**Package ‘distribglm’**

April 15, 2021

**Title**  Distributed Generalized Linear Models

**Version**  0.4.1

**Description**  Distributed generalized linear models (GLM) fitting using Fisher scoring from McCullagh and Nelder (1989) <ISBN:0412317605>. Models are to be fit using a primary-secondary relationship, where the results are written to a synced folder, but data can be elsewhere though it is loaded in memory. Additional functions are available for deploying a plumber ‘API’.

**License**  GPL-3

**Encoding**  UTF-8

**RoxygenNote**  7.1.1

**Imports**  readr, stats

**URL**  [https://github.com/muschellij2/distribglm](https://github.com/muschellij2/distribglm)

**BugReports**  [https://github.com/muschellij2/distribglm/issues](https://github.com/muschellij2/distribglm/issues)

**Suggests**  testthat (>= 2.1.0), httr, jsonlite, plumberDeploy, analogsea (>= 0.9.0)

**NeedsCompilation**  no

**Author**  John Muschelli [aut, cre] (<https://orcid.org/0000-0001-6469-1750>)

**Maintainer**  John Muschelli <muschellij2@gmail.com>

**Repository**  CRAN

**Date/Publication**  2021-04-15 11:50:02 UTC

**R topics documented:**

- aggregate_gradients .................................................. 2
- api_url ................................................................. 2
- clear_model .......................................................... 5
- do_provision_glm_api .................................................. 6
- estimate_new_beta .................................................... 7
- estimate_site_gradient ............................................... 9
aggregate_gradients  Aggregate Gradient values

Description

Aggregate Gradient values

Usage

aggregate_gradients(all_gradient_files, iteration_number)

Arguments

all_gradient_files
  vector of character paths to files for gradients to combine together on the computing site

iteration_number
  number of fitting iteration, used for tracking and checking

Value

A list of estimated values, including the gradient, sample size, iteration number, covariance matrix (A_mat), number of samples with non-zero weights, the sum of the dispersion values (for overdispersion estimates)
Usage

api_url(url = NULL)

api_set_url(url)

api_available_models(url = api_url(), config = list(), ...)

api_get_current_beta(model_name, url = api_url(), config = list(), ...)

api_model_trace(model_name, url = api_url(), config = list(), ...)

api_model_specification(model_name, url = api_url(), config = list(), ...)

api_submit_gradient(
    model_name,
    url = api_url(),
    data,
    site_name,
    shuffle_rows = TRUE,
    verbose = TRUE,
    dry_run = FALSE,
    config = list(),
    ...
)

api_model_converged(model_name, url = api_url(), config = list(), ...)

api_setup_model(
    model_name,
    url = api_url(),
    formula = "y ~ x1 + x2",
    family = "binomial",
    link = "logit",
    all_site_names,
    config = list(),
    tolerance = 1e-09,
    ...
)

api_clear_model(model_name, url = api_url(), config = list(), ...)

api_estimate_model(
    model_name,
    url = api_url(),
    data,
    site_name,
    wait_time = 1,
    config = list(),
    ...
Arguments

url URL to the Plumber Server
config additional configuration settings such as http authentication and additional headers.
... additional arguments to send to api_submit_gradient
model_name name of your model
data dataset to get gradient value from. The code runs gradient_value to calculate the gradient, no individual data is submitted.
site_name name of the site, needs to be one of the all_site_names
shuffle_rows should the rows of the dataset be permuted, so as to decrease privacy concerns
verbose print out diagnostic messages
dry_run if TRUE, nothing with respect to the data is submitted to the server, but returned to see what would be submitted.
formula model formula to fit, with tilde syntax
family generalized linear model family, see family
link link function to use with family
all_site_names all the site names to fit this model
tolerance tolerance for convergence
wait_time Time, in seconds, to wait until to try to get new estimate

Value

The api_available_models function returns the available models running or already run.
The api_get_current_beta function returns the current beta estimates.
The api_model_trace function returns a list of the values throughout iterations of the model fitting.
The api_model_specification function returns a list of the parameters of the model specification, if the model is present.
The api_submit_gradient function returns a list from the result of the API call.
The api_model_converged function returns an indicator if the model converges or not.
The api_setup_model function submits a model to set up on the server.
The api_clear_model function clears out a model and returns the output from the API.

Examples

api_url()
api_set_url(api_url())
api_available_models()
clear_model

Clear Out Model and Other Helper Functions

Description
Clear Out Model and Other Helper Functions

Usage

clear_model(model_name, synced_folder)

folder_names(synced_folder)

model_output_file(model_name, synced_folder)

master_beta_file(model_name, synced_folder)

get_current_beta(model_name, synced_folder)

get_beta(model_name, synced_folder, iteration_number)

Arguments

model_name name of your model
synced_folder synced folder to do computation
iteration_number number of fitting iteration, used for tracking

Value
No return value, called for side effects.

Examples

synced_folder = tempfile()
dir.create(synced_folder)
model_name = "logistic_example"
form_file = setup_model(model_name = model_name,
synced_folder = synced_folder,
formula = y ~ x1 + x2, family = binomial())

fnames = folder_names(synced_folder)
model_output_file(model_name, synced_folder)
master_beta_file(model_name, synced_folder)
get_current_beta(model_name, synced_folder)
clear_model(model_name, synced_folder)
do_provision_glm_api

---

**do_provision_glm_api** *Deploy GLM API on Digital Ocean (DO)*

### Description

Deploy GLM API on Digital Ocean (DO)

### Usage

```r
do_provision_glm_api(
  ..., 
  application_name = "glm",
  port = 8000,
  example = FALSE,
  r_packages = NULL,
  github_r_packages = NULL
)

do_remove_glm_api(droplet, application_name = "glm", ...)

do_deploy_glm_api(
  ..., 
  application_name = "glm",
  port = 8000,
  docs = TRUE,
  forward = TRUE,
  example = FALSE
)

do_deploy_glm_api_only(
  droplet, 
  application_name = "glm",
  port = 8000,
  docs = TRUE,
  forward = TRUE,
  ...
)

do_list_plumber(droplet, ...)
```

### Arguments

- **arguments** arguments to pass to `do_provision` from `plumberDeploy` package
- **application_name** Name of application, passed to `path` argument of `do_deploy_api` function from `plumberDeploy` package
- **port** port to deploy on Digital Ocean
### Description

Estimate the updated beta value

### Usage

```r
estimate_new_beta(
  model_name,
  synced_folder,
  all_site_names = NULL,
)```

#### example

If TRUE, will deploy an example API named hello to the server on port 8000.

#### r_packages

Additional R packages to install, using `install.packages`

#### github_r_packages

Additional R packages to install from GitHub, using `remotes::install_github`

#### droplet

droplet to deploy on

#### docs

enable the Swagger interface, passed to `do_deploy_api` function from `plumberDeploy` package

#### forward

setup requests targeting the root URL on the server to point to this application, passed to `do_deploy_api` function from `plumberDeploy` package

### Value

A droplet instance

### Examples

```r
## Not run:
d = analogsea::droplets()
if (length(d) == 0) {
  droplet = NULL
} else {
  droplet = d[[1]]
}
droplet = do_provision_glm_api(droplet = droplet, region = "sfo3")
droplet = do_deploy_glm_api_only(droplet)
ip = droplet$network$v4[[1]]$ip_address
applet_url = paste0("http://", ip, "/", droplet$application_name, "/__docs__/"
if (interactive()) {
  browseURL(applet_url)
}

## End(Not run)
```
estimate_new_beta

verbose = TRUE
)

compute_model(model_name, synced_folder, all_site_names = NULL, wait_time = 5)

model_trace(model_name, synced_folder)

Arguments

model_name  name of your model
synced_folder synced folder to do computation
all_site_names all the site names used to fit this model
verbose      print diagnostic messages
wait_time    Time, in seconds, to wait until to try to get new estimate

Value

A file name of the estimated values necessary for the final estimates

Examples

data = data.frame(y = c(0, 0, 1),
                pois_y = c(4, 1, 0),
x2 = c(-2.19021287072066,
                -0.344307138450805, 3.47215796952745),
x1 = c(-0.263859503846267,
                -0.985160029707486, 0.227262373184513))
synced_folder = tempfile()
dir.create(synced_folder)
model_name = "logistic_example"
form_file = setup_model(model_name = model_name,
                        synced_folder = synced_folder,
                        formula = y ~ x1 + x2, family = binomial(),
                        tolerance = 5)
outfile = estimate_site_gradient(
    model_name = model_name, synced_folder = synced_folder,
    all_site_names = "site1",
    data = data)
estimate_new_beta(model_name, synced_folder,
                   all_site_names = "site1")
master_beta_file(model_name, synced_folder)
outfile = estimate_site_gradient(
    model_name = model_name, synced_folder = synced_folder,
    all_site_names = "site1",
    data = data)
estimate_new_beta(model_name, synced_folder,
                   all_site_names = "site1")
master_beta_file(model_name, synced_folder)
**estimate_site_gradient**

*Estimate Site Gradient*

**Description**

Estimate Site Gradient

**Usage**

```r
estimate_site_gradient(
  model_name, 
  synced_folder, 
  site_name = "site1", 
  data, 
  all_site_names = NULL, 
  shuffle_rows = TRUE, 
  experimental = FALSE
)
```

```r
estimate_model(
  model_name, 
  synced_folder, 
  site_name = "site1", 
  data, 
  all_site_names = NULL, 
  shuffle_rows = TRUE, 
  wait_time = 1, 
  run_compute = FALSE, 
  experimental = FALSE
)
```

**Arguments**

- **model_name**: name of your model
- **synced_folder**: synced folder to do computation
- **site_name**: name of the site, needs to be one of the `all_site_names`
- **data**: dataset to get gradient value from
- **all_site_names**: all the site names used to fit this model
- **shuffle_rows**: should the rows of the dataset be permuted, so as to decrease privacy concerns
- **experimental**: using the `glm` function rather than a custom-written function
- **wait_time**: Time, in seconds, to wait until to try to get new estimate
- **run_compute**: if TRUE, when estimating the model, it will also try to run `estimate_new_beta` if all other sites have submitted. This allows all sites to be a potential computation site.
Estimate the update gradient value

**Value**

A character filename of the gradient file, with the output from `gradient_value`

**Examples**

```r
data = data.frame(y = c(0, 0, 1),
    pois_y = c(4, 1, 0),
    x2 = c(-2.19021287072066, 
    -0.344307138450805, 3.47215796952745),
    x1 = c(-0.263859503846267, 
    -0.985160029707486, 0.227262373184513))

tt = tempfile()
dir.create(tt)
model_name = "logistic_example"
form_file = setup_model(model_name = model_name,
    synced_folder = tt,
    formula = "y ~ x1 + x2", family = "binomial")

outfile = estimate_site_gradient(
    model_name = model_name, synced_folder = tt,
    all_site_names = "site1",
    data = data)

clear_model(model_name, tt)
testthat::expect_error({
    outfile = estimate_site_gradient(
        model_name = model_name, synced_folder = tt,
        all_site_names = "site1",
        data = data)
})
```

**Description**

Estimate the update gradient value

**Usage**

```r
gradient_value(
    beta = NULL,
    data,
    formula,
    family = binomial(),
    iteration_number = 0,
    shuffle_rows = TRUE,
    link = NULL
)

use_glm_gradient_value(
```
gradient_value

beta = NULL,
data,
formula,
family = binomial(),
iteration_number = 0,
shuffle_rows = TRUE
)

Arguments

beta current beta value, leave NULL to initialize
data dataset to get gradient value from
formula model formula to fit, with tilde syntax
family generalized linear model family, see family
iteration_number number of fitting iteration, used for tracking
shuffle_rows should the rows of the dataset be permuted, so as to decrease privacy concerns
link link function to use with family

Value

A list of estimated values, including the gradient, sample size, iteration number, covariance matrix (A_mat), number of samples with non-zero weights, the sum of the dispersion values (for overdispersion estimates), and a vector of values for combining to create the population gradient (u), with length of the number of beta values

Examples

data = data.frame(y = c(0, 0, 1),
pois_y = c(4, 1, 0),
x2 = c(-2.19021287072066,
-0.344307138450805, 3.47215796952745),
x1 = c(-0.263859503846267,
-0.985160029707486, 0.227262373184513))

use_glm_gradient_value(data = data, formula = y ~ x1 + x2,
family = "binomial")

use_glm_gradient_value(data = data, formula = y ~ x1 + x2,
family = "poisson")

use_glm_gradient_value(data = data, formula = y ~ x1 + x2,
family = "probit")
Description

This infix function makes it easy to replace NULLs with a default value. It’s inspired by the way that Ruby's or operation (||) works.

Usage

\[ x \%||\% y \]

Arguments

\[ x, y \]

If \( x \) is NULL, will return \( y \); otherwise returns \( x \).

Value

A vector of \( x \) or \( y \)

setup_model

Description

Setup Model and Formula

Usage

```r
setup_model(
    model_name,
    synced_folder,
    clear_model = TRUE,
    formula = y ~ x1 + x2,
    family = binomial(),
    all_site_names = NULL,
    link = NULL,
    max_iterations = 100,
    tolerance = 1e-09
)
```

```r
make_family(family, link = NULL)
```
Arguments

- `model_name` name of your model
- `synced_folder` synced folder to do computation
- `clear_model` Should the model be cleared (all files deleted model with same name) before creating new model
- `formula` model formula to fit, with tilde syntax
- `family` generalized linear model family, see `family`
- `all_site_names` all the site names to fit this model
- `link` link function to use with family
- `max_iterations` maximum number of iterations to run
- `tolerance` tolerance for convergence

Value

A character path to a formula/model file

Examples

```r
tdir = tempfile()
dir.create(tdir)
model_name = "logistic_example"
form_file = setup_model(model_name = model_name,
synced_folder = tdir,
formula = y ~ x1 + x2, family = binomial())
```
Index

aggregate_gradients, 2
api_available_models (api_url), 2
api_clear_model (api_url), 2
api_estimate_model (api_url), 2
api_get_current_beta (api_url), 2
api_model_converged (api_url), 2
api_model_specification (api_url), 2
api_model_trace (api_url), 2
api_set_url (api_url), 2
api_setup_model (api_url), 2
api_submit_gradient, 4
api_submit_gradient (api_url), 2
api_url, 2

clear_model, 5
compute_model (estimate_new_beta), 7
do_deploy_glm_api
   (do_provision_glm_api), 6
do_deploy_glm_api_only
   (do_provision_glm_api), 6
do_list_plumber (do_provision_glm_api), 6
do_provision_glm_api, 6
do_remove_glm_api
   (do_provision_glm_api), 6

estimate_model
   (estimate_site_gradient), 9
estimate_new_beta, 7, 9
estimate_site_gradient, 9

family, 4, 11, 13
folder_names (clear_model), 5

get_beta (clear_model), 5
generate_current_beta (clear_model), 5
gradient_value, 4, 10, 10

make_family (setup_model), 12
master_beta_file (clear_model), 5

model_output_file (clear_model), 5
model_trace (estimate_new_beta), 7

op=null-default, 12

setup_model, 12

use_glm_gradient_value
   (gradient_value), 10