基本分类模型案例：儿童汽车座椅销售量预测

吴翔

2019-03-19

概述

我们通过R语言 `ISLR` 包中儿童使用的汽车座椅销售量的案例来阐述如何使用如下基本分类模型：

- 决策树
- 装袋法
- 随机森林
- 提升法

数据集 `Carseats` 包含500家商店的儿童用汽车座椅的销售情况，以及商店/所在社区相关的变量，其变量如下所示。

```
# clean the work directory
rm(list = ls())

# set seeds
set.seed(123)

# read dataset
suppressMessages(library(ISLR))
suppressMessages(library(tidyverse))
data("Carseats")
# display the variables
str(Carseats)
```

```
## 'data.frame':    400 obs. of  11 variables:
##  $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
##  $ CompPrice  : num  138 111 113 117 141 124 115 136 132 132 ...
##  $ Income     : num  73 48 35 100 64 113 105 81 110 113 ...
##  $ Advertising: num  11 16 10 4 3 13 0 15 0 0 ...
##  $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
##  $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
##  $ ShelveLoc  : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...
##  $ Age        : num  42 65 59 55 38 78 71 67 76 76 ...
##  $ Education  : num  17 10 12 14 13 16 15 10 10 17 ...
##  $ Urban      : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
##  $ US         : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...
```

```
# summary of dataset
summary(Carseats)
```

```
##      Sales          CompPrice       Income        Advertising
##  Min.   : 0.00   Min.   : 77   Min.   : 21.0   Min.   : 0.00
##  1st Qu.: 5.39   1st Qu.:115   1st Qu.: 42.8   1st Qu.: 0.00
##  Median : 7.49   Median :125   Median : 69.0   Median : 5.00
##  Mean   : 7.50   Mean   :125   Mean   : 68.7   Mean   : 6.63
##  3rd Qu.: 9.32   3rd Qu.:135   3rd Qu.: 91.0   3rd Qu.:12.00
##  Max.   :16.27   Max.   :175   Max.   :120.0   Max.   :29.00
##    Population       Price       ShelveLoc        Age          Education
##  Min.   : 10   Min.   : 24   Bad   : 96   Min.   :25.0   Min.   :10.0
##  1st Qu.:139   1st Qu.:100   Good  : 85   1st Qu.:39.8   1st Qu.:12.0
##  Median :272   Median :117   Medium:219   Median :54.5   Median :14.0
##  Mean   :265   Mean   :116                Mean   :53.3   Mean   :13.9
##  3rd Qu.:398   3rd Qu.:131                3rd Qu.:66.0   3rd Qu.:16.0
##  Max.   :509   Max.   :191                Max.   :80.0   Max.   :18.0
##  Urban       US
##  No :118   No :142
##  Yes:282   Yes:258
##
##
##
##
```

决策树

我们先根据销售量是否大于8，将销售量转化为分类变量。

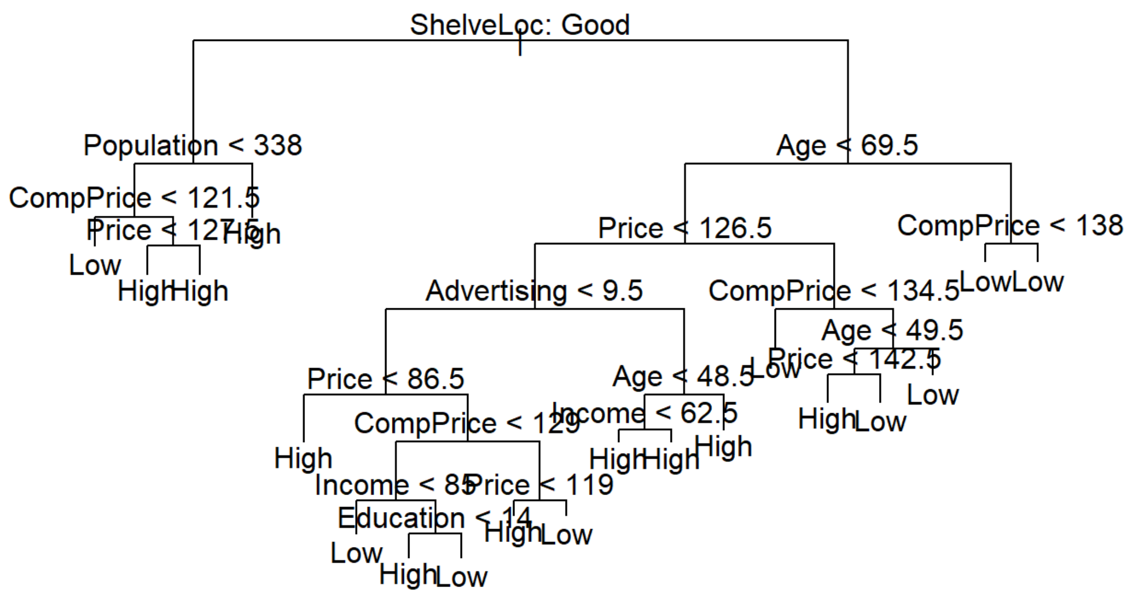
```
# convert to categorical variable
Carseats <- Carseats %>%
  within(Sales <- as.factor(ifelse(Sales <= 8, "Low", "High")))
# training set and validation set
train <- sample(1:nrow(Carseats), 200)
carseats.train <- Carseats[train, ]
carseats.test <- Carseats[-train, ]
```

在训练集上运行决策树模型，

```
suppressMessages(library(tree))
# decision tree
tree.fit <- tree(Sales ~ ., data = carseats.train)
summary(tree.fit)
```

```
##
## Classification tree:
## tree(formula = Sales ~ ., data = carseats.train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Population"  "CompPrice"   "Price"       "Age"
## [6] "Advertising" "Income"      "Education"
## Number of terminal nodes:  19
## Residual mean deviance:  0.539 = 97.6 / 181
## Misclassification error rate: 0.13 = 26 / 200
```

```
# plot the tree
plot(tree.fit)
text(tree.fit, pretty = 0)
```



`summary()` 函数给出的分类树偏差由$-2\sum_m\sum_k n_{mk}\log(p_{mk})$计算的。进而在测试集上使用决策树模型，并计算分类准确率。

```
# predictions
tree.pred <- predict(tree.fit, carseats.test, type = "class")
# compare predictions with true values
table(tree.pred, carseats.test$Sales)
```

```
##
## tree.pred High  Low
##      High   52   29
##      Low    33   86
```

```
# performance
mean(tree.pred == carseats.test$Sales)
```

```
## [1] 0.69
```

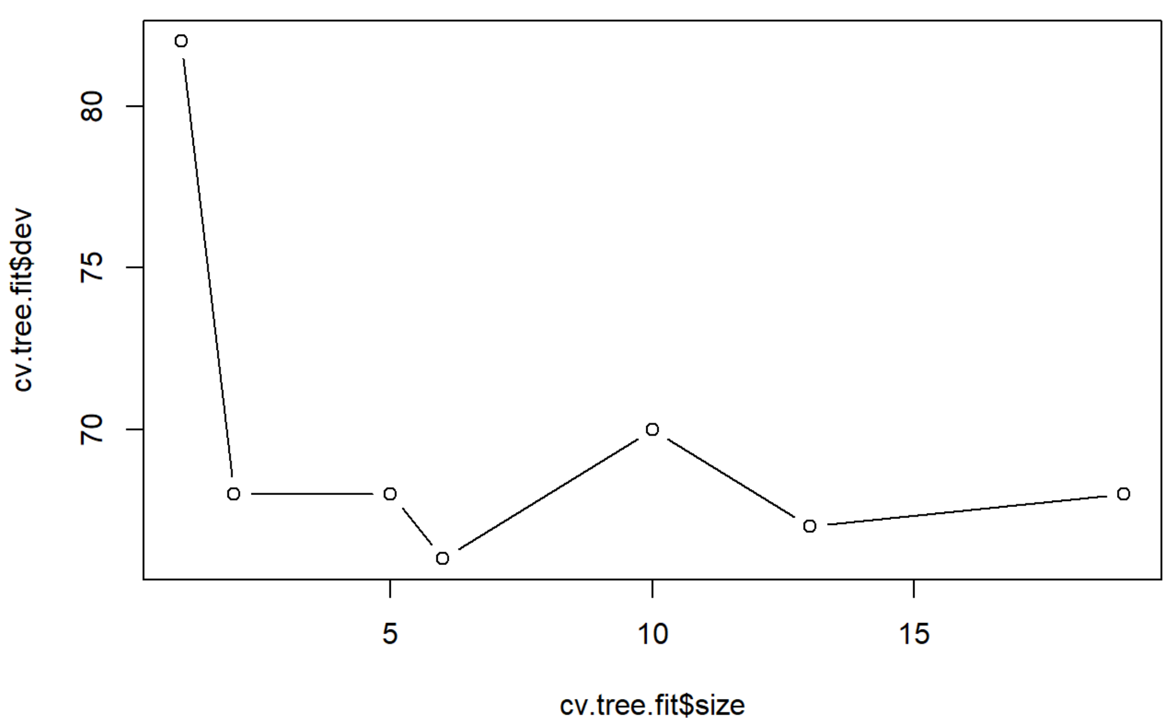
此时决策树模型的分类准确率为0.69。显然，训练集和测试集的准确率差异较大，出现了明显的过度拟合现象。

为此，我们通过剪枝来改进分类效果，并采用交叉验证来选取最佳的成本复杂性参数 `k` 。

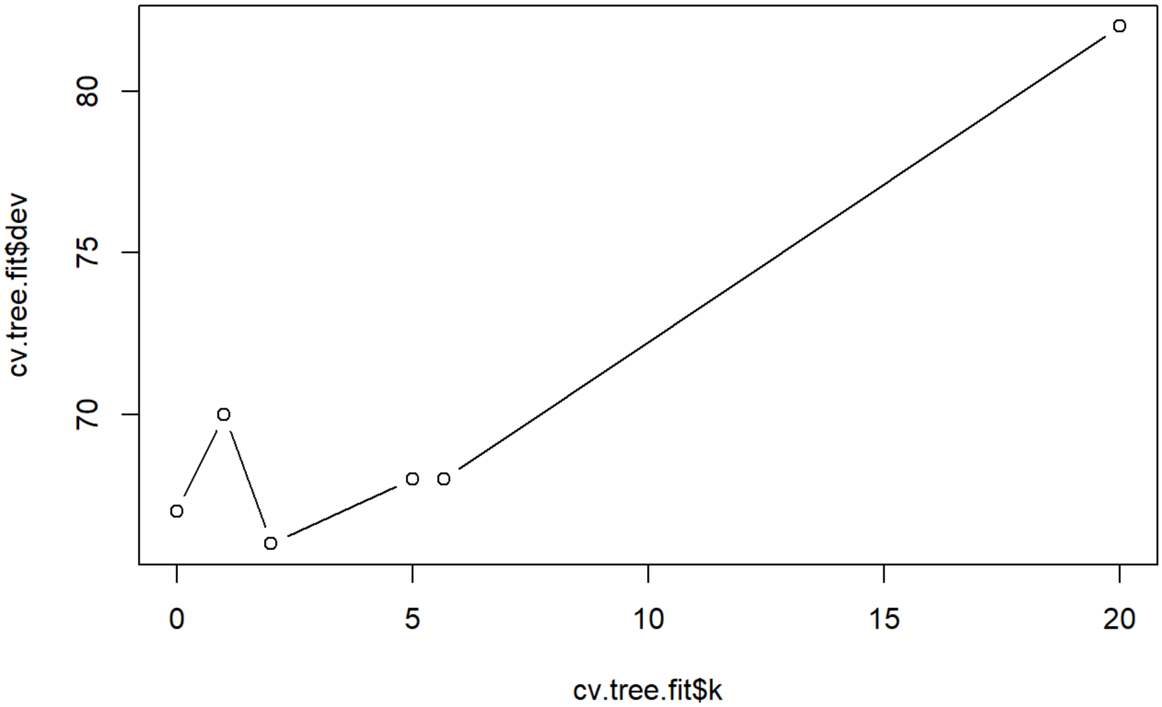
```
# cross validation
cv.tree.fit <- cv.tree(tree.fit, FUN = prune.misclass)
cv.tree.fit
```

```
## $size
## [1] 19 13 10  6  5  2  1
##
## $dev
## [1] 68 67 70 66 68 68 82
##
## $k
## [1]  -Inf  0.00  1.00  2.00  5.00  5.67 20.00
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
# plot the results
plot(cv.tree.fit$size, cv.tree.fit$dev, type = "b")
```

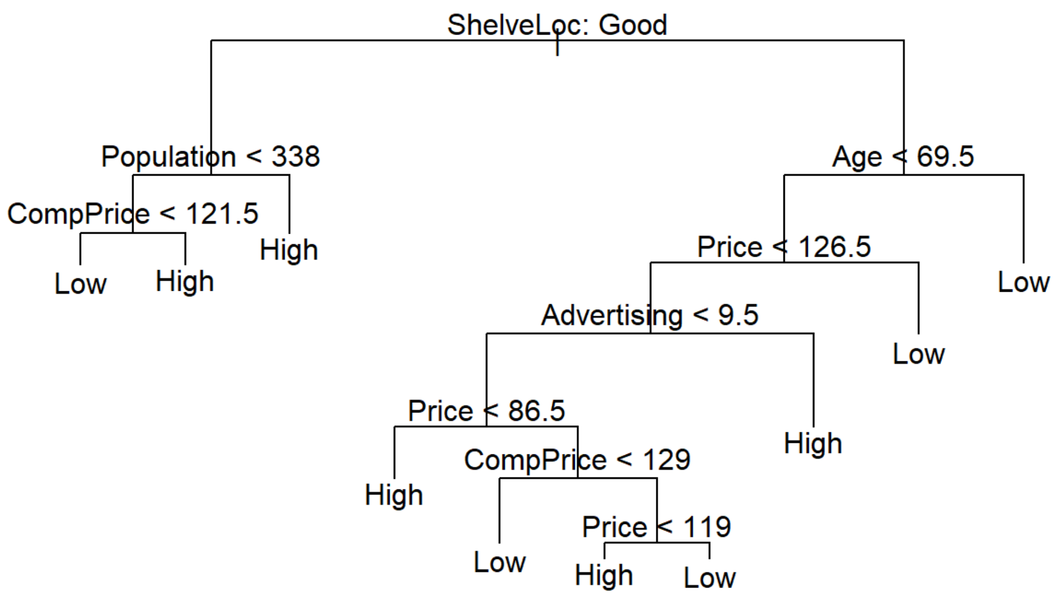



```
plot(cv.tree.fit$k, cv.tree.fit$dev, type = "b")
```



`dev` 为交叉验证错误率。因此，我们选择$size = 9$的子树。

```
# subtree
prune.tree.fit <- prune.misclass(tree.fit, best = 9)
plot(prune.tree.fit)
text(prune.tree.fit, pretty = 0)
```



重新在测试集上运行，并评估效果。

```
# predictions
tree.pred <- predict(prune.tree.fit, carseats.test, type = "class")
# compare predictions with true values
table(tree.pred, carseats.test$Sales)
```

```
##
## tree.pred High Low
##      High   51  22
##      Low    34  93
```

```
# performance
mean(tree.pred == carseats.test$Sales)
```

```
## [1] 0.72
```

此时决策树模型的分类准确率为0.72。显然，测试集的准确率有明显改善。

装袋法

使用 `randomForest` 包实现装袋法和随机森林模型。

```
suppressMessages(library(randomForest))
# bagging
bag.fit <- randomForest(Sales~., data = carseats.train, mtry = 10, importance = TRUE)
bag.fit
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = carseats.train, mtry = 10,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 10
##
##         OOB estimate of  error rate: 23%
## Confusion matrix:
##      High Low class.error
## High   48  31       0.392
## Low    15 106       0.124
```

评估在测试集上的分类效果。

```
# predictions
bag.pred <- predict(bag.fit, carseats.test, type = "class")
# compare predictions with true values
table(bag.pred, carseats.test$Sales)
```

```
##
## bag.pred High Low
##     High   60  11
##     Low    25 104
```

```
# performance
mean(bag.pred == carseats.test$Sales)
```

```
## [1] 0.82
```

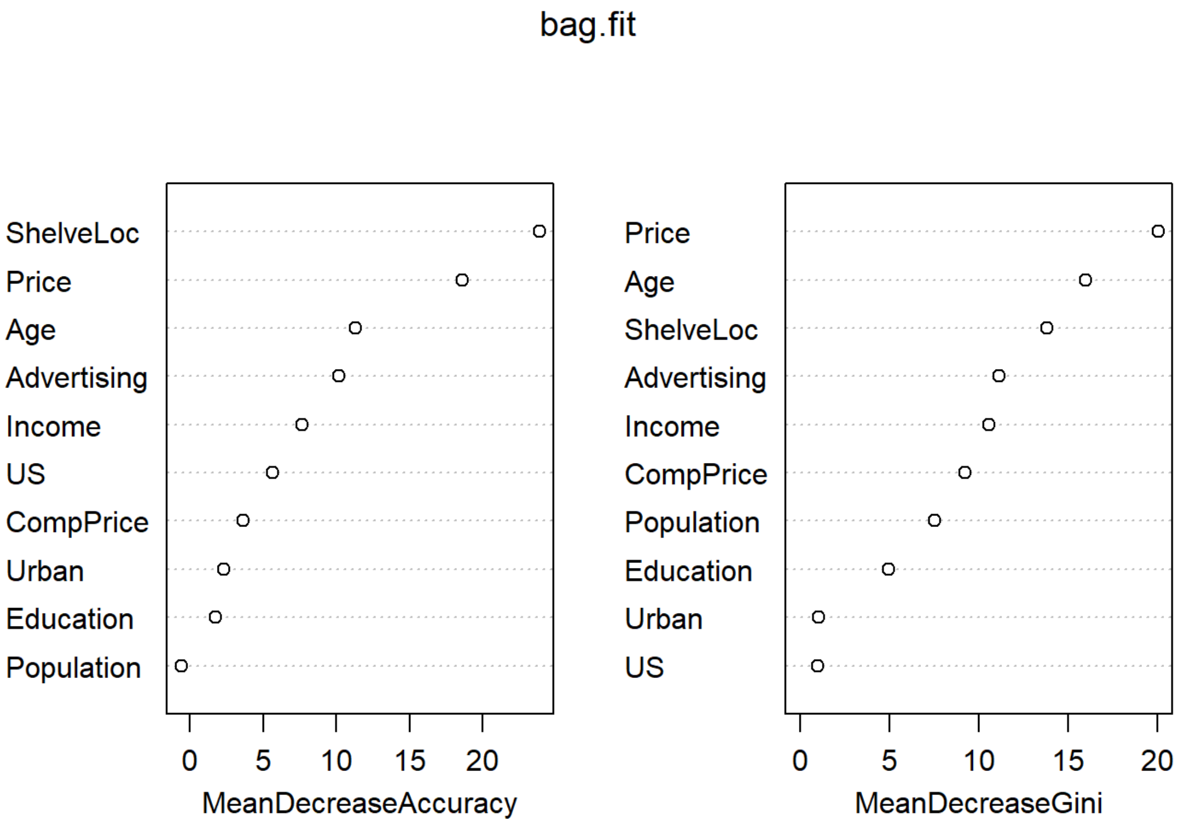
装袋法模型的分类准确率为0.82，显著优于基本决策树模型的分类效果。

进一步，可以看到装袋法中各个预测变量的重要程度。

```
# important features
importance(bag.fit)
```

```
##               High   Low MeanDecreaseAccuracy MeanDecreaseGini
## CompPrice    0.452  4.46                3.623            9.257
## Income       4.076  6.97                7.625           10.573
## Advertising 10.502  4.20               10.190           11.129
## Population   3.702 -3.82               -0.613            7.525
## Price       12.876 14.80               18.602           20.028
## ShelveLoc   20.864 16.68               23.867           13.795
## Age         12.659  4.69               11.310           16.001
## Education    0.655  2.06                1.743            4.942
## Urban        1.566  1.92                2.308            1.060
## US           4.211  4.71                5.611            0.982
```

```
# plot
varImpPlot(bag.fit)
```



随机森林

随机森林与装袋法的区别仅仅在于，是否考虑所有预测变量。随机森林模型中，取 `\sqrt{p} = \sqrt{10} = 3` 个预测变量，即 `mtry = 3` 。

```
# random forest
rf.fit <- randomForest(Sales~., data = carseats.train, mtry = 3, importance = TRUE)
rf.fit
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = carseats.train, mtry = 3,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 20.5%
## Confusion matrix:
##      High Low class.error
## High   51  28       0.354
## Low    13 108       0.107
```

评估在测试集上的分类效果。

```
# predictions
rf.pred <- predict(rf.fit, carseats.test, type = "class")
# compare predictions with true values
table(rf.pred, carseats.test$Sales)
```

```
##
## rf.pred High Low
##    High   58  12
##    Low    27 103
```

```
# performance
mean(rf.pred == carseats.test$Sales)
```

```
## [1] 0.805
```

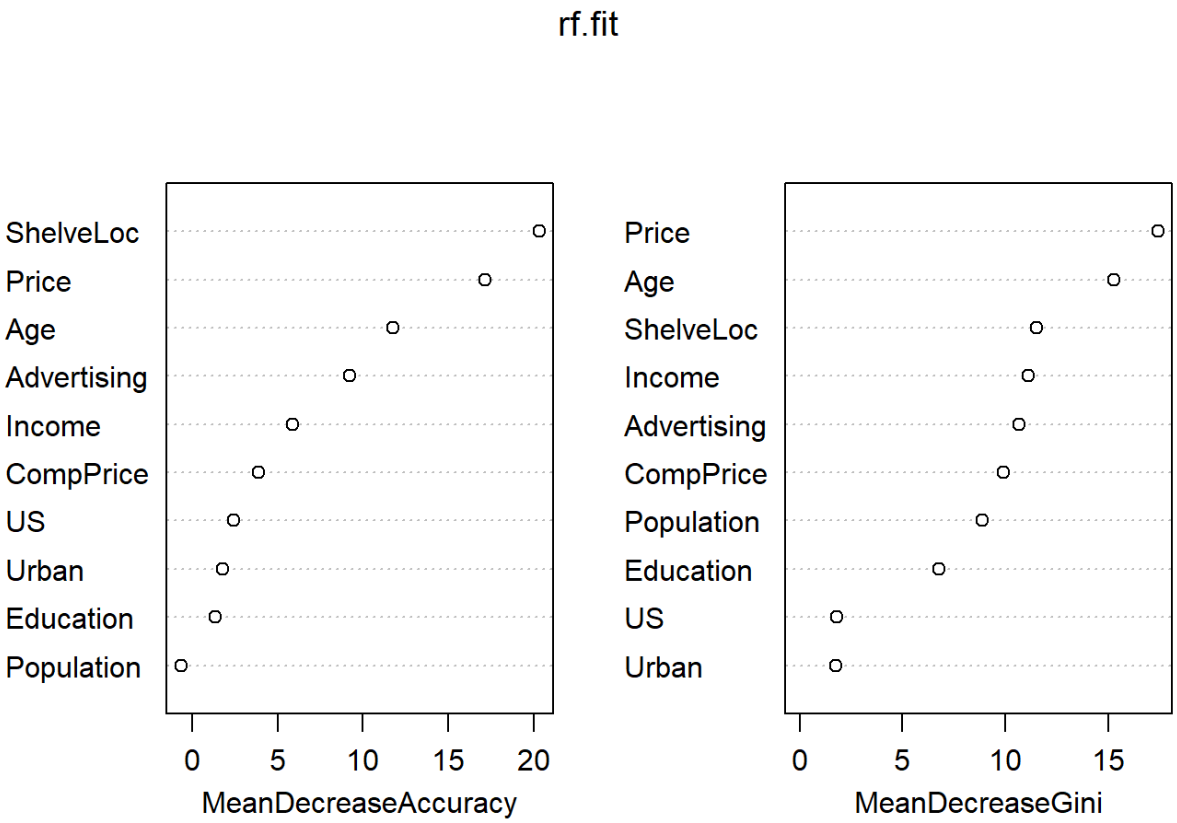
随机森林模型的分类准确率为0.805，显著优于基本决策树模型的分类效果。

进一步，可以看到随机森林中各个预测变量的重要程度。

```
# important features
importance(rf.fit)
```

```
##               High    Low MeanDecreaseAccuracy MeanDecreaseGini
## CompPrice    3.749  1.757                3.903             9.92
## Income       2.236  5.808                5.889            11.12
## Advertising  9.938  3.747                9.202            10.66
## Population   2.798 -3.640               -0.659             8.87
## Price       13.777 11.874               17.142            17.40
## ShelveLoc   17.064 15.072               20.305            11.51
## Age         11.746  5.778               11.758            15.26
## Education   -0.586  2.280                1.360             6.77
## Urban        1.406  1.315                1.775             1.74
## US           4.807 -0.738                2.448             1.79
```

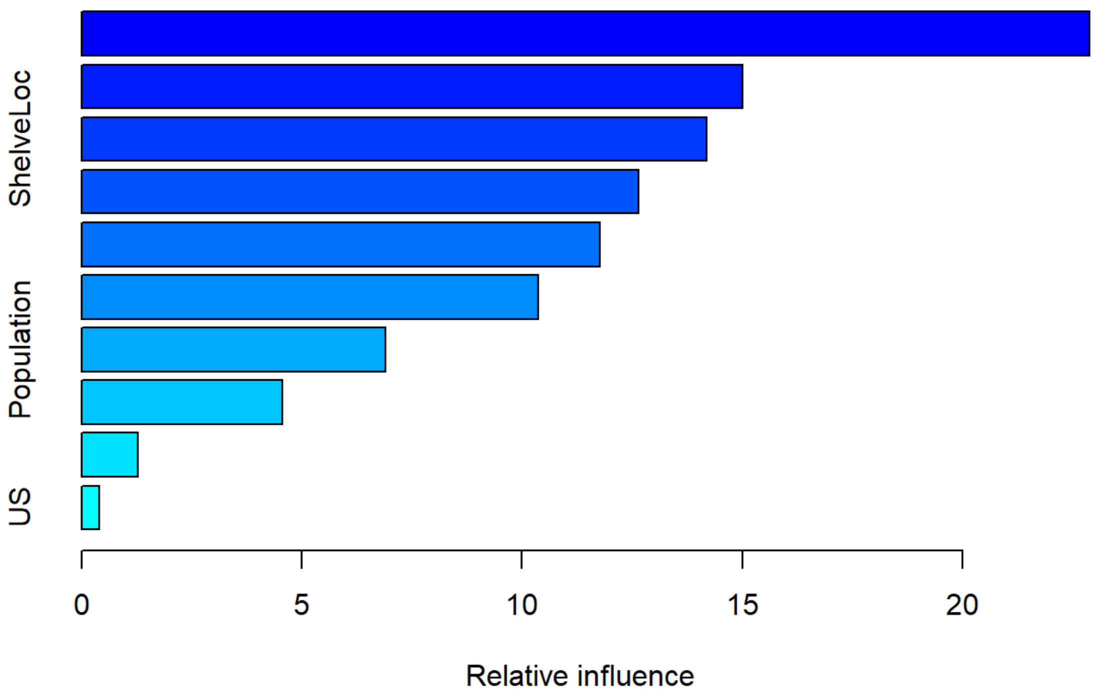
```
# plot
varImpPlot(rf.fit)
```



提升法

采用 `gbm` 包运行提升法模型。

```
suppressMessages(library(gbm))
carseats.train$Sales <- ifelse(carseats.train$Sales == "High", 1, 0)
# boosting
boost.fit <- gbm(Sales~., data = carseats.train, distribution = "bernoulli", n.trees = 500, interaction.d
epth = 4)
summary(boost.fit)
```




```
##                     var rel.inf
## Price             Price  22.884
## Age                 Age  15.007
## ShelveLoc     ShelveLoc  14.191
## CompPrice     CompPrice  12.648
## Income           Income  11.760
## Advertising Advertising  10.363
## Population   Population   6.904
## Education     Education   4.565
## Urban             Urban   1.271
## US                   US   0.407
```

评估在测试集上的分类效果。

```
# predictions
boost.pred <- ifelse(predict(boost.fit, carseats.test, n.trees = 500, type = "response") >= 0.5, "High",
"Low")
# compare predictions with true values
table(boost.pred, carseats.test$Sales)
```

```
## 
## boost.pred High Low
##       High   67  11
##       Low    18 104
```

```
# performance
mean(boost.pred == carseats.test$Sales)
```

```
## [1] 0.855
```

提升法模型的分类准确率为0.855，显著优于基本决策树模型的分类效果。

总结

最后，我们给出各个分类模型的效果。

```
# performance comparison
performance <- c(mean(tree.pred == carseats.test$Sales), mean(bag.pred == carseats.test$Sales), mean(rf.p
red == carseats.test$Sales), mean(boost.pred == carseats.test$Sales))
names(performance) <- c("tree", "bagging", "random forest", "boosting")
performance
```

```
##          tree       bagging random forest      boosting
##         0.720         0.820         0.805         0.855
```