Introduction to R

Eva Enns & Zoe Kao

2020-01-01

Contents

R Basics

Install R and RStudio

Data type

Matrix & Array

Data frame

List

Data input and output

For loop

If statement

R function

Plots

Decision tree

For a simple decision tree, we can draw the tree easily. As the decision tree grows, there is software helping you draw the tree such as TreeAge and Amua. In this class, we use Amua because it is free. We will show how to use Amua to build a decision tree and export the tree to R script for CEA analysis.

Install Amua

Follow the instruction here to install Amua. Be aware of the difference between Mac and Windows users! After you install Amua, remember where you store the software.

Create a decision tree

Open the `Amua.jar`.

Decision, chance, terminal nodes, and decision tree

Parameterization

Adding more outcomes

Change analysis

Save the model and export the tree to R code

- Notice that Amua created a folder, called `temp_Export/`.
- In this folder, there are two `.R` files: `main.R` and `functions.R`.

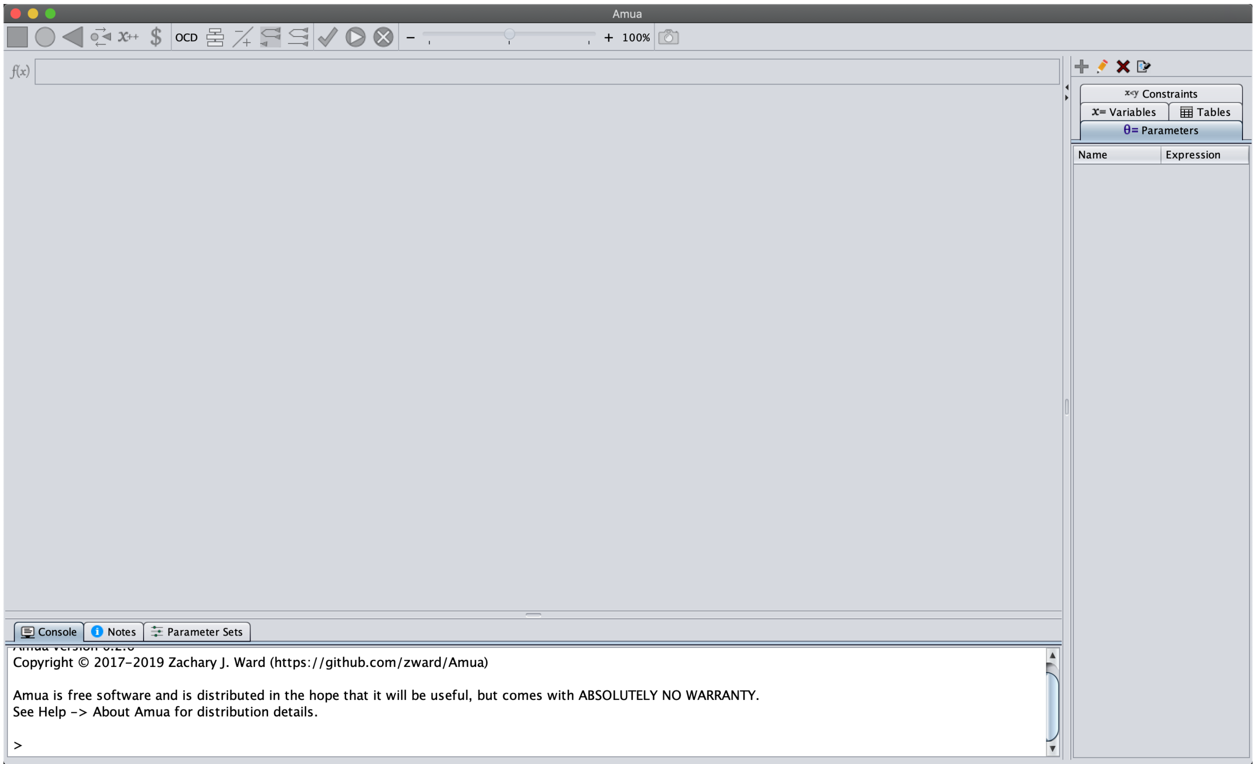


Figure 1:

- The body of the decision tree model is in `main.R`.

Example

Dracula is preparing for a spring break party tonight. But there's one final decision that Dracula is struggling with. Being a vampire, he intends to bite and suck the blood of some of his guests at the party (about 25% of them, by his best estimate). While being bitten by a vampire won't turn his guests into vampires or zombies, like some horror movies might suggest, there is a 50% chance that a vampire bite results in a rather severe bacterial infection, of which 66% of cases require hospitalization for an average of 1 night.

Being a gracious host, Dracula is considering different ways of administering antibiotic prophylaxis to his guests to reduce their risk of infection. One option is to administer the antibiotics to his victim-guests just before he bites them – this would reduce the risk of a vampire bite infection by 20%. However, the antibiotics can be even more effective, reducing the risk of infection by 90%, if administered at least 30 minutes before being bitten. To achieve this, Dracula is considering putting the antibiotics into the drinks served at the party to ensure that his guests are all properly dosed before he bites his victims. But this means that all his guests would be exposed to the antibiotics (not just those he intends to bite), and he knows that about 5% of people are severely allergic to these antibiotics and would require immediate hospitalization if exposed. Dracula is therefore also considering not administering antibiotic prophylaxis at all to avoid this harm.

However, all the healthy blood doesn't come without cost. This party will cost Dracula $1000 for a total of 200 guests (an average of $5 per guest). In addition, Dracula expects the cost of antibiotic to be $10 for each guest he bites. If Dracula decides to administer the antibiotics in all the drinks served at the party, the total cost of antibioitics is expected to be $100 (Dracula gets discount for buying a large batch of antibiotics!). Also, Dracula is willing to pay for the cost of hospitalization for any guest who experience bacteiral infection due to his bites because he feels responsible. The cost of hospitalization per person per night is $500. Dracula expects to get an average of 470 mL healthy blood by biting a guest. However, if the guest ends up hospitalized due to bacterial infection, the healthy blood that Dracula can get is reduced by 10%.

Dracula hopes you can help him determine which strategy he should choose.

You can build a decision tree via Amua.

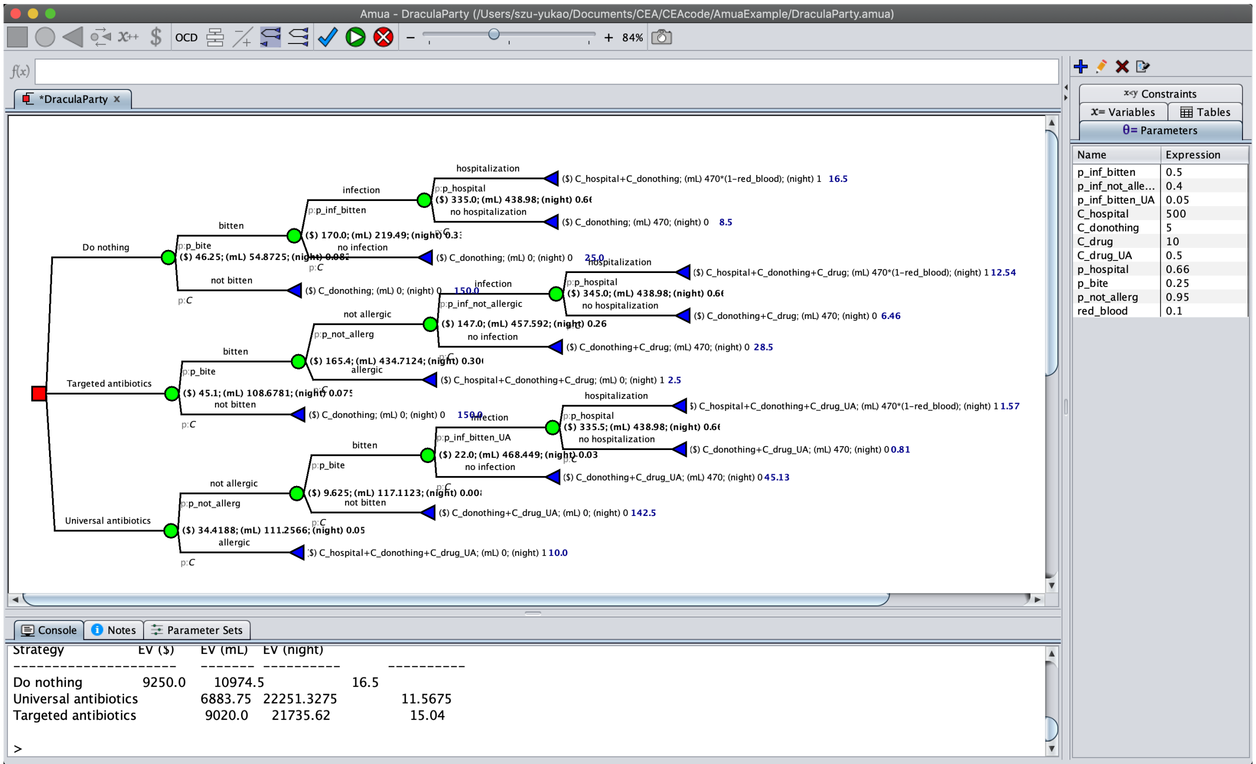


Figure 2:

Wrapper function for decision tree exported from Amua

- The decision tree created in Amua can be exported to R.
- However, the R script created by Amua is not easy to use for advance CEA analysis (e.g., PSA, one/two-way sensitivity analysis, EVPI, etc.).
- We created some wrapper functions to reshape the R script from Amua. This transformation of code allows us to do further analysis using other packages in R.
- Install the package using the following code.

```
if(!require("remotes")) install.packages("remotes")
if(!require("CEAutil")) remotes::install_github("syzoekao/CEAutil", dependencies = TRUE)

library(CEAutil)
```

- We will use two functions from the package to convert the R script exported from Amua.

1. `parse_amua_tree()`:

This function only takes an input argument, the path to the `main.R` file of the Amua decision model. The function returns a list of outputs. Output 1 `param_ls` is a list of input parameters with basecase values used in Amua. Output 2 `treefunc` is the R code of the Amua decision model formatted as text.

```r
treetxt <- parse_amua_tree("AmuaExample/DraculaParty_Export/main.R")

print(treetxt$param_ls)
```

```
## $p_inf_bitten
## [1] 0.5
##
## $p_inf_not_allergic
## [1] 0.4
##
## $p_inf_bitten_UA
## [1] 0.05
##
## $C_hospital
## [1] 500
##
## $C_donothing
## [1] 5
##
## $C_drug
## [1] 10
##
## $C_drug_UA
## [1] 0.5
##
## $p_hospital
## [1] 0.66
##
## $p_bite
## [1] 0.25
##
## $p_not_allerg
## [1] 0.95
##
## $red_blood
## [1] 0.1
```

2. `dectree_wrapper()`:

We use the two outputs from the `parse_amua_tree()` as the input arguments in the `dectree_wrapper()`. The input arguments in the wrapper function include `params_basecase`, `treefunc`, and `popsize`. `params_basecase` takes a list of named input parameters. `treefunc` takes the text file organized by the `parse_amua_tree()`. `popsize` is defaulted as 1 but you could change your population size.

```r
param_ls <- treetxt[["param_ls"]]
treefunc <- treetxt[["treefunc"]]

tree_output <- dectree_wrapper(params_basecase = param_ls, treefunc = treefunc, popsize = 200)
```

- The cost and effectiveness of each of Dracula's strategy based on the basecase values

```r
print(tree_output)
```

```
##                    name expectedEff1 expectedEff2 expectedCost
## 1             Donothing     10974.50      16.5000      9250.00
## 2 Universalantibiotics     22251.33      11.5675      6883.75
```

```
## 3  Targetedantibiotics     21735.62      15.0400      9020.00
```

Change parameter values

- Many of the parameters are organized as a variable in the decision tree. We can easily see how the cost and effectiveness vary with changes of parameters of interest.

Example 1. Dracula has been starved over the long cruel winter in Minnesota. The Spring break is the first time that his guests are willing to come to his party in several months. Therefore, Dracula is going to seize the chance to bite as many guests as possible. The probability that he bites a guest is now increased to 50%. What are the cost and effectiveness of each strategy?

```r
param_ls$p_bite <- 0.5

tree_output <- dectree_wrapper(params_basecase = param_ls, treefunc = treefunc, popsize = 200)

print(tree_output)
```

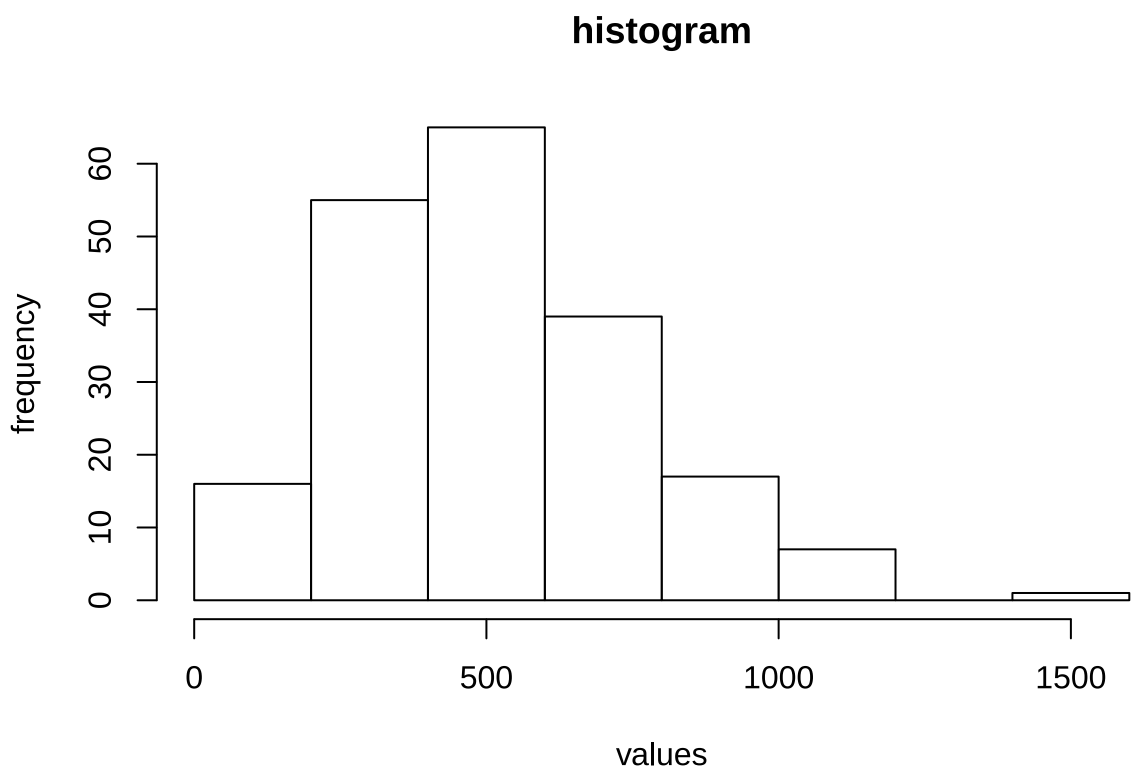
```
##                    name expectedEff1 expectedEff2 expectedCost
## 1             Donothing     21949.00       33.000      17500.0
## 2 Universalantibiotics     44502.65       13.135       7667.5
## 3  Targetedantibiotics     43471.24       30.080      17040.0
```

Example 2. The cost of hospitalization due to bacterial infection varies from guest to guest depending on the healthcare that a guest has. Overall, the cost of hospitalization has a mean of \$500 with a standard deviation \$300 following a gamma distribution. What are the mean cost and effectiveness of the party across all 200 guests?

```r
if(!require(dampack)) remotes::install_github("DARTH-git/dampack", dependencies = TRUE)

require(dampack)

C_hospital <- gen_psa_samp(params = c("C_hospital"),
                           dists = c("gamma"),
                           parameterization_types = c("mean, sd"),
                           dists_params = list(c(500, 250)),
                           nsamp = 200)

cat("mean and sd are", c(mean(C_hospital$C_hospital), sd(C_hospital$C_hospital)), ", respectively.")
```

```
## mean and sd are 510.1781 242.8418 , respectively.
```

```r
hist(C_hospital$C_hospital, main = "histogram", xlab = "values", ylab = "frequency")
```




```r
tree_vary <- dectree_wrapper(params_basecase = param_ls,
                             treefunc = treefunc,
                             popsize = 1,
                             vary_param_samp = C_hospital)
```

- The elements in the tree_vary output

```r
print(names(tree_vary))
```

```
## [1] "expectedEff1" "expectedEff2" "expectedCost" "param_samp"
```

- Let's take a look at some elements in the tree_vary output

```r
print(head(tree_vary$param_samp))
```

```
##   nsamp C_hospital
## 1     1   504.2479
## 2     2   623.0702
## 3     3   290.6661
## 4     4   411.9706
## 5     5   286.8879
## 6     6   364.5243
```

```r
print(head(tree_vary$expectedEff1))
```

```
##   Donothing Universalantibiotics Targetedantibiotics
## 1   109.745             222.5133            217.3562
## 2   109.745             222.5133            217.3562
## 3   109.745             222.5133            217.3562
## 4   109.745             222.5133            217.3562
## 5   109.745             222.5133            217.3562
## 6   109.745             222.5133            217.3562
```

```r
print(head(tree_vary$expectedCost))
```

```
##   Donothing Universalantibiotics Targetedantibiotics
## 1  88.20091             38.61648            85.83889
## 2 107.80658             46.42013           103.70975
## 3  52.95991             24.58950            53.71619
## 4  72.97516             32.55617            71.96038
## 5  52.33650             24.34136            53.14794
## 6  65.14651             29.44013            64.82445
```

- The mean cost

```r
print(summary(tree_vary$expectedCost))
```

```
##    Donothing      Universalantibiotics Targetedantibiotics
##  Min.   : 15.69   Min.   :  9.755      Min.   : 19.74
##  1st Qu.: 56.34   1st Qu.: 25.935      1st Qu.: 56.80
##  Median : 85.21   Median : 37.425      Median : 83.11
##  Mean   : 89.18   Mean   : 39.006      Mean   : 86.73
##  3rd Qu.:109.64   3rd Qu.: 47.150      3rd Qu.:105.38
##  Max.   :250.79   Max.   :103.334      Max.   :234.05
```

Markov model

- We will show how to code a Markov model step by step.

Structure of a Markov model

```r
markov_model <- function(params_basecase) {
  #### Transform parameters if needed

  #### Create the transition probability matrices using array

  #### Create the trace matrix and perform calculation using the Markov model

  #### Return the relevant results
  return()
}
```

DAMPACK