Package: footBayes (via r-universe)

October 25, 2024

Type Package

Title Fitting Bayesian and MLE Football Models

Version 0.2.0

Date 2023-08-25

Author Leonardo Egidi[aut, cre], Vasilis Palaskas[aut].

Maintainer Leonardo Egidi <legidi@units.it>

License GPL-2

Description This is the first package allowing for the estimation, visualization and prediction of the most well-known football models: double Poisson, bivariate Poisson, Skellam, student_t, diagonal-inflated bivariate Poisson, and zero-inflated Skellam. The package allows Hamiltonian Monte Carlo (HMC) estimation through the underlying Stan environment and Maximum Likelihood estimation (MLE, for 'static' models only). The model construction relies on the most well-known football references, such as Dixon and Coles (1997) <doi:10.1111/1467-9876.00065>, Karlis and Ntzoufras (2003) <doi:10.1111/1467-9884.00366> and Egidi, Pauli and Torelli (2018) <doi:10.1177/1471082X18798414>.

URL https://github.com/leoegidi/footbayes

Encoding UTF-8

SystemRequirements pandoc (>= 1.12.3), pandoc-citeproc

Depends R (>= 3.1.0)

Imports rstan (>= 2.18.1), arm, reshape2, ggplot2, bayesplot, matrixStats, extraDistr, parallel, metRology, dplyr, numDeriv, tidyverse, magrittr

Suggests testthat, knitr (>= 1.37), rmarkdown (>= 2.10), loo

VignetteBuilder knitr

RoxygenNote 7.1.2

LazyData true

BuildManual yes

Repository https://leoegidi.r-universe.dev

RemoteUrl https://github.com/leoegidi/footbayes

RemoteRef HEAD

RemoteSha e50cec331c6b3241e180cf7ee190717da95e7c26

Contents

`england` *English league results 1888-2022*

Description

All results for English soccer games in the top 4 tiers from 1888/89 season to 2021/22 season.

Usage

```
england
```

Format

A data frame with 203956 rows and 12 variables:

Date Date of match

Season Season of match - refers to starting year

home Home team

visitor Visiting team

FT Full-time result

hgoal Goals scored by home team

vgoal Goals scored by visiting team

division Division: 1,2,3,4 or 3N (Old 3-North) or 3S (Old 3-South)

tier Tier of football pyramid: 1,2,3,4

totgoal Total goals in game

goaldif Goal difference in game home goals - visitor goals

result Result: H-Home Win, A-Away Win, D-Draw

`foot_abilities` *Plot football abilities from Stan and MLE models*

Description

Depicts teams' abilities either from the Stan models fitted via the `stan_foot` function or from MLE models fitted via the `mle_foot` function.

Usage

```
foot_abilities(object, data, type = c("attack", "defense", "both"), team, ...)
```

Arguments

`object`	An object either of class `stanfit` as given by `stan_foot` function or class `list` containing the Maximum Likelihood Estimates (MLE) for the model parameters fitted with `mle_foot`.
`data`	A data frame, or a matrix containing the following mandatory items: home team, away team, home goals, away goals.
`type`	Type of ability in Poisson models: one among `"defense"`, `"attack"` or `"both"`.
`team`	Valid team names.
`...`	Optional graphical parameters.

Value

Abilities plots for the selected teams: for Poisson models only, red denotes the attack, blue the defense.

Author(s)

Leonardo Egidi <legidi@units.it>

Examples

```
## Not run:
require(dplyr)
require(tidyverse)

data("italy")
italy <- as_tibble(italy)

### no dynamics, no prediction
```

```
italy_2000_2002<- italy %>%
 dplyr::select(Season, home, visitor, hgoal, vgoal) %>%
 dplyr::filter(Season=="2000" |  Season=="2001" | Season =="2002")

fit1 <- stan_foot(data = italy_2000_2002,
                model="double_pois") # double poisson

fit2 <- stan_foot(data = italy_2000_2002,
                model="biv_pois")    # bivariate poisson

fit3 <- stan_foot(data = italy_2000_2002,
                model="skellam")     # skellam

fit4 <- stan_foot(data = italy_2000_2002,
                model="student_t")   # student_t

foot_abilities(fit1, italy_2000_2002)
foot_abilities(fit2, italy_2000_2002)
foot_abilities(fit3, italy_2000_2002)
foot_abilities(fit4, italy_2000_2002)

### seasonal dynamics, predict the last season

fit5 <-stan_foot(data = italy_2000_2002,
                      model="biv_pois", predict =306,
                      dynamic_type = "seasonal")   # bivariate poisson
foot_abilities(fit5, italy_2000_2002)


## End(Not run)
```

`foot_prob` *Plot football matches probabilities for out-of-sample football matches.*

Description

The function provides a table containing the home win, draw and away win probabilities for a bunch of out-of-sample matches as specified by `stan_foot` or `mle_foot`.

Usage

```
foot_prob(object, data, home_team, away_team)
```

Arguments

`object`	An object either of class `stanfit` as given by `stan_foot` function or `list` as given by `mle_foot`.
`data`	A data frame, or a matrix containing the following mandatory items: home team, away team, home goals, away goals.

home_team	The home team(s) for the predicted matches.
away_team	The away team(s) for the predicted matches.

Details

For Bayesian models fitted via stan_foot the results probabilities are computed according to the simulation from the posterior predictive distribution of future (out-of-sample) matches. For MLE models fitted via the mle_foot the probabilities are computed by simulating from the MLE estimates.

Value

A data.frame containing the number of out-of-sample matches specified through the argument predict passed either in the mle_foot or in the stan_foot function. For Bayesian Poisson models the function returns also the most likely outcome (mlo) and a posterior probability plot for the exact results.

Author(s)

Leonardo Egidi <legidi@units.it>

Examples

```
## Not run:
### predict the last two weeks
require(tidyverse)
require(dplyr)

data("italy")
italy_2000<- italy %>%
 dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
 dplyr::filter(Season=="2000")

fit <- stan_foot(data = italy_2000,
                 model="double_pois", predict =18)  # double pois

foot_prob(fit, italy_2000, "Inter",
          "Bologna FC")

foot_prob(fit, italy_2000) # all the out-of-sample matches

## End(Not run)
```

---

foot_rank *Rank and points predictions*

Description

Posterior predictive plots and final rank table for football seasons.

Usage

```
foot_rank(data, object, team_sel, visualize = c("aggregated", "individual"))
```

Arguments

data	A data frame, or a matrix containing the following mandatory items: home team, away team, home goals, away goals.
object	An object of class stanfit as given by stan_foot function.
team_sel	Selected team(s). By default, all the teams are selected.
visualize	Type of plot, default is "aggregated".

Details

For Bayesian models fitted via stan_foot the final rank tables are computed according to the simulation from the posterior predictive distribution of future (out-of-sample) matches. The dataset should refer to one or more seasons from a given national football league (Premier League, Serie A, La Liga, etc.).

Value

Final rank tables and plots with the predicted points for the selected teams as given by the models fitted via the stan_foot function.

Author(s)

Leonardo Egidi <legidi@units.it>

Examples

```
## Not run:
require(tidyverse)
require(dplyr)

data("italy")
italy_1999_2000<- italy %>%
dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
dplyr::filter(Season == "1999"|Season=="2000")

fit <- stan_foot(italy_1999_2000, "double_pois", iter = 200)
foot_rank(italy_1999_2000, fit)
foot_rank(italy_1999_2000, fit, visualize =  "individual")

## End(Not run)
```

foot_round_robin *Round-robin for football leagues*

Description

Posterior predictive probabilities for a football season in a round-robin format

Usage

```
foot_round_robin(data, object, team_sel)
```

Arguments

data	A data frame, or a matrix containing the following mandatory items: home team, away team, home goals, away goals.
object	An object of class stanfit as given by stan_foot function.
team_sel	Selected team(s). By default, all the teams are selected.

Details

For Bayesian models fitted via stan_foot the round-robin table is computed according to the simulation from the posterior predictive distribution of future (out-of-sample) matches. The dataset should refer to one or more seasons from a given national football league (Premier League, Serie A, La Liga, etc.).

Value

Round-robin plot with the home-win posterior probabilities computed from the ppd of the fitted model via the stan_foot function.

Author(s)

Leonardo Egidi <legidi@units.it>

Examples

```
## Not run:
require(dplyr)

data("italy")
italy_1999_2000<- italy %>%
dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
dplyr::filter(Season == "1999"|Season=="2000")

fit <- stan_foot(italy_1999_2000, "double_pois", predict = 45, iter = 200)

foot_round_robin(italy_1999_2000, fit)
foot_round_robin(italy_1999_2000, fit, c("Parma AC", "AS Roma"))
```

```
## End(Not run)
```

`italy` *Italy league results 1934-2022*

Description

All results for Italian soccer games in the top tier from 1934/35 season to 2021/22 season.

Usage

```
italy
```

Format

A data frame with 27684 rows and 8 variables:

Date Date of match

Season Season of match - refers to starting year

home Home team

visitor Visiting team

FT Full-time result

hgoal Goals scored by home team

vgoal Goals scored by visiting team

tier Tier of football pyramid: 1

`mle_foot` *Fit football models with Maximum Likelihood*

Description

ML football modelling for the most famous models: double Poisson, bivariate Poisson, Skellam and student t.

Usage

```
mle_foot(data, model, predict, ...)
```

Arguments

data	A data frame, or a matrix containing the following mandatory items: season, home team, away team, home goals, away goals.
model	The type of model used to fit the data. One among the following: "double_pois", "biv_pois", "skellam", "student_t".
predict	The number of out-of-sample matches. If missing, the function returns the fit for the training set only.
...	Optional arguments for MLE fit algorithms.

Details

See documentation of stan_foot function for model details. MLE can be obtained only for static models, with no time-dependence. Likelihood optimization is performed via the BFGS method of the optim function.

Value

MLE and 95% profile likelihood deviance confidence intervals for the model's parameters: attack, defence, home effect and goals' correlation.

Author(s)

Leonardo Egidi <legidi@units.it>

References

Baio, G. and Blangiardo, M. (2010). Bayesian hierarchical model for the prediction of football results. Journal of Applied Statistics 37(2), 253-264.

Egidi, L., Pauli, F., and Torelli, N. (2018). Combining historical data and bookmakers' odds in modelling football scores. Statistical Modelling, 18(5-6), 436-459.

Gelman, A. (2014). Stan goes to the World Cup. From "Statistical Modeling, Causal Inference, and Social Science" blog.

Karlis, D. and Ntzoufras, I. (2003). Analysis of sports data by using bivariate poisson models. Journal of the Royal Statistical Society: Series D (The Statistician) 52(3), 381-393.

Karlis, D. and Ntzoufras,I. (2009). Bayesian modelling of football outcomes: Using the Skellam's distribution for the goal difference. IMA Journal of Management Mathematics 20(2), 133-145.

Owen, A. (2011). Dynamic Bayesian forecasting models of football match outcomes with estimation of the evolution variance parameter. IMA Journal of Management Mathematics, 22(2), 99-113.

Examples

```
## Not run:
require(tidyverse)
require(dplyr)

data("italy")
italy <- as_tibble(italy)
```

```
italy_2008<- italy %>%
   dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
   dplyr::filter( Season=="2008")

mle_fit <- mle_foot(data  = italy_2008,
                    model = "double_pois")

## End(Not run)
```

---

pp_foot *Posterior predictive checks for football models*

Description

The function provides posterior predictive plots to check the adequacy of the Bayesian models as returned by the `stan_foot` function.

Usage

```
pp_foot(data, object, type = c("aggregated", "matches"), coverage = 0.95)
```

Arguments

`data`	A data frame, or a matrix containing the following mandatory items: home team, away team, home goals, away goals.
`object`	An object of class `stanfit` as given by `stan_foot` function.
`type`	Type of plots, one among `"aggregated"` or `"matches"`.
`coverage`	Argument to specify the width $1 - \alpha$ of posterior probability intervals. Default is 0.95.

Value

Posterior predictive plots: when `"aggregated"` (default) is selected, the function returns a frequency plot for some pre-selected goal-difference values, along with their correspondent Bayesian p-values, computed as $Pr(y_rep \geq y)|y)$, where y_rep is a data replication from the posterior predictive distribution (more details in Gelman et al., 2013). Bayesian p-values very close to 0 or 1 could exhibit possible model misfits.

When `"matches"` is selected an ordered-frequency plot for all the goal-differences in the considered matches is provided, along with the empirical Bayesian coverage at level $1 - \alpha$.

Author(s)

Leonardo Egidi <legidi@units.it>

References

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). Bayesian data analysis. CRC press.

Examples

```
## Not run:
require(dplyr)

data("italy")
italy_2000<- italy %>%
 dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
 dplyr::filter(Season=="2000")

fit <- stan_foot(italy_2000, "double_pois", iter = 200)

pp_foot(italy_2000, fit)


## End(Not run)
```

`priors` *Football priors distributions and options*

Description

This prior specification is just a duplicate of some of the priors used by the **rstanarm** package.

These prior distributions can be passed to the `stan_foot` function, through the arguments `prior` and `prior_sd`. See the vignette *Prior Distributions for rstanarm Models* for further details (to view the priors used for an existing model see prior_summary). The default priors used in the **stan_foot** modeling function are intended to be *weakly informative* in that they provide moderate regularlization and help stabilize computation.

You can choose between: `normal`, `cauchy`, `laplace`, `student_t`.

Usage

```
normal(location = 0, scale = NULL, autoscale = TRUE)

student_t(df = 1, location = 0, scale = NULL, autoscale = TRUE)

cauchy(location = 0, scale = NULL, autoscale = TRUE)

laplace(location = 0, scale = NULL, autoscale = TRUE)
```

Arguments

location	Prior location. In most cases, this is the prior mean, but for cauchy (which is equivalent to student_t with df=1), the mean does not exist and location is the prior median. The default value is 0.
scale	Prior scale. The default depends on the family (see **Details**).
autoscale	A logical scalar, defaulting to TRUE.
df	Prior degrees of freedom. The default is 1 for student_t, in which case it is equivalent to cauchy.

Details

The details depend on the family of the prior being used:

Student t family: Family members:

- normal(location, scale)
- student_t(df, location, scale)
- cauchy(location, scale)

Each of these functions also takes an argument autoscale.

For the prior distribution for the intercept, location, scale, and df should be scalars. For the prior for the other coefficients they can either be vectors of length equal to the number of coefficients (not including the intercept), or they can be scalars, in which case they will be recycled to the appropriate length. As the degrees of freedom approaches infinity, the Student t distribution approaches the normal distribution and if the degrees of freedom are one, then the Student t distribution is the Cauchy distribution.

If scale is not specified it will default to 10 for the intercept and 2.5 for the other coefficients.

If the autoscale argument is TRUE (the default), then the scales will be further adjusted as described above in the documentation of the autoscale argument in the **Arguments** section.

Laplace family: Family members:

- laplace(location, scale)

Each of these functions also takes an argument autoscale.

The Laplace distribution is also known as the double-exponential distribution. It is a symmetric distribution with a sharp peak at its mean / median / mode and fairly long tails. This distribution can be motivated as a scale mixture of normal distributions and the remarks above about the normal distribution apply here as well.

Value

A named list to be used internally by the stan_foot model fitting function.

Author(s)

Leonardo Egidi <legidi@units.it>

References

Gelman, A., Jakulin, A., Pittau, M. G., and Su, Y. (2008). A weakly informative default prior distribution for logistic and other regression models. *Annals of Applied Statistics*. 2(4), 1360–1383.

See Also

The various vignettes for the **rstanarm** package also discuss and demonstrate the use of some of the supported prior distributions.

`stan_foot` *Fit football models with Stan*

Description

Stan football modelling for the most famous models: double Poisson, bivariate Poisson, Skellam, student t, diagonal-inflated bivariate Poisson and zero-inflated Skellam.

Usage

```
stan_foot(
  data,
  model,
  predict,
  ranking,
  dynamic_type,
  prior,
  prior_sd,
  ind_home = "TRUE",
  ...
)
```

Arguments

`data`	A data frame, or a matrix containing the following mandatory items: season, home team, away team, home goals, away goals.
`model`	The type of Stan model used to fit the data. One among the following: `"double_pois"`, `"biv_pois"`, `"skellam"`, `"student_t"`, `"diag_infl_biv_pois"`, `"zero_infl_skellam"`.
`predict`	The number of out-of-sample matches. If missing, the function returns the fit for the training set only.
`ranking`	Eventual numeric ranking provided for the teams in the dataset (e.g., the Coca-Cola Fifa ranking)
`dynamic_type`	One among `"weekly"` or `"seasonal"` for weekly dynamic parameters or seasonal dynamic parameters.
`prior`	The prior distribution for the team-specific abilities. Possible choices: `normal`, `student_t`, `cauchy`, `laplace`. See the **rstanarm** for a deep overview and read the vignette *Prior Distributions for rstanarm Models*
`prior_sd`	The prior distribution for the team-specific standard deviations. See the `prior` argument for more details.
`ind_home`	Home effect (default is `TRUE`).
`...`	Optional parameters passed to the function in the **rstan** package. It is possibly to specify `iter`, `chains`, `cores`, `refresh`, etc.

Details

Let (y_n^H, y_n^A) denote the observed number of goals scored by the home and the away team in the n-th game, respectively. A general bivariate Poisson model allowing for goals' correlation (Karlis & Ntzoufras, 2003) is the following:

$$
\begin{aligned}
Y_n^H, Y_n^A | \lambda_{1n}, \lambda_{2n}, \lambda_{3n} &\sim \mathsf{BivPoisson}(\lambda_{1n}, \lambda_{2n}, \lambda_{3n}) \\
\log(\lambda_{1n}) &= \mu + att_{h_n} + def_{a_n} \\
\log(\lambda_{2n}) &= att_{a_n} + def_{h_n} \\
\log(\lambda_{3n}) &= \beta_0,
\end{aligned}
$$

where the case $\lambda_{3n} = 0$ reduces to the double Poisson model (Baio & Blangiardo, 2010). $\lambda_{1n}, \lambda_{2n}$ represent the scoring rates for the home and the away team, respectively, where: μ is the home effect; the parameters att_T and def_T represent the attack and the defence abilities, respectively, for each team T, $T = 1, \ldots, N_T$; the nested indexes $h_n, a_n = 1, \ldots, N_T$ denote the home and the away team playing in the n-th game, respectively. Attack/defence parameters are imposed a sum-to-zero constraint to achieve identifiability and assigned some weakly-informative prior distributions:

$$
\begin{aligned}
att_T &\sim \mathcal{N}(\mu_{att}, \sigma_{att}) \\
def_T &\sim \mathcal{N}(\mu_{def}, \sigma_{def}),
\end{aligned}
$$

with hyperparameters $\mu_{att}, \sigma_{att}, \mu_{def}, \sigma_{def}$.

Instead of using the marginal number of goals, another alternative is to modelling directly the score difference $(y_n^H - y_n^A)$. We can use the Poisson-difference distribution (or Skellam distribution) to model goal difference in the n-th match (Karlis & Ntzoufras, 2009):

$$
y_n^H - y_n^A | \lambda_{1n}, \lambda_{2n} \sim PD(\lambda_{1n}, \lambda_{2n}),
$$

and the scoring rates $\lambda_{1n}, \lambda_{2n}$ are unchanged with respect to the bivariate/double Poisson model. If we want to use a continue distribution, we can use a student t distribution with 7 degrees of freedom (Gelman, 2014):

$$
\begin{aligned}
y_n^H - y_n^A &\sim t(7, ab_{h_n} - ab_{a(n)}, \sigma_y) \\
ab_t &\sim \mathcal{N}(\mu + b \times prior\_score_t, sigma_{ab}),
\end{aligned}
$$

where ab_t is the overall ability for the t-th team, whereas $prior\_score_t$ is a prior measure of team's strength (for instance a ranking).

These model rely on the assumption of static parameters. However, we could assume dynamics in the attach/defence abilities (Owen, 2011; Egidi et al., 2018) in terms of weeks or seasons through the argument `dynamic_type`. In such a framework, for a given number of times $1, \ldots, \mathcal{T}$, the models above would be unchanged, but the priors for the abilities parameters at each time $\tau, \tau = 2, \ldots, \mathcal{T}$, would be:

$$
\begin{aligned}
att_{T,\tau} &\sim \mathcal{N}(att_{T,\tau-1}, \sigma_{att}) \\
def_{T,\tau} &\sim \mathcal{N}(def_{T,\tau-1}, \sigma_{def}),
\end{aligned}
$$

whereas for $\tau = 1$ we have:

$$att_{T,1} \sim \mathcal{N}(\mu_{att}, \sigma_{att})$$
$$def_{T,1} \sim \mathcal{N}(\mu_{def}, \sigma_{def}).$$

Of course, the identifiability constraint must be imposed for each time τ.

The current version of the package allows for the fit of a diagonal-inflated bivariate Poisson and a zero-inflated Skellam model in the spirit of (Karlis & Ntzoufras, 2003) to better capture draw occurrences. See the vignette for further details.

Value

An object of S4 class, stanfit-class.

Author(s)

Leonardo Egidi <legidi@units.it>, Vasilis Palaskas <vasilis.palaskas94@gmail.com>.

References

Baio, G. and Blangiardo, M. (2010). Bayesian hierarchical model for the prediction of football results. Journal of Applied Statistics 37(2), 253-264.

Egidi, L., Pauli, F., and Torelli, N. (2018). Combining historical data and bookmakers' odds in modelling football scores. Statistical Modelling, 18(5-6), 436-459.

Gelman, A. (2014). Stan goes to the World Cup. From "Statistical Modeling, Causal Inference, and Social Science" blog.

Karlis, D. and Ntzoufras, I. (2003). Analysis of sports data by using bivariate poisson models. Journal of the Royal Statistical Society: Series D (The Statistician) 52(3), 381-393.

Karlis, D. and Ntzoufras,I. (2009). Bayesian modelling of football outcomes: Using the Skellam's distribution for the goal difference. IMA Journal of Management Mathematics 20(2), 133-145.

Owen, A. (2011). Dynamic Bayesian forecasting models of football match outcomes with estimation of the evolution variance parameter. IMA Journal of Management Mathematics, 22(2), 99-113.

Examples

```
## Not run:
require(tidyverse)
require(dplyr)

### Use Italian Serie A from 2000 to 2002

data("italy")
italy <- as_tibble(italy)
italy_2000_2002<- italy %>%
 dplyr::select(Season, home, visitor, hgoal,vgoal) %>%
 dplyr::filter(Season=="2000" |  Season=="2001"| Season=="2002")
```

```
### Fit Stan models
## no dynamics, no predictions

fit1 <- stan_foot(data = italy_2000_2002,
                model="double_pois") # double poisson
print(fit1, pars =c("home", "sigma_att",
                    "sigma_def"))

fit2 <- stan_foot(data = italy_2000_2002,
                model="biv_pois")    # bivariate poisson
print(fit2, pars =c("home", "rho",
                    "sigma_att", "sigma_def"))

fit3 <- stan_foot(data = italy_2000_2002,
                model="skellam")     # skellam
print(fit3, pars =c("home", "sigma_att",
                    "sigma_def"))

fit4 <- stan_foot(data = italy_2000_2002,
                model="student_t")   # student_t
print(fit4, pars =c("home", "beta"))

## seasonal dynamics, no prediction

fit5 <- stan_foot(data = italy_2000_2002,
                model="double_pois",
                dynamic_type ="seasonal") # double poisson
print(fit5, pars =c("home", "Sigma_att",
                    "Sigma_def"))

## seasonal dynamics, prediction for the last season

fit6 <- stan_foot(data = italy_2000_2002,
                model="double_pois",
                dynamic_type ="seasonal",
                predict = 306) # double poisson
print(fit6, pars =c("home", "Sigma_att",
                    "Sigma_def"))

## other priors' options

fit_p <- stan_foot(data = italy_2000_2002,
                   model="double_pois",
                   priors = student_t (4, 0, NULL),
                   prior_sd = laplace(0,1)) # double poisson with
                                            # student_t priors for teams abilities
                                            # and laplace prior for the hyper sds
print(fit_p,  pars = c("home", "sigma_att",
                    "sigma_def"))

## End(Not run)
```

Index