Assignment 2

Xhesina Hita, Paolo Totaro

2023-10-04

Problem 1: Discriminant Analysis (LDA)

Linear Discriminant Analysis

Considering the same data set as Assignment 1, we now start our analysis by performing LDA on the training data in order to predict the dependent variable *High* (which defines whether the variable Sales is high or low when the value is respectively greater or smaller than 8), using the 7 numerical predictor variables:

```
Call:
lda(High ~ ., data = CarseatsS, subset = train)

Prior probabilities of groups:
       0        1
0.609375 0.390625

Group means:
  CompPrice   Income Advertising Population    Price      Age Education
0  123.7333 65.32308    4.876923   257.2308 122.1128 55.94872  13.90769
1  125.1920 73.37600    8.720000   266.9760 104.5520 48.47200  13.80000

Coefficients of linear discriminants:
                      LD1
CompPrice    0.0545973023
Income       0.0089629655
Advertising  0.1003191627
Population  -0.0003771351
Price       -0.0536983527
Age         -0.0300024169
Education   -0.0168941186
```

The prior probabilities of the two groups for the variable High are 60.9% for the 0 group (not High Sales) and 39.1% for the 1 group (High Sales), using training data.

The LDA analysis provides us the mean values of the numerical predictors according to each group of the dependent variable High (e.g. Advertising mean in group 0 = 4.87, while Advertising mean in group 1 = 8.72, in this case the means of the two groups are substantially different).

Let's plot the values for each observation on the Linear Discriminant Analysis:

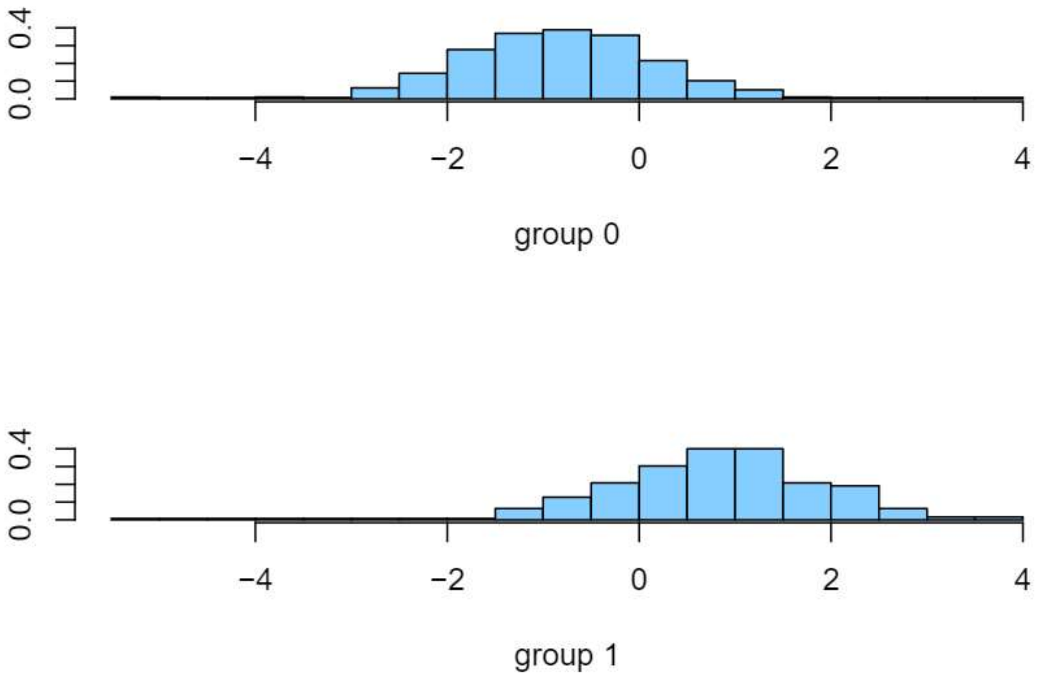

Figure 1: LDA Histograms showing the separation between group 0 (not High sales) and group 1 (High sales)

From the plot in *Figure 1* we notice that the two groups are fairly separated but we see a bit of overlapping of the observations. If we consider for example 0 as cut-off point there are some observations that are in group 1 but have a value of the LDA lower than 0 and we also have observations that have a higher value of LDA but belong to group 0, therefore we expect some error in classification. A high value on the discriminant function indicates a higher probability of High being equal to 1.

We now compute the test error of the model obtained, in order to do so we start by computing the percentage of correct predictions:

```
0.7875
```

In order to get the test error of the model we do 1 - correct classification = 1-0.7875= 0.2125 (21.25%).

	0	1
0	36	5
1	12	27

From the confusion table above (rows = real values, columns = predicted values) we can compute the values of sensitivity and specificity for LDA:

sensitivity	specificity
0.6923077	0.8780488

We get that the probability of correctly classifying Sales as lower than 8 (87.8%) is higher than correctly classifying Sales as higher than 8 (69.2%).

Quadratic Discriminant Analysis (QDA)

We now keep going with our analysis by performing Quadratic Discriminant Analysis:

```
Call:
qda(High ~ ., data = CarseatsS, subset = train)

Prior probabilities of groups:
       0        1
0.609375 0.390625

Group means:
  CompPrice   Income Advertising Population    Price      Age Education
0  123.7333 65.32308    4.876923   257.2308 122.1128 55.94872  13.90769
1  125.1920 73.37600    8.720000   266.9760 104.5520 48.47200  13.80000
```

Just like we have previously seen in the LDA analysis, from the QDA analysis as well we get the mean values of the numerical predictors according to each group of the dependent variable High (e.g. Advertising mean in group 0 = 4.87, while Advertising mean in group 1 = 8.72). Also the prior probabilities of the two groups for the variable High stay the same, they are 60.9% for the 0 (not High Sales) group and 39.1% for the 1 (High Sales) group.

Now we compute the test error of the model obtained as before, we firstly get the proportion of correct predictions which is:

```
0.7375
```

Therefore, just like before, to get the test error of the model we do 1 - correct classification = 1-0.7375= 0.2625 (26.25%)

	0	1
0	34	7
1	14	25

From the confusion table above we can compute the values of sensitivity and specificity for QDA (rows = real values, columns = predicted values):

sensitivity	specificity
0.6410256	0.8292683

With the QDA we obtain a higher classification error than LDA, also the sensitivity and specificity is lower than the LDA.

We now compare the analysis of LDA and QDA with the KNN analysis in Assignment 1. Using KNN we found out that the value of K that would result in the higher proportion of correct classification was K=9, with a test error of 1 - 0.725 = 0.275 (27.5%).

We now compute the values of Sensitivity and Specificity from the following table.

```
   Cell Contents
|-------------------------|
|                   Count |
|             Row Percent |
|-------------------------|

Total Observations in Table:  80

             | nearest9
High[-train] |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |        35 |         6 |        41 |
             |   85.366% |   14.634% |   51.250% |
-------------|-----------|-----------|-----------|
           1 |        16 |        23 |        39 |
             |   41.026% |   58.974% |   48.750% |
-------------|-----------|-----------|-----------|
Column Total |        51 |        29 |        80 |
-------------|-----------|-----------|-----------|
```

We get a Sensitivity of 58.974% and a Specificity of 85.366%

When performing Cross-Validation on training data we got that the highest proportion of correct classification is with K=2, with a test error of 1 - 0.719 = 0.281 (28.1%). We perform then also a KNN with K=2

We now compute the values of Sensitivity and Specificity from the following table.

```
   Cell Contents
|-------------------------|
|                   Count |
|             Row Percent |
|-------------------------|

Total Observations in Table:  80

             | nearest2
High[-train] |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |        33 |         8 |        41 |
             |   80.488% |   19.512% |   51.250% |
-------------|-----------|-----------|-----------|
           1 |        22 |        17 |        39 |
             |   56.410% |   43.590% |   48.750% |
-------------|-----------|-----------|-----------|
Column Total |        55 |        25 |        80 |
-------------|-----------|-----------|-----------|
```

We get a Sensitivity of 43.590% and a specificity of 80.488%.

The values of specificity and sensitivity for K=2 are lower than the ones for K=9, therefore we prefer to perform our analysis with K=9. Moreover, since K=2 is even we may have issues with classification.

We now compare it with the results of Logistic Regression, focusing on cut-point 0.5 (since it has lower test error than a cut-point of 0.3 and 0.8 we analysed in Assignment 1) we got a test error of 1-0.7875=0.2125 (21.25%).

```
   Cell Contents
|-------------------------|
|                   Count |
|             Row Percent |
|-------------------------|

Total Observations in Table:  80

               | predlog
High.f[-train] |        No |       Yes | Row Total |
---------------|-----------|-----------|-----------|
            No |        36 |         5 |        41 |
               |   87.805% |   12.195% |   51.250% |
---------------|-----------|-----------|-----------|
           Yes |        12 |        27 |        39 |
               |   30.769% |   69.231% |   48.750% |
---------------|-----------|-----------|-----------|
  Column Total |        48 |        32 |        80 |
---------------|-----------|-----------|-----------|
```

We get a Sensitivity of 69.231% and a specificity of 87.805%.

Compare the results

In this part we are doing classification of the variable High (1 if the number of unit sales of car-seats is greater than 8 thousands, 0 otherwise) using only numerical variables as predictors.

Table 1:

Model	Test Error	Sensitivity	Specificity
LDA	21.25%	69.23%	87.80%
QDA	26.25%	64.10%	82.92%
KNN	27.50%	58.97%	85.37%
Log Reg	21.25%	69.23%	87.80%

We observe that LDA and Logistic Regression perform better in terms of lower test error and higher Sensitivity and Specificity as we can see from *Table 1.* The logistic regression estimates the conditional probability of High=1 given the predictors. LDA and QDA are alternative approaches to estimate this probability, they use prior probabilities and density functions of the predictors. In both cases we are assigning an observation to the class for which the posterior probability is greatest, in this case the cut-off is 0.5 . This means that the LDA, QDA classifier and Logistic Regression use a threshold of 50% for the posterior probability of High=1 in order to assign an observation to the class Sales>8 thousands. If we are interested in a better classification of the observation that belong to group 1 (Sales>8) then we can consider a lower threshold, (e.g. 0.2 - 20%), viceversa if we want a higher correct classification for observations in group 0 we should increase the threshold (e.g. 0.8 - 80%). Our dependent variable is High, which again refers to the number of Sales of car seats in thousands; since we don't have a preference whether to increase either sensibility or specificity we can use the cut-off that increases both of them.

We investigated all the possible cut-offs between 0 and 1 for LDA and QDA and the best ones for the two analyses are respectively 0.350 and 0.345 .

If we compare LDA and QDA we know that LDA is a less flexible classifier than QDA, and therefore has substantially lower variance, this can lead to improve prediction performance, but if LDA's assumption that the classes share a common covariance matrix is badly off, then LDA can suffer from high bias. In our data LDA performs better than QDA (LDA has lower test error), this is due to the fact that QDA fits a more flexible classifier than necessary. When the true decision boundaries are linear, then LDA and Logistic Regression approaches will tend to perform well, QDA may give better results when the boundaries are moderately non linear.

The KNN is a classification method that uses the known classification of the K nearest point to classify an observation. KNN applies Bayes rule and classifies the test observation to the class with the largest probability. The choice of K has a drastic effect on the KNN classifier, if we choose a K=1 the classifier will have a lower bias but a very high variance. As K increases, the method becomes less flexible and produces a decision boundary that is close to linear, this corresponds to a low-variance but high-bias classifier. Notice that the variance refers to the amount by which the model would change if we estimated it using a different training data. The bias refers to the error that is introduced by approximating a real-life problem, by a much simpler model. Because KNN is completely non-parametric, we can expect this approach to dominate LDA and logistic regression when the decision boundary is highly non-linear, provided that n is very large and the number of predictors is small. In our data LDA and Logistic Regression provide the best classifications, which means that the linear boundary is preferable to a more flexible one.

Problem 2: Classification Trees and Regression Trees

Part 1 - Classification Tree

We now analyse the Carseats data set using Classification Trees including all the variables (not only the numerical ones). We again add the variable High defining whether a Sale is high or low when the value is respectively greater or smaller than 8.

We train the model in order to get a classification tree and we plot the graphical representation of it.

```
Classification tree:
tree(formula = as.factor(High) ~ ., data = CarseatsS, subset = train)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "Income"      "Population"  "Age"
[6] "CompPrice"   "Advertising" "US"
Number of terminal nodes:  29
Residual mean deviance:  0.4073 = 118.5 / 291
Misclassification error rate: 0.09062 = 29 / 320
```

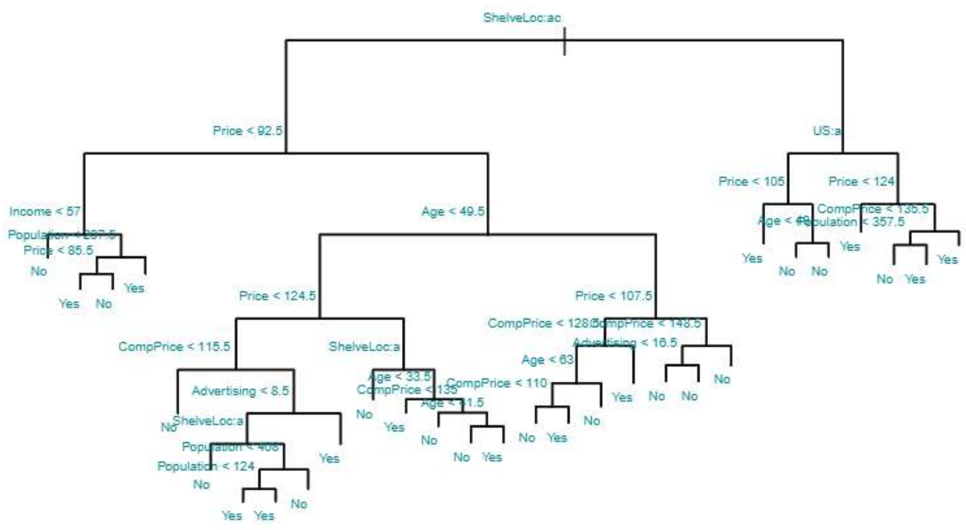

Figure 2: Classification Tree of the variable High

From the plot we observe 29 terminal nodes and the usage of 8 variables whiche are: ShelveLoc, Price, Income, Population, Age, CompPrice, Advertising and US.

We now compute the percentages of correct classification for both the train and test data.

From the summary of the tree we get that the misclassification error rate is equal to 0.09062 and therefore the percentage of correct classification for training data is (1-0.09062)*100 = 90.94%

We use the train model to get the predictions for test data, and then we compute the percentage of correct classification which is 71.25%

We expected to get a higher percentage of correct classification for the training data since the model is actually trained on these data, whereas the test data are unseen data.

Let's now compute sensitivity and specificity using data from the following confusion table (rows = true values, columns = predicted values):

	No	Yes
No	38	3
Yes	20	19

We find a value of the Sensitivity of 48.72% whereas for the Specificity we get 92.68%. Therefore the model correctly classifies unit sales at each location that are lower than 8000, whereas when classifying the unit sales at each location greater than 8000 it doesn't do as good.

We now perform Cross-Validation with k=10.

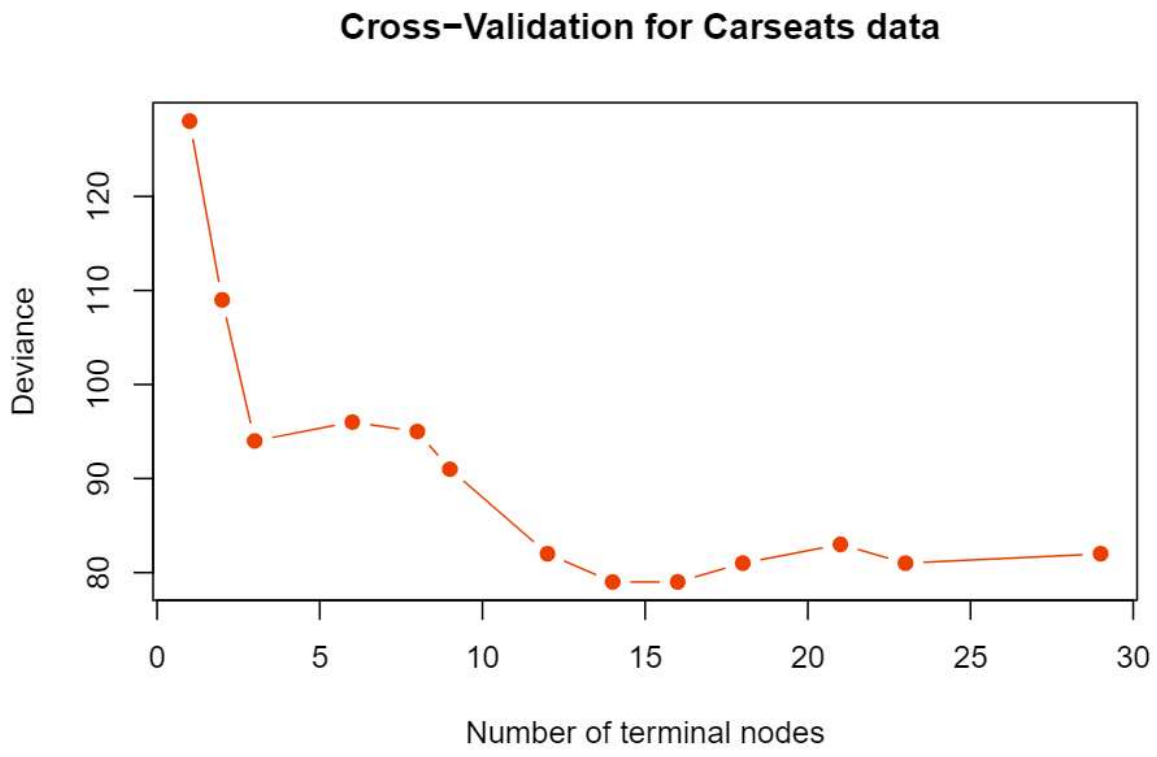

Figure 3: Cross-Validation plot for the Classification tree of Carseats data

Cross-validation provides us the values of the deviances and the respective terminal nodes. From *Figure 3* we notice that two terminal nodes, 14 and 16, have the lowest deviance among all which is 79. In this case we select the lower of the two (14) in order to have a trade off between complexity and goodness of fit.

We now prune the original tree (with 29 terminal nodes) in order to get one with 14 terminal nodes which is the optimal number obtained from Cross Validation.

```
Classification tree:
snip.tree(tree = cstree, nodes = c(13L, 23L, 44L, 330L, 7L, 21L,
9L))
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"      "Income"      "Age"         "CompPrice"
[6] "Advertising" "Population"  "US"
Number of terminal nodes:  14
Residual mean deviance:  0.689 = 210.8 / 306
Misclassification error rate: 0.1156 = 37 / 320
```

From the summary we obtain that the variables used are the same as before, however the percentage of correct classification is a bit lower (1-0.1156)*100 = 88.44%. The decrease in percentage of correct classification is due to the fact that we are working with a fewer number of terminal nodes; a higher number of terminal nodes provided a slight overfit in the previous model, which meant higher accuracy but higher variance as well leading to a poor test set performance. On the other hand a smaller tree with fewer splits might lead to lower variance and better interpretation at the cost of a little bias.

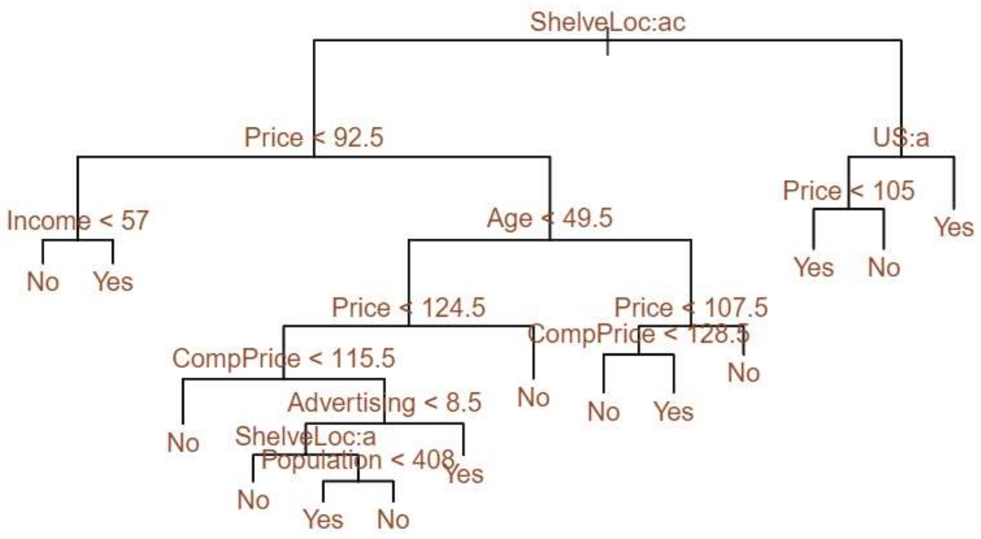

Figure 4: Pruned Classification Tree with 14 terminal nodes

In conclusion, we firsly built a classification tree based on the training data, the three had 29 terminal nodes in 8 variables. We then improved the classification using Cross Validation and this led us to a pruned classification tree with 14 terminal nodes instead, and a slightly lower percentage of correct classification than before. It is better to use a pruned tree to avoid overfitting, moreover the pruned tree is way easier to interpret since it has a reduced number of terminal nodes. Indeed when looking at *Figure 4* it is very easy to classify an observation with Shelveloc equal to Bad or Medium, Price lower than 92.5 and Income lower than 57 as Sales<8. *Note*: in ShelveLoc:ac the "a" stands for Bad and the "c" for medium following in general this classification: a=Bad, b=Good, c=Medium. For US: a=NO and b=YES

Part 2 - Regression Tree

We now perform an analysis of the data using a Regression Tree, therefore we use the original data set Carseats with dependent variable "Sales"

```
Regression tree:
tree(formula = Sales ~ ., data = Carseats, subset = train)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "CompPrice"   "Advertising" "Age"
[6] "Income"
Number of terminal nodes:  16
Residual mean deviance:  2.761 = 839.4 / 304
Distribution of residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-5.22500 -1.07700 -0.05906  0.00000  1.15800  4.09800
```

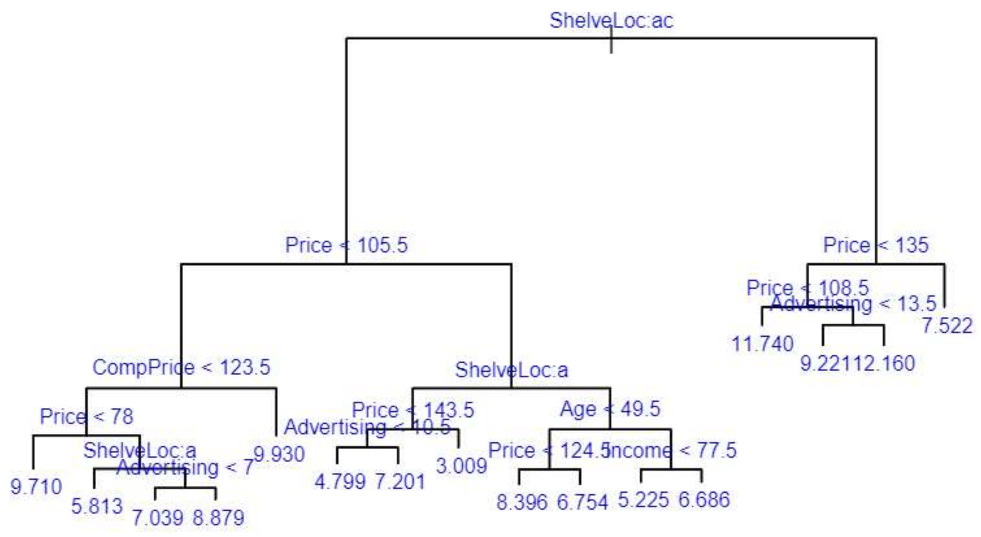

Figure 5: Regression Tree of the variable Sales

We observe that the regression tree presents 16 terminal nodes and 6 variables: ShelveLec, Price, CompPrice, Advertising, Age and Income.

The mean square error in the train set corresponds to the Residual Mean Deviance which is 2.761

Whereas the Mean Square Error in the test set is equal to:

```
4.703219
```

Note: the MSE for the test set is again higher than the MSE for the train set for the same reason as before, which is that the test data are unseen data.

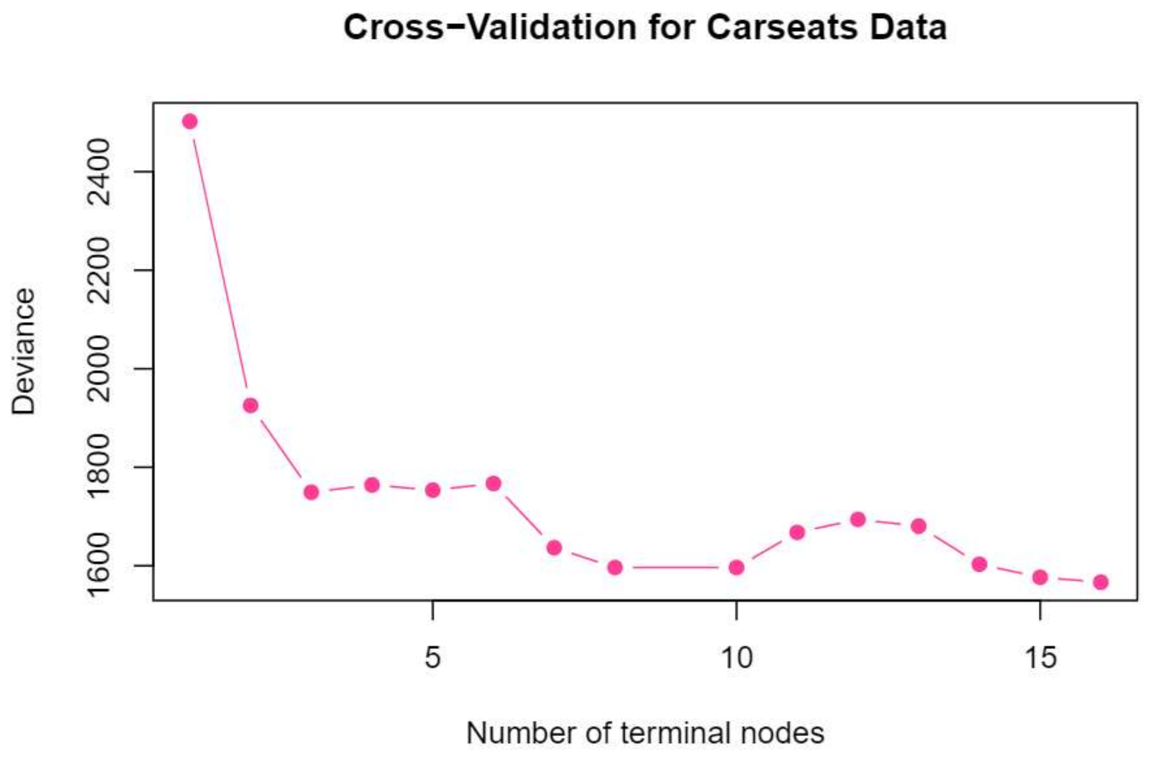

Figure 6: Cross-Validation plot for the Regression Tree of Carseats data

After performing Cross-Validation we observe that the lowest deviance corresponds to 16 terminal nodes, however we get a very low deviance as well at 8 terminal nodes, therefore we prefer to build a 8 terminal nodes regression tree for a better trade off between complexity and goodness of fit.

We now prune the tree using 8 terminal nodes.

```
Regression tree:
snip.tree(tree = tree.car, nodes = c(23L, 22L, 10L, 17L, 6L))
Variables actually used in tree construction:
[1] "ShelveLoc" "Price"     "CompPrice" "Age"
Number of terminal nodes:  8
Residual mean deviance:  3.654 = 1140 / 312
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-5.6890 -1.3050  0.1241  0.0000  1.2030  5.3140
```

From the summary we obtain that the variables used are ShelveLoc, Price, CompPrice and Age, the Residual Mean Deviance is higher than the one of the full tree (16 terminal nodes) and is equal to 3.654. The increase in Residual Mean Deviance is due to the fact that the we are working with a fewer number of terminal nodes, a higher number of terminal nodes provided a slight overfit in the previous model, which meant higher accuracy but higher variance as well, leading to a poor test set performance. On the other hand a smaller tree with fewer splits might lead to lower variance and better interpretation at the cost of a little bias.

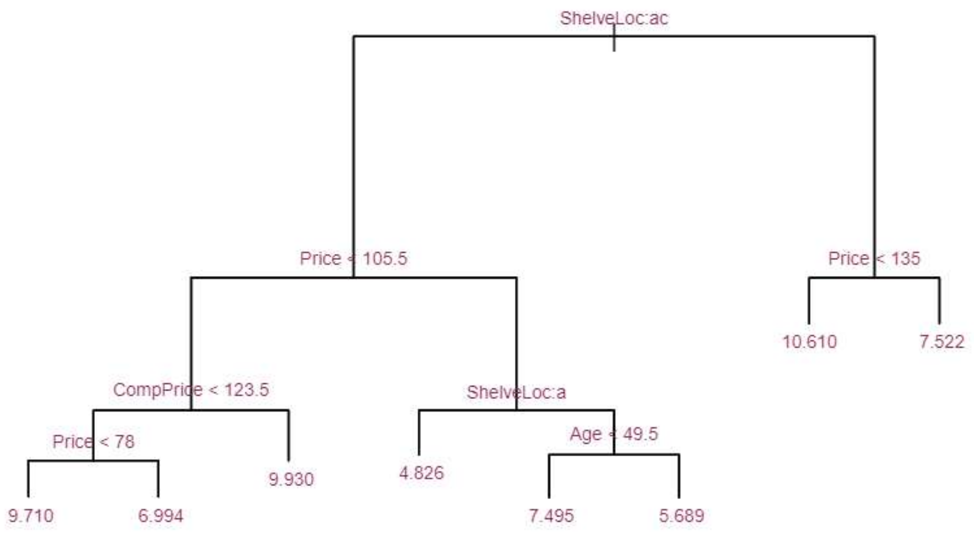

Figure 7: Pruned Regression tree with 8 terminal nodes

In conclusion, we firsly built a regression tree based on the training data, the three had 16 terminal nodes in 6 variables. We then improved the model using Cross Validation and this led us to a pruned regression tree with 8 terminal nodes instead, and a residual mean deviance than before. It is better to use a pruned tree to avoid overfitting, moreover the pruned tree is way easier to interpret since it has a reduced number of terminal nodes. Indeed when looking at *Figure 7* it is very easy to predict the value of Sales for an observation with Shelveloc equal to Bad or Medium, Price lower than 78 and CompPrice lower than 123.5 to be around 9.710 .

Part 3 Comparison

When we do a comparison we need to distinguish between classification and regression methods. In a regression method we assume that the response variable Y is quantitative; whereas f we want to predict qualitative responses, we have to perform classification. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category.

When performing multiple linear regression we only took into account the numerical variables and we observed which one of them had a significant effect on the dependent variable Sales. This model is a good choice when we are interested in analysing the relation between the predictors and the dependent variable, indeed when we increase by a unit the value of a predictor, controlling for all the others, we will capture how much the dependent variable changes on average.

Another kind of regression is performed by Regression Trees which predict the values of the observations according to the region they fall into. However, differently from Multiple Linear Regression we do not have a relation between the dependent variable and the predicotrs. Indeed, regression tree provides a good model for prediction but not for inference.

For what concerns classification, we perform a Logistic Regression which is both a good model for inference and classification. For inference, we know that if we increase by one unit the value of a predictor, controlling for all the others, we will either get an increase or a decrease in the probability of Sales being greater than 8 thousands depending on the sign of the coefficients beta. Note that together with LDA it provided the best classification among all methods.

For classification we also used KNN and Classification Trees, note that in KNN we didn't use all the variables, only the numerical ones, whereas in the classification trees we used all of them. Classification trees are very easy to visualise and interpret, therefore if this is our goal they can be a good model. Moreover, it also captures non linear relations between the predictors and the dependent variable. Another advantage is that we are able to spot the important variables which can be found at the beginning (top) of the tree. However the classification tree can fall into overfitting, and high test error as a consequence. The KNN model is quite easy to understand as well, however it has higher computational complexity (distance computation and standardization).

If we compare LDA and QDA we know that LDA is a less flexible classifier than QDA, and therefore has substantially lower variance, this can lead to improve prediction performance, but if LDA's assumption that the classes share a common covariance matrix is badly off, then LDA can suffer from high bias. In our data LDA performs better than QDA (LDA has lower test error), this is due to the fact that QDA fits a more flexible classifier than necessary.

Table 2:

Model	Test Error	Sensitivity	Specificity
LDA	21.25%	69.23%	87.80%
QDA	26.25%	64.10%	82.92%
KNN	27.50%	58.97%	85.37%
Log Reg	21.25%	69.23%	87.80%
Class Tree	28.75%	48.72%	92.68%

In conclusion, depending on whether our ultimate goal is prediction or inference (interested in understanding the way that the dependent variable is affected by the predictors), different methods may be appropriate. If our aim is to get the best prediction we will either choose LDA or Logistic Regression since the test error is low and the values for Sensitivity and Specificity are high. Otherwise, if we are interested in the relation between the predictors and the dependent variables we can either perform a Logistic Regression or a Multiple Linear Regression. If the goal is to get an interpretable and intuitive graphical representation we can choose Classification Trees.