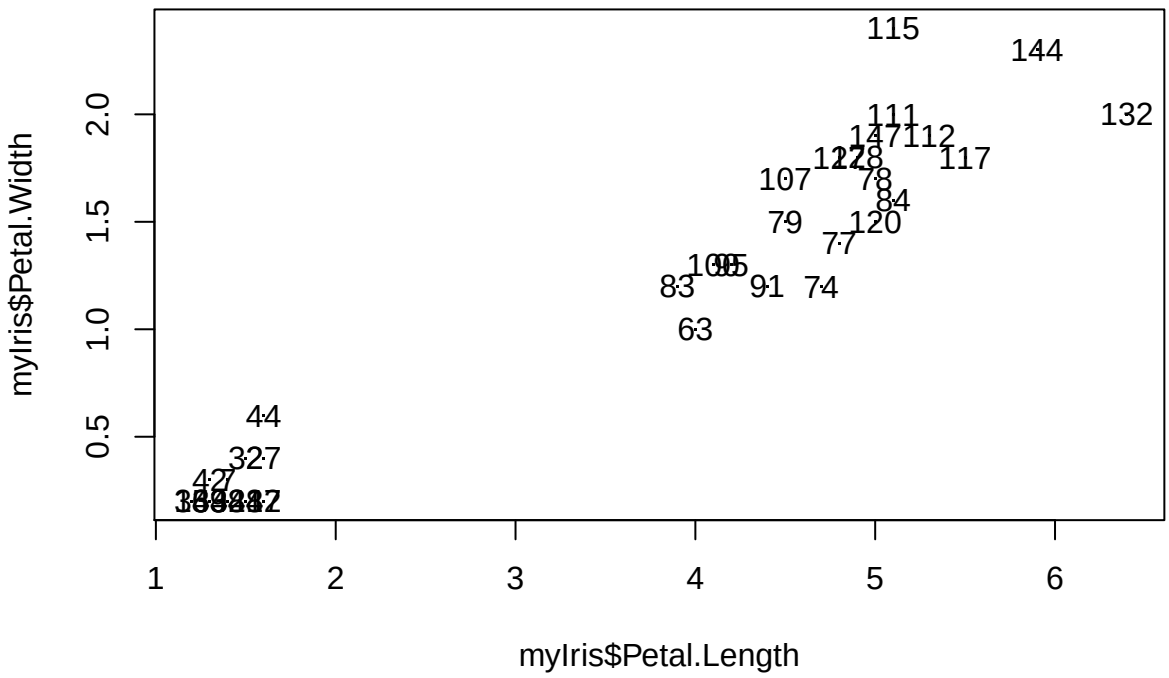


hierarchichal clustering

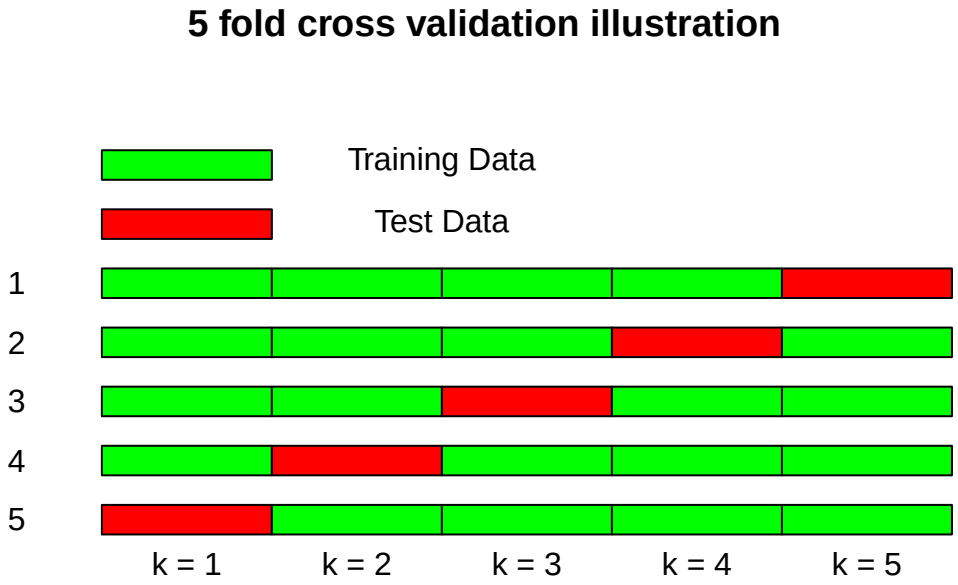





Cluster Dendrogram
Height
0
1
2
3
4
5
44
27
32
28
12
47
15
36
7
42
34
48
39
43
63
83
91
95
100
79
107
74
77
132
144
115
112
117
120
78
84
127
128
111
147
disM
hclust (*, "complete")


myIris$Petal.Width
0.5
1.0
1.5
2.0
1
2
3
4
5
6
myIris$Petal.Length

Cross-validation



baye's theorem

```
#head(Titanic)
```

svm

```
#svmModel <- svm()
```