

Package ‘Rbonaut2’

December 13, 2016

Type Package

Title CLIP2's Rbonaut

Version 0.8

Date 2016-11-07

Author Cavorit

Maintainer Harald Fiedler <harald.fiedler@cavorit.de>

Depends data.table, RPostgreSQL, sfsmisc

Description The CLIP2-Version of CLIP1's Rbonaut-Package

License This package is private and internal of Cavorit Consulting GmbH

LazyData TRUE

RoxygenNote 5.0.1

R topics documented:

Rbonaut2-package	2
adaptiv.BL16.fullRandom.modelPredict	2
adaptiv.BL16.fullRandom.modelTransform	4
adaptiv.BL16.modelPredict	6
adaptiv.BL16.modelTransform	8
askDB	10
augmentRAW	11
calcFiedler2016a	11
detectItemID	12
detectItemIDLIVE	13
detectItemResponse	13
detectRotatedItems	14
erstelleRaschMatrixSkeleton	15
fillRaschMatrixSkeleton	16
getAdrWAlsListe	16
getFirstAdrW	17
getHW	18

getNachname	19
getSessionTimeStamp	19
getVorname	20
gibZahlFuehrendeNullen	21
implodeRaschMatrix4Quality	22
isMultiTarget	22
istFormatNachnameKommaVorname	23
itemID2Params	24
NormTree	24
NormTree6.16	25
playedAngle	25
plotFBN	26
plotSeaShell	26
readAUGMENTED	27
readItemBank	28
readRAW	29
writeAUGMENTED	29
writeRAW	30
Index	31

Rbonaut2-package	<i>Rbonaut2</i>
------------------	-----------------

Description

CLIP2-Paket

Author(s)

Harald Fiedler (c) Cavorit

adaptiv.BL16.fullRandom.modelPredict
<i>adaptiv.BL16.fullRandom.modelPredict</i>

Description

modelPredict() fuer yhat-Modell eines voll randomisierten Samplers des BL32-Testraums mit Stop nach 16 Bällen

Usage

adaptiv.BL16.fullRandom.modelPredict(AnfrageDF)

Arguments

AnfrageDF	<p>data.frame mit den Spalten:</p> <ul style="list-style-type: none"> • TestID: den Namen des Testformats. Beispielsweise: 'BL32' • idS: die aktuelle Session-ID, beispielsweise: '002b6573-cf12-436d-bccd-0856b0bb0a25' • idP: die Player-ID des jeweiligen Probanden, beispielsweise: 'fe553db4-bbde-43dd-a6a0-804b9e46c57' • NamePlayer: Klarnamen des Spielers, beispielsweise: 'Mustermann, Tim' • Birthday: Geburtsdatum des Spielers, beispielsweise: '2002-07-17' • Team: Bezeichnung des Teams, z.B. "U17" • SessionStart: der Zeitstempel für den Sessionsstart als String im Format "JJJJ-MM-TT HH:MM:SS", beispielsweise: '2014-03-22 13:42:03' • adrB: eine Liste mit den Adressen der Ballkanonen, in der Reihenfolge ihrer Aktivierung, beispielsweise: [10, 45, 28] oder bei Sessionstart eine leere Liste '[]'. • adrW: eine Liste von Listen mit den aufleuchtenden Zielfeldern. Beispielsweise: '[[[4, 6], [21, 22], [50, 18]]]' oder bei Sessionstart '[[[]]' • adrCol: Eine Liste von Strings mit der Farbe der Zielfelder aus [[adrW]]. Die Farben werden als String aus rgb-Werten (max=255) übergeben. Das satte Blau in Farben von Hoffenheim ist rgb(1,95,169, maxColorValue=255). Farbe B ist die Gegnerfarbe rgb(200,100,0, maxColorValue=255). Beispielsweise: "[[[[\"1,95,169\"],[\"200,100,0\"]],[[\"1,95,169\"],[\"200,100,0\"]],[[\"200,100,0\"],[\"1,95,169\"]]]]" oder bei Sessionstart '[[[]]' • adrOut: eine Liste mit den Adressen, wo der Ball tatsächlich raus ist. Beispielsweise: '[4, 23, 40]' oder bei Sessionstart '[]' • FBt: eine Liste mit den Angaben über die Zeitdauer zwischen Ballkanoneneinwurf und Lichtschrankensignal beim Rausschießen des Balls (in Millisekunden), beispielsweise: '[2140, 2600, 8600]' oder bei Sessionstart '[]'
-----------	--

Details

Es handelt sich hier um die modelPredict-Funktion für yhat-Architektur. Bei dem Dienst handelt es sich um einen reinen Item-Sampler. Solange die Session-History weniger als 16 Bälle aufweist, wird aus dem Item-Raum der BL32 ein Item gewählt.

Value

Ein JSON-gültiger String mit den folgenden Informationen:

- TicketID ein stochastischer Identifikator für die REST-Request, z.B. "JTMfOgfTEiq6ZMxb"
- GameOver ein boolescher Wert der angibt, ob das Abbruchkriterium für die Testung erreicht wurde
- NextB Eine liste mit Informationen über den nächsten Ball
 - ballmachine welche als nächstes aktiv ist
 - goal_target Zielfeld

- speed_left Geschwindigkeit der linken Walze
- speed_right Geschwindigkeit der rechten Walze
- vertical_angle Vertikaler Winkel der Ballmaschine
- shoot_delay Verzögerungszeit nach dem das Signal ertönt ist
- expiration_time Zeit in der das Zielfeld valid_color leuchten soll, dann mit expiration_color
- valid_color String für die Farbe '13,83,97'
- expiration_color String für die Farbe '200,100,0'
- Testergebnis Das Testergebnis, dass der Spieler auf der latenten Kompetenzdimension erhält. (Noch nicht implementiert, daher vorläufig NA)
- Seashell_output_base64, Grafik im base64 im JSON format, wenn gameover. Um es im Browser zu öffnen: <!DOCTYPE html><html><body></body></html>

#' Example Output, jedes Anführungszeichen, außer erstes und letztes sind backslashed

```
[["TicketID":["YxubhQICsGdrXcGnhSZo"],"nextB":"ballmachine":[28],"goal_target":[72],"speed_left":[50],
"speed_right":[60],"vertical_angle":[4],"shot_delay":[800],"expiration_time":[2195],
"expiration_color":["200,100,0"],"Testergebnis":[null],"GameOver":[false],
"Seashell_output_base64":[null]]]
```

Author(s)

Harald Fiedler

adaptiv.BL16.fullRandom.modelTransform

adaptiv.BL16.fullRandom.modelTransform

Description

modelTransform() fuer yhat-Modell eines voll randomisierten Samplers des BL32-Testraums mit Stop nach 16 Bällen

Usage

```
adaptiv.BL16.fullRandom.modelTransform(AnfrageJSONstring)
```

Arguments

AnfrageJSONstring

String, der in R zu einer Liste umgewandelt werden kann mit folgenden Elementen:

- TestID: den Namen des Testformats. Beispielsweise: 'BL32'
- idS: die aktuelle Session-ID, beispielsweise: '002b6573-cf12-436d-bccd-0856b0bb0a25'
- idP: die Player-ID des jeweiligen Probanden, beispielsweise: 'fe553db4-bbde-43dd-a6a0-804b9e46c57'

- NamePlayer: Klarname des Spielers, beispielsweise: 'Mustermann, Tim'
- Birthday: Geburtsdatum des Spielers, beispielsweise: '2002-07-21'
- Team: Bezeichnung des Teams, z.B. "U17"
- SessionStart: der Zeitstempel für den Sessionsstart als String im Format "JJJJ-MM-TT HH:MM:SS", beispielsweise: '2014-03-22 13:42:03'
- adrB: eine Liste mit den Adressen der Ballkanonen, in der Reihenfolge ihrer Aktivierung, beispielsweise: [10, 45, 28] oder bei Sessionstart eine leere Liste '[]'.
- adrW: eine Liste von Listen mit den aufleuchtenden Zielfeldern. Beispielsweise: '[[4, 6], [21, 22], [50, 18]]' oder bei Sessionstart '[[[]]'
- adrCol: Eine Liste von Strings mit der Farbe der Zielfelder aus [[adrW]]. Die Farben werden als String aus rgb-Werten (max=255) übergeben. Das satte Blau in Farben von Hoffenheim ist rgb(1,95,169, maxColorValue=255). Farbe B ist die Gegnerfarbe rgb(200,100,0, maxColorValue=255). Beispielsweise: "[[["1,95,169"],["200,100,0"]],[["1,95,169"],["200,100,0"]],[["200,100,0"],["1,95,169"]]]" oder bei Sessionstart '[[[]]'
- adrOut: eine Liste mit den Adressen, wo der Ball tatsächlich raus ist. Beispielsweise: '[4, 23, 40]' oder bei Sessionstart '[]'
- FBt: eine Liste mit den Angaben über die Zeitdauer zwischen Ballkanoneneinwurf und Lichtschrankensignal beim Rausschießen des Balls (in Millisekunden), beispielsweise: '[2140, 2600, 8600]' oder bei Sessionstart '[]'

Hier ein Beispiel für eine gültige Anfrage für die Abfrage eines ersten Balls:

```
#' testJSONrequest <- '{
"TestID" : "BL32",
"idS" : "002b6573-cf12-436d-bccd-0856b0bb0a25",
"idP" : "fe553db4-bbde-43dd-a6a0-804b9e46c57",
"NamePlayer" : "Mustermann, Tim",
"Birthday" : "2002-07-21",
"Team" : "U14",
"SessionStart" : "2014-03-22 13:42:03",
"adrB" : [],
"adrW" : [],
"adrCol" : [],
"adrOut" : [],
"FBt" : []
}'
```

Für einen vierten Ball:

```
testJSONrequest <- '{
"TestID" : "BL32",
"idS" : "002b6573-cf12-436d-bccd-0856b0bb0a25",
"idP" : "fe553db4-bbde-43dd-a6a0-804b9e46c57",
"NamePlayer" : "Mustermann, Tim",
"Birthday" : "2002-07-21",
"Team" : "U14",
"SessionStart" : "2014-03-22 13:42:03",
```

```
"adrB" : [10, 45, 28],
"adrW" : [[4, 6], [21, 22], [50, 18]],
"adrCol" : [[["1,95,169"],["200,100,0"]],["1,95,169"],["200,100,0"]],["200,100,0"],["1,95,169"]],
"adrOut" : [4, 23, 40],
"FBt" : [2140, 2600, 8600]
}'
```

Details

Es handelt sich hier um die modelTransform-Funktion für yhat-Architektur. Bei dem Dienst handelt es sich um einen reinen Item-Sampler. Solange die Session-History weniger als 16 Bälle aufweist, wird aus dem Item-Raum der BL32 ein Item gewählt.

Value

data.frame mit den oben beschriebenen Spalten.

Author(s)

Harald Fiedler

adaptiv.BL16.modelPredict

adaptiv.BL16.modelPredict

Description

modelPredict() fuer yhat-Modell eines adaptiven Samplers nach 5 Bällen des BL32-Testraums mit Stop nach 16 Bällen. Basiert auf dem Skript BL32MultiTargetSimTest.r (Teilcode wurde kopiert) Der Output Ergebnis wurde nach dem adaptiv.BL16.fullRandom.modelPredict.R implementiert

Usage

```
adaptiv.BL16.modelPredict>AnfrageDF)
```

Arguments

AnfrageDF	data.frame mit den Spalten:
	<ul style="list-style-type: none"> • TestID: den Namen des Testformats. Beispielsweise: 'BL32' • idS: die aktuelle Session-ID, beispielsweise: '002b6573-cf12-436d-bccd-0856b0bb0a25' • idP: die Player-ID des jeweiligen Probanden, beispielsweise: 'fe553db4-bbde-43dd-a6a0-804b9e46c57' • NamePlayer: Klarname des Spielers, beispielsweise: 'Mustermann, Tim' • Birthday: Geburtsdatum des Spielers, beispielsweise: '2002-07-17' • Team: Bezeichnung des Teams, z.B. "U17"

- SessionStart: der Zeitstempel für den Sessionsstart als String im Format "JJJJ-MM-TT HH:MM:SS", beispielsweise: '2014-03-22 13:42:03'
- adrB: eine Liste mit den Adressen der Ballkanonen, in der Reihenfolge ihrer Aktivierung, beispielsweise: [10, 45, 28] oder bei Sessionstart eine leere Liste '[]'.
- adrW: eine Liste von Listen mit den aufleuchtenden Zielfeldern. Beispielsweise: '[[4, 6], [21, 22], [50, 18]]' oder bei Sessionstart '[]'
- adrCol: Eine Liste von Strings mit der Farbe der Zielfelder aus [[adrW]]. Die Farben werden als String aus rgb-Werten (max=255) übergeben. Das satte Blau in Farben von Hoffenheim ist rgb(1,95,169, maxColorValue=255). Farbe B ist die Gegnerfarbe rgb(200,100,0, maxColorValue=255). Beispielsweise: "[[\"1,95,169\"],[\"200,100,0\"]],[[\"1,95,169\"],[\"200,100,0\"]],[[\"200,100,0\"],[\"1,95,169\"]]]" oder bei Sessionstart '[]'
- adrOut: eine Liste mit den Adressen, wo der Ball tatsächlich raus ist. Beispielsweise: '[4, 23, 40]' oder bei Sessionstart '[]'
- FBt: eine Liste mit den Angaben über die Zeitdauer zwischen Ballkanoneneinwurf und Lichtschrankensignal beim Rausschießen des Balls (in Millisekunden), beispielsweise: '[2140, 2600, 8600]' oder bei Sessionstart '[]'

Details

Es handelt sich hier um die modelPredict-Funktion für yhat-Architektur. Bei dem Dienst handelt es sich um einen reinen Item-Sampler. Solange die Session-History weniger als 16 Bälle aufweist, wird aus dem Item-Raum der BL32 ein Item gewählt. Der erste Ball ist immer BL01 (adrB: 64, adrW: 68), die nächsten 4 Bälle werden ohne Zurücklegen gezogen. Falls ein Ball gespielt wird, der nicht in der ItemBank vorhanden ist, wird überprüft ob dieser auf ein anderes Item abgebildet werden kann durch Rotation.

Value

Ein JSON-gültiger String mit den folgenden Informationen:

- TicketID ein stochastischer Identifikator für die REST-Request, z.B. "JTMfOgfTEiq6ZMxb"
- GameOver ein boolescher Wert der angibt, ob das Abbruchkriterium für die Testung erreicht wurde
- NextB Eine liste mit Informationen über den nächsten Ball
 - ballmachine welche als nächstes aktiv ist
 - goal_target Zielfeld
 - speed_left Geschwindigkeit der linken Walze
 - speed_right Geschwindigkeit der rechten Walze
 - vertical_angle Vertikaler Winkel der Ballmaschine
 - shoot_delay Verzögerungszeit nach dem das Signal ertönt ist
 - expiration_time Zeit in der das Zielfeld valid_color leuchten soll, dann mit expiration_color
 - valid_color String für die Farbe '13,83,97'
 - expiration_color String für die Farbe '200,100,0'

- Testergebnis Das Testergebnis, dass der Spieler auf der latenten Kompetenzdimension erhält. (Noch nicht implementiert, daher vorläufig NA)
- Seashell_output_base64, Grafik im base64 im JSON format, wenn gameover. Um es im Browser zu öffnen: `<!DOCTYPE html><html><body></body></html>`

Example Output, jedes Anführungszeichen, außer erstes und letztes sind backslashed

```
[{"TicketID":["YxubhQICsGdrXcGnhSZo"],"nextB":"ballmachine":[28],"goal_target":[72],"speed_left":[50],
"speed_right":[60],"vertical_angle":[4],"shot_delay":[800],"expiration_time":[2195],
"expiration_color":["200,100,0"],"Testergebnis":[null],"GameOver":[false],
"Seashell_output_base64":[null]}
```

Author(s)

Harald Fiedler, Kateryna Budzyak

adaptiv.BL16.modelTransform

adaptiv.BL16.modelTransform

Description

modelTransform() fuer yhat-Modell eines voll adaptiven Samplers des BL32-Testraums mit Stop nach 16 Bällen

Usage

```
adaptiv.BL16.modelTransform(AnfrageJSONstring)
```

Arguments

AnfrageJSONstring

String, der in R zu einer Liste umgewandelt werden kann mit folgenden Elementen:

- TestID: den Namen des Testformats. Beispielsweise: 'BL32'
- idS: die aktuelle Session-ID, beispielsweise: '002b6573-cf12-436d-bccd-0856b0bb0a25'
- idP: die Player-ID des jeweiligen Probanden, beispielsweise: 'fe553db4-bbde-43dd-a6a0-804b9e46c57'
- NamePlayer: Klarnamen des Spielers, beispielsweise: 'Mustermann, Tim'
- Birthday: Geburtsdatum des Spielers, beispielsweise: '2002-07-21'
- Team: Bezeichnung des Teams, z.B. "U17"
- SessionStart: der Zeitstempel für den Sessionsstart als String im Format "JJJJ-MM-TT HH:MM:SS", beispielsweise: '2014-03-22 13:42:03'
- adrB: eine Liste mit den Adressen der Ballkanonen, in der Reihenfolge ihrer Aktivierung, beispielsweise: [10, 45, 28] oder bei Sessionstart eine leere Liste '[]'.

- **adrW**: eine Liste von Listen mit den aufleuchtenden Zielfeldern. Beispielsweise: '[4, 6], [21, 22], [50, 18]]' oder bei Sessionstart '[]'
- **adrCol**: Eine Liste von Strings mit der Farbe der Zielfelder aus [[adrW]]. Die Farben werden als String aus rgb-Werten (max=255) übergeben. Das satte Blau in Farben von Hoffenheim ist rgb(1,95,169, maxColorValue=255). Farbe B ist die Gegnerfarbe rgb(200,100,0, maxColorValue=255). Beispielsweise: '[[["1,95,169"],["200,100,0"]],["1,95,169"],["200,100,0"]],["200,100,0"],["1,95,169"]]]' oder bei Sessionstart '[]'
- **adrOut**: eine Liste mit den Adressen, wo der Ball tatsächlich raus ist. Beispielsweise: '[4, 23, 40]' oder bei Sessionstart '[]'
- **FBt**: eine Liste mit den Angaben über die Zeitdauer zwischen Ballkannoneneinwurf und Lichtschrankensignal beim Rausschießen des Balls (in Millisekunden), beispielsweise: '[2140, 2600, 8600]' oder bei Sessionstart '[]'

Hier ein Beispiel für eine gültige Anfrage für die Abfrage eines ersten Balls:

```
#' testJSONrequest <- '{
"TestID" : "BL32",
"idS" : "002b6573-cf12-436d-bccd-0856b0bb0a25",
"idP" : "fe553db4-bbde-43dd-a6a0-804b9e46c57",
"NamePlayer" : "Mustermann, Tim",
"BirthDay" : "2002-07-21",
"Team" : "U14",
"SessionStart" : "2014-03-22 13:42:03",
"adrB" : [],
"adrW" : [],
"adrCol" : [],
"adrOut" : [],
"FBt" : []
}'
```

Für einen vierten Ball:

```
testJSONrequest <- '{
"TestID" : "BL32",
"idS" : "002b6573-cf12-436d-bccd-0856b0bb0a25",
"idP" : "fe553db4-bbde-43dd-a6a0-804b9e46c57",
"NamePlayer" : "Mustermann, Tim",
"BirthDay" : "2002-07-21",
"Team" : "U14",
"SessionStart" : "2014-03-22 13:42:03",
"adrB" : [10, 45, 28],
"adrW" : [[4, 6], [21, 22], [50, 18]],
"adrCol" : [[["1,95,169"],["200,100,0"]],["1,95,169"],["200,100,0"]],["200,100,0"],["1,95,169"]]],
"adrOut" : [4, 23, 40],
"FBt" : [2140, 2600, 8600]
}'
```

Details

Es handelt sich hier um die modelTransform-Funktion für yhat-Architektur. Bei dem Dienst handelt es sich um einen reinen Item-Sampler. Solange die Session-History weniger als 16 Bälle aufweist, wird aus dem Item-Raum der BL32 ein Item gewählt.

Value

data.frame mit den oben beschriebenen Spalten.

Author(s)

Harald Fiedler

askDB	<i>askDB</i>
-------	--------------

Description

Fragt die DB ab

Usage

```
askDB(Anfangsdatum, Enddatum)
```

Arguments

Anfangsdatum	character der Länge 1 im Format "JJJJ-MM-DD", welches dann zu einem Datumsobjekt umgewandelt wird. Achtung: Zeitzone könnte ein paar Probleme aufwerfen.
Enddatum	character der Länge 1 im Format "JJJJ-MM-DD"

Details

Diese Funktion fragt auf localhost einen DB-dump der fbn-Datenbank ab und ersetzt das Copy&Paste-Verfahren der shinyApp

Value

data.frame das dann von augmentRAW() weiterverarbeitet werden kann.

Author(s)

Harald Fiedler

`augmentRAW`*augmentRAW*

Description

data.frame SQL wird angereichert

Usage

```
augmentRAW(SQL)
```

Arguments

SQL data.frame, dass per askDB() oder readRAW() eingelesen wurde

Details

Es werden Derivate gebildet und Punktzahlen eingebunden. Problemhafte Sessions werden eliminiert. Dazu zählen zwei Sessions aus dem November 2014 ohne adrW.

Value

data.frame

Author(s)

Harald Fiedler

`calcFiedler2016a`*calcFiedler2016a*

Description

Berechnet für eine einzelne Session (DF) theta-Hat nach jedem Ball

Usage

```
calcFiedler2016a(SessionDF, ItemBank = readItemBank())
```

Arguments

SessionDF data.frame mit einer Session

ItemBank die ItemBank, wird per default mittels readItemBank() eingelesen.

Details

DF darf nur eine Session beinhalten. Dann berechnet die Funktion thetaHat nach jedem Ball und stellt dies am Ende in der Spalte Fiedler2016a zur Verfügung.

Value

data.frame mit der Spalte für die IRT-Points

Author(s)

Harald Fiedler

detectItemID	<i>detectItemID</i>
--------------	---------------------

Description

Liefert die ItemID eines Balls/Stimulus zurück

Usage

```
detectItemID(Stimulus)
```

Arguments

Stimulus	ein data.frame mit den Spalten isMultiTarg, MultiTargs, RW, AW, HW, vA, sL und sR und einer Zeile. Es handelt sich also um eine Zeile aus DF, die einen Ball darstellt
----------	--

Details

Liefert die ItemID eines Balls/Stimulus zurück, z.B. "BL03". Die Funktion ist nicht vektorwertig implementiert, sondern kann immer nur eine Abfrage auf einmal durchführen

Value

character der Länge 1, z.B. c("BL03")

Author(s)

Harald Fiedler

detectItemIDLive	<i>detectItemIDLive</i>
------------------	-------------------------

Description

Liefert live die ItemID eines Balls

Usage

```
detectItemIDLive(adrB, adrW)
```

Arguments

adrB	numeric
adrW	numeric

Details

Liefert die ItemID eines Balls/Stimulus zurück für eine REST-Anfrage des simFBN

Value

character der Länge 1, z.B. c("BL03")

Author(s)

Harald Fiedler

detectItemResponse	<i>detectItemResponse</i>
--------------------	---------------------------

Description

Liefert das Ergebnis eines Balls/Stimulus zurück, z.B. 0 oder 1

Usage

```
detectItemResponse(Stimulus)
```

Arguments

Stimulus	ein data.frame mit den Spalten isMultiTarg, MultiTargs, RW, AW, HW, vA, sL und sR und einer Zeile. Es handelt sich also um eine Zeile aus DF, die einen Ball darstellt
----------	--

Details

Liefert das Ergebnis eines Balls/Stimulus zurück, z.B. 0 oder 1. Aus dem data.frame ist nicht ersichtlich, welches für welches Modell die ItemResponse erhoben wird. Im dichotomen Rasch Modell wird das Ergebnis auf 0-1 codiert, während es für andere Modelle andere Erfassungen geben mag. Hier muss extern geklärt werden, welche ItemID welchem Modell zugeordnet ist.

Value

data frame mit der zusätzlichen Spalte ItemResponse

Author(s)

Harald Fiedler

detectRotatedItems	<i>detectRotatedItems</i>
--------------------	---------------------------

Description

Vergleicht Aktionswinkel zwischen adrB und adrW um den gespielten Ball bei der Adaptivität zu berücksichtigen

Usage

```
detectRotatedItems(playedballs)
```

Arguments

playedballs	data.frame, required fields
-------------	-----------------------------

- adrA numeric Adressen der Ausgangsfelder
- adrB numeric Adressen der Zielfelder

Details

Falls ein Ball gespielt wurde, der nicht in der ItemBank vorhanden ist, werden die Aktionswinkel verglichen um den gespielten Ball bei der Adaptivität zu berücksichtigen

Value

Vektor mit identifizierten ItemID's

Author(s)

Kateryna Budzyak

Examples

```
playedballs <- structure(list(adrB = c(10L, 45L), adrW = c(12L, 21L), adrOut = c(4L, 23L),
  FBt = c(2140L, 2140L), ItemID = c("unbekannt", "unbekannt")),
  .Names = c("adrB", "adrW", "adrOut", "FBt", "ItemID"), row.names = 1:2, class = "data.frame")
detectRotatedItems(playedballs=playedballs)
```

```
erstelleRaschMatrixSkeleton
      erstelleRaschMatrixSkeleton
```

Description

erstellt eine NA-Matrix mit den Sessions als Zeilenindex und den Item-Namen als Spaltenindex

Usage

```
erstelleRaschMatrixSkeleton(DF, ItemIDNamen)
```

Arguments

DF	data.frame auf Ballebene. Eine Spalte muss "idS".
ItemIDNamen	character array mit den Itembezeichnungen, für die eine Rasch-Matrix erstellt werden soll.

Details

Achtung: eine Spalte des data.frame muss den Spaltennamen "idS" haben.

Value

Eine Matrix voller NA, mit colnames=Itembezeichnungen und rownames=unique(idS)

Author(s)

Harald Fiedler

Examples

```
rm(list=ls())
DF <- data.frame(c("SessionA", "SessionB"), c(22, 90), c(23, 18), c(10,12))
colnames(DF) <- c("idS", "It1", "It2", "It_von_wo_ganz_anders")
ItemIDNamen <- c("Item1", "Item2", "Item3")
print(DF)
erstelleRaschMatrixSkeleton(DF=DF, ItemIDNamen=ItemIDNamen)
```

```
fillRaschMatrixSkeleton  
  fillRaschMatrixSkeleton
```

Description

Füllt die NA-Matrix mit 0 und 1, wo es zutreffend ist.

Usage

```
fillRaschMatrixSkeleton(DF, RaschMatrixSkeleton)
```

Arguments

DF data.frame auf Ballebene, etwa per SQL2DF erworben
RaschMatrixSkeleton matrix , belabeled mit SessionIDs und ItemIDs, wird etwa aus erstelleRaschMatrixSkeleton() erworben.

Value

Eine RaschMatrix mit vielen NA, und wenigen 0 und einigen 1en.

Author(s)

Harald Fiedler

```
getAdrWAlsListe      getAdrWAlsListe
```

Description

Hilfsfunktion von SQL2DF()

Usage

```
getAdrWAlsListe(adrW)
```

Arguments

adrW character

Details

In den DB-Abfragen von CGoal findet sich die Variable adrW für die Zielfelder. Beim Umstellen von Single-Target auf Multi-Target wurde aus einer Zahl nun einen String, der einen JSON-Vektor darstellt. Wenn also Früher nur das Zielfeld 7 angegeben war, kann bei Multitarget nun der Ausdruck "7,2,21" angegeben sein. Die hier vorliegende Funktion arbeitet Vektorwertig und macht beispielsweise aus den Tabelleneinträgen c("1,2,3,4", "11,12,13,14") eine List der Form list(c(1,2,3,4), c(11,12,13,14))

Value

list mit numerischen Elementen

Author(s)

Harald Fiedler

Examples

```
adrW <- c("{1, 2, 3, 4}", "{11, 12, 13, 14}")
getAdrWAlsListe(adrW = adrW)
```

getFirstAdrW

getFirstArdW

Description

Hilfsfunktion von SQL2DF(): Gibt erstes Ziel in adrW im numerischen Format

Usage

```
getFirstAdrW(adrW)
```

Arguments

adrW character Vektor, etwa c("2, 4, 5", "12,19", "4")

Details

Bei der Umstellung von Single-Target auf Multi-Target wurden die Einträge in der FBN-Datenbank stark abgeändert. Wo früher beispielsweise eine Zahl 7 für das Zielfeld mit der Adresse 7 stand, ist nun "3, 5, 15" ein String, der die unterschiedlichen Zielfelder darstellt. Unabhängig davon, ob in adrW ein multiTarget oder singleTarget-Design hinterlegt wird, liefert diese Funktion nur das erste Ziel zurück, und zwar als Zahl.

Value

numeric

Author(s)

Harald Fiedler

Examples

```
getFirstAdrW(adrW=c("{2, 4, 5}", "{12, 19}", "{4}"))
```

getHW

getHW

Description

Hilfsfunktion von augmentRAW zur Ermittlung von Höhenwinkel FF-FH-HF-HH

Usage

```
getHW(SQL)
```

Arguments

SQL data.frame welches durch read.csv() einer SQL-Query entnommen wurde

Details

Je nachdem ob eine obere Ballkanonen oder eine untere Ballkanone zum Zuge kommt, oder ein unteres Ziel respektive oberes Ziel, kommt ein anderer Höhenwinkel zu stande.

Value

character mit Einträgen aus c("FF", "FH", "HF", "HH"), wobei FF=Flach Flach bedeutet und HH=Hoch Hoch.

Author(s)

Harald Fiedler

getNachname	<i>getNachname</i>
-------------	--------------------

Description

Hilfsfunktion von augmentRWA(): Gibt aus einem Spielernamen den Vornamen

Usage

```
getNachname(Spielername)
```

Arguments

Spielername character Vektor von beliebiger Länge

Details

Spielernamen können in SQL-Abfragen des FBN beispielsweise "Dogan, Isa" sein. Es wird "Isa" zurückgegeben.

Value

character Vektor der gleichen Länge wie der an die Funktion übergebene Vektor

Author(s)

Harald Fiedler

Examples

```
Spielername <- c("Fiedler, Harald", "Mayer, Jan", "A-Team")
getNachname(Spielername = Spielername)
```

getSessionTimeStamp	<i>getSessionTimeStamp</i>
---------------------	----------------------------

Description

Hilfsfunktion von SQL2DF(): ermittelt Sessionstart

Usage

```
getSessionTimeStamp(DatumString)
```

Arguments

DatumString String, etwa "2015-08-27 18:59:25.328383+02"

Details

Macht aus 2015-08-27 18:59:25.328383+02 den String 18:59:25

Value

Ein String, etwas "18:59:25"

Author(s)

Harald Fiedler

Examples

```
DatumString <- c("2015-08-27 18:59:25.328383+02", "2015-08-27 18:59:25.328383+02", "2015-08-27 18:59:25.328383+02")
```

getVorname

getVorname

Description

Hilfsfunktion von augmentDF(): Gibt aus einem Spielernamen den Vornamen

Usage

```
getVorname(Spielernamen)
```

Arguments

Spielernamen character Vektor von beliebiger Länge

Details

Spielernamen können in SQL-Abfragen des FBN beispielsweise "Dogan, Isa" sein. Es wird "Isa" zurückgegeben.

Value

character Vektor der gleichen Länge wie der an die Funktion übergebene Vektor

Author(s)

Harald Fiedler

Examples

```
Spielernamen <- c("Fiedler, Harald", "Mayer, Jan", "A-Team")  
getVorname(Spielernamen = Spielernamen)
```

```
gibZahlFuehrendeNullen
      gibZahlFuehrendeNullen
```

Description

Hilfsfunktion von SQL2DF: aus c(3) mach c("003")

Usage

```
gibZahlFuehrendeNullen(k, digits = 3)
```

Arguments

<code>k</code>	numeric (besser wäre integer, sonst wird das Ergebnis korumpiert)
<code>digits</code>	numeric der Länge 1, das die Wortlänge bezeichnet. "0004" erhält man beispielsweise mit digits=4

Details

Wenn man `idX <- 1:31` nutzt, um einen Index `idB` zu erstellen, erhält man einen eindeutigen Schlüssel. Allerdings verhält sich die lexikografische Sortierung nicht, wie man es vielleicht möchte. So würde auf die `idB=1` nicht etwa `idB=2` folgen, sondern `idB=11`. Daher macht es Sinn, bei der Konvertierung einer Ziffer oder Zahl in ein Character eine gewisse Anzahl an Nullen voranzustellen. So wird etwa aus der Zahl 2 das Wort "002" gemacht, wodurch die lexikografische Sortierung wieder so funktioniert, wie man es gerne hätte.

Value

Ein Vektor mit der gleichen Länge wie `k`, dessen Elemente Worte sind. Sie example.

Author(s)

Harald Fiedler

Examples

```
k = c(2, 7, 17, 299)
gibZahlFuehrendeNullen(k=k, digits=9)
```

```
implodeRaschMatrix4Quality
      implodeFilledRaschMatrixSkeleton4Quality
```

Description

Lässt alle Probanden/Sessions weg, die zu viele NA haben, um eine sinnvolle ItemAnylse durchzuführen.

Usage

```
implodeRaschMatrix4Quality(RaschMatrixSkeletonFilled, MissingToleranz = 0.1)
```

Arguments

RaschMatrixSkeletonFilled
matrix bestehend aus vielen NA und einigen 0 und 1en, wie man sie aus fill-RaschMatrixSkeleton() erwirbt

MissingToleranz
numeric der Länge 1. Wie viel Prozent fehlende Bälle werden höchstens erlaubt. Default ist 10 Prozent

Value

matrix bestehend aus 0en und 1en und ganz ganz wenigen NA. Die Spalten tragen Item-Namen, und die Zeilen die Probandennamen, hier: idS

Author(s)

Harald Fiedler

```
isMultiTarget      isMultiTarget
```

Description

Hilfsfunktion von augmentRAW(): ist adrW multitargetting?

Usage

```
isMultiTarget(adrW)
```

Arguments

adrW character Array, etwa c("22, 33, 44, 55", "11, 22222, 11111", "99")

Details

Sagt, ob 11, 21, 16 oder 23 unter adrW abgespeichert wurde

Value

boolescher Vektor

Author(s)

Harald Fiedler

istFormatNachnameKommaVorname

istFormatNachnameKommaVorname

Description

Hilfsfunktion von augmentRAW()

Usage

```
istFormatNachnameKommaVorname(Spielername)
```

Arguments

Spielername String

Details

In den SQL-Auszügen des FBN finden sich Spielername vom Format "Fiedler, Harald", aber auch "A_TEST_Forschung". Die Funktion testet komponentenweise, ob zwei Strings kommagetrennt gepastet sind.

Value

Boolescher Wert, der angibt, ob das Format Name, Vorname (mutmaßlich) vorliegt

Author(s)

Harald Fiedler

Examples

```
Spielername <- c("Fiedler, Harald", "Mayer, Jan", "A-Team")
istFormatNachnameKommaVorname(Spielername = Spielername)
```

itemID2Params	<i>itemID2Params</i>
---------------	----------------------

Description

Gibt Informationen an simFBN() zurück, wie der nächste Ball gespielt werden soll

Usage

```
itemID2Params(ItemID)
```

Arguments

ItemID	charakter der Länge 1, welches den Namen des Items angibt. Implementiert sind BL01:BL32
--------	---

Details

Für eine genauere Beschreibung verweise ich auf das Markdown-Manual für BL32MultiTargetSimTest.md. Diese Funktion erstellt den Knoten "nextB"

Value

list

Author(s)

Harald Fiedler

Examples

```
itemID2Params("BL03")
```

NormTree	<i>NormTree ist eine verschachtelte Liste</i>
----------	---

Description

NormTree |__ Altersgruppen/Mannschaften |__ RAW : data.frame mit SessionID, Fiedler2016a, FBt, FBq |__ HIST : list of counts and breaks |__ Kernel |__ FBq |__ FBt |__ Level |__ FBqFBt

Usage

```
data(NormTree)
```

Format

A nested list about 1194 players

NormTree6.16

NormTree6.16 ist eine verschachtelte Liste

Description

NormTree6.16 |__ Altersgruppen/Mannschaften |__ RAW : data.frame mit SessionID, Fiedler2016a, FBt, FBq |__ HIST : list of counts and breaks |__ Kernel |__ FBq |__ FBt |__ Level |__ FBqFBt

Usage

```
data(NormTree6.16)
```

Format

A nested list about 1194 players

playedAngle

playedAngle

Description

Winkel zwischen zwei Adressen.

Usage

```
playedAngle(adrA, adrB)
```

Arguments

adrA	numeric Adressen der Ausgangsfelder
adrB	numeric Adressen der Zielfelder

Details

Gibt den Winkel zwischen zwei FBN-Adressen

Value

numeric mit Winkel aus -170:180 wobei der Winkel positiv im Uhrzeigersinn gemessen wird

Author(s)

Harald Fiedler

Examples

```
adrA=10  
adrB=18  
plotFBN()  
playedAngle(adrA=adrA, adrB=adrB)
```

plotFBN

plotFBN

Usage

```
plotFBN(Adresses = TRUE)
```

Arguments

Adresses boolescher Wert der angibt, ob die Fensteradressen mit eingegeben werden sollen.

Details

Zeichnet schematisch den Footbonaut

Author(s)

Harald Fiedler

Examples

```
plotFBN(Adresses=FALSE)
```

plotSeaShell

plotSeaShell

Description

Plottet die SeaShellGrafik

Usage

```
plotSeaShell(x, A, B, TitelA, TitelB, developerMode = FALSE,  
            gameover = FALSE)
```

Arguments

x	numeric Der Item-Response-Wert des Probanden
A	numeric Die Item-Response-Werte von Gruppe A
B	numeric die Item-Response-Werte von Gruppe B
TitelA	character mit der Headline für Gruppe A
TitelB	character mit der Headline für Gruppe B
developerMode	boolischer Wert, der eine Augenscheinkontrolle des Kernels und der Histogramms erlaubt.
gameover	boolischer Wert, der bei TRUE base64 kodierte Image wiedergibt

Details

Es handelt sich um die extra für die Normgruppenvergleiche von Hoffenheim entwickelte Grafik

Value

gameover=FALSE hat die Funktion keinen Ausgabe-wert, sonst base64 codiertes Image

Author(s)

Harald Fiedler

readAUGMENTED

readAUGMENTED

Description

Liest die per writeAUGMENTED() gespeicherten Dateien ein

Usage

```
readAUGMENTED(Dateiname,
  Pfad = "~/Dropbox (Cavorit)/Cavorit/Forschungsprojekte/Hoffenheim/RAW/")
```

Arguments

Dateiname	character der Länge 1 mit Dateiname (ohne Endung). Der Pfad wird automatisch auf die Dropbox gesetzt, genauer in den Ordner RAW vom Ordner Hoffenheim
Pfad	CharacterString mit Pfadangabe zur Dropbox

Author(s)

Harald Fiedler

Examples

```
#Dateiname = "RAW-2015-04"  
#head(readAUGMENTED(Dateiname = Dateiname))
```

readItemBank	<i>readItemBank</i>
--------------	---------------------

Description

Liest die ItemBank ein

Usage

```
readItemBank(file = NA)
```

Arguments

file	Pfad character der Länge 1, der den Pfadname zu einer .csv-Datei darstellt. Die Datei muss eine gültige ItemBank im Sinne des 4-PL-Modells sein. Als Default-Wert für den Pfad fungiert ein Pfad zu einer Pakte-Datei, die in der Lib installiert wurde (was der eigentliche Clou dieser Funktion ist).
------	---

Details

Es wird die ItemBank im 1:4PL-Modell eingelesen.

Value

data.frame für das 4PL-Modell

Author(s)

Harald Fiedler

Examples

```
ItemBank <- readItemBank()  
head(ItemBank)
```

readRAW	<i>readRAW</i>
---------	----------------

Description

Liest die per writeRAW() gespeicherten Dateien ein

Usage

```
readRAW(Dateiname,
        Pfad = "~/Dropbox (Cavorit)/Cavorit/Forschungsprojekte/Hoffenheim/RAW/")
```

Arguments

Dateiname	character der Länge 1 mit Dateiname (ohne Endung). Der Pfad wird automatisch auf die Dropbox gesetzt, genauer in den Ordner RAW vom Ordner Hoffenheim
Pfad	CharacterString mit Pfadangabe zur Dropbox

Author(s)

Harald Fiedler

Examples

```
Dateiname = "RAW-2015-04"
#head(readRAW(Dateiname = Dateiname))
```

writeAUGMENTED	<i>writeAUGMENTED</i>
----------------	-----------------------

Description

Der per augmentRAW() erzeugte Datensatz (ein data.frame) wird als R-Objekt in der Dropbox abgespeichert.

Usage

```
writeAUGMENTED(DF, Dateiname)
```

Arguments

DF	data.frame der per augmentRAW() erzeugte Datensatz
Dateiname	character der Länge 1, gibt den Dateinamen ohne Endung an. Der Pfad ist hard coded zur Dropbox

Author(s)

Harald Fiedler

`writeRAW`*writeRAW*

Description

Der per askDB() erzeugte Datensatz (ein data.frame) wird als R-Objekt in der Dropbox abgespeichert.

Usage

```
writeRAW(SQL, Dateiname)
```

Arguments

SQL	data.frame der per askDB() erzeugte Datensatz
Dateiname	character der Länge 1, gibt den Dateinamen ohne Endung an. Der Pfad ist hard coded zur Dropbox

Author(s)

Harald Fiedler

Index

*Topic **datasets**

NormTree, [24](#)

NormTree6.16, [25](#)

*Topic **package**

Rbonaut2-package, [2](#)

adaptiv.BL16.fullRandom.modelPredict,
[2](#)

adaptiv.BL16.fullRandom.modelTransform,
[4](#)

adaptiv.BL16.modelPredict, [6](#)

adaptiv.BL16.modelTransform, [8](#)

askDB, [10](#)

augmentRAW, [11](#)

calcFiedler2016a, [11](#)

detectItemID, [12](#)

detectItemIDLIVE, [13](#)

detectItemResponse, [13](#)

detectRotatedItems, [14](#)

erstelleRaschMatrixSkeleton, [15](#)

fillRaschMatrixSkeleton, [16](#)

getAdrWAlsListe, [16](#)

getFirstAdrW, [17](#)

getHW, [18](#)

getNachname, [19](#)

getSessionTimeStamp, [19](#)

getVorname, [20](#)

gibZahlFuehrendeNullen, [21](#)

implodeRaschMatrix4Quality, [22](#)

isMultiTarget, [22](#)

istFormatNachnameKommaVorname, [23](#)

itemID2Params, [24](#)

NormTree, [24](#)

NormTree6.16, [25](#)

playedAngle, [25](#)

plotFBN, [26](#)

plotSeaShell, [26](#)

Rbonaut2 (Rbonaut2-package), [2](#)

Rbonaut2-package, [2](#)

readAUGMENTED, [27](#)

readItemBank, [28](#)

readRAW, [29](#)

writeAUGMENTED, [29](#)

writeRAW, [30](#)