|  |  |
| --- | --- |
| Big Data and Machine Learning in Logistics | 2nd semester 202 |
| Student number: 502115907 | Student name: Nolwenn PIGEON 놀웬 피지언 |

**R Self-work submission 5**

Note: This assignment is based on the lectures on Week 13.

1. Grow classification tree using Carseats dataset. The prediction variable is created using ‘Sales’ variable by making categorical variable using ifelse function. Provide interpretation of the classification tree.

Answer)

LOADING DATA

Urban & US are categorical data but more or less predictor variables here.

Y is the variable we are interested to predict. Here Sales can be our interest. We can predict sales based on predictors.

We can categorize Sales variable into categorical data.

We want to see how the variable is distributed by drawing an histogram: 8 would be the mean or median.

Let’s create our new categorical variable High

Now we need to combine the new categorical data in the original data using combine or data frame. Now we have 12 variables (+1 in the fix carseats)

> library(ISLR)

> library(tree)

> attach(Carseats)

> fix(Carseats)

> ?Carseats

> names(Carseats)

[1] "Sales" "CompPrice" "Income" "Advertising"

[5] "Population" "Price" "ShelveLoc" "Age"

[9] "Education" "Urban" "US"

> hist(Sales)

> High=ifelse(Sales<=8, "No","Yes")

> data.frame(Carseats, High)

GROWING TREE & DO CLASSIFICATION FOR HIGH

Function really similar to the linear model so we include all the variables except Sales cause we made High variable using Sales. If we have an error it means that the class of High is not the right one so we change it using as.factor

Then let’s see how the tree plot look like

> class(Carseats$High)

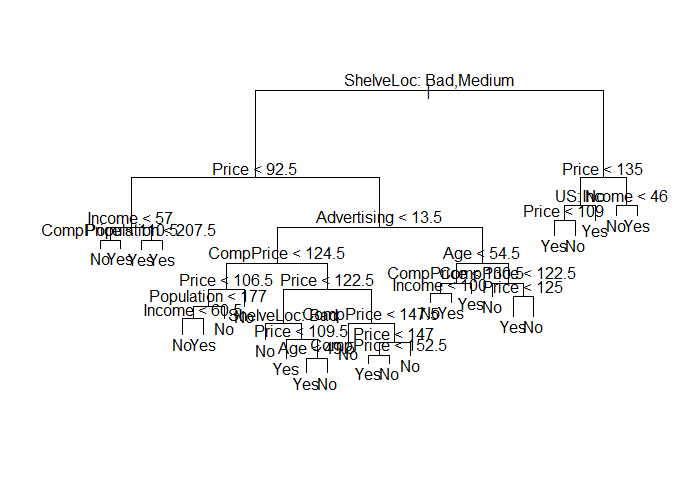
[1] "NULL"

> High<-as.factor(High)

> tree.carseats=tree(High~.-Sales, data=Carseats)

> plot(tree.carseats)

> text(tree.carseats, pretty=0)



INTERPRETATION

The most important indicator of Sales appears to be ShelveLoc (at the top of the tree). Since the first branch differentiates Good locations from Bad and medium locations.

Left side is good

Right side is bad and medium

Then Price is the 2nd important splitter

2. Prediction using the classification tree

1) Develop training and test dataset, build the classification tree using the training set and evaluate the performance on the test dataset

Answer)

We can’t use this tree cause there are too much data

We make training data and testing data and we re gonna evaluate the training data using the testing data.

In order to properly evaluate the performance of a classification tree on these data, we must estimate the test error rather than simply computing the training error. We split the observations into a training set and a test set, build the tree using the training set and evaluate its performance on the test data.

> set.seed(2)

> train=sample(1:nrow(Carseats),200)

> carseats.test=Carseats[-train,]

> High.test=High[-train]

> tree.carseats=tree(High~.-Sales, data=Carseats, subset = train)

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

> tree.pred=predict(tree.carseats, carseats.test, type="class")

> table(tree.pred, High.test)

High.test

tree.pred No Yes

No 104 33

Yes 13 50

> (104+50)/200

[1] 0.77

2) Consider pruning process for the tree for the improved results and show the results

Answer)

We want to improve the model by pruning. We do the cross-validation for tree.carseats. This means that we use argument FUN=prune.misclass. It’s to indicate that we want classification error rate to guide cross validation and pruning process rather than default for the symmetry function which is deviance. The $dev shows the deviance but it’s actually the classification error rate. Which one is the lowest ? 🡪 74. According to the first plot of size and dev we see that 21 gives the minimum. On the second plot of k and dev we see that the minimum is in 2. So by comparing those two plots we can prune the model where the best is [here] 21. Finally, we want to do a prediction again using the prune

> set.seed(3)

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

> cv.carseats

$size

[1] 21 19 14 9 8 5 3 2 1

$dev

[1] 74 76 81 81 75 77 78 85 81

$k

[1] -Inf 0.0 1.0 1.4 2.0 3.0 4.0 9.0 18.0

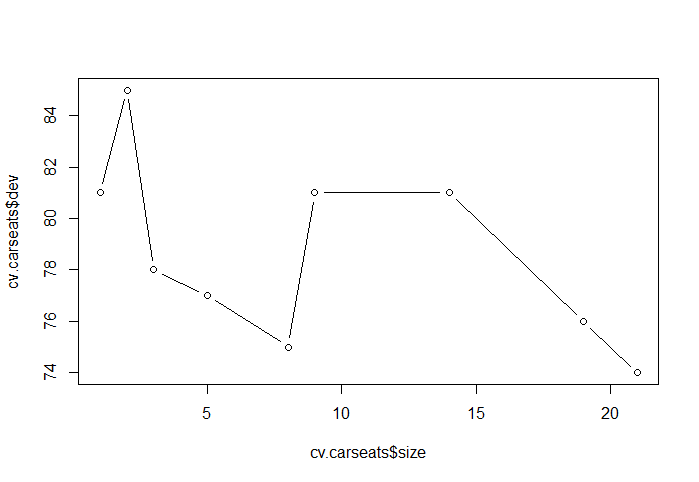
$method

[1] "misclass"

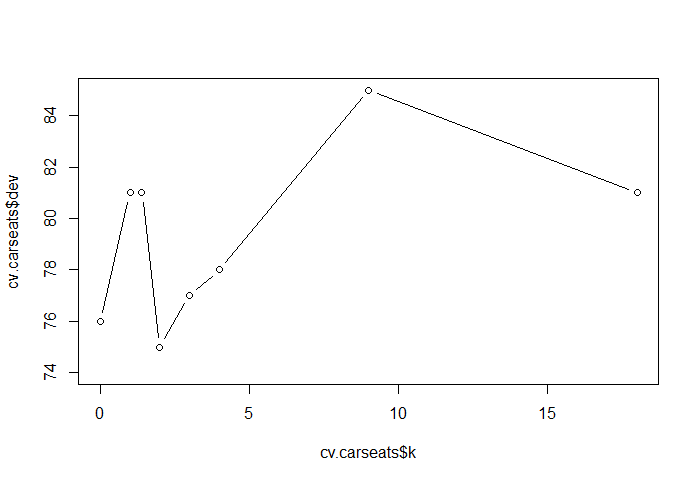
attr(,"class")

[1] "prune" "tree.sequence"

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



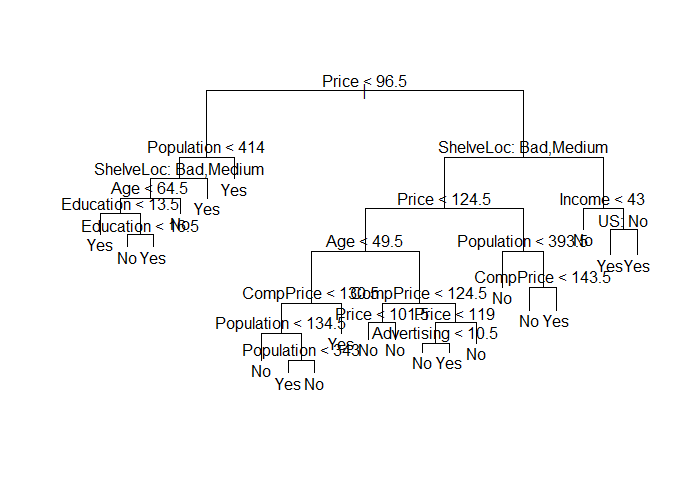
> plot(cv.carseats$k,cv.carseats$dev,type="b")



> prune.carseats=prune.misclass(tree.carseats, best=21)

> plot(prune.carseats)

> text(prune.carseats, pretty=0)



> prune.carseats=prune.misclass(tree.carseats, best=21)

> plot(prune.carseats)

> text(prune.carseats, pretty=0)

> tree.pred=predict(prune.carseats, carseats.test)

> tree.pred=predict(prune.carseats, carseats.test, type="c")

> tree.pred=predict(prune.carseats, carseats.test, type="class")

> table(tree.pred, High.test)

High.test

tree.pred No Yes

No 105 31

Yes 12 52

> (105+52)/200

[1] 0.785

🡪 The performances have just improved a little bit