Package 'memoryROC'

Jörn Alexander Quent

6 September 2016

Contents

General information

Version: 1.1

License: GPL-2

URL: https://github.com/JAQuent/memoryROC

Contact: alexander.quent@rub.de

Introduction

This package was written to analyse recognition memory performance and estimate the Dual Process Signal Detection (DPSD) parameters within R. Additional functions might be added, once I need them. This manual is supposed to provide a short description and explanation how to use the functions. If you have corrections and/or questions, feel free to contact me.

Functions

sampleData

The sample data is taken from a pilot recognition experiment using the remember/know procedure. In this experiment, this suject was asked to rate the recognition confidence from 1 (sure new) to 5 (sure old) or recollected (= 6). The data frame is ***sampleData*** contains the variables ***confidenceRatings***, which contains confidence ratings, and ***oldNew***, which contains information whether a stimuli had been studied (i.e. old, = 1) or had not been studied (i.e. new, = 0).

```
head(sampleData)
```

```
##   confidenceRatings oldNew
## 1                 1      1
## 2                 6      1
## 3                 1      0
## 4                 6      1
## 5                 2      0
## 6                 2      0
```

cumRates

This function allows you to extract cumulative hit and false alarm rates for further memory ROC analysis. In the example above, the false alarm rate is given by the probability that a stimulus was rated as 6 (i.e. sure the stimulus was old/remember) under the condition that it was new/not-studied. The first value of the hit rate is given by the probability that a stimulus was rated as 6 (i.e. sure the stimulus was old/remember) under the condition that the stimulus was indeed old. The next two values are obtained by calculating the probability that a stimulus was rated with 6 or 5 under the condition that the stimulus was new for the false-alarm rate and under the condition that the stimulus was old for the hit rate. This is done for all confidence levels.

Example usage:

```
responseScale <- 6:1
rates <- cumRates(responseScale, sampleData$confidenceRatings, sampleData$oldNew)
rates
```

```
##   falseAlarm       hit
## 1 0.03333333 0.5666667
## 2 0.10000000 0.6833333
## 3 0.16666667 0.7666667
## 4 0.40000000 0.8666667
## 5 0.76666667 0.9666667
```

fitDPSD

This function allows to estimate recollection and familiarity by fitting data to the DPSD model. The optimization is attempted by minimizing the total squared difference between observed and predicted hit and false alarm rates. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm from the function ***optim {stats}***. The function first uses standard start values and then random values in order to find the set of parameters, which fit the data best by returning the values with the lowest total squared difference. A high number of iterations is necessary to get stable estimates.

Example usage:

```
fitDPSD(rates$falseAlarm, rates$hit)
```

```
## $recollection
## [1] 0.4895526
##
## $familiarity
## [1] 0.9709316
```

DPSD

This function combindes the functions ***cumRates*** and ***fitDPSD*** for easier usage.

Example usage:

```
responseScale <- 6:1
DPSD(responseScale, sampleData$confidenceRatings, sampleData$oldNew)
```

```
## $recollection
## [1] 0.4895474
##
## $familiarity
## [1] 0.9709581
```

rememberKnow

This function allows to estimate recollection and familiarity using Remember/Know procedure. In this procedure, recollection is given by the probability that an old/studied item were given a remember response, while familiarity is given by the probility that an old/studied item were given a know response divived by the probility that an old/studied item were not given a remember response.

$$recollection = P(remember)$$

$$familiarity = P(know)/(1 - P(remember))$$

Example usage:

In the variable ***confidenceRatings*** the number 6 represents remember responses, while 5 & 4 represent know responses.

```
rememberLevels <- 6
knowLevels     <- c(5, 4)
rememberKnow(rememberLevels, knowLevels,  sampleData$confidenceRatings,  sampleData$oldNew)
```

```
## $recollection
## [1] 0.5666667
##
## $familiarity
## [1] 0.4615385
```

rocAUC

This function calculates the AUC by summing the tri- and rectangles, which can be made of the ROC points. If missing, the y-intercept is added by linear interpolation. The last point, where both false Alarm rates reach 1, is also added.

```
rememberLevels <- 6
rates          <- cumRates(responseScale, sampleData$confidenceRatings, sampleData$oldNew)
rocAUC(rates$falseAlarm, rates$hit)
```

```
## [1] 0.8640278
```

dPrime

This function caculates the dimensionless statistic d-prime (d'). To do this, the percentage values are converted to z-values with the help of inverse cumulative distribution function (CDF) of the Gaussian distribution with a mean of 0 and a standard deviation of 1. Or as a equation:

$$d' = z(hit\ rate) - z(false\ alarm\ rate)$$

where

$$z(p), p \in [0, 1]$$

is the inverse Gaussian CDF.

Example:

```
dPrime(c(6,5,4), sampleData$confidenceRatings, sampleData$oldNew)
```

```
## [1] 1.455827
```

returnFittedROC

This function allows you to get the corresponding false alarm and hit rates for a given set of recollection and familiarity assuming that the variance of the old item distirbution is 1. This is helpful to compare raw hit and false alarm rates with fitted ones.

Example:

```
recollection <- 0.4895505
familiarity  <- 0.9709519
fittedRates <- returnFittedROC(recollection, familiarity)
head(fittedRates)
```

```
##   falseAlarm       hit
## 1      0.001 0.4982452
## 2      0.011 0.5372852
## 3      0.021 0.5630496
## 4      0.031 0.5841376
## 5      0.041 0.6024469
## 6      0.051 0.6188243
```

License

memoryROC A package to analyse recognition memory data within R.