3

Decision trees

3.1 Introduction

The decision tree is the "fundamental analytic tool for decision analysis" (Weinstein and Fineberg, 1980). It provides a basic graphical representation of the temporal and logical sequence of events under each alternative strategy being assessed in a decision problem (Kuntz et al., 2016). In the context of health decision sciences, the decision tree is a commonly used tool to estimate the expected value of the outcomes of interest for alternative strategies, such as life years, quality-adjusted life years (QALYs), and costs (McNeil and Pauker, 1984). A strategy in the context of a decision tree constitutes a feasible action that could be taken by the decision maker. Examples of decision trees are ample in the literature. An early decision tree-based model in 1979 outlined the costs, risks and benefits of pertussis vaccination among children in the US (Koplan et al., 1979). Similarly, Tilson et. al used a tree-based model to estimate the average lifetime cost of care for colorectal cancer in Ireland (Tilson et al., 2012). More recently, Gregory et al. used a decision tree as part of a larger decision model to evaluate the cost-effectiveness of diagnostic protocols and clinical decision rules in the diagnosis of appendicitis (Gregory et al., 2016).

The objectives of this chapter are: 1) to offer a structured approach to constructing and solving a decision tree within a statistical programming environment, 2) to describe how existing methods for the calculation of expected value in a decision tree could be implemented in R and 3) to introduce tools in R that facilitate visualization and design of decision trees.

The structure of the chapter is as follows: We first introduce mathematical notation to formalize the structure of a decision tree. Subsequently, we describe two different approaches to solve a decision tree and describe the necessary steps to implement these approaches in R. Following that, we introduce OpenTree and DecTree, two solutions in visualizing and solving decision trees in R. We then illustrate the use of R and these tools in constructing decision trees using a previously published model-based economic evaluation. Finally, we discuss the advantages and disadvantages of the two approaches as well as the strengths and limitations of building decision tree models in R.

3.2 Methods

Brief introduction to decision trees

Decision trees can be represented through a mathematical model that incorporates three structural components of a decision making process: 1) the alternative strategies that are considered for a particular decision; 2) the events that follow from application of any of these strategies and their likelihood; and 3) the outcomes (for an individual, a cohort or a population) that are associated with each event, or series of events. More specifically, a decision tree consists of three different types of *nodes* connected via *branches*. A decision tree starts with a *decision node* (Figure 3.1). The decision node represents a decision point and is used to represent the choices a decision maker has between mutually exclusive strategies. A *chance node* represents uncertain events. This node is associated with two or more mutually exclusive possible events and the probabilities that those events occur. Each branch that originates from a given chance node represents the occurence of a specific event with a specific probability of occurence. The probabilities of all possible events for a given chance node must sum to one. The third type of node is a *terminal node*. This node represents a *terminal point* in the decision algorithm where the outcome(s) of interest are assessed. Conventionally, decision nodes are represented as a square, chance nodes as a circle and terminal nodes as a triangle. Each particular sequence of events starting from a decision node and ending at a terminal node is referred to as a *path*. The *height* of the tree is defined by the longest path from any strategy to a terminal node (Heineman et al., 2016). Every path is associated with one or more outcomes relevant to the decision, denoted as numerical values at the terminal nodes. In the context of economic evaluation, these are often measures of cost and effectiveness of each path.

In this section, we introduce mathematical notation related to the components of a decision tree in order to facilitate translating expected value calculations into an algorithmic form and, by extension, into R code. For a given strategy, let G be the number of paths and, for a given path $g = 1 \ldots G$, let L_g be the number of levels. Further let H be the height of the decision tree or the longest path in the tree such that $H = \max_g L_g$. In addition, let $p_{[l,g]}$ be the probability of an event at a level l within path g occurring, with $l = 1 \ldots L_g$. Finally, let $\mathbf{y}$ be the vector of size G capturing the terminal values of every path.

Figure 3.1 presents a simple decision tree and depicts graphically the notation structure described above.

Estimating the expected value using a decision tree

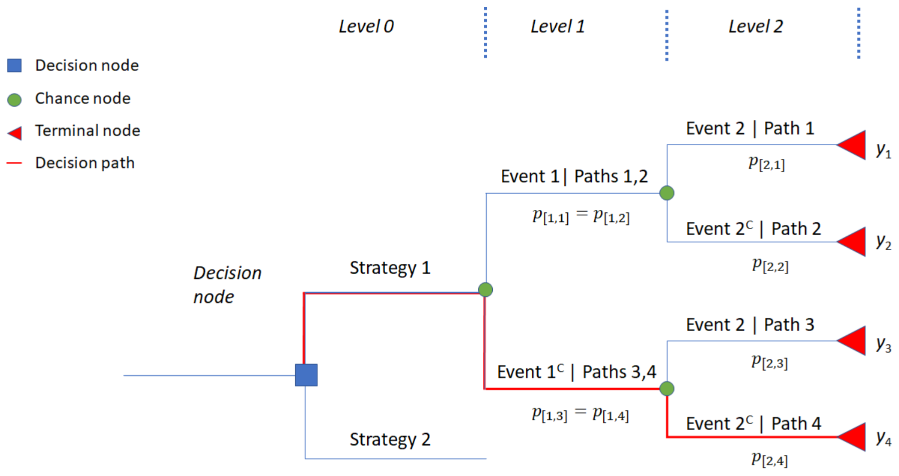

FIGURE 3.1: The structure for Strategy 1 as part of a decision tree for two alternative hypothetical strategies. The structure for Strategy 2 (not shown) can be identical to the structure of Strategy 1.

The expected value of the outcomes of a strategy in a decision tree can be calculated either through closed-form mathematical expressions or through algorithmic, iterative processes (e.g. the foldback algorithm). Below we describe some of such solutions in a mathematical/algorithmic form and provide R code and functionality.

A closed-form solution

Conceptually, the expected outcome $EV(y)$ of a strategy can be calculated by summing, across all G paths, the product of all conditional probabilities in the path $\prod_{l=1}^{L_g} p_{[l,g]}$ and the path's outcome y_g, formally

$$EV(y) = \sum_{g=1}^{G} \prod_{l=1}^{L_g} p_{[l,g]} y_g$$

All probabilities in a decision tree are conditional (on arriving at a particular node), this is why we build decision trees to represent the complexity. Regardless of the size or complexity of the decision tree, the mathematical expression above would suffice. However, applying this closed-form solution in R implies an iterative process. This means that for each decision path, the probabilities for all the levels of the tree are multiplied with each other as well as with the outcome associated with the path, followed by a summation across the paths. The algorithmic solution to the problem would be:

```
for $g = 1$ to $G$ do
    for $l = 1$ to $L_g$ do
```

```
        $w_g = p_{[l,g]} * w_g$
    end
    $EV_g(y) = w_g * y_g$
    $EV(y) = EV(y) + EV_g(y)$
end
```

where g is the vector of size G capturing the probability of experiencing the g^{th} path.

Box 1: A generic R implementation of the iterative decision tree algorithm. Variable names are defined in Table 1.

```r
# G is the number of paths
# l.p is a list containing G vectors,
# each of size Lg capturing the probabilities across Lg levels
# for the gth path
# vector of probabilities for the 1st path
v.p1 <- vector(length = L[1])
# vector of probabilities for the 2nd path
v.p2 <- vector(length = L[2])
...
# vector of probabilities for the 3rd path
v.pG <- vector(length = L[G])

# list of probabilities for each path
l.p <- list(v.p1, v.p2,...v.pG)
# v.w is the vector of size G capturing the probability of
# experiencing each of the G paths
v.w <- vector(length = G)
 for(g in 1:G){
    v.w[g] <- prod(l.p[[g]])
 }
# multiplying the probability of each path with its outcome
# and summing across paths
ev <- sum(v.w * v.y)
```

Algorithm solutions

There have been a number of algorithmic approaches to the calculation of the decision tree solution. The algorithms used are providing a more efficient solution to the decision tree problem by reducing the number of required operations. A widely used algorithmic method to calculate the expected outcome of a decision tree is the “folding back” or “roll-back” method. (Sarin and

Wakker, 1994) When a decision tree is “folded back”, the expected outcome for the strategy is estimated starting from the end of the tree (i.e., from right to left). Hence, starting from the last chance node, the outcomes associated with each of the branches of the chance node are weighted by their chance of occurrence. The sum of weighted outcomes across branches belonging to the same chance node is assigned to that chance node. This process is repeated for all branches and nodes starting from the end (i.e. terminal nodes) to the root (i.e., decision node) of the tree for all strategies.

The “folding back” method requires sequential steps that are implemented iteratively. These iterative processes can become computationally demanding if the number of nodes in the tree, and the number of branches attached to them, increases. The application of the folding back method in R requires the connectivity of the tree to be described first before the algorithm is executed. Given the computational inefficiency of the algorithm we do not provide a “folding back” solution in R.

An alternative algorithmic approach could be used where, once the connectivity of the network is described in R, an algorithm can be used to detect the complete paths from the decision node to any terminal node and compute the probability weight for each such path. The vector of probability weights can be multiplied with the vector of outcomes per path using linear algebra to calculate the expected outcome for the strategy. In R, the package `igraph` provides such functionality, which we utilize in a set of custom functions that facilitate the solution of the decision tree problem. We have wrapped those functions in `DecTree`, an R package which we illusterate it’s use in the Examples section below.

OpenTree

A drawback of using R for developing decision trees is the lack of a user-friendly interface that can facilitate drawing the tree. In an attempt to alleviate this limitation we developed the OpenTree software (www.myopentree.com). OpenTree is a free, JavaScript-based web interface that allows the user to construct decision trees through a graphical user interface. Once the structure of the tree is drawn, OpenTree can translate the decision tree into a set of R functions and objects, so it can be readily incorporated into R.

DecTree

DecTree is an R package that allows for the construction, estimation and graphical representation of decision trees in R. Given the absence of a graphical interface that allows users to draw a decision tree in R, DecTree relies on a data frame that presents the series of branches as characterized by

- the starting node
- the terminal node

- the name of the branch
- the probability of the branch occuring
- the outcomes associated with traversing this branch

An illustration of the input data structure is provided in the example below. Through this information the functions in the DecTree package can automatically identify the type of node (decision, chance or terminal), the characteristics of the tree (height, paths etc) and calculate the probability of each terminal path to be traversed. The package also provides functionality for the visualization of the decision tree. Finally, DecTree can function with strucutre input regarding the tree that originates directly from OpenTree.

3.3 Examples

We illustrate the "closed-form", OpenTree and DecTree methods described above using a previously published decision tree designed to evaluate the cost-effectiveness of ondansetron compared to metoclopramide as prophylaxis against nausea and vomitting in cancer patients undergoing chemotherapy (Buxton and O'Brien, 1992). This decision tree has been used for educational purposes before. (Drummond et al., 2015) In this decision tree, patients receiving either of the two prophylactic treatments can still experience significant vomiting ("emesis"), as well as adverse drug effects (ADEs) (Figure 3.2). Patients can receive treatment for the ADEs which is expected to affect the probability of an ADE being resolved. The two treatments differ with respect to the probability of emesis, and presence and resolution of ADEs. The economic outcome of the cost-effectiveness analysis (CEA) was the total expected healthcare costs of a patient under each prophylaxis strategy. The clinical outcome was the probability of successful treatment, defined as absence of significant emesis or ADEs. The probabilities and costs associated with each event for both treatments are presented in Table 3.1.

TABLE 3.1: Description of parameters with their R name and value.

Parameters	**R name**	**Ondansetron**	**Metoclopramide**
Strategy names	`Strategies`	Ondansetron	Metoclopramide
Probabilities			
Significant emesis	`p.sem`	0.25	0.58
Significant ADE			
- after sign. emesis	`p.semADE`	0.26	0.34
- without sign. emesis	`p.nosemADE`	0.11	0.12
Treatment of sign. ADE			

Parameters	**R name**	**Ondansetron**	**Metoclopramide**
- after sign. emesis	`p.semADETrt`	0.6	0.6
- without sign. emesis	`p.nosemADETrt`	0.17	0.5
Resolution of treated ADE			
- after sign. emesis	`p.semADETrtRes`	0.66	0.78
- without sign. emesis	`p.nosemADETrtRes`	1	1
Costs			
- Treatment	`c.trt`	$10	$10
- Episode of sign. emesis	`c.sem`	$30	$30
- ADE	`c.ade`	$20	$20
- Treatment of ADE	`c.adeTrt`	$5	$5

3.4 Defining the input parameters

We import the information from 3.1 in R. We follow the R notation used in previous chapters. We store all information for every strategy in a list. Within the list we construct variables that capture the height of the tree, the number of paths, the (conditional) probability of each branch to be transversed, as well as the health and cost outcome of each path. Similarly, we capture the cost and outcome parameters associated with each terminal node in corresponding variables. For simplicity, we only present the parameters related to the Ondansetron strategy. The R code for both strategies is found in Supplementary Appendix 2. Alternatively to storing all information in a list, we could also load the data as a dataframe.

```r
l.On <- list(
  # Height of the decision tree
  H = 4,
  # number of terminal nodes, or paths
  G = 10,

  # Prob. of significant emesis
  p.sem = 0.25,
  # Prob. of ADE after emesis
  p.semAde = 0.26,
  # Prob. of treatment after ADE with emesis
  p.semAdeTrt = 0.60,
  # Prob. of resolution of treated ADE with emesis
  p.semAdeTrtRes = 0.66,
```

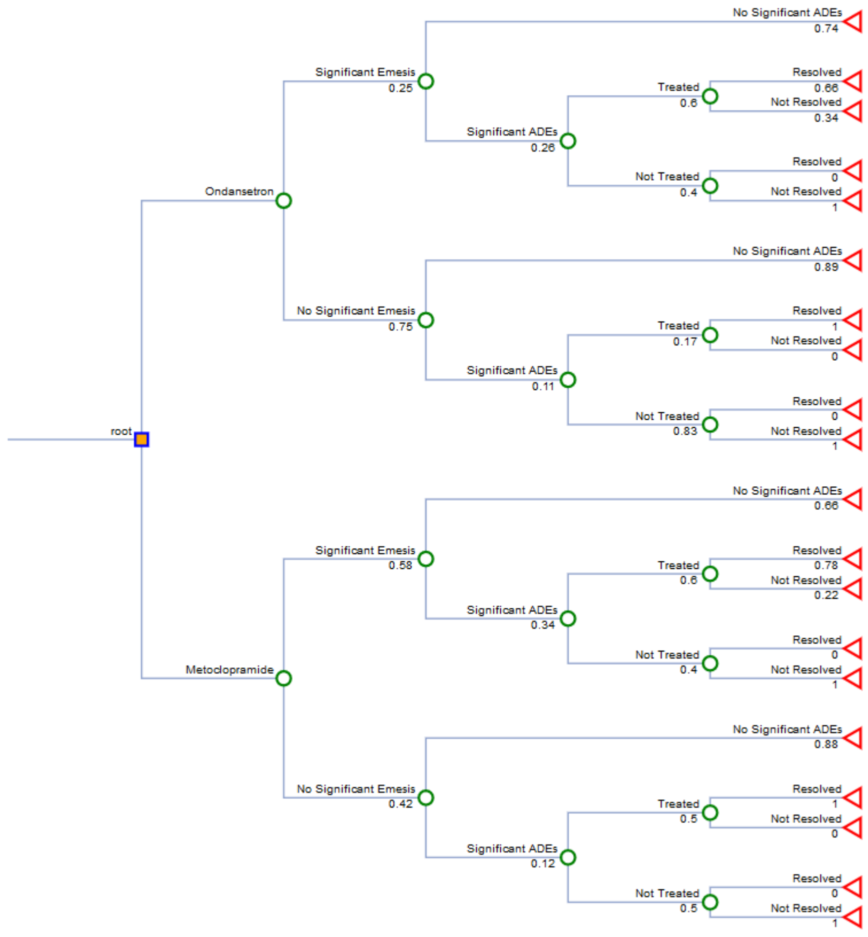

FIGURE 3.2: Decision tree of Ondansetron and Metoclopramide cost effectiveness. ADEs : Adverse Drug Events. This figure was built with the OpenTree software.

```
  # Prob. of resolution after ADE that did not require treatment
  p.semAdeNotrtRes = 1,

  # Prob. of ADE without emesis
  p.nosemAde = 0.11,
  # Prob. of treatment after ADE without emesis
  p.nosemAdeTrt = 0.17,
  # Prob. of resolution of treated ADE without emesis
  p.nosemAdeTrtRes = 1,
  # Prob. of resolution of non treated ADE without emesis
  p.nosemAdeNotrtRes = 1,

  # cost of treatment
  c.trt = 10,
  # cost of episode of emesis
  c.sem = 30,
  # cost of ADE
  c.ade = 20,
  # cost of treating an ADE
  c.adeTrt = 5
)
```

We then create vectors, both of size G, capturing the cost and effects outcomes for each path.

```
# create vector of cost outcomes for the the Ondansetron arm
# and the Metoclopramide arm
# each element after a comma within the c()
# object represents an element of the vector

# Vector of Costs per Path
v.c <- with(l.On,
{
  c.trt + c(c.sem,
            c.sem + c.ade + c.adeTrt,
            c.sem + c.ade + c.adeTrt,
            c.sem + c.ade,
            c.sem + c.ade,
            0,
            c.ade + c.adeTrt,
            c.ade + c.adeTrt,
            c.ade,
            c.ade)
})

# vector of health outcomes
v.e <-  c(0, 0, 0, 0, 0, 1, 0, 0, 0, 0)
```

3.5 Estimating the expected value

Once all variables have been defined, we calculate the expected outcome(s) using the closed form solution described above. Using equation (1), operationalized in R similarly to Box 1, we first create vectors of conditional probabilities for each path and subsequently multiply them across each path. Finally, we multiply the weights of each path with the corresponding outcome and sum across all paths. Below we present the approach for the Ondansetron strategy. Similar code is developed for the Metoclopramide strategy, it is provided in the Appendix of this chapter.

```
v.w.On <-  with(data = l.On,
{
  l.p <- list(
    c(p.sem, 1 - p.semAde),
    c(p.sem,p.semAde, p.semAdeTrt, p.semAdeTrtRes),
    c(p.sem, p.semAde, p.semAdeTrt,1 - p.semAdeTrtRes),
    c(p.sem, p.semAde, 1 - p.semAdeTrt,p.semAdeNotrtRes),
    c(p.sem, p.semAde, 1 - p.semAdeTrt, 1 - p.semAdeNotrtRes),
    c(1 - p.sem, 1 - p.nosemAde),
    c(1 - p.sem,p.nosemAde, p.nosemAdeTrt, p.nosemAdeTrtRes),
    c(1 - p.sem,p.nosemAde, p.nosemAdeTrt, 1 - p.nosemAdeTrtRes),
    c(1 - p.sem,p.nosemAde, 1 - p.nosemAdeTrt,
      p.nosemAdeNotrtRes),
    c(1 - p.sem,p.nosemAde, 1 - p.nosemAdeTrt,
      1 - p.nosemAdeNotrtRes)
  )

  v.w <- vector(length = G)  # initialize vector of probabilities
  for (g in 1:G)
  {
    # the product of (conditional) probabilities for every path
    v.w[g] <- prod(l.p[[g]])
  }
  return(v.w)
})

# expected costs - Ondansetron
ev.c.On  <- sum(v.w.On * v.c)
# expected effects - Ondansetron
ev.e.On  <- sum(v.w.On * v.e)
```

Once the cost and effects are calculated for both strategies, we calculate the incremental costs and effects, and incremental cost-effectiveness ratio (ICER).

```
Strategies <- c("Metoclopramide", "Ondansetron")
delta.c <- ev.c.On - ev.c.Met
delta.e <- ev.e.On - ev.e.Met
icer    <- delta.c / delta.e
table.res <- matrix(c(ev.c.Met, ev.e.Met, 0, 0, NA,
                    ev.c.On, ev.e.On, delta.c, delta.e, icer),
                    nrow = 2, byrow = T)
rownames(table.res) <- Strategies
colnames(table.res) <- c("Costs", "Effects","$\\Delta$ Costs",
                         "$\\Delta$ Effects","ICER")
```

TABLE 3.2: The incremental cost-effectiveness table for the emesis prophylaxis example

	Costs	Effects	Δ Costs	Δ Effects	ICER
Metoclopramide	33.07	0.370	0.00	0.000	NA
Ondansetron	20.71	0.668	-12.35	0.298	Dominant

The incremental cost-effectiveness table for the emesis prophylaxis example is presented in Table 3.2.

ICER: Incremental Cost Effectiveness Ratio

From Table 3.2, treatment with Ondansetron has both a lower expected cost and a higher expected probability of being successful. This means that Ondasetron is a dominant strategy.

3.6 Using DecTree and OpenTree to solve decision trees

As discussed above, an alternative approach is to use DecTree to estimate the decision tree in R. We create a new tree in the OpenTree interface, presented in Figure 3.2. The tree structure can be presented through a .csv file with information that is compatible to the requirements of the DecTree functions. The .csv file can be found in the supplementary materials of the book. Subsequently, we load the package DecTree with all necessary functionality.

```r
# loading the branches structure
branches <- read.csv("data/branches.csv")
library(dectree)
```

Notice that all nodes need to have a unique name that defines them. This tabular structure of the decision tree outlined in the 'branches' object can be utilized in constructing a decision tree through the function `create_tree` from the DecTree package.

```r
# create a tree using the structure provided in the 'branches'
# data frame
tree <- create_tree(branches)
```

The object `tree` is an `igraph` object which contains information on the name and number of nodes, their connectivity, their attributes etc. The information in the `tree` object is sufficient for us to draw the tree within R using the `plot_tree` function.

```r
# plot_tree(tree, font.size = 3, vertex.size = 3) # plots the tree
```

The arguments in the function `plot_tree` allow for significant modification of the graphical output to make the decision tree plot of publication quality. Details can be found in the help file of the function.

The final step is the estimation of the tree which can be achieved using the `estimate_tree` function.

The function esimates the tree by creating a vector of probability weights and vector(s) of outcomes for all terminal paths, and subsequently calculate the expected outcomes per strategy. The output of the function is a list containing the probability weights, the outcomes (or "payoffs") for each path and the expected value of these outcomes per strategy.

Estimation of incremental costs, incremental effectiveness and ICER can be achieved in the same way as described above with results identical to those in Table 3.1.

3.7 Discussion

This chapter focuses on the development of a decision tree model within the R environment. It provides the reader with two methodological solutions for the calculation of an outcome's expected value using a decision tree. Specifically, we have presented a solution based on the multiplication of conditional probabilities and an alternative solution based on an automated R function. For both methods, we provide the R-based solutions through an illustrative example on the cost-effectiveness of Ondansetron versus Metoclopramide for emesis prevention during chemotherapy treatment.

While the two methods covered in this chapter are not the only methods for expected value calculations in decision trees, they do span a space of algorithmic approaches. The iterative approach is arguably more intuitive to implement, but may become computationally intensive for more complex trees. The approach that relies on DecTree is simpler to code although with it the solution of the tree becomes a bit more of a "black box".