Package ‘StepReg’

September 30, 2025

Version 1.6.0

Title Stepwise Regression Analysis

Date 2025-09-30

Description Stepwise regression is a statistical technique used for model selection. This package streamlines stepwise regression analysis by supporting multiple regression types(linear, Cox, logistic, Poisson, Gamma, and negative binomial), incorporating popular selection strategies(forward, backward, bidirectional, and subset), and offering essential metrics. It enables users to apply multiple selection strategies and metrics in a single function call, visualize variable selection processes, and export results in various formats. StepReg offers a data-splitting option to address potential issues with invalid statistical inference and a randomized forward selection option to avoid overfitting. We validated StepReg's accuracy using public datasets within the SAS software environment. Additionally, StepReg features an interactive Shiny application to enhance usability and accessibility.

License MIT + file LICENSE

BugReports `https://github.com/JunhuiLi1017/StepReg/issues`

VignetteBuilder knitr

Suggests knitr, testthat, BiocStyle, kableExtra

Imports dplyr, ggplot2, ggrepel, MASS, stringr, survival, flextable, cowplot, shiny, ggcorrplot, tidyr, summarytools, shinythemes, rmarkdown, DT, shinycssloaders, shinyjs, pROC, survAUC

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Author Junhui Li [cre] (ORCID: <`https://orcid.org/0000-0003-3973-1700`>),
Junhui Li [aut],
Kai Hu [aut],
Xiaohuan Lu [aut],
Sushmita N Nayak [ctb],
Cesar Bautista Sotelo [ctb],
Michael A Lodato [ctb],
Wenxin Liu [aut],
Lihua Julie Zhu [aut]

Maintainer Junhui Li <junhui.li11@umassmed.edu>

Date/Publication 2025-09-30 16:10:02 UTC

Contents

`affairs` *Fair's Extramarital Affairs Data*

Description

Infidelity data, known as Fair's Affairs. Cross-section data from a survey conducted by Psychology Today in 1969. This dataset is sourced from the AER package.

Usage

```
data(affairs)
```

Format

A data frame containing 601 observations on 9 variables.

affairs numeric. How often engaged in extramarital sexual intercourse during the past year? 0 = none, 1 = once, 2 = twice, 3 = 3 times, 7 = 4–10 times, 12 = monthly, 12 = weekly, 12 = daily.

gender factor indicating gender.

age numeric variable coding age in years: 17.5 = under 20, 22 = 20–24, 27 = 25–29, 32 = 30–34, 37 = 35–39, 42 = 40–44, 47 = 45–49, 52 = 50–54, 57 = 55 or over.

yearsmarried numeric variable coding number of years married: 0.125 = 3 months or less, 0.417 = 4–6 months, 0.75 = 6 months–1 year, 1.5 = 1–2 years, 4 = 3–5 years, 7 = 6–8 years, 10 = 9–11 years, 15 = 12 or more years.

children factor. Are there children in the marriage?

religiousness numeric variable coding religiousness: 1 = anti, 2 = not at all, 3 = slightly, 4 = somewhat, 5 = very.

education numeric variable coding level of education: 9 = grade school, 12 = high school graduate, 14 = some college, 16 = college graduate, 17 = some graduate work, 18 = master's degree, 20 = Ph.D., M.D., or other advanced degree.

occupation numeric variable coding occupation according to Hollingshead classification (reverse numbering).

rating numeric variable coding self rating of marriage: 1 = very unhappy, 2 = somewhat unhappy, 3 = average, 4 = happier than average, 5 = very happy.

Source

Fair, R.C. (1978). A Theory of Extramarital Affairs. Journal of Political Economy, 86, 45–61.

References

- Greene, W.H. (2003). Econometric Analysis, 5th edition. Upper Saddle River, NJ: Prentice Hall.
- Fair, R.C. (1978). A Theory of Extramarital Affairs. Journal of Political Economy, 86, 45–61.

See Also

`Affairs` for the original dataset in the AER package

Examples

```
data(affairs)
summary(affairs)
```

`creditCard` *Credit Card Application Dataset*

Description

A dataset containing credit history information for credit card applicants. This dataset is sourced from the AER package.

Usage

```
data(creditCard)
```

Format

A data frame with 1,319 observations and 12 variables:

card Factor. Whether the credit card application was accepted (Yes/No)

reports Numeric. Number of major derogatory reports on the applicant's credit history

age Numeric. Age in years plus twelfths of a year (e.g., 30.5 represents 30 years and 6 months)

income Numeric. Annual income in USD (divided by 10,000)

share Numeric. Ratio of monthly credit card expenditure to yearly income

expenditure Numeric. Average monthly credit card expenditure in USD

owner Factor. Home ownership status (Yes/No)

selfemp Factor. Self-employment status (Yes/No)

dependents Numeric. Number of dependents

months Numeric. Number of months living at current address

majorcards Numeric. Number of major credit cards held

active Numeric. Number of active credit accounts

Details

This dataset is commonly used for credit risk analysis and modeling credit card approval decisions. It provides a comprehensive view of various factors that may influence credit card application outcomes.

Source

Greene, W.H. (2003). Econometric Analysis, 5th edition. Upper Saddle River, NJ: Prentice Hall.

See Also

`CreditCard` for the original dataset in the AER package

Examples

```
data(creditCard)
summary(creditCard)
```

`lung` *NCCTG Lung Cancer Data*

Description

Survival in patients with advanced lung cancer from the North Central Cancer Treatment Group. Performance scores rate how well the patient can perform usual daily activities.

Usage

```
data(lung)
```

Format

A data frame containing 228 observations on 10 variables.

inst Institution code

time Survival time in days

status censoring status 1=censored, 2=dead

age Age in years

sex Male=1 Female=2

ph.ecog ECOG performance score as rated by the physician. 0=asymptomatic, 1= symptomatic but completely ambulatory, 2= in bed < 50% of the day, 3= in bed > 50% of the day but not bedbound, 4 = bedbound

ph.karno Karnofsky performance score (bad=0-good=100) rated by physician

pat.karno Karnofsky performance score as rated by patient

meal.cal Calories consumed at meals

wt.loss Weight loss in last six months (pounds)

Note

This dataset is sourced from the survival package.

Source

Terry Therneau. (2021). survival: Survival Analysis. R package version 3.4-2. `https://CRAN.R-project.org/package=survival`

References

- Loprinzi CL. Laurie JA. Wieand HS. Krook JE. Novotny PJ. Kugler JW. Bartel J. Law M. Bateman M. Klatt NE. et al. Prospective evaluation of prognostic variables from patient-completed questionnaires. North Central Cancer Treatment Group. Journal of Clinical Oncology. 12(3):601-7, 1994.

See Also

`lung` for the original dataset in the survival package

Examples

```
data(lung)
summary(lung)
```

`performance`	*Model Performance Summary Across Different Selection Strategies*

Description

Creates a summary table showing the performance of the selected models by different combinations of stepwise regression strategies and selection metrics.

Usage

```
performance(x, ...)
```

Arguments

`x`	A list object returned by the `stepwise()` function
`...`	Additional arguments (currently not used)

Value

A data frame where:

`model`	The formula of each selected model
`strategy:metric`	Columns for each combination of strategy and metric used

- **For linear, poisson, gamma, and negative binomial regression:**
  - `adj_r2_train/adj_r2_test`: Adjusted R-squared measures the proportion of variance explained by the model, adjusted for the number of predictors. Values range from 0 to 1, with higher values indicating better model fit. A good model should have high adjusted R-squared on both training and test data, with minimal difference between them. Large differences suggest overfitting.
  - `mse_train/mse_test`: Mean Squared Error measures the average squared difference between predicted and actual values. Lower values indicate better model performance. The test MSE should be close to training MSE; significantly higher test MSE suggests overfitting.
  - `mae_train/mae_test`: Mean Absolute Error measures the average absolute difference between predicted and actual values. Lower values indicate better model performance. Like MSE, test MAE should be close to training MAE to avoid overfitting.
- **For logistic regression:**
  - `accuracy_train/accuracy_test`: Accuracy measures the proportion of correct predictions (true positives + true negatives) / total predictions. Values range from 0 to 1, with higher values indicating better classification performance. Test accuracy should be close to training accuracy; large differences suggest overfitting.
  - `auc_train/auc_test`: Area Under the Curve measures the model's ability to distinguish between classes. Values range from 0.5 (random) to 1.0 (perfect discrimination). AUC > 0.7 is considered acceptable, > 0.8 is good, > 0.9 is excellent. Test AUC should be close to training AUC to avoid overfitting.

- `log_loss_train/log_loss_test`: Log Loss (logarithmic loss) penalizes confident wrong predictions more heavily. Lower values indicate better model performance. Values close to 0 are ideal. Test log loss should be close to training log loss; higher test log loss suggests overfitting.

- **For Cox regression:**
  - `c-index_train/c-index_test`: Concordance Index (C-index) measures the model's ability to correctly rank survival times. Values range from 0.5 (random) to 1.0 (perfect ranking). C-index > 0.7 is considered acceptable, > 0.8 is good, > 0.9 is excellent. Test C-index should be close to training C-index to avoid overfitting.
  - `auc_hc`: Harrell's C-index for time-dependent AUC, measuring discrimination at specific time points. Higher values indicate better discrimination ability.
  - `auc_uno`: Uno's C-index for time-dependent AUC, providing an alternative measure of discrimination that may be more robust to censoring patterns.
  - `auc_sh`: Schemper and Henderson's C-index for time-dependent AUC, offering another perspective on model discrimination performance.

Each cell contains the performance of the model by the corresponding strategy-metric combination. For the subset strategy with Information Criteria (IC), only the single best model across all variable numbers is shown. This does not apply to Significance Level (SL) since F/Rao statistics can only be compared between models with the same number of variables.

Examples

```
# Load example data
data(mtcars)

# Run stepwise regression with multiple strategies and metrics
formula <- mpg ~ .
result <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = c("forward", "backward", "bidirection"),
  metric = c("AIC", "BIC")
)

# Get performance summary
performance(result)
```

`plot.StepReg` *Visualize Stepwise Regression Results*

Description

Creates informative visualizations of the variable selection process from a StepReg object. This function generates two types of plots: detailed step-by-step selection process and an overview of the final selected variables.

Usage

```
## S3 method for class 'StepReg'
plot(
  x,
  strategy = attr(x, "nonhidden"),
  process = c("overview", "detail"),
  num_digits = 6,
  ...
)
```

Arguments

`x`	A StepReg object containing the results of stepwise regression analysis.
`strategy`	Character. Specifies which selection strategy to visualize: • "forward" - Forward selection • "backward" - Backward elimination • "bidirection" - Bidirectional selection • "subset" - Best subset selection Default is the first strategy name in the StepReg object.
`process`	Character. Specifies the type of visualization to display: • "detail" - Shows detailed step-by-step selection process with variable entry/removal • "overview" - Shows summary of the selection process with metric values Default is "overview".
`num_digits`	Integer. Number of decimal places to display in the plots. Default is 6.
`...`	Additional argument passed to plotting functions (currently not used).

Details

The function creates different types of visualizations based on the selection strategy:

- For forward/backward/bidirectional selection:
  - detail view shows a heatmap with green tiles for added variables, tan tiles for removed variables, and gray tiles for non-selected variables
  - Overview shows metric values across steps with variable labels
- For subset selection:
  - detail view shows a heatmap of selected variables at each step
  - Overview shows metric values for different subset sizes

Value

A ggplot object showing either:

- For "detail" process: A heatmap showing variable selection status at each step
- For "overview" process: A line plot showing metric values across steps

See Also

stepwise for creating StepReg objects

Examples

```
## Not run:
# Load example data
data(mtcars)

# Run stepwise regression with multiple strategies
formula <- mpg ~ .
result <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = c("forward", "bidirection", "subset"),
  metric = c("AIC", "BIC", "SL")
)

# Generate default overview plot
plot(result)

# Generate detailed plot for forward selection
plot(result, strategy = "forward", process = "detail")

# Generate overview plot with 3 decimal places
plot(result, strategy = "bidirection", process = "overview", num_digits = 3)

## End(Not run)
```

---

print.StepReg *Print Stepwise Regression Results*

Description

Displays the final model fit statistics from a StepReg object. This function provides a concise summary of the selected model's performance metrics.

Usage

```
## S3 method for class 'StepReg'
print(x, ...)
```

Arguments

x	A StepReg object containing the results of stepwise regression analysis.
...	Additional arguments passed to the print method (currently not used).

Details

The print method provides a focused view of the final model's performance, showing the selected variables and their corresponding fit statistics. This is useful for quickly assessing the model's quality without the detailed step-by-step selection process (which can be viewed using `stepwise`).

Value

Invisibly returns the printed object. The function displays:

- Final model fit statistics for each strategy and metric combination

See Also

`stepwise` for creating StepReg objects

`plot.StepReg` for visualization of results

Examples

```
## Not run:
# Load example data
data(mtcars)

# Run stepwise regression
formula <- mpg ~ .
result <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = "forward",
  metric = "AIC"
)

# Print final model statistics
result

## End(Not run)
```

`remission` *Leukemia Remission Dataset*

Description

A dataset containing information about leukemia remission and associated risk factors. This dataset is commonly used for demonstrating logistic regression analysis in medical research.

Usage

```
data(remission)
```

Format

A data frame with 27 observations and 7 variables:

remiss Binary outcome variable indicating leukemia remission status:

- 1 = Remission occurred
- 0 = No remission

cell Numeric. Cellularity of the marrow clot section (percentage)

smear Numeric. Smear differential percentage of blasts

infil Numeric. Percentage of absolute marrow leukemia cell infiltrate

li Numeric. Percentage labeling index of the bone marrow leukemia cells

blast Numeric. Absolute number of blasts in the peripheral blood

temp Numeric. Highest temperature (in Fahrenheit) before treatment

Details

This dataset is particularly useful for:

- Demonstrating logistic regression analysis
- Studying risk factors for leukemia remission
- Teaching medical statistics and predictive modeling

Source

Lee, E. T. (1974). "A Computer Program for Linear Logistic Regression Analysis." Computer Programs in Biomedicine 4:80–92.

References

- Lee, E. T. (1974). "A Computer Program for Linear Logistic Regression Analysis." Computer Programs in Biomedicine 4:80–92.
- Penn State University Statistics Online. "Logistic Regression Example." https://online.stat.psu.edu/stat501/book/export/html/1011

Examples

```
## Not run:
# Load the dataset
data(remission)

# View first few rows
head(remission)

# Summary statistics
summary(remission)

# Run logistic regression
model <- glm(remiss ~ ., data = remission, family = binomial)
```

```
summary(model)

## End(Not run)
```

report Generate Stepwise Regression Report

Description

Creates formatted reports from StepReg objects in various document formats. This function generates comprehensive reports containing all tables and results from the stepwise regression analysis.

Usage

```
report(x, report_name, format = c("html", "docx", "rtf", "pptx"))
```

Arguments

`x`	A StepReg object containing the results of stepwise regression analysis.
`report_name`	Character. The name of the output report file(s) without extension.
`format`	Character vector. The output format(s) for the report. Choose from: • "html" - Web page format (default) • "docx" - Microsoft Word document • "pptx" - Microsoft PowerPoint presentation • "rtf" - Rich Text Format. Multiple formats can be specified simultaneously.

Details

The generated report includes:

- Summary of model parameters and selection criteria
- Variable types and classifications
- Step-by-step selection process
- Final selected model and fit statistics
- Model coefficients and significance levels

Value

Creates report file(s) in the specified format(s) in the current working directory. The file name will be `report_name.format` (e.g., "myreport.html", "myreport.docx").

See Also

stepwise for creating StepReg objects

plot.StepReg for visualization of results

Examples

```
## Not run:
# Load leukemia remission data
data(remission)

# Run stepwise logistic regression
formula <- remiss ~ .
result <- stepwise(
  formula = formula,
  data = remission,
  type = "logit",
  strategy = c("forward", "bidirection"),
  metric = c("AIC", "BIC")
)

# Generate reports in multiple formats
report(
  x = result,
  report_name = "leukemia_analysis",
  format = c("html", "docx")
)

## End(Not run)
```

`StepRegShinyApp` *Launch StepReg Shiny Application*

Description

Launches an interactive Shiny application for performing stepwise regression analysis. The application provides a user-friendly interface for data analysis, model selection, and visualization of results.

Usage

```
StepRegShinyApp()
```

Details

The application consists of two main steps:

Step 1: Data Preparation

- Upload custom datasets or select from built-in examples

- Configure data import settings (headers, separators, quotes)
- View and modify variable types (numeric, factor, integer, character)
- Generate exploratory data visualizations

Step 2: Model Analysis

- Select regression type:
  - Linear regression
  - Logistic regression
  - Cox proportional hazards
  - Poisson regression
  - Gamma regression
  - Negative binomial regression
- Choose dependent and independent variables
- Specify stepwise selection strategy:
  - Forward selection
  - Backward elimination
  - Bidirectional elimination
  - Best subset selection
- Set model selection criteria (AIC, BIC, etc.)
- Configure significance levels for variable entry/removal
- Generate comprehensive reports and visualizations

Value

Launches the Shiny application in the user's default web browser.

See Also

`stepwise` for the core stepwise regression function

`report` for generating analysis reports

`plot.StepReg` for visualization functions

Examples

```
## Not run:
# Launch the StepReg Shiny application
StepRegShinyApp()

## End(Not run)
```

`stepwise` *Stepwise Regression Model Selection*

Description

Performs stepwise regression model selection using various strategies and selection criteria. Supports multiple regression types including linear, logistic, Cox, Poisson, and Gamma regression.

Usage

```
stepwise(
  formula,
  data,
  type = c("linear", "logit", "cox", "poisson", "gamma", "negbin"),
  strategy = c("forward", "backward", "bidirection", "subset"),
 metric = c("AIC", "AICc", "BIC", "CP", "HQ", "adjRsq", "SL", "SBC", "IC(3/2)", "IC(1)"),
  sle = 0.15,
  sls = 0.15,
  include = NULL,
  test_method_linear = c("Pillai", "Wilks", "Hotelling-Lawley", "Roy"),
  test_method_glm = c("Rao", "LRT"),
  test_method_cox = c("efron", "breslow", "exact"),
  tolerance = 1e-07,
  weight = NULL,
  best_n = 3,
  test_ratio = 0,
  feature_ratio = 1,
  seed = 123,
  num_digits = 6
)
```

Arguments

`formula` A formula object specifying the model structure:

- Response variable(s) on left side of ~
- Predictor variable(s) on right side of ~
- Use + to separate multiple predictors
- Use * for main effect and interaction terms
- Use : for continuous-nested-within-class variable, make sure class variable is a factor variable, e.g. X:A or A:X means a continuous variable X nested within a factor variable A
- Use . to include all variables
- Use cbind() for multiple responses
- Use 0 or -1 to exclude intercept
- Use strata() to include strata variable for Cox regression

`data`	A data frame containing the variables in the model
`type`	The type of regression model to fit: • "linear" - Linear regression (default) • "logit" - Logistic regression • "poisson" - Poisson regression • "cox" - Cox proportional hazards regression • "gamma" - Gamma regression • "negbin" - Negative binomial regression
`strategy`	The model selection strategy: • "forward" - Forward selection (default) • "backward" - Backward elimination • "bidirection" - Bidirectional elimination • "subset" - Best subset selection
`metric`	The model selection criterion: • "AIC" - Akaike Information Criterion (default) • "AICc" - Corrected AIC • "BIC" - Bayesian Information Criterion • "CP" - Mallows' Cp • "HQ" - Hannan-Quinn criterion • "adjRsq" - Adjusted R-squared • "SL" - Significance Level • "SBC" - Schwarz Bayesian Criterion • "IC(3/2)" - Information Criterion with penalty 3/2 • "IC(1)" - Information Criterion with penalty 1
`sle`	Significance Level to Enter (default: 0.15). A predictor must have p-value < sle to enter the model.
`sls`	Significance Level to Stay (default: 0.15). A predictor must have p-value < sls to remain in the model.
`include`	Character vector of predictor variables that must be included in all models.
`test_method_linear`	Test method for multivariate linear regression: • "Pillai" (default) • "Wilks" • "Hotelling-Lawley" • "Roy" For univariate regression, F-test is used.
`test_method_glm`	Test method for GLM models: • "Rao" (default) • "LRT" Only "Rao" available for subset strategy.

`test_method_cox`	Test method for Cox regression: • "efron" (default) • "breslow" • "exact"
`tolerance`	Threshold for detecting multicollinearity (default: 1e-07). Lower values are more strict.
`weight`	Optional numeric vector of observation weights. Values are coerced to [0,1].
`best_n`	Maximum number of models to retain for each variable count (default: 3)
`test_ratio`	Proportion of the dataset allocated for testing (e.g., 0.3, which means 30% of the dataset is used for testing), with the remainder reserved for training, enabling train-test validation.
`feature_ratio`	Proportion of candidate features sampled uniformly at random during forward selection (default = 1). This randomized selection helps identify the best variables while reducing the risk of overfitting, and is only valid when strategy is "forward".
`seed`	Seed for random number generation (default: 123), this is only valid when test_ratio or feature_ratio is specified.
`num_digits`	Number of decimal places to round results (default: 6)

Value

A StepReg class object, which is a structured list containing both the input specifications and the outcomes of the stepwise regression analysis. The key components of this object are detailed below, providing a comprehensive framework for model exploration and validation.

- `argument` A data.frame containing the user-specified settings and parameters used in the analysis, including the initial formula, regression type, selection strategy, chosen metrics, significance levels (sle/sls), tolerance threshold, test method, and other control parameters.
- `variable` A data.frame containing information about all variables in the model, including variable names, data types (numeric, factor, etc.), and their roles (Dependent/Independent) in the model.
- `performance` A data.frame providing detailed performance metrics for the selected models across different strategies and metrics. For both training and test datasets (when test_ratio < 1), the output includes model-specific performance indicators:
  - **For linear, poisson, gamma, and negative binomial regression:**
    - `adj_r2_train/adj_r2_test`: Adjusted R-squared measures the proportion of variance explained by the model, adjusted for the number of predictors. Values range from 0 to 1, with higher values indicating better model fit. A good model should have high adjusted R-squared on both training and test data, with minimal difference between them. Large differences suggest overfitting.
    - `mse_train/mse_test`: Mean Squared Error measures the average squared difference between predicted and actual values. Lower values indicate better model performance. The test MSE should be close to training MSE; significantly higher test MSE suggests overfitting.

    * `mae_train/mae_test`: Mean Absolute Error measures the average absolute difference between predicted and actual values. Lower values indicate better model performance. Like MSE, test MAE should be close to training MAE to avoid overfitting.
  - **For logistic regression:**
    * `accuracy_train/accuracy_test`: Accuracy measures the proportion of correct predictions (true positives + true negatives) / total predictions. Values range from 0 to 1, with higher values indicating better classification performance. Test accuracy should be close to training accuracy; large differences suggest overfitting.
    * `auc_train/auc_test`: Area Under the Curve measures the model's ability to distinguish between classes. Values range from 0.5 (random) to 1.0 (perfect discrimination). AUC > 0.7 is considered acceptable, > 0.8 is good, > 0.9 is excellent. Test AUC should be close to training AUC to avoid overfitting.
    * `log_loss_train/log_loss_test`: Log Loss (logarithmic loss) penalizes confident wrong predictions more heavily. Lower values indicate better model performance. Values close to 0 are ideal. Test log loss should be close to training log loss; higher test log loss suggests overfitting.
  - **For Cox regression:**
    * `c-index_train/c-index_test`: Concordance Index (C-index) measures the model's ability to correctly rank survival times. Values range from 0.5 (random) to 1.0 (perfect ranking). C-index > 0.7 is considered acceptable, > 0.8 is good, > 0.9 is excellent. Test C-index should be close to training C-index to avoid overfitting.
    * `auc_hc`: Harrell's C-index for time-dependent AUC, measuring discrimination at specific time points. Higher values indicate better discrimination ability.
    * `auc_uno`: Uno's C-index for time-dependent AUC, providing an alternative measure of discrimination that may be more robust to censoring patterns.
    * `auc_sh`: Schemper and Henderson's C-index for time-dependent AUC, offering another perspective on model discrimination performance.

- `overview` A nested list organized by strategy and metric, containing step-by-step summaries of the model-building process. Each element shows which variables were entered or removed at each step along with the corresponding metric values (e.g., AIC, BIC, SBC).
- `detail` A nested list organized by strategy and metric, providing granular information about each candidate step. This includes which variables were tested, their evaluation statistics, p-values, and whether they were ultimately selected or rejected.
- `fitted model object within the strategy-specific list` A nested list object organized with a first layer representing the selection strategy (e.g., forward, backward, bidirection, subset) and a second layer representing the metric (e.g., AIC, BIC, SBC). For each strategy-metric combination, the function returns fitted model objects that can be further analyzed using S3 generic functions such as `summary()`, `anova()`, or `coefficients()`. These functions adapt to the model type (e.g., `coxph`, `lm`, `glm`) through call-specific methods. Specific statistics can be directly retrieved using the `$` operator, such as `result$forward$AIC$coefficients`. The level of detail in these analyses depends on the model type: the **survival** package enriches `coxph` objects with detailed statistics including hazard ratios, standard errors, z-statistics, p-values, and likelihood ratio tests, while base R functions like `lm` and `glm` offer basic output with coefficients by default, requiring `summary()` or `anova()` to reveal standard errors, t-values, p-values, and R-squared values.

Author(s)

Junhui Li, Kai Hu, Xiaohuan Lu

References

- Alsubaihi et al. (2002) Variable strategy in multivariable regression using sas/iml
- Darlington (1968) Multiple regression in psychological research and practice
- Dharmawansa et al. (2014) Roy's largest root under rank-one alternatives
- Hannan & Quinn (1979) The determination of the order of an autoregression
- Hotelling (1992) The Generalization of Student's Ratio
- Hocking (1976) The analysis and strategy of variables in linear regression
- Hurvich & Tsai (1989) Regression and time series model strategy in small samples
- Judge (1985) The Theory and practice of econometrics
- Mallows (1973) Some comments on cp
- Mardia et al. (1979) Multivariate analysis
- Mckeon (1974) F approximations to the distribution of hotelling's t20
- Mcquarrie & Tsai (1998) Regression and Time Series Model strategy
- Pillai (1955) Some new test criteria in multivariate analysis
- Sparks et al. (1985) On variable strategy in multivariate regression
- Sawa (1978) Information criteria for discriminating among alternative regression models
- Schwarz (1978) Estimating the dimension of a model

Examples

```
# Multivariate linear regression with bidirectional selection
data(mtcars)
formula <- cbind(mpg, drat) ~ . + 0
result1 <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = "bidirection",
  metric = "AIC"
)

summary(result1$bidirection$AIC)
anova(result1$bidirection$AIC)
coefficients(result1$bidirection$AIC)

# Linear regression with multiple strategies and metrics
formula <- mpg ~ . + 1
result2 <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = c("forward", "bidirection"),
  metric = c("AIC", "SBC", "SL", "AICc", "BIC", "HQ")
)

summary(result2$forward$AIC)
anova(result2$forward$AIC)
coefficients(result2$forward$AIC)

# Logistic regression with significance level criteria
data(remission)
formula <- remiss ~ .
result3 <- stepwise(
  formula = formula,
  data = remission,
  type = "logit",
  strategy = "forward",
  metric = "SL",
  sle = 0.05,
  sls = 0.05
)

summary(result3$forward$SL)
anova(result3$forward$SL)
coefficients(result3$forward$SL)

# Linear regression with continuous-nested-within-class effects
mtcars$am <- factor(mtcars$am)
formula <- mpg ~ am + cyl + wt:am + disp:am + hp:am
result4 <- stepwise(
  formula = formula,
  data = mtcars,
  type = "linear",
  strategy = "bidirection",
  metric = "AIC"
)

summary(result4$bidirection$AIC)
anova(result4$bidirection$AIC)
coefficients(result4$bidirection$AIC)
```

`tobacco` *Tobacco Leaf Chemical Composition Dataset*

Description

A dataset containing chemical composition measurements from 25 tobacco leaf samples. This dataset is commonly used for demonstrating multivariate regression analysis and exploring relationships between chemical components and burn rate.

Usage

```
data(tobacco)
```

Format

A data frame with 25 rows and 9 variables:

cigarette Numeric. Rate of cigarette burn in inches per 1000 seconds.

sugar Numeric. Percentage of sugar content in the leaf.

nicotine Numeric. Percentage of nicotine content.

nitrogen Numeric. Percentage of nitrogen content.

chlorine Numeric. Percentage of chlorine content.

potassium Numeric. Percentage of potassium content.

phosphorus Numeric. Percentage of phosphorus content.

calcium Numeric. Percentage of calcium content.

magnesium Numeric. Percentage of magnesium content.

Details

This dataset is particularly useful for:

- Demonstrating multivariate regression analysis
- Studying relationships between chemical composition and burn rate
- Analyzing correlations between different chemical components
- Teaching statistical methods in agricultural research

Source

Anderson, R. L. and Bancroft, T. A. (1952), Statistical Theory in Research, McGraw-Hill Book Company, Inc., New York, NY.

Examples

```
# Load the dataset
data(tobacco)

# View the first few rows
head(tobacco)

# Summary statistics
summary(tobacco)

# Correlation analysis
cor(tobacco)
```

Index