Package: subsampling (via r-universe)

October 15, 2026

Type Package

Title Optimal Subsampling Methods for Statistical Models

Version 0.4.0

Date 2026-06-20

Maintainer Qingkai Dong <qingkai.dong@uconn.edu>

Description Balancing computational and statistical efficiency, subsampling techniques offer a practical solution for handling large-scale data analysis. Subsampling methods enhance statistical modeling for massive datasets by efficiently drawing representative subsamples from full dataset based on tailored sampling probabilities. These probabilities are optimized for specific goals, such as minimizing the variance of coefficient estimates or reducing prediction error. Based on specified modeling assumptions and subsampling techniques, the package provides functions to draw subsamples from the full data, fit the model on the subsamples, and perform statistical inference.

License GPL-3

URL https://github.com/dqksnow/subsampling, https://dqksnow.github.io/subsampling/

BugReports https://github.com/dqksnow/Subsampling/issues

Imports expm, nnet, quantreg, Rcpp (>= 1.0.12), stats, survey

Suggests knitr, MASS, rmarkdown, tinytest

LinkingTo Rcpp, RcppArmadillo

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation yes

Author Qingkai Dong [aut, cre, cph], Yaqiong Yao [aut], Haiying Wang [aut], Qiang Zhang [ctb], Jun Yan [ctb]

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

Date/Publication 2026-06-21 15:20:18 UTC

RemoteUrl https://github.com/cran/subsampling

RemoteRef HEAD

RemoteSha 0502fe497e95f1f06ddd41c2e4555700cdb3371e

Contents

ssp.glm 2
ssp.glm.rF 8
ssp.quantreg 11
ssp.relogit 14
ssp.softmax 17
subsampling 21

Index **22**

`ssp.glm` *Optimal Subsampling Methods for Generalized Linear Models*

Description

Draw subsample from full dataset and fit a generalized linear model (GLM) on the subsample. For a quick start, refer to the vignette.

Usage

```
ssp.glm(
  formula,
  data,
  subset = NULL,
  n.plt,
  n.ssp,
  family = "binomial",
  criterion = "optL",
  sampling.method = "poisson",
  likelihood = "weighted",
  control = list(...),
  contrasts = NULL,
  ...
)
```

Arguments

formula	A model formula object of class "formula" that describes the model to be fitted.
data	A data frame containing the variables in the model. Denote N as the number of observations in data.
subset	An optional vector specifying a subset of observations from data to use for the analysis. This subset will be viewed as the full data.
n.plt	The pilot subsample size (first-step subsample size). This subsample is used to compute the pilot estimator and estimate the optimal subsampling probabilities.
n.ssp	The expected size of the optimal subsample (second-step subsample). For sampling.method = 'withReplacement', The exact subsample size is n.ssp. For sampling.method = 'poisson', n.ssp is the expected subsample size.
family	family can be a character string naming a family function, a family function or the result of a call to a family function.
criterion	The choices include optA, optL(default), LCC and uniform. • optA Minimizes the trace of the asymptotic covariance matrix of the subsample estimator. • optL Minimizes the trace of a transformation of the asymptotic covariance matrix. The computational complexity of optA is $O(Nd^2)$ while that of optL is $O(Nd)$. • LCC Local Case-Control sampling probability, used as a baseline subsampling strategy. • uniform Assigns equal subsampling probability $\frac{1}{N}$ to each observation, serving as a baseline subsampling strategy.
sampling.method	The sampling method to use. Options include withReplacement and poisson (default). withReplacement draws exactly n.ssp subsamples from size N full dataset with replacement, using the specified subsampling probabilities. poisson draws observations independently by comparing each subsampling probability with a realization of uniform random variable $U(0, 1)$. Differences between methods: • Sample size: withReplacement draws exactly n.ssp subsamples while poisson draws subsamples with expected size n.ssp, meaning the actual size may vary. • Memory usage: withReplacement requires the entire dataset to be loaded at once, while poisson allows for processing observations sequentially (will be implemented in future version). • Estimator performance: Theoretical results show that the poisson tends to get a subsample estimator with lower asymptotic variance compared to the withReplacement
likelihood	The likelihood function to use. Options include weighted (default) and logOddsCorrection. A bias-correction likelihood function is required for subsample since unequal subsampling probabilities introduce bias. • weighted Applies a weighted likelihood function where each observation is weighted by the inverse of its subsampling probability.

- `logOddsCorrection` This likelihood is currently implemented only for logistic regression with `sampling.method = "poisson"` (i.e., when `family` is binomial or quasibinomial). It uses a conditional likelihood, where each element of the likelihood represents the probability of $Y = 1$, given that this subsample was drawn.

`control` The argument `control` contains two tuning parameters `alpha` and `b`.

- `alpha` $\in [0, 1]$ is the mixture weight of the user-assigned subsampling probability and uniform subsampling probability. The actual subsample probability is $\pi = (1-\alpha)\pi^{opt} + \alpha\pi^{uni}$. This protects the estimator from extreme small subsampling probability. The default value is 0.01.
- `b` is a positive number which is used to constraint the poisson subsampling probability. `b` close to 0 results in subsampling probabilities closer to uniform probability $\frac{1}{N}$. `b=2` is the default value. See relevant references for further details.

`contrasts` An optional list. It specifies how categorical variables are represented in the design matrix. For example, `contrasts = list(v1 = 'contr.treatment', v2 = 'contr.sum')`.

`...` A list of parameters which will be passed to `svyglm()`.

Details

A pilot estimator for the unknown parameter β is required because both optA and optL subsampling probabilities depend on β. There is no "free lunch" when determining optimal subsampling probabilities. Fortunately the pilot estimator only needs to satisfy mild conditions. For logistic regression, this is achieved by drawing a size `n.plt` subsample with replacement from full dataset. The case-control subsample probability is applied, that is, $\pi_i = \frac{1}{2N_1}$ for $Y_i = 1$ and $\pi_i = \frac{1}{2N_0}$ for $Y_i = 0$, $i = 1, ..., N$, whereN_0 and N_1 are the counts of observations with $Y = 0$ and $Y = 1$, respectively. For other families, uniform subsampling probabilities are applied. Typically, `n.plt` is relatively small compared to `n.ssp`.

When `criterion = 'uniform'`, there is no need to compute the pilot estimator. In this case, a size `n.plt + n.ssp` subsample will be drawn with uniform sampling probability and `coef` is the corresponding estimator. Accordingly, the returned `subsample.size.expect` is `n.plt + n.ssp` rather than `n.ssp`.

As suggested by `survey::svyglm()`, for binomial and poisson families, use `family=quasibinomial()` and `family=quasipoisson()` to avoid a warning "In eval(family$initialize) : non-integer #successes in a binomial glm!". The quasi versions of the family objects give the same point estimates and suppress the warning. Since subsampling methods only rely on point estimates from svyglm() for further computation, using the quasi families does not introduce any issues.

For Gamma family, `ssp.glm` returns only the estimation of coefficients, as the dispersion parameter is not estimated.

The `logOddsCorrection` likelihood is intended only for logistic regression with Poisson subsampling and is not implemented for `sampling.method = "withReplacement"` or non-binomial families.

Value

`ssp.glm` returns an object of class "ssp.glm" containing the following components (some are optional):

model.call The original function call.

coef.plt The pilot estimator. See Details for more information.

coef.ssp The estimator obtained from the optimal subsample.

coef The weighted linear combination of `coef.plt` and `coef.ssp`. The combination weights depend on the relative size of `n.plt` and `n.ssp` and the estimated covariance matrices of `coef.plt` and `coef.ssp`. We blend the pilot subsample information into optimal subsample estimator since the pilot subsample has already been drawn. The coefficients and standard errors reported by summary are `coef` and the square root of `diag(cov)`.

cov.ssp The covariance matrix of `coef.ssp`.

cov The covariance matrix of `coef`.

index.plt Row indices of pilot subsample in the full dataset.

index Row indices of the drawn subsample in the full dataset. For `criterion = "uniform"`, this is the single-stage uniform subsample.

N The number of observations in the full dataset.

subsample.size.expect The expected subsample size, equals to `n.ssp` for `ssp.glm`. Note that for other functions, such as ssp.relogit, this value may differ.

comp.time The total time of computing subsampling probabilities, drawing subsample and fitting model on the subsample.

terms The terms object for the fitted model.

References

Wang, H. (2019). More efficient estimation for logistic regression with optimal subsamples. *Journal of machine learning research*, **20**(132), 1-59.

Ai, M., Yu, J., Zhang, H., & Wang, H. (2021). Optimal subsampling algorithms for big data regressions. *Statistica Sinica*, **31**(2), 749-772.

Wang, H., & Kim, J. K. (2022). Maximum sampled conditional likelihood for informative subsampling. *Journal of machine learning research*, **23**(332), 1-50.

Examples

```
# logistic regression
set.seed(2)
N <- 1e4
beta0 <- rep(-0.5, 7)
d <- length(beta0) - 1
corr <- 0.5
sigmax  <- matrix(corr, d, d) + diag(1-corr, d)
X <- MASS::mvrnorm(N, rep(0, d), sigmax)
Y <- rbinom(N, 1, 1 - 1 / (1 + exp(beta0[1] + X %*% beta0[-1])))
colnames(X) <- paste0("X", 1:d)
```

```
data <- data.frame(Y = Y, X)
formula <- Y ~ .
n.plt <- 500
n.ssp <- 1000
subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'quasibinomial',
criterion = "optL",
sampling.method = 'poisson',
likelihood = "logOddsCorrection")
summary(subsampling.results)
subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'quasibinomial',
criterion = "optL",
sampling.method = 'withReplacement',
likelihood = "weighted")
summary(subsampling.results)
Uni.subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'quasibinomial',
criterion = 'uniform')
summary(Uni.subsampling.results)
####################
# poisson regression
set.seed(1)
N <-  1e4
beta0 <- rep(0.5, 7)
d <- length(beta0) - 1
X <- matrix(runif(N * d), N, d)
epsilon <- runif(N)
lambda <- exp(beta0[1] + X %*% beta0[-1])
Y <- rpois(N, lambda)
colnames(X) <- paste0("X", 1:d)
data <- data.frame(Y = Y, X)
formula <- Y ~ .
n.plt <- 200
n.ssp <- 600
subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'poisson',
criterion = "optL",
sampling.method = 'poisson',
likelihood = "weighted")
summary(subsampling.results)
```

```
subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'poisson',
criterion = "optL",
sampling.method = 'withReplacement',
likelihood = "weighted")
summary(subsampling.results)
Uni.subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'poisson',
criterion = 'uniform')
summary(Uni.subsampling.results)
##################
# normal linear regression
set.seed(4)
N <- 1e4
beta0 <- rep(-0.5, 7)
d <- length(beta0) - 1
corr <- 0.5
sigmax  <- matrix(corr, d, d) + diag(1-corr, d)
X <- MASS::mvrnorm(N, rep(0, d), sigmax)
epsilon <- rnorm(N, sd = 1)
Y <- beta0[1] + X %*% beta0[-1] + epsilon
colnames(X) <- paste0("X", 1:d)
data <- data.frame(Y = Y, X)
formula <- Y ~ .
n.plt <- 200
n.ssp <- 1000
subsampling.results <- ssp.glm(
formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = "gaussian",
criterion = "optA",
sampling.method = 'poisson',
likelihood = "weighted"
)
summary(subsampling.results)
##################
# gamma regression
set.seed(1)
N <- 1e4
p <- 3
beta0 <- rep(0.5, p + 1)
d <- length(beta0) - 1
shape <- 2
X <- matrix(runif(N * d), N, d)
link_function <- function(X, beta0) 1 / (beta0[1] + X %*% beta0[-1])
```

```
scale <- link_function(X, beta0) / shape
Y <- rgamma(N, shape = shape, scale = scale)
colnames(X) <- paste0("X", 1:d)
data <- data.frame(Y = Y, X)
formula <- Y ~ .
n.plt <- 200
n.ssp <- 1000
subsampling.results <- ssp.glm(formula = formula,
data = data,
n.plt = n.plt,
n.ssp = n.ssp,
family = 'Gamma',
criterion = "optL",
sampling.method = 'poisson',
likelihood = "weighted")
summary(subsampling.results)
```

`ssp.glm.rF` *Balanced Subsampling Methods for Generalized Linear Models with Rare Features*

Description

Rare features are binary covariates with low prevalence of being one. Because uniform or classical optimal subsampling can miss expressed rare-feature observations or produce unstable pilot estimates, this function uses rarity-aware sampling probabilities to preserve information for estimating rare-feature coefficients.

The function extends `ssp.glm` by supporting rarity-aware designs, optional response balancing for binary outcomes, weighted or unweighted pilot objectives, and a combined estimator based on the union of the pilot and second-step subsamples.

Usage

```
ssp.glm.rF(
  formula,
  data,
  subset = NULL,
  n.plt,
  n.ssp,
  family = "binomial",
  criterion = "BL-Uni",
  sampling.method = "poisson",
  objective.weight.plt = "weighted",
  objective.weight = "weighted",
  control = list(...),
  contrasts = NULL,
  balance.X.plt = FALSE,
  balance.Y.plt = FALSE,
```

```
  balance.Y.ssp = FALSE,
  balance.Y.all = FALSE,
  record.stage.time = FALSE,
  rareFeature.index = NULL,
  rareThreshold = 0.09,
  na.action = getOption("na.action"),
  ...
)
```

Arguments

formula	A model formula object.
data	A data frame containing the variables in the model.
subset	An optional vector specifying a subset of observations to be used as the full dataset.
n.plt	The expected pilot sample size for two-step methods. For one-step methods (criterion = "Uni" or "BL-Uni"), the expected sample size is n.plt + n.ssp.
n.ssp	The expected second-step subsample size. For Poisson subsampling, the actual sample size may vary.
family	A character string naming a family, a family function, or the result of a call to a family function. Supported families include "binomial", "quasibinomial", "poisson", "quasipoisson", "gaussian", and "Gamma".
criterion	The subsampling criterion. Choices include: • "BL-Uni" (default): probabilities proportional to the balance score. • "Uni": uniform Poisson subsampling. • "Lopt": classical L-optimality. • "Aopt": classical A-optimality. • "R-Lopt": rareness-aware L-optimality. • "BL-Lopt": balance score combined with L-optimality.
sampling.method	The sampling method. Currently only "poisson" is supported.
objective.weight.plt	Objective weighting for the pilot fit. Use "weighted" for inverse-probability weighting or "unweighted" for an unweighted pilot objective. Unweighted pilot fitting is not allowed when the pilot sampling probability depends on the response.
objective.weight	Objective weighting for the one-step or second-step fit. Two-step methods currently require "weighted".
control	A list passed to glm.control(). Supported entries include: • alpha: mixture weight between optimal and uniform probabilities. • b: pilot-based truncation tuning parameter. • poi.method: "exact" or "estimated" for Poisson probability normalization.

contrasts	Optional list specifying how categorical variables are encoded in the design matrix.
balance.X.plt	Logical. Whether to use balance-score sampling for the pilot sample in two-step methods.
balance.Y.plt	Logical. Whether to balance the binary response in the pilot sample. Ignored for non-binary response families.
balance.Y.ssp	Logical. For one-step "Uni" and "BL-Uni" methods, whether to allocate the expected sample size across Y = 0 and Y = 1 groups in a case-control style. Ignored for two-step optimality criteria and non-binary response families.
balance.Y.all	Logical. Whether to include all Y = 1 observations and subsample from Y = 0. Ignored for non-binary response families.
record.stage.time	Logical. Whether to store timing for major internal stages in the returned object.
rareFeature.index	Rare-feature columns. Numeric values follow the same convention as the original data/model variables: if the model contains an intercept, the function internally shifts the indices to account for the intercept column in the design matrix. Character values are matched to design-matrix column names. If NULL, rare binary features are detected automatically using rareThreshold.
rareThreshold	Prevalence threshold used to automatically identify rare binary features, and to warn when user-supplied rare features have prevalence at or above the threshold.
na.action	Currently accepted for interface compatibility.
...	Additional arguments passed to glm.fit() or lm.wfit().

Details

Two-step criteria ("Lopt", "Aopt", "R-Lopt", and "BL-Lopt") draw a pilot sample, compute second-step Poisson probabilities, fit the second-step weighted GLM, and then refit on the union of the pilot and second-step samples. One-step criteria ("Uni" and "BL-Uni") draw a single Poisson subsample with expected size n.plt + n.ssp.

Value

An object of class "ssp.glm.rF" containing fitted coefficients, covariance estimates, selected row indices, rare-feature counts, response-composition summaries, and optional stage timings.

Examples

```
set.seed(2)
N <- 1000
Z1 <- rbinom(N, 1, 0.04)
Z2 <- rbinom(N, 1, 0.07)
X1 <- rnorm(N)
X2 <- rnorm(N)
eta <- 0.5 + 0.5 * Z1 + 0.5 * Z2 + 0.5 * X1 + 0.5 * X2
Y <- rbinom(N, 1, plogis(eta))
data <- data.frame(Y, Z1, Z2, X1, X2)
```

```
fit_bl <- ssp.glm.rF(
  Y ~ .,
  data = data,
  n.plt = 100,
  n.ssp = 150,
  family = "quasibinomial",
  criterion = "BL-Uni",
  rareFeature.index = 1:2
)
summary(fit_bl)

fit_rl <- ssp.glm.rF(
  Y ~ .,
  data = data,
  n.plt = 100,
  n.ssp = 150,
  family = "quasibinomial",
  criterion = "R-Lopt",
  balance.X.plt = TRUE,
  rareFeature.index = c("Z1", "Z2")
)
summary(fit_rl)
```

`ssp.quantreg` *Optimal Subsampling Methods for Quantile Regression Model*

Description

Draw subsample from full dataset and fit quantile regression model. For a quick start, refer to the vignette.

Usage

```
ssp.quantreg(
  formula,
  data,
  subset = NULL,
  tau = 0.5,
  n.plt,
  n.ssp,
  B = 5,
  boot = TRUE,
  criterion = "optL",
  sampling.method = "withReplacement",
  likelihood = c("weighted"),
  control = list(...),
  contrasts = NULL,
```

```
  ...
)
```

Arguments

formula	A model formula object of class "formula" that describes the model to be fitted.
data	A data frame containing the variables in the model. Denote N as the number of observations in data.
subset	An optional vector specifying a subset of observations from data to use for the analysis. This subset will be viewed as the full data.
tau	The interested quantile.
n.plt	The pilot subsample size (first-step subsample size). This subsample is used to compute the pilot estimator and estimate the optimal subsampling probabilities.
n.ssp	The expected size of the optimal subsample (second-step subsample). For sampling.method = 'withReplacement', The exact subsample size is n.ssp. For sampling.method = 'poisson', n.ssp is the expected subsample size.
B	The number of subsamples for the iterative sampling algorithm. Each subsample contains n.ssp observations. This allows us to estimate the covariance matrix.
boot	If TRUE then perform iterative sampling algorithm and estimate the covariance matrix. If FALSE then only one subsample with size B*n.ssp is returned.
criterion	It determines how subsampling probabilities are computed. Choices include optL(default) and uniform. • optL Minimizes the trace of a transformation of the asymptotic covariance matrix of the subsample estimator. • uniform Assigns equal subsampling probability $\frac{1}{N}$ to each observation, serving as a baseline subsampling strategy.
sampling.method	The sampling method for drawing the optimal subsample. Choices include withReplacement and poisson(default). withReplacement draws exactly n.ssp subsamples from size N full dataset with replacement, using the specified subsampling probabilities. poisson draws observations independently by comparing each subsampling probability with a realization of uniform random variable $U(0,1)$.
likelihood	The type of the maximum likelihood function used to calculate the optimal subsampling estimator. Currently weighted is implemented which applies a weighted likelihood function where each observation is weighted by the inverse of its subsampling probability.
control	The argument control contains two tuning parameters alpha and b. • alpha $\in [0,1]$ is the mixture weight of the user-assigned subsampling probability and uniform subsampling probability. The actual subsample probability is $\pi = (1-\alpha)\pi^{opt} + \alpha\pi^{uni}$. This protects the estimator from extreme small subsampling probability. The default value is 0. • b is a positive number which is used to constraint the poisson subsampling probability. b close to 0 results in subsampling probabilities closer to uniform probability $\frac{1}{N}$. b=2 is the default value. See relevant references for further details.

`contrasts` An optional list. It specifies how categorical variables are represented in the design matrix. For example, `contrasts = list(v1 = 'contr.treatment', v2 = 'contr.sum')`.

`...` A list of parameters which will be passed to `quantreg::rq()`.

Details

Most of the arguments and returned variables have the same meaning with ssp.glm. Refer to vignette

A pilot estimator for the unknown parameter β is required because optL subsampling probabilities depend on β. There is no "free lunch" when determining optimal subsampling probabilities. For quantile regression, this is achieved by drawing a size `n.plt` subsample with replacement from full dataset, using uniform sampling probability.

If `boot=TRUE`, the returned value `subsample.size.expect` equals to `B*n.ssp`, and the covariance matrix for `coef` would be calculated. If `boot=FALSE`, the returned value `subsample.size.expect` equals to `B*n.ssp`, but the covariance matrix won't be estimated.

Value

`ssp.quantreg` returns an object of class "ssp.quantreg" containing the following components (some are optional):

model.call The original function call.

coef.plt The pilot estimator. See Details for more information.

coef The estimator obtained from the optimal subsample.

cov The covariance matrix of `coef`

index.plt Row indices of pilot subsample in the full dataset.

index.ssp Row indices of of optimal subsample in the full dataset.

N The number of observations in the full dataset.

subsample.size.expect The expected subsample size

terms The terms object for the fitted model.

References

Wang, H., & Ma, Y. (2021). Optimal subsampling for quantile regression in big data. *Biometrika*, **108**(1), 99-112.

Examples

```
#quantile regression
set.seed(1)
N <- 1e4
B <- 5
tau <- 0.75
beta.true <- rep(1, 7)
d <- length(beta.true) - 1
corr  <- 0.5
sigmax  <- matrix(0, d, d)
```

```
for (i in 1:d) for (j in 1:d) sigmax[i, j] <- corr^(abs(i-j))
X <- MASS::mvrnorm(N, rep(0, d), sigmax)
err <- rnorm(N, 0, 1) - qnorm(tau)
Y <- beta.true[1] + X %*% beta.true[-1] +
err * rowMeans(abs(X))
data <- as.data.frame(cbind(Y, X))
colnames(data) <- c("Y", paste("V", 1:ncol(X), sep=""))
formula <- Y ~ .
n.plt <- 200
n.ssp <- 100
optL.results <- ssp.quantreg(formula,data,tau = tau,n.plt = n.plt,
n.ssp = n.ssp,B = B,boot = TRUE,criterion = 'optL',
sampling.method = 'withReplacement',likelihood = 'weighted')
summary(optL.results)
uni.results <- ssp.quantreg(formula,data,tau = tau,n.plt = n.plt,
n.ssp = n.ssp,B = B,boot = TRUE,criterion = 'uniform',
sampling.method = 'withReplacement', likelihood = 'weighted')
summary(uni.results)
```

`ssp.relogit` *Optimal Subsampling for Logistic Regression Model with Rare Events Data*

Description

Draw subsample from full dataset and fit logistic regression model on subsample. For a quick start, refer to the vignette.

Usage

```
ssp.relogit(
  formula,
  data,
  subset = NULL,
  n.plt,
  n.ssp,
  criterion = "optL",
  likelihood = "logOddsCorrection",
  control = list(...),
  contrasts = NULL,
  ...
)
```

Arguments

`formula`	A model formula object of class "formula" that describes the model to be fitted.
`data`	A data frame containing the variables in the model. Denote N as the number of observations in `data`.

`subset` An optional vector specifying a subset of observations from `data` to use for the analysis. This subset will be viewed as the full data.

`n.plt` The pilot subsample size (first-step subsample size). This subsample is used to compute the pilot estimator and estimate the optimal subsampling probabilities.

`n.ssp` The expected subsample size (the second-step subsample size) drawn from those samples with `Y=0`. All rare events (`Y=1`) are included in the optimal subsample automatically.

`criterion` The choices include `optA`, `optL`(default), `LCC` and `uniform`.

- `optA` Minimizes the trace of the asymptotic covariance matrix of the subsample estimator.
- `optL` Minimizes the trace of a transformation of the asymptotic covariance matrix. The computational complexity of optA is $O(Nd^2)$ while that of optL is $O(Nd)$.
- `LCC` Local Case-Control sampling probability, used as a baseline subsampling strategy.
- `uniform` Assigns equal subsampling probability $\frac{1}{N}$ to each observation, serving as a baseline subsampling strategy.

`likelihood` The likelihood function to use. Options include `weighted` and `logOddsCorrection` (default). A bias-correction likelihood function is required for subsample since unequal subsampling probabilities introduce bias.

- `weighted` Applies a weighted likelihood function where each observation is weighted by the inverse of its subsampling probability.
- `logOddsCorrection` This likelihood is available only for logistic regression model (i.e., when family is binomial or quasibinomial). It uses a conditional likelihood, where each element of the likelihood represents the probability of $Y = 1$, given that this subsample was drawn.

`control` The argument `control` contains two tuning parameters `alpha` and `b`.

- `alpha` $\in [0, 1]$ is the mixture weight of the user-assigned subsampling probability and uniform subsampling probability. The actual subsample probability is $\pi = (1-\alpha)\pi^{opt} + \alpha\pi^{uni}$. This protects the estimator from extreme small subsampling probability. The default value is 0.
- `b` is a positive number which is used to constraint the poisson subsampling probability. `b` close to 0 results in subsampling probabilities closer to uniform probability $\frac{1}{N}$. `b=2` is the default value. See relevant references for further details.

`contrasts` An optional list. It specifies how categorical variables are represented in the design matrix. For example, `contrasts = list(v1 = 'contr.treatment', v2 = 'contr.sum')`.

`...` A list of parameters which will be passed to `svyglm()`.

Details

'Rare event' stands for the number of observations where $Y = 1$ is rare compare to the number of $Y = 0$ in the full data. In the face of logistic regression with rare events, @wang2021nonuniform shows that the available information ties to the number of positive instances instead of the full

data size. Based on this insight, one can keep all the rare instances and perform subsampling on the non-rare instances to reduce the computational cost. When `criterion = optA, optL or LCC`, all observations with $Y = 1$ are preserved and it draw `n.ssp` subsamples from observations with Y=0. When `criterion = uniform`, it draws (n.plt+n.ssp) subsmples from the full sample with equal sampling probability.

A pilot estimator for the unknown parameter β is required because both optA and optL subsampling probabilities depend on β. This is achieved by drawing half size subsample from rare observations and half from non-rare observations.

Most of the arguments and returned variables have similar meaning with ssp.glm. Refer to vignette

Value

`ssp.relogit` returns an object of class "ssp.relogit" containing the following components (some are optional):

model.call The original function call.

coef.plt The pilot estimator. See Details for more information.

coef.ssp The estimator obtained from the optimal subsample.

coef The weighted linear combination of `coef.plt` and `coef.ssp`. The combination weights depend on the relative size of `n.plt` and `n.ssp` and the estimated covariance matrices of `coef.plt` and `coef.ssp`. We blend the pilot subsample information into optimal subsample estimator since the pilot subsample has already been drawn. The coefficients and standard errors reported by summary are `coef` and the square root of `diag(cov)`.

cov.ssp The covariance matrix of `coef.ssp`.

cov The covariance matrix of `beta.cmb`.

index.plt Row indices of pilot subsample in the full dataset.

index.ssp Row indices of of optimal subsample in the full dataset.

N The number of observations in the full dataset.

subsample.size.expect The expected subsample size.

terms The terms object for the fitted model.

References

Wang, H., Zhang, A., & Wang, C. (2021). Nonuniform negative sampling and log odds correction with rare events data. *Advances in Neural Information Processing Systems*, **34**, 19847-19859.

Examples

```
set.seed(1)
N <- 2 * 1e4
beta0 <- c(-5, -rep(0.7, 6))
d <- length(beta0) - 1
X <- matrix(0, N, d)
corr <- 0.5
sigmax <- corr ^ abs(outer(1:d, 1:d, "-"))
sigmax <- sigmax / 4
```

```
X <- MASS::mvrnorm(n = N, mu = rep(0, d), Sigma = sigmax)
Y <- rbinom(N, 1, 1 - 1 / (1 + exp(beta0[1] + X %*% beta0[-1])))
print(paste('N: ', N))
print(paste('sum(Y): ', sum(Y)))
n.plt <- 200
n.ssp <- 1000
data <- as.data.frame(cbind(Y, X))
colnames(data) <- c("Y", paste("V", 1:ncol(X), sep=""))
formula <- Y ~ .
subsampling.results <- ssp.relogit(formula = formula,
                                   data = data,
                                   n.plt = n.plt,
                                   n.ssp = n.ssp,
                                   criterion = 'optA',
                                   likelihood = 'logOddsCorrection')
summary(subsampling.results)
```

`ssp.softmax` *Optimal Subsampling Method for Softmax (multinomial logistic) Regression Model*

Description

Draw subsample from full dataset and fit softmax(multinomial logistic) regression model on the subsample. Refer to vignette for a quick start.

Usage

```
ssp.softmax(
  formula,
  data,
  subset,
  n.plt,
  n.ssp,
  criterion = "MSPE",
  sampling.method = "poisson",
  likelihood = "MSCLE",
  constraint = "summation",
  control = list(...),
  contrasts = NULL,
  ...
)
```

Arguments

`formula`	A model formula object of class "formula" that describes the model to be fitted.
`data`	A data frame containing the variables in the model. Denote N as the number of observations in `data`.

`subset` An optional vector specifying a subset of observations from `data` to use for the analysis. This subset will be viewed as the full data.

`n.plt` The pilot subsample size (first-step subsample size). This subsample is used to compute the pilot estimator and estimate the optimal subsampling probabilities.

`n.ssp` The expected size of the optimal subsample (second-step subsample). For `sampling.method = 'withReplacement'`, The exact subsample size is `n.ssp`. For `sampling.method = 'poisson'`, `n.ssp` is the expected subsample size.

`criterion` The criterion of optimal subsampling probabilities. Choices include `optA`, `optL`, `MSPE`(default), `LUC` and `uniform`.

- `MSPE` Minimizes the mean squared prediction error between subsample estimator and full data estimator.
- `optA` Minimizes the trace of the asymptotic covariance matrix of the subsample estimator.
- `optL` Minimizes the trace of a transformation of the asymptotic covariance matrix, which reduces computational costs than `optA`.
- `LUC` Local uncertainty sampling method, serving as a baseline subsampling strategy. See Wang and Kim (2022).
- `uniform` Assigns equal subsampling probability $\frac{1}{N}$ to each observation, serving as a baseline subsampling strategy.

`sampling.method`

The sampling method to use. Choices include `withReplacement` and `poisson`(default). `withReplacement` draws exactly `n.ssp` subsamples from size N full dataset with replacement, using the specified subsampling probabilities. `poisson` draws observations independently by comparing each subsampling probability with a realization of uniform random variable $U(0, 1)$. Differences between methods:

- Sample size: `withReplacement` draws exactly `n.ssp` subsamples while `poisson` draws subsamples with expected size `n.ssp`, meaning the actual size may vary.
- Memory usage: `withReplacement` requires the entire dataset to be loaded at once, while `poisson` allows for processing observations sequentially (will be implemented in future version).
- Estimator performance: Theoretical results show that the `poisson` tends to get a subsample estimator with lower asymptotic variance compared to the `withReplacement`

`likelihood` A bias-correction likelihood function is required for subsample since unequal subsampling probabilities introduce bias. Choices include `weighted` and `MSCLE`(default).

- `weighted` Applies a weighted likelihood function where each observation is weighted by the inverse of its subsampling probability.
- `MSCLE` It uses a conditional likelihood, where each element of the likelihood represents the density of Y_i given that this observation was drawn.

`constraint` The constraint for identifiability of softmax model. Choices include `baseline` and `summation`(default). The baseline constraint assumes the coefficient for the baseline category are 0. Without loss of generality, we set the category $Y = 0$ as the baseline category so that $\boldsymbol{\beta}_0 = 0$. The summation constraint $\sum_{k=0}^{K} \boldsymbol{\beta}_k$ is also used in the subsampling method for the purpose of calculating subsampling

probability. These two constraints lead to different interpretation of coefficients but are equal for computing $P(Y_{i,k} = 1 \mid \mathbf{x}_i)$. The estimation of coefficients returned by `ssp.softmax()` is under baseline constraint.

`control` A list of parameters for controlling the sampling process. There are two tuning parameters `alpha` and `b`. Default is `list(alpha=0, b=2)`.

- `alpha` $\in [0, 1]$ is the mixture weight of the user-assigned subsampling probability and uniform subsampling probability. The actual subsample probability is $\pi = (1-\alpha)\pi^{opt} + \alpha\pi^{uni}$. This protects the estimator from extreme small subsampling probability. The default value is 0.
- `b` is a positive number which is used to constraint the poisson subsampling probability. `b` close to 0 results in subsampling probabilities closer to uniform probability $\frac{1}{N}$. `b=2` is the default value. See relevant references for further details.

`contrasts` An optional list. It specifies how categorical variables are represented in the design matrix. For example, `contrasts = list(v1 = 'contr.treatment', v2 = 'contr.sum')`.

`...` A list of parameters which will be passed to `nnet::multinom()`.

Details

A pilot estimator for the unknown parameter β is required because MSPE, optA and optL subsampling probabilities depend on β. There is no "free lunch" when determining optimal subsampling probabilities. For softmax regression, this is achieved by drawing a size `n.plt` subsample with replacement from full dataset with uniform sampling probability.

Value

ssp.softmax returns an object of class "ssp.softmax" containing the following components (some are optional):

model.call The original function call.

coef.plt The pilot estimator. See Details for more information.

coef.ssp The estimator obtained from the optimal subsample.

coef The weighted linear combination of `coef.plt` and `coef.ssp`, under baseline constraint. The combination weights depend on the relative size of `n.plt` and `n.ssp` and the estimated covariance matrices of `coef.plt` and `coef.ssp`. We blend the pilot subsample information into optimal subsample estimator since the pilot subsample has already been drawn. The coefficients and standard errors reported by summary are `coef` and the square root of `diag(cov)`.

coef.plt.sum The pilot estimator under summation constraint. `coef.plt.sum = G %*% as.vector(coef.plt)`.

coef.ssp.sum The estimator obtained from the optimal subsample under summation constraint. `coef.ssp.sum = G %*% as.vector(coef.ssp)`.

coef.sum The weighted linear combination of `coef.plt` and `coef.ssp`, under summation constraint. `coef.sum = G %*% as.vector(coef)`.

cov.plt The covariance matrix of `coef.plt`.

cov.ssp The covariance matrix of `coef.ssp`.

cov The covariance matrix of `coef.cmb`.

cov.plt.sum The covariance matrix of `coef.plt.sum`.

cov.ssp.sum The covariance matrix of `coef.ssp.sum`.

cov.sum The covariance matrix of `coef.sum`.

index.plt Row indices of pilot subsample in the full dataset.

index.ssp Row indices of of optimal subsample in the full dataset.

N The number of observations in the full dataset.

subsample.size.expect The expected subsample size.

terms The terms object for the fitted model.

References

Yao, Y., & Wang, H. (2019). Optimal subsampling for softmax regression. *Statistical Papers*, **60**, 585-599.

Han, L., Tan, K. M., Yang, T., & Zhang, T. (2020). Local uncertainty sampling for large-scale multiclass logistic regression. *Annals of Statistics*, **48**(3), 1770-1788.

Wang, H., & Kim, J. K. (2022). Maximum sampled conditional likelihood for informative subsampling. *Journal of machine learning research*, **23**(332), 1-50.

Yao, Y., Zou, J., & Wang, H. (2023). Optimal poisson subsampling for softmax regression. *Journal of Systems Science and Complexity*, **36**(4), 1609-1625.

Yao, Y., Zou, J., & Wang, H. (2023). Model constraints independent optimal subsampling probabilities for softmax regression. *Journal of Statistical Planning and Inference*, **225**, 188-201.

Examples

```
# softmax regression
d <- 3 # dim of covariates
K <- 2 # K + 1 classes
G <- rbind(rep(-1/(K+1), K), diag(K) - 1/(K+1)) %x% diag(d)
N <- 1e4
beta.true.baseline <- cbind(rep(0, d), matrix(-1.5, d, K))
beta.true.summation <- cbind(rep(1, d), 0.5 * matrix(-1, d, K))
set.seed(1)
mu <- rep(0, d)
sigma <- matrix(0.5, nrow = d, ncol = d)
diag(sigma) <- rep(1, d)
X <- MASS::mvrnorm(N, mu, sigma)
prob <- exp(X %*% beta.true.summation)
prob <- prob / rowSums(prob)
Y <- apply(prob, 1, function(row) sample(0:K, size = 1, prob = row))
n.plt <- 500
n.ssp <- 1000
data <- as.data.frame(cbind(Y, X))
colnames(data) <- c("Y", paste("V", 1:ncol(X), sep=""))
head(data)
formula <- Y ~ . -1
WithRep.MSPE <- ssp.softmax(formula = formula,
```

```
 data = data,
 n.plt = n.plt,
 n.ssp = n.ssp,
 criterion = 'MSPE',
 sampling.method = 'withReplacement',
 likelihood = 'weighted',
 constraint = 'baseline')
summary(WithRep.MSPE)
```

`subsampling` *Optimal Subsampling Methods for Statistical Models*

Description

Subsampling methods are utilized in statistical modeling for massive datasets. These methods aim to draw representative subsamples from the full dataset based on specific sampling probabilities, with the goal of maintaining inference efficiency. The sampling probabilities are tailored to particular objectives, such as minimizing the variance of the estimated coefficients or reducing prediction error. By using subsampling techniques, the package balances the trade-off between computational efficiency and statistical efficiency, making it a practical tool for massive data analysis.

Models Supported

- Generalized Linear Models (GLMs)
- Softmax (Multinomial) Regression
- Rare Event Logistic Regression
- Quantile Regression

Author(s)

Maintainer: Qingkai Dong <qingkai.dong@uconn.edu> [copyright holder]

Authors:

- Yaqiong Yao
- Haiying Wang

Other contributors:

- Qiang Zhang [contributor]
- Jun Yan [contributor]

See Also

Useful links:

- https://github.com/dqksnow/subsampling
- https://dqksnow.github.io/subsampling/
- Report bugs at https://github.com/dqksnow/Subsampling/issues

Index

ssp.glm, 2, *13*, *16*
ssp.glm.rF, 8
ssp.quantreg, 11
ssp.relogit, *5*, 14
ssp.softmax, 17
subsampling, 21
subsampling-package *(*subsampling*)*, 21