Exercise 6

191820019 陈文杰

2020/12/1

R Markdown

#一、分别采用至少5种分类算法对“鸢尾花”数据集进行分析，并计算其分类结果的混淆矩阵、准确率（Accuracy）、查准率/精确率（Precision）、查全率/召回率（Recall）和F1值（F1-Score） 解：

##[策略]

（1）读取iris数据，进行数据概览，并作预处理（将分类变量Species因子化）

（2）用sample函数划分好测试集和训练集，比例为0.7：0.3

（3）完成模型构建，用到的包与函数如下：

①party包的ctree函数建立C4.5决策树模型

②tree包的tree函数建立Cart决策树模型

③C50包的C5.0函数建立C5.0决策树模型

④nnet包的nnet函数建立BP神经网络模型

⑤e1071包的naiveBayes函数建立朴素贝叶斯分类模型

⑥kknn包的kknn函数建立knn模型

⑦e1071包的svm函数建立SVM模型

⑧MASS包的ida函数建立IDA模型

⑨randomForest包的randomForest函数建立随机森林模型

（4）用predict函数，结合分类模型分别对训练数据及和测试数据及进行分类预测

（5）用cbind函数，将原始数据与预测数据结合，得到分类矩阵

（6）用table函数，传入真实列与预测列，分别作为actual与predictedclass的值，构建混淆矩阵 （7）利用自定义的prf函数，计算多分类模型的Accuracy、Precision、Recall和F1-Score，对模型进行评价

##[过程|结果] 0、编写自定义函数，用于多分类问题的P、R、F值计算

```
prf <- function(test_confusion)
{
  Accuracy <- sum(diag(test_confusion))/sum(test_confusion)
  for (i in 1:nrow(test_confusion))
  {
    confusion <- test_confusion
    classname <- colnames(confusion)[i]
    TP <- confusion[i,i]
    FN <- sum(confusion[i,])-TP
    FP <- sum(confusion[,i])-TP
    TN <- sum(diag(confusion))-confusion[i,i]
    Precision <- TP/(TP+FP)
    Recall <- TP/sum(confusion[i,])
    F1 <- 2*Precision*Recall/(Precision+Recall)
    print(paste(as.character(classname),"'s Precision:",     as.character(Precision),sep="",coll
apse=NULL))
    print(paste(as.character(classname),"'s Recall:", as.character(Recall),sep="",collapse=NULL
))
    print(paste(as.character(classname),"'s F1-Score:", as.character(F1),sep="",collapse=NULL))
  }
  print(paste('Accuracy: ',Accuracy))
}
```

1、决策树

```
#iris

# 设置随机种子
set.seed(1234)

#分类变量因子化
Data <- iris
Data$Species = as.factor(Data$Species)

# 随机抽取70%定义为训练数据集，30%为测试数据集
ind <- sample(2,nrow(Data),replace=TRUE,prob=c(0.7,0.3))
traindata <- Data[ind==1, ]
testdata <- Data[ind==2, ]

#一、建立决策树模型进行预测

library(grid)
library(mvtnorm)
library(stats4)
library(modeltools)
library(sandwich)
library(strucchange)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(zoo)



#1、C4.5决策树
#构建C4.5决策树
library(party)
ctree.model <- ctree(Species~., data=traindata)

#输出C4.5决策树图
#plot(ctree.model, type="simple")
#plot(ctree.model, type="extended")

print("***C4.5***")
```

```
## [1] "***C4.5***"
```

```
#预测C4.5决策树结果
train_predict <- predict(ctree.model) #训练数据集
test_predict <- predict(ctree.model, newdata=testdata) #测试数据集

print("***traindata***")
```

```
## [1] "***traindata***"
```

```
#输出训练集的混淆矩阵，Accuracy、P、R、F1值
train_predictdata <- cbind(traindata, predictedclass = train_predict)
(train_confusion <- table(actual = traindata$Species, predictedclass = train_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         37         1
##   virginica       0          3        31
```

```
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.925"
## [1] "versicolor's Recall:0.973684210526316"
## [1] "versicolor's F1-Score:0.948717948717949"
## [1] "virginica's Precision:0.96875"
## [1] "virginica's Recall:0.911764705882353"
## [1] "virginica's F1-Score:0.939393939393939"
## [1] "Accuracy:  0.964285714285714"
```

```
print("***testdata***")
```

```
## [1] "***testdata***"
```

```
#输出测试集的混淆矩阵，Accuracy、P、R、F1值
test_predictdata <- cbind(testdata, predictedclass=test_predict)
(test_confusion <- table(actual = testdata$Species, predictedclass = test_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          2        14
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

```
#2、Cart决策树
library(tree)
#建立CART决策树模型
tree.model <- tree(Species~., data=traindata)
#决策树图输出
#plot(tree.model, type="uniform")
#text(tree.model)

print("***cart***")
```

```
## [1] "***cart***"
```

```
#预测结果
train_predict <- predict(tree.model, newdata=traindata, type="class")
test_predict <- predict(tree.model, newdata=testdata, type="class")

print("***traindata***")
```

```
## [1] "***traindata***"
```

```
#输出训练数据的分类结果与混淆矩阵
train_predictdata <- cbind(traindata, predictedclass=train_predict)
(train_confusion <- table(actual=traindata$Species, predictedclass=train_predict))
```

```
##            predictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         37         1
##   virginica       0          3        31
```

```
#输出训练结果的Accuracy、P、R、F1值
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.925"
## [1] "versicolor's Recall:0.973684210526316"
## [1] "versicolor's F1-Score:0.948717948717949"
## [1] "virginica's Precision:0.96875"
## [1] "virginica's Recall:0.911764705882353"
## [1] "virginica's F1-Score:0.939393939393939"
## [1] "Accuracy:  0.964285714285714"
```

```
print("***testdata***")
```

```
## [1] "***testdata***"
```

```
#输出测试数据的分类结果与混淆矩阵
test_predictdata <- cbind(testdata,predictedclass = test_predict)
(test_confusion <- table(actual=testdata$Species, predictedclass=test_predict))
```

```
##            predictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          2        14
```

```
#输出测试结果的Accuracy、P、R、F1值
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

```
#3、C5.0决策树
library(C50)
#构建决策树模型
c50.model <- C5.0(Species~., data=traindata)
#plot(c50.model)

print("***C5.0***")
```

```
## [1] "***C5.0***"
```

```
#预测结果
train_predict <- predict(c50.model, newdata=traindata, type="class")
test_predict <- predict(c50.model, newdata=testdata, type="class")

print("***traindata***")
```

```
## [1] "***traindata***"
```

```
#输出训练数据的分类结果与混淆矩阵
train_predictdata <- cbind(traindata, predictedclass=train_predict)
(train_confusion <- table(actual=traindata$Species, predictedclass=train_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         37         1
##   virginica       0          3        31
```

```
#输出训练结果的Accuracy、P、R、F1值
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.925"
## [1] "versicolor's Recall:0.973684210526316"
## [1] "versicolor's F1-Score:0.948717948717949"
## [1] "virginica's Precision:0.96875"
## [1] "virginica's Recall:0.911764705882353"
## [1] "virginica's F1-Score:0.939393939393939"
## [1] "Accuracy:  0.964285714285714"
```

```
print("***testdata***")
```

```
## [1] "***testdata***"
```

```
#输出测试数据的分类结果与混淆矩阵
test_predictdata <- cbind(testdata,predictedclass = test_predict)
(test_confusion <- table(actual=testdata$Species, predictedclass=test_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          2        14
```

```
#输出测试结果的Accuracy、P、R、F1值
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

```
#4、ID3决策树
library(rpart)
print("***ID3***")
```

```
## [1] "***ID3***"
```

```
ID3.model<-rpart(Species~.,data = traindata,method = "class")

train_predict <- predict(ID3.model, data=traindata, type="class")
test_predict <- predict(ID3.model, newdata=testdata, type="class")

print("***traindata***")
```

```
## [1] "***traindata***"
```

```
train_predictdata <- cbind(traindata, predictedclass=train_predict)
(train_confusion <- table(actual=traindata$Species, pretictedclass=train_predict))
```

```
##             pretictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         37         1
##   virginica       0          3        31
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

```
print("***testdata***")
```

```
## [1] "***testdata***"
```

```
test_predictdata <- cbind(testdata, predictedclass=test_predict)
(test_confusion <- table(actual=testdata$Species, pretictedclass=test_predict))
```

```
##            pretictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          2        14
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

2、BP神经网络

```
#BP神经网络
library(nnet)

#设置参数
size <- 10 #设置节点数
decay <- 0.05 #设置权值衰减系数
nnet.model <- nnet(Species~., traindata, size=size, decay=decay) #建立BP模型
```

```
## # weights:  83
## initial  value 200.717468
## iter  10 value 56.810758
## iter  20 value 21.636468
## iter  30 value 16.717897
## iter  40 value 15.223857
## iter  50 value 14.534830
## iter  60 value 14.352886
## iter  70 value 14.272249
## iter  80 value 14.209957
## iter  90 value 14.195980
## iter 100 value 14.188723
## final  value 14.188723
## stopped after 100 iterations
```

```
#summary(nnet.model)  #输出模型概要

#预测结果
train_predict <- predict(nnet.model, newdata=traindata, type="class")
test_predict <- predict(nnet.model, newdata=testdata, type="class")


#输出训练数据的分类结果与混淆矩阵
train_predictdata <- cbind(traindata, predictedclass=train_predict)

(train_confusion <- table(actual=traindata$Species, predictedclass=train_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         36         2
##   virginica       0          1        33
```

```
#输出训练结果的Accuracy、P、R、F1值
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.972972972972973"
## [1] "versicolor's Recall:0.947368421052632"
## [1] "versicolor's F1-Score:0.96"
## [1] "virginica's Precision:0.942857142857143"
## [1] "virginica's Recall:0.970588235294118"
## [1] "virginica's F1-Score:0.956521739130435"
## [1] "Accuracy:  0.973214285714286"
```

```
#输出测试数据的分类结果与混淆矩阵
test_predictdata <- cbind(testdata,predictedclass = test_predict)
(test_confusion <- table(actual=testdata$Species, predictedclass=test_predict))
```

```
##            predictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          0        16
```

```
#输出测试结果的Accuracy、P、R、F1值
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:1"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:1"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:1"
## [1] "virginica's F1-Score:1"
## [1] "Accuracy:  1"
```

3、贝叶斯分类

```
#利用e1071包的naiveBayes函数建立朴素贝叶斯分类模型
library(e1071)
naiveBayes.model <- naiveBayes(Species~., data=traindata)

# 预测结果
train_predict <- predict(naiveBayes.model, newdata = traindata)  # 训练数据集
test_predict <- predict(naiveBayes.model, newdata = testdata)  # 测试数据集
# 输出训练数据的分类结果、混淆矩阵
train_predictdata <- cbind(traindata, predictedclass = train_predict)
(train_confusion <- table(actual = traindata$Species, predictedclass = train_predict))
```

```
##            predictedclass
## actual       setosa versicolor virginica
##   setosa         40          0         0
##   versicolor      0         36         2
##   virginica       0          2        32
```

```
##输出训练结果的Accuracy、P、R、F1值
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.947368421052632"
## [1] "versicolor's Recall:0.947368421052632"
## [1] "versicolor's F1-Score:0.947368421052632"
## [1] "virginica's Precision:0.941176470588235"
## [1] "virginica's Recall:0.941176470588235"
## [1] "virginica's F1-Score:0.941176470588235"
## [1] "Accuracy:  0.964285714285714"
```

```
# 输出测试数据的分类结果、混淆矩阵
test_predictdata <- cbind(testdata, predictedclass = test_predict)
(test_confusion <- table(actual = testdata$Species, predictedclass = test_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         12         0
##   virginica       0          2        14
```

```
# 输出测试数据的Precision、Recall、Accuracy以及F1-SCOREs
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.857142857142857"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:0.923076923076923"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:0.875"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.947368421052632"
```

4、KNN

```
#载入数据
Data <- iris
Data$Species <- as.factor(Data$Species)
set.seed(123)

#随机抽取70%定义为训练数据集，30%为测试数据集
ind <- sample(2,nrow(Data),replace=TRUE,prob=c(0.7,0.3))
traindata <- Data[ind==1, ]
testdata <- Data[ind==2, ]

#使用kknn函数构建knn模型
library(kknn)
kknn.model <- kknn(Species~., train=traindata, test=traindata, k=5)  #训练数据
kknn.model2 <- kknn(Species~., train=traindata, test=testdata, k=5) #测试数据


#预测结果
train_predict <- predict(kknn.model) #训练
test_predict <- predict(kknn.model2)  #测试

#输出训练集的混淆矩阵，Accuracy、P、R、F1值
(train_confusion <- table(actual = traindata$Species, predictedclass=train_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         35          0         0
##   versicolor      0         36         0
##   virginica       0          0        35
```

```
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:1"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:1"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:1"
## [1] "virginica's F1-Score:1"
## [1] "Accuracy:  1"
```

```
#输出测试集的混淆矩阵，Accuracy、P、R、F1值
(test_confusion <- table(actual=testdata$Species,predictedclass=test_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         10         4
##   virginica       0          2        13
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.833333333333333"
## [1] "versicolor's Recall:0.714285714285714"
## [1] "versicolor's F1-Score:0.769230769230769"
## [1] "virginica's Precision:0.764705882352941"
## [1] "virginica's Recall:0.866666666666667"
## [1] "virginica's F1-Score:0.8125"
## [1] "Accuracy:  0.863636363636364"
```

5、利用SVM模型进行分类与预测

```
# 利用SVM建模
library(e1071)
svm.model <- svm(Species~ ., data = traindata)

#预测结果
train_predict <- predict(svm.model, newdata = traindata,type="class")  # 训练数据集
test_predict <- predict(svm.model, newdata = testdata,type="class")  #测试数据集

#输出训练集的混淆矩阵，Accuracy、P、R、F1值
(train_confusion <- table(actual = traindata$Species, predictedclass=train_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         35          0         0
##   versicolor      0         35         1
##   virginica       0          0        35
```

```
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:1"
## [1] "versicolor's Recall:0.972222222222222"
## [1] "versicolor's F1-Score:0.985915492957746"
## [1] "virginica's Precision:0.972222222222222"
## [1] "virginica's Recall:1"
## [1] "virginica's F1-Score:0.985915492957746"
## [1] "Accuracy:  0.990566037735849"
```

```
#输出测试集的混淆矩阵，Accuracy、P、R、F1值
(test_confusion <- table(actual=testdata$Species,predictedclass=test_predict))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         11         3
##   virginica       0          1        14
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.916666666666667"
## [1] "versicolor's Recall:0.785714285714286"
## [1] "versicolor's F1-Score:0.846153846153846"
## [1] "virginica's Precision:0.823529411764706"
## [1] "virginica's Recall:0.933333333333333"
## [1] "virginica's F1-Score:0.875"
## [1] "Accuracy:  0.909090909090909"
```

6、利用LDA模型分类预测

```
# 建立lda分类模型
library(MASS)
lda.model <- lda(Species ~ ., data = traindata)


#预测结果
train_predict <- predict(lda.model, newdata = traindata, type = "class")  # 训练数据集
test_predict <- predict(lda.model, newdata = testdata, type = "class")  # 测试数据集

#输出训练集的混淆矩阵，Accuracy、P、R、F1值
(train_confusion <- table(actual = traindata$Species, predictedclass=train_predict$class))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         35          0         0
##   versicolor      0         36         0
##   virginica       0          0        35
```

```
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:1"
## [1] "versicolor's Recall:1"
## [1] "versicolor's F1-Score:1"
## [1] "virginica's Precision:1"
## [1] "virginica's Recall:1"
## [1] "virginica's F1-Score:1"
## [1] "Accuracy:  1"
```

```
#输出测试集的混淆矩阵，Accuracy、P、R、F1值
(test_confusion <- table(actual=testdata$Species,predictedclass=test_predict$class))
```

```
##             predictedclass
## actual       setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         13         1
##   virginica       0          1        14
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.928571428571429"
## [1] "versicolor's Recall:0.928571428571429"
## [1] "versicolor's F1-Score:0.928571428571429"
## [1] "virginica's Precision:0.933333333333333"
## [1] "virginica's Recall:0.933333333333333"
## [1] "virginica's F1-Score:0.933333333333333"
## [1] "Accuracy:  0.954545454545455"
```

7、利用随机森林模型分类预测

```
# 建立randomForest模型
library(randomForest)
```

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

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

```
randomForest.model <- randomForest(Species ~ ., data = traindata)

# 预测结果
test_predict <- predict(randomForest.model, newdata = testdata)  # 测试数据集


# 输出训练数据的混淆矩阵
(train_confusion <- randomForest.model$confusion)
```

```
##            setosa versicolor virginica class.error
## setosa         35          0         0  0.00000000
## versicolor      0         35         1  0.02777778
## virginica       0          2        33  0.05714286
```

```
prf(train_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.945945945945946"
## [1] "versicolor's Recall:0.971472629144179"
## [1] "versicolor's F1-Score:0.958539368581209"
## [1] "virginica's Precision:0.970588235294118"
## [1] "virginica's Recall:0.941320293398533"
## [1] "virginica's F1-Score:0.955730244104262"
## [1] "Accuracy:  0.970920272019272"
```

```
# 输出测试数据的混淆矩阵
(test_confusion <- table(actual = testdata$Species, predictedclass = test_predict))
```

```
##            predictedclass
## actual      setosa versicolor virginica
##   setosa        15          0         0
##   versicolor     0         11         3
##   virginica      0          1        14
```

```
prf(test_confusion)
```

```
## [1] "setosa's Precision:1"
## [1] "setosa's Recall:1"
## [1] "setosa's F1-Score:1"
## [1] "versicolor's Precision:0.916666666666667"
## [1] "versicolor's Recall:0.785714285714286"
## [1] "versicolor's F1-Score:0.846153846153846"
## [1] "virginica's Precision:0.823529411764706"
## [1] "virginica's Recall:0.933333333333333"
## [1] "virginica's F1-Score:0.875"
## [1] "Accuracy:  0.909090909090909"
```

#二、观察分类结果中的误差，尝试分析误差产生的原因与改进措施。

1、决策树（ID3） （1）模型概述 以信息增益度量属性选择，选择分裂后信息增益最大的属性进行分裂 （2）误差分析 实验结果表明，ID3算法对于iris数据集的分类精度较高（accuracy为0.947），而且对于traindata和testdata的评估指数几乎一致，模型训练效果较好，存在的误差主要在versicolor与virginica两类中，误差产生的原因可能是ID3算法用信息增益过分偏向取值多的属性。 (3)改进措施 利用C4.5算法，在生成过程中用信息增益比来选择特征；同时在树的构造过程中进行剪枝。

2、BP神经网络 （1）模型概述 利用输出后的误差来估计输出层的直接前导层的误差，再用这个误差估计更前一层的误差，如此一层一层的反传下去，就获得了所有其他各层的误差估计。 （2）误差分析 在iris数据集中，BP神经网络的整体精度（Accuracy）高于决策树，其中train数据集出现部分少量的误差，test则完全命中，训练集中的误差可能与设置的节点数与权重衰减有关。 （3）改进措施 不断调整节点数和权重衰减系数，“炼丹”

3、朴素贝叶斯 （1）模型概述 确定属性特征，获取训练样本，对每个类别计算P（yi），对每个特征属性计算所有划分的条件概率，对每个类别根据先验概率得出后验，取最大值确定分类 （2）误差分析 朴素贝叶斯模型在iris数据集中，训练集的精度高于测试数据集，误差可能来源于其对于类条件独立的假设，未考虑到不同属性之间的相关性。 （3）改进措施 使用半朴素贝叶斯分类器，放松一定的独立性假定

4、SVM （1）模型概述 在线性可分以及线性不可分的情况下，获得最优分类面 （2）误差分析 在iris数据集中，SVM模型之于train数据集的精度高于test数据集，误差主要在于“versicolor”和“virginica”两类中，原因可能在于①不能直接适用于多类线性不可分②容易被偏性样本误导 （3）改进措施 调整模型，选用不同的核函数拟合

5、KNN （1）模型概述 根据k近邻对未知元组xq的距离贡献，计算预测值 （2）误差分析 相对而言误差较大，尤其是测试集数据，原因可能在于KNN算法容易产生维数灾难，相邻距离可能由不想关的属性主导，从而降低模型精度 （3）改进措施 通过轴拉伸，删除不相关的属性

6、LDA算法 （1）模型概述 给定训练样例集，设法将样例投影到一条直线上，使得同类样例的投影点尽可能接近，异样样例的投影点尽可能远离；在对新样本进行分类时，将其投影到同样的直线上，再根据投影点的位置来确定新样本的类别 。 （2）误差分析 LDA算法比较朴素，但是对于iris数据集而言分类精度较高。

7、随机森林 （1）模型概述 它基于决策树集合的多数投票来进行预测。单个决策树往往是不稳定的，不能提供稳定的预测。而通过许多树的预测，随机森林方法能更加稳定，且与其他分类器相比具有更好的性能。 （2）误差分析 在针对iris数据集的分类中，随机森林模型精度适中，train的精度大于test的精度，误差原因比较难解释，因为随机森林模型过程是难以解释的黑箱，不过可能与样本的分布有关。

（3）改进措施 调整模型参数：随机森林中树的数量，以及用于分裂的随机采样的基因数量