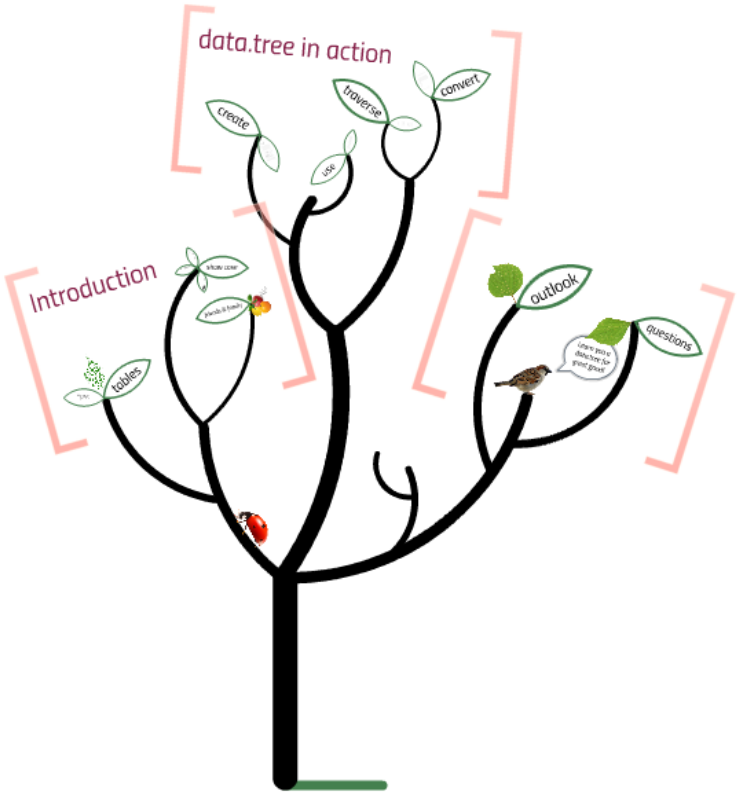
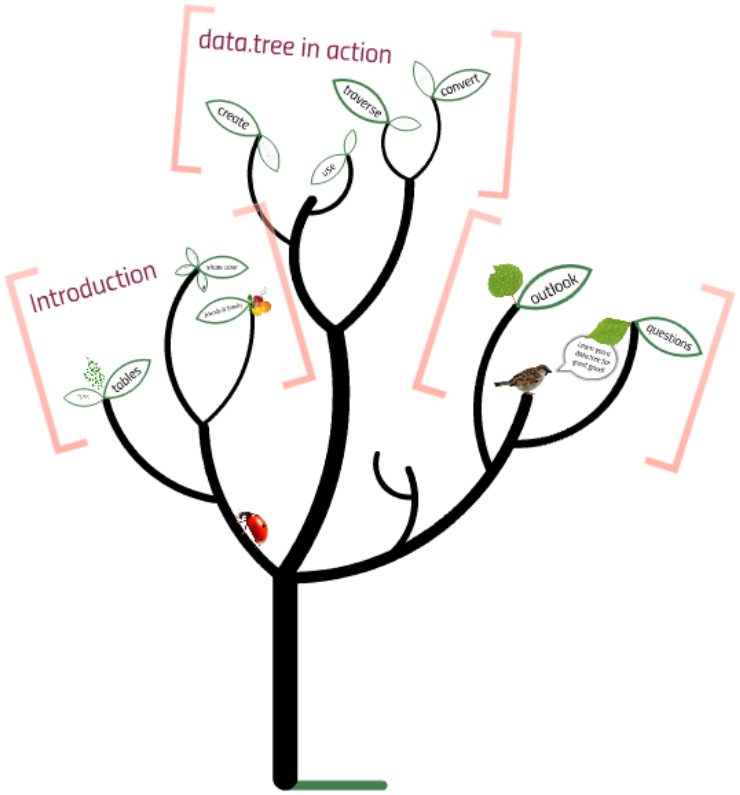


data.tree

A better way to manage hierarchical data

useR! 2015
http://github.com/gluc/useR15

christoph.glur@ipub.com



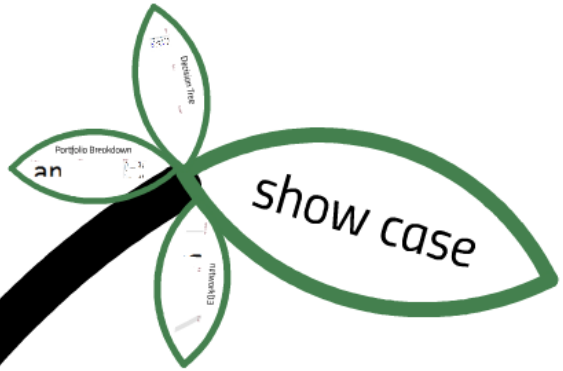
data.tree

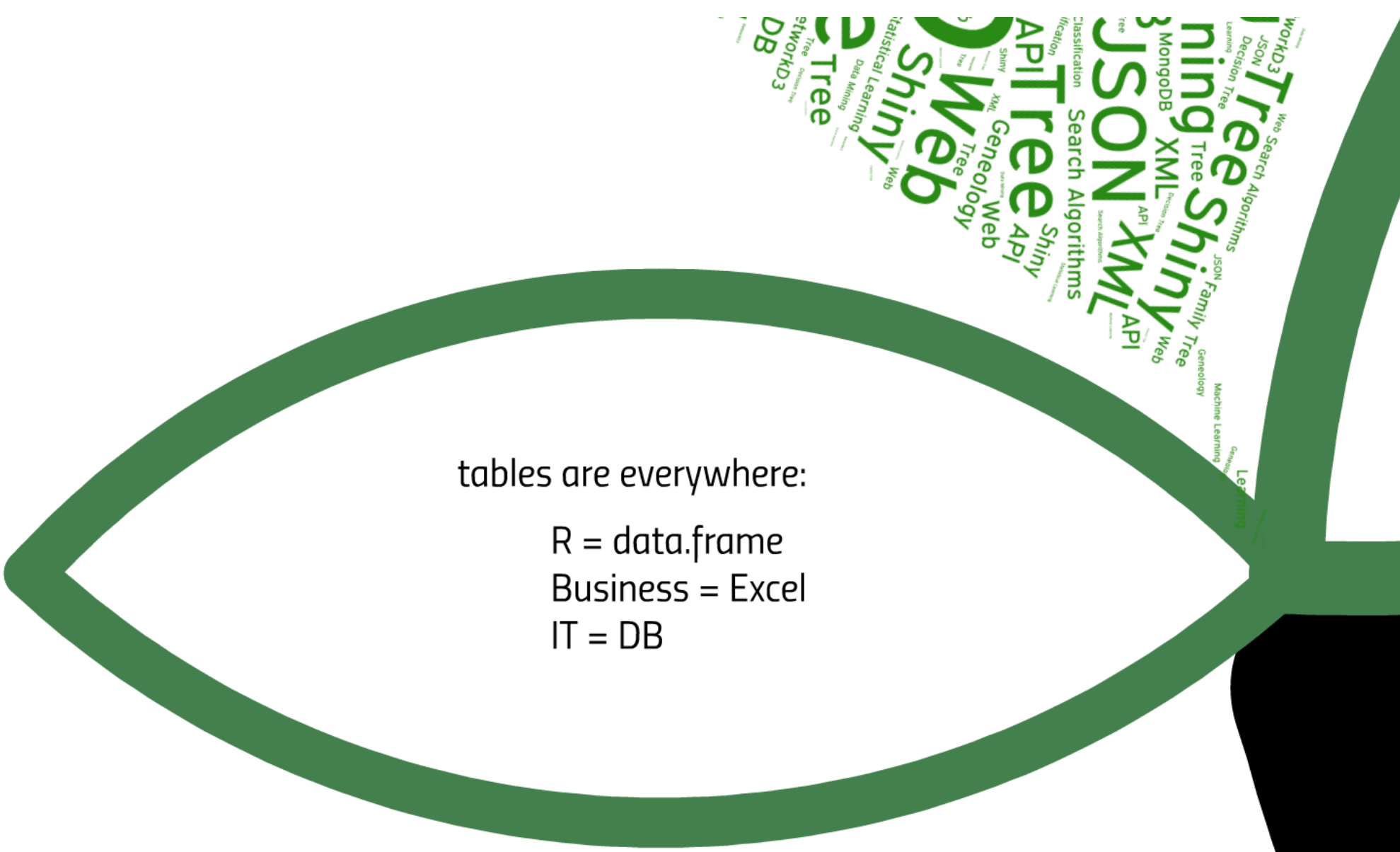
A better way to manage hierarchical data

useR! 2015
http://github.com/gluc/useR15

christoph.glur@ipub.com

Introduction





tables are everywhere:

R = data.frame

Business = Excel

IT = DB


Web
Decision
Tree
NetworkD3
Search Algorithms
XML
JSON
API
Data Mining
MongoDB
Learning
Shiny
Classification
Family Tree
Geneology
Machine Learning
Statistical Learning
Decision Tree

everywhere:

show case

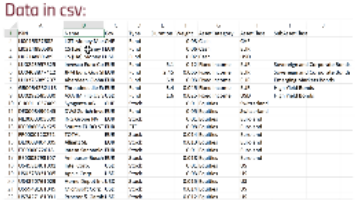
Decision Tree

Portfolio Breakdown

networkD3

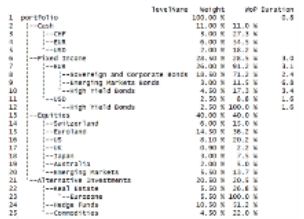
Portfolio Breakdown

Data in csv:



Code:

Result:



	TreeName	Weight	YoY Duration
1	portfolio	100.00 %	0.0
2	├─Cash	11.00 %	13.0 %
3	│ ├─CHF	5.00 %	27.5 %
4	│ ├─EUR	6.00 %	58.5 %
5	│ └─USD	0.00 %	18.2 %
6	├─Fixed Income	28.50 %	28.5 %	3.0
7	│ ├─AAA	24.00 %	61.2 %	6.1
8	│ │ ├─sovereign and corporate bonds	18.50 %	71.2 %	2.4
9	│ │ └─emerging markets bonds	5.50 %	11.5 %	6.8
10	│ └─high yield Bonds	4.50 %	17.3 %	3.4
11	└─USD	2.50 %	0.0 %	1.0
12	└─high yield Bonds	2.50 %	100.0 %	1.0
13	├─Equities	40.00 %	40.0 %	
14	│ ├─Switzerland	6.00 %	15.0 %	
15	│ ├─Euroland	14.50 %	36.2 %	
16	│ ├─US	8.10 %	20.2 %	
17	│ ├─UK	0.90 %	2.2 %	
18	│ ├─Japan	3.00 %	7.5 %	
19	│ ├─Australia	2.00 %	5.0 %	
20	│ └─emerging markets	5.50 %	13.7 %	
21	└─Alternative Investments	20.50 %	20.5 %	
22	├─real estate	5.50 %	26.8 %	
23	│ └─Eurozone	5.50 %	100.0 %	
24	├─Hedge Funds	10.50 %	51.2 %	
25	└─Commodities	4.50 %	22.0 %	

Data in csv:

	A	B	C	D	E	F	G	H	I	J
1	ISIN	Name	Ccy	Type	Duration	Weight	AssetCategory	AssetClass	SubAssetClass	
2	LI0015327682	LGT Money Mar	CHF	Fund		0.03	Cash	CHF		
3	LI0214880598	CS (Lie) ney M	EUR	Fund		0.06	Cash	EUR		
4	LI0214880689	CS (Lie) Money M	USD	Fund		0.02	Cash	USD		
5	LU0243957825	Invesco Euro Co	EUR	Fund	5.1	0.12	Fixed Income	EUR	Sovereign and Corporate Bonds	
6	LU0408877412	JPM Euro Gov Sh	EUR	Fund	2.45	0.065	Fixed Income	EUR	Sovereign and Corporate Bonds	
7	LU0376989207	Aberdeen Globa	EUR	Fund	6.8	0.03	Fixed Income	EUR	Emerging Markets Bonds	
8	GB00B42R2118	Threadneedle Eu	EUR	Fund	3.4	0.045	Fixed Income	EUR	High Yield Bonds	
9	LU0292585030	AXA IM FIIS US S	USD	Fund	1.6	0.025	Fixed Income	USD	High Yield Bonds	
10	CH0011037469	Syngenta AG	CHF	Stock		0.01	Equities	Switzerland		
11	DE0008490145	DWS Zurich Inve	EUR	Fund		0.05	Equities	Switzerland		
12	NL0000303600	ING Grope NV	EUR	Stock		0.01	Equities	Euroland		
13	IE00B60SWX25	Source EURO ST	EUR	ETF		0.08	Equities	Euroland		
14	FR0000120271	TOTAL	EUR	Stock		0.014	Equities	Euroland		
15	DE0008404005	Allianz SE	EUR	Stock		0.013	Equities	Euroland		
16	IT0000072618	Intesa Sanpaolo	EUR	Stock		0.01	Equities	Euroland		
17	BE0003793107	Anheuser-Busch	EUR	Stock		0.018	Equities	Euroland		
18	US4581401001	Intel Corp.	USD	Stock		0.01	Equities	US		
19	US0378331005	Apple Corp	USD	Stock		0.03	Equities	US		
20	US4370761029	Home Depot Inc	USD	Stock		0.015	Equities	US		
21	US5949181045	Microsoft Corp.	USD	Stock		0.014	Equities	US		
22	US7427181091	Procter & Gamb	USD	Stock		0.012	Equities	US		

Code:

```
1  library(data.tree)
2
3  #read from file
4  pfodf <- read.csv('../useR15/data/portfolio.csv', stringsAsFactors = FALSE)
5  pfodf
6
7  #convert to data.tree
8  pfodf$pathString <- paste("portfolio", pfodf$AssetCategory, pfodf$AssetClass, pfodf$SubAssetClass, pfodf$ISIN, sep = "/")
9  pfo <- as.Node(pfodf)
10
11 #Calculate breakdown
12 pfo$Get(Aggregate, "Weight", sum, assign = "Weight")
13 pfo$Get(function(x) x$Weight / x$parent$Weight, traversal = "post-order", assign = "WoP")
14 pfo$Get(attribute = Aggregate,
15         function(x) x$Weight * x$Duration / x$parent$Weight,
16         fun = function(x) sum(x, na.rm = TRUE),
17         traversal = "post-order",
18         assign = "Duration")
19
20 #Formatters
21 pfo$formatters$WoP <- function(x) FormatPercent(x, digits = 1)
22 pfo$formatters$Weight <- FormatPercent
23 pfo$formatters$Duration <- function(x) {
24   if (x != 0) res <- FormatFixedDecimal(x, digits = 1)
25   else res <- ""
26   return (res)
27 }
28
29 #Print
30 print(pfo,
31       "Weight",
32       "WoP",
33       "Duration",
34       filterFun = function(x) !x$isLeaf)
```

Result:

	levelName	Weight	WoP	Duration
1	portfolio	100.00 %		0.8
2	¦--Cash	11.00 %	11.0 %	
3	¦ ¦--CHF	3.00 %	27.3 %	
4	¦ ¦--EUR	6.00 %	54.5 %	
5	¦ °--USD	2.00 %	18.2 %	
6	¦--Fixed Income	28.50 %	28.5 %	3.0
7	¦ ¦--EUR	26.00 %	91.2 %	3.1
8	¦ ¦ ¦--Sovereign and Corporate Bonds	18.50 %	71.2 %	2.4
9	¦ ¦ ¦--Emerging Markets Bonds	3.00 %	11.5 %	6.8
10	¦ ¦ °--High Yield Bonds	4.50 %	17.3 %	3.4
11	¦ °--USD	2.50 %	8.8 %	1.6
12	¦ °--High Yield Bonds	2.50 %	100.0 %	1.6
13	¦--Equities	40.00 %	40.0 %	
14	¦ ¦--Switzerland	6.00 %	15.0 %	
15	¦ ¦--Euroland	14.50 %	36.2 %	
16	¦ ¦--US	8.10 %	20.2 %	
17	¦ ¦--UK	0.90 %	2.2 %	
18	¦ ¦--Japan	3.00 %	7.5 %	
19	¦ ¦--Australia	2.00 %	5.0 %	
20	¦ °--Emerging Markets	5.50 %	13.7 %	
21	°--Alternative Investments	20.50 %	20.5 %	
22	¦--Real Estate	5.50 %	26.8 %	
23	¦ °--Eurozone	5.50 %	100.0 %	
24	¦--Hedge Funds	10.50 %	51.2 %	
25	°--Commodities	4.50 %	22.0 %	

Decision Tree

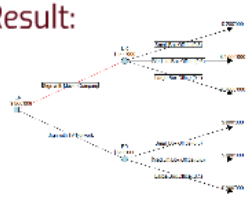
Data:

yaml

```yaml
name: Jenny Lind
type: decision
sign with Movie company:
  type: chance
  Small Box Office:
    type: terminal
    p: 0.3
    payoff: 200000
  Medium Box Office:
    type: terminal
    p: 0.6
    payoff: 1000000
  Large Box Office:
    type: terminal
    p: 0.1
    payoff: 3000000
Sign with TV Network:
  type: chance
  Small Box Office:
    type: terminal
    p: 0.3
    payoff: 600000
  Medium Box Office:
```

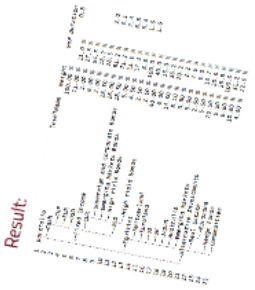
Code:

Result:



kdown

Result:



Data:

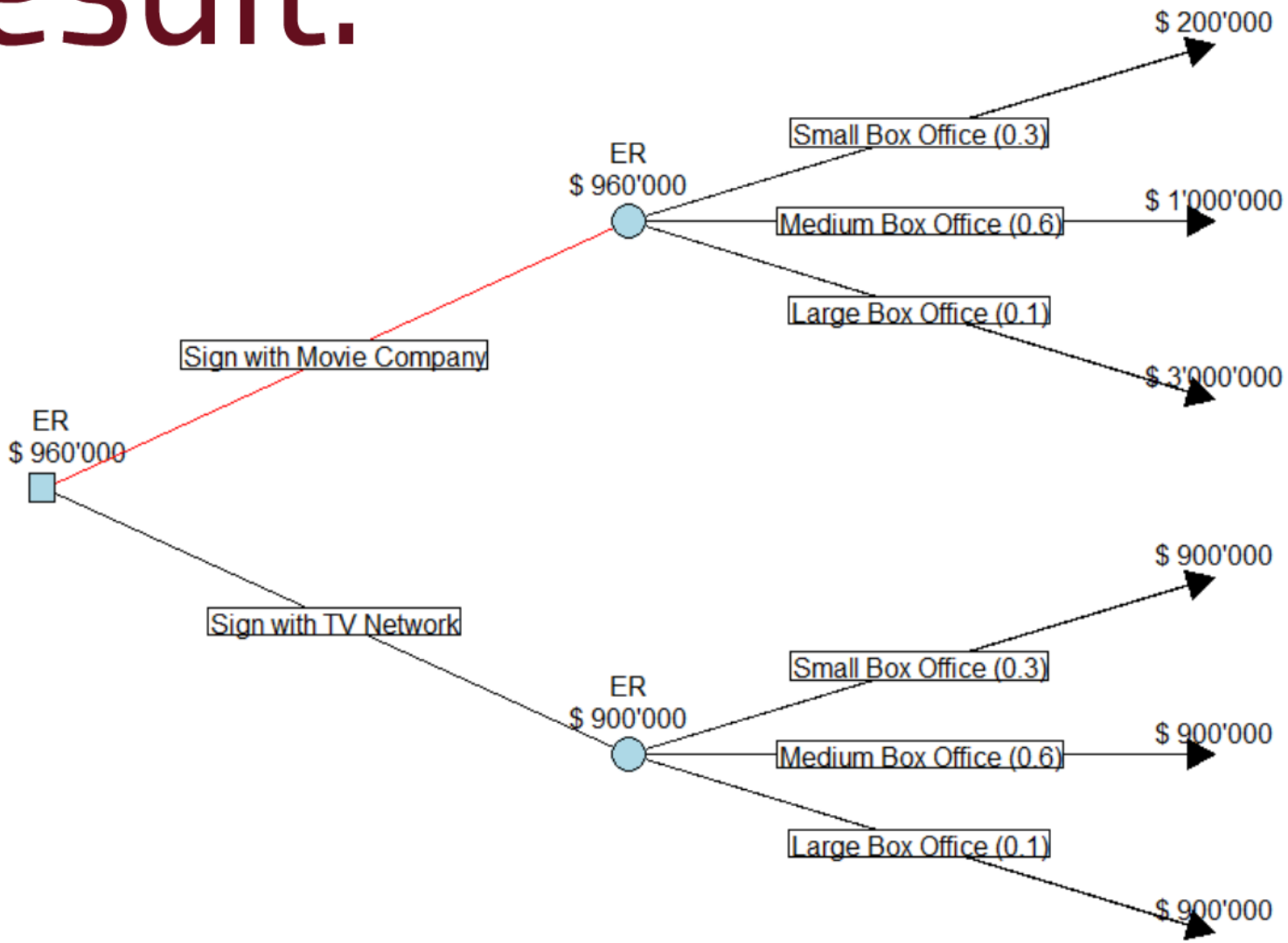
yaml

```yaml
name: Jenny Lind
type: decision
Sign with Movie Company:
  type: chance
  Small Box Office:
    type: terminal
    p: 0.3
    payoff: 200000
  Medium Box Office:
    type: terminal
    p: 0.6
    payoff: 1000000
  Large Box Office:
    type: terminal
    p: 0.1
    payoff: 3000000
Sign with TV Network:
  type: chance
  Small Box Office:
    type: terminal
    p: 0.3
    payoff: 900000
  Medium Box Office:
```

Code:

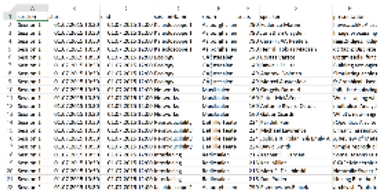
```r
library(data.tree)
library(yaml)

#load from file
fileName <- '../useR15/data/jennylind.yaml'
l <- yaml.load_file(fileName)
jl <- as.Node(l)

#calculate decision tree

payoff <- function(x) {
  if (x$type == 'terminal') res <- x$payoff
  else if (x$type == 'chance') res <- x$Aggregate(function(node) node$payoff * node$p, sum)
  else if (x$type == 'decision') res <- x$Aggregate("payoff", max)
  return (res)
}

jl$Get(payoff, traversal = "post-order", assign = "payoff")

decision <- function(x) {
  if (x$type == 'decision') {
    po <- sapply(x$children, function(child) child$payoff)
    res <- names(po[po == x$payoff])
  } else res <- NULL
  return (res)
}

jl$Get(decision, assign = "decision")

#Plot the decision tree whit ape

library(ape)
jl$Revert()
jlp <- as.phylo(jl)
par(mar=c(1,1,1,1))
plot(jlp, show.tip.label = FALSE, type = "cladogram")


nodelabel <- function(x) {
  po <- paste0( '$ ', format(x$payoff, scientific = FALSE, big.mark = "'"))
  if (x$type == 'terminal') return (po)
  return ( paste0('ER\n', po) )
}

for (node in jl$leaves) edges(GetPhyloNr(node$parent, "node"), GetPhyloNr(node, "node"), arrows = 2, type = "triangle", angle = 60)

for(node in jl$Get(function(x) x)) {
  if(node$type == 'decision') {
    nodelabels(nodelabel(node), GetPhyloNr(node, "node"), frame = 'none', adj = c(0.3, -0.5))
  } else if(node$type == 'chance') {
    if (node$name == node$parent$decision) edges(GetPhyloNr(node$parent, "node"), GetPhyloNr(node, "node"), col = "red")
    nodelabels(" ", GetPhyloNr(node, "node"), frame = "circle")
    nodelabels(nodelabel(node), GetPhyloNr(node, "node"), frame = 'none', adj = c(0.5, -0.5))
    edgelabels(node$name, GetPhyloNr(node, "edge"), bg = "white")
  } else if(node$type == 'terminal') {
    tiplabels(nodelabel(node), GetPhyloNr(node, "node"), frame = "none", adj = c(0.5, -0.6))
    edgelabels(paste0(node$name," (", node$p, ")"), GetPhyloNr(node, "edge"), bg = "white")
  }
}

nodelabels("   ", GetPhyloNr(jl, "node"), frame = "rect")
```

Result:



networkD3

Data in csv:



Code:

Result:

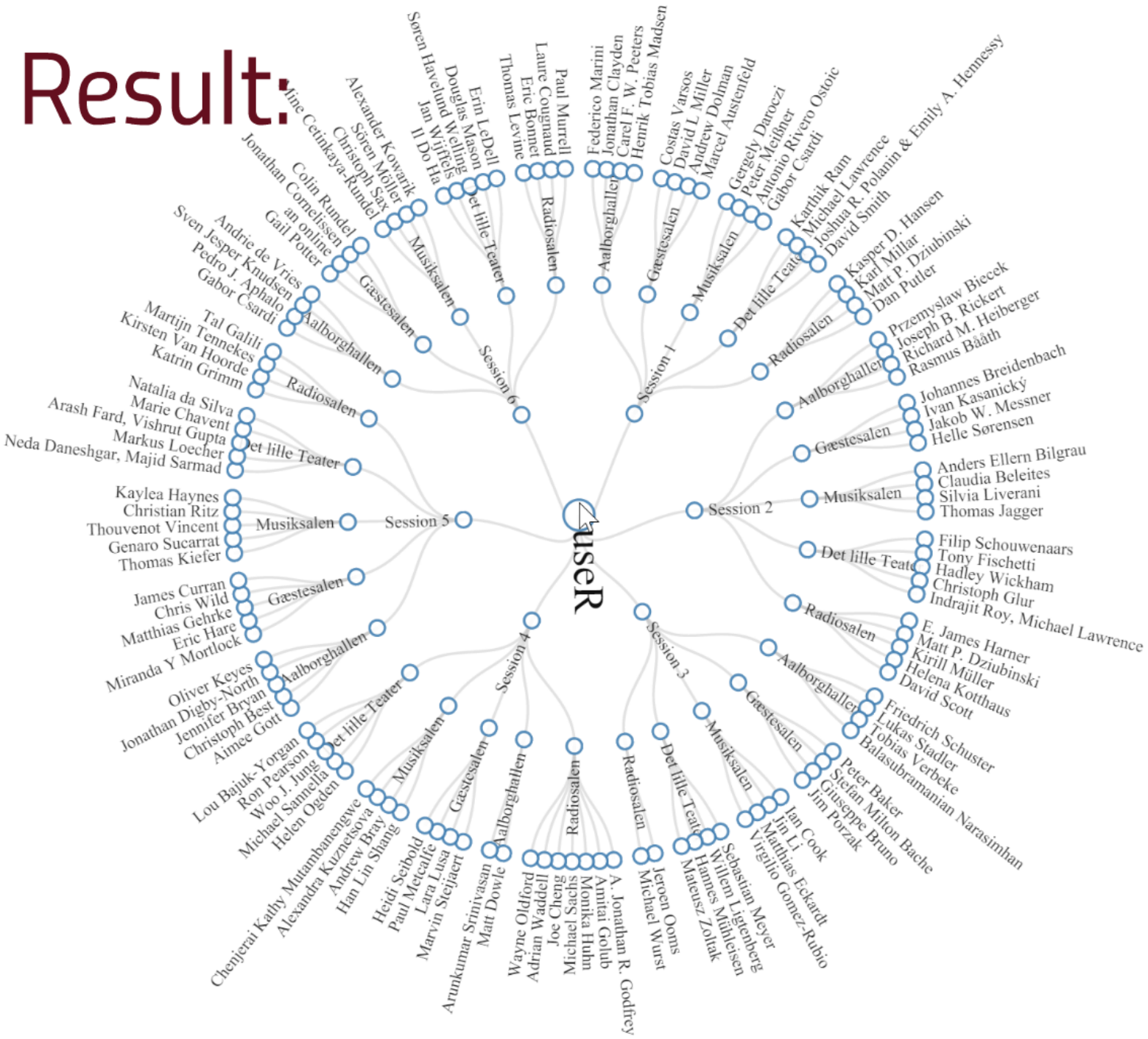
Data in csv:

	A	B	C	D	E	F	G	H
1	session	start	end	sessionName	room	seats	speaker	presentation
2	Session 1	01.07.2015 10:30	01.07.2015 12:00	Kaleidoscope 1	Aalborghallen	790	Federico Marini	flowcatchR: A use
3	Session 1	01.07.2015 10:30	01.07.2015 12:00	Kaleidoscope 1	Aalborghallen	790	Jonathan Clayden	Image processing
4	Session 1	01.07.2015 10:30	01.07.2015 12:00	Kaleidoscope 1	Aalborghallen	790	Carel F. W. Peeters	rags2ridges: Ridge
5	Session 1	01.07.2015 10:30	01.07.2015 12:00	Kaleidoscope 1	Aalborghallen	790	Henrik Tobias Madsen	dgRaph: Discrete
6	Session 1	01.07.2015 10:30	01.07.2015 12:00	Ecology	GÃ¦stesalen	149	Costas Varsos	Optimized R funct
7	Session 1	01.07.2015 10:30	01.07.2015 12:00	Ecology	GÃ¦stesalen	149	David L Miller	Building ecologica
8	Session 1	01.07.2015 10:30	01.07.2015 12:00	Ecology	GÃ¦stesalen	149	Andrew Dolman	Simulating ecolog
9	Session 1	01.07.2015 10:30	01.07.2015 12:00	Ecology	GÃ¦stesalen	149	Marcel Austenfeld	Graphical User
10	Session 1	01.07.2015 10:30	01.07.2015 12:00	Networks	Musiksalen	160	Gergely Daroczi	fbRads: Analyzing
11	Session 1	01.07.2015 10:30	01.07.2015 12:00	Networks	Musiksalen	160	Peter MeiÃŸner	Web scraping wit
12	Session 1	01.07.2015 10:30	01.07.2015 12:00	Networks	Musiksalen	160	Antonio Rivero Ostoic	multiplex: Analysi
13	Session 1	01.07.2015 10:30	01.07.2015 12:00	Networks	Musiksalen	160	Gabor Csardi	What's new in igr
14	Session 1	01.07.2015 10:30	01.07.2015 12:00	Reproducibility	Det lille Teate	224	Karthik Ram	rOpenSci: A suite
15	Session 1	01.07.2015 10:30	01.07.2015 12:00	Reproducibility	Det lille Teate	224	Michael Lawrence	Enhancing reprod
16	Session 1	01.07.2015 10:30	01.07.2015 12:00	Reproducibility	Det lille Teate	224	Joshua R. Polanin & Emily A.	A Review of Meta
17	Session 1	01.07.2015 10:30	01.07.2015 12:00	Reproducibility	Det lille Teate	224	David Smith	Simple reproducil
18	Session 1	01.07.2015 10:30	01.07.2015 12:00	Interfacing	Radiosalen	216	Kasper D. Hansen	Some lessons rele
19	Session 1	01.07.2015 10:30	01.07.2015 12:00	Interfacing	Radiosalen	216	Karl Millar	CXXR: Modernizir
20	Session 1	01.07.2015 10:30	01.07.2015 12:00	Interfacing	Radiosalen	216	Matt P. Dziubinski	Naturally Sweet R
21	Session 1	01.07.2015 10:30	01.07.2015 12:00	Interfacing	Radiosalen	216	Dan Putler	Linking R to the Sp
22	Session 2	01.07.2015 13:30	01.07.2015 15:00	Kaleidoscope 2	Aalborghallen	790	Przemyslaw Biecek	archivist: Tools fo

Code:

```
1  library(data.tree)
2
3  useRdf <- read.csv('../useR15/data/useR15.csv', stringsAsFactors = FALSE, encoding = "UTF-8")
4  useRdf$pathString <- paste("useR", useRdf$session, useRdf$room, useRdf$speaker, sep="|")
5  useRtree <- as.Node(useRdf, pathDelimiter = "|")
6
7  library(networkD3)
8  network <- treeNetwork(useRtree$ToList(mode = "explicit", unname = TRUE))
9  saveNetwork(network, "network.html", selfcontained = TRUE)
10
```

Result:



friends & family

- igraph
- dendrogram
- data.table
- partykit and friends
- list of lists of lists

- igraph
- dendrogram
- data.table
- partykit and friends
- list of lists of lists

data.tree in action

Create a tree programmatically

Tree structure:

- level 1/root: useR! 2015 Sessions
- level 2: session (name, start, end)
- level 3: room (name, seats)
- level 4: presentation (speaker, presentation name)

```
library(data.tree)


useR <- Node$new("UseR! Conference 2015")
  session1 <- useR$AddChild("Session 1")
    aalborghallen <- session1$AddChild("Aalborghallen")
    aalborghallen$seats <- 790
      p1 <- aalborghallen$AddChild("Federico Marini")
      p2 <- aalborghallen$AddChild("Jonathan Clayden")
      #... etc.
    gaestsalen <- session1$AddChild("Gaestsalen")
    gaestsalen$seats <- 149
      p1 <- gaestsalen$AddChild("Costas Varsos")
      #... etc.
```

```
  p2 <- aalborgharen$AddChild("Jonathan Clayden")
  #... etc.
gaestsalen <- session1$AddChild("Gaestsalen")
gaestsalen$seats <- 149
  p1 <- gaestsalen$AddChild("Costas Varsos")
  #... etc.
```

Normally, you would do this in an algorithm.

See:

```
vignette("ID3")
```

OO-Style method calling

```
useR$ToList()
```

But you can still use “classical” R generics if you prefer. The following is equivalent:

```
as.list(useR)
```

Reference Semantics

Nodes exhibit reference semantics:

```
session1$day <- "Wed"
useR$Find("Session 1")$day
```

```
## [1] "Wed"
```

Equivalent operation with lists doesn't work:

```
useRlist <- list()
  session1list <- list()
  useRlist$session1 <- session1list
  session1list$day <- "Wed"
useRlist$session1$day
```

```
## NULL
```

Basic actives on nodes

```
useR$isLeaf
```

```
## [1] FALSE
```

```
useR$isRoot
```

```
## [1] TRUE
```

```
useR$depth
```

```
## [1] 4
```

```
useR$depth
```

```
## [1] 4
```

```
useR$count
```

```
## [1] 6
```

```
useR$totalCount
```

```
## [1] 164
```

```
useR$leafCount
```

```
## [1] 127
```

Navigation

Navigate to a specific node:

```
glur <- useR$Find("Session 2",
                  "Det lille Teater",
                  "Christoph Glur")
```

…and navigate relatively to it:

```
glur$parent$name
```

```
## [1] "Det lille Teater"
```

```
glur$root$name
```

```
## [1] "useR"
```

Attributes

Find attributes:

```
glur$fields
```

```
## [1] "presentation" "speaker"
```

```
glur$fieldsAll
```

```
## [1] "presentation" "speaker"
```

… and access them:

```
glur$presentation
```

```
## [1] "A better way to manage hierarchical data"
```

```
glur$presentation
```

```
## [1] "A better way to manage hierarchical data"
```

Add new ones if you wish:

```
glur$package <- 'data.tree'
glur$fields
```

```
## [1] "package"      "presentation" "speaker"
```

```
glur$root$fieldsAll
```

```
## [1] "end"          "session"      "sessionName"
## [4] "start"        "room"         "seats"
## [7] "presentation" "speaker"      "package"
```

Create a deep copy

```
useR2 <- useR$Clone()
```

Prune

```
myPrune <- function(x) {
  return(x$level!=3 || x$room == 'Radiosalen')
}

useR2$Prune(pruneFun = myPrune)
useR2$totalCount
```

```
## [1] 39
```

```
head(useR2$ToDataFrame(), n = 10)
```

```
useR2$totalCount
```

```
## [1] 39
```

```
head(useR2$ToDataFrame(), n = 10)
```

```
##                                levelName
## 1  useR
## 2    ¦--Session 1
## 3    ¦   °--Radiosalen
## 4    ¦       ¦--Kasper D. Hansen
## 5    ¦       ¦--Karl Millar
## 6    ¦       ¦--Matt P. Dziubinski
## 7    ¦       °--Dan Putler
## 8    ¦--Session 2
## 9    ¦   °--Radiosalen
## 10   ¦       ¦--E. James Harner
```

```
## 7    ¦       °--Dan Putler
## 8    ¦--Session 2
## 9    ¦   °--Radiosalen
## 10   ¦       ¦--E. James Harner
```

Sort

```
useR$Sort("speaker")
print(useR$Find("Session 2", "Det lille Teater"))
```

```
##                                levelName
## 1 Det lille Teater
## 2  ¦--Christoph Glur
## 3  ¦--Filip Schouwenaars
## 4  ¦--Hadley Wickham
## 5  ¦--Indrajit Roy, Michael Lawrence
## 6  °--Tony Fischetti
```

data.frame

Load in the tabular data from csv:

```
library(data.tree)
useRdf <- read.csv('../data/useR15.csv', stringsAsFactors = FALSE, encoding = "UTF-8")
head(useRdf[,c("session", "start", "sessionName", "room", "seats", "speaker")])
```

```
##     session             start    sessionName          room seats              speaker
## 1 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790      Federico Marini
## 2 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790     Jonathan Clayden
## 3 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790  Carel F. W. Peeters
## 4 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790 Henrik Tobias Madsen
## 5 Session 1 01.07.2015 10:30        Ecology    Gæstesalen   149        Costas Varsos
## 6 Session 1 01.07.2015 10:30        Ecology    Gæstesalen   149       David L Miller
```

Convert it to a data.tree

```
#1. the pathString defines the hierarchy:
useRdf$pathString <- paste("useR", useRdf$session, useRdf$room, useRdf$speaker, sep="|")

#2. advanced and optional: define which attributes go to which level in the tree
#By default, they go to the leaf
cols <- list(NULL,                                          #root
             c('session', 'start', 'end', 'sessionName'), #session
             c('room', 'seats')                           #room
             )

#3. convert to tree
```

```
## 2 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790     Jonathan Clayden
## 3 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790  Carel F. W. Peeters
## 4 Session 1 01.07.2015 10:30 Kaleidoscope 1 Aalborghallen   790 Henrik Tobias Madsen
## 5 Session 1 01.07.2015 10:30        Ecology    Gæstesalen   149         Costas Varsos
## 6 Session 1 01.07.2015 10:30        Ecology    Gæstesalen   149        David L Miller
```

Convert it to a data.tree

```
#1. the pathString defines the hierarchy:
useRdf$pathString <- paste("useR", useRdf$session, useRdf$room, useRdf$speaker, sep="|")

#2. advanced and optional: define which attributes go to which level in the tree
#By default, they go to the leaf
cols <- list(NULL,                                          #root
             c('session', 'start', 'end', 'sessionName'), #session
             c('room', 'seats')                           #room
             )

#3. convert to tree
useRtree <- as.Node(useRdf, pathDelimiter = "|", colLevels = cols)


print(useRtree,
      "start", "seats",
      filterFun = function(x) x$level < 4)
```

```
##                   levelName             start seats
## 1  useR                                          NA
## 2    ¦--Session 1            01.07.2015 10:30    NA
## 3    ¦   ¦--Aalborghallen                        790
## 4    ¦   ¦--Gæstesalen                          149
## 5    ¦   ¦--Musiksalen                          160
## 6    ¦   ¦--Det lille Teater                    224
## 7    ¦   °--Radiosalen                          216
```

```r
print(useRtree,
      "start", "seats",
      filterFun = function(x) x$level < 4)
```

```
##                   levelName             start seats
## 1  useR                                          NA
## 2   ¦--Session 1             01.07.2015 10:30    NA
## 3   ¦   ¦--Aalborghallen                        790
## 4   ¦   ¦--Gæstesalen                          149
## 5   ¦   ¦--Musiksalen                          160
## 6   ¦   ¦--Det lille Teater                    224
## 7   ¦   °--Radiosalen                         216
## 8   ¦--Session 2             01.07.2015 13:30    NA
## 9   ¦   ¦--Aalborghallen                        790
## 10  ¦   ¦--Gæstesalen                          149
## 11  ¦   ¦--Musiksalen                          160
## 12  ¦   ¦--Det lille Teater                    224
## 13  ¦   °--Radiosalen                         216
## 14  ¦--Session 3             01.07.2015 16:00    NA
## 15  ¦   ¦--Aalborghallen                        790
## 16  ¦   ¦--Gæstesalen                          149
## 17  ¦   ¦--Musiksalen                          160
## 18  ¦   ¦--Det lille Teater                    224
## 19  ¦   °--Radiosalen                         216
## 20  ¦--Session 4             02.07.2015 10:30    NA
## 21  ¦   ¦--Radiosalen                         216
## 22  ¦   ¦--Aalborghallen                        790
## 23  ¦   ¦--Gæstesalen                          149
## 24  ¦   ¦--Musiksalen                          160
## 25  ¦   °--Det lille Teater                    224
## 26  ¦--Session 5             02.07.2015 13:00    NA
## 27  ¦   ¦--Aalborghallen                        790
```

traversal

```
session2 <- useR$Find("Session 2")
session2$Set(preo = 1:session2$totalCount)
print(session2, "preo")
```

```
##                                    levelName preo
## 1  Session 2                                    1
## 2    ¦--Aalborghallen                           2
## 3    ¦   ¦--Przemyslaw Biecek                   3
## 4    ¦   ¦--Joseph B. Rickert                   4
## 5    ¦   ¦--Richard M. Heiberger                5
## 6    ¦   °--Rasmus Bååth                        6
## 7    ¦--Gæstesalen                              7
## 8    ¦   ¦--Johannes Breidenbach                8
## 9    ¦   ¦--Ivan Kasanický                      9
## 10   ¦   ¦--Jakob W. Messner                   10
## 11   ¦   °--Helle Sørensen                     11
## 12   ¦--Musiksalen                             12
## 13   ¦   ¦--Anders Ellern Bilgrau               13
```

Compare this with post-order and level (breadth first):

```
session2$Set(posto = 1:session2$totalCount,
             traversal = "post-order")
session2$Set(lev = 1:session2$totalCount,
             traversal = "level")
print(session2, "preo", "posto", "lev")
```

```
##                                levelName preo posto lev
## 1  Session 2                                1    28   1
## 2    ¦--Aalborghallen                       2     5   2
## 3    ¦   ¦--Przemyslaw Biecek               3     1   7
## 4    ¦   ¦--Joseph B. Rickert               4     2   8
## 5    ¦   ¦--Richard M. Heiberger            5     3   9
## 6    ¦   °--Rasmus Bååth                    6     4  10
## 7    ¦--Gæstesalen                          7    10   3
## 8    ¦   ¦--Johannes Breidenbach            8     6  11
## 9    ¦   ¦--Ivan Kasanický                  9     7  12
## 10   ¦   ¦--Jakob W. Messner               10     8  13
## 11   ¦   °--Helle Sørensen                 11     9  14
## 12   ¦--Musiksalen                         12    15   4
```

You can also collect data instead of setting data:

```
datamanagement <- session2$Find("Det lille Teater")
datamanagement$Get("name")
```

```
## [1] "Det lille Teater"
## [2] "Filip Schouwenaars"
## [3] "Tony Fischetti"
## [4] "Hadley Wickham"
## [5] "Christoph Glur"
## [6] "Indrajit Roy, Michael Lawrence"
```

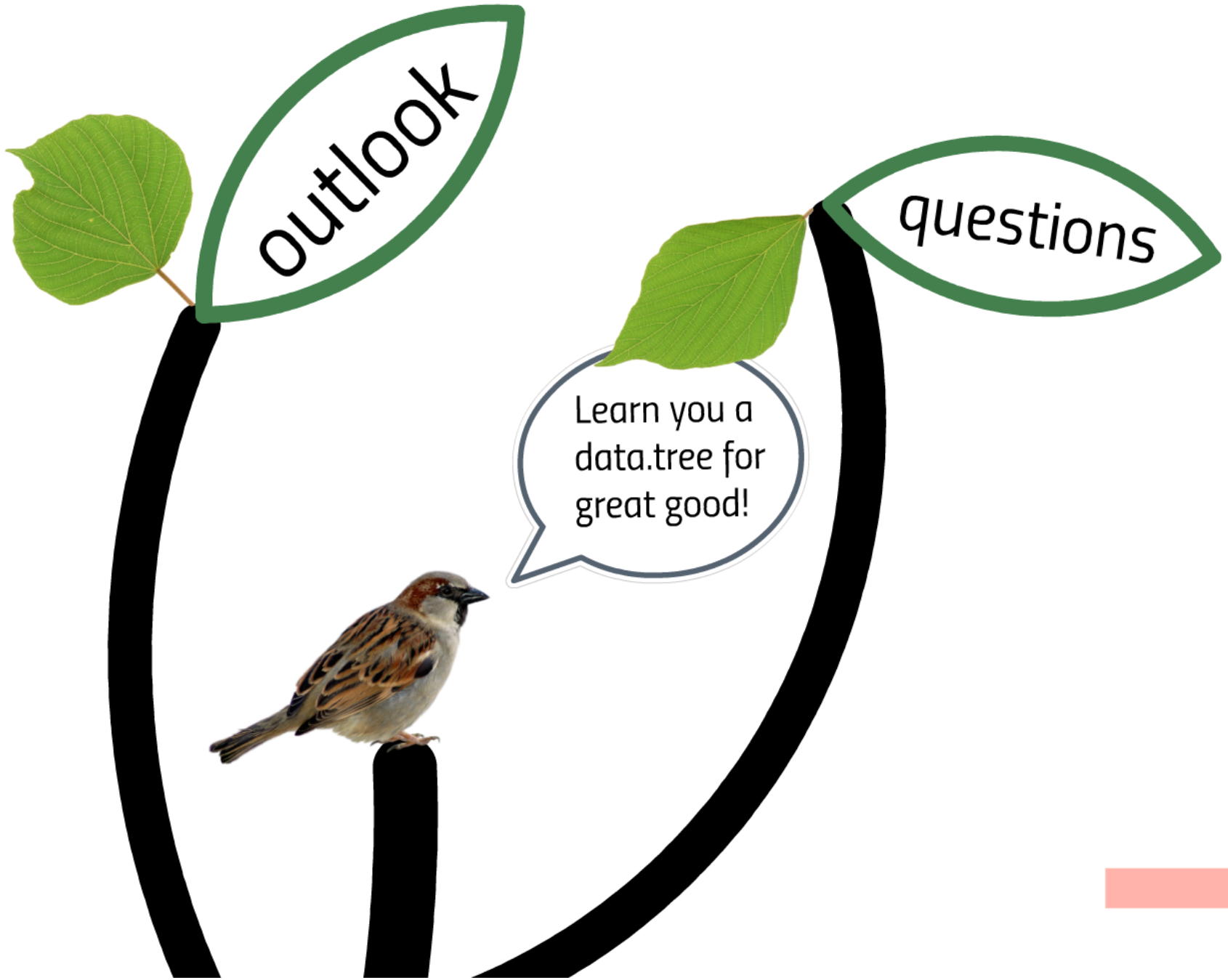
Works also on functions:

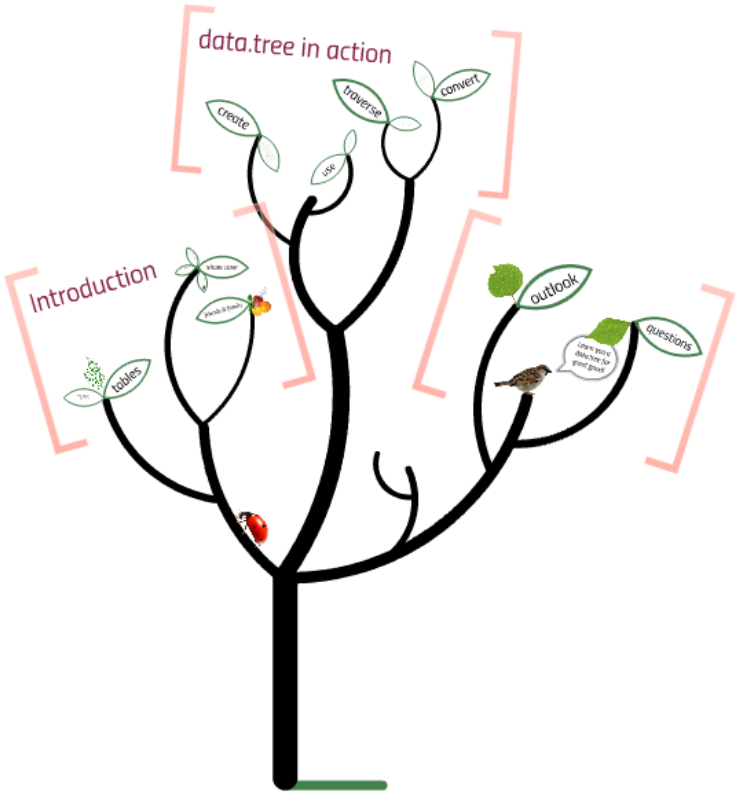
```
toUpperName <- function(node) {
  toupper(node$name)
}

datamanagement$Get(toUpperName)
```

```
## [1] "DET LILLE TEATER"
## [2] "FILIP SCHOUWENAARS"
## [3] "TONY FISCHETTI"
## [4] "HADLEY WICKHAM"
## [5] "CHRISTOPH GLUR"
## [6] "INDRAJIT ROY, MICHAEL LAWRENCE"
```

And on actives, with pruning and filtering, and and and:

```
useR$Get("totalCount",
         traversal = "level",
         pruneFun = function(x) {
           x$level != 3 || x$room == "Det lile Teater"
         },
         filterFun = function(x) !x$isLeaf)
```


outlook
questions
Learn you a data.tree for great good!



data.tree

A better way to manage hierarchical data

useR! 2015
http://github.com/gluc/useR15

christoph.glur@ipub.com