Package ‘pda’
March 4, 2024

**Type**  Package  
**Title**  Privacy-Preserving Distributed Algorithms  
**Version**  1.2.7  
**Date**  2024-03-04  

**Description**  A collection of privacy-preserving distributed algorithms for conducting multi-site data analyses. The regression analyses can be linear regression for continuous outcome, logistic regression for binary outcome, Cox proportional hazard regression for time-to event outcome, Poisson regression for count outcome, or multi-categorical regression for nominal or ordinal outcome. The PDA algorithm runs on a lead site and only requires summary statistics from collaborating sites, with one or few iterations. The package can be used together with the online system (<https://pda-ota.pdamethods.org/>) for safe and convenient collaboration. For more information, please visit our software websites: <https://github.com/Penncil/pda>, and <https://pdamethods.org/>.

**Maintainer**  Jiajie Chen <jiajie.chen@pennmedicine.upenn.edu>

**License**  Apache License 2.0  
**Suggests**  imager, lme4  
**Depends**  R (>= 4.1.0)  
**Imports**  Rcpp (>= 0.12.19), stats, httr, rvest, jsonlite, data.table, survival, minqa, glmnet, MASS, numDeriv, metafor, ordinal, plyr  
**LinkingTo**  Rcpp, RcppArmadillo  
**RoxygenNote**  7.2.3  
**Encoding**  UTF-8  
**LazyData**  true  

**NeedsCompilation**  yes  

**Author**  Chongliang Luo [aut],  
Rui Duan [aut],  
Mackenzie Edmondson [aut],  
Jiayi Tong [aut],  
Xiaokang Liu [aut],  
Kenneth Locke [aut],  
Jiajie Chen [cre],  
Yong Chen [aut],  
Penn Computing Inference Learning (PennCIL) lab [cph]
Description

A simulated data set for ADAP demonstration

Usage

ADAP_data

Format

A list containing the following elements:

- **sites**: site id, 300 'site1', 300 'site2', 300 'site3'
- **status**: binary outcome of length 900
- **x**: 900 by 49 matrix generated by standard normal distribution, representing the covariates
covid  

**COVID-19 LOS and mortality data**

**Description**
A simulated data set of hospitalization Length of Stay (LOS) and mortality from 6 sites

**Usage**
covid

**Format**
A data frame with 2100 rows and 6 variables:
- **site** site id, 600 'site1', 500 'site2', 400 'site3', 300 'site4', 200 'site5', 100 'site6'
- **age** continuous age in year, min 3 max 97
- **sex** 2 categories, '1' for male and '0' for female
- **lab** lab test results, continuous value ranging from 2.3 to 97.4
- **los** LOS in days, ranging from 1 to 29
- **death** mortality status, '1' for death and '0' for alive.

---

```
cs  

**CrabSatellites data**

**Description**
A data set modified from the CrabSatellites data in countreg package (see demo(ODAH)).

**Usage**
cs

**Format**
A data frame containing 173 observations on 4 variables.
- **site** Simulated site id, 85 'site1' and 88 'site2'.
- **satellites** Number of satellites. Treated as (zero-inflated) count outcome in ODAH
- **width** Carapace width (cm).
- **weight** Weight (kg).

**Source**
https://rdrr.io/rforge/countreg/man/CrabSatellites.html```
getCloudConfig  

**Description**

gather cloud settings into a list

**Usage**

getCloudConfig(site_id, dir, uri, secret)

**Arguments**

- **site_id**: site identifier
- **dir**: shared directory path if flat files
- **uri**: web uri if web service
- **secret**: web token if web service

**Value**

A list of cloud parameters: site_id, secret and uri

**See Also**

pda

---

**LOS**  

**Description**

A simulated data set of hospitalization Length of Stay (LOS) from 3 sites

**Usage**

LOS

**Format**

A data frame with 1000 rows and 5 variables:

- **site**: site id, 500 'site1', 400 'site2' and 100 'site3'
- **age**: 3 categories, 'young', 'middle', and 'old'
- **sex**: 2 categories, 'M' for male and 'F' for female
- **lab**: lab test results, continuous value ranging from 0 to 100
- **los**: LOS in days, ranging from 1 to 28. Treated as continuous outcome in DLM
**lung2**

*lung2*  

**Lung cancer survival time data**

**Description**

A data set modified from the lung data in survival package (see demo(ODAC)).

**Usage**

`lung2`

**Format**

A data frame with 228 rows and 5 variables:

- **site**  
  simulated site id, 86 'site1', 83 'site2' and 59 'site3'
- **time**  
  survival time in days
- **status**  
  censoring status 0=censored, 1=dead
- **age**  
  age in years
- **sex**  
  1 for female and 0 for male

**Source**

[https://CRAN.R-project.org/package=survival](https://CRAN.R-project.org/package=survival)

---

**ODACAT_nominal**  

*ODACAT simulated data*

**Description**

A simulated data set for ODACAT demonstration

**Usage**

`ODACAT_nominal`

**Format**

A data frame with 300 rows and 5 variables:

- **id.site**  
  site id, 102 'site1', 100 'site2', 98 'site3'
- **outcome**  
  3-category outcome, possible values are 1,2,3. Category 3 will be used as reference
- **X1**  
  the first covariate, continuous
- **X2**  
  the second covariate, binary
- **X3**  
  the third covariate, binary
**ODACAT_ordinal**  

*ODACAT simulated data*

**Description**

A simulated data set for ODACAT demonstration

**Usage**

`ODACAT_ordinal`

**Format**

A data frame with 300 rows and 5 variables:

- **id.site**  site id, 105 'site1', 105 'site2', 90 'site3'
- **outcome**  3-category outcome, possible values are 1,2,3. Category 3 will be used as reference
- **X1**  the first covariate, continuous
- **X2**  the second covariate, binary
- **X3**  the third covariate, binary

---

**pda**  

*PDA: Privacy-preserving Distributed Algorithm*

**Description**

Fit Privacy-preserving Distributed Algorithms for linear, logistic, Poisson and Cox PH regression with possible heterogeneous data across sites.

**Usage**

`pda(ipdata,site_id,control,dir,uri,secret,hosdata)`

**Arguments**

- **ipdata**  Local IPD data in data frame, should include at least one column for the outcome and one column for the covariates
- **site_id**  Character site name
- **control**  pda control data
- **dir**  directory for shared flat file cloud
- **uri**  Universal Resource Identifier for this run
- **secret**  password to authenticate as site_id on uri
- **hosdata**  hospital-level data, should include the same name as defined in the control file
Value

control

control

References

(ADAP) Xiaokang Liu, et al. (2021) ADAP: multisite learning with high-dimensional heterogeneous data via A Distributed Algorithm for Penalized regression.
(dGEM) Jiayi Tong, et al. (2022) dGEM: Decentralized Generalized Linear Mixed Effects Model

See Also

pdaPut, pdaList, pdaGet, getCloudConfig and pdaSync.

Examples

```
require(survival)
require(data.table)
require(pda)
data(lung)
```
## assign to 3 sites: 'site1', 'site2', 'site3'. we demonstrate using PDA ODAL can
do obtain a surrogate estimator that is close to the pooled estimate. We run the
example in local directory. In actual collaboration, account/password for pda server
will be assigned to the sites at the server https://pda.one.
Each site can access via web browser to check the communication of the summary stats.

## for more examples, see demo(ODAC) and demo(ODAP)

# Create 3 sites, split the lung data amongst them
sites = c('site1', 'site2', 'site3')
set.seed(42)
lung2 <- lung[,c('status', 'age', 'sex')]
lung2$sex <- lung2$sex - 1
lung2$status <- ifelse(lung2$status == 2, 1, 0)
lung_split <- split(lung2, sample(1:length(sites), nrow(lung), replace=TRUE))

## fit logistic reg using pooled data
fit.pool <- glm(status ~ age + sex, family = 'binomial', data = lung2)

# ################################# STEP 1: initialize #################################
control <- list(project_name = 'Lung cancer study',
step = 'initialize',
sites = sites,
heterogeneity = FALSE,
model = 'ODAL',
family = 'binomial',
outcome = 'status',
variables = c('age', 'sex'),
optim_maxit = 100,
lead_site = 'site1',
upload_date = as.character(Sys.time()) )

# run the example in local directory:
# specify your working directory, default is the tempdir
mydir <- tempdir()
# assume lead site1: enter "1" to allow transferring the control file
pda(site_id = 'site1', control = control, dir = mydir)
# in actual collaboration, account/password for pda server will be assigned, thus:
# Not run: pda(site_id = 'site1', control = control, uri = 'https://pda.one', secret = 'abc123')
# you can also set your environment variables, and no need to specify them in pda:
# Not run: Sys.setenv(PDA_USER = 'site1', PDA_SECRET = 'abc123', PDA_URI = 'https://pda.one')
# Not run: pda(site_id = 'site1', control = control)

#' assume remote site3: enter "1" to allow tranferring your local estimate
pda(site_id = 'site3', ipdata = lung_split[[3]], dir = mydir)

#' assume remote site2: enter "1" to allow allowing your local estimate
pda(site_id = 'site2', ipdata = lung_split[[2]], dir = mydir)

# assume lead site1: enter "1" to allow tranferring your local estimate
# control.json is also automatically updated
pda(site_id = 'site1', ipdata = lung_split[[1]], dir = mydir)
## if lead site1 initialized before other sites,
## lead site1: uncomment to sync the control before STEP 2
## Not run: pda(site_id = 'site1', control = control)
## Not run: config <- getCloudConfig(site_id = 'site1')
## Not run: pdaSync(config)

#' STEP 2: derivative

assumed remote site3: enter "1" to allow transferring your derivatives
pda(site_id = 'site3', ipdata = lung_split[[3]], dir=mydir)

assumed remote site2: enter "1" to allow transferring your derivatives
pda(site_id = 'site2', ipdata = lung_split[[2]], dir=mydir)

assumed lead site1: enter "1" to allow transferring your derivatives
pda(site_id = 'site1', ipdata = lung_split[[1]], dir=mydir)

#' STEP 3: estimate

assumed lead site1: enter "1" to allow transferring the surrogate estimate
pda(site_id = 'site1', ipdata = lung_split[[1]], dir=mydir)

the PDA ODAL is now completed!
All the sites can still run their own surrogate estimates and broadcast them.

' compare the surrogate estimate with the pooled estimate
config <- getCloudConfig(site_id = 'site1', dir=mydir)
fit.odal <- pdaGet(name = 'site1_estimate', config = config)
cbind(b.pool=fit.pool$coef,
  b.odal=fit.odal$btilde,
  sd.pool=summary(fit.pool)$coef[,2],
  sd.odal=sqrt(diag(solve(fit.odal$Htilde)/nrow(lung2))))

`see demo(ODAL) for more optional steps`

---

**pdaGet**

Function to download json and return as object

**Description**

Function to download json and return as object

**Usage**

```r
pdaGet(name, config)
```

**Arguments**

<table>
<thead>
<tr>
<th>name</th>
<th>of file</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>cloud configuration</td>
</tr>
</tbody>
</table>
Value

A list of data objects from the json file on the cloud

See Also

pda

pdaList

Function to list available objects

Description

Function to list available objects

Usage

pdaList(config)

Arguments

config a list of variables for cloud configuration

Value

A list of (json) files on the cloud

See Also

pda

pdaPut

Function to upload object to cloud as json

Description

Function to upload object to cloud as json

Usage

pdaPut(obj,name,config)

Arguments

obj R object to encode as json and uploaded to cloud
name of file
config a list of variables for cloud configuration
pdaSync

Value
NONE

See Also
pda

---

pdaSync   \hspace{1cm} pda control synchronize

Description
update pda control if ready (run by lead)

Usage
pdaSync(config)

Arguments
config\hspace{1cm}cloud configuration

Value
control

See Also
pda
Index

* datasets
  ADAP_data, 2
  covid, 3
  cs, 3
  LOS, 4
  lung2, 5
  ODACAT_nominal, 5
  ODACAT_ordinal, 6

ADAP_data, 2

  covid, 3
  cs, 3

  getCloudConfig, 4

los, 4
lung2, 5

ODACAT_nominal, 5
ODACAT_ordinal, 6

pda, 6
pdaGet, 9
pdaList, 10
pdaPut, 10
pdaSync, 11