Package 'metaSVR'

June 25, 2025

Type Package

Title Hybrid Support Vector Regression with Metaheuristic Optimization

Version 0.1.0

Author Rechtiana Putri Arini, Robert Kurniawan

Maintainer Rechtiana <rparini17@gmail.com>

Description This package provides hybrid of SVR with several metaheuristic algorithms. The use of metaheuristic algorithm is to optimize the parameter of Cost, Gamma, and Epsilon so it can get better evaluation. Metaheuristic classification used in this package are: Coot Bird Optimization (CBO), Archimedes Optimization (AO), Combined Archimedes Optimization with Coot Bird Optimization (AOCBO), Harris Hawks Optimization (HHO), Gray Wolf Optimizer (GWO), Ant Lion Optimization (ALO), and Enhanced Harris Hawk Optimization with Coot Bird Optimization (EHHOCBO)

Imports e1071, stats, dplyr, readxl, hms

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/rechtianaputri/metaSVR

BugReports https://github.com/rechtianaputri/metaSVR/issues

RoxygenNote 7.3.1

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

R topics documented:

`ALO` *Ant Lion Optimizer*

Description

An algorithm built by Mirjalili (2015) inspired by the hunting behaviour of antlion whose making pit trap for ant prey in order to optimized real-valued objective function in continuous search space in a population-based manner.

Usage

```
ALO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

`N`	An integer indicate population size.
`Max_iter`	An integer indicate maximum number of iterations.
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`fobj`	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

The algorithm mimics the ALO hunting behaviour by simulating a stochastic search where ants move around randomly under the influence of selected antlions and an elite antlion.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors 'lb' and 'ub' must have the same length as the number of dimensions 'dim'.

This optimization function used inside svrHybrid function.

References

Mirjalili, S. (2015). The ant lion optimizer. Advances in engineering software, 83, 80-98. https://doi.org/10.1016/j.adveng

---

AO *Archimedes Optimization*

Description

An algorithm built by Hashim et al. (2021) use buoyancy law and fluid dynamics behavior in Archimedes principle to optimized real-valued objective function in continuous search space in a population-based manner.

Usage

```
AO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

`N`	An integer indicate population size.
`Max_iter`	An integer indicate maximum number of iterations.
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`fobj`	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

This algorithm uses population-based search to conduct physical law such as volume, density difference, and acceleration in every iteration. It balancing the exploration and exploitation phase by using Transfer Function (TF) as a shifting indicates.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors 'lb' and 'ub' must have the same length as the number of dimensions 'dim'.

This optimization function used inside svrHybrid function.

Constant of C3 = 1 and C4 = 2 used in basic standard optimization function.

References

Hashim, F. A., Hussain, K., Houssein, E. H., Mabrouk, M. S., & Al-Atabany, W. (2021). Archimedes Optimization Algorithm: A New Metaheuristic Algorithm for Solving Optimization Problems. Applied Intelligence, 51(3), 1531–1551. https://doi.org/10.1007/s10489-020-01893-z

---

`AOCBO` *Combined Archimedes Optimization with Coot Bird Optimization*

Description

A hybrid metaheuristic algorithm that combines Archimedes Optimization (AO) with Coot Bird Optimization (CBO) to optimized real-valued objective function in continuous search space.

Usage

```
AOCBO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

`N`	An integer indicate population size.
`Max_iter`	An integer indicate maximum number of iterations.
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension.

`dim` An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.

`fobj` An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

This metaheuristic implement combination of all step of Archimedes Optimization with first step used after initialization is Coot Leader selection stage in CBO as early exploration step. The hybrid design enhances convergence and stability in optimization step so it can maximize the best parameter.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors ‘lb‘ and ‘ub‘ must have the same length as the number of dimensions ‘dim‘.

This optimization function used inside svrHybrid function.

`CBO` *Coot Bird Optimization*

Description

An algorithm built by Naruei & Keynia (2021) that mimics the regular-irregular movement behaviour of Coot birds. Its population divided by two groups as leaders to guide the process and coots to follow leaders and randomly explore search space. This movement use to optimized real-valued objective function in continuous search space.

Usage

```
CBO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

N	An integer indicate population size.
Max_iter	An integer indicate maximum number of iterations.
lb	A numeric vector that show lower bounds of the search space. One value per dimension.
ub	A numeric vector that show upper bounds of the search space. One value per dimension.
dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
fobj	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

This algorithms used movement such as: random movement, chain movement, adjusting the position based on the group leaders, and leader movement to emphasize the exploration and exploitation phase to get the best fitness.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors 'lb' and 'ub' must have the same length as the number of dimensions 'dim'.

This optimization function used inside svrHybrid function.

References

Naruei, I., & Keynia, F. (2021). A New Optimization Method Based on COOT Bird Natural Life Model. Expert Systems with Applications, 183. https://doi.org/10.1016/j.eswa.2021.115352

denormalize *Denormalize*

Description

Convert normalized data back to original scale using given min and max.

Usage

```
denormalize(x, min, max)
```

Arguments

x	Numeric vector that has been normalized (values is between 0 and 1).
min	The minimum value of the original data.
max	The maximum value of the original data.

Value

A numeric vector already converted to original scale.

Examples

```
# Example of the original data
original_data <- c(10, 20, 30, 40, 50)

# Data being normalized
normalized_data <- (original_data - min(original_data)) /
                   (max(original_data) - min(original_data))

# Denormalization function use to change value to the original
denormalize(normalized_data, min(original_data), max(original_data))
```

EHHOCBO *Enhanced Harris Hawks Optimization with Coot Bird Optimization*

Description

This function implements a hybrid metaheuristic optimization algorithm that combines Harris Hawks Optimization with leader selection of Coot Bird Optimization to optimized real-valued objective function in continuous search space in a population-based manner built by Cui et al. (2023).

Usage

```
EHHOCBO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

N	An integer indicate population size.
Max_iter	An integer indicate maximum number of iterations.
lb	A numeric vector that show lower bounds of the search space. One value per dimension.
ub	A numeric vector that show upper bounds of the search space. One value per dimension.
dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
fobj	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

This algorithm start by adding leadership mechanism of CBO into HHO process so it can make better foundation for the global search. Ensemble Mutation Strategy (EMS) to improve the exploration trend and population diversity also Refracted Opposition-Based Learning (ROBL) to update current optimal solution in the swarm added to enhanced the combination of HHO and CBO.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors ‘lb‘ and ‘ub‘ must have the same length as the number of dimensions ‘dim‘.

This optimization function used inside svrHybrid function.

References

Cui, H., Guo, Y., Xiao, Y., Wang, Y., Li, J., Zhang, Y., & Zhang, H. (2023). Enhanced Harris Hawks Optimization Integrated with Coot Bird Optimization for Solving Continuous Numerical Optimization Problems. CMES - Computer Modeling in Engineering and Sciences, 137(2), 1635–1675. https://doi.org/10.32604/cmes.2023.026019

get_default_bounds *Default Bounds Initialization for SVR Optimization*

Description

This function return the default value of lower and upper bounds also the dimension for SVR optimization. The three dimensions represent as the parameter that need to be optimized in SVR with exact range of bound. Three dimension and the range represent as: Cost (C): 2^0 to 2^10; Gamma: 2^(-8) to 2^0; Epsilon: 2^(-8) to 2^0.

Usage

```
get_default_bounds()
```

Value

A list containing:

lb A numeric vector of lower bounds.

ub A numeric vector of upper bounds.

dim An integer representing the number of dimensions, 3.

Examples

```
bounds <- get_default_bounds()
bounds$lb  # Lower bounds
bounds$ub  # Upper bounds
bounds$dim # Number of parameters
```

GWO *Grey Wolf Optimizer*

Description

An algorithm built by Mirjalili et al. (2014) inspired by leadership hierarchy and hunting mechanism of grey wolves in nature to optimized real-valued objective function in continuous search space in a population-based manner.

Usage

```
GWO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

N	An integer indicate population size.
Max_iter	An integer indicate maximum number of iterations.
lb	A numeric vector that show lower bounds of the search space. One value per dimension.
ub	A numeric vector that show upper bounds of the search space. One value per dimension.
dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
fobj	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

This algorithm proposed social hierarchy on GWO to obtain the best fitness and get the best proposed hunting method to locate probable position of the pray. Adaptive values on alpha and A make it possible smooth transition between exploration and exploitation phase.

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors 'lb' and 'ub' must have the same length as the number of dimensions 'dim'.

This optimization function used inside svrHybrid function.

References

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

HHO *Harris Hawks Optimization*

Description

An algorithm built by Heidari et al. (2019) that inspired by the movement of Harris Hawks on cooperative hunting behaviour to optimized real-valued objective function in continous search space in a population-based manner.

Usage

```
HHO(N, Max_iter, lb, ub, dim, fobj)
```

Arguments

`N`	An integer indicate population size.
`Max_iter`	An integer indicate maximum number of iterations.
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`fobj`	An objective function used to be minimized. It is return single numeric value that show evaluation matrix result in every iteration. It used to calculate the best fitness in every iteration.

Details

There are two phase of Harris Hawks hunting, namely exploration and exploitation that will be modelized to find optimization result. The movement used in this algorithms such as: exploration phase; transition between exploitation and exploration phase; and exploitation phase that has 4 different strategies based on E and r (soft besiege, hard besiege, soft besiege with progressive rapid, and hard besiege with progressive rapid)

The algorithm performs until maximum iteration reached or convergence condition when the difference in objective values for ten consecutive times is less than 10^-5.

Value

A list containing:

best_fitness The best (minimum) fitness value found.

best_position The parameter vector (position) corresponding to the best fitness.

jml_iter The number of iterations executed.

param Matrix of best parameters found across every iterations (dim × iter).

param_list Vector of best fitness values at each iteration.

Note

The input vectors ‘lb‘ and ‘ub‘ must have the same length as the number of dimensions ‘dim‘.

This optimization function used inside svrHybrid function.

References

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872. https://doi.org/10.1016/j.future.2019.02.028

`initALO` *Initialize Position on Ant Lion Optimizer*

Description

This function generates the initial position of antlions and ants within the defined upper and lower bound in every dimension.

Usage

```
initALO(N, dim, ub, lb)
```

Arguments

`N`	An integer indicate population size.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension.
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.

Value

A numeric matrix of shape (`N, dim`) representing initialized positions.

Note

This function used inside ALO function for initialization process.

`initCBO` *Initialize Position on Coot Bird Optimization*

Description

This function generates the initial position of leaders and coots within the defined upper and lower bound in every dimension.

Usage

```
initCBO(N, dim, ub, lb)
```

Arguments

`N`	An integer indicate population size.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.

Value

A numeric matrix of shape `(N, dim)` representing initialized positions.

Note

This function used inside CBO function for initialization process.

`initEHHOCBO` *Initialize Position on Enhanced Harris Hawks Optimization with Coot Bird Optimization*

Description

This function generates the initial position of all agents (X) within the defined upper and lower bound in every dimension.

Usage

```
initEHHOCBO(N, dim, ub, lb)
```

Arguments

N	An integer indicate population size.
dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
ub	A numeric vector that show upper bounds of the search space. One value per dimension
lb	A numeric vector that show lower bounds of the search space. One value per dimension.

Value

A numeric matrix of shape (N, dim) representing initialized positions.

Note

This function used inside EHHOCBO function for initialization process.

---

initGWO *Initialize Position on Grey Wolf Optimizer*

Description

This function generates the initial position of gray wolf within the defined upper and lower bound in every dimension.

Usage

```
initGWO(N, dim, ub, lb)
```

Arguments

N	An integer indicate population size.
dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
ub	A numeric vector that show upper bounds of the search space. One value per dimension
lb	A numeric vector that show lower bounds of the search space. One value per dimension.

Value

A numeric matrix of shape (N, dim) representing initialized positions.

Note

This function used inside GWO function for initialization process.

`initHHO` *Initialize Position on Harris Hawks Optimization*

Description

This function generates the initial position of Harris Hawk agents within the defined upper and lower bound in every dimension.

Usage

```
initHHO(N, dim, ub, lb)
```

Arguments

`N`	An integer indicate population size.
`dim`	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
`ub`	A numeric vector that show upper bounds of the search space. One value per dimension
`lb`	A numeric vector that show lower bounds of the search space. One value per dimension.

Value

A numeric matrix of shape (`N`, `dim`) representing initialized positions.

Note

This function used inside HHO function for initialization process.

`levyEHHOCBO` *Levy Flight Generator*

Description

Generates a random step vector based on Lévy flight distribution, used in the exploitation phase of HHO that used as combined in EHHOCBO.

Usage

```
levyEHHOCBO(dim)
```

Arguments

`dim`	An integer that indicate the dimensionality of search space.

Value

A numeric vector of length `dim` representing the Lévy flight step.

#' @note This function used inside EHHOCBO function to generate random Levy Flight vector.

levyHHO *Levy Flight Generator*

Description

Generates a random step vector based on Lévy flight distribution, used in the exploitation phase of the HHO algorithm.

Usage

```
levyHHO(dim)
```

Arguments

dim	An integer that indicate the dimensionality of search space.

Value

A numeric vector of length dim representing the Lévy flight step.

Note

This function used inside HHO function to generate random Levy Flight vector.

loss_calculate *Calculate Loss Based on Selected Objective Function*

Description

Compute the loss between predictive and actual values using a selected objective function. Supported objecive functions used in this functions are: "SMAPE', "MAPE", "RMSE", and "MAE".

Usage

```
loss_calculate(preds, actuals, objective)
```

Arguments

preds	A numeric vector of predicted values.
actuals	A numeric vector of actual (true) values.
objective	A string character that indicates the loss function type: "SMAPE", "MAPE", "RMSE", or "MAE".

Value

A numeric value that represent the computed loss.

Examples

```
preds <- c(80, 120, 180)
actuals <- c(95, 115, 177)
loss_calculate(preds, actuals, "RMSE")
```

---

mae *Mean Absolute Error*

Description

Calculate the RMSE value between predicted and actual values.

Usage

```
mae(preds, actuals)
```

Arguments

preds	A numeric vector of predicted values.
actuals	A numeric vector of actual (true) values.

Value

MAE value.

Examples

```
preds <- c(80, 120, 180)
actuals <- c(95, 115, 177)
mae(preds, actuals)
```

---

mape *Mean Absolute Percentage Error*

Description

Calculate the MAPE value between predicted and actual values. Can't be used if the actual values contain 0 value.

Usage

```
mape(preds, actuals)
```

Arguments

preds	A numeric vector of predicted values.
actuals	A numeric vector of actual (true) values.

Value

MAPE value (percentage).

Examples

```
preds <- c(80, 120, 180)
actuals <- c(95, 115, 177)
mape(preds, actuals)
```

`normalize` *Normalize*

Description

Normalize data using min-max scale.

Usage

```
normalize(x)
```

Arguments

`x`	is a predictor variable that is a numeric vector to be normalized.

Value

A numeric vector scaled between 0 and 1.

Examples

```
# Normalize example use:
data <- c(10, 20, 30, 40, 50)
normalize(data)
```

`Random_walk_around_antlion`
Perform Random Walk Around Antlion

Description

Function simulates random walk of an ant within the boundaries influenced by an antlion's position.

Usage

```
Random_walk_around_antlion(dim, Max_iter, lb, ub, antlion, current_iter)
```

Arguments

dim	An integer show the number of dimension (parameters) of the problem to optimize. It indicate the number of parameters to be optimized.
Max_iter	An integer indicate maximum number of iterations.
lb	A numeric vector that show lower bounds of the search space. One value per dimension.
ub	A numeric vector that show upper bounds of the search space. One value per dimension.
antlion	A numeric vector representing the position of the selected antlion.
current_iter	The current iteration count.

Value

A numeric matrix of shape (N, dim) representing the position of the ant in each step of the random walk.

Note

This function used inside ALO function to update the position of ants.

---

rmse *Root Mean Squared Error*

Description

Calculate the RMSE value between predicted and actual values.

Usage

```
rmse(preds, actuals)
```

Arguments

preds	A numeric vector of predicted values.
actuals	A numeric vector of actual (true) values.

Value

RMSE value.

Examples

```
preds <- c(80, 120, 180)
actuals <- c(95, 115, 177)
rmse(preds, actuals)
```

RouletteWheelSelection *Roulette Wheel Selection*

Description

Function used to select an individual index based on fitness-proportional selection (inverse fitness weight).

Usage

```
RouletteWheelSelection(weights)
```

Arguments

weights	A numeric vector of weights.

Value

An integer representing the selected index.

Note

This function used inside ALO function to probabilistically select antlions for guiding ants.

smape *Symmetric Mean Absolute Percentage Error*

Description

Calculate the SMAPE value between predicted and actual values.

Usage

```
smape(preds, actuals)
```

Arguments

preds	A numeric vector of predicted values.
actuals	A numeric vector of actual (true) values.

Value

SMAPE value (percentage).

Examples

```
preds <- c(80, 120, 180)
actuals <- c(95, 115, 177)
smape(preds, actuals)
```

svrHybrid *Hybrid SVR Optimization with Metaheuristic Algorithms*

Description

Trains a Support vector Regression Model by optimizing its parameter (Cost, Gamma, and Epsilon) using Metaheuristic Algorithms such as: Archimedes Optimization (AO), Coot Bird Optimization (CBO), Combined Archimedes Optimization with Coot Bird Optimization (AOCBO), Harris Hawks Optimization (HHO), Grey Wolf Optimizer (GWO), Ant Lion Optimization (ALO), and Enhanced Harris Hawks Optimization with Coot Bird Optimization (EHHOCBO).

Usage

```
svrHybrid(
  x_train,
  y_train,
  x_test,
  y_test,
  kernel = "radial",
  optimizer = "AO",
  objective = "RMSE",
  is.y.normalize = FALSE,
  min.y = min.y,
  max.y = max.y,
  max_iter = 100,
  N = 30,
  seed = 123,
  degree = 3,
  coef0 = 0,
  nu = 0.5,
  class.weights = NULL,
  cachesize = 40,
  tolerance = 0.001,
  scale = TRUE,
  shrinking = TRUE,
  cross = 0,
  probability = FALSE,
  fitted = TRUE,
  ...,
  subset,
  na.action = na.omit
)
```

Arguments

x_train	A matrix or data frame contain predictors variable for training the model.
y_train	A numeric vector of target values for training model.
x_test	A matrix or data frame contain predictors variable for testing the model. It can be replaced by data validation to get the parameter if you separated the data as three categories.

y_test	A numeric vector of target values for training model. It can be replaced by data validation to get the parameter if you separated the data as three categories.
kernel	SVR kernel type used for modelling. Options: "radial", "polynomial", and "sigmoid". Default is radial.
optimizer	Metaheuristic Algorithms selection, such as: "AO", "CBO", "AOCBO", "HHO", "GWO", "ALO", and "EHHOCBO". Default is AO.
objective	Objective function used for optimization as prediction quality measures. Options: "SMAPE", "MAPE", "RMSE", and "MAE". Default is RMSE.
is.y.normalize	Logical; use when prediction of target variable 'y' is on min-max scalling normalization. Default is FALSE.
min.y	Minimum value of target (used for denormalization).
max.y	Maximum value of target (used for denormalization).
max_iter	Maximum number of iterations for the optimizer. Default is 100.
N	Population size for the optimizer. Default is 30.
seed	Random seed for reproducibility. Default is 123.
degree	Degree parameter for polynomial kernel.
coef0	Coefficient parameter used in polynomial/sigmoid kernels.
nu	Parameter for 'nu-regression' to controlling max proportion of error training and minimum proportion of support vectors. Default is 0.5, range: 0.1-0.9. Only use if the type of regression choosen is 'nu-regression'.
class.weights	A named list of class weights.
cachesize	Size of kernel cache (in MB). Default is 40.
tolerance	Tolerance of termination criterion.
scale	Logical; whether to scale inputs. Default is TRUE.
shrinking	Logical; whether to use shrinking heuristics. Default is TRUE.
cross	Number of folds for cross-validation. Default is 0, no cross validation.
probability	Logical; whether to enable probability model. Default is FALSE.
fitted	Logical; whether to keep fitted values. Default is TRUE.
...	Additional arguments passed to 'svm()'.
subset	Optional vector specifying subset of observations to be used in the training fit.
na.action	Function which indicates what should happen when the data contain NAs.

Value

A list containing:

best_params A list with the best values for 'cost', 'gamma', and 'epsilon'.

total_iter Total number of iterations run by the optimizer.

model The final trained SVR model (using 'e1071::svm').

time Total training time in HMS format.

Examples

```
## Not run:
set.seed(1)
x <- matrix(rnorm(100), ncol = 2)
y <- x[,1] * 3 + rnorm(50)
model <- svrHybrid(x_train = x[1:40,], y_train = y[1:40],
                   x_test = x[41:50,], y_test = y[41:50],
                   kernel = "radial", optimizer = "AO",
                   objective = "RMSE", is.y.normalize = FALSE)
model$best_params

## End(Not run)
```

Index