Consider the `Carseats` data set from the `ISLR` library. This is a data set containing sales of child car seats at 400 different stores. It includes 11 variables, one of which is `Sales`. It is of interest to predict if the sales of a store are high (more than 8000 car seats to sell) based on 10 predictors (categorical or continuous). Therefore `high` is a categorical variable that is to be created based on `Sales` values. It is also of interest to identify which variables are most useful to predict high sales. Divide the dataset into a training (50%) and a test set.

a) Use function `tree` from library `tree` to fit a categorical tree.

b) Which predictors are found most important?

c) Report the test MSPE error rate.

d) Use Cross validation on the classification error rate to find the best number of terminal nodes. Prune the tree to find MSPE for the test set.

```
library(tree)     # tree() cv.tree()
library(ISLR)     # data set

?Carseats

d0=Carseats
str(d0)
# '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 ...

# there are 10 predictors, some categorical
n = nrow(d0)

# create categorical response
high=ifelse(d0$Sales<=8,"No","Yes")
d1=data.frame(d0,high)

# tree - 2 predictors, full dataset
#==================================================================
tree0=tree(high~Price+Advertising,d1)

# scatterplot on predictors space (Price~Advertising does not work?)
plot(Advertising~Price,d1,pch=19,cex=0.6,col="red")
# regions and predicted category
partition.tree(tree0,add = T,col="blue")

# tree plot
plot(tree0)
text(tree0,cex=0.75)
```

```
# full model -Sales
#===================================================================
tree1=tree(high~.-Sales,d1)
summary(tree1)
# 8 Variables actually used in tree construction
# [1] "ShelveLoc" "Price"  "Income" "CompPrice" "Population" "Advertising" "Age" "US"
# Number of terminal nodes:  27
# Residual mean deviance:  0.4575 = 170.7 / 373
# Misclassification error rate: 0.09 = 36 / 400

# Total deviance 170.7 is sum of deviances of terminal nodes
#   with 400-27 = 373 dof

# misclassifications
ypred = predict(tree1,d1,type="class")
table(ypred,d1$high)
#ypred   No Yes
#   No  213  13
#  Yes   23 151

# there are 36 misclassified obs
# training error rate 36/400 = 0.09

 names(tree1)
# [1] "frame"   "where"   "terms"   "call"    "y"       "weights"

dim(tree1$frame)   # [1] 53  6
head(tree1$frame)
#          var   n        dev yval splits.cutleft splits.cutright  yprob.No yprob.Yes
# 1  ShelveLoc 400 541.486837   No            :ac              :b 0.5900000 0.4100000
# 2      Price 315 390.591685   No          <92.5           >92.5 0.6888889 0.3111111
# 4     Income  46  56.534305  Yes            <57             >57 0.3043478 0.6956522
# 8  CompPrice  10  12.217286   No         <110.5          >110.5 0.7000000 0.3000000
# 16    <leaf>   5   0.000000   No                                1.0000000 0.0000000
# 17    <leaf>   5   6.730117  Yes                                0.4000000 0.6000000

tail(tree1$frame)
#       var  n      dev yval splits.cutleft splits.cutright   yprob.No  yprob.Yes
# 24 <leaf>  8  0.00000  Yes                                0.00000000 1.00000000
# 25 <leaf>  9 11.45726   No                                0.66666667 0.33333333
# 13 <leaf> 51 16.87524  Yes                                0.03921569 0.96078431
# 7  Income 17 22.07444   No            <46             >46 0.64705882 0.35294118
# 14 <leaf>  6  0.00000   No                                1.00000000 0.00000000
# 15 <leaf> 11 15.15820  Yes                                0.45454545 0.54545455

# Factor ShelveLoc is most important classifier
# <leaf> rows are terminal nodes
```

```
plot(tree1)
text(tree1,cex=0.6,pretty=0)   # pretty shows class names on tree
title("Tree from the full dataset")

# 1st branch differentiates good locations from bad & medium
# Bad & medium is the left-hand branch

# training and test sets
#=========================================================================
set.seed(2)
n = 1:nrow(d1)
train=sample(n, 200)
d1.test=d1[-train,]      # test set with response
y.test=high[-train]   # response in test set

# training model
tree2=tree(high~.-Sales,d1,subset=train)

summary(tree2)
# Classification tree:
# tree(formula = high ~ . - Sales, data = d1, subset = train)
# 7 Variables actually used in tree construction:
# [1] "ShelveLoc" "Price" "Income" "Age" "Advertising" "CompPrice" "Population"
# Number of terminal nodes:  19
# Residual mean deviance:  0.4282 = 77.51 / 181
# Misclassification error rate: 0.105 = 21 / 200

plot(tree2)
text(tree2,cex=0.6,pretty=0)
title("Tree from the training set")

# test error rate
pred2=predict(tree2,d1.test,type="class")
# prediction is classification "No" or "Yes" high sales
# use type="class" for classification trees

table(pred2,y.test)
#      y.test
# pred2 No Yes
#   No  86  27
#   Yes 30  57

# out of 200 obs

aux=prop.table(table(pred2,y.test))
sum(diag(aux))    #[1] 0.715     accuracy test rate
```

```
# CV on (mis)classification error rate
#========================================================================
set.seed(3)
tree3=cv.tree(tree2,FUN=prune.misclass)  # compare misclassifications

names(tree3)  # [1] "size"   "dev"    "k"      "method"

tree3$size        #[1] 19 17 14 13  9  7  3  2  1
tree3$dev         #[1] 55 55 53 52 50 56 69 65 80
round(tree3$k,2)  #[1]-Inf  0.00  0.67  1.00  1.75  2.00  4.25  5.00 23.00

# size values are n. terminal nodes
# dev is n. obs. misclassified  (the CV error rate)
# k is alpha = complexity parameter
# tree with 9 terminal nodes has lowest dev (CV error rate)


# cv on deviance - to compare

tree4=cv.tree(tree2)
tree4$size
#[1] 19    16    14    13    12    11    10     9     8     7     6     5     3     2     1
round(tree4$dev,1)
# 565.0 512.3 502.5 485.8 485.8 409.9 395.3 366.6 342.5 343.9 335.3 311.8 310.1 290.4 278.6

# plot n. of misclassifications vs size, k
par(mfrow=c(1,2))
plot(tree3$dev~tree3$size,type="l");grid()
plot(tree3$dev~tree3$k,type="l");grid()
# smallest n. of misclassifications
# with 9 terminal nodes
# with k = 1.75

# use FUN=prune.misclass to
# prune training tree2 to 9 terminal nodes
prune9=prune.misclass(tree2,best=9)
par(mfrow=c(1,1))
plot(prune9)
text(prune9,cex=0.75,pretty=0)
title("CV optimized tree with 9 nodes")
```

```
# test error of the pruned tree
yhat9=predict(prune9,d1.test,type="class")
table(yhat9,y.test)
#      y.test
# yhat9 No Yes
#   No  94  24
#   Yes 22  60

aux=prop.table(table(yhat9,y.test))
sum(diag(aux))    #[1] 0.77     accuracy test rate


# cluster plot
#==========================================================================

summary(prune9)
# Classification tree:
# snip.tree(tree = tree2, nodes = c(159L, 6L, 8L, 38L))
# Variables actually used in tree construction:
# [1] "ShelveLoc"   "Price"   "Advertising" "Age"  "CompPrice"
# Number of terminal nodes:  9
# Residual mean deviance:  0.8103 = 154.8 / 191
# Misclassification error rate: 0.155 = 31 / 200

# dataset based on best 5 variables and response
d2.test = d1.test[,c(2,4,6,7,8,12)]
d3.test = data.frame(d2.test,yhat9)

# categorical to numerical
d3.test$ShelveLoc = as.numeric(d3.test$ShelveLoc)
d3.test$high = as.numeric(d3.test$high)

# plot PC2 vs PC1 coordinates of all rows in test set
library(cluster)  # clusplot

# group rows using yhat9
clusplot(d3.test,d3.test$yhat9,color=T,plotchar=F)
clusplot(d3.test,d3.test$yhat9,color=T,shade=T,labels=2,main="",cex=0.5,plotchar=F)
grid()
legend("bottomleft", legend = c("Yes","No"), pch=1, col=1:2,cex=0.75)
```

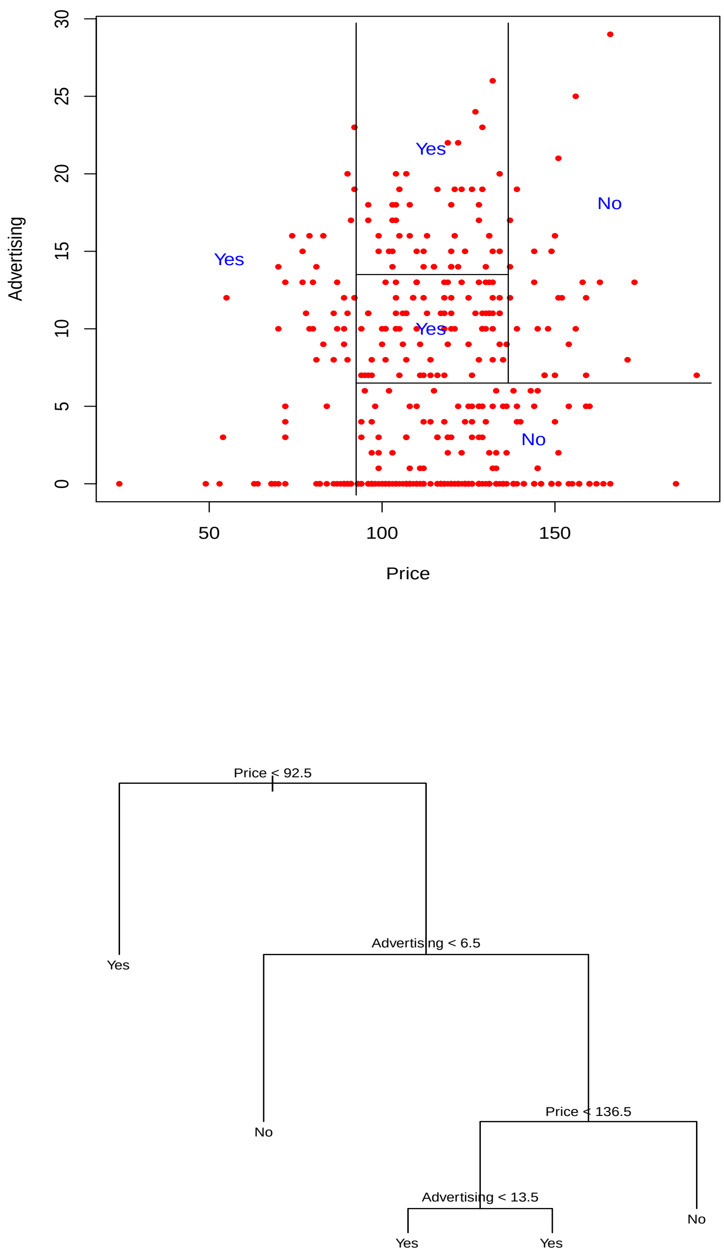
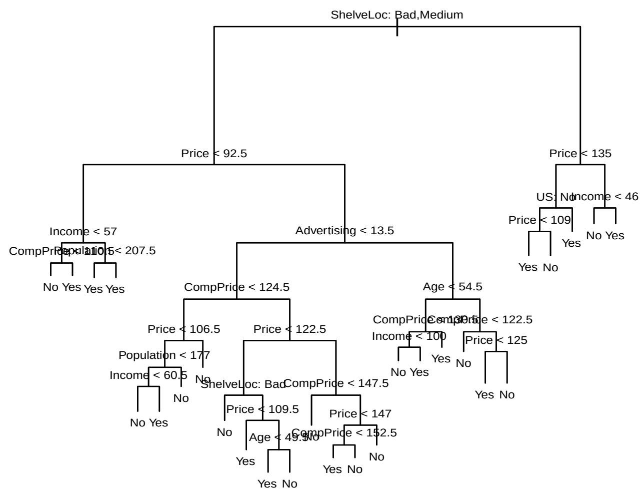
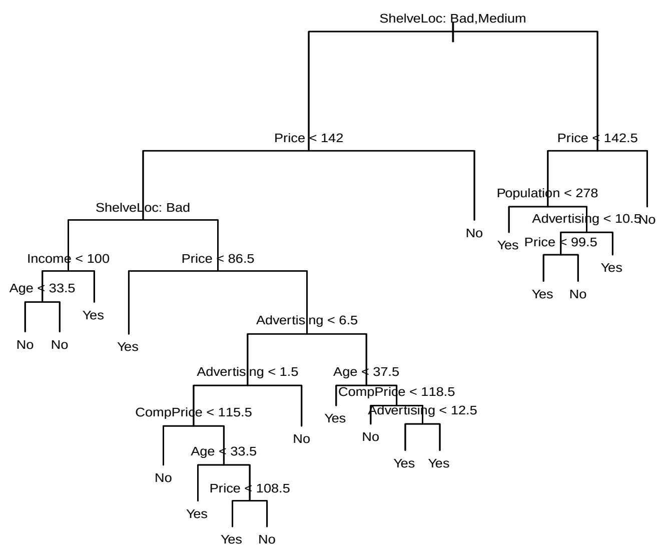


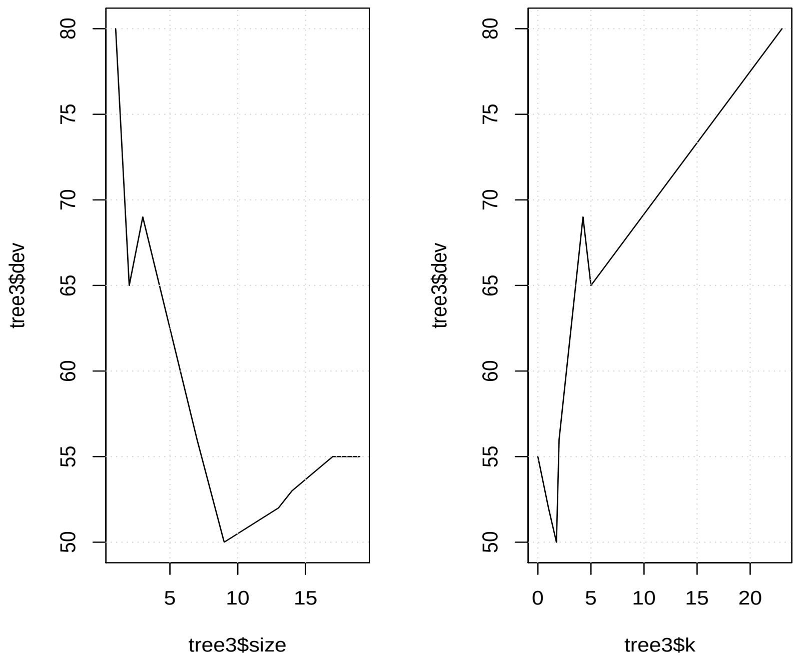
Figure 1: Categorical tree with two numerical predictors

Tree from the full dataset

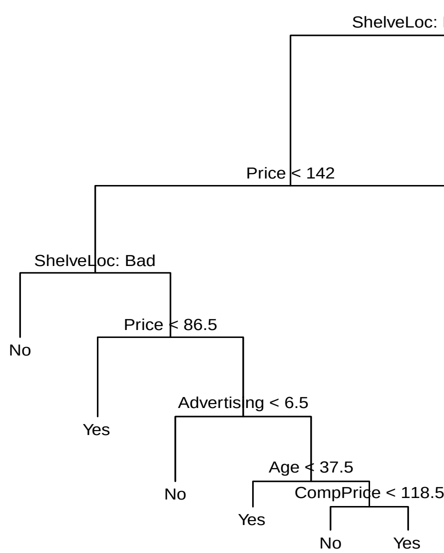


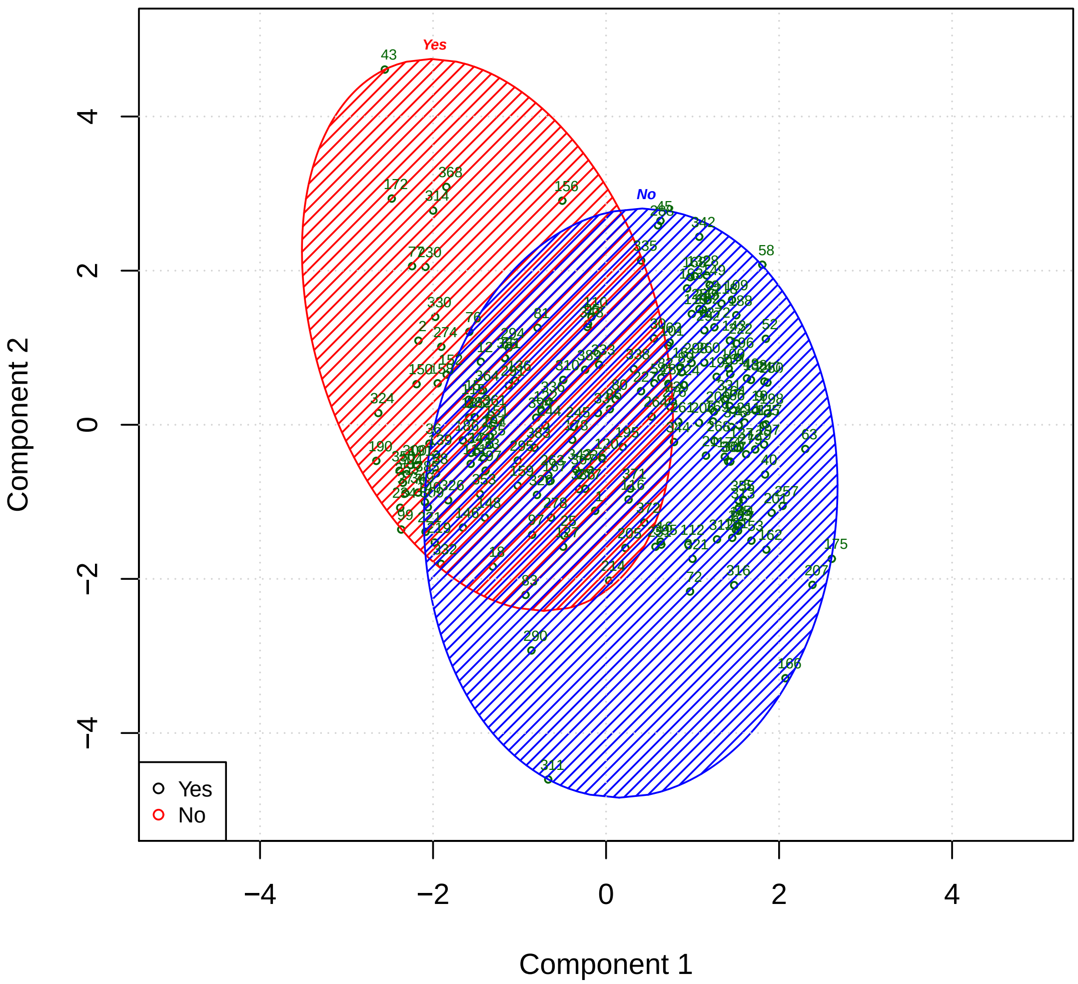
Tree from the training set





CV optimized tree with 9 nodes





These two components explain 51.6 % of the point variability.