Package ‘piratings’

May 27, 2019

Type Package
Title Calculate Pi Ratings for Teams Competing in Sport Matches
Version 0.1.9
Author Lars Van Cutsem
Maintainer Lars Van Cutsem <lars.vancutsem@hotmail.com>
Depends R (>= 2.10)
Imports methods, ggplot2
Description Calculate and optimize dynamic performance ratings of association football
teams competing in matches, in accordance with the method used in
the research paper “Determining the level of ability of football teams by
dynamic ratings based on the relative discrepancies in scores between adversaries”,
by Constantinou and Fenton (2013)
<doi:10.1515/jqas-2012-0036>
This dynamic rating system has proven to provide superior
results for predicting association football outcomes.
License GPL-2
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
Suggests knitr, rmarkdown
VignetteBuilder knitr
BugReports https://github.com/larsvancutsem/piratings/issues
NeedsCompilation no
Repository CRAN
Date/Publication 2019-05-27 08:30:07 UTC

R topics documented:
calculate_pi_ratings .................................................. 2
EPL2008_2015 .......................................................... 3
optimize_pi_ratings .................................................. 3
calculate_pi_ratings

Description
This function calculates dynamic performance ratings called "pi ratings" for sport teams in competitive matches. The pi rating system was developed by Constantinou and Fenton (2013) <doi:10.1515/jqas-2012-0036>

Usage

```r
calculate_pi_ratings(teams, outcomes, lambda, gamma, b, c, return_e)
```

Arguments

- **teams**: an (n x 2) character matrix, contains unique names for the respective home and away teams in n subsequent matches
- **outcomes**: an (n x 2) numeric matrix, contains the points that the respective home and away teams scored in n subsequent matches
- **lambda**: a constant, the learning rate for performance from recent matches, default value: 0.035
- **gamma**: a constant, the learning rate for performance from home to away and vice versa, default value: 0.7
- **b**: a constant, logarithmic base, default value: 10
- **c**: a constant, default value: 3
- **return_e**: a boolean variable, conditions the function to return either the mean squared error when return_e = TRUE, or the pi ratings when return_e = FALSE, default value: FALSE

Value

either an (n x 2) matrix containing the pi ratings for the teams in the n input matches or the mean squared error for the specific parameter setting, conditional on boolean parameter return_e being FALSE or TRUE

Examples

```r
# toy example
teams <- matrix(c("team A", "team B", "team B", "team A"), nrow = 2)
outcomes <- matrix(c(1, 3, 2, 1), nrow = 2)
calculate_pi_ratings(teams, outcomes)
```
**EPL2008_2015**  
*English Premier League match outcomes*

**Description**
All English Premier League match outcomes during the seasons 2008/2009 to 2014/2015

**Usage**
EPL2008_2015

**Format**
A data frame with 3040 rows and 5 variables:

- **date**  date of the match
- **home_team**  name of the home team
- **away_team**  name of the away team
- **home_goals**  number of goals the home team scored
- **away_goals**  number of goals the away team scored ...

**Source**
https://www.kaggle.com/hugomathien/soccer

---

**optimize_pi_ratings**  
*Optimize Pi Ratings*

**Description**
This function performs grid optimization on a prespecified set of parameters to find the optimal learning rates for calculating the pi ratings for sport teams in competitive matches for a set of teams in their respective set of sport matches. The pi rating system was developed by Constantinou and Fenton Constantinou and Fenton (2013) <doi:10.1515/jqas-2012-0036>

**Usage**
optimize_pi_ratings(teams, outcomes, lambda_in, gamma_in, b_in, c_in)
Arguments

- **teams**: an \((n \times 2)\) character matrix, contains unique names for the respective home and away teams in \(n\) subsequent matches.
- **outcomes**: an \((n \times 2)\) numeric matrix, contains the points that the respective home and away teams scored in \(n\) subsequent matches.
- **lambda_in**: a numerical vector, learning rate values to consider in the grid optimization, default value: \(\text{seq}(0, 0.1, 0.005)\).
- **gamma_in**: a numerical vector, learning rate values to consider in the grid optimization, default value: \(\text{seq}(0, 1, 0.05)\).
- **b_in**: a constant, logarithmic base, default value: 10.
- **c_in**: a constant, default value: 3.

Value

A dataframe with the results of the grid optimization, the mean squared error for every combination of learning rates \(\lambda\) and \(\gamma\) specified in the parameter vectors.

Examples

```r
# toy example
teams <- matrix(c("team A", "team B", "team B", "team A"), nrow = 2)
outcomes <- matrix(c(1, 3, 2, 1), nrow = 2)
optimize_pi_ratings(teams, outcomes, seq(0.05, 0.07, 0.005), seq(0.4, 0.6, 0.05))
```
Index

*Topic datasets
  EPL2008_2015, 3

calculate_pi_ratings, 2
EPL2008_2015, 3
optimize_pi_ratings, 3