Package ‘sport’

January 7, 2020

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Sequential Pairwise Online Rating Techniques</td>
</tr>
<tr>
<td>Version</td>
<td>0.2.0</td>
</tr>
<tr>
<td>Depends</td>
<td>R (&gt;= 3.0)</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Dawid Kałędkowski <a href="mailto:dawid.kaledkowski@gmail.com">dawid.kaledkowski@gmail.com</a></td>
</tr>
<tr>
<td>Description</td>
<td>Calculates ratings for two-player or multi-player challenges. Methods included in package such as are able to estimate ratings (players strengths) and their evolution in time, also able to predict output of challenge. Algorithms are based on Bayesian Approximation Method, and they don't involve any matrix inversions nor likelihood estimation. Parameters are updated sequentially, and computation doesn't require any additional RAM to make estimation feasible. Additionally, base of the package is written in C++ what makes sport computation even faster. Methods used in the package refers to Mark E. Glickman (1999) <a href="http://www.glicko.net/research/glicko.pdf">http://www.glicko.net/research/glicko.pdf</a>; Mark E. Glickman (2001) <a href="">doi:10.1080/02664760120059219</a>; Ruby C. Weng, Chih-Jen Lin (2011) <a href="http://jmlr.csail.mit.edu/papers/volume12/weng11a/weng11a.pdf">http://jmlr.csail.mit.edu/papers/volume12/weng11a/weng11a.pdf</a>; W. Penny, Stephen J. Roberts (1999) <a href="">doi:10.1109/IJCNN.1999.832603</a>.</td>
</tr>
<tr>
<td>BugReports</td>
<td><a href="https://github.com/gogonzo/sport/issues">https://github.com/gogonzo/sport/issues</a></td>
</tr>
<tr>
<td>Imports</td>
<td>Rcpp, data.table, ggplot2</td>
</tr>
<tr>
<td>LinkingTo</td>
<td>Rcpp</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>Language</td>
<td>en-US</td>
</tr>
<tr>
<td>URL</td>
<td><a href="https://github.com/gogonzo/sport">https://github.com/gogonzo/sport</a></td>
</tr>
<tr>
<td>LazyData</td>
<td>true</td>
</tr>
<tr>
<td>RoxygenNote</td>
<td>7.0.2</td>
</tr>
<tr>
<td>Suggests</td>
<td>dplyr, knitr, lobstr, magrittr, pkgdown, rmarkdown, spelling, testthat</td>
</tr>
</tbody>
</table>
R topics documented:

bbt_run .................................................. 2
dbl_run ................................................. 4
glicko2_run .......................................... 6
glicko_run ............................................ 9
gpheats .................................................. 11
gpsquads ............................................... 12
plot.rating ............................................. 12
predict.rating ......................................... 13
rating_run ............................................. 13
summary.rating ....................................... 15

Index  17

---

bbt_run  Bayesian Bradley-Terry

**Description**

Bayesian Bradley-Terry

**Usage**

bbt_run(
  formula,
  data,
  r = numeric(0),
  rd = numeric(0),
  init_r = 25,
  init_rd = 25/3,
  lambda = NULL,
  share = NULL,
  weight = NULL,
  kappa = 0.5
)
### Arguments

**formula**  
formula which specifies the model. RHS Allows only player rating parameter and it should be specified in following manner:

`rank | id ~ player(name)`.

- rank player position in event.
- id event identifier in which pairwise comparison is assessed.
- player(name) name of the contestant. In this case `player(name)` helps algorithm point name of the column where player names are stored.

Users can also specify formula in different way: `rank | id ~ player(name|team)`. Which means that players are playing in teams, and results are observed for teams not for players. For more see vignette.

**data**  
data.frame which contains columns specified in formula, and optional columns defined by `lambda`, `weight`.

**r**  
named vector of initial players ratings estimates. If not specified then `r` will be created automatically for parameters specified in `formula` with initial value `init_r`.

**rd**  
rd named vector of initial rating deviation estimates. If not specified then `rd` will be created automatically for parameters specified in `formula` with initial value `init_rd`.

**init_r**  
initial values for `r` if not provided. Default (`glicko = 1500`, `glicko2 = 1500`, `bbt = 25`, `dbl = 0`)

**init_rd**  
initial values for `rd` if not provided. Default (`glicko = 350`, `glicko2 = 350`, `bbt = 25/3`, `dbl = 1`)

**lambda**  
name of the column in `data` containing lambda values or one constant value (eg. `lambda = colname` or `lambda = 0.5`). Lambda impact prior variance, and uncertainty of the matchup result. The higher lambda, the higher prior variance and more uncertain result of the matchup. Higher lambda flattens chances of winning.

**share**  
name of the column in `data` containing player share in team efforts. It’s used to first calculate combined rating of the team and then redistribute ratings update back to players level. Warning - it should be used only if formula is specified with players nested within teams (`player(player|team)`).

**weight**  
name of the column in `data` containing weights values or one constant (eg. `weight = colname` or `weight = 0.5`). Weights increasing (`weight > 1`) or decreasing (`weight < 1`) update change. Higher weight increasing impact of event result on rating estimate.

**kappa**  
controls `rd` shrinkage not to be greater than `rd*(1-kappa)`. ‘kappa=1’ means that `rd` will not be decreased.

### Value

A "rating" object is returned:

- `final_r` named vector containing players ratings.
• `final_rd` named vector containing players ratings deviations.
• `r` data.frame with evolution of the ratings and ratings deviations estimated at each event.
• `pairs` pairwise combinations of players in analysed events with prior probability and result of a challenge.
• `class` of the object.
• `method` type of algorithm used.
• `settings` arguments specified in function call.

Examples

```r
# the simplest example
data <- data.frame(
  id = c(1, 1, 1, 1),
  team = c("A", "A", "B", "B"),
  player = c("a", "b", "c", "d"),
  rank_team = c(1, 2, 2, 1),
  rank_player = c(3, 4, 1, 2)
)

bbt <- dbl_run(
  data = data,
  formula = rank_player | id ~ player(player),
  r = setNames(c(25, 23.3, 25.83, 28.33), c("a", "b", "c", "d")),
  rd = setNames(c(4.76, 0.71, 2.38, 7.14), c("a", "b", "c", "d"))
)

# nested matchup
bbt <- dbl_run(
  data = data,
  formula = rank_team | id ~ player(player | team)
)
```

---

### `dbl_run`

#### Dynamic Bayesian Logit

**Description**

Dynamic Bayesian Logit

**Usage**

```r
dbl_run(
  formula,
  data,
  r = NULL,
  rd = NULL,
```
Arguments

- **formula**: formula which specifies the model. Unlike other algorithms in the packages (glicko_run, glicko2_run, bbt_run), this method doesn’t allow players nested in teams with ‘player(player | team)’ and user should matchup in formula using ‘player(player)’. DBL allows user specify multiple parameters also in interaction with others.

- **data**: data.frame which contains columns specified in formula, and optional columns defined by lambda, weight.

- **r**: named vector of initial players ratings estimates. If not specified then r will be created automatically for parameters specified in formula with initial value init_r.

- **rd**: rd named vector of initial rating deviation estimates. If not specified then rd will be created automatically for parameters specified in formula with initial value init_rd.

- **lambda**: name of the column in ‘data’ containing lambda values or one constant value (eg. lambda = colname or lambda = 0.5). Lambda impact prior variance, and uncertainty of the matchup result. The higher lambda, the higher prior variance and more uncertain result of the matchup. Higher lambda flattens chances of winning.

- **weight**: name of the column in ‘data’ containing weights values or one constant (eg. weight = colname or weight = 0.5). Weights increasing (weight > 1) or decreasing (weight < 1) update change. Higher weight increasing impact of event result on rating estimate.

- **kappa**: controls rd shrinkage not to be greater than rd*(1 -kappa). ‘kappa=1’ means that rd will not be decreased.

- **init_r**: initial values for r if not provided. Default (glicko = 1500, glicko2 = 1500, bbt = 25, dbl = 0)

- **init_rd**: initial values for rd if not provided. Default (glicko = 350, glicko2 = 350, bbt = 25/3, dbl = 1)

Value

A "rating" object is returned:

- **final_r** named vector containing players ratings.
- **final_rd** named vector containing players ratings deviations.
- **r** data.frame with evolution of the ratings and ratings deviations estimated at each event.
glicko2_run

- **pairs** pairwise combinations of players in analysed events with prior probability and result of a challenge.
- **class** of the object.
- **method** type of algorithm used.
- **settings** arguments specified in function call.

**Examples**

```r
# the simplest example
data <- data.frame(
  id = c(1, 1, 1, 1),
  name = c("A", "B", "C", "D"),
  rank = c(3, 4, 1, 2),
  gate = c(1, 2, 3, 4),
  factor1 = c("a", "a", "b", "b"),
  factor2 = c("a", "b", "a", "b")
)
dbl <- glicko2_run(
data = data,
  formula = rank | id ~ player(name)
)
dbl <- glicko2_run(
data = data,
  formula = rank | id ~ player(name) + gate * factor1)
```

---

glicko2_run  
**Glicko2 rating algorithm**

**Description**

Glicko2 rating algorithm

**Usage**

```r
glicko2_run(
  formula,
  data,
  r = numeric(0),
  rd = numeric(0),
  sigma = numeric(0),
  lambda = NULL,
  share = NULL,
  weight = NULL,
  init_r = 1500,
  init_rd = 350,
```

glicko2_run

init_sigma = 0.05,
kappa = 0.5,
tau = 0.5
)

Arguments

formula formula which specifies the model. RHS Allows only player rating parameter and it should be specified in following manner:
  rank | id ~ player(name).
  • rank player position in event.
  • id event identifier in which pairwise comparison is assessed.
  • player(name) name of the contestant. In this case player(name) helps algorithm point name of the column where player names are stored.
Users can also specify formula in in different way: rank | id ~ player(name|team).
Which means that players are playing in teams, and results are observed for teams not for players. For more see vignette.
data data.frame which contains columns specified in formula, and optional columns defined by lambda, weight.
r named vector of initial players ratings estimates. If not specified then r will be created automatically for parameters specified in formula with initial value init_r.
rd rd named vector of initial rating deviation estimates. If not specified then rd will be created automatically for parameters specified in formula with initial value init_rd.
sigma (only for glicko2) named vector of initial players ratings estimates. If not specified then sigma will be created automatically for parameters specified in formula with initial value init_sigma.
lambda name of the column in 'data' containing lambda values or one constant value (eg. lambda = colname or lambda = 0.5). Lambda impact prior variance, and uncertainty of the matchup result. The higher lambda, the higher prior variance and more uncertain result of the matchup. Higher lambda flattens chances of winning.
share name of the column in 'data' containing player share in team efforts. It’s used to first calculate combined rating of the team and then redistribute ratings update back to players level. Warning - it should be used only if formula is specified with players nested within teams ('player(player|team)').
weight name of the column in 'data' containing weights values or one constant (eg. weight = colname or weight = 0.5). Weights increasing (weight > 1) or decreasing (weight < 1) update change. Higher weight increasing impact of event result on rating estimate.
init_r initial values for r if not provided. Default (glicko = 1500, glicko2 = 1500, bbt = 25, dbl = 0)
init_rd initial values for rd if not provided. Default (glicko = 350, glicko2 = 350, bbt = 25/3, dbl = 1)
init_sigma: initial values for sigma if not provided. Default = 0.5.

kappa: controls rd shrinkage not to be greater than rd*(1 -kappa). ‘kappa=1’ means that rd will not be decreased.

tau: The system constant. Which constrains the change in volatility over time. Reasonable choices are between 0.3 and 1.2 (default = 0.5), though the system should be tested to decide which value results in greatest predictive accuracy. Smaller values of tau prevent the volatility measures from changing by large amounts, which in turn prevent enormous changes in ratings based on very improbable results. If the application of Glicko-2 is expected to involve extremely improbable collections of game outcomes, then ‘tau’ should be set to a small value, even as small as, say, tau= 0.

Value

A “rating” object is returned:

- final_r named vector containing players ratings.
- final_rd named vector containing players ratings deviations.
- final_sigma named vector containing players ratings volatile.
- r data.frame with evolution of the ratings and ratings deviations estimated at each event.
- pairs pairwise combinations of players in analysed events with prior probability and result of a challenge.
- class of the object.
- method type of algorithm used.
- settings arguments specified in function call.

Examples

# the simplest example
data <- data.frame(
id = c(1, 1, 1, 1),
team = c("A", "A", "B", "B"),
player = c("a", "b", "c", "d"),
rank_team = c(1, 1, 2, 2),
rank_player = c(3, 4, 1, 2)
)

glicko2 <- glicko2_run(
data = data,
formula = rank_player | id ~ player(player),
r = setNames(c(1500.0, 1400.0, 1550.0, 1700.0), c("a", "b", "c", "d")),
rd = setNames(c(200.0, 30.0, 100.0, 300.0), c("a", "b", "c", "d"))
)

# nested matchup
glicko2 <- glicko2_run(
data = data,
glicko_run

formula = rank_team | id ~ player(player | team)
)

glicko_run  Glicko rating algorithm

Description

Glicko rating algorithm

Usage

glicko_run(
  data,
  formula,
  r = numeric(0),
  rd = numeric(0),
  init_r = 1500,
  init_rd = 350,
  lambda = numeric(0),
  share = numeric(0),
  weight = numeric(0),
  kappa = 0.5
)

Arguments

data  data.frame which contains columns specified in formula, and optional columns defined by lambda, weight.

formula  formula which specifies the model. RHS Allows only player rating parameter and it should be specified in following manner:
  rank | id ~ player(name).
  • rank player position in event.
  • id event identifier in which pairwise comparison is assessed.
  • player(name) name of the contestant. In this case player(name) helps algorithm point name of the column where player names are stored.

Users can also specify formula in different way: rank | id ~ player(name|team). Which means that players are playing in teams, and results are observed for teams not for players. For more see vignette.

r  named vector of initial players ratings estimates. If not specified then r will be created automatically for parameters specified in formula with initial value init_r.

rd  rd named vector of initial rating deviation estimates. If not specified then rd will be created automatically for parameters specified in formula with initial value init_rd.
### glicko_run

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>init_r</td>
<td>initial values for $r$ if not provided. Default ($glicko = 1500, glicko2 = 1500$, $bbt = 25, dbl = 0$)</td>
</tr>
<tr>
<td>init_rd</td>
<td>initial values for $rd$ if not provided. Default ($glicko = 350, glicko2 = 350, bbt = 25/3, dbl = 1$)</td>
</tr>
<tr>
<td>lambda</td>
<td>name of the column in <code>data</code> containing lambda values or one constant value (eg. $\lambda = \text{colname}$ or $\lambda = 0.5$). Lambda impact prior variance, and uncertainty of the matchup result. The higher lambda, the higher prior variance and more uncertain result of the matchup. Higher lambda flattens chances of winning.</td>
</tr>
<tr>
<td>share</td>
<td>name of the column in <code>data</code> containing player share in team efforts. It’s used to first calculate combined rating of the team and then redistribute ratings update back to players level. Warning - it should be used only if formula is specified with players nested within teams (‘player(player</td>
</tr>
<tr>
<td>weight</td>
<td>name of the column in <code>data</code> containing weights values or one constant (eg. $\text{weight} = \text{colname}$ or $\text{weight} = 0.5$). Weights increasing ($\text{weight} &gt; 1$) or decreasing ($\text{weight} &lt; 1$) update change. Higher weight increasing impact of event result on rating estimate.</td>
</tr>
<tr>
<td>kappa</td>
<td>controls $rd$ shrinkage not to be greater than $rd * (1 - \kappa)$. ‘kappa=1’ means that $rd$ will not be decreased.</td>
</tr>
</tbody>
</table>

### Value

A "rating" object is returned:
- `final_r` named vector containing players ratings.
- `final_rd` named vector containing players ratings deviations.
- `r` data.frame with evolution of the ratings and ratings deviations estimated at each event.
- `pairs` pairwise combinations of players in analysed events with prior probability and result of a challenge.
- `class` of the object.
- `method` type of algorithm used.
- `settings` arguments specified in function call.

### Examples

```r
# the simplest example
data <- data.frame(  
id = c(1, 1, 1, 1),  
team = c("A", "A", "B", "B"),  
player = c("a", "b", "c", "d"),  
rank_team = c(1, 1, 2, 2),  
rank_player = c(3, 4, 1, 2)
)

# Example from Glickman
glicko <- glicko_run(  
data = data,
```
# nested matchup
glicko <- glicko_run(
  data = data,
  formula = rank_team | id ~ player(player | team)
)

---

**gpheats**  
*Heat results of Speedway Grand-Prix*

**Description**

Actual dataset containing heats results of all Speedway Grand-Prix tournaments gpheats.

**Format**

A data frame with >19000 rows and 11 variables:

- **id** event identifier
- **season** year of Grand-Prix, 1995-now
- **date** date of tournament
- **round** round in season
- **name** Tournament name
- **heat** heat number, 1-23
- **field** number of gate, 1-4
- **rider** rider name, string
- **points** paints gained, integer
- **position** position at finish line, string
- **rank** rank at finish line, integer

**Source**

internal
gpsquads  
_Tournament results of Speedway Grand-Prix_

**Description**

Actual dataset containing turnament results of all Speedway Grand-Prix events.

**Format**

A data frame with >4000 rows and 9 variables:

- **id**  event identifier
- **season** year of Grand-Prix, 1995-now
- **date** date of turnament
- **place** stadium of event
- **round** round in season
- **name** Turnament name
- **rider** rider names, 1-6
- **points** points gained, integer
- **classification** classification after an event

**Source**

internal

---

plot.rating  
_Plot rating object_

**Description**

Plot rating object

**Usage**

```r
## S3 method for class 'rating'
plot(x, n = 10, players, ...)
```

**Arguments**

- `x` of class rating
- `n` number of teams to be plotted
- `players` optional vector with names of the contestants (coefficients) to plot their evolution in time.
- `...` optional arguments
**predict.rating**

*Predict rating model*

**Description**

Predict rating model

**Usage**

```r
## S3 method for class 'rating'
predict(object, newdata, ...)
```

**Arguments**

- `object`: of class rating
- `newdata`: data.frame with data to predict
- `...`: optional arguments

**Value**

probabilities of winning challenge by player over his opponent in all provided events.

**Examples**

```r
glicko <- glicko_run(data = gpheats[1:16, ],
                    formula = rank | id ~ player(rider))
predict(glicko, gpheats[17:20, ])
```

---

**rating_run**

*Apply rating algorithm*

**Description**

Apply rating algorithm

**Usage**

```r
rating_run(
  method,
  data,
  formula,
  r = numeric(0),
  rd = numeric(0),
  sigma = numeric(0),
  init_r = numeric(0),
```
init_rd = numeric(0),
init_sigma = numeric(0),
lambda = numeric(0),
share = numeric(0),
weight = numeric(0),
kappa = numeric(0),
tau = numeric(0)
)

Arguments

method
  one of c("glicko","glicko2","bbt","dbl")
data
  data.frame which contains columns specified in formula, and optional columns
defined by lambda, weight.formula
  formula which specifies the model. RHS Allows only player rating parameter
and it should be specified in following manner:
  rank | id ~ player(name).
  • rank player position in event.
  • id event identifier in which pairwise comparison is assessed.
  • player(name) name of the contestant. In this case player(name) helps
    algorithm point name of the column where player names are stored.

Users can also specify formula in different way: rank | id ~ player(name|team).
Which means that players are playing in teams, and results are observed for
teams not for players. For more see vignette.
r
  named vector of initial players ratings estimates. If not specified then r will
  be created automatically for parameters specified in formula with initial value
  init_r.
rd
  rd named vector of initial rating deviation estimates. If not specified then rd will
  be created automatically for parameters specified in formula with initial value
  init_rd.
sigma
  (only for glicko2) named vector of initial players ratings estimates. If not specified
  then sigma will be created automatically for parameters specified in formula
  with initial value init_sigma.
init_r
  initial values for r if not provided. Default (glicko = 1500, glicko2 = 1500,
  bbt = 25, dbl = 0)
init_rd
  initial values for rd if not provided. Default (glicko = 350, glicko2 = 350, bbt
  = 25/3, dbl = 1)
init_sigma
  initial values for sigma if not provided. Default = 0.5
lambda
  name of the column in ‘data’ containing lambda values or one constant value
  (eg. lambda = colname or lambda = 0.5). Lambda impact prior variance, and
  uncertainty of the matchup result. The higher lambda, the higher prior variance
  and more uncertain result of the matchup. Higher lambda flattens chances of
  winning.
share
name of the column in 'data' containing player share in team efforts. It's used to first calculate combined rating of the team and then redistribute ratings update back to players level. Warning - it should be used only if formula is specified with players nested within teams ('player(player|team)').

weight
name of the column in 'data' containing weights values or one constant (eg. weight = colname or weight = 0.5). Weights increasing (weight > 1) or decreasing (weight < 1) update change. Higher weight increasing impact of event result on rating estimate.

kappa
controls rd shrinkage not to be greater than rd*(1-kappa). ‘kappa=1’ means that rd will not be decreased.

tau
The system constant. Which constrains the change in volatility over time. Reasonable choices are between 0.3 and 1.2 (default = 0.5), though the system should be tested to decide which value results in greatest predictive accuracy. Smaller values of tau prevent the volatility measures from changing by large amounts, which in turn prevent enormous changes in ratings based on very improbable results. If the application of Glicko-2 is expected to involve extremely improbable collections of game outcomes, then ‘tau’ should be set to a small value, even as small as, say, tau= 0.

---

**summary.rating**

### Description

Summarizing rating objects Summary for object of class 'rating'

### Usage

```r
## S3 method for class 'rating'
summary(object, ...)
```

### Arguments

- `object` of class rating
- `...` optional arguments

### Value

List with following elements

- `formula` modeled formula.
- `method` type of algorithm used.
- `Overall Accuracy` named vector containing players ratings.
- `r` data.frame summarized players ratings and model winning probabilities. Probabilities are returned only in models with one variable (ratings)
- name of a player
- r players ratings
- rd players ratings deviation
- `Model probability` mean predicted probability of winning the challenge by the player.
- `True probability` mean observed probability of winning the challenge by the player.
- `Accuracy` Accuracy of prediction.
- `pairings` number of pairwise occurrences.

Examples

```r
model <- glicko_run(formula = rank | id ~ player(rider),
                     data = gpheats[1:102, ])
summary(model)
```
Index

bbt_run, 2

dbl_run, 4

glicko2_run, 6
glicko_run, 9
gpheats, 11
gpsquads, 12

plot.rating, 12
predict.rating, 13

rating_run, 13
summary.rating, 15