作业 06：Machine Learning

qinluoao

2017/8/11

1. 机器学习定义

机器学习研究的是如何赋予计算机在没有被明确编程的情况下仍能够学习的能力，让计算机“ learn from experience”，能自动获取与整合知识的一套系统。

2. 机器学习工作流程（类似 **cross-industry standard process for data mining)**

2.0 业务理解

定义研究问题，梳理业务流程、逻辑、指标体系，明确分析目标。

2.1 数据获取

确定取数目的、数据用途、数据量、时间范围（周期）。对数据进行探索，包括数据描述性统计、可视化探索、数据质量探索（数据缺失值、异常值、重复值情况）

2.2 数据预处理

数据整理：包括新建列、数据合并、构造子集、数据排序、数据分组、数据规范化等。数据清洗：缺失值、异常值、重复值处理，数据格式转换等。数据预处理常涉及的包：字符处理 stringr 包、日期变换 lubridate 包、数据操作的 dplyr 包、数据清理的 tidyr 包、可视化 ggplot2 包等，最终形成满足分析需求的多维度、格式正确的宽表。

2.3 特征选择

特征选择是指选择获得相应模型和算法最好性能的特征集，决定了机器学习的上限。特征选择的作用：• Improve the accuracy of a machine learning algorithm • Boost model performance for high-dimensional data sets • Improve model interpretability • Prevent overftting 常见的处理有：

a. 数据降维度

降低数据维度，常用主成分分析、因子分析、奇异值分解等方法。

b. 显著性检验

目的是剔除显著性低的自变量，如删除常数自变量、删除方差极小的自变量

c. 强相关性检验

指的是某自变量与其他自变量有很强的相关性

d. 多重共线性检验

剔除自变量中多重共线性的变量，用 VIF(方差膨胀因子) 来检验多重共线性问题、偏相关系数来处理

2.4 建模及模型训练

根据想要获得的数据洞察、数据的大小、算法的内存需求、算法计算速度、准确度和易用性等来选择模型。模型建立后需要对模型性能进行评估，常采用交叉验证思想，交叉验证形式有 K-fold cross-validation 验证和留一验证等。

2.5 模型评估

通过不断对比尝试、反复迭代建立最优的模型。对分类算法模型评估的方法有：混淆矩阵 confusion matrix 和 ROC 曲线。

3. 机器学习算法

3.1 算法分类

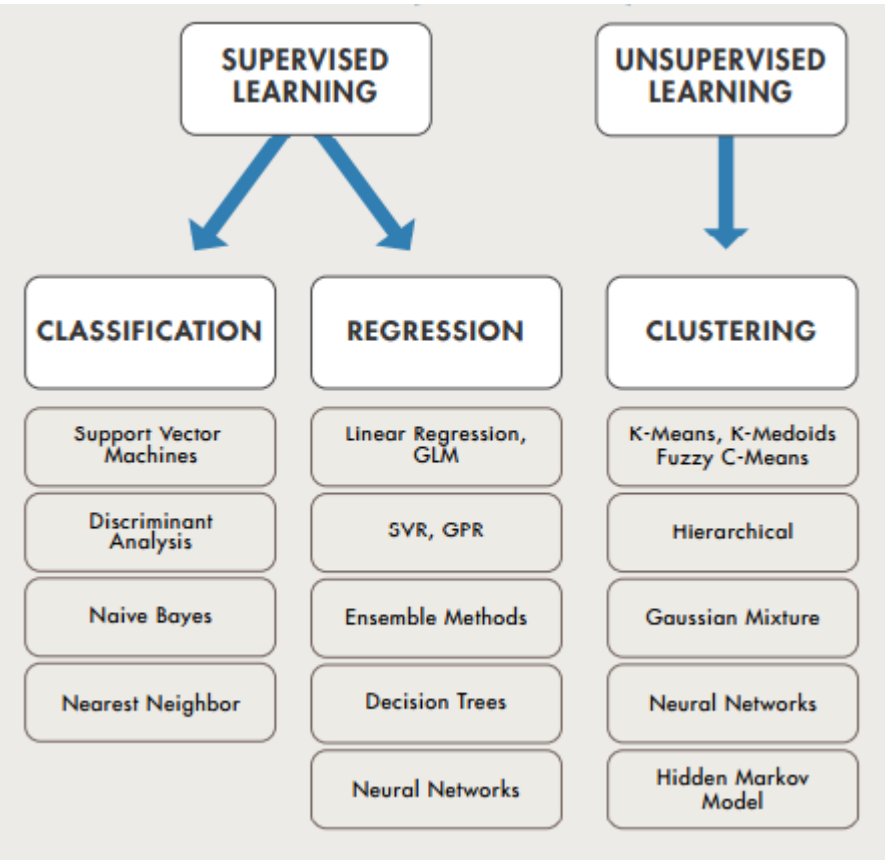


Figure 1: 有监督和无监督学习算法

回归算法: 学习预测连续

分类算法: 学习决策边界

聚类算法: 学习潜在模式和事物内在结构

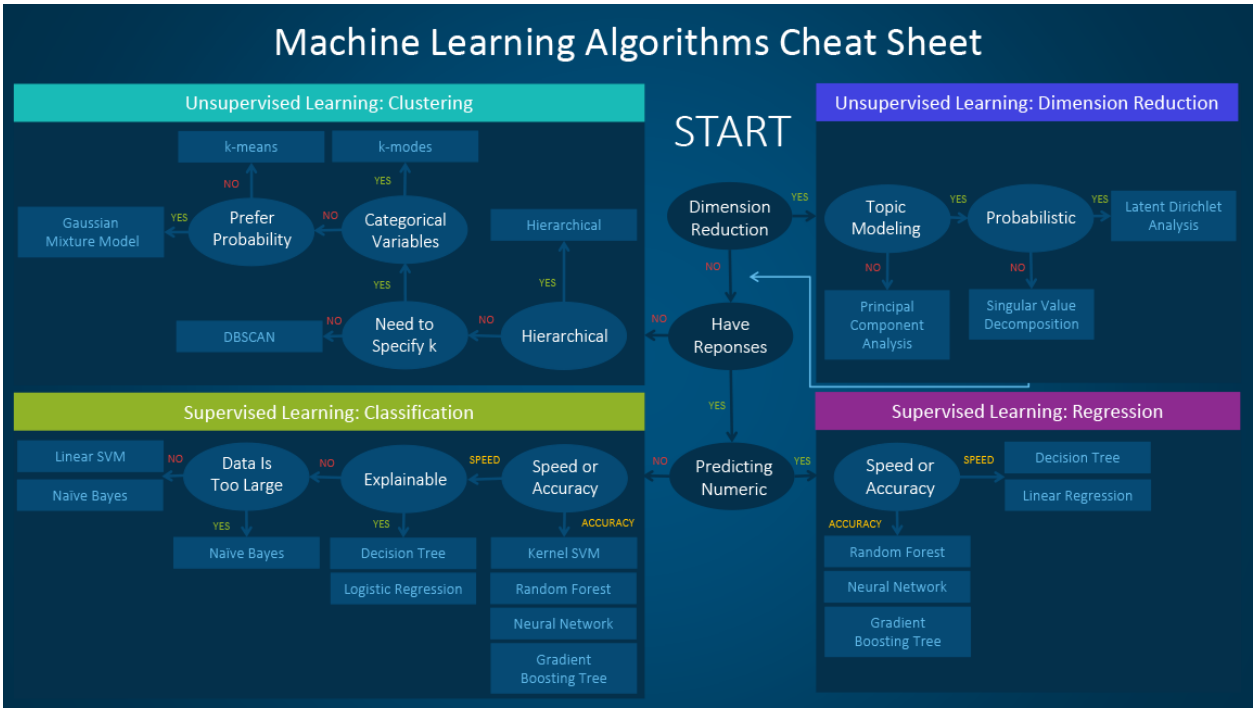


Figure 2: 机器学习算法选择流程图

3.2 如何选择算法

3.3 算法优化

目的是提高模型准确性，避免过拟合，过拟合时模型无法区分有效数据和噪音。涉及特征转换、超参数调整（识别最佳模型参数集的过程，控制机器学习算法如何匹配数据模型）等内容。

4. 应用案例

```
library(caret)
```

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

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

```
#1. 数据获取：鸢尾花数据集
data("iris")
iris.data <- iris
#2.1 数据探索
str(iris.data)
```

```
## '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 ...
```

```r
head(iris.data)
```

```
##   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
```

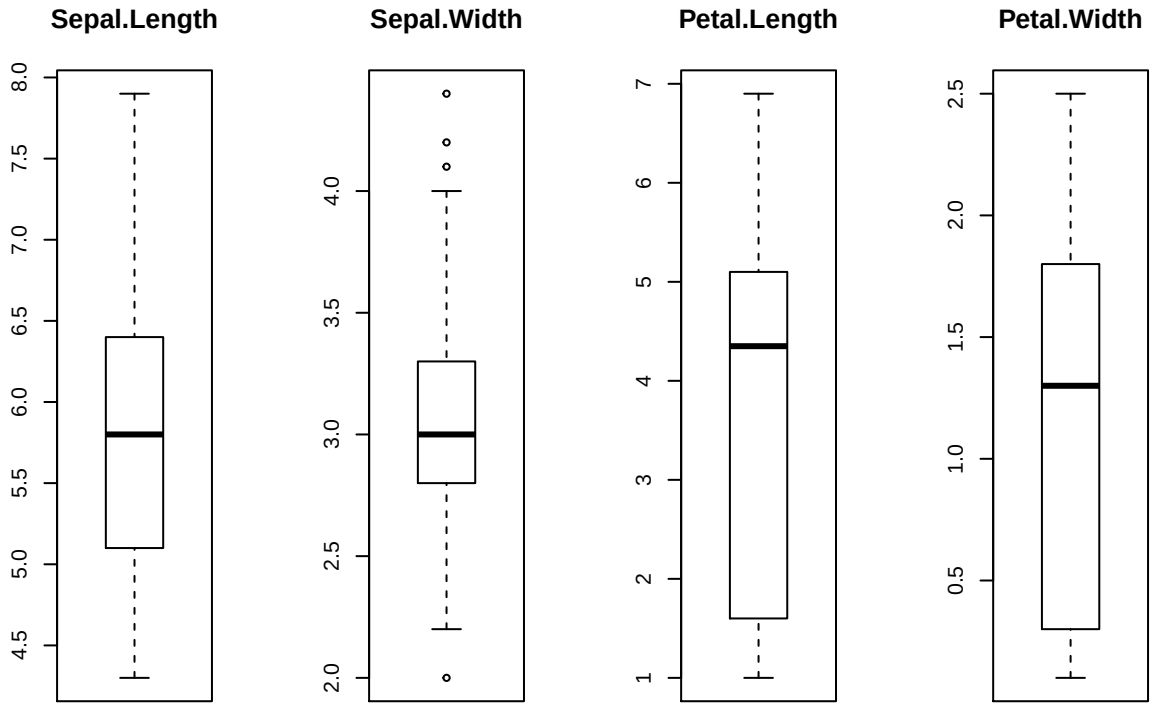
```r
summary(iris.data)  # 查看数据集摘要信息
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

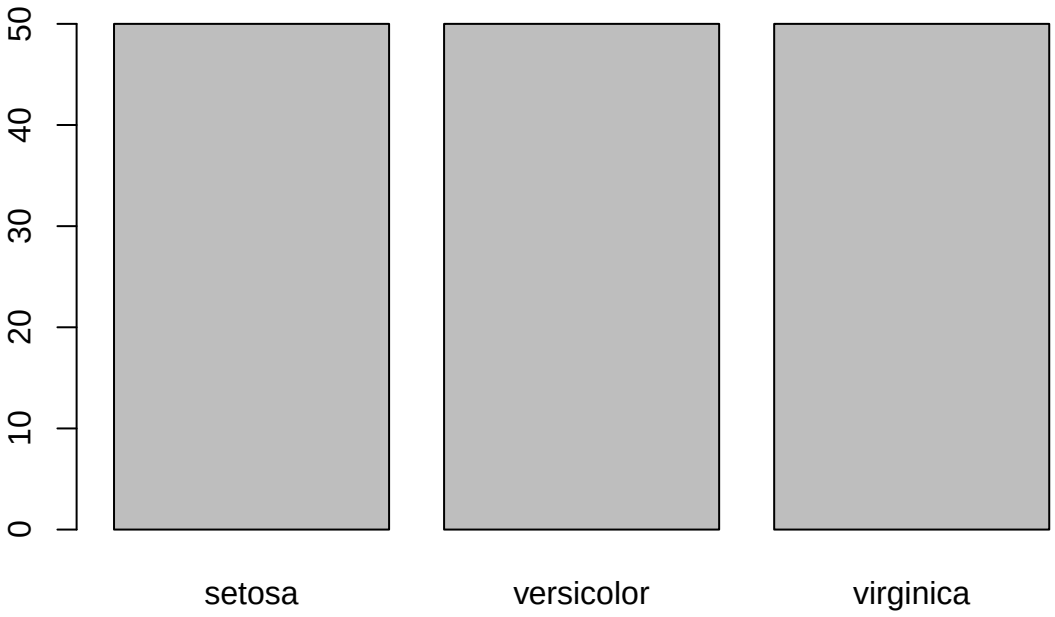
```r
attach(iris.data)
percentage <- prop.table(table(Species))*100
freq <-table(Species)
cbind(freq,percentage)
```

```
##            freq percentage
## setosa       50   33.33333
## versicolor   50   33.33333
## virginica    50   33.33333
```

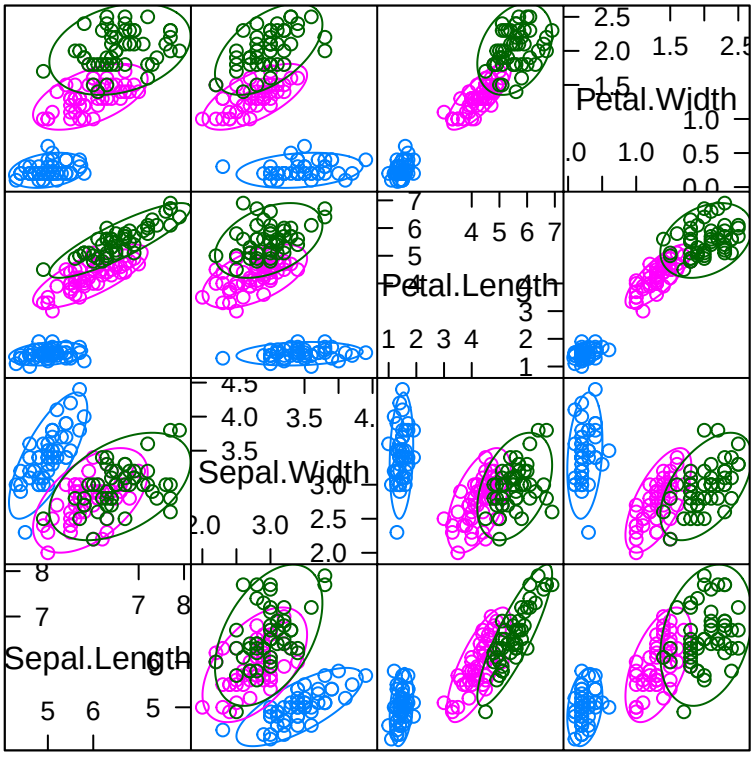
```r
#2.2 可视化描述，理解变量分布
#2.2.1 单变量可视化
## 花瓣和花萼四个数值变量分布情况的箱线图
input.val <- iris.data[,1:4]
par(mfrow=c(1,4))
for(i in 1:4){
  boxplot(input.val[,i],main=names(iris.data)[i])
}
```




```
## 因子变量直方图
par(mfrow=c(1,1))
output.val <- iris.data[,5]
plot(output.val)
```

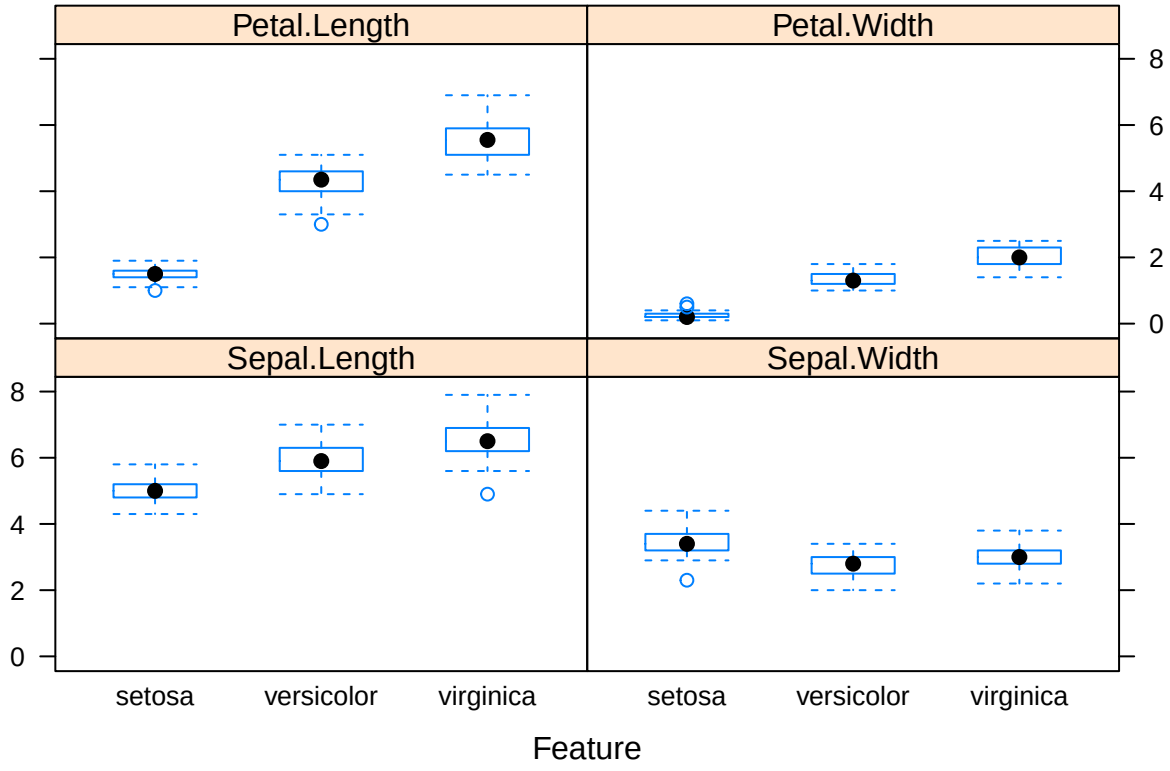



```
#2.2.2 多变量可视化--caret 包 featurePlot 绘图
library(ellipse)
featurePlot(x=input.val,y=output.val,plot="ellipse")
```



Scatter Plot Matrix

```
featurePlot(x=input.val, y=output.val, plot = 'box')
```




```
#3. 数据分析：训练集和测试集
# 获取原数据集的 80% 的行索引号，返回行索引的 0.8N*1 的 Resample1 矩阵
validation.index <- createDataPartition(iris.data$Species, p=0.80, list=FALSE)
head(validation.index)
```

```
##      Resample1
## [1,]         1
## [2,]         2
## [3,]         3
## [4,]         4
## [5,]         5
## [6,]         6
```

```
# 构建 80% 训练集和 20% 的测试集
train.data <-iris.data[validation.index,]
validation.data <- iris.data[-validation.index,]
#4. 建模
#4.1 交叉验证：分为 K-fold、留一验证，目的选择最优模型
control <- trainControl(method = 'cv',number = 10) #10-fold Cross-Validation
metric <-"Accuracy"
#4.2 构建模型-分类算法选择
#LDA 线性判别分类，CART 分类与回归树，RF 随机森林
#(1) 线性算法：e1071 包
library(e1071)
set.seed(7)
lda.model <- train(Species~.,data = train.data,method="lda",metric=metric,trControl=control)
```

```
## Loading required package: MASS
```

```
#(2) 非线性算法
library(rpart)
set.seed(7)
cart.model <- train(Species~., data=train.data, method="rpart", metric=metric, trControl=control)
#(3)Random Forest
library(randomForest)
```

```
## randomForest 4.6-12
```

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

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

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

```
set.seed(7)
rf.model <- train(Species~., data=train.data, method="rf", metric=metric, trControl=control)
#5 模型评估：选择最优算法
results.model <- resamples(list(lda=lda.model, cart=cart.model, rf=rf.model))
str(results.model)
```

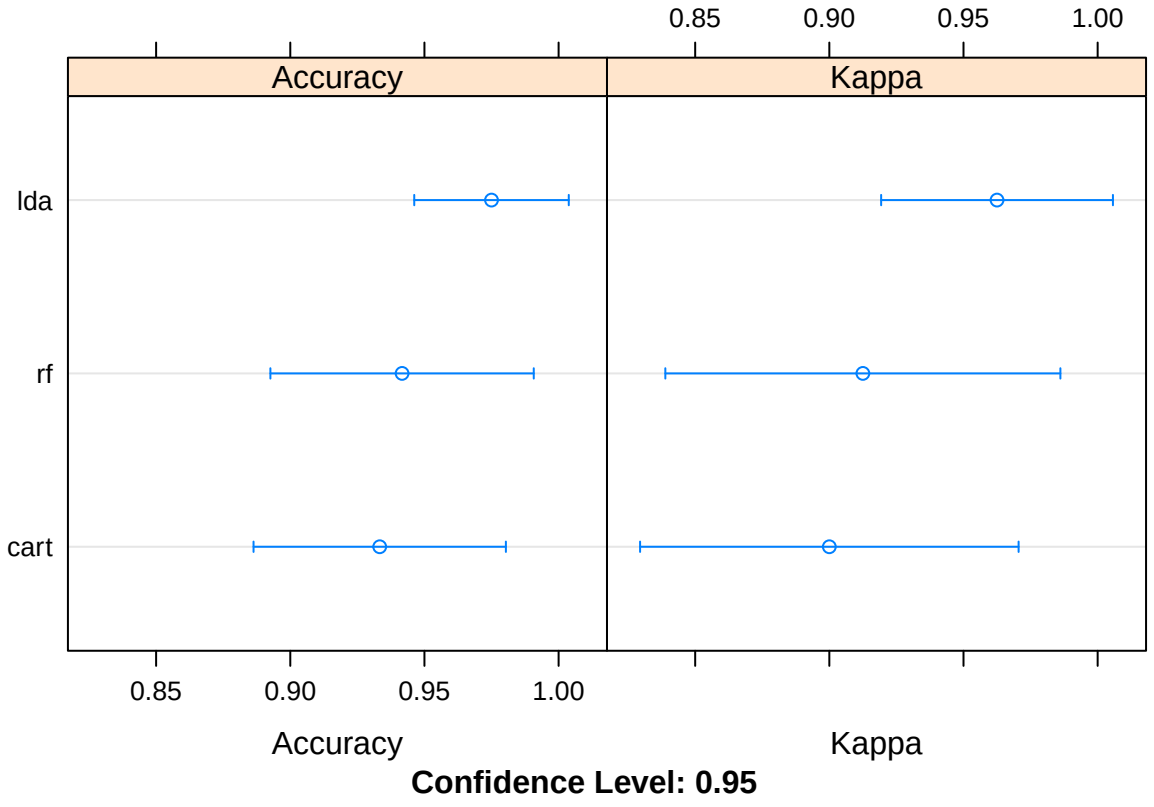
```
## List of 6
##  $ call  : language resamples.default(x = list(lda = lda.model, cart = cart.model, rf = rf.model))
##  $ values :'data.frame': 10 obs. of  7 variables:
##   ..$ Resample     : chr [1:10] "Fold01" "Fold02" "Fold03" "Fold04" ...
##   ..$ lda~Accuracy : num [1:10] 1 1 1 1 0.917 ...
##   ..$ lda~Kappa    : num [1:10] 1 1 1 1 0.875 0.875 0.875 1 1 1
##   ..$ cart~Accuracy: num [1:10] 1 1 1 0.833 0.833 ...
##   ..$ cart~Kappa   : num [1:10] 1 1 1 0.75 0.75 0.875 0.875 0.875 0.875 1
##   ..$ rf~Accuracy  : num [1:10] 1 1 1 0.833 0.833 ...
##   ..$ rf~Kappa     : num [1:10] 1 1 1 0.75 0.75 0.875 0.875 1 0.875 1
##  $ models : chr [1:3] "lda" "cart" "rf"
##  $ metrics: chr [1:2] "Accuracy" "Kappa"
##  $ timings:'data.frame': 3 obs. of  3 variables:
##   ..$ Everything: num [1:3] 1.03 1.1 2.39
##   ..$ FinalModel: num [1:3] 0 0 0.03
##   ..$ Prediction: num [1:3] NA NA NA
##  $ methods: Named chr [1:3] "lda" "rpart" "rf"
##   ..- attr(*, "names")= chr [1:3] "lda" "cart" "rf"
##  - attr(*, "class")= chr "resamples"
```

```
summary(results.model)
```

```
##
## Call:
## summary.resamples(object = results.model)
##
## Models: lda, cart, rf
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
## lda  0.9166667 0.9375000 1.0000000 0.9750000       1    1    0
```

```
## cart 0.8333333 0.9166667 0.9166667 0.9333333       1    1    0
## rf   0.8333333 0.9166667 0.9583333 0.9416667       1    1    0
##
## Kappa
##       Min. 1st Qu. Median   Mean 3rd Qu. Max. NA's
## lda   0.875 0.90625 1.0000 0.9625       1    1    0
## cart 0.750 0.87500 0.8750 0.9000       1    1    0
## rf   0.750 0.87500 0.9375 0.9125       1    1    0
```

```
dotplot(results.model)
```




```
#6 模型应用
#lda 算法运用
pred.result <- predict(lda.model, validation.data)
# 混淆矩阵评价分类效果：总体分类精度、Kappa 系数
confusionMatrix(pred.result, validation.data$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         10         0
##   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: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            1.0000           1.0000
## Specificity                 1.0000            1.0000           1.0000
## Pos Pred Value              1.0000            1.0000           1.0000
## Neg Pred Value              1.0000            1.0000           1.0000
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3333           0.3333
## Detection Prevalence        0.3333            0.3333           0.3333
## Balanced Accuracy           1.0000            1.0000           1.0000
```