influence\_diagram

# influence\_diagram

source('functions.R')

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':  
  
 filter, lag

The following objects are masked from 'package:base':  
  
 intersect, setdiff, setequal, union

Attaching package: 'rlang'

The following objects are masked from 'package:purrr':  
  
 %@%, flatten, flatten\_chr, flatten\_dbl, flatten\_int, flatten\_lgl,  
 flatten\_raw, invoke, splice

# Variable Elimination - Decision Networks

## Example (Weather Decision Network)

In the example we have two chance nodes, 1) weather, that represents the random variable of the weather on any day either has rain or no rain; 2) forecast, a random variable that depends on the weather and is a noisy signal of the weather. The forecast can be sunny, cloudy, or rainy.

### Weighted relations for the chance nodes factors

In the example, we have two chance nodes: 1) weather, which represents the random variable of the weather on any day, either rain or no rain; 2) forecast, a random variable that depends on the weather and is a noisy signal of the weather. The forecast can be sunny, cloudy, or rainy.

The following two weighted relations define the factored probabilistic model of the weather decision network. The first factor, c\_weather, gives the prior probability of weather: rain with a probability of 0.3 and no rain with a probability of 0.7. The second factor, c\_forecast, gives the forecast’s conditional probability given the weather’s state. If the weather is no rain, the forecast is sunny with a probability of 0.7, cloudy with a probability of 0.2, and rainy with a probability of 0.1. If the weather equals rain, the forecast is sunny with a probability of 0.15, cloudy with a probability of 0.25, and rainy with a probability of 0.6. The Weight attribute when eliminating a chance node represents a probability where the ‘FUN’ parameter equals ‘sum’ for summation. The Weight attribute when eliminating a decision node represents a utility function where the ‘FUN’ parameter equals ‘max’ for maximization.

When combining weighted relations we note the following:

* If the combined relations contain at least one relation for chance variables then the ‘FUN’ for the ‘comb’ function is a multiplication ‘\*’.
* If the combined relations are all representing utilities then the the ‘FUN’ for the ‘comb’ function is a multiplication ‘+’.

weather <- data.frame(  
 weather = c('no\_rain',"rain"),  
 Weight = c(0.7, 0.3),  
 stringsAsFactors = FALSE)  
forecast <- data.frame(  
 weather=c('no\_rain','no\_rain','no\_rain','rain','rain','rain'),  
 forecast=c('sunny','cloudy','rainy','sunny','cloudy','rainy'),  
 Weight=c(0.7,0.2,0.1,0.15,0.25,0.6),  
 stringsAsFactors = FALSE)

## Weighted relations for the decision node’s factor

utility <- data.frame(  
 weather = c('no\_rain','no\_rain','rain','rain'),  
 umbrella = c('take\_it','leave\_it','take\_it','leave\_it'),  
 Weight = c(20, 100, 70, 0),  
 stringsAsFactors = FALSE)

write.csv(weather,file = "weather.csv")  
write.csv(forecast,file = "forecast.csv")  
write.csv(utility,file = "utility.csv")

### Initialization

#### Create variable-relations association dictionary: ‘assoc1’

Create variable-relations association. Select the variable ordering for the nodes of the influence diagram: 1) forecast, 2) umbrella, 3) weather.

assoc1 <- assoc(list(forecast, utility, weather), c("forecast", "umbrella", "weather"))  
print(assoc1)

$forecast  
NULL  
  
$umbrella  
NULL  
  
$weather  
$weather[[1]]  
 weather forecast Weight  
1 no\_rain sunny 0.70  
2 no\_rain cloudy 0.20  
3 no\_rain rainy 0.10  
4 rain sunny 0.15  
5 rain cloudy 0.25  
6 rain rainy 0.60  
  
$weather[[2]]  
 weather umbrella Weight  
1 no\_rain take\_it 20  
2 no\_rain leave\_it 100  
3 rain take\_it 70  
4 rain leave\_it 0  
  
$weather[[3]]  
 weather Weight  
1 no\_rain 0.7  
2 rain 0.3

### Step 1 Eliminate chance node: ‘Weather’

#### Combination query: ‘q\_comb\_weather’

Combination query on list of relations associated with ‘weather’ in the assoc1 dictionary

q\_comb\_weather = comb\_list(assoc1$weather)  
print(q\_comb\_weather)

weather forecast umbrella Weight  
1 no\_rain sunny take\_it 9.80  
2 no\_rain sunny leave\_it 49.00  
3 no\_rain cloudy take\_it 2.80  
4 no\_rain cloudy leave\_it 14.00  
5 no\_rain rainy take\_it 1.40  
6 no\_rain rainy leave\_it 7.00  
7 rain sunny take\_it 3.15  
8 rain sunny leave\_it 0.00  
9 rain cloudy take\_it 5.25  
10 rain cloudy leave\_it 0.00  
11 rain rainy take\_it 12.60  
12 rain rainy leave\_it 0.00

write.csv(q\_comb\_weather,file = "q\_comb\_weather.csv")

#### Elimination qurey: ‘q\_elim\_weather’

Elimination query to eliminate ‘weather’ from the weighted relation factor induced from the composition query for ‘weather’

q\_elim\_weather <- elim(q\_comb\_weather,"weather")  
print(q\_elim\_weather)

forecast umbrella Weight  
cloudy\_leave\_it cloudy leave\_it 14.00  
cloudy\_take\_it cloudy take\_it 8.05  
rainy\_leave\_it rainy leave\_it 7.00  
rainy\_take\_it rainy take\_it 14.00  
sunny\_leave\_it sunny leave\_it 49.00  
sunny\_take\_it sunny take\_it 12.95

write.csv(q\_elim\_weather,file = "q\_elim\_weather.csv")

#### Mutate Variable-Relations association dictionary: ‘assoc2’

We use the function ‘add\_relation\_remove\_var’ to update the current dictionary; the input parameters are the previous dictionary ‘assoc1’ and the eliminated variable ‘weather’. The result is a mutation of the previous dictionary to the current dictionary ‘assoc2’ where the induced relation of the query ‘q\_elim\_weather’ is added and the the elimination variable ‘weather’ is removed.

assoc2 <- add\_relation\_remove\_var(assoc1, q\_elim\_weather, "weather")  
print(assoc2)

$forecast  
NULL  
  
$umbrella  
$umbrella[[1]]  
 forecast umbrella Weight  
cloudy\_leave\_it cloudy leave\_it 14.00  
cloudy\_take\_it cloudy take\_it 8.05  
rainy\_leave\_it rainy leave\_it 7.00  
rainy\_take\_it rainy take\_it 14.00  
sunny\_leave\_it sunny leave\_it 49.00  
sunny\_take\_it sunny take\_it 12.95

### Step 2 Eliminate decision node: ‘Umbrella’

Since there is only one weighted relation associated with the decision node ‘umbrella’ in the dictionary ‘assoc2’, we can bypass the combination query and just run the elimination query, q\_elim\_umbrella, using the ‘elim’ function with the input parameters being:

* the weighted relation associated with ‘umbrella’ which is ‘q\_elim\_weather’
* the variable to eliminate which is ‘umbrella’
* the ‘FUN’ parameter set to ‘max’ specifying the aggregation function on the Weight attribute to be a maximization for the Weight is representing a utility that we need to maximize

q\_elim\_umbrella <- elim(q\_elim\_weather, "umbrella", FUN = max)  
print(q\_elim\_umbrella)

forecast Weight  
cloudy cloudy 14  
rainy rainy 14  
sunny sunny 49

write.csv(q\_elim\_umbrella,file = "q\_elim\_umbrella.csv")

#### Mutate Variable-Relations association dictionary: ‘assoc3’

We use the function ‘add\_relation\_remove\_var’ to update the current dictionary; the input parameters are the previous dictionary ‘assoc2’ and the eliminated variable ‘umbrella’. The result is a mutation of the previous dictionary to the current dictionary ‘assoc3’ where the induced relation of the query ‘q\_elim\_umbrella’ is added and the the elimination variable ‘umbrella’ is removed.

assoc3 <- add\_relation\_remove\_var(assoc2, q\_elim\_umbrella, "umbrella")  
print(assoc3)

$forecast  
$forecast[[1]]  
 forecast Weight  
cloudy cloudy 14  
rainy rainy 14  
sunny sunny 49

### Step 3 Computing the expected utility for all policies

The ‘forecast’ node is the last node so we are ready to compute the maximum expected utility and associated decision policy. There is only one relation associated with the last node in the elimination ordering which is the ‘forecast’ node. There is no need to use the ‘comb\_list’ function.

To compute the expected utility for all situation-decision policy pairs where the situation refers to the value of the ‘forecast’ node and the decision refers to the value of the ‘umbrella’ node.

In this decision network the expected utility equals the inner-join of the relation associated with ‘forecast’ in the ‘assoc3’ dictionary and the relation associated with ‘umbrella’ in the ‘assoc2’ dictionary.

r\_forecast <- assoc3$forecast[[1]]  
r\_umbrella <- assoc2$umbrella[[1]]  
print(r\_forecast)

forecast Weight  
cloudy cloudy 14  
rainy rainy 14  
sunny sunny 49

print(r\_umbrella)

forecast umbrella Weight  
cloudy\_leave\_it cloudy leave\_it 14.00  
cloudy\_take\_it cloudy take\_it 8.05  
rainy\_leave\_it rainy leave\_it 7.00  
rainy\_take\_it rainy take\_it 14.00  
sunny\_leave\_it sunny leave\_it 49.00  
sunny\_take\_it sunny take\_it 12.95

To perform the inner-join we will use the inner-join operator imported from the ‘dplry’ package.

library(dplyr)  
#  
# Perform inner join   
policy\_EU <- dplyr::inner\_join(r\_forecast, r\_umbrella, by = c("forecast","Weight"))  
  
  
print(policy\_EU)

forecast Weight umbrella  
1 cloudy 14 leave\_it  
2 rainy 14 take\_it  
3 sunny 49 leave\_it

write.csv(policy\_EU,"policy\_EU.csv")

### Step 6 Maximum Expected Utility

MEU <- sum(policy\_EU$Weight)  
print(MEU)

[1] 77