PROMETHEE Package for R

Giuliano Resce, Menelaos Tasiou, Francesco Vidoli

24th October 2018

1. Introduction

In this vignette, we describe the way to format the data that will be later evaluated based on the PROMETHEE I & II methods. In what follows, we describe the inputs needed, along with a brief but concise example for each input. We refer the reader interested in a more in-depth explanation of these methods to the seminal work of Brans and Vincke (1985), or for a newer conclusive version to Brans and Mareschal (2005), while for a literature review of the wide variety of applications in existence, one might look into Behzadian et al. (2010).

2. Step-by-step analysis of the file format (with an example)

Each input may be held in a separate or a single file, yet they have to be individually loaded and saved as such. In this example we have each input held in a separate sheet within a single excel file. In what follows, we discuss a simple example concerning buying a house, and how running the PROMETHEE I & II methods is accomplished with this package. We should note that this is a case-sensitive package, meaning that you should use these exact names (i.e. of inputs) in your analysis (namely *data, PreferenceF, PreferenceT, IndifferenceT, Weights, Min_Max, S_Gauss* - we discuss each and one of these in detail in the following subsections).

2.1 Data (Alternatives & Criteria)

Let us first load the data. Suppose that we are interested in buying a house. We are in-between four options (i.e. alternatives), that will be evaluated based on four criteria, namely *distance to work, price, number of bedrooms* and *age*. These are loaded and illustrated below.

```
data <-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/1_data.csv")
head(data)
```

```
##         Location Distance.to.work  Price Bedrooms Age
## 1 Hayling Island               10 250000        3  20
## 2    Southampton               25 270000        4  10
## 3       Southsea                4 320000        2  15
## 4 Gunwharf Quays                2 350000        2   7
```

One may extract the dataset (i.e. neglecting the alternatives' names, purely focusing on the evaluation matrix) in the following way

```
# Extracting the information related to the evaluation matrix
dataset <- data[,-c(1)]
head(dataset)
```

```
##   Distance.to.work  Price Bedrooms Age
## 1               10 250000        3  20
## 2               25 270000        4  10
## 3                4 320000        2  15
## 4                2 350000        2   7
```

2.2 Preferences

Having loaded the data (alternatives and criteria) in step 1, one has to declare the preferences, based on which the alternatives will be evaluated. These involve setting a preference function, and the preference/indifference thresholds accordingly. For more information about each, we refer the reader to the references at the bottom of this vignette.

2.2a Preference function

This package supports the *Level, Linear, V-shape* and *Gaussian* functions, however future editions will also support the *Usual* and *U-shape* functions. The preference functions need to be explicitly stated for each alternative in each criterion. Please note that these functions are case-sensitive, so they need to be set correctly in the excel file, otherwise the package will fail to work.

For instance, in the following, we load our excel file which considers the *Gaussian* function for the distance to work criterion, the *Linear* function for the price criterion, the *V-shape* function for the no. of bedrooms criterion and the *Level* function for the age criterion.

```
# Loading the matrix of Preference Functions (nested in an as.matrix command)
PreferenceF<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/2_pref.csv")
head(PreferenceF)
```

```
##   Distance.to.work  Price Bedrooms   Age
## 1         Gaussian Linear  V-shape Level
## 2         Gaussian Linear  V-shape Level
## 3         Gaussian Linear  V-shape Level
## 4         Gaussian Linear  V-shape Level
```

2.2b Preference Threshold

After declaring the preference function, the decision-maker shall set the preference thresholds, again, for each criterion and for each alternative. It basically states the difference (under a specific function) between alternatives in each criterion that is of utter significance for the evaluation. See example below.

```
# Loading the matrix of preference thresholds (nested in a data.matrix command)
PreferenceT<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/3_pret.csv")
head(PreferenceT)
```

```
##   Distance.to.work Price Bedrooms Age
## 1                2 50000        2   5
## 2                2 50000        2   5
## 3                2 50000        2   5
## 4                2 50000        2   5
```

2.2c Indifference Threshold

The indifference threshold states the exact opposite; that is, the difference between two alternatives (under a specific function) that deens the comparison between alternatives on a specific criterion insignificant. The decision-maker shall set the indifference thresholds, again, for each criterion and for each alternative. See example below.

```
# Loading the matrix of indifference thresholds (nested in a data.matrix command)
IndifferenceT<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/4_indt.csv")
head(IndifferenceT)
```

```
##   Distance.to.work Price Bedrooms Age
## 1                1 10000        0   2
## 2                1 10000        0   2
## 3                1 10000        0   2
## 4                1 10000        0   2
```

2.2d Gauss preference threshold

This step applies irrespectively of the decision-maker's preference to include this function or not. For instance, in this case example we put forward the hypothesis that the "distance to work" criterion is evaluated based on the Gauss_criterion, so we have to declare the "s" value inherent in this preference function. Other criteria not involving this function take the value 0. See example below.

Note: If your particular example does not involve this preference function, it should still be loaded but 0 values could be filled in each criterion for each alternative accordingly.

```
# Loading the matrix of Gauss Preferences (nested in a data.matrix command)
S_Gauss<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/7_gauss.csv")
head(S_Gauss)
```

```
##   Distance.to.work Price Bedrooms Age
## 1                2     0        0   0
## 2                2     0        0   0
## 3                2     0        0   0
## 4                2     0        0   0
```

2.3 Weights

The weights reflect the importance of each criterion and they typically range between 0 and 1 (with the sum of all weights being 1). In this case scenario, we suppose that the decision-maker is equally interested in all criteria, so she does not discriminate between them, eventually giving each criterion a weight that equals 1/n (where n the number of criteria). Given that we have four criteria, and the decision-maker is equally interested in all of them, each weighs 1/4 thus 25%.

```
# Loading the matrix of weights (nested in a data.matrix command)
Weights<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/5_weig.csv")
head(Weights)
```

```
##   Distance.to.work Price Bedrooms  Age
## 1             0.25  0.25     0.25 0.25
## 2             0.25  0.25     0.25 0.25
## 3             0.25  0.25     0.25 0.25
## 4             0.25  0.25     0.25 0.25
```

2.4 Direction of criteria

This is the fourth and final step in the inputs required from the decision-maker; the direction of criterion. This basically states whether a criterion is supposed to be *minimized* (min) or *maximized* (max) respectively. For instance, the criteria *distance to work, price* and *age* should be minimized, as lower values in this criteria denote a better performance in the evaluation, whereas the *number of bedrooms* criterion should be maximized instead, as the more bedrooms a house has the better for the taste of the decision-maker. The string values attached to each criterion are case-sensitive, so the criteria should involve a "min" or "max" string for each alternative as the example below:

```
# Loading the matrix of directions (nested in an as.matrix command)
Min_Max<-read.csv("https://fvidoli.weebly.com/uploads/2/3/0/8/23088460/6_mima.csv")
head(Min_Max)
```

```
##   Distance.to.work Price Bedrooms Age
## 1              min   min      max min
## 2              min   min      max min
## 3              min   min      max min
```

```
## 4              min   min      max min
```

3. Evaluation phase

Once the previous file-formatting step is done, the global environment should be loaded with the data, preference function, preference and indifference thresholds, the weights and the direction of criteria (i.e. Min/Max) and the Gauss Preference. These are named as *data, PreferenceF, PreferenceT, IndifferenceT, Weights, Min_Max, S_Gauss* accordingly. See example below:

To call the profethee function, one can simply run the following lines:

```
library("PROMETHEE")
PF=PROMETHEE(dataset,PreferenceF,PreferenceT,IndifferenceT,Weights,Min_Max,S_Gauss)
```

that returns a list of the outputs (outranking/non-outranking matrices, Unicriterion flows and the PROMETHEE I & II scores [flows, phi])

In this case example, these are found by calling:

Outranking Matrix

```
PF$Outranking
```

```
##           [,1]   [,2]  [,3]  [,4]
## [1,] 0.2500000 0.5625 0.250 0.000
## [2,] 0.0000000 0.5000 0.625 0.375
## [3,] 0.4972228 0.1250 0.000 0.125
## [4,] 0.5982835 0.0000 0.000 0.625
```

Non-Outranking Matrix

```
PF$Nonoutranking
```

```
##            [,1]   [,2]  [,3]  [,4]
## [1,] 0.49713889 0.0000 0.125 0.625
## [2,] 0.75000000 0.0625 0.000 0.125
## [3,] 0.09836734 0.5000 0.375 0.375
## [4,] 0.00000000 0.6250 0.375 0.000
```

Uni-criterion Net Flows

```
PF$UnicriterionNetFlows
```

```
##            [,1]    [,2]   [,3]   [,4]
## [1,] -0.2471389  0.5625  0.125 -0.625
## [2,] -0.7500000  0.4375  0.625  0.250
## [3,]  0.3988554 -0.3750 -0.375 -0.250
## [4,]  0.5982835 -0.6250 -0.375  0.625
```

PROMETHEE I (Phi+ and Phi-)

```
PF$PROMETHEE1
```

```
##            [,1]       [,2]
## [1,] 0.08854167 0.10392824
## [2,] 0.12500000 0.07812500
## [3,] 0.06226856 0.11236394
## [4,] 0.10194029 0.08333333
```

PROMETHEE II (Phi-net)

```
PF$PROMETHEE2
```

```
##             [,1]
## [1,] -0.01538657
## [2,]  0.04687500
## [3,] -0.05009538
## [4,]  0.01860696
```

So, the verdict from, say, *PROMETHEE II* is that, given the preferences expressed above, location 2 is the most desirable, followed by location 4, 1 and 3 in that exact order.

References

Behzadian, M., Kazemzadeh, R. B., Albadvi, A., & Aghdasi, M. (2010). PROMETHEE: A comprehensive literature review on methodologies and applications. European journal of Operational research, 200(1), 198-215.

Brans, J. P., & Mareschal, B. (2005). PROMETHEE methods. In Multiple criteria decision analysis: state of the art surveys (pp. 163-186). Springer, New York, NY.

Brans, J. P., & Vincke, P. (1985). Note-A Preference Ranking Organisation Method: (The PROMETHEE Method for Multiple Criteria Decision-Making). Management science, 31(6), 647-656.