Machine Learning

Ravi Kumar Tiwari

14 June 2016

```
library(caret)
library(rpart.plot)
library(rattle)
library(calibrate)
library(randomForest)
library(e1071)
library(class)
library(knitr)
library(party)
library(ada)
library(gbm)
library(leaps)
```

Regression

Used for predicting continuous variable

1. Code to Build the model object

```
lmModel <- lm(mpg ~ wt, data = mtcars)
```

2. Code to obtain the model parameters

```
sumModel <- summary(lmModel)
sumModel$coefficients
```

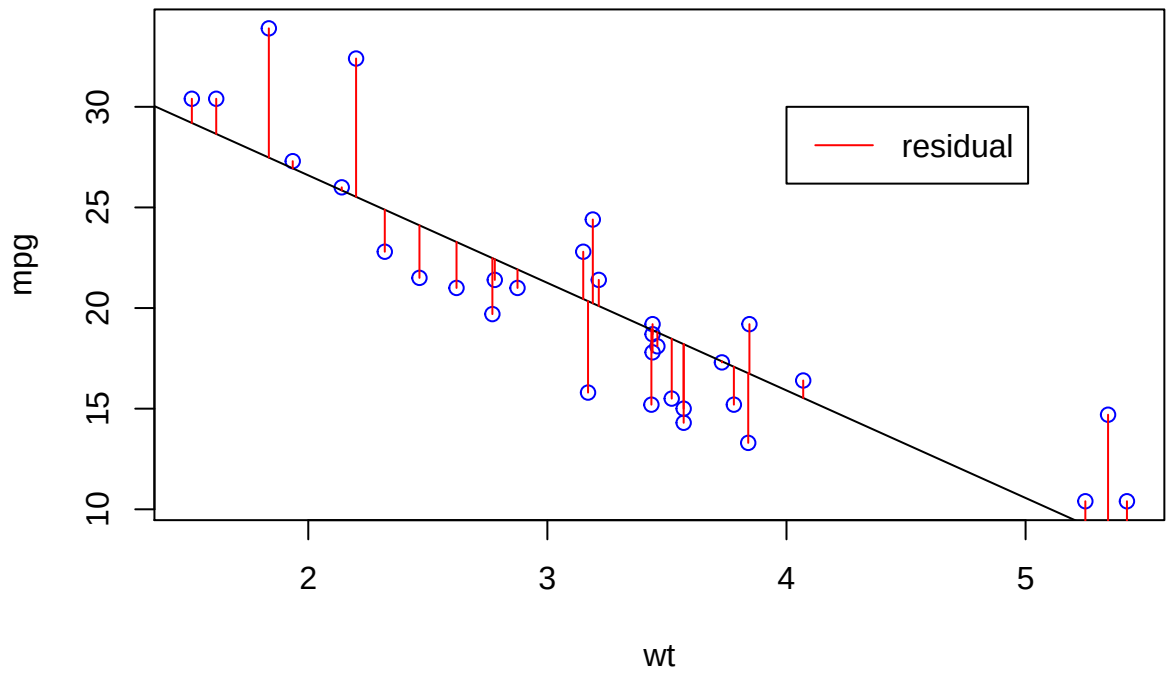
```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 37.285126   1.877627 19.857575 8.241799e-19
## wt          -5.344472   0.559101 -9.559044 1.293959e-10
```

3. Prediction of response using the regression model

```
predValue <- predict(lmModel, data.frame(wt = mtcars$wt))
```

4. Assessing the accuracy of the model

- Visual Inspection



- R-squared value

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

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

- F-statistics

```
sumModel$fstatistic
```

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

5. Code to build linear regression model using multiple 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
```

6. Code to do subset selection

identifies the best model that contains a given number of predictors, where best is quantified using RSS

```r
fwdSelection <- regsubsets(mpg ~ ., data = mtcars, method = "forward")
sumFwdSel <- summary(fwdSelection)
names(sumFwdSel)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```r
sumFwdSel$outmat
```

```
##          cyl disp hp  drat wt  qsec vs  am  gear carb
## 1  ( 1 ) " " " "  " " " "  "*" " "  " " " " " "  " "
## 2  ( 1 ) "*" " "  " " " "  "*" " "  " " " " " "  " "
## 3  ( 1 ) "*" " "  "*" " "  "*" " "  " " " " " "  " "
## 4  ( 1 ) "*" " "  "*" " "  "*" " "  " " "*" " "  " "
## 5  ( 1 ) "*" " "  "*" " "  "*" "*"  " " "*" " "  " "
## 6  ( 1 ) "*" "*"  "*" " "  "*" "*"  " " "*" " "  " "
## 7  ( 1 ) "*" "*"  "*" "*"  "*" "*"  " " "*" " "  " "
## 8  ( 1 ) "*" "*"  "*" "*"  "*" "*"  " " "*" "*"  " "
```

```r
sumFwdSel$rsq
```

```
## [1] 0.7528328 0.8302274 0.8431500 0.8490314 0.8580721 0.8659105 0.8675989
## [8] 0.8684657
```

```r
which.max(sumFwdSel$adjr2)
```

```
## [1] 6
```

```r
coef(fwdSelection,6)
```

```
## (Intercept)         cyl        disp          hp          wt        qsec
## 20.05169952 -0.50206577  0.01396099 -0.01956054 -3.99773180  0.81017782
##          am
##  2.94074955
```

7. Challenge

- Use backward selection model to find the best model for mpg

Tree based algorithm

Used both for classification and regression

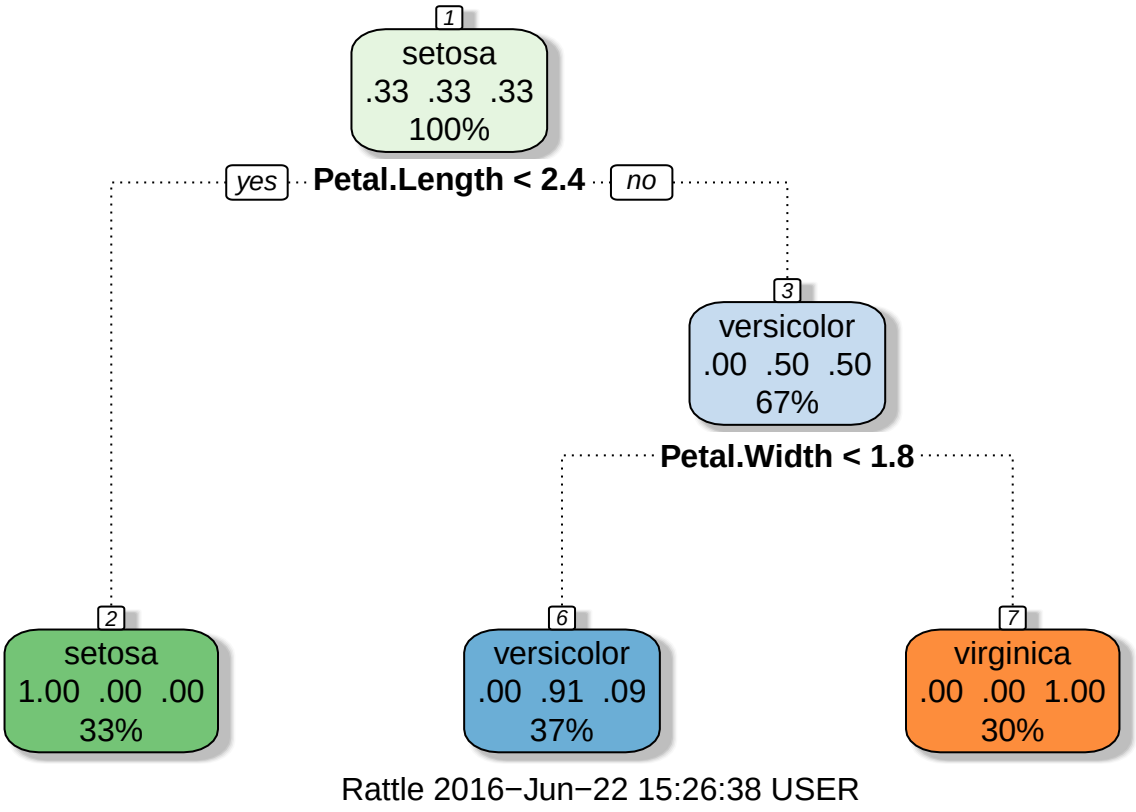
Problem Description

Use Iris dataset to predict the Species based on Sepal.Length, Sepal.Width, Petal.Length, Petal.Width

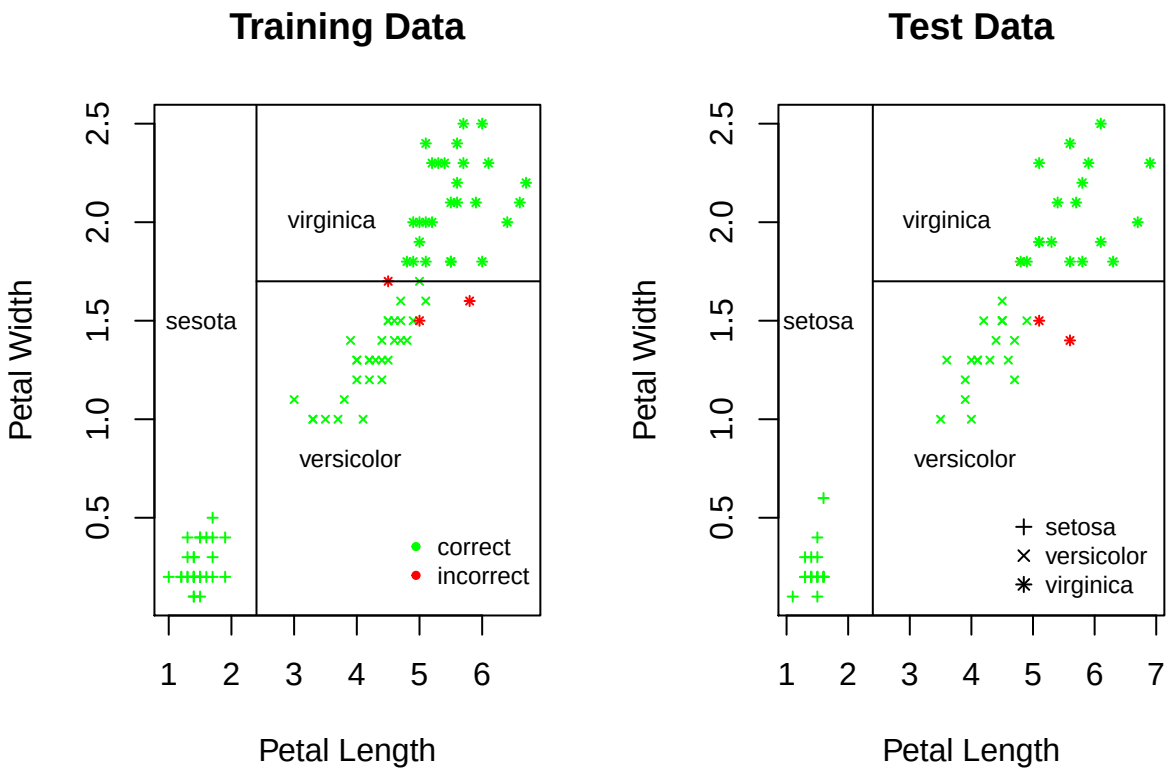
```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Decision tree: splitting rule



Decision tree: partitioned area



Create data partition

```
set.seed(100)
inTrain <- createDataPartition(iris$Species, p = 0.6, list = FALSE)
trainData <- iris[inTrain,]
testData <- iris[-inTrain,1:4]
testClass <- iris[-inTrain,5]
```

Build a decision tree model and use it for prediction on test data set

```
treeModel <- train(Species ~ ., data = trainData, method = "rpart")
predClass <- predict(treeModel, newdata = testData)
cMatrix <- confusionMatrix(predClass, testClass)
cMatrix$table
```

```
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0         19         2
##   virginica       0          1        18
```

Look at what are the important variables

```
varImp(treeModel)
```

```
## rpart variable importance
##
##              Overall
## Petal.Width   100.00
## Petal.Length   89.53
## Sepal.Length   18.24
## Sepal.Width     0.00
```

Add some challenge

Advantages of decision tree

Easy to interpret

Problem with the decision tree

Lower prediction accuracy

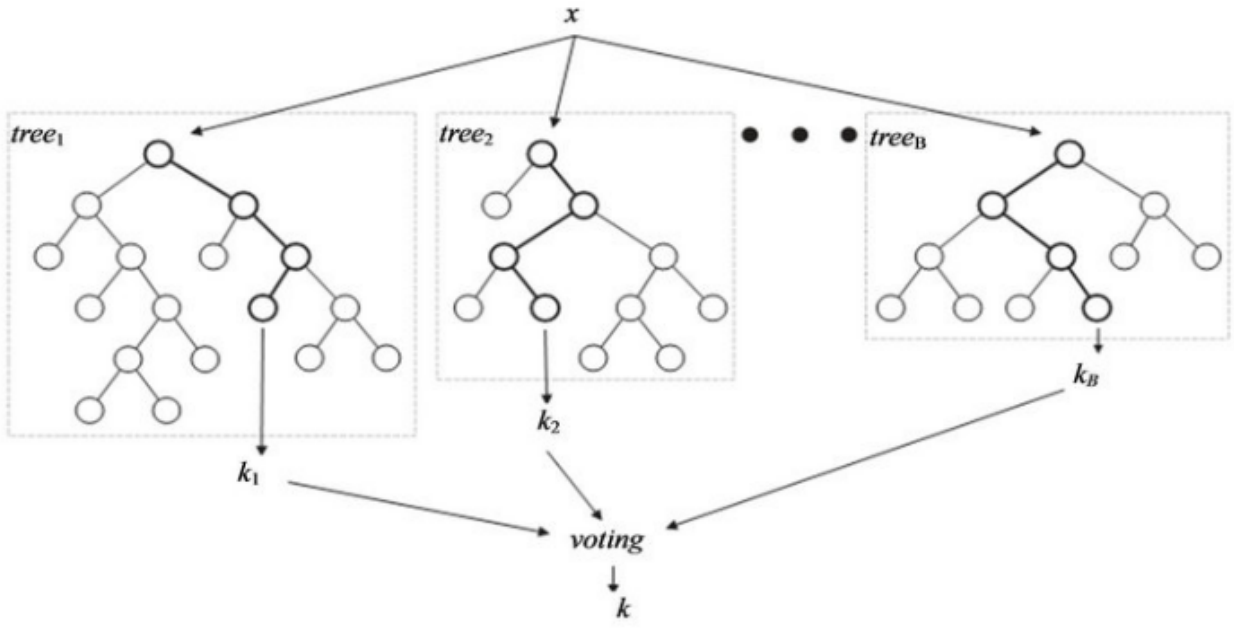
Solution

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

random Forest

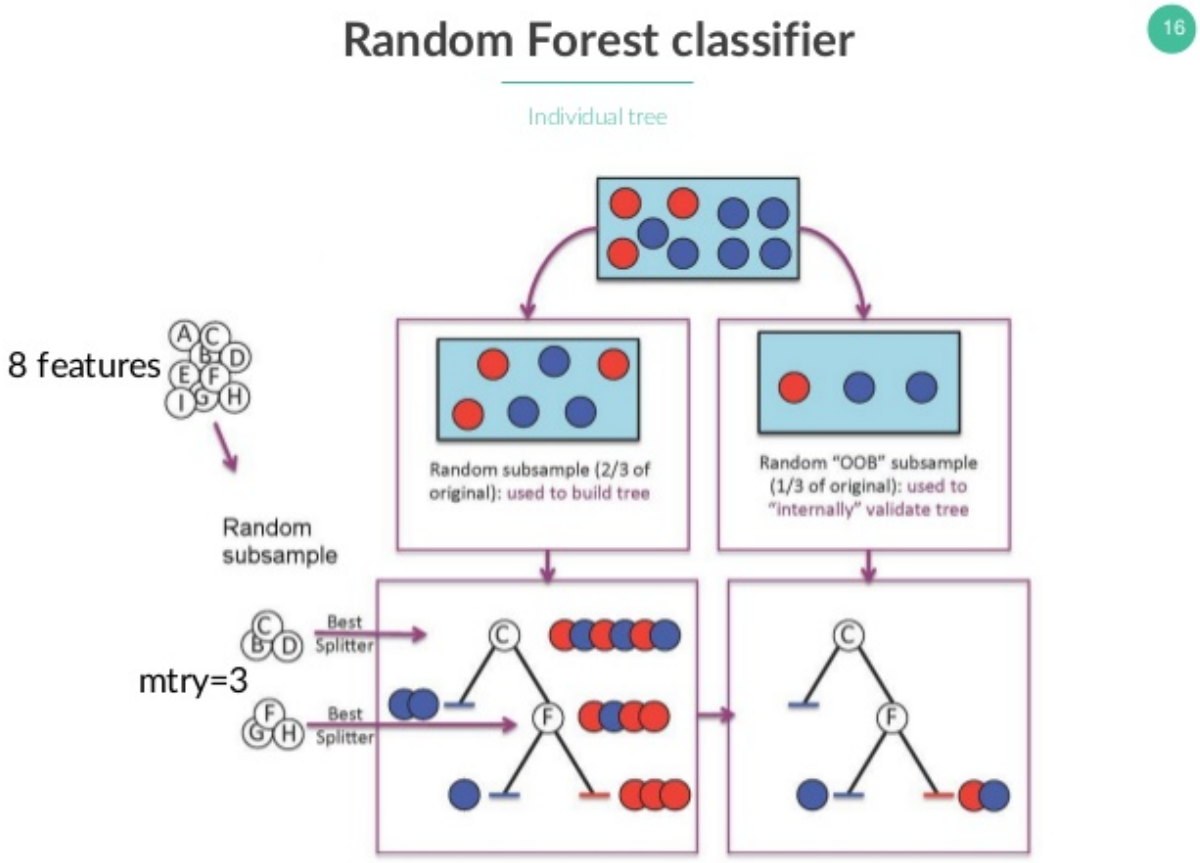
Need to decorrelate the trees. Making it more accurate

ntree



mtry

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

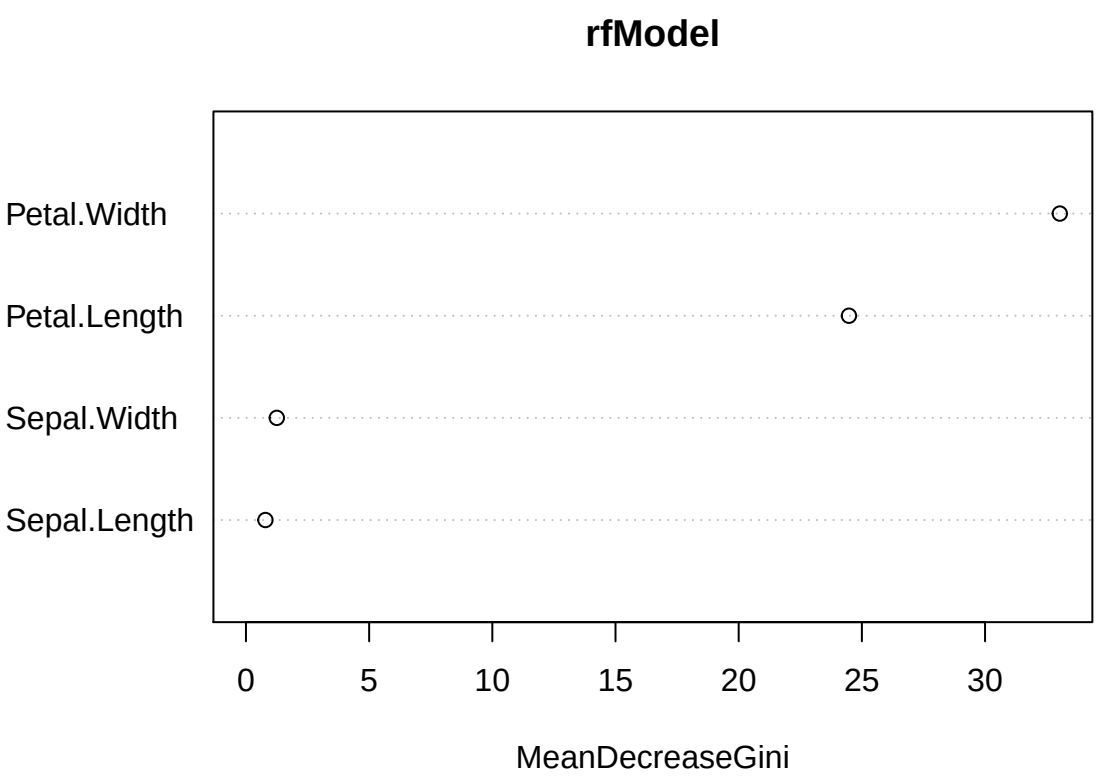


random forest example

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

```
##             testClass
## predClass    setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0         19         2
##   virginica       0          1        18
```

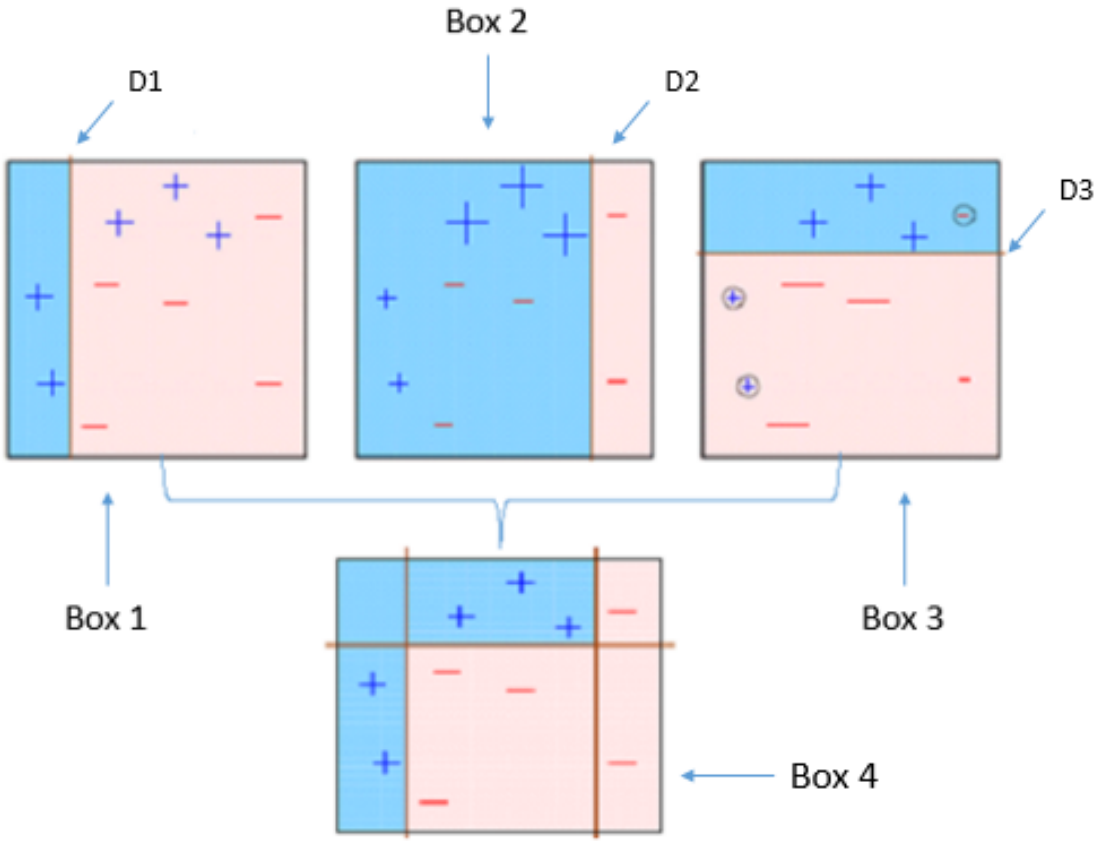
```
varImpPlot(rfModel)
```


rfModel
Petal.Width
Petal.Length
Sepal.Width
Sepal.Length
0
5
10
15
20
25
30
MeanDecreaseGini

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

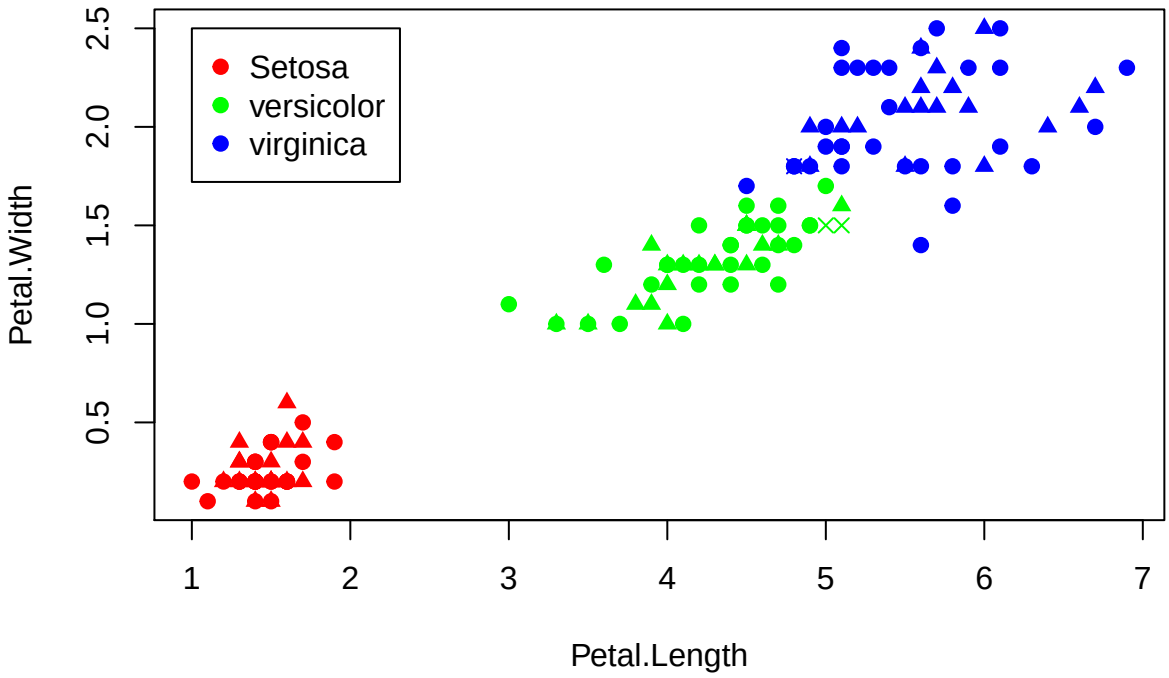
```
myIris <- iris[,3:5]
head(myIris)
```

```
##   Petal.Length Petal.Width Species
## 1          1.4         0.2  setosa
## 2          1.4         0.2  setosa
## 3          1.3         0.2  setosa
## 4          1.5         0.2  setosa
## 5          1.4         0.2  setosa
## 6          1.7         0.4  setosa
```

```
inTrain <- createDataPartition(myIris$Species, p = 0.6, list = FALSE)
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)
```

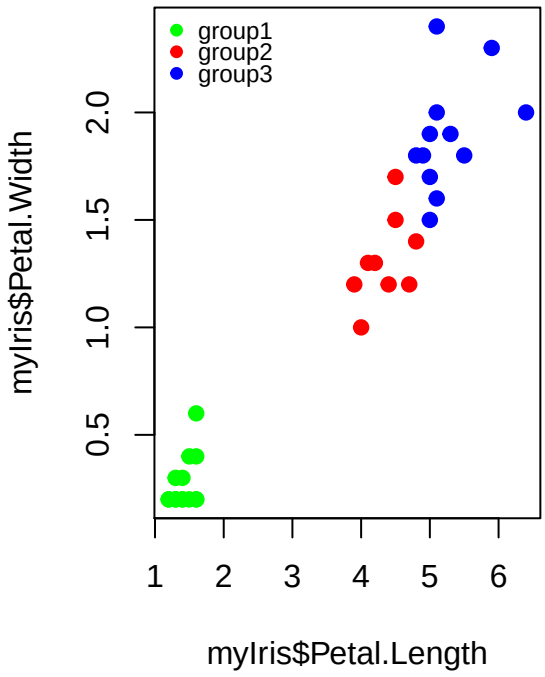
```
##             testClass
## predClass    setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0         19         2
##   virginica       0          1        18
```

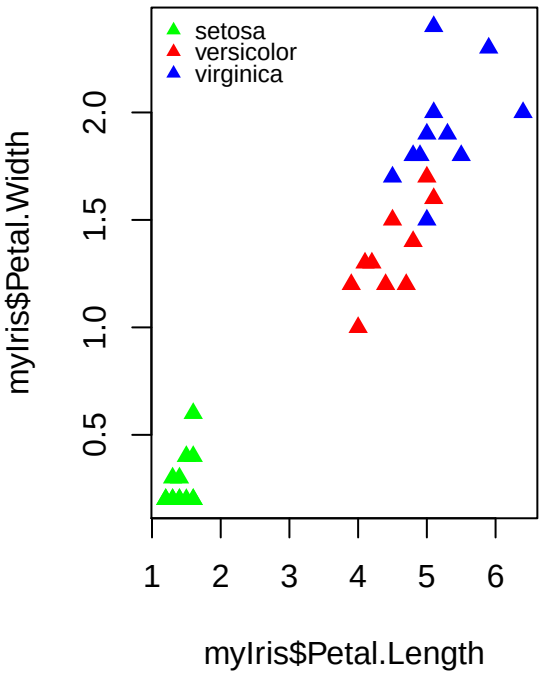

Setosa
versicolor
virginica
Petal.Width
Petal.Length
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7

clustering example

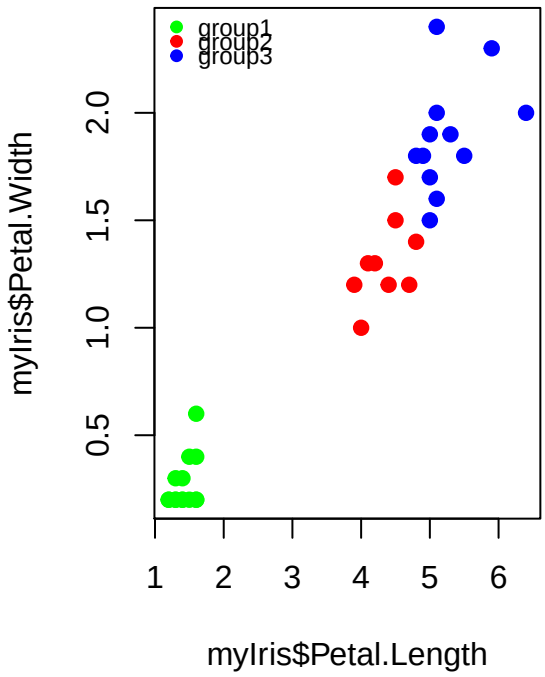
kmeans clustering

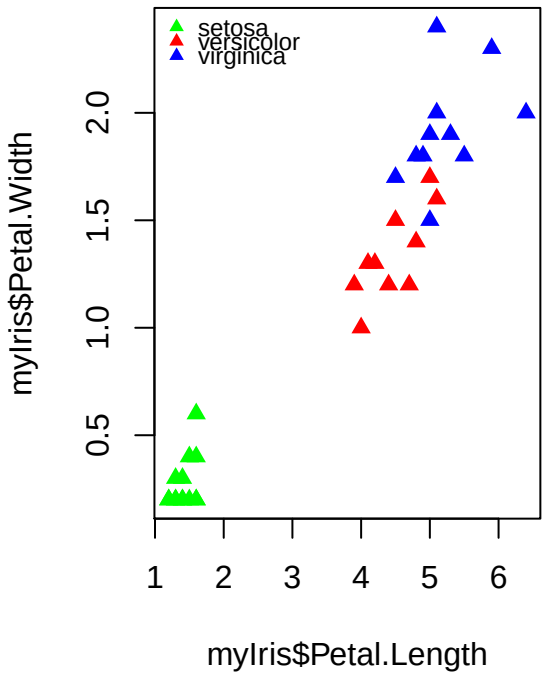
```
##              group
## predGroupC    setosa versicolor virginica
##    setosa         14          0         0
##    versicolor      0          8         1
##    virginnica      0          2        10
```

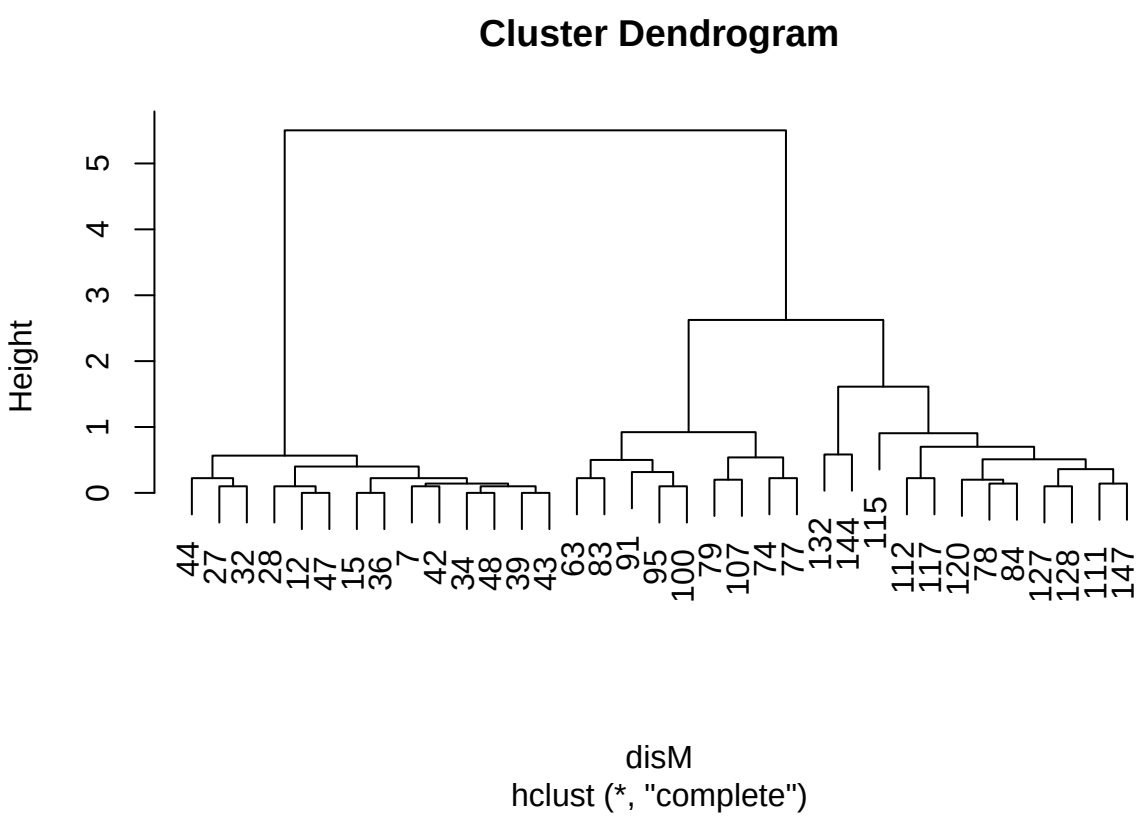


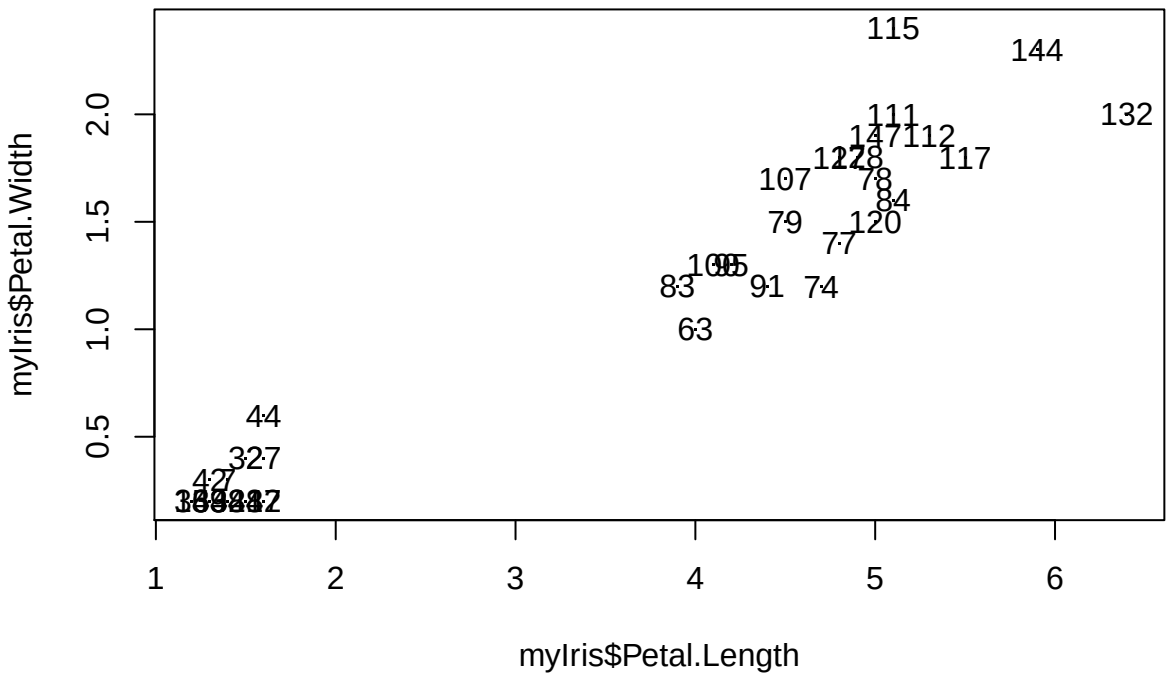


hierarchichal clustering

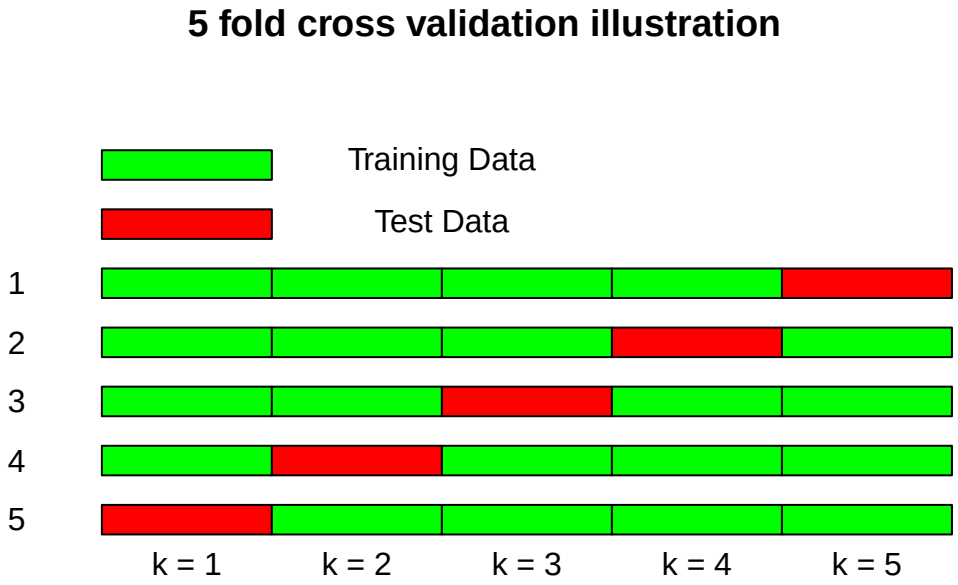





Cluster Dendrogram
Height
0
1
2
3
4
5
44
27
32
28
12
47
15
36
7
42
34
48
39
43
63
83
91
95
100
79
107
74
77
132
144
115
112
117
120
78
84
127
128
111
147
disM
hclust (*, "complete")


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()
```