# Package: gelnet (via r-universe)

September 3, 2024

**Version** 1.3.0

**Date** 2018-12-01

**License** GPL (>= 3)

**Title** Generalized Elastic Nets

**Description** Implements several extensions of the elastic net regularization scheme. These extensions include individual feature penalties for the L1 term, feature-feature penalties for the L2 term, as well as translation coefficients for the latter.

**Author** Artem Sokolov

**Maintainer** Artem Sokolov <artem.sokolov@gmail.com>

**Depends** R (>= 3.1.0)

**Suggests** knitr, rmarkdown, tidyverse, rlang, testthat

**Encoding** UTF-8

**VignetteBuilder** knitr

**RoxygenNote** 6.1.1

**LinkingTo** Rcpp, RcppArmadillo

**Imports** Rcpp

**Repository** https://artemsokolov.r-universe.dev

**RemoteUrl** https://github.com/artemsokolov/gelnet

**RemoteRef** HEAD

**RemoteSha** df9b372e19c782665cc84635f67ec37018e0352e

## Contents

---

`+.geldef` *Composition operator for GELnet model definition*

---

## Description

Composition operator for GELnet model definition

## Usage

```
## S3 method for class 'geldef'
lhs + rhs
```

## Arguments

| | |
|---|---|
| `lhs` | left-hand side of composition (current chain) |
| `rhs` | right-hand side of composition (new module) |

`adj2lapl` *Generate a graph Laplacian*

## Description

Generates a graph Laplacian from the graph adjacency matrix.

## Usage

```
adj2lapl(A)
```

## Arguments

| | |
|---|---|
| `A` | n-by-n adjacency matrix for a graph with n nodes |

## Details

A graph Laplacian is defined as: $l_{i,j} = deg(v_i)$, if $i = j$; $l_{i,j} = -1$, if $i \neq j$ and $v_i$ is adjacent to $v_j$; and $l_{i,j} = 0$, otherwise

## Value

The n-by-n Laplacian matrix of the graph

## See Also

`adj2nlapl`

`adj2nlapl` *Generate a normalized graph Laplacian*

## Description

Generates a normalized graph Laplacian from the graph adjacency matrix.

## Usage

```
adj2nlapl(A)
```

## Arguments

| | |
|---|---|
| `A` | n-by-n adjacency matrix for a graph with n nodes |

## Details

A normalized graph Laplacian is defined as: $l_{i,j} = 1$, if $i = j$; $l_{i,j} = -1/\sqrt{deg(v_i)deg(v_j)}$, if $i \neq j$ and $v_i$ is adjacent to $v_j$; and $l_{i,j} = 0$, otherwise

### Value

The n-by-n Laplacian matrix of the graph

### See Also

adj2nlapl

---

`gelnet` *GELnet model definition*

---

### Description

Starting building block for defining a GELnet model

### Usage

```
gelnet(X)
```

### Arguments

| | |
|---|---|
| X | n-by-p matrix of n samples in p dimensions |

### Value

A GELnet model definition

---

`gelnet.ker` *Kernel models for linear regression, binary classification and one-class problems.*

---

### Description

Infers the problem type and learns the appropriate kernel model.

### Usage

```
gelnet.ker(K, y, lambda, a, max.iter = 100, eps = 1e-05,
  v.init = rep(0, nrow(K)), b.init = 0, fix.bias = FALSE,
  silent = FALSE, balanced = FALSE)
```

### Arguments

| | |
|---|---|
| K | n-by-n matrix of pairwise kernel values over a set of n samples |
| y | n-by-1 vector of response values. Must be numeric vector for regression, factor with 2 levels for binary classification, or NULL for a one-class task. |
| lambda | scalar, regularization parameter |
| a | n-by-1 vector of sample weights (regression only) |
| max.iter | maximum number of iterations (binary classification and one-class problems only) |
| eps | convergence precision (binary classification and one-class problems only) |
| v.init | initial parameter estimate for the kernel weights (binary classification and one-class problems only) |
| b.init | initial parameter estimate for the bias term (binary classification only) |
| fix.bias | set to TRUE to prevent the bias term from being updated (regression only) (default: FALSE) |
| silent | set to TRUE to suppress run-time output to stdout (default: FALSE) |
| balanced | boolean specifying whether the balanced model is being trained (binary classification only) (default: FALSE) |

### Details

The entries in the kernel matrix K can be interpreted as dot products in some feature space $\phi$. The corresponding weight vector can be retrieved via $w = \sum_i v_i \phi(x_i)$. However, new samples can be classified without explicit access to the underlying feature space:

$$w^T \phi(x) + b = \sum_i v_i \phi^T(x_i)\phi(x) + b = \sum_i v_i K(x_i, x) + b$$

The method determines the problem type from the labels argument y. If y is a numeric vector, then a ridge regression model is trained by optimizing the following objective function:

$$\frac{1}{2n} \sum_i a_i (z_i - (w^T x_i + b))^2 + w^T w$$

If y is a factor with two levels, then the function returns a binary classification model, obtained by optimizing the following objective function:

$$-\frac{1}{n} \sum_i y_i s_i - \log(1 + \exp(s_i)) + w^T w$$

where

$$s_i = w^T x_i + b$$

Finally, if no labels are provided (y == NULL), then a one-class model is constructed using the following objective function:

$$-\frac{1}{n} \sum_i s_i - \log(1 + \exp(s_i)) + w^T w$$

where

$$s_i = w^T x_i$$

In all cases, $w = \sum_i v_i \phi(x_i)$ and the method solves for $v_i$.

## Value

A list with two elements:

**v** n-by-1 vector of kernel weights

**b** scalar, bias term for the linear model (omitted for one-class models)

## See Also

gelnet

---

`gelnet_blr_obj` *Binary logistic regression objective function*

---

## Description

Evaluates the logistic regression objective function value for a given model. See details.

## Usage

```
gelnet_blr_obj(w, b, X, y, l1, l2, balanced = FALSE, d = NULL,
  P = NULL, m = NULL)
```

## Arguments

| | |
|---|---|
| `w` | p-by-1 vector of model weights |
| `b` | the model bias term |
| `X` | n-by-p matrix of n samples in p dimensions |
| `y` | n-by-1 binary response vector sampled from 0,1 |
| `l1` | L1-norm penalty scaling factor $\lambda_1$ |
| `l2` | L2-norm penalty scaling factor $\lambda_2$ |
| `balanced` | boolean specifying whether the balanced model is being evaluated |
| `d` | p-by-1 vector of feature weights |
| `P` | p-by-p feature-feature penalty matrix |
| `m` | p-by-1 vector of translation coefficients |

### Details

Computes the objective function value according to

$$-\frac{1}{n}\sum_i y_i s_i - \log(1+\exp(s_i)) + R(w)$$

where

$$s_i = w^T x_i + b$$

$$R(w) = \lambda_1 \sum_j d_j |w_j| + \frac{\lambda_2}{2}(w-m)^T P(w-m)$$

When balanced is TRUE, the loss average over the entire data is replaced with averaging over each class separately. The total loss is then computes as the mean over those per-class estimates.

### Value

The objective function value.

### See Also

gelnet

---

`gelnet_blr_opt` *GELnet optimizer for binary logistic regression*

---

### Description

Constructs a GELnet model for logistic regression using the Newton method.

### Usage

```
gelnet_blr_opt(X, y, l1, l2, max_iter = 100L, eps = 1e-05,
  silent = FALSE, verbose = FALSE, balanced = FALSE,
  nonneg = FALSE, w_init = NULL, b_init = NULL, d = NULL,
  P = NULL, m = NULL)
```

### Arguments

| | |
|---|---|
| `X` | n-by-p matrix of n samples in p dimensions |
| `y` | n-by-1 vector of binary response labels (must be in 0,1) |
| `l1` | coefficient for the L1-norm penalty |
| `l2` | coefficient for the L2-norm penalty |
| `max_iter` | maximum number of iterations |
| `eps` | convergence precision |
| `silent` | set to TRUE to suppress run-time output to stdout (default: FALSE) |

| | |
|---|---|
| `balanced` | boolean specifying whether the balanced model is being trained |
| `nonneg` | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE ) |
| `w_init` | initial parameter estimate for the weights |
| `b_init` | initial parameter estimate for the bias term |
| `d` | p-by-1 vector of feature weights |
| `P` | p-by-p feature association penalty matrix |
| `m` | p-by-1 vector of translation coefficients |

## Details

The method operates by constructing iteratively re-weighted least squares approximations of the log-likelihood loss function and then calling the linear regression routine to solve those approximations. The least squares approximations are obtained via the Taylor series expansion about the current parameter estimates.

## Value

A list with two elements:

**w** p-by-1 vector of p model weights

**b** scalar, bias term for the linear model

## See Also

`gelnet.lin`

---

`gelnet_cv` *k-fold cross-validation for parameter tuning.*

---

## Description

Performs k-fold cross-validation to select the best pair of the L1- and L2-norm penalty values.

## Usage

```
gelnet_cv(X, y, nL1, nL2, nFolds = 5, a = rep(1, n), d = rep(1, p),
  P = diag(p), m = rep(0, p), max.iter = 100, eps = 1e-05,
  w.init = rep(0, p), b.init = 0, fix.bias = FALSE, silent = FALSE,
  balanced = FALSE)
```

### Arguments

| | |
|---|---|
| `X` | n-by-p matrix of n samples in p dimensions |
| `y` | n-by-1 vector of response values. Must be numeric vector for regression, factor with 2 levels for binary classification, or NULL for a one-class task. |
| `nL1` | number of values to consider for the L1-norm penalty |
| `nL2` | number of values to consider for the L2-norm penalty |
| `nFolds` | number of cross-validation folds (default:5) |
| `a` | n-by-1 vector of sample weights (regression only) |
| `d` | p-by-1 vector of feature weights |
| `P` | p-by-p feature association penalty matrix |
| `m` | p-by-1 vector of translation coefficients |
| `max.iter` | maximum number of iterations |
| `eps` | convergence precision |
| `w.init` | initial parameter estimate for the weights |
| `b.init` | initial parameter estimate for the bias term |
| `fix.bias` | set to TRUE to prevent the bias term from being updated (regression only) (default: FALSE) |
| `silent` | set to TRUE to suppress run-time output to stdout (default: FALSE) |
| `balanced` | boolean specifying whether the balanced model is being trained (binary classification only) (default: FALSE) |

### Details

Cross-validation is performed on a grid of parameter values. The user specifies the number of values to consider for both the L1- and the L2-norm penalties. The L1 grid values are equally spaced on [0, L1s], where L1s is the smallest meaningful value of the L1-norm penalty (i.e., where all the model weights are just barely zero). The L2 grid values are on a logarithmic scale centered on 1.

### Value

A list with the following elements:

**l1** the best value of the L1-norm penalty

**l2** the best value of the L2-norm penalty

**w** p-by-1 vector of p model weights associated with the best (l1,l2) pair.

**b** scalar, bias term for the linear model associated with the best (l1,l2) pair. (omitted for one-class models)

**perf** performance value associated with the best model. (Likelihood of data for one-class, AUC for binary classification, and -RMSE for regression)

### See Also

`gelnet`

## gelnet_lin_obj *Linear regression objective function*

### Description

Evaluates the linear regression objective function value for a given model. See details.

### Usage

```
gelnet_lin_obj(w, b, X, z, l1, l2, a = NULL, d = NULL, P = NULL,
  m = NULL)
```

### Arguments

| | |
|---|---|
| w | p-by-1 vector of model weights |
| b | the model bias term |
| X | n-by-p matrix of n samples in p dimensions |
| z | n-by-1 response vector |
| l1 | L1-norm penalty scaling factor $\lambda_1$ |
| l2 | L2-norm penalty scaling factor $\lambda_2$ |
| a | n-by-1 vector of sample weights |
| d | p-by-1 vector of feature weights |
| P | p-by-p feature-feature penalty matrix |
| m | p-by-1 vector of translation coefficients |

### Details

Computes the objective function value according to

$$\frac{1}{2n} \sum_i a_i (z_i - (w^T x_i + b))^2 + R(w)$$

where

$$R(w) = \lambda_1 \sum_j d_j |w_j| + \frac{\lambda_2}{2} (w-m)^T P (w-m)$$

### Value

The objective function value.

gelnet_lin_opt *GELnet optimizer for linear regression*

## Description

Constructs a GELnet model for linear regression using coordinate descent.

## Usage

```
gelnet_lin_opt(X, z, l1, l2, max_iter = 100L, eps = 1e-05,
  fix_bias = FALSE, silent = FALSE, verbose = FALSE,
  nonneg = FALSE, w_init = NULL, b_init = NULL, a = NULL,
  d = NULL, P = NULL, m = NULL)
```

## Arguments

| | |
|---|---|
| X | n-by-p matrix of n samples in p dimensions |
| z | n-by-1 vector of response values |
| l1 | coefficient for the L1-norm penalty |
| l2 | coefficient for the L2-norm penalty |
| max_iter | maximum number of iterations |
| eps | convergence precision |
| fix_bias | set to TRUE to prevent the bias term from being updated (default: FALSE) |
| silent | set to TRUE to suppress run-time output; overwrites verbose (default: FALSE) |
| verbose | set to TRUE to see extra output; is overwritten by silent (default: FALSE) |
| nonneg | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE ) |
| w_init | initial parameter estimate for the weights |
| b_init | initial parameter estimate for the bias term |
| a | n-by-1 vector of sample weights |
| d | p-by-1 vector of feature weights |
| P | p-by-p feature association penalty matrix |
| m | p-by-1 vector of translation coefficients |

## Details

The method operates through cyclical coordinate descent. The optimization is terminated after the desired tolerance is achieved, or after a maximum number of iterations.

## Value

A list with two elements:

**w** p-by-1 vector of p model weights

**b** scalar, bias term for the linear model `gelnet_oclr_obj` *One-class logistic regression objective function*

## Description

Evaluates the one-class objective function value for a given model See details.

## Usage

```
gelnet_oclr_obj(w, X, l1, l2, d = NULL, P = NULL, m = NULL)
```

## Arguments

| | |
|---|---|
| `w` | p-by-1 vector of model weights |
| `X` | n-by-p matrix of n samples in p dimensions |
| `l1` | L1-norm penalty scaling factor $\lambda_1$ |
| `l2` | L2-norm penalty scaling factor $\lambda_2$ |
| `d` | p-by-1 vector of feature weights |
| `P` | p-by-p feature-feature penalty matrix |
| `m` | p-by-1 vector of translation coefficients |

## Details

Computes the objective function value according to

$$-\frac{1}{n}\sum_i s_i - \log(1+\exp(s_i)) + R(w)$$

where

$$s_i = w^T x_i$$

$$R(w) = \lambda_1 \sum_j d_j |w_j| + \frac{\lambda_2}{2}(w-m)^T P(w-m)$$

## Value

The objective function value.

## See Also

gelnet

gelnet_oclr_opt *GELnet optimizer for one-class logistic regression*

## Description

Constructs a GELnet model for one-class regression using the Newton method.

## Usage

```
gelnet_oclr_opt(X, l1, l2, max_iter = 100L, eps = 1e-05,
  silent = FALSE, verbose = FALSE, nonneg = FALSE, w_init = NULL,
  d = NULL, P = NULL, m = NULL)
```

## Arguments

| | |
|---|---|
| `X` | n-by-p matrix of n samples in p dimensions |
| `l1` | coefficient for the L1-norm penalty |
| `l2` | coefficient for the L2-norm penalty |
| `max_iter` | maximum number of iterations |
| `eps` | convergence precision |
| `silent` | set to TRUE to suppress run-time output to stdout (default: FALSE) |
| `nonneg` | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE ) |
| `w_init` | initial parameter estimate for the weights |
| `d` | p-by-1 vector of feature weights |
| `P` | p-by-p feature association penalty matrix |
| `m` | p-by-1 vector of translation coefficients |

## Details

The function optimizes the following objective:

$$-\frac{1}{n}\sum_i s_i - \log(1+\exp(s_i)) + R(w)$$

where

$$s_i = w^T x_i$$

$$R(w) = \lambda_1 \sum_j d_j |w_j| + \frac{\lambda_2}{2}(w-m)^T P(w-m)$$

The method operates by constructing iteratively re-weighted least squares approximations of the log-likelihood loss function and then calling the linear regression routine to solve those approximations. The least squares approximations are obtained via the Taylor series expansion about the current parameter estimates.

## Value

A list with one element:

**w** p-by-1 vector of p model weights

---

`gelnet_train` *Trains a GELnet model*

---

## Description

Trains a model on the definition constructed by gelnet()

## Usage

```
gelnet_train(modeldef, max_iter = 100L, eps = 1e-05, silent = FALSE,
  verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `modeldef` | model definition constructed through gelnet() arithmetic |
| `max_iter` | maximum number of iterations |
| `eps` | convergence precision |
| `silent` | set to TRUE to suppress run-time output to stdout; overrides verbose (default: FALSE) |
| `verbose` | set to TRUE to see extra output; is overridden by silent (default: FALSE) |

## Details

The training is performed through cyclical coordinate descent, and the optimization is terminated after the desired tolerance is achieved or after a maximum number of iterations.

## Value

A GELNET model, expressed as a list with two elements:

**w** p-by-1 vector of p model weights

**b** scalar, bias term for the linear model (omitted for one-class models)

## gel_init — *Initializer for GELnet models*

### Description

Defines initial values for model weights and the bias term

### Usage

```
gel_init(w_init = NULL, b_init = NULL)
```

### Arguments

| | |
|---|---|
| `w_init` | p-by-1 vector of initial weight values |
| `b_init` | scalar, initial value for the bias term |

### Details

If an initializer is NULL, the values are computed automatically during training

### Value

An initializer that can be combined with a model definition using + operator

## L1_ceiling — *The largest meaningful value of the L1 parameter*

### Description

Computes the smallest value of the LASSO coefficient L1 that leads to an all-zero weight vector for a given gelnet model

### Usage

```
L1_ceiling(modeldef)
```

### Arguments

| | |
|---|---|
| `modeldef` | model definition constructed through gelnet() arithmetic |

### Details

The cyclic coordinate descent updates the model weight $w_k$ using a soft threshold operator $S(\cdot, \lambda_1 d_k)$ that clips the value of the weight to zero, whenever the absolute value of the first argument falls below $\lambda_1 d_k$. From here, it is straightforward to compute the smallest value of $\lambda_1$, such that all weights are clipped to zero.

## Value

The largest meaningful value of the L1 parameter (i.e., the smallest value that yields a model with all zero weights)

---

`model_blr` *Binary logistic regression*

---

## Description

Defines a binary logistic regression task

## Usage

```
model_blr(y, nonneg = FALSE, balanced = FALSE)
```

## Arguments

| | |
|---|---|
| `y` | n-by-1 factor with two levels |
| `nonneg` | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE) |
| `balanced` | boolean specifying whether the balanced model is being trained (default: FALSE) |

## Details

The binary logistic regression objective function is defined as

$$-\frac{1}{n} \sum_i y_i s_i - \log(1 + \exp(s_i)) + R(w)$$

where

$$s_i = w^T x_i + b$$

## Value

A GELnet task definition that can be combined with gelnet() output

## model_lin *Linear regression*

### Description

Defines a linear regression task

### Usage

```
model_lin(y, a = NULL, nonneg = FALSE, fix_bias = FALSE)
```

### Arguments

| | |
|---|---|
| `y` | n-by-1 numeric vector of response values |
| `a` | n-by-1 vector of sample weights |
| `nonneg` | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE) |
| `fix_bias` | set to TRUE to prevent the bias term from being updated (default: FALSE) |

### Details

The objective function is given by

$$\frac{1}{2n} \sum_i a_i (y_i - (w^T x_i + b))^2 + R(w)$$

### Value

A GELnet task definition that can be combined with gelnet() output

## model_oclr *One-class logistic regression*

### Description

Defines a one-class logistic regression (OCLR) task

### Usage

```
model_oclr(nonneg = FALSE)
```

### Arguments

| | |
|---|---|
| `nonneg` | set to TRUE to enforce non-negativity constraints on the weights (default: FALSE) |

## Details

The OCLR objective function is defined as

$$-\frac{1}{n}\sum_i s_i - \log(1 + \exp(s_i)) + R(w)$$

where

$$s_i = w^T x_i$$

## Value

A GELnet task definition that can be combined with gelnet() output

---

`perturb.gelnet` *Perturbs a GELnet model*

---

## Description

Given a linear model, perturbs its i^th coefficient by delta

## Usage

```
perturb.gelnet(model, i, delta)
```

## Arguments

| | |
|---|---|
| `model` | The model to perturb |
| `i` | Index of the coefficient to modify, or 0 for the bias term |
| `delta` | The value to perturb by |

## Value

Modified GELnet model

---

`rglz_L1` *L1 regularizer*

---

## Description

Defines an L1 regularizer with optional per-feature weights

## Usage

```
rglz_L1(l1, d = NULL)
```

## Arguments

| | |
|---|---|
| `l1` | coefficient for the L1-norm penalty |
| `d` | p-by-1 vector of feature weights |

## Details

The L1 regularization term is defined by

$$R1(w) = \lambda_1 \sum_j d_j |w_j|$$

## Value

A regularizer definition that can be combined with a model definition using + operator

---

`rglz_L2` *L2 regularizer*

---

## Description

Defines an L2 regularizer with optional feature-feature penalties and translation coefficients

## Usage

```
rglz_L2(l2, P = NULL, m = NULL)
```

## Arguments

| | |
|---|---|
| `l2` | coefficient for the L2-norm penalty |
| `P` | p-by-p feature association penalty matrix |
| `m` | p-by-1 vector of translation coefficients |

### Details

The L2 regularizer term is define by

$$R2(w) = \frac{\lambda_2}{2}(w-m)^T P(w-m)$$

### Value

A regularizer definition that can be combined with a model definition using + operator

---

## rglz_nf *Alternative L1 regularizer*

---

### Description

Defines an L1 regularizer that results in the desired number of non-zero feature weights

### Usage

```
rglz_nf(nFeats, d = NULL)
```

### Arguments

| | |
|---|---|
| `nFeats` | desired number of features with non-zero weights in the model |
| `d` | p-by-1 vector of feature weights |

### Details

The corresponding regularization coefficient is determined through binary search

### Value

A regularizer definition that can be combined with a model definition using + operator

# Index