Package ‘glmtlp’

October 13, 2022

Type Package
Title Generalized Linear Models with Truncated Lasso Penalty
Version 2.0.1
Date 2021-12-17

URL https://yuyangyy.com/glmtlp/
Depends R (>= 3.5.0)
Imports foreach, doParallel, ggplot2
Suggests rmarkdown, knitr, testthat (>= 3.0.0)

Description Extremely efficient procedures for fitting regularization path with l0, l1, and truncated lasso penalty for linear regression and logistic regression models. This version is a completely new version compared with our previous version, which was mainly based on R. New core algorithms are developed and are now written in C++ and highly optimized.

Encoding UTF-8
License GPL-3
LazyData true

Author Chunlin Li [aut],
Yu Yang [aut, cre],
Chong Wu [aut]

Maintainer Yu Yang <yang6367@umn.edu>

RoxygenNote 7.1.2
VignetteBuilder knitr

Config/testthat.edition 3

Date/Publication 2021-12-17 23:00:02 UTC
R topics documented:

bin_data ................................................................. 2
cv.glmtlp ............................................................... 3
gau_data ................................................................. 5
gen.binomial.data ...................................................... 5
gen.gaussian.data ....................................................... 6
glmtlp ................................................................. 7
plot.cv.glmtlp ....................................................... 11
plot.glmtlp ........................................................... 12
predict.cv.glmtlp ................................................... 14
predict.glmtlp ....................................................... 15
setup_lambda ......................................................... 18

Index 19

bin_data A simulated binomial data set.

Description
A data set simulated for illustrating logistic regression models. Generated by gen.binomial.data(n = 200, p = 20, seed = 2021).

Usage
data(bin_data)

Format
A list with three elements: design matrix X, response y, and the true coefficient vector beta.

X design matrix
y response
beta the true coefficient vector

Examples
data("bin_data")
cv.fit <- cv.glmtlp(bin_data$X, bin_data$y, family = "binomial", penalty = "l1")
plot(cv.fit)
Description

Performs k-fold cross-validation for l0, l1, or TLP-penalized regression models over a grid of values for the regularization parameter lambda (if penalty="l0") or kappa (if penalty="l0").

Usage

```
cv.glmtlp(X, y, ..., seed = NULL, nfolds = 10, obs.fold = NULL, ncores = 1)
```

Arguments

- `X` input matrix, of dimension nob x nvars, as in glmtlp.
- `y` response, of length nob, as in glmtlp.
- `...` Other arguments that can be passed to glmtlp.
- `seed` the seed for reproduction purposes
- `nfolds` number of folds; default is 10. The smallest value allowable is nfolds=3
- `obs.fold` an optional vector of values between 1 and nfolds identifying what fold each observation is in. If supplied, nfolds can be missing.
- `ncores` number of cores utilized; default is 1. If greater than 1, then doParallel::foreach will be used to fit each fold; if equal to 1, then for loop will be used to fit each fold. Users don’t have to register parallel clusters outside.

Details

The function calls glmtlp nfolds+1 times; the first call to get the lambda or kappa sequence, and then the rest to compute the fit with each of the folds omitted. The cross-validation error is based on deviance (check here for more details). The error is accumulated over the folds, and the average error and standard deviation is computed.

When family = "binomial", the fold assignment (if not provided by the user) is generated in a stratified manner, where the ratio of 0/1 outcomes are the same for each fold.

Value

an object of class "cv.glmtlp" is returned, which is a list with the ingredients of the cross-validation fit.

- `call` the function call
- `cv.mean` The mean cross-validated error - a vector of length length(kappa) if penalty = "l0" and length(lambda) otherwise.
- `cv.se` estimate of standard error of cv.mean.
- `fit` a fitted glmtlp object for the full data.
idx.min  the index of the lambda or kappa sequence that corresponding to the smallest cv mean error.
kappa  the values of kappa used in the fits, available when penalty = 'l0'.
kappa.min  the value of kappa that gives the minimum cv.mean, available when penalty = 'l0'.
lambda  the values of lambda used in the fits.
lambda.min  value of lambda that gives minimum cv.mean, available when penalty is 'l1' or 'tlp'.
null.dev  null deviance of the model.
obs.fold  the fold id for each observation used in the CV.

Author(s)
Chunlin Li, Yu Yang, Chong Wu
Maintainer: Yu Yang <yang6367@umn.edu>

References
Two R package Github: ncvreg and glmnet.

See Also
glmtlp and plot, predict, and coef methods for "cv.glmtlp" objects.

Examples

# Gaussian
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
cv.fit <- cv.glmtlp(X, y, family = "gaussian", penalty = "l1", seed=2021)

# Binomial
X <- matrix(rnorm(100 * 20), 100, 20)
y <- sample(c(0,1), 100, replace = TRUE)
cv.fit <- cv.glmtlp(X, y, family = "binomial", penalty = "l1", seed=2021)
**gau_data**

A simulated gaussian data set.

**Description**

A data set simulated for illustrating linear regression models. Generated by `gen.gaussian.data(n = 200, p = 20, seed = 2021)`.

**Usage**

```r
data(gau_data)
```

**Format**

A list with five elements: design matrix $X$, response $y$, correlation structure of the covariates $\Sigma$, true beta $\beta$, and the noise level $\sigma$.

- $X$ design matrix
- $y$ response
- $\beta$ true beta values
- $\sigma$ the noise level

**Examples**

```r
data("gau_data")
cv.fit <- cv.glmtlp(gau_data$X, gau_data$y, family = "gaussian", penalty = "tlp")
plot(cv.fit)
```

**gen.binomial.data**

Simulate a binomial data set.

**Description**

Simulate a data set with binary response following the logistic regression model.

**Usage**

```r
gen.binomial.data(n, p, rho = 0, kappa = 5, beta.type = 1, seed = 2021)
```
## gen.gaussian.data

Simulate a gaussian data set

### Arguments

- **n**  
  Sample size.

- **p**  
  Number of covariates.

- **rho**  
  The parameter defining the AR(1) correlation matrix.

- **kappa**  
  The number of nonzero coefficients.

- **beta.type**  
  Numeric indicator for choosing the beta type. For beta.type = 1, the true coefficient vector has kappa components being 1, roughly equally distributed between 1 to p. For beta.type = 2, the first kappa values are 1, and the rest are 0. For beta.type = 3, the first kappa values are equally-spaced values from 10 to 0.5, and the rest are 0. For beta.type = 4, the first kappa values are the first kappa values in c(-10, -6, -2, 2, 6, 10), and the rest are 0. For beta.type = 5, the first kappa values are 1, and the rest decay exponentially to 0 with base 0.5.

- **seed**  
  The seed for reproducibility. Default is 2021.

### Value

A list containing the simulated data.

- **X**  
  the covariate matrix, of dimension n x p.

- **y**  
  the response, of length n.

- **beta**  
  the true coefficients, of length p.

### Examples

```r
bin_data <- gen.binomial.data(n = 200, p = 20, seed = 2021)
head(bin_data$X)
head(bin_data$y)
head(bin_data$beta)
```

### Description

Simulate a data set with gaussian response following the linear regression model.

### Usage

```r
gen.gaussian.data(
  n,
  p,
  rho = 0,
  kappa = 5,
  beta.type = 1,
  snr = 1,
  seed = 2021
)
```
Arguments

- **n**: Sample size.
- **p**: Number of covariates.
- **rho**: The parameter defining the AR(1) correlation matrix.
- **kappa**: The number of nonzero coefficients.
- **beta.type**: Numeric indicator for choosing the beta type. For `beta.type = 1`, the true coefficient vector has `kappa` components being 1, roughly equally distributed between 1 to `p`. For `beta.type = 2`, the first `kappa` values are 1, and the rest are 0. For `beta.type = 3`, the first `kappa` values are equally-spaced values from 10 to 0.5, and the rest are 0. For `beta.type = 4`, the first `kappa` values are the first `kappa` values in c(-10, -6, -2, 2, 6, 10), and the rest are 0. For `beta.type = 5`, the first `kappa` values are 1, and the rest decay exponentially to 0 with base 0.5.
- **snr**: Signal-to-noise ratio. Default is 1.
- **seed**: The seed for reproducibility. Default is 2021.

Value

A list containing the simulated data.

- **X**: the covariate matrix, of dimension `n x p`.
- **y**: the response, of length `n`.
- **beta**: the true coefficients, of length `p`.
- **sigma**: the standard error of the noise.

Examples

gau_data <- gen.gaussian.data(n = 200, p = 20, seed = 2021)
head(gau_data$X)
head(gau_data$y)
head(gau_data$beta)
gau_data$sigma

glmtp

*Fit a GLM with L0, L1, or TLP Penalization*

Description

Fit generalized linear models via penalized maximum likelihood. The regularization path is computed for the l0, lasso, or truncated lasso penalty at a grid of values for the regularization parameter `lambda` or `kappa`. Fits linear and logistic regression models.

The package provides 3 penalties: l0, l1, and tlp and 3 distribution families: gaussian, binomial, and poisson.
Usage

```r
glmtlp(
  X,
  y,
  family = c("gaussian", "binomial"),
  penalty = c("l0", "l1", "tlp"),
  nlambda = ifelse(penalty == "l0", 50, 100),
  lambda.min.ratio = ifelse(nobs < nvars, 0.05, 0.001),
  lambda = NULL,
  kappa = NULL,
  tau = 0.3 * sqrt(log(nvars)/nobs),
  delta = 2,
  tol = 1e-04,
  weights = NULL,
  penalty.factor = rep(1, nvars),
  standardize = FALSE,
  dc.maxit = 20,
  cd.maxit = 10000,
  nr.maxit = 20,
  ...
)
```

Arguments

- **X**: Input matrix, of dimension nobs x nvars; each row is an observation vector.
- **y**: Response variable, of length nobs. For `family="gaussian"`, it should be quantitative; for `family="binomial"`, it should be either a factor with two levels or a binary vector.
- **family**: A character string representing one of the built-in families. See Details section below.
- **penalty**: A character string representing one of the built-in penalties. "l0" represents the L0 penalty, "l1" represents the lasso-type penalty (L1 penalty), and "tlp" represents the truncated lasso penalty.
- **nlambda**: The number of lambda values. Default is 100.
- **lambda.min.ratio**: The smallest value for lambda, as a fraction of lambda.max, the smallest value for which all coefficients are zero. The default depends on the sample size nobs relative to the number of variables nvars. If nobs > nvars, the default is 0.0001, and if nobs < nvars, the default is 0.01.
- **lambda**: A user-supplied lambda sequence. Typically, users should let the program compute its own lambda sequence based on nlambdas and lambda.min.ratio. Supplying a value of lambda will override this. WARNING: please use this option with care. glmtlp relies on warms starts for speed, and it's often faster to fit a whole path than a single fit. Therefore, provide a decreasing sequence of lambda values if you want to use this option. Also, when penalty = 'l0', it is not recommended for the users to supply this parameter.
kappa A user-supplied kappa sequence. Typically, users should let the program compute its own kappa sequence based on nvars and nob. This sequence is used when penalty = 'l0'.

tau A tuning parameter used in the TLP-penalized regression models. Default is $0.3 \times \sqrt{\log(nvars)/nobs}$.


tol Tolerance level for all iterative optimization algorithms.

weights Observation weights. Default is 1 for each observation.

penalty.factor Separate penalty factors applied to each coefficient, which allows for differential shrinkage. Default is 1 for all variables.

standardize Logical. Whether or not standardize the input matrix X; default is TRUE.

dc.maxit Maximum number of iterations for the DC (Difference of Convex Functions) programming; default is 20.

cd.maxit Maximum number of iterations for the coordinate descent algorithm; default is $10^4$.

nr.maxit Maximum number of iterations for the Newton-Raphson method; default is 500.

... Additional arguments.

Details

The sequence of models indexed by lambda (when penalty = c('l1', 'tlp')) or kappa (when penalty = 'l0') is fit by the coordinate descent algorithm.

The objective function for the "gaussian" family is:

$$1/2 RSS/nobs + \lambda \times penalty,$$

and for the other models it is:

$$-loglik/nobs + \lambda \times penalty.$$

Also note that, for "gaussian", glmtlp standardizes y to have unit variance (using $1/(n-1)$ formula).

## Details on family option

glmtlp currently only supports built-in families, which are specified by a character string. For all families, the returned object is a regularization path for fitting the generalized linear regression models, by maximizing the corresponding penalized log-likelihood. glmtlp(..., family="binomial") fits a traditional logistic regression model for the log-odds.

## Details on penalty option

The built-in penalties are specified by a character string. For l0 penalty, kappa sequence is used for generating the regularization path, while for l1 and tlp penalty, lambda sequence is used for generating the regularization path.
Value

An object with S3 class "glmtlp".

- **beta**: a \( nvars \times \text{length}(\kappa) \) matrix of coefficients when \( \text{penalty} = \text{"l0"} \); or a \( nvars \times \text{length}(\lambda) \) matrix of coefficients when \( \text{penalty} = \text{c(\"l1\", \"tlp\")} \).
- **call**: the call that produces this object.
- **family**: the distribution family used in the model fitting.
- **intercept**: the intercept vector, of length(\( \kappa \)) when \( \text{penalty} = \text{\"l0\"} \) or length(\( \lambda \)) when \( \text{penalty} = \text{c(\"l1\", \"tlp\")} \).
- **lambda**: the actual sequence of lambda values used. Note that the length may be smaller than the provided nlambda due to removal of saturated values.
- **penalty**: the penalty type in the model fitting.
- **penalty.factor**: the penalty factor for each coefficient used in the model fitting.
- **tau**: the tuning parameter used in the model fitting, available when \( \text{penalty} = \text{\"tlp\"} \).

**glmtlp functions**

- ‘glmtlp()’, ‘cv.glmtlp()’

**Author(s)**

Chunlin Li, Yu Yang, Chong Wu

Maintainer: Yu Yang <yang6367@umn.edu>

**References**


Two R package Github: *ncvreg* and *glmnet*.

**See Also**

print, predict, coef and plot methods, and the cv.glmtlp function.

**Examples**

```r
# Gaussian
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
fit1 <- glmtlp(X, y, family = "gaussian", penalty = "l0")
fit2 <- glmtlp(X, y, family = "gaussian", penalty = "l1")
```
fit3 <- glmtlp(X, y, family = "gaussian", penalty = "tlp")

# Binomial
X <- matrix(rnorm(100 * 20), 100, 20)
y <- sample(c(0, 1), 100, replace = TRUE)
fit <- glmtlp(X, y, family = "binomial", penalty = "l1")

---

**Description**

Plots the cross-validation curve, and the upper and lower standard deviation curves, as a function of the lambda or kappa values.

**Usage**

```r
## S3 method for class 'cv.glmtlp'
plot(x, vertical.line = TRUE, ...)
```

**Arguments**

- `x`: Fitted cv.glmtlp object
- `vertical.line`: Logical. Whether or not include a vertical line indicating the position of the index which gives the smallest CV error.
- `...`: Additional arguments.

**Details**

The generated plot is a ggplot object, and therefore, the users are able to customize the plots following the ggplot2 syntax.

**Author(s)**

Chunlin Li, Yu Yang, Chong Wu
Maintainer: Yu Yang <yang6367@umn.edu>

**References**

Two R package Github: `ncvreg` and `glmnet`. 
Examples

```r
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
cv.fit <- cv.glmtlp(X, y, family = "gaussian", penalty = "tlp")
plot(cv.fit)
plot(cv.fit, vertical.line = FALSE)
cv.fit2 <- cv.glmtlp(X, y, family = "gaussian", penalty = "l0")
plot(cv.fit2)
plot(cv.fit2, vertical.line = FALSE)

data("gau_data")
cv.fit <- cv.glmtlp(gau_data$X, gau_data$y, family = "gaussian", penalty = "tlp")
plot(cv.fit)

data("bin_data")
cv.fit <- cv.glmtlp(bin_data$X, bin_data$y, family = "binomial", penalty = "l1")
plot(cv.fit)
```

---

**plot.glmtlp**

*Plot Method for a "glmtlp" Object*

**Description**

Generates a solution path plot for a fitted "glmtlp" object.

**Usage**

```r
## S3 method for class 'glmtlp'
plot(
x,  
  xvar = c("lambda", "kappa", "deviance", "l1_norm", "log_lambda"),  
  xlab = iname,  
  ylab = "Coefficients",  
  title = "Solution Path",  
  label = FALSE,  
  label.size = 3,  
  ...)
```

**Arguments**

- `x` Fitted glmtlp object.
- `xvar` The x-axis variable to plot against, including "lambda", "kappa", "deviance", "l1_norm", and "log_lambda".
- `xlab` The x-axis label of the plot, default is "Lambda", "Kappa", "Fraction of Explained Deviance", "L1 Norm", and "Log Lambda".
plot.glmtlp

ylab The y-axis label of the plot, default is "Coefficients".
title The main title of the plot, default is "Solution Path".
label Logical, whether or not attach the labels for the non-zero coefficients, default is FALSE.
label.size The text size of the labels, default is 3.
... Additional arguments.

Details

The generated plot is a ggplot object, and therefore, the users are able to customize the plots following the ggplot2 syntax.

Value

A ggplot object.

Author(s)

Chunlin Li, Yu Yang, Chong Wu
Maintainer: Yu Yang <yang6367@umn.edu>

References

Two R package Github: ncvreg and glmnet.

See Also

print, predict, coef and plot methods, and the cv.glmtlp function.

Examples

```r
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
fit <- glmtlp(X, y, family = "gaussian", penalty = "l1")
plot(fit, xvar = "lambda")
plot(fit, xvar = "log_lambda")
plot(fit, xvar = "l1_norm")
plot(fit, xvar = "log_lambda", label = TRUE)
fit2 <- glmtlp(X, y, family = "gaussian", penalty = "l0")
plot(fit2, xvar = "kappa", label = TRUE)
```
predict.cv.glmtlp  

Predict Method for a "cv.glmtlp" Object.

Description

Makes predictions for a cross-validated glmtlp model, using the stored "glmtlp" object, and the optimal value chosen for lambda.

Usage

## S3 method for class 'cv.glmtlp'
predict(
  object,
  X,
  type = c("link", "response", "class", "coefficients", "numnzs", "varnzs"),
  lambda = NULL,
  kappa = NULL,
  which = object$idx.min,
  ...
)

## S3 method for class 'cv.glmtlp'
coef(object, lambda = NULL, kappa = NULL, which = object$idx.min, ...)

Arguments

- **object**: Fitted "cv.glmtlp" object.
- **X**: X Matrix of new values for X at which predictions are to be made. Must be a matrix.
- **type**: Type of prediction to be made. For "gaussian" models, type "link" and "response" are equivalent and both give the fitted values. For "binomial" models, type "link" gives the linear predictors and type "response" gives the fitted probabilities. Type "coefficients" computes the coefficients at the provided values of lambda or kappa. Note that for "binomial" models, results are returned only for the class corresponding to the second level of the factor response. Type "class" applies only to "binomial" models, and gives the class label corresponding to the maximum probability. Type "numnzs" gives the total number of non-zero coefficients for each value of lambda or kappa. Type "varnzs" gives a list of indices of the nonzero coefficients for each value of lambda or kappa.
- **lambda**: Value of the penalty parameter lambda at which predictions are to be made. Default is NULL.
- **kappa**: Value of the penalty parameter kappa at which predictions are to be made. Default is NULL.
- **which**: Index of the penalty parameter lambda or kappa sequence at which predictions are to be made. Default is the idx.min stored in the cv.glmtlp object.
- **...**: Additional arguments.
Value

The object returned depends on type.

Author(s)

Chunlin Li, Yu Yang, Chong Wu
Maintainer: Yu Yang <yang6367@umn.edu>

References

Two R package Github: ncvreg and glmnet.

See Also

print, predict, coef and plot methods, and the cv.glmtlp function.

Examples

```r
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
cv.fit <- cv.glmtlp(X, y, family = "gaussian", penalty = "l1")
predict(cv.fit, X = X[1:5, ])
coef(cv.fit)
predict(cv.fit, X = X[1:5, ], lambda = 0.1)
```

predict.glmtlp

Predict Method for a "glmtlp" Object

Description

Predicts fitted values, logits, coefficients and more from a fitted glmtlp object.

Usage

```r
## S3 method for class 'glmtlp'
predict(
  object,
  X,
  type = c("link", "response", "class", "coefficients", "numnz", "varnz"),
)```
lambda = NULL,
kappa = NULL,
which = 1:(ifelse(object$penalty == "l0", length(object$kappa),
    length(object$lambda))),
...
)

## S3 method for class 'glmtlp'
coef(
    object,
    lambda = NULL,
    kappa = NULL,
    which = 1:(ifelse(object$penalty == "l0", length(object$kappa),
        length(object$lambda))),
    drop = TRUE,
    ...
)

Arguments

- **object**: Fitted glmtlp model object.
- **X**: Matrix of new values for X at which predictions are to be made. Must be a matrix. This argument will not be used for type=c("coefficients","numnz", "varnz").
- **type**: Type of prediction to be made. For "gaussian" models, type "link" and "response" are equivalent and both give the fitted values. For "binomial" models, type "link" gives the linear predictors and type "response" gives the fitted probabilities. Type "coefficients" computes the coefficients at the provided values of lambda or kappa. Note that for "binomial" models, results are returned only for the class corresponding to the second level of the factor response. Type "class" applies only to "binomial" models, and gives the class label corresponding to the maximum probability. Type "numnz" gives the total number of non-zero coefficients for each value of lambda or kappa. Type "varnz" gives a list of indices of the nonzero coefficients for each value of lambda or kappa.
- **lambda**: Value of the penalty parameter lambda at which predictions are to be made. Default is NULL.
- **kappa**: Value of the penalty parameter kappa at which predictions are to be made. Default is NULL.
- **which**: Index of the penalty parameter lambda or kappa sequence at which predictions are to be made. Default are the indices for the entire penalty parameter sequence.
- **drop**: Whether or not keep the dimension that is of length 1.

Details

c coef(...) is equivalent to predict(type="coefficients",...)

predict.glmtlp

Value

The object returned depends on type.

Author(s)

Chunlin Li, Yu Yang, Chong Wu
Maintainer: Yu Yang <yang6367@umn.edu>

References

Two R package Github: ncvreg and glmnet.

See Also

print, predict, coef and plot methods, and the cv.glmtlp function.

Examples

# Gaussian
X <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
fit <- glmtlp(X, y, family = "gaussian", penalty = "l1")
predict(fit, X = X[1:5, ])
coef(fit)
predict(fit, X = X[1:5, ], lambda = 0.1)

# Binomial
X <- matrix(rnorm(100 * 20), 100, 20)
y <- sample(c(0,1), 100, replace = TRUE)
fit <- glmtlp(X, y, family = "binomial", penalty = "l1")
coef(fit)
predict(fit, X = X[1:5, ], type = "response")
predict(fit, X = X[1:5, ], type = "response", lambda = 0.01)
predict(fit, X = X[1:5, ], type = "class", lambda = 0.01)
predict(fit, X = X[1:5, ], type = "numnz", lambda = 0.01)
**setup_lambda**

*Generate lambda sequence.*

---

**Description**

Generate lambda sequence.

**Usage**

```r
setup_lambda(X, y, weights, lambda.min.ratio, nlambda)
```

**Arguments**

- `X`  
  Input matrix, of dimension `nobs x nvars`; each row is an observation vector.

- `y`  
  Response variable, of length `nobs`. For `family="gaussian"`, it should be quantitative; for `family="binomial"`, it should be either a factor with two levels or a binary vector.

- `weights`  
  Observation weights.

- `lambda.min.ratio`  
  The smallest value for lambda, as a fraction of `lambda.max`, the smallest value for which all coefficients are zero. The default depends on the sample size `nobs` relative to the number of variables `nvars`.

- `nlambda`  
  The number of lambda values.
Index

* datasets
  bin_data, 2
  gau_data, 5

* models
  cv.glmtlp, 3
  glmtlp, 7
  plot.cv.glmtlp, 11
  plot.glmtlp, 12
  predict.cv.glmtlp, 14
  predict.glmtlp, 15

* plot
  plot.cv.glmtlp, 11
  plot.glmtlp, 12

* regression
  cv.glmtlp, 3
  glmtlp, 7
  predict.cv.glmtlp, 14
  predict.glmtlp, 15

bin_data, 2

go.glmtlp (predict.cv.glmtlp), 14
coef.glmtlp (predict.glmtlp), 15

v.glmtlp, 3

go_data, 5
gen.binomial.data, 5
gen.gaussian.data, 6

glmtlp, 7

plot.cv.glmtlp, 11
plot.glmtlp, 12
predict.cv.glmtlp, 14
predict.glmtlp, 15

setup_lambda, 18