Lecture 10: Data Mining Techniques: Classification Trees + ‘rpart’ package

If we have a binary response variable and some covariates, we can build a model connecting the binary response variable with the covariates using logistic regression. If we have a ternary response variable and some covariates, we can build a multinomial logistic regression model to summarize the data. The logistic regression model is probabilistic in nature. There are a number of other approaches. One approach popular with engineers and physicists is to treat the problem as a pattern recognition or classification problem. Let us look at the abdominal sepsis problem.

Response variable

Y = 1 if the patient dies after surgery

= 0 if the patient survives after surgery

Independent variables

X1: Is the patient in a state of shock?

X2: Is the patient suffering from undernourishment?

X3: Is the patient alcoholic?

X4: Age

X5: Did the patient have bowel infarction sometime in the past?

In logistic regression, the probability distribution of Y is modeled in terms of the covariates.

If we view this problem as a pattern recognition problem, we need to identify what the patterns are. The situation Y = 1 (Death after surgery) is regarded as one pattern and Y = 0 (survival after surgery) as the other. Once we have information on the independent variables, predictors, or covariates for a patient, we need to classify him/her into one of the two patterns. We have to come up with a protocol, which will classify the patient as falling into one of the patterns. In other words, we have to say whether he will die or survive after surgery. We will not make a probability statement. We will worry about its accuracy later.

Look at the crocodile problem. What is inside its belly? Fish? Invertebrates? Other? The response Y is ternary. Y = Fish is one pattern. Y = Invertebrates is another pattern. Y = Other is yet another pattern. What information do we have on the crocodile: its size? Look at its size. Classify the crocodile into one of the patterns directly. No probability

Any classification protocol one comes up cannot be expected to be free of errors. A classification protocol is judged based on its misclassification error rate. We will make precise this concept later.

Core idea: Look at the space of predictors in the context of the sepsis problem. We want to break up the predictor space into boxes (5-dimensional parallelepipeds) so that each box is identified with one pattern. For example, Shock = 1, Malnourishment = 0, Alcoholism = 1, Age > 45, Infarction = 1 is one such box. Can we say that most of the patients that fall into this box die? We want to divide the predictor space into mutually exclusive and exhaustive boxes so that the patients falling into each box have predominantly one pattern, either death or survival. Mine the prediction space. Identify boxes so that patients falling into each box have one pattern predominantly. The creation of such boxes is the main objective of this lecture. More simplistically, we should be able to make statements like:

Aha! If X1 ≤ 0.5, X2 > 0.5, X3 < 0.5, X4 > 80, and X5 > 0.5, most patients die!

Let us look at a concrete example. Activate the package ‘rpart.’ Download the data ‘kyphosis’ from the package.

> data(kyphosis)

> dim(kyphosis)

[1] 81 4

> head(kyphosis)

Kyphosis Age Number Start

1 absent 71 3 5

2 absent 158 3 14

3 present 128 4 5

4 absent 2 5 1

5 absent 1 4 15

6 absent 1 2 16

What is kyphosis? A misalignment of vertebrae.

Solution: Surgery to align the problematic vertebrae

Result: Is the surgery successful? If misalignment disappears, we say that kyphosis is absent. Otherwise, kyphosis is present.

Start: Number of misaligned vertebra. Surgery is done on only the topmost misaligned vertebrae.

Age: age in months

Number: The number of the topmost vertebrae surgery was done

Kyphosis: outcome of surgery

Let us explore some parallelpipeds:

> PipedOne <- subset(kyphosis, Start >= 14.5)

> dim(PipedOne)

[1] 29 4

> PipedOne

Kyphosis Age Number Start

5 absent 1 4 15

6 absent 1 2 16

7 absent 61 2 17

8 absent 37 3 16

9 absent 113 2 16

12 absent 148 3 16

15 absent 168 3 18

16 absent 1 3 16

17 absent 78 6 15

19 absent 80 5 16

21 absent 22 2 16

29 absent 4 3 16

30 absent 151 2 16

31 absent 31 3 16

34 absent 112 3 16

36 absent 93 3 16

45 absent 97 3 16

47 absent 136 4 15

52 absent 9 2 17

54 absent 2 2 17

55 absent 140 4 15

56 absent 72 5 15

64 absent 118 3 16

65 absent 118 4 16

67 absent 195 2 17

70 absent 15 5 16

73 absent 87 4 16

75 absent 11 3 15

76 absent 178 4 15

Look at all the children with Start ≥ 14.5. This is tantamount Start ≥ 15. Look at the outcome of surgery. All these children in this parallelepiped had the same outcome: kyphosis was absent.

Eureka! I found a parallelepiped in the prediction space with only one pattern. I can come up with a recommendation. If the vertebrae number operated on is 15 or more, kyphosis will be absent. How confident I am? 100%.

Look at another parallelepiped!

> PipedTwo <- subset(kyphosis, Start >= 8.5 & Start <= 14.5 & Age >= 111)

> dim(PipedTwo)

[1] 14 4

> PipedTwo

Kyphosis Age Number Start

2 absent 158 3 14

18 absent 175 5 13

32 absent 125 2 11

33 absent 130 5 13

35 absent 140 5 11

46 present 139 3 10

48 absent 131 5 13

50 absent 177 2 14

68 absent 159 4 13

71 absent 158 5 14

72 absent 127 4 12

74 absent 206 4 10

77 present 157 3 13

79 absent 120 2 13

Another discovery! All the children falling into this parallelepiped had only one predominant pattern: kyphosis was absent. Another eureka moment! I come up with my recommendation. If the vertebrae number operated on is between 9 and 14 both inclusive, my prediction is kyphosis absent. How confident I am? 12/14 = 85.7%.

Another comment: These two parallelepipeds are disjoint.

How to find such parallelepipeds?

One popular method in classification or pattern recognition is the so called the ‘classification tree methodology,’ which is a data mining method. The methodology was first proposed by Breiman, Friedman, Olshen, and Stone in their monograph published in 1984. This goes by the acronym CART (Classification and Regression Trees). A commercial program called CART can be purchased from Salford Systems. Other more standard statistical software such as SPLUS, SPSS, SAS, MATLAB, and R also provide tree construction procedures with user-friendly graphical interface. The packages ‘rpart’ and ‘tree’ do classification trees. Some of the material I am presenting in this lecture is culled from the following book.

L Breiman, J H Friedman, R A Olshen, and C J Stone – Classification and Regression Trees, Wadsworth International Group, 1984.

Basic ideas in the development of a classification tree

Let me work with an artificial example: One binary response variable Y and two predictors.

ID Y X1 X2

1 0 1 2

2 1 6 5

3 1 5 7

4 0 10 9

5 0 5 5

6 1 4 8

7 1 10 2

8 0 4 3

9 1 8 4

10 0 9 7

11 1 3 9

12 0 8 8

13 1 9 2

14 0 3 1

15 0 7 7

16 1 2 10

17 0 6 10

18 1 7 5

19 1 1 6

20 0 2 4

Goal: I know one with given X1 and X2 values. I need to classify him as having the pattern Y = 0 or Y = 1. We have the training data given above to develop a classification protocol. (I could have done a logistic regression here.)

Another view point: What ranges of X1 and X2 values identify the pattern {Y = 0} mostly and what for the pattern {Y = 1} mostly?

I am going to build a tree with my bare hands.

Step 1: Put all the subjects into the root node. There are ten subjects with the pattern Y = 0 and ten with Y = 1.

Step 2: Let us split the mother node into two daughter nodes. We need to choose one of the covariates. Let us choose X1. We need to choose one of the numbers taken by X1. The possible values of X1 are 1, 2, … , 10. Let us choose 5. All those subjects with X1 ≤ 5 go into the left daughter node. All those subjects with X1 > 5 go into the right daughter node.

Members of the left daughter node: ID 1, 3, 5, 6, 8, 11, 14, 16, 19, 20. Five of these subjects have the pattern {Y = 0} and the rest {Y = 1}.

Members of the right daughter node: ID 2, 4, 7, 9, 10, 12, 13, 15, 17, 18. Five of these subjects have the pattern {Y = 0} and the rest {Y = 1}.

Step 3. Let us split the left daughter node. Choose one of the covariates. Let us choose now X2. Let us choose one of the numbers taken by X2. Let us choose 5. Shepherd all those subjects with X2 ≤ 5 into the left grand daughter node and those with X2 > 5 into the right grand daughter node.

Composition of the subjects in the left granddaughter node: ID 1, 5, 8, 14, 20. All these subjects have the pattern {Y = 0}. This granddaughter is the purest. There is no need to split the granddaughter. This is a terminal node. Declare this node as {Y = 0} node.

Composition of the subjects in the right granddaughter node: ID 3, 6, 11, 16, 19. All these subjects have the pattern {Y = 1}. This granddaughter is the purest. There is no need to split the granddaughter. This is a terminal node. Declare this node as {Y = 1} node.

Step 4. Let us split the right daughter node. Choose one of the covariates. Let us choose X2. Let us choose one of the numbers taken by X2. Let us choose 5. Shepherd all those subjects with X2 ≤ 5 into the left grand daughter node and those with X2 > 5 into the right grand daughter node.

Composition of the subjects in the left granddaughter node: ID 2, 7, 9, 13, 18. All these subjects have the pattern {Y = 1}. This granddaughter is the purest. This is a terminal node. Declare this node as {Y = 1} node.

Composition of the subjects in the right grand daughter node: ID 4, 10, 12, 15, 17. All these subjects have the pattern {Y = 0}. This granddaughter is the purest. This is a terminal node. Declare this node as {Y = 0} node.

The task of building a tree is complete. Look at the tree that results.

Let us now calculate the misclassification error rate. Let us pour all the subjects into the mother node. We know the pattern each subject has. Check which terminal node they fall into. Check whether its true pattern matches with the pattern of the terminal node. The percentage of mismatches is the misclassification rate.

Misclassification rate = 0%.

How does one use this classification protocol in practice? Take a subject whose pattern is unknown. We have its covariate values. Pour this subject into the mother node. See where he lands. Note the identity of the terminal node. That is the pattern he is classified into.

I built the tree with my bare hands. This tree can also be drawn in a different way. We use the ‘polygon’ command of R. First, present a verbal description of the tree I built.

If X1 ≤ 5 and X2 ≤ 5, classify the subject to have the pattern {Y = 0}.

If X1 ≤ 5 and X2 ≥ 6, classify the subject to have the pattern {Y = 1}.

If X1 ≥ 6 and X2 ≤ 5, classify the subject to have the pattern {Y = 1}.

If X1 ≥ 5 and X2 ≥ 6, classify the subject to have the pattern {Y = 0}.

The statement X1 ≤ 5 and X2 ≤ 5 is equivalent to, graphically, the rectangle with vertices (1, 1), (1, 5), (5, 5), (5, 1) in the X1 – X2 plane. The command ‘polygon’ draws the rectangle. First, we need to create a blank plot setting the X1- and X2-axes. The input type = “n” exhorts the plot that there should be no points imprinted on the graph.

> plot(c(1,10), c(1, 10), type = "n", xlab = "X1", ylab = "X2", main = "Classification Protocol")

The ‘polygon’ command has essentially two major inputs. The x-input should have all the x coordinates of the points. The y-input should have all the corresponding y-coordinates of the points. The polygon thus created latches onto the existing plot.

> polygon(c(1, 1, 5, 5), c(1, 5, 5, 1), col = "gray", border = "blue", lwd = 2)

The statement X1 ≤ 5 and X2 ≥ 6 is equivalent to, graphically, the rectangle with vertices (1, 6), (1,10), (5, 10), (5, 6) in the X1 – X2 plane.

> polygon(c(1, 1, 5, 5), c(6, 10, 10,6), col = "yellow", border = "blue", lwd = 2)

The other polygons are created in the same way.

> polygon(c(6, 6, 10, 10), c(6, 10, 10,6), col = "mistyrose", border = "blue", lwd = 2)

> polygon(c(6, 6, 10, 10), c(1, 5, 5, 1), col = "cyan", border = "blue", lwd = 2)

We need to identify each rectangle with a pattern. The text command needs the coordinates (x-coordinate and y-coordinate) at which the legend is to be implanted. The coordinates are to be followed by the legend in double quotes. The color is optional. The default is ‘black.’

> text(3, 3, "{Y = 0}", col = "red")

> text(3, 8, "{Y = 1}", col = "blue")

> text(8, 8, "{Y = 0}", col = "red")

> text(8, 3, "{Y = 1}", col = "blue")



Questions in general:

1. Which variable one should choose to split a node?
2. Once the variable is chosen, what cut-point is to be chosen to create daughter nodes?

We use entropy to guide our choices.

Suppose we have a random variable X taking finitely many values with some probability distribution.

X: 1 2 … m

Pr.: p1 p2 … pm

We want to measure the degree of uncertainty in the distribution (p1, p2, … , pm). For example, suppose m = 2. Look at the distributions (1/2, 1/2) and (0.99, 0.01). There is more uncertainty in the first distribution than in the second. Suppose someone is about to crank out or simulate X. I am more comfortable in betting on the outcome of X if the underlying distribution is (0.99, 0.01) than when the distribution is (1/2,1/2). We want to assign a numerical quantity to measure the degree of uncertainty. Entropy of a distribution is introduced as a measure of uncertainty.

Entropy (p1, p2, … , pm) =  = Entropy impurity = Measure of Chaos, with the convention that 0\*ln 0 = 0.

Properties

1. 0 ≤ Entropy ≤ ln m.
2. The minimum 0 is attained for each of the distributions (1, 0, 0, … , 0), (0, 1, 0, … , 0), … , (0, 0, … , 0, 1). For each of these distributions, there is no uncertainty. The entropy is zero.
3. The maximum ln m is attained at the distribution (1/m, 1/m, … , 1/m). The uniform distribution is the most chaotic. Under this uniform distribution, uncertainty is maximum.

There are other measures of uncertainty available in the literature.

Gini’s measure of uncertainty for the distribution (p1, p2, … , pm) = .

The package ‘rpart’ uses Gini.

Properties

1. 0 ≤ Gini’s measure ≤ (m-1)/m.
2. The minimum 0 is attained for each of the distributions (1, 0, 0, … , 0), (0, 1, 0, … , 0), … , (0, 0, … , 0, 1). For each of these distributions, there is no uncertainty. The Gini’s measure is zero.
3. The maximum (m-1)/m is attained at the most chaotic distribution (1/m, 1/m, … , 1/m). Under this uniform distribution, the uncertainty is maximum.

Another measure of uncertainty is defined by min {p1, p2, … , pm}.

How entropy is used to pick up a covariate for splitting the mother node? Discuss.

Step 1: Put all the data into the root node. Count the frequency of each pattern. Calculate its entropy E. Create two daughter nodes.

Step 2: Choose a predictor X, say, and a cut point c, say. Shepherd all subjects with X ≤ c into the left-daughter node. Count the frequency of each pattern in this node. Calculate its entropy E1.

Step 3: Shepherd all subjects with X > c into the right-daughter node. Count the frequency of each pattern in this node. Calculate its entropy E2.

Step 4: Calculate combined entropy of the daughters: W1\*E1 + W2\*E2, with weights W1 and W2 proportional to the sizes of the daughter nodes. The smaller the combined entropy is, the less chaotic the nodes are. The smaller the combined entropy is, the better the split is. Why?

Step 5: Choose that predictor and that cut point for which the combined entropy is the least.

Step 6: Now focus on the left-daughter node. Split it …

Computations

Terry Therneau and Elizabeth Atkinson (Mayo Foundation) have developed ‘rpart’ (recursive partitioning) package to implement classification trees and regression trees in all their glory. The method depends what kind of response variable we have.

Categorical → method = “class”

Continuous → method = “anova”

Count → method = “poisson”

Survival → method = “exp”

They have two monographs on their package available on the internet.

An introduction to Recursive Partitioning using the RPART routines, February, 2000

Same title, September, 1997

Both are very informative.

Let me illustrate ‘rpart’ command in the context of a binary classification problem. Four data sets are available in the package.

Activate the packages ‘rpart’ and ‘rpart.plot.’ What are the data sets available in the package ‘rpart?’

* data(package = “rpart”)

Data sets in the package ‘rpart’:

car.test.frame Automobile Data from 'Consumer Reports' 1990

cu.summary Automobile Data from ‘Consumer Reports' 1990

kyphosis Data on Children who have had Corrective Spinal

Surgery

solder Soldering of Components on Printed-Circuit Boards

We have already looked at the ‘kyphosis’ data.

> summary(kyphosis)

Kyphosis Age Number Start

absent :64 Min. : 1.00 Min. : 2.000 Min. : 1.00

present:17 1st Qu.: 26.00 1st Qu.: 3.000 1st Qu.: 9.00

Median : 87.00 Median : 4.000 Median :13.00

Mean : 83.65 Mean : 4.049 Mean :11.49

3rd Qu.:130.00 3rd Qu.: 5.000 3rd Qu.:16.00

Max. :206.00 Max. :10.000 Max. :18.00

Look at the documentation on ‘rpart.’

Pruning:

If we let the partition continue without any break, we will end up with a saturated tree. Every terminal node is pure. It is quite possible some terminal nodes contain only one data point. We need to arrest the growth of the tree. One possibility is to demand that if a node contains 20 observations or less no more splitting is to be done at this node. This is the default setting in ‘rpart.’ One has to declare each **terminal node** as one of the two types: present or absent. Majority decides the designation.

Let us check.

MB <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)

The tree is described by the summary statement.

> summary(MB)

Call:

rpart(formula = Kyphosis ~ Age + Number + Start, data = kyphosis)

n= 81

CP nsplit rel error xerror xstd

1 0.17647059 0 1.0000000 1.000000 0.2155872

2 0.01960784 1 0.8235294 1.058824 0.2200975

3 0.01000000 4 0.7647059 1.058824 0.2200975

Variable importance

Start Age Number

64 24 12

Node number 1: 81 observations, complexity param=0.1764706

predicted class=absent expected loss=0.2098765 P(node) =1

class counts: 64 17

probabilities: 0.790 0.210

left son=2 (62 obs) right son=3 (19 obs)

Primary splits:

Start < 8.5 to the right, improve=6.762330, (0 missing)

Number < 5.5 to the left, improve=2.866795, (0 missing)

Age < 39.5 to the left, improve=2.250212, (0 missing)

Surrogate splits:

Number < 6.5 to the left, agree=0.802, adj=0.158, (0 split)

Node number 2: 62 observations, complexity param=0.01960784

predicted class=absent expected loss=0.09677419 P(node) =0.7654321

class counts: 56 6

probabilities: 0.903 0.097

left son=4 (29 obs) right son=5 (33 obs)

Primary splits:

Start < 14.5 to the right, improve=1.0205280, (0 missing)

Age < 55 to the left, improve=0.6848635, (0 missing)

Number < 4.5 to the left, improve=0.2975332, (0 missing)

Surrogate splits:

Number < 3.5 to the left, agree=0.645, adj=0.241, (0 split)

Age < 16 to the left, agree=0.597, adj=0.138, (0 split)

Node number 3: 19 observations

predicted class=present expected loss=0.4210526 P(node) =0.2345679

class counts: 8 11

probabilities: 0.421 0.579

Node number 4: 29 observations

predicted class=absent expected loss=0 P(node) =0.3580247

class counts: 29 0

probabilities: 1.000 0.000

Node number 5: 33 observations, complexity param=0.01960784

predicted class=absent expected loss=0.1818182 P(node) =0.4074074

class counts: 27 6

probabilities: 0.818 0.182

left son=10 (12 obs) right son=11 (21 obs)

Primary splits:

Age < 55 to the left, improve=1.2467530, (0 missing)

Start < 12.5 to the right, improve=0.2887701, (0 missing)

Number < 3.5 to the right, improve=0.1753247, (0 missing)

Surrogate splits:

Start < 9.5 to the left, agree=0.758, adj=0.333, (0 split)

Number < 5.5 to the right, agree=0.697, adj=0.167, (0 split)

Node number 10: 12 observations

predicted class=absent expected loss=0 P(node) =0.1481481

class counts: 12 0

probabilities: 1.000 0.000

Node number 11: 21 observations, complexity param=0.01960784

predicted class=absent expected loss=0.2857143 P(node) =0.2592593

class counts: 15 6

probabilities: 0.714 0.286

left son=22 (14 obs) right son=23 (7 obs)

Primary splits:

Age < 111 to the right, improve=1.71428600, (0 missing)

Start < 12.5 to the right, improve=0.79365080, (0 missing)

Number < 3.5 to the right, improve=0.07142857, (0 missing)

Node number 22: 14 observations

predicted class=absent expected loss=0.1428571 P(node) =0.1728395

class counts: 12 2

probabilities: 0.857 0.143

Node number 23: 7 observations

predicted class=present expected loss=0.4285714 P(node) =0.08641975

class counts: 3 4

probabilities: 0.429 0.571

Let us look at the tree that results.

> rpart.plot(MB, type = 4, extra = 1, digits = 3, col = “red”)



Description of the tree:

1. The root node has 81 subjects, for 64 of them kyphosis is absent and 17 present.
2. All those subjects with Start ≥ 8.5 go into the left node. Total number of subjects in the left node is 62, 56 of them have kyphosis absent.
3. All those subjects with Start < 8.5 go into the right node. Total number of subjects in the right node is 19, 8 of them have kyphosis absent.
4. This node is a terminal node. No further split is envisaged because the total number of observations is 19 ≤ 20. The command stops splitting a node if the size of the node is 20 or less (default). This is a pruning strategy. This node is declared ‘present’ as per the ‘majority rule’ paradigm.
5. The node on the left is split again. The best covariate as per the entropy purity calculations is ‘Start’ again. All those subjects with Start ≥ 14.5 go into left node. This node is pure. No split is possible. This node has 29 subjects for all of whom kyphosis is absent. Obviously, we declare this terminal node as ‘absent.’ All those subjects with Start < 14.5 go into the right node, which 33 subjects. And so on.
6. Other terminal nodes are self-explanatory.

The classification protocol as per this tree is given by:

1. If a child has Start < 8.5, predict that kyphosis will be present.
2. If a child has 14.5 ≤ Start, predict that kyphosis will be absent.
3. If a child has 8.5 ≤ Start < 14.5 and Age < 55 months, predict that kyphosis will be absent.
4. If a child has 8.5 ≤ Start < 14.5 and Age ≥ 111 months, predict that kyphosis will be absent.
5. If a child has 8.5 ≤ Start < 14.5 and 55 ≤ Age < 111 months, predict that kyphosis will be present.
6. The covariate ‘Number’ has no role in the classification.

Draw a diagram.

How reliable the judgment of this tree is?

We have 81 children in our study. We know for each child whether kyphosis is present or absent. Pour the data on the covariates of a child into the root node. See which terminal node the child settles in. Classify the child accordingly. We know the true status of the child. Note down whether or not a mismatch occurred. Find the total number of mismatches.

Misclassification rate = re-substitution error

= 100\*(8 + 0 + 2 + 3)/81 = 16%.

We can increase the size of the tree by reducing the threshold number 20. Let us do it. If the size of the node is 10 or less, don’t split it. The following is the R command.

> MB1 <- rpart(Kyphosis ~ ., data = kyphosis, control = rpart.control(minsplit = 10))



> rpart.plot(MB1, type = 4, extra = 1)

Interpretation?

Misclassification rate = (0 + 0 + 2 + 1 + 0 + 2 + 2)/81 = 7/81 = 8.6%