481 HW 5

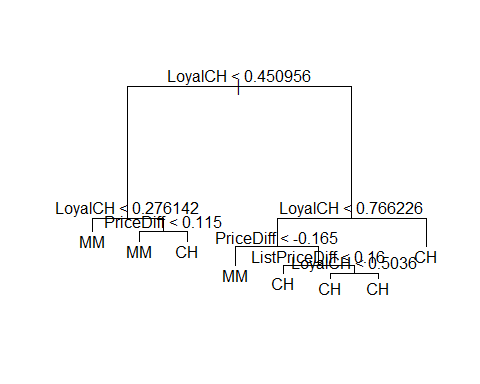
Nicholas Thompson

10/26/2021

#use OJ dataset  
set.seed(1138)  
samp=sample(1:nrow(OJ), 800)  
  
OJ.tr=OJ[samp,]  
OJ.te=OJ[-samp,]  
  
#1  
tree1=tree(Purchase~.,data = OJ.tr)  
summary(tree1)

##   
## Classification tree:  
## tree(formula = Purchase ~ ., data = OJ.tr)  
## Variables actually used in tree construction:  
## [1] "LoyalCH" "PriceDiff" "ListPriceDiff"  
## Number of terminal nodes: 8   
## Residual mean deviance: 0.7503 = 594.2 / 792   
## Misclassification error rate: 0.1538 = 123 / 800

#8 terminal nodes, missclassification error rate of 0.1475  
#Variables LoyalCH, SpecialCH, SalePriceMM, ListPriceDiff, PriceDiff were the  
#only ones used  
  
#2  
plot(tree1);text(tree1)



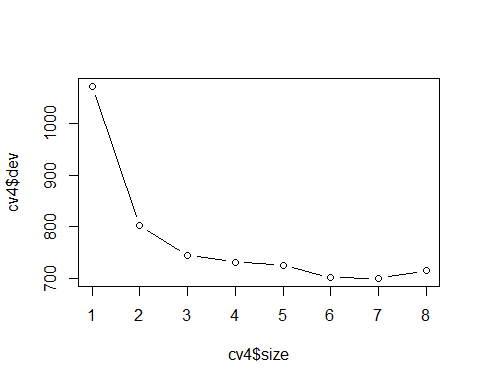
#When Loyal CH < 0.276142, the predicted value for SalePriceMM is 1 and SpecialCH is 0  
  
#3  
pred3=predict(tree1,OJ.te,type="class")  
table(pred3,OJ.te$Purchase)

##   
## pred3 CH MM  
## CH 142 34  
## MM 24 70

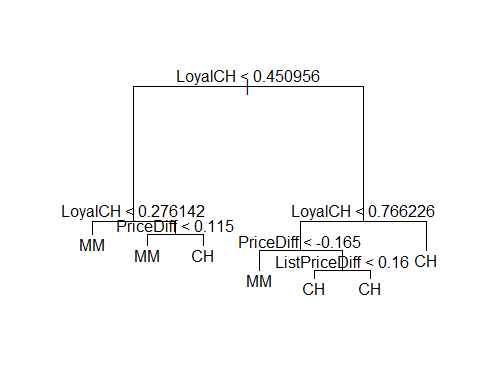
mean(pred3 != OJ.te$Purchase)#test error rate

## [1] 0.2148148

#0.2148148  
#4  
cv4=cv.tree(tree1)  
plot(cv4$size, cv4$dev, type="b")



#It appears a tree with 7 terminal nodes produces the lowest error rate  
  
#5  
prune5=prune.tree(tree1,best=7)   
plot(prune5);text(prune5)



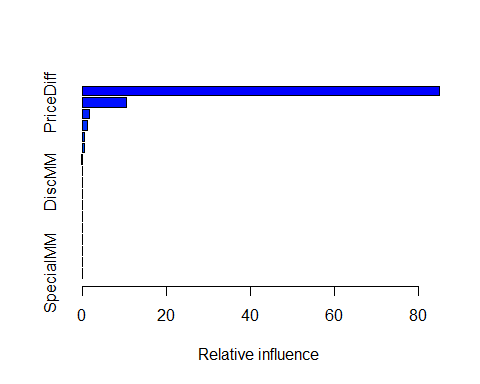
#6  
pred6=predict(prune5,OJ.te,type="class")  
table(pred6,OJ.te$Purchase)

##   
## pred6 CH MM  
## CH 141 34  
## MM 25 70

mean(pred6 != OJ.te$Purchase)#test error rate

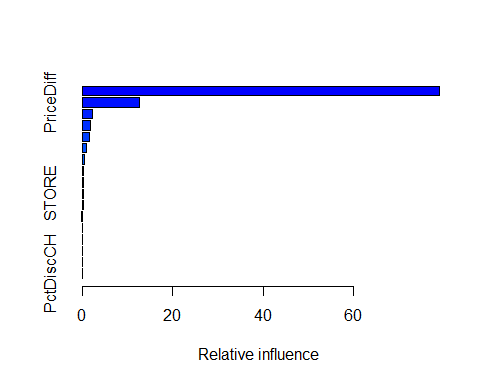
## [1] 0.2185185

#0.2185185  
#it appears the original tree has the lower test error  
  
#7  
boost7=gbm(Purchase~.,data=OJ.tr,distribution="gaussian",n.trees=1000, interaction.depth=4, shrinkage = 0.0001)  
boost7b=gbm(Purchase~.,data=OJ.tr,distribution="gaussian",n.trees=1000, interaction.depth=4, shrinkage = 0.01)  
boost7c=gbm(Purchase~.,data=OJ.tr,distribution="gaussian",n.trees=1000, interaction.depth=4, shrinkage = 0.1)  
boost7d=gbm(Purchase~.,data=OJ.tr,distribution="gaussian",n.trees=1000, interaction.depth=4, shrinkage = 0.001)  
summary(boost7)



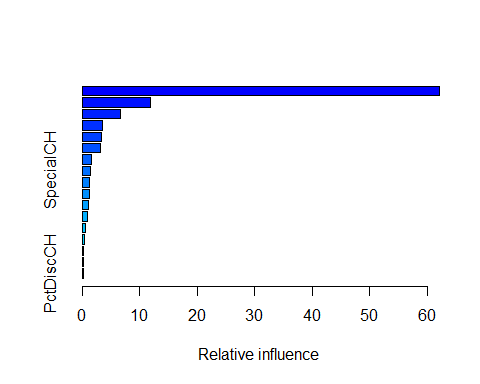
## var rel.inf  
## LoyalCH LoyalCH 85.025431535  
## PriceDiff PriceDiff 10.434912823  
## ListPriceDiff ListPriceDiff 1.793700428  
## SalePriceMM SalePriceMM 1.334216223  
## StoreID StoreID 0.439079250  
## SpecialCH SpecialCH 0.437337958  
## DiscCH DiscCH 0.121394644  
## PriceMM PriceMM 0.097856089  
## DiscMM DiscMM 0.088047060  
## SalePriceCH SalePriceCH 0.071425273  
## STORE STORE 0.068494664  
## WeekofPurchase WeekofPurchase 0.047444893  
## PctDiscCH PctDiscCH 0.012208909  
## PriceCH PriceCH 0.010232260  
## Store7 Store7 0.010104753  
## PctDiscMM PctDiscMM 0.008113237  
## SpecialMM SpecialMM 0.000000000

summary(boost7d)



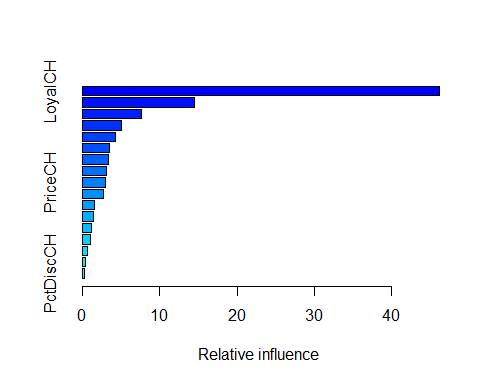
## var rel.inf  
## LoyalCH LoyalCH 79.00308332  
## PriceDiff PriceDiff 12.61462371  
## ListPriceDiff ListPriceDiff 2.14778496  
## SalePriceMM SalePriceMM 1.73773861  
## StoreID StoreID 1.66591635  
## SpecialCH SpecialCH 0.89459541  
## WeekofPurchase WeekofPurchase 0.40169712  
## DiscCH DiscCH 0.34353233  
## SalePriceCH SalePriceCH 0.27976712  
## STORE STORE 0.26061568  
## PriceMM PriceMM 0.23648291  
## DiscMM DiscMM 0.13033426  
## Store7 Store7 0.09617769  
## PriceCH PriceCH 0.09186988  
## SpecialMM SpecialMM 0.05596626  
## PctDiscMM PctDiscMM 0.02461239  
## PctDiscCH PctDiscCH 0.01520201

summary(boost7b)



## var rel.inf  
## LoyalCH LoyalCH 62.1526317  
## PriceDiff PriceDiff 11.8542280  
## WeekofPurchase WeekofPurchase 6.6524838  
## ListPriceDiff ListPriceDiff 3.5792067  
## StoreID StoreID 3.3585261  
## SalePriceMM SalePriceMM 3.2197793  
## STORE STORE 1.6659692  
## SpecialCH SpecialCH 1.3647268  
## SalePriceCH SalePriceCH 1.2934390  
## PriceMM PriceMM 1.2421468  
## DiscMM DiscMM 1.0188177  
## PriceCH PriceCH 0.9847604  
## DiscCH DiscCH 0.5439996  
## SpecialMM SpecialMM 0.4041491  
## PctDiscMM PctDiscMM 0.2621750  
## Store7 Store7 0.2489674  
## PctDiscCH PctDiscCH 0.1539933

summary(boost7c)



## var rel.inf  
## LoyalCH LoyalCH 46.1581688  
## WeekofPurchase WeekofPurchase 14.5052254  
## PriceDiff PriceDiff 7.6878207  
## ListPriceDiff ListPriceDiff 5.0738516  
## SalePriceMM SalePriceMM 4.2669442  
## StoreID StoreID 3.5087835  
## PriceMM PriceMM 3.4320313  
## STORE STORE 3.0717977  
## PriceCH PriceCH 2.9606126  
## SalePriceCH SalePriceCH 2.6780516  
## DiscMM DiscMM 1.6115914  
## SpecialCH SpecialCH 1.4456897  
## DiscCH DiscCH 1.1834134  
## SpecialMM SpecialMM 1.1126724  
## PctDiscMM PctDiscMM 0.6781355  
## Store7 Store7 0.3565062  
## PctDiscCH PctDiscCH 0.2687041

#8  
#increasing shrinkage parameter leads to increased influence of variables  
#from lowest shrinkage, we can see that the 3 most influential variables are LoyalCH, PriceDiff and ListPriceDiff  
#7d:LoyalCH,PriceDiff, ListPriceDiff, SalePriceMM  
#7b: LoyalCH, PriceDiff, WeekofPurchase (ListPriceDiff is now #5)  
#7c: LoyalCH, Week of Purchase, PriceDiff, SalePriceMM, ListPriceDiff  
#I would say that LoyalCH, PriceDiff and WeekofPurchase, and ListPriceDiff are the most important variables across all shrinkages  
  
#9  
yhat.boost =predict(boost7b,newdata=OJ[-samp,])

## Using 1000 trees...

test=OJ[-samp, "Purchase"]  
mean((yhat.boost-test)^2)

## Warning in Ops.factor(yhat.boost, test): '-' not meaningful for factors

## [1] NA

#my results produce an NA and I'm not sure how to fix it  
  
#10  
rf.10=randomForest(Purchase~.,data=OJ,subset=samp, mtry=5,importance =TRUE)   
yhat.rf = predict(rf.10,newdata=OJ[-samp,])   
plot(yhat.rf,test)

