高级统计方法 第8次作业:

序号：08 姓名：软2107 学号：20212241116 班级：软2107

**概念**

2.问题（略）

根据算法8.2，第一个树桩将由单个变量的分裂组成。通过归纳，第一次拟合的残差将导致对另一个不同的单一变量的第二次残差拟合。

f(X)=∑pj=1fj(Xj)

0) f^(x)=0,ri=yi

1) a) f^1(x)=β11I(X1<t1)+β01

1) b) f^(x)=λf^1(x)

1) c) ri=yi−λf^1(xi)

为了最大化残差的拟合，必须在下一次迭代中拟合另一个不同的树桩，随后的迭代将每次拟合xj个不同的树桩。下面是第j次迭代，其中b=j:

j) a) f^j(x)=β1jI(Xj<tj)+β0j

j) b) f^(x)=λf^1(X1)+⋯+f^j(Xj)+⋯+f^p−1(Xp−1)+f^p(Xp)

由于每次迭代的拟合是一个不同的变量残桩，因此基于“j) b)”的拟合只有p次。

f(X)=∑j=1pfj(Xj)

3.问题（略）

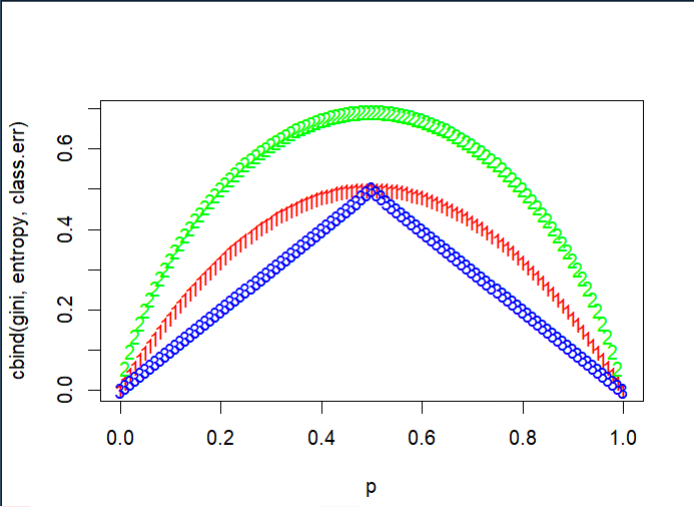
p = seq(0, 1, 0.01)

gini = p \* (1 - p) \* 2

entropy = -(p \* log(p) + (1 - p) \* log(1 - p))

class.err = 1 - pmax(p, 1 - p)

matplot(p, cbind(gini, entropy, class.err), col = c("red", "green", "blue"))



4.问题（略）

（a）问题（略）

[X1 < 1]

| |

[X2 < 1] 5

| |

[X1 < 0] 15

| |

3 [X2<0]

| |

10 0

（b）问题（略）

par(xpd = NA)

plot(NA, NA, type = "n", xlim = c(-2, 2), ylim = c(-3, 3), xlab = "X1", ylab = "X2")

# X2 < 1

lines(x = c(-2, 2), y = c(1, 1))

# X1 < 1 with X2 < 1

lines(x = c(1, 1), y = c(-3, 1))

text(x = (-2 + 1)/2, y = -1, labels = c(-1.8))

text(x = 1.5, y = -1, labels = c(0.63))

# X2 < 2 with X2 >= 1

lines(x = c(-2, 2), y = c(2, 2))

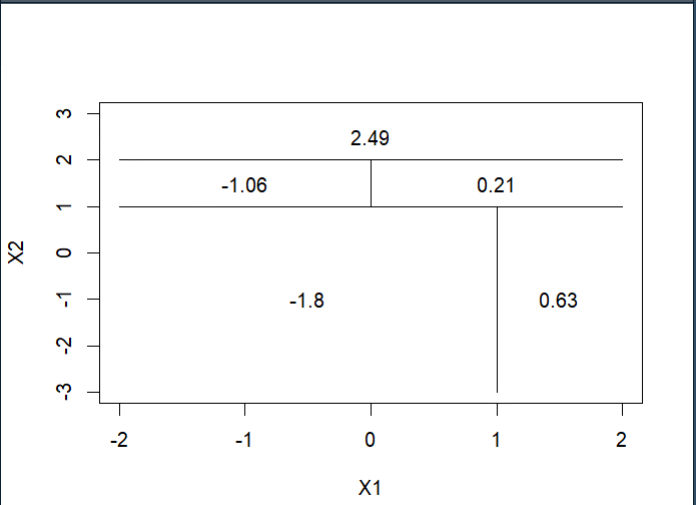
text(x = 0, y = 2.5, labels = c(2.49))

# X1 < 0 with X2<2 and X2>=1

lines(x = c(0, 0), y = c(1, 2))

text(x = -1, y = 1.5, labels = c(-1.06))

text(x = 1, y = 1.5, labels = c(0.21))



5.问题（略）

p = c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)

**Majority approach：**

sum(p >= 0.5) > sum(p < 0.5)



基于50%的阈值，红色预测的数量大于绿色预测的数量，因此为RED。

**Average approach：**

mean(p)



概率的平均值小于50%的阈值，因此为GREEN。

**应用**

8.问题（略）

（a）问题（略）

library(ISLR)

attach(Carseats)

set.seed(1)

train = sample(dim(Carseats)[1], dim(Carseats)[1]/2)

Carseats.train = Carseats[train, ]

Carseats.test = Carseats[-train, ]

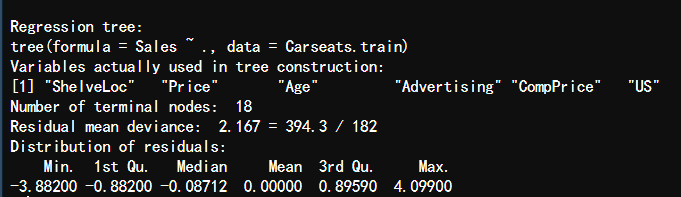
（b）问题（略）

install.packeges(“tree”)

library(tree)

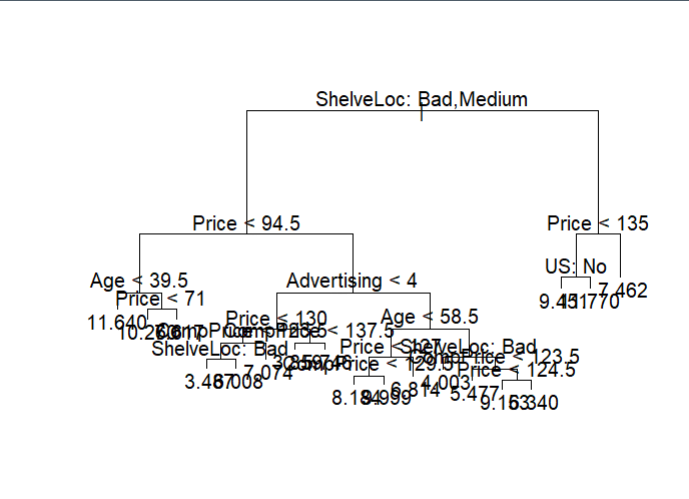
tree.carseats = tree(Sales ~ ., data = Carseats.train)

summary(tree.carseats)



plot(tree.carseats)

text(tree.carseats, pretty = 0)



pred.carseats = predict(tree.carseats, Carseats.test)

mean((Carseats.test$Sales - pred.carseats)^2)



测试MSE大概为4.92

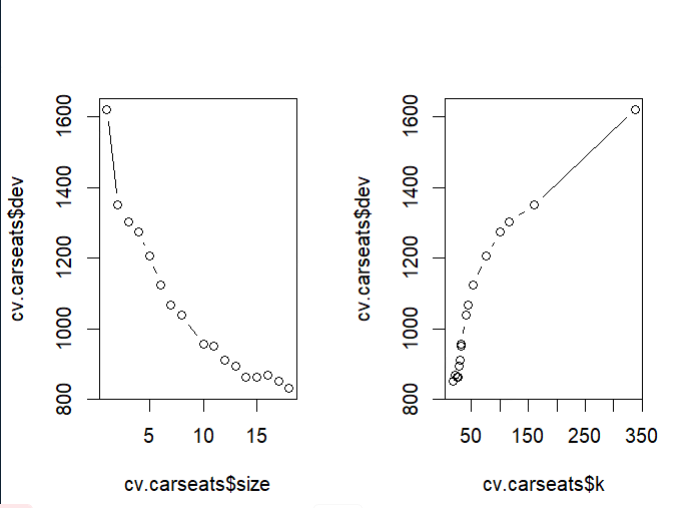
（c）问题（略）

cv.carseats = cv.tree(tree.carseats, FUN = prune.tree)

par(mfrow = c(1, 2))

plot(cv.carseats$size, cv.carseats$dev, type = "b")

plot(cv.carseats$k, cv.carseats$dev, type = "b")



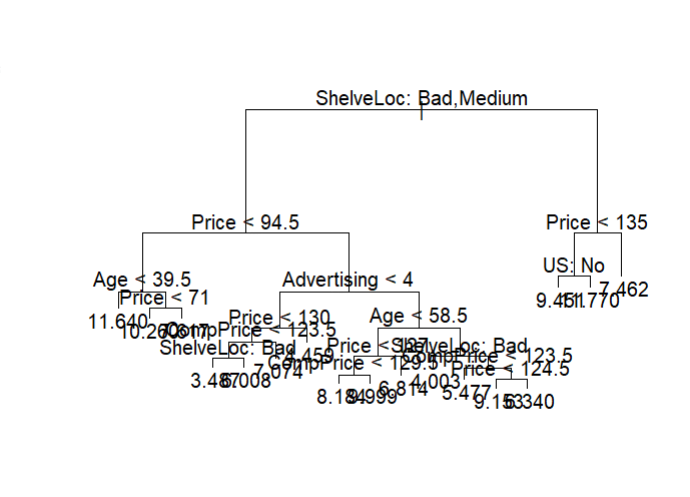
# Best size = 17

pruned.carseats = prune.tree(tree.carseats, best = 17)

par(mfrow = c(1, 1))

plot(pruned.carseats)

text(pruned.carseats, pretty = 0)



pred.pruned = predict(pruned.carseats, Carseats.test)

mean((Carseats.test$Sales - pred.pruned)^2)



修剪树将测试MSE降低到4.82。剪枝确实降低了测试错误率。

（d）问题（略）

install.packeges(“rangdomForest”)

libary(“rangdomForest”)

bag.carseats = randomForest(Sales ~ ., data = Carseats.train, mtry = 10, ntree = 500,

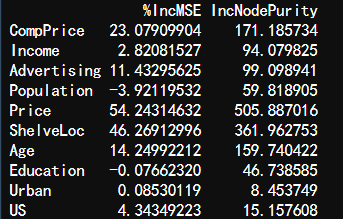
importance = T)

bag.pred = predict(bag.carseats, Carseats.test)

mean((Carseats.test$Sales - bag.pred)^2)



importance(bag.carseats)



装袋的测试MSE为2.65。我们还看到，price、ShelveLoc和age是Sale的三个最重要的预测指标。

（e）问题（略）

rf.carseats = randomForest(Sales ~ ., data = Carseats.train, mtry = 5, ntree = 500,

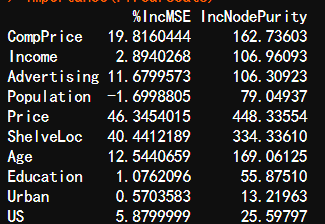
importance = T)

rf.pred = predict(rf.carseats, Carseats.test)

mean((Carseats.test$Sales - rf.pred)^2)



importance(rf.carseats)



在这种情况下，随机森林得到的测试集的MSE为2.70。改变m会使测试MSE在2.6到3之间变化。我们再次看到，Price、ShelveLoc和Age是销售的三个最重要的预测因素。

9.问题（略）

（a）问题（略）

library(ISLR)

attach(OJ)

set.seed(1013)

train = sample(dim(OJ)[1], 800)

OJ.train = OJ[train, ]

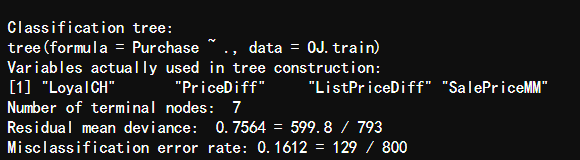
OJ.test = OJ[-train, ]

（b）问题（略）

library(tree)

oj.tree = tree(Purchase ~ ., data = OJ.train)

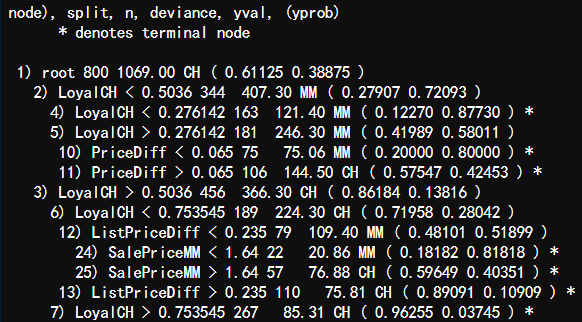
summary(oj.tree)



该树使用四个变量:LoyalCH、PriceDiff、ListPriceDiff和SalePriceMM。它有7个终端节点。树的训练错误率(误分类误差)为0.161。

（c）问题（略）

oj.tree

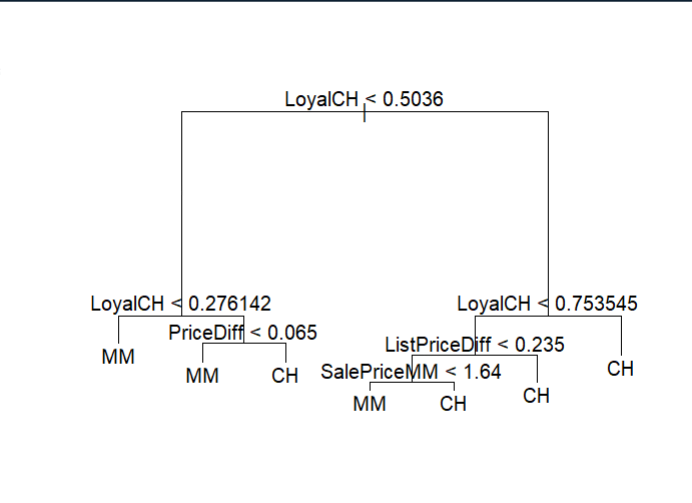


让我们选择标记为“10)”的终端节点。这个节点上的分割变量是PriceDiff。该节点的分裂值为0.065。在这个节点下面的子树中有75个点。该节点下方区域中所有点的偏差为75.06。行中的A \*表示这实际上是一个终端节点。该节点的预测值为Sales = MM，该节点中约有20%的点的Sales值为CH。其余80%的点有MM作为销售额的值。

（d）问题（略）

plot(oj.tree)

text(oj.tree, pretty = 0)

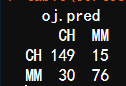


LoyalCH是树中最重要的变量，实际上前3个节点包含LoyalCH。如果LoyalCH<0.27，树预测MM。如果LoyalCH>0.76，树预测CH。对于LoyalCH的中间值，决策取决于PriceDiff、SalePriceDiff和ListPriceDiff的值。

（e）问题（略）

oj.pred = predict(oj.tree, OJ.test, type = "class")

table(OJ.test$Purchase, oj.pred)

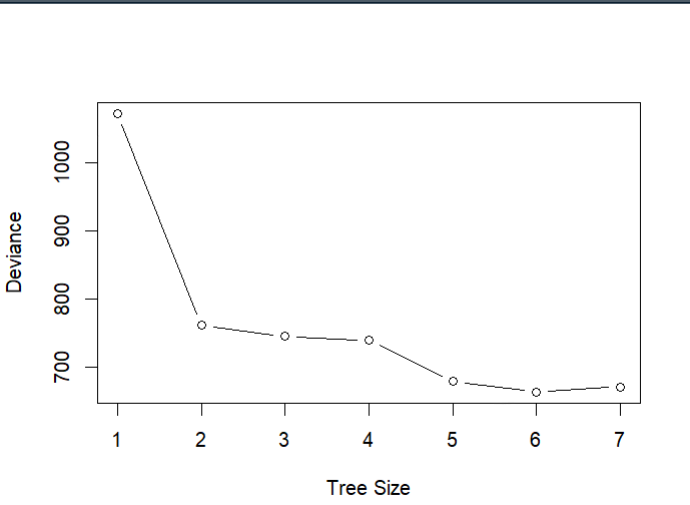


（f）问题（略）

cv.oj = cv.tree(oj.tree, FUN = prune.tree)

（g）问题（略）

plot(cv.oj$size, cv.oj$dev, type = "b", xlab = "Tree Size", ylab = "Deviance")



（h）问题（略）

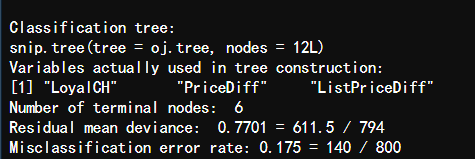
Size为6的交叉验证误差最小

1. 问题（略）

oj.pruned = prune.tree(oj.tree, best = 6)

（j）问题（略）

summary(oj.pruned)



剪枝树的误差为0.175。

（k）问题（略）

pred.unpruned = predict(oj.tree, OJ.test, type = "class")

misclass.unpruned = sum(OJ.test$Purchase != pred.unpruned)

misclass.unpruned/length(pred.unpruned)



pred.pruned = predict(oj.pruned, OJ.test, type = "class")

misclass.pruned = sum(OJ.test$Purchase != pred.pruned)

misclass.pruned/length(pred.pruned)



修剪过的树与未修剪过的树测试错误率不同，未修剪过的错误率为0.166，修剪过的错误率为0.2。

11.问题（略）

（a）问题（略）

library(ISLR)

train = 1:1000

Caravan$Purchase = ifelse(Caravan$Purchase == "Yes", 1, 0)

Caravan.train = Caravan[train, ]

Caravan.test = Caravan[-train, ]

（b）问题（略）

install.packages(“gbm”)

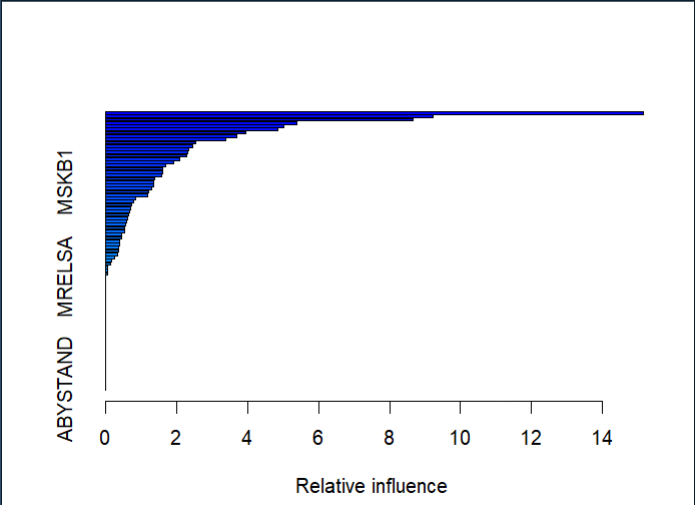
libary(gbm)

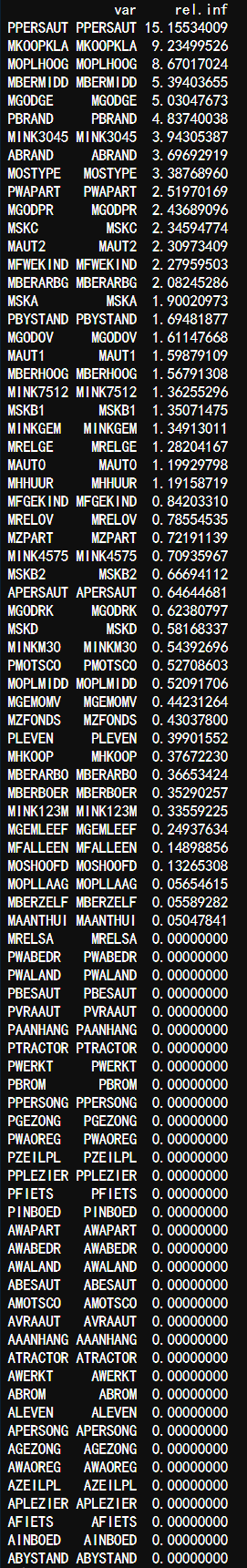
set.seed(342)

boost.caravan = gbm(Purchase ~ ., data = Caravan.train, n.trees = 1000, shrinkage = 0.01,

distribution = "bernoulli")

summary(boost.caravan)





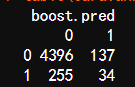
PPERSAUT、MKOOPKLA和MOPLHOOG是最重要的三个变量。

（c）问题（略）

boost.prob = predict(boost.caravan, Caravan.test, n.trees = 1000, type = "response")

boost.pred = ifelse(boost.prob > 0.2, 1, 0)

table(Caravan.test$Purchase, boost.pred)



34/（137+34）=0.1988

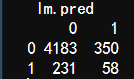
预测会购物的人中，最终会购物的人大概有20%

lm.caravan = glm(Purchase ~ ., data = Caravan.train, family = binomial)

lm.prob = predict(lm.caravan, Caravan.test, type = "response")

lm.pred = ifelse(lm.prob > 0.2, 1, 0)

table(Caravan.test$Purchase, lm.pred)



58/（350+58）= 0.1422

使用逻辑回归预测会购物的人中，约有14%的人最终真的买了。