Package 'NeEDS4BigData'

May 7, 2026

Type Package

Title New Experimental Design Based Subsampling Methods for Big Data

Version 1.0.1

Maintainer Amalan Mahendran <amalan0595@gmail.com>

Description Subsampling methods for big data under different models and assumptions. Starting with linear regression and leading to Generalised Linear Models, softmax regression, and quantile regression. Specifically, the model-robust subsampling method proposed in Mahendran, A., Thompson, H., and McGree, J. M. (2023) <doi:10.1007/s00362-023-01446-9>, where multiple models can describe the big data, and the subsampling framework for potentially misspecified Generalised Linear Models in Mahendran, A., Thompson, H., and McGree, J. M. (2025) <doi:10.48550/arXiv.2510.05902>.

License MIT + file LICENSE

URL https://github.com/Amalan-ConStat/NeEDS4BigData,https://amalan-constat.github.io/NeEDS4BigData/index.html

BugReports https://github.com/Amalan-ConStat/NeEDS4BigData/issues

Depends R (>= 4.1.0)

Imports dplyr, foreach, gam, ggh4x, ggplot2, ggridges, matrixStats, mvnfast, psych, Rdpack, Rfast, rlang, stats, tidyr

RdMacros Rdpack

Suggests doParallel, ggpubr, kableExtra, knitr, parallel, rmarkdown, spelling, testthat, vctrs, pillar

Encoding UTF-8

Language en-GB

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Amalan Mahendran [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0643-9052>)

Repository CRAN

Date/Publication 2025-10-22 19:00:08 UTC

Contents

`ALoptimalGLMSub` *A- and L-optimality criteria based subsampling under Generalised Linear Models*

Description

Using this function sample from big data under linear, logistic and Poisson regression to describe the data. Subsampling probabilities are obtained based on the A- and L- optimality criteria.

Usage

```
ALoptimalGLMSub(r0,rf,Y,X,N,family)
```

Arguments

r0	sample size for initial random sample
rf	final sample size including initial(r0) and optimal(r) samples
Y	response data or Y
X	covariate data or X matrix that has all the covariates (first column is for the intercept)
N	size of the big data
family	a character value for "linear", "logistic" and "poisson" regression from Generalised Linear Models

Details

Two stage subsampling algorithm for big data under Generalised Linear Models (linear, logistic and Poisson regression).

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A- and L-optimality criteria.

Through the estimated subsampling probabilities an optimal sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

A character value is provided for `family` and if it is not of the any three types an error message will be produced.

Value

The output of `ALoptimalGLMSub` gives a list of

`Beta_Estimates` estimated model parameters in a data.frame after subsampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon in a data.frame after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Sample_L-Optimality` list of indexes for the initial and optimal samples obtained based on L-Optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A- and L- optimality criteria

References

Wang H, Zhu R, Ma P (2018). "Optimal subsampling for large sample logistic regression." *Journal of the American Statistical Association*, **113**(522), 829–844.

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

Yao Y, Wang H (2021). "A review on optimal subsampling methods for massive datasets." *Journal of Data Science*, **19**(1), 151–172.

Examples

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1,Error_Variance=0.5)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000; Family<-"linear"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(c(6,9)*100,50); Original_Data<-Full_Data$Complete_Data;

ALoptimalGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                family = "linear")->Results

plot_Beta(Results)

Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000; Family<-"logistic"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(c(6,9)*100,50); Original_Data<-Full_Data$Complete_Data;

ALoptimalGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                family = "logistic")->Results

plot_Beta(Results)

Dist<-"Normal";
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000; Family<-"poisson"
Full_Data<-GenGLMdata(Dist,NULL,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(c(6,9)*100,50); Original_Data<-Full_Data$Complete_Data;

ALoptimalGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                family = "poisson")->Results

plot_Beta(Results)
```

`AoptimalGauLMSub` *A-optimality criteria based subsampling under Gaussian Linear Models*

Description

Using this function sample from big data under Gaussian linear regression models to describe the data. Subsampling probabilities are obtained based on the A-optimality criteria.

Usage

```
AoptimalGauLMSub(r0,rf,Y,X,N)
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data

Details

Two stage subsampling algorithm for big data under Gaussian Linear Model.

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A-optimality criteria.

Through the estimated subsampling probabilities an optimal sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

Value

The output of `AoptimalGauLMSub` gives a list of

`Beta_Estimates` estimated model parameters in a data.frame after subsampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon in a data.frame after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A-optimality criteria

References

Lee J, Schifano ED, Wang H (2021). "Fast optimal subsampling probability approximation for generalized linear models." *Econometrics and Statistics*.

Examples

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1,Error_Variance=0.5)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-10000; Family<-"linear"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(100*c(6,12),50); Original_Data<-Full_Data$Complete_Data;

AoptimalGauLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),
                 N = nrow(Original_Data))->Results

plot_Beta(Results)
```

`AoptimalMCGLMSub` *A-optimality criteria based subsampling under measurement constraints for Generalised Linear Models*

Description

Using this function sample from big data under linear, logistic and Poisson regression to describe the data when response y is partially unavailable. Subsampling probabilities are obtained based on the A-optimality criteria.

Usage

```
AoptimalMCGLMSub(r0,rf,Y,X,N,family)
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`family`	a character value for "linear", "logistic" and "poisson" regression from Generalised Linear Models

Details

Two stage subsampling algorithm for big data under Generalised Linear Models (linear, logistic and Poisson regression) when the response is not available for subsampling probability evaluation.

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A-optimality criteria.

Through the estimated subsampling probabilities an optimal sample of size $r \geq r_0$ is obtained. Finally, only the optimal sample is used and the model parameters are estimated.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

A character value is provided for `family` and if it is not of the any three types an error message will be produced.

Value

The output of `AoptimalMCGLMSub` gives a list of

`Beta_Estimates` estimated model parameters in a data.frame after subsampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon in a data.frame after subsampling (valid only for linear regression)

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A-optimality criteria

References

Zhang T, Ning Y, Ruppert D (2021). "Optimal sampling for generalized linear models under measurement constraints." *Journal of Computational and Graphical Statistics*, **30**(1), 106–114.

Examples

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1,Error_Variance=0.5)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-10000; Family<-"linear"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(100*c(6,12),50); Original_Data<-Full_Data$Complete_Data;

AoptimalMCGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 family = "linear")->Results

plot_Beta(Results)
```

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-10000; Family<-"logistic"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(100*c(6,12),50); Original_Data<-Full_Data$Complete_Data;

AoptimalMCGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 family = "logistic")->Results

plot_Beta(Results)

Dist<-"Normal";
No_Of_Var<-2; Beta<-c(-1,2,1); N<-10000; Family<-"poisson"
Full_Data<-GenGLMdata(Dist,NULL,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(100*c(6,12),50); Original_Data<-Full_Data$Complete_Data;

AoptimalMCGLMSub(r0 = r0, rf = rf,Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 family = "poisson")->Results

plot_Beta(Results)
```

`Electric_consumption` *Electric consumption data*

Description

Hebrail and Berard (2012) described data which contains 2,049,280 completed measurements for a house located at Sceaux, France between December 2006 and November 2010. The log scale minute-averaged current intensity is selected as the response and the covariates are voltage, active electrical energy (watt-hour) in the kitchen, the laundry room, and electric water-heater and an air-conditioner.

Usage

```
Electric_consumption
```

Format

A data frame with 4 columns and 2,049,280 rows.

`Intensity` Minute-averaged current intensity

`Voltage` Voltage

`EE_Kitchen` Active electrical energy (watt-hour) in the kitchen

`EE_Laundry` Active electrical energy (watt-hour) in the laundry room

`EE_WH_AC` Active electrical energy (watt-hour) of electric water-heater and an air-conditioner

Source

Extracted from

Hebrail G, Berard A (2012) Individual Household Electric Power Consumption. UCI Machine Learning Repository.

Available at: doi:10.24432/C58K54

Examples

```
nrow(Electric_consumption)
```

`GenGLMdata` *Generate data for Generalised Linear Models*

Description

Function to simulate big data under linear, logistic and Poisson regression for sampling. Covariate data X is through Normal, Multivariate Normal or Uniform distribution for linear regression. Covariate data X is through Exponential, Normal, Multivariate Normal or Uniform distribution for logistic regression. Covariate data X is through Normal or Uniform distribution for Poisson regression.

Usage

```
GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,family)
```

Arguments

`Dist`	a character value for the distribution "Normal", "MVNormal", "Uniform or "Exponential"
`Dist_Par`	a list of parameters for the distribution that would generate data for covariate X
`No_Of_Var`	number of variables
`Beta`	a vector for the model parameters, including the intercept
`N`	the big data size
`family`	a character vector for "linear", "logistic" and "poisson" regression from Generalised Linear Models

Details

Big data for the Generalised Linear Models are generated by the "linear", "logistic" and "poisson" regression types.

We have limited the covariate data generation for linear regression through normal, multivariate normal and uniform distribution, logistic regression through exponential, normal, multivariate normal and uniform distribution Poisson regression through normal and uniform distribution.

Value

The output of GenGLMdata gives a list of

Complete_Data a matrix for Y and X

References

Lee Y, Nelder JA (1996). "Hierarchical generalized linear models." *Journal of the Royal Statistical Society Series B: Statistical Methodology*, **58**(4), 619–656.

Examples

```
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000;

Dist<-"MVNormal";
Dist_Par<-list(Mean=rep(0,No_Of_Var),Variance=diag(rep(2,No_Of_Var)),Error_Variance=0.5)
Family<-"linear"
Results<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1);
Family<-"logistic"
Results<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

Dist<-"Uniform"; Family<-"poisson"
Results<-GenGLMdata(Dist,NULL,No_Of_Var,Beta,N,Family)
```

GenModelMissGLMdata *Generate data for Generalised Linear Models under model misspecification scenario*

Description

Function to simulate big data under Generalised Linear Models for the model misspecification scenario through any misspecification type.

Usage

```
GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon,family)
```

Arguments

N	the big data size
X_Data	a matrix for the covariate data
Misspecification	a vector of values for the misspecification
Beta	a vector for the model parameters, including the intercept and misspecification term

Var_Epsilon	variance value for the residuals
family	a character vector for "linear", "logistic" and "poisson" regression from Generalised Linear Models

Details

Big data for the Generalised Linear Models are generated by the "linear", "logistic" and "poisson" regression types under model misspecification.

Value

The output of GenModelMissGLMdata gives a list of

Complete_Data a matrix for Y,X and f(x)

References

Adewale AJ, Wiens DP (2009). "Robust designs for misspecified logistic models." *Journal of Statistical Planning and Inference*, **139**(1), 3–15.

Adewale AJ, Xu X (2010). "Robust designs for generalized linear models with possible overdispersion and misspecified link functions." *Computational statistics & data analysis*, **54**(4), 875–890.

Examples

```
Beta<-c(-1,0.75,0.75,1); Var_Epsilon<-0.5; family <- "linear"; N<-10000
X_1 <- replicate(2,stats::runif(n=N,min = -1,max = 1))

Temp<-Rfast::rowprods(X_1)
Misspecification <- (Temp-mean(Temp))/sqrt(mean(Temp^2)-mean(Temp)^2)
X_Data <- cbind(X0=1,X_1);

Results<-GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon,family)

Results<-GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon=NULL,family="logistic")

Results<-GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon=NULL,family="poisson")
```

LCCsampling *Local case control sampling for logistic regression*

Description

Using this function sample from big data under logistic regression to describe the data. Sampling probabilities are obtained based on local case control method.

Usage

```
LCCsampling(r0,rf,Y,X,N)
```

Arguments

r0	sample size for initial random sample
rf	final sample size including initial(r0) and case control(r) samples
Y	response data or Y
X	covariate data or X matrix that has all the covariates (first column is for the intercept)
N	size of the big data

Details

Two stage sampling algorithm for big data under logistic regression.

First obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters sampling probabilities are evaluated for local case control.

Through the estimated sampling probabilities an optimal sample of size $r \geq r_0$ is obtained. Finally, the optimal sample is used and the model parameters are estimated.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

Value

The output of `LCCsampling` gives a list of

`Beta_Estimates` estimated model parameters in a data.frame after sampling

`Sample_LCC_Sampling` list of indexes for the initial and optimal samples obtained based on local case control sampling

`Sampling_Probability` vector of calculated sampling probabilities for local case control sampling

References

Fithian W, Hastie T (2015). "Local case-control sampling: Efficient subsampling in imbalanced data sets." *Quality control and applied statistics*, **60**(3), 187–190.

Examples

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-10000; Family<-"logistic"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

r0<-300; rf<-rep(100*c(6,9,12),50); Original_Data<-Full_Data$Complete_Data;

LCCsampling(r0 = r0, rf = rf, Y = as.matrix(Original_Data[,1]),
            X = as.matrix(Original_Data[,-1]),
```

```
            N = nrow(Original_Data))->Results

plot_Beta(Results)
```

`LeverageSampling` *Basic and shrinkage leverage sampling for Generalised Linear Models*

Description

Using this function sample from big data under linear, logistic and Poisson regression to describe the data. Sampling probabilities are obtained based on the basic and shrinkage leverage method.

Usage

```
LeverageSampling(rf,Y,X,N,S_alpha,family)
```

Arguments

`rf`	sample size
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`S_alpha`	shrinkage factor in between 0 and 1
`family`	a character vector for "linear", "logistic" and "poisson" regression from Generalised Linear Models

Details

Leverage sampling algorithm for big data under Generalised Linear Models (linear, logistic and Poisson regression).

First is to obtain a random sample of size $min(rf)/2$ and estimate the model parameters. Using the estimated parameters leverage scores are evaluated for leverage sampling.

Through the estimated leverage scores a sample of size rf was obtained. Finally, the sample of size rf is used and the model parameters are estimated.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If rf is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

If $0 < \alpha_S < 1$ is not satisfied an error message will be produced.

A character vector is provided for `family` and if it is not of the any three types an error message will be produced.

Value

The output of `LeverageSampling` gives a list of

`Beta_Estimates` estimated model parameters in a data.frame after sampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon in a data.frame after sampling (valid only for linear regression)

`Sample_Basic_Leverage` list of indexes for the optimal samples obtained based on basic leverage

`Sample_Shrinkage_Leverage` list of indexes for the optimal samples obtained based on shrinkage leverage

`Sampling_Probability` matrix of calculated sampling probabilities for basic and shrinkage leverage

References

Ma P, Mahoney M, Yu B (2014). "A statistical perspective on algorithmic leveraging." In *International conference on machine learning*, 91–99. PMLR.

Ma P, Sun X (2015). "Leveraging for big data regression." *Wiley Interdisciplinary Reviews: Computational Statistics*, **7**(1), 70–76.

Examples

```
Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1,Error_Variance=0.5)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000; Family<-"linear"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

rf<-rep(100*c(6,10),50); Original_Data<-Full_Data$Complete_Data;

LeverageSampling(rf = rf, Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 S_alpha = 0.95,
                 family = "linear")->Results

plot_Beta(Results)

Dist<-"Normal"; Dist_Par<-list(Mean=0,Variance=1)
No_Of_Var<-2; Beta<-c(-1,2,1); N<-5000; Family<-"logistic"
Full_Data<-GenGLMdata(Dist,Dist_Par,No_Of_Var,Beta,N,Family)

rf<-rep(100*c(6,10),25); Original_Data<-Full_Data$Complete_Data;

LeverageSampling(rf = rf, Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 S_alpha = 0.95,
                 family = "logistic")->Results

plot_Beta(Results)

Dist<-"Normal";
No_Of_Var<-2; Beta<-c(-1,0.5,0.5); N<-5000; Family<-"poisson"
Full_Data<-GenGLMdata(Dist,NULL,No_Of_Var,Beta,N,Family)
```

```
rf<-rep(100*c(6,10),25); Original_Data<-Full_Data$Complete_Data;

LeverageSampling(rf = rf, Y = as.matrix(Original_Data[,1]),
                 X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                 S_alpha = 0.95,
                 family = "poisson")->Results

plot_Beta(Results)
```

`modelMissLinSub` *Subsampling under linear regression for a potentially misspecified model*

Description

Using this function sample from big data under linear regression for a potentially misspecified model. Subsampling probabilities are obtained based on the A-, L- and L1- optimality criteria with the RLmAMSE (Reduction of Loss by minimizing the Average Mean Squared Error).

Usage

```
modelMissLinSub(r0,rf,Y,X,N,Alpha,proportion,model="Auto")
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`Alpha`	scaling factor when using Log Odds or Power functions to magnify the probabilities
`proportion`	a proportion of the big data is used to help estimate AMSE values from the subsamples
`model`	formula for the model used in the GAM or the default choice

Details

The article for this function is in preparation for publication. Please be patient.

Two stage subsampling algorithm for big data under linear regression for potential model misspecification.

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A-, L-, L1-optimality criteria, RLmAMSE and enhanced RLmAMSE (log-odds and power) subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for A-, L-, L1-optimality, RLmAMSE and enhanced RLmAMSE (log-odds and power).

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y,F_estimate_Full and if they are not aligned an error message will be produced.

If $\alpha > 1$ for the scaling factor is not satisfied an error message will be produced.

If proportion is not in the region of $(0, 1]$ an error message will be produced.

`model` is a formula input formed based on the covariates through the spline terms (s()), squared term (I()), interaction terms (lo()) or automatically. If `model` is empty or NA or NAN or not one of the defined inputs an error message is printed. As a default we have set `model="Auto"`, which is the main effects model wit the spline terms.

Value

The output of `modelMissLinSub` gives a list of

`Beta_Estimates` estimated model parameters after subsampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon after subsampling

`Utility_Estimates` estimated A-, L- and L1- optimality values for the obtained subsamples

`AMSE_Estimates` matrix of estimated AMSE values after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Sample_L-Optimality` list of indexes for the initial and optimal samples obtained based on L-Optimality criteria

`Sample_L1-Optimality` list of indexes for the initial and optimal samples obtained based on L1-Optimality criteria

`Sample_RLmAMSE` list of indexes for the optimal samples obtained based obtained based on RLmAMSE

`Sample_RLmAMSE_Log_Odds` list of indexes for the optimal samples obtained based on RLmAMSE with Log Odds function

`Sample_RLmAMSE_Power` list of indexes for the optimal samples obtained based on RLmAMSE with Power function

`Subsampling_Probability` matrix of calculated subsampling probabilities

References

Adewale AJ, Wiens DP (2009). "Robust designs for misspecified logistic models." *Journal of Statistical Planning and Inference*, **139**(1), 3–15.

Adewale AJ, Xu X (2010). "Robust designs for generalized linear models with possible overdispersion and misspecified link functions." *Computational statistics & data analysis*, **54**(4), 875–890.

Mahendran A, Thompson H, McGree JM (2025). "A subsampling approach for large data sets when the Generalised Linear Model is potentially misspecified." 2510.05902, https://arxiv.org/abs/2510.05902.

Examples

```
Beta <- c(-1, 0.75, 0.75, 1); Var_Epsilon <- 0.5;
family <- "linear"; N <- 500
X_1 <- replicate(2, stats::runif(n = N, min = -1, max = 1))

Temp <- Rfast::rowprods(X_1)
Misspecification <- (Temp - mean(Temp)) / sqrt(mean(Temp^2) - mean(Temp)^2)
X_Data <- cbind(X0 = 1, X_1)

Full_Data <- GenModelMissGLMdata(N, X_Data, Misspecification, Beta, Var_Epsilon, family)
r0 <- 40; rf <- rep(10 * c(8, 12), 25)
Original_Data <- Full_Data$Complete_Data[, -ncol(Full_Data$Complete_Data)]

Results <- modelMissLinSub(r0 = r0, rf = rf,
                           Y = as.matrix(Original_Data[, 1]),
                           X = as.matrix(Original_Data[, -1]),
                           N = N, Alpha = 10, proportion = 0.5)

plot_Beta(Results)
plot_AMSE(Results)
```

`modelMissLogSub` *Subsampling under logistic regression for a potentially misspecified model*

Description

Using this function sample from big data under logistic regression for a potentially misspecified model. Subsampling probabilities are obtained based on the A-, L- and L1- optimality criteria with the RLmAMSE (Reduction of Loss by minimizing the Average Mean Squared Error).

Usage

```
modelMissLogSub(r0,rf,Y,X,N,Alpha,proportion,model="Auto")
```

Arguments

r0	sample size for initial random sample
rf	final sample size including initial(r0) and optimal(r) samples
Y	response data or Y
X	covariate data or X matrix that has all the covariates (first column is for the intercept)
N	size of the big data
Alpha	scaling factor when using Log Odds or Power functions to magnify the probabilities
proportion	a proportion of the big data is used to help estimate AMSE values from the subsamples
model	formula for the model used in the GAM or the default choice

Details

The article for this function is in preparation for publication. Please be patient.

Two stage subsampling algorithm for big data under logistic regression for potential model misspecification.

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A-, L-, L1-optimality criteria, RLmAMSE and enhanced RLmAMSE(log-odds and power) subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for A-, L-, L1-optimality, RLmAMSE and enhanced RLmAMSE (log-odds and power).

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y,F_estimate_Full and if they are not aligned an error message will be produced.

If $\alpha > 1$ for the scaling vector is not satisfied an error message will be produced.

If proportion is not in the region of $(0, 1]$ an error message will be produced.

`model` is a formula input formed based on the covariates through the spline terms (s()), squared term (I()), interaction terms (lo()) or automatically. If `model` is empty or NA or NAN or not one of the defined inputs an error message is printed. As a default we have set `model="Auto"`, which is the main effects model wit the spline terms.

Value

The output of `modelMissLogSub` gives a list of

`Beta_Estimates` estimated model parameters after subsampling

`Utility_Estimates` estimated A-, L- and L1- optimality values for the obtained subsamples

`AMSE_Estimates` matrix of estimated AMSE values after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Sample_L-Optimality` list of indexes for the initial and optimal samples obtained based on L-Optimality criteria

`Sample_L1-Optimality` list of indexes for the initial and optimal samples obtained based on L1-Optimality criteria

`Sample_RLmAMSE` list of indexes for the optimal samples obtained based on RLmAMSE

`Sample_RLmAMSE_Log_Odds` list of indexes for the optimal samples obtained based on RLmAMSE with Log Odds function

`Sample_RLmAMSE_Power` list of indexes for the optimal samples obtained based on RLmAMSE with Power function

`Subsampling_Probability` matrix of calculated subsampling probabilities

References

Adewale AJ, Wiens DP (2009). "Robust designs for misspecified logistic models." *Journal of Statistical Planning and Inference*, **139**(1), 3–15.

Adewale AJ, Xu X (2010). "Robust designs for generalized linear models with possible overdispersion and misspecified link functions." *Computational statistics & data analysis*, **54**(4), 875–890.

Mahendran A, Thompson H, McGree JM (2025). "A subsampling approach for large data sets when the Generalised Linear Model is potentially misspecified." 2510.05902, https://arxiv.org/abs/2510.05902.

Examples

```
Beta<-c(-1,0.75,0.75,1); family <- "logistic"; N<-100
X_1 <- replicate(2,stats::runif(n=N,min = -1,max = 1))

Temp<-Rfast::rowprods(X_1)
Misspecification <- (Temp-mean(Temp))/sqrt(mean(Temp^2)-mean(Temp)^2)
X_Data <- cbind(X0=1,X_1);

Full_Data<-GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon=NULL,family)
r0 <- 20; rf <- rep(10 * c(4, 6), 25)
Original_Data<-Full_Data$Complete_Data[,-ncol(Full_Data$Complete_Data)];

Results<-modelMissLogSub(r0 = r0, rf = rf,
                         Y = as.matrix(Original_Data[,1]),
                         X = as.matrix(Original_Data[,-1]),
                         N = N, Alpha = 10, proportion = 0.3)

plot_Beta(Results)
plot_AMSE(Results)
```

`modelMissPoiSub` *Subsampling under Poisson regression for a potentially misspecified model*

Description

Using this function sample from big data under Poisson regression for a potentially misspecified model. Subsampling probabilities are obtained based on the A-, L- and L1- optimality criteria with the RLmAMSE (Reduction of Loss by minimizing the Average Mean Squared Error).

Usage

```
modelMissPoiSub(r0,rf,Y,X,N,Alpha,proportion,model="Auto")
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`Alpha`	scaling factor when using Log Odds or Power functions to magnify the probabilities
`proportion`	a proportion of the big data is used to help estimate AMSE values from the subsamples
`model`	formula for the model used in the GAM or the default choice

Details

The article for this function is in preparation for publication. Please be patient.

Two stage subsampling algorithm for big data under Poisson regression for potential model misspecification.

First stage is to obtain a random sample of size r_0 and estimate the model parameters. Using the estimated parameters subsampling probabilities are evaluated for A-, L-, L1-optimality criteria, RLmAMSE and enhanced RLmAMSE (log-odds and power) subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for A-, L-, L1-optimality, RLmAMSE and enhanced RLmAMSE (log-odds and power).

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y,F_estimate_Full and if they are not aligned an error message will be produced.

If $\alpha > 1$ for the scaling vector is not satisfied an error message will be produced.

If proportion is not in the region of $(0, 1]$ an error message will be produced.

`model` is a formula input formed based on the covariates through the spline terms (s()), squared term (I()), interaction terms (lo()) or automatically. If `model` is empty or NA or NAN or not one of the defined inputs an error message is printed. As a default we have set `model="Auto"`, which is the main effects model wit the spline terms.

Value

The output of `modelMissPoiSub` gives a list of

`Beta_Estimates` estimated model parameters after subsampling

`Utility_Estimates` estimated A-, L- and L1- optimality values for the obtained subsamples

`AMSE_Estimates` matrix of estimated AMSE values after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Sample_L-Optimality` list of indexes for the initial and optimal samples obtained based on L-Optimality criteria

`Sample_L1-Optimality` list of indexes for the initial and optimal samples obtained based on L1-Optimality criteria

`Sample_RLmAMSE` list of indexes for the optimal samples obtained based on RLmAMSE

`Sample_RLmAMSE_Log_Odds` list of indexes for the optimal samples obtained based on RLmAMSE with Log Odds function

`Sample_RLmAMSE_Power` list of indexes for the optimal samples obtained based on RLmAMSE with Power function

`Subsampling_Probability` matrix of calculated subsampling probabilities

References

Adewale AJ, Wiens DP (2009). "Robust designs for misspecified logistic models." *Journal of Statistical Planning and Inference*, **139**(1), 3–15.

Adewale AJ, Xu X (2010). "Robust designs for generalized linear models with possible overdispersion and misspecified link functions." *Computational statistics & data analysis*, **54**(4), 875–890.

Mahendran A, Thompson H, McGree JM (2025). "A subsampling approach for large data sets when the Generalised Linear Model is potentially misspecified." 2510.05902, https://arxiv.org/abs/2510.05902.

Examples

```
Beta<-c(-1,0.75,0.75,1); family <- "poisson"; N<-100
X_1 <- replicate(2,stats::runif(n=N,min = -1,max = 1))

Temp<-Rfast::rowprods(X_1)
Misspecification <- (Temp-mean(Temp))/sqrt(mean(Temp^2)-mean(Temp)^2)
```

```
X_Data <- cbind(X0=1,X_1);

Full_Data<-GenModelMissGLMdata(N,X_Data,Misspecification,Beta,Var_Epsilon=NULL,family)
r0 <- 20; rf <- rep(10 * c(4, 6), 25)
Original_Data<-Full_Data$Complete_Data[,-ncol(Full_Data$Complete_Data)];

Results<-modelMissPoiSub(r0 = r0, rf = rf,
                         Y = as.matrix(Original_Data[,1]),
                         X = as.matrix(Original_Data[,-1]),
                         N = N, Alpha = 10, proportion = 0.3)

plot_Beta(Results)
plot_AMSE(Results)
```

`modelRobustLinSub` *Model robust optimal subsampling for A- and L- optimality criteria under linear regression*

Description

Using this function sample from big data under linear regression when there are more than one model to describe the data. Subsampling probabilities are obtained based on the A- and L- optimality criteria.

Usage

```
modelRobustLinSub(r0,rf,Y,X,N,Apriori_probs,All_Combinations,All_Covariates)
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`Apriori_probs`	vector of a priori model probabilities that are used to obtain the model robust subsampling probabilities
`All_Combinations`	list of possible models that can describe the data
`All_Covariates`	all the covariates in the models

Details

Two stage subsampling algorithm for big data under linear regression for multiple models that can describe the big data.

First stage is to obtain a random sample of size r_0 and estimate the model parameters for all models. Using the estimated parameters subsampling probabilities are evaluated for A-, L-optimality criteria and model averaging A-, L-optimality subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for all the models.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

If $0 < \alpha_q < 1$ for the a priori model probabilities are not satisfied an error message will be produced, where $q = 1, \ldots, Q$ and Q is the number of models in the model set.

Value

The output of `modelRobustLinSub` gives a list of

`Beta_Estimates` estimated model parameters for each model in a list after subsampling

`Variance_Epsilon_Estimates` matrix of estimated variance for epsilon for each model after subsampling

`Sample_A-Optimality` list of indexes for the initial and optimal samples obtained based on A-Optimality criteria

`Sample_A-Optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on A-Optimality criteria

`Sample_L-Optimality` list of indexes for the initial and optimal samples obtained based on L-Optimality criteria

`Sample_L-Optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on L-Optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A- and L- optimality criteria

References

Mahendran A, Thompson H, McGree JM (2023). "A model robust subsampling approach for Generalised Linear Models in big data settings." *Statistical Papers*, **64**(4), 1137–1157.

Examples

```
indexes<-1:ceiling(nrow(Electric_consumption)*0.005)
Original_Data<-cbind(Electric_consumption[indexes,1],1,
                     Electric_consumption[indexes,-1])
```

```
colnames(Original_Data)<-c("Y",paste0("X",0:ncol(Original_Data[,-c(1,2)])))
for (j in 3:5) {
  Original_Data[,j]<-scale(Original_Data[,j])
}

No_of_Variables<-ncol(Original_Data[,-c(1,2)])
Squared_Terms<-paste0("X",1:No_of_Variables,"^2")
term_no <- 2
All_Models <- list(c("X0",paste0("X",1:No_of_Variables)))

Original_Data<-cbind(Original_Data,Original_Data[,-c(1,2)]^2)
colnames(Original_Data)<-c("Y","X0",paste0("X",1:No_of_Variables),
                           paste0("X",1:No_of_Variables,"^2"))

for (i in 1:No_of_Variables){
  x <- as.vector(combn(Squared_Terms,i,simplify = FALSE))
    for(j in 1:length(x)){
       All_Models[[term_no]] <- c("X0",paste0("X",1:No_of_Variables),x[[j]])
       term_no <- term_no+1
     }
   }

All_Models<-All_Models[c(1,12:16)]
names(All_Models)<-paste0("Model_",1:length(All_Models))

r0<-300; rf<-rep(100*c(6,9),50);

modelRobustLinSub(r0 = r0, rf = rf, Y = as.matrix(Original_Data[,1]),
                  X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                  Apriori_probs = rep(1/length(All_Models),length(All_Models)),
                  All_Combinations = All_Models,
                  All_Covariates = colnames(Original_Data)[-1])->Results

Beta_Plots<-plot_Beta(Results)
```

`modelRobustLogSub` *Model robust optimal subsampling for A- and L- optimality criteria under logistic regression*

Description

Using this function sample from big data under logistic regression when there are more than one model to describe the data. Subsampling probabilities are obtained based on the A- and L- optimality criteria.

Usage

```
modelRobustLogSub(r0,rf,Y,X,N,Apriori_probs,All_Combinations,All_Covariates)
```

Arguments

r0	sample size for initial random sample
rf	final sample size including initial(r0) and optimal(r) samples
Y	response data or Y
X	covariate data or X matrix that has all the covariates (first column is for the intercept)
N	size of the big data
Apriori_probs	vector of a priori model probabilities that are used to obtain the model robust subsampling probabilities
All_Combinations	list of possible models that can describe the data
All_Covariates	all the covariates in the models

Details

Two stage subsampling algorithm for big data under logistic regression for multiple models that can describe the big data.

First stage is to obtain a random sample of size r_0 and estimate the model parameters for all models. Using the estimated parameters subsampling probabilities are evaluated for A-, L-optimality criteria and model averaging A-, L-optimality subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for all the models.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

If $0 < \alpha_q < 1$ for the a priori model probabilities are not satisfied an error message will be produced, where $q = 1, \ldots, Q$ and Q is the number of models in the model set.

Value

The output of `modelRobustLinSub` gives a list of

`Beta_Data` estimated model parameters for each model in a list after subsampling

`Sample_L-optimality` list of indexes for the initial and optimal samples obtained based on L-optimality criteria

`Sample_L-optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on L-optimality criteria

`Sample_A-optimality` list of indexes for the initial and optimal samples obtained based on A-optimality criteria

`Sample_A-optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on A-optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A- and L- optimality criteria

References

Mahendran A, Thompson H, McGree JM (2023). "A model robust subsampling approach for Generalised Linear Models in big data settings." *Statistical Papers*, **64**(4), 1137–1157.

Examples

```
indexes<-1:ceiling(nrow(Skin_segmentation)*0.25)
Original_Data<-cbind(Skin_segmentation[indexes,1],1,Skin_segmentation[indexes,-1])
colnames(Original_Data)<-c("Y",paste0("X",0:ncol(Original_Data[,-c(1,2)])))

for (j in 3:5) {
  Original_Data[,j]<-scale(Original_Data[,j])
}
No_of_Variables<-ncol(Original_Data[,-c(1,2)])

Squared_Terms<-paste0("X",1:No_of_Variables,"^2")
term_no <- 2
All_Models <- list(c("X0",paste0("X",1:No_of_Variables)))

Original_Data<-cbind(Original_Data,Original_Data[,-c(1,2)]^2)
colnames(Original_Data)<-c("Y","X0",paste0("X",1:No_of_Variables),
                           paste0("X",1:No_of_Variables,"^2"))

for (i in 1:No_of_Variables){
  x <- as.vector(combn(Squared_Terms,i,simplify = FALSE))
  for(j in 1:length(x)){
    All_Models[[term_no]] <- c("X0",paste0("X",1:No_of_Variables),x[[j]])
    term_no <- term_no+1
  }
}
All_Models<-All_Models[-c(5:7)]
names(All_Models)<-paste0("Model_",1:length(All_Models))

r0<-300; rf<-rep(100*c(6,9),50);

modelRobustLogSub(r0 = r0, rf = rf, Y = as.matrix(Original_Data[,1]),
                  X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                  Apriori_probs = rep(1/length(All_Models),length(All_Models)),
                  All_Combinations = All_Models,
                  All_Covariates = colnames(Original_Data)[-1])->Results

Beta_Plots<-plot_Beta(Results)
```

`modelRobustPoiSub` *Model robust optimal subsampling for A- and L- optimality criteria under Poisson regression*

Description

Using this function sample from big data under Poisson regression when there are more than one model to describe the data. Subsampling probabilities are obtained based on the A- and L- optimality criteria.

Usage

```
modelRobustPoiSub(r0,rf,Y,X,N,Apriori_probs,All_Combinations,All_Covariates)
```

Arguments

`r0`	sample size for initial random sample
`rf`	final sample size including initial(r0) and optimal(r) samples
`Y`	response data or Y
`X`	covariate data or X matrix that has all the covariates (first column is for the intercept)
`N`	size of the big data
`Apriori_probs`	vector of a priori model probabilities that are used to obtain the model robust subsampling probabilities
`All_Combinations`	list of possible models that can describe the data
`All_Covariates`	all the covariates in the models

Details

Two stage subsampling algorithm for big data under Poisson regression for multiple models that can describe the big data.

First stage is to obtain a random sample of size r_0 and estimate the model parameters for all models. Using the estimated parameters subsampling probabilities are evaluated for A-, L-optimality criteria and model averaging A-, L-optimality subsampling methods.

Through the estimated subsampling probabilities a sample of size $r \geq r_0$ is obtained. Finally, the two samples are combined and the model parameters are estimated for all the models.

NOTE : If input parameters are not in given domain conditions necessary error messages will be provided to go further.

If $r \geq r_0$ is not satisfied then an error message will be produced.

If the big data X, Y has any missing values then an error message will be produced.

The big data size N is compared with the sizes of X, Y and if they are not aligned an error message will be produced.

If $0 < \alpha_q < 1$ for the a priori model probabilities are not satisfied an error message will be produced, where $q = 1, \ldots, Q$ and Q is the number of models in the model set.

Value

The output of `modelRobustLinSub` gives a list of

`Beta_Data` estimated model parameters for each model in a list after subsampling

`Sample_L-optimality` list of indexes for the initial and optimal samples obtained based on L-optimality criteria

`Sample_L-optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on L-optimality criteria

`Sample_A-optimality` list of indexes for the initial and optimal samples obtained based on A-optimality criteria

`Sample_A-optimality_MR` list of indexes for the initial and model robust optimal samples obtained based on A-optimality criteria

`Subsampling_Probability` matrix of calculated subsampling probabilities for A- and L- optimality criteria

References

Mahendran A, Thompson H, McGree JM (2023). "A model robust subsampling approach for Generalised Linear Models in big data settings." *Statistical Papers*, **64**(4), 1137–1157.

Examples

```
indexes<-1:ceiling(nrow(One_million_songs)*0.5)
Original_Data<-One_million_songs[indexes,]
colnames(Original_Data)<-c("Y",paste0("X",1:ncol(Original_Data[,-1])))

# Scaling the covariate data
for (j in 2:4) {
  Original_Data[,j]<-scale(Original_Data[,j])
}

No_of_Variables<-ncol(Original_Data[,-1])
Squared_Terms<-paste0("X",1:No_of_Variables,"^2")
term_no <- 2
All_Models <- list(paste0("X",1:No_of_Variables))

Original_Data<-cbind(Original_Data,Original_Data[,-1]^2)
colnames(Original_Data)<-c("Y",paste0("X",1:No_of_Variables),
                           paste0("X",1:No_of_Variables,"^2"))

for (i in 1:No_of_Variables)
{
  x <- as.vector(combn(Squared_Terms,i,simplify = FALSE))
  for(j in 1:length(x))
  {
    All_Models[[term_no]] <- c(paste0("X",1:No_of_Variables),x[[j]])
    term_no <- term_no+1
  }
}
All_Models<-All_Models[1:4]
```

```
names(All_Models)<-paste0("Model_",1:length(All_Models))

r0<-300; rf<-rep(100*c(6,9),25);

modelRobustPoiSub(r0 = r0, rf = rf, Y = as.matrix(Original_Data[,1]),
                  X = as.matrix(Original_Data[,-1]),N = nrow(Original_Data),
                  Apriori_probs = rep(1/length(All_Models),length(All_Models)),
                  All_Combinations = All_Models,
                  All_Covariates = colnames(Original_Data)[-1])->Results

Beta_Plots<-plot_Beta(Results)
```

`One_million_songs` *One million songs data*

Description

This data set contains 1,019,318 unique users' music play counts in the Echo Nest, which is available at "http://millionsongdataset.com/tasteprofile/". As a basic step, it is interesting to predict the play counts using the song information collected in the Million Song Dataset (Bertin-Mahieux et al. (2011)). After cleaning up and feature engineering the data in total contains 205,032 observations where we consider the covariates duration, loudness, tempo, artist hotness and song hotness to model the response, the number of song counts.

Usage

```
One_million_songs
```

Format

A data frame with 4 columns and 309,685 rows.

`Counts` Number of playback counts for songs

`Duration` Duration of the song

`Loudness` Loudness of the song

`Tempo` Tempo of the song

`Artist_Hotness` A value between 0 and 1

`Song_Hotness` A value between 0 and 1

References

McFee B, Bertin-Mahieux T, Ellis DP, Lanckriet GR (2012). "The million song dataset challenge." In *Proceedings of the 21st International Conference on World Wide Web*, 909–916.

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

Examples

```
nrow(One_million_songs)
```

`plot_AMSE` *Plotting AMSE outputs for the samples under model misspecification*

Description

After using the subsampling methods under potential model misspecification we obtain their respective AMSE values for the predictions. They are summarised as plots here.

Usage

```
plot_AMSE(object)
```

Arguments

`object`	Any object after subsampling from our subsampling function under potential model misspecification

Details

For A- and L-optimality criteria and RLmAMSE subsampling under Generalised Linear Models with potential model misspecification the facets are for variance and bias^2 of AMSE values.

Value

The output is a faceted ggplot result

`plot_Beta` *Plotting model parameter outputs after subsampling*

Description

After using the subsampling methods we mostly obtain the estimated model parameter estimates. Here, they are summarised as histogram plots.

Usage

```
plot_Beta(object)
```

Arguments

`object`	Any object after subsampling from our subsampling functions

Details

For local case control sampling the facets are for sample sizes and beta values.

For leverage sampling the facets are for sample sizes and beta values.

For A- and L-optimality criteria subsampling under Generalised Linear Models the facets are for sample sizes and beta values.

For A-optimality criteria subsampling under Gaussian Linear Models the facets are for sample sizes and beta values.

For A-optimality criteria subsampling under Generalised Linear Models with response variable not inclusive the facets are for sample sizes and beta values.

For A- and L-optimality criteria subsampling under Generalised Linear Models where multiple models can describe the data the facets are for sample sizes and beta values.

For A- and L-optimality criteria and LmAMSE subsampling under Generalised Linear Models with potential model misspecification the facets are for sample sizes and beta values.

Value

The output is a faceted ggplot result

Skin_segmentation *Skin segmentation data*

Description

Rajen and Abhinav (2012) addressed the challenge of detecting skin-like regions in images as a component of the intricate process of facial recognition. To achieve this goal, they curated the "Skin segmentation" data set, comprising RGB (R-red, G-green, B-blue) values of randomly selected pixels from N = 245,057 facial images, including 50,859 skin samples and 194,198 nonskin samples, spanning diverse age groups, racial backgrounds, and genders.

Usage

```
Skin_segmentation
```

Format

A data frame with 4 columns and 245,057 rows.

`Skin_presence` Skin presence in the randomly selected pixels

`Red` Red values in the randomly selected pixels

`Green` Green values in the randomly selected pixels

`Blue` Blue values in the randomly selected pixels

Source

Extracted from

Rajen B, Abhinav D (2012) Skin segmentation. UCI Machine Learning Repository.

Available at: doi:10.24432/C5T30C

Examples

```
nrow(Skin_segmentation)
```

Index