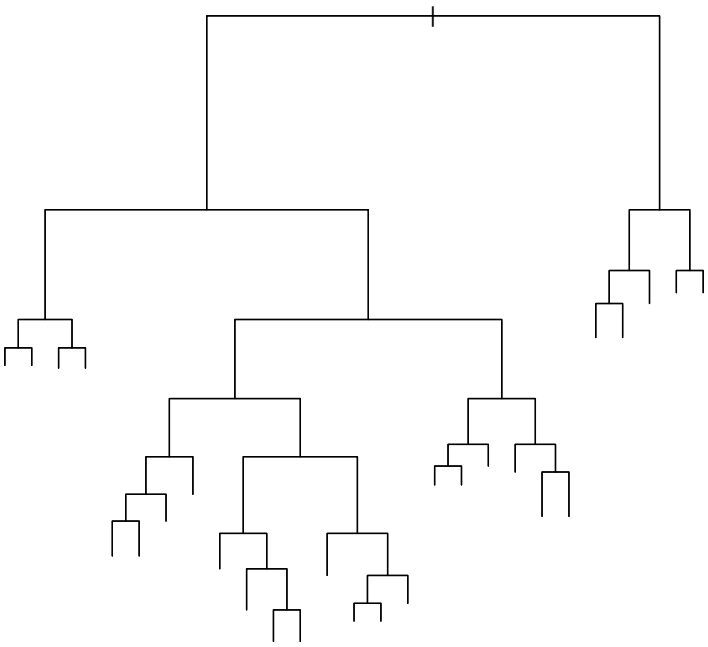
Assignment 5

Mats

May 5, 2015



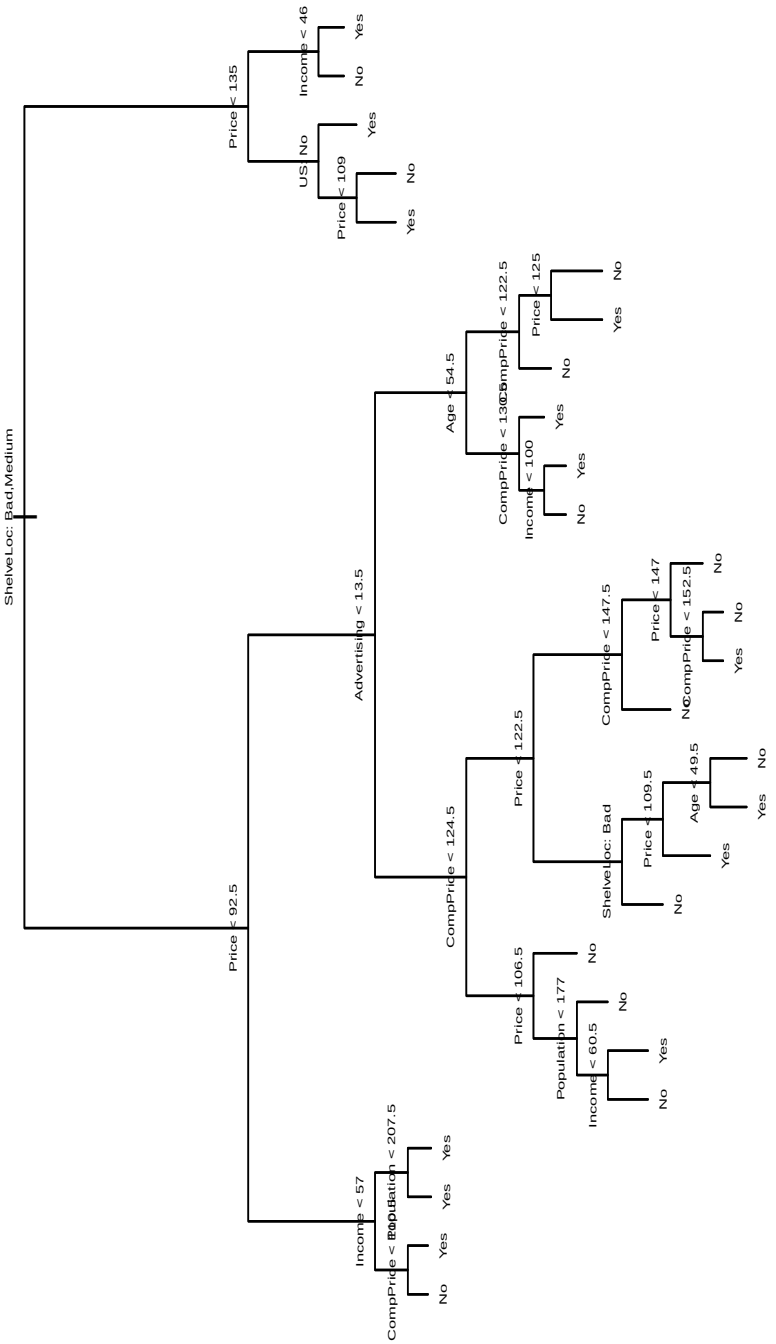
The data is loaded from ISLR packages, the files include 400 row and 11 columns. The sales variable is transform to a categorical variable, if the sales are higher than 8 than it "Yes", otherwise "No". The summary of the data is:

	Numeric	Mean	Variance	lowest	highest	Data
Sales	TRUE	7.50	7.98	0.00	16.27	qualitative
CompPrice	TRUE	124.97	235.15	77.00	175.00	qualitative
Income	TRUE	68.66	783.22	21.00	120.00	qualitative
Advertising	TRUE	6.63	44.23	0.00	29.00	qualitative
Population	TRUE	264.84	21719.81	10.00	509.00	qualitative
Price	TRUE	115.80	560.58	24.00	191.00	qualitative
ShelveLoc	FALSE					quantitative
Age	TRUE	53.32	262.45	25.00	80.00	qualitative
Education	TRUE	13.90	6.87	10.00	18.00	quantitative
Urban	FALSE					quantitative
US	FALSE					quantitative
High	FALSE					quantitative

There are both quantitative and qualitative variable and there are no missing value. The Shelveloc have 3 outputs, where it is bad, good and medium. The Urban variable has two outputs, "Yes" and "No". The US variable has also two outputs, "Yes" and "No". In this assignment we are using the r packages library(tree) to classify The high variable, by using the others variable as predictors except sales. The summary of the model is:

```
##
## Classification tree:
## tree(formula = High ~ . - Sales, data = Carseats, split = c("deviance"))
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Income"      "CompPrice"   "Population"
## [6] "Advertising" "Age"         "US"
## Number of terminal nodes:  27
## Residual mean deviance:  0.458 = 171 / 373
## Misclassification error rate: 0.09 = 36 / 400
```

We can see that Education and Urban is not in the complete model, this indicate that Education and Urban cannot explain High compare to the others variable. There are 27 nodes in the model, the nodes is the leaf in the tree and this mean there are 27 output in the tree. The misclassification error rate is how many error there is in the training sample, which mean we have 36 error of 400 observations (9 %) To get a better view how the model work, we can use the plot function in r. The plot of the tree is:


ShelveLoc: Bad,Medium
Price < 92.5
Price < 135
Income < 46
No
Yes
US: No
Yes
Price < 109
No
Yes
Advertising < 13.5
Age < 54.5
CompPrice < 122.5
Price < 125
No
Yes
CompPrice < 130.5
No
Income < 100
Yes
No
Yes
CompPrice < 124.5
Price < 122.5
CompPrice < 147.5
Price < 147
No
CompPrice < 152.5
No
Yes
No
ShelveLoc: Bad
Price < 109.5
Age < 49.5
No
Yes
Yes
Price < 106.5
No
Population < 177
No
Income < 60.5
Yes
No
Yes
Income < 57
CompPrice < 110.5
No
Yes
Population < 207.5
Yes
Yes

The most important variable in the model is ShelvLoc, where it is on the top of the tree and next most important variable is Price. This will continue until we get to the nodes in the tree, where the model include 27 nodes. I think the best advantage with the tree model is easy to understand and also it has a nice graphical representation.

A numeric summary of the model:

	var	n	dev	yval	splits.cutleft	splits.cutright	yprob. No	yprob. Yes
1	ShelveLoc	400	541.49	No	:ac	:b	0.5900	0.4100
2	Price	315	390.59	No	<92.5	>92.5	0.6889	0.3111
4	Income	46	56.53	Yes	<57	>57	0.3043	0.6957
8	CompPrice	10	12.22	No	<110.5	>110.5	0.7000	0.3000
16	<leaf>	5	0.00	No			1.0000	0.0000
17	<leaf>	5	6.73	Yes			0.4000	0.6000

The first row is for ShelveLoc, where it have 400 observations and the prediction is "No" (59 %). The split to the left is for bad and medium, and for the right is for good. The next row is for price, where is has 315 observations, and the split is at 92.5 units, where the prediction for this split is "no". The next row is income where it has 46 observations and the split is at 57 units and the probability is "Yes". The last to row in the table is the leaf and this is the output if is yes or no. Therefore it is possible to create the tree by just using this information.

Some example how the tree works:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	High
2	11.22	111.00	48.00	16.00	260.00	83.00	Good	65.00	10.00	Yes
10	4.69	132.00	113.00	0.00	131.00	124.00	Medium	76.00	17.00	No
50	10.61	157.00	93.00	0.00	51.00	149.00	Good	32.00	17.00	Yes

Observation nr 2:

- In the first step we look at the top at the tree, where we can find ShelvLoc, if is good than we move to the right otherwise left. For observation 2 the ShelveLoc is Good, which means right. Next step: The price is 83, this mean go left. The third step is if the US is yes, than go right. The output is "Yes", which is the same as the true value.

Observation nr 10:

- The ShelvLoc is Medium, go left. The price=124, go right. Advertising=0, go left. CompPrice=132, go right. Price=124, go right. CompPrice=132, go left. The output is "No" which is the same as the true value.

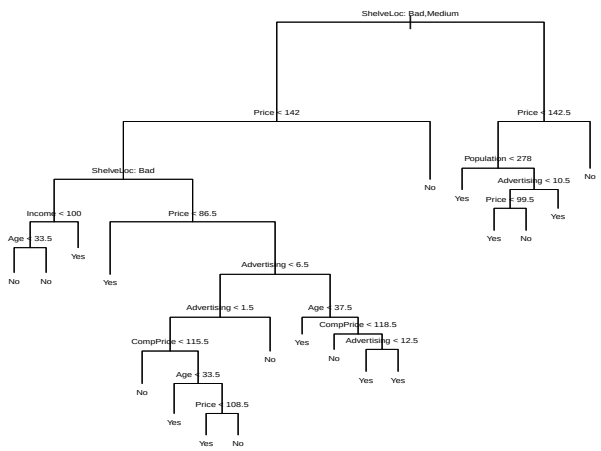
Observation 50:

- ShelvLoc=Good, go right. Price=149, go right. Income=93, go right. The output is "Yes" which is the same as the true value.

Training set and test set: The problem with this tree is train and test on the same data. To make a better tree I use one training sample and one test sample. I use 200 observations to create the tree and the other 200 observations to classify them by the model. The model for the training sample is:

```
##
## Classification tree:
## tree(formula = High ~ . - Sales, data = Carseats, subset = train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Income"      "Age"         "Advertising"
## [6] "CompPrice"   "Population"
## Number of terminal nodes:  19
## Residual mean deviance:  0.428 = 77.5 / 181
## Misclassification error rate: 0.105 = 21 / 200
```

and the plot is:



The model include 21 misclassification, where the error term is 10.5 %. There is also 19 nodes in this model. We can also see that ShelvLoc, Price, income, age, Advertising, CompPrice and population include in this model, which mean the model exclude education, Urban and US. I use this model to classify the test data, the confusion matrix is:

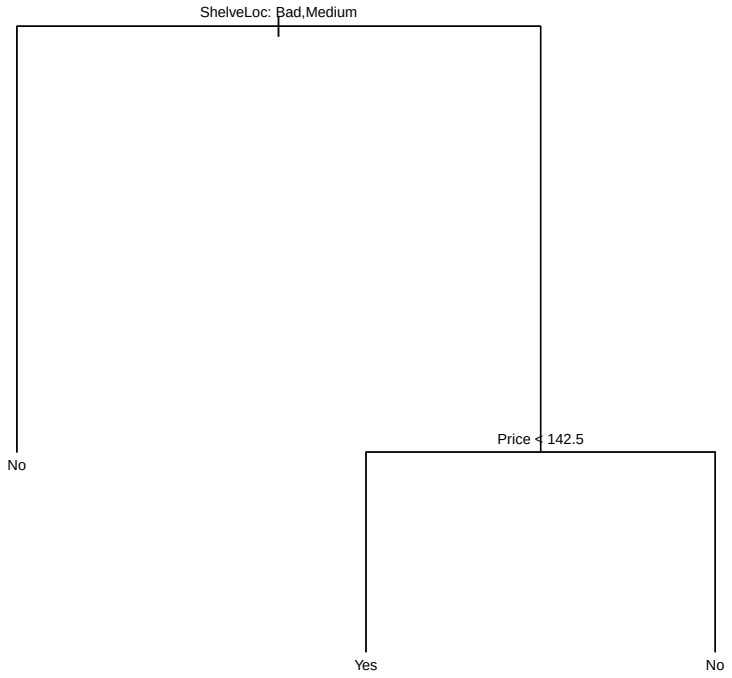
	No	Yes
No	71	42
Yes	52	35

The error rate is 28.5%. The model also predicts 113 units for "No" and 87

units for" Yes". This can been compared to the real value, where "No" is 116 units and "Yes" is 84 units. The error rate is always higher for the test data compare to the training data and this can been explain by the model is built on the training data and therefore it should have a lower error rate. The problem with this results is that the error rate is on one training sample and one test sample. Therefore I use the loop function to create a more stable error term. I loop 100 times and for each loop I construct a tree for the training data and then classify the test data.

This will return me 100 different error rate, therefore I divide it by 100. The mean rate is 27.385 %, which is a little bit lower than before. This can be explained by the randomness. The range for my loop is 19 % to 36 %, and the variance is 0.11. To create a more accurate tree we can prune the tree, this mean we set the number of nodes in the model. I use the prune.misclass() function in r and set best=3 for my model, this give us the best model when there are only tree nodes. The summary and the plot for the model is:

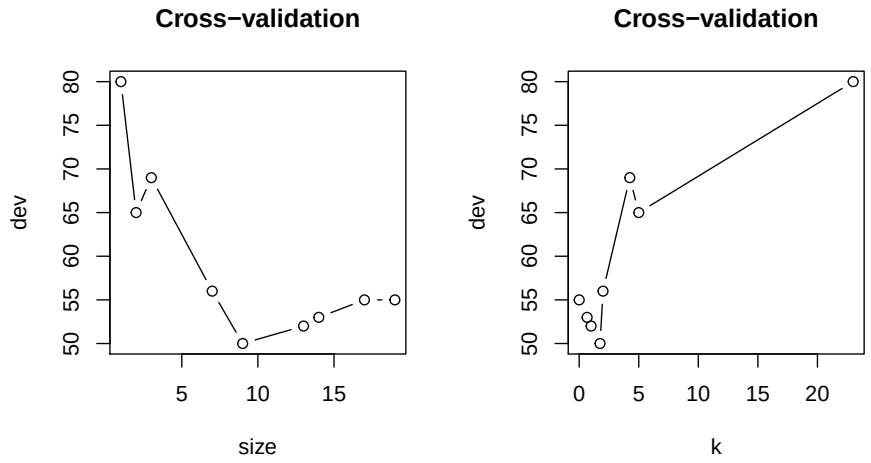
```
##
## Classification tree:
## snip.tree(tree = tree.carseats, nodes = c(6L, 2L))
## Variables actually used in tree construction:
## [1] "ShelveLoc" "Price"
## Number of terminal nodes:  3
## Residual mean deviance:  1.14 = 225 / 197
## Misclassification error rate: 0.26 = 52 / 200
```



We can see that the model with 200 observations (set.seed(2)) only include ShelvLoc and Price, the others variables is excluded. The misclassification for the training sample is 26 % and there are only 3 nodes. I use the new model to classify my test sample, the confusion matrix for the test data is:

	No	Yes
No	111	54
Yes	5	30

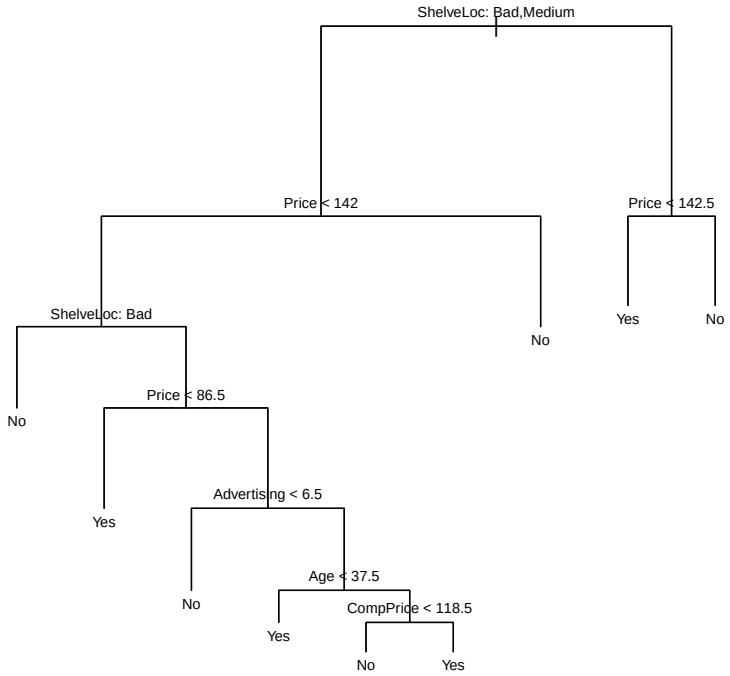
The error rate for my test sample is 29.5 %. To create a more stable result I use the loop function to create 100 new training and test sample, the error mean for the loop is 30.1 % and the variance is 0.13, this result can be compared to my other loop. Therefore my model with 19 nodes is better compare to the model with 3 nodes. To find the best model we use the cv.tree()function in r. This functions cross-validate my model and give us the best nodes size in return, where this method minimizes the means square error on the testing data. The cross validation is:



In the picture to the left, we can see the size is on the x-axis and dev is the y-axis, the dev is the deviance. To find the best model we have to minimize the deviance, where the lowest deviance is when the size is equal to 9. The other picture is for the cost-complexity parameters against deviance. Where the deviance is lowest when k=1.75, this mean the best number of nodes is also when the deviance is minimized against k. This mean if we increase or decrease k than the model is not optimal, Therefore i prune my model twith 9 nodes. the summary of the new model is:

```
##
## Classification tree:
## snip.tree(tree = tree.carseats, 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:  0.81 = 155 / 191
## Misclassification error rate: 0.155 = 31 / 200
```

and the plot is:



As we can see in the plot above the tree model have 9 nodes, and error rate is 15.5%. Therefore the best models include the ShelveLoc, Price, Advertising, Age and Comprice variable. I use this model to classify the test data, the confusion matrix is:

	No	Yes
No	94	24
Yes	22	60

Where the error rate is 23%, this indicates the model with 9 nodes is the best model compared to the others. I choose also to loop for this model to get a more stable result, the mean error rate for loop is 24.51 % and the variance is less than 0.001. Therefore the model with 9 nodes is the best tree.

Conclusion:

In this assignment we have "High"as the response variable and the others variables as the predictors and we can see that all the variables cannot explain the sale. The variable that is the most important to predict High is ShelveLoc, Price, Advertising, Age and CompPrice. If we only include this variable and set best=9 than the error rate is 23 %, which is the lowest error rate. We have also to remember that we have only using set.seed(2) on my training data and this is a problem. Therefore to get a better model we have to run more tree on different training sample.