Package ‘PenalReg’

August 24, 2026

Type Package

Title Automated Penalized Regression Analysis Using Ridge, Lasso and Elastic Net

Version 0.1.0

Description Provides an automated framework for penalized regression analysis using Ridge Regression, Lasso Regression and Elastic Net Regression. The package performs data standardization, training-testing data partitioning, cross-validation for hyperparameter tuning, model fitting, coefficient estimation, variable importance assessment, prediction, and performance evaluation. It simplifies regularized regression analysis by integrating the complete modeling workflow into a single function suitable for researchers for better understanding of the data.The methods are based on Hoerl and Kennard (1970) <doi:10.1080/00401706.1970.10488634>, Zou and Hastie (2005) <doi:10.1111/j.1467-9868.2005.00503.x>, and Friedman et al. (2010) <doi:10.18637/jss.v033.i01>.

License GPL-3

Encoding UTF-8

Depends R (>= 4.0.0)

Imports caret, stats, utils

Suggests glmnet

NeedsCompilation no

Config/roxygen2/version 8.1.0

Author S. Vishnu Shankar [aut, cre],
V. Lavanya [aut],
Santosha Rathod [aut],
Mrinmoy Ray [aut],
Anil Kumar [aut]

Maintainer S. Vishnu Shankar <S.vishnushankar55@gmail.com>

Repository CRAN

Date/Publication 2026-08-24 14:40:07 UTC

Contents

PenalReg *Automated Penalized Regression Analysis*

Description

Provides an automated framework for penalized regression analysis using Ridge Regression, Lasso Regression and Elastic Net Regression. The package performs data standardization, training-testing data partitioning, cross-validation for hyperparameter tuning, model fitting, coefficient estimation, variable importance assessment, prediction, and performance evaluation. It simplifies regularized regression analysis by integrating the complete modeling workflow into a single function suitable for researchers for better understanding of the data.

Usage

```
PenalReg(
  data,
  response,
  train_ratio = 0.8,
  standardize = TRUE,
  cv = 10,
  lambda = seq(0.01, 5, length.out = 10),
  alpha = seq(0, 1, 0.01),
  verbose = TRUE
)
```

Arguments

`data`	A data frame containing the response variable and predictor variables.
`response`	A character string specifying the response variable name or a numeric value specifying the column index of the response variable.
`train_ratio`	Numeric value between 0 and 1 specifying the proportion of observations allocated to the training set. The remaining observations are used for testing. Default is `0.80`.
`standardize`	Logical value indicating whether predictor variables should be standardized before model fitting. Possible values are `TRUE` and `FALSE`. Default is `TRUE`.
`cv`	Integer specifying the number of folds used for k-fold cross-validation during model tuning. Default is `10`.
`lambda`	Numeric vector of candidate regularization parameter (λ) values. The default is `seq(0.01, 5, length.out = 10)`.

alpha	Numeric vector specifying candidate Elastic Net mixing parameter (α) values. Values range from 0 to 1, where 0 corresponds to Ridge Regression and 1 corresponds to Lasso Regression. In the current implementation, this argument is retained for compatibility but the Elastic Net model uses an internal sequence of alpha values from 0.05 to 0.95 in increments of 0.05.
verbose	Logical value indicating whether progress messages and model results are displayed during execution. Default is TRUE.

Details

The function automatically fits three penalized regression models:

- Ridge Regression
- Lasso Regression
- Elastic Net Regression

Predictor variables that are not numeric are converted into a numeric design matrix using `model.matrix`. Predictor variables can optionally be standardized using the training-set mean and standard deviation. The same training-set scaling parameters are applied to the testing data.

The data are partitioned into training and testing subsets according to `train_ratio`. Hyperparameters are optimized using k-fold cross-validation implemented through the **caret** package and the **glmnet** modeling method.

Model performance is evaluated using:

- Coefficient of Determination (R-squared)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Mean Absolute Percentage Error (MAPE)
- Root Mean Squared Percentage Error (RMSPE)

The model with the highest testing-set R-squared is identified as the best-performing model and displayed when `verbose = TRUE`.

Value

A list of class `"PenalReg"` containing:

importance Variable importance measures for Ridge, Lasso, and Elastic Net models.

coefficients Estimated regression coefficients for each penalized regression model.

Fitted_Values A data frame containing observed and fitted values for the training data from Ridge, Lasso, and Elastic Net models.

Predicted_Values A data frame containing observed and predicted values for the testing data from Ridge, Lasso, and Elastic Net models.

Selected_Parameters A data frame containing the selected lambda and alpha values for each fitted model.

Training_Performance A data frame containing R-squared, MSE, RMSE, MAE, MAPE, and RMSPE values calculated on the training data.

Testing_Performance A data frame containing R-squared, MSE, RMSE, MAE, MAPE, and RMSPE values calculated on the testing data.

References

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55–67. doi:10.1080/00401706.1970.10488634

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2), 301–320. doi:10.1111/j.14679868.2005.00503.x

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1), 1–22. doi:10.18637/jss.v033.i01

See Also

train, glmnet

Examples

```
data(mtcars)

mtcars_subset <- data.frame(
  mpg = mtcars$mpg,
  cyl = mtcars$cyl,
  disp = mtcars$disp,
  hp = mtcars$hp,
  drat = mtcars$drat,
  wt = mtcars$wt,
  qsec = mtcars$qsec,
  gear = mtcars$gear,
  carb = mtcars$carb
)

fit <- PenalReg(
  data = mtcars_subset,
  response = "mpg",
  cv = 5,
  verbose = TRUE
)

fit
```

Index