

Package ‘twinsvm’

June 9, 2026

Title Twin Support Vector Machines

Version 0.0.2

Author Shamika Tissera [aut, cre]

Maintainer Shamika Tissera <nimeshshamika@gmail.com>

Description Provides twin support vector machine classifiers and visualization tools for small to moderate classification problems. Includes one-vs-one multi-class classification and a standard support vector machine baseline for comparison.

License GPL-3

Encoding UTF-8

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppArmadillo

Imports ggplot2, Rcpp, rlang

Suggests e1071, ganimate, knitr, plotly, rmarkdown, shiny, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-06-09 21:40:02 UTC

Contents

twinsvm-package	2
coef.tsvm	3
compare_methods	3
confusion	4
cv_tsvm	5
gen_circles	6

gen_moons	6
kernel_lift	7
lift_plot	8
lift_plotly	8
morph_boundary	9
plot.cv_tsvm	10
plot.svms	11
plot.tsvm	11
predict.svms	12
predict.svms_multiclass	13
predict.tsvm_multiclass	14
print.svms_multiclass	15
print.tsvm_multiclass	15
svms	16
tsvm	17
twomoons	19
Index	20

twinsvm-package

twinsvm: Twin Support Vector Machines

Description

Twin support vector machine classifiers with a standard C-SVC baseline.

Author(s)

Maintainer: Shamika Tissera <nimeshshamika@gmail.com>

Authors:

- Shamika Tissera <nimeshshamika@gmail.com>

References

Jayadeva, Khemchandani, R., and Chandra, S. (2007). Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5), 905-910.

Kumar, M. A. and Gopal, M. (2009). Least squares twin support vector machines for pattern classification. *Expert Systems with Applications*, 36(4), 7535-7543.

coef.tsvm	<i>Extract Twin-SVM Coefficients</i>
-----------	--------------------------------------

Description

Extract Twin-SVM Coefficients

Usage

```
## S3 method for class 'tsvm'
coef(object, ...)
```

Arguments

object	A fitted tsvm object.
...	Unused.

Value

A list with the two plane coefficients.

Examples

```
dat <- gen_moons(30)
fit <- tsvm(dat$x, dat$y)
coef(fit)
```

compare_methods	<i>Compare Twin-SVM and Standard SVM Boundaries</i>
-----------------	---

Description

Fits least-squares twin SVM, original QP twin SVM, and the standard C-SVC SVM baseline on the same two-dimensional data, then plots the three decision boundaries side by side.

Usage

```
compare_methods(x, y, kernel = "rbf", gamma = 0.5, c1 = 1, c2 = 1, cost = 1)
```

Arguments

x	Numeric two-column matrix or data frame.
y	Two-class response.
kernel	Kernel name: "linear", "rbf", or "poly".
gamma	Kernel scale.
c1, c2	Positive twin-SVM regularization parameters.
cost	Positive C-SVC cost parameter.

Value

A faceted ggplot object.

See Also

Other visualization: [kernel_lift\(\)](#), [lift_plot\(\)](#), [lift_plotly\(\)](#)

Examples

```
set.seed(30)
dat <- gen_moons(50, noise = 0.1)
compare_methods(dat$x, dat$y, gamma = 1, c1 = 0.2, c2 = 0.2, cost = 1)
```

confusion

Confusion Matrix and Accuracy

Description

Predicts on x and returns a confusion matrix plus overall accuracy. This works for both binary and one-vs-one multiclass `tsvm` and `svms` fits.

Usage

```
confusion(object, x, y)
```

Arguments

<code>object</code>	A fitted <code>tsvm</code> , <code>tsvm_multiclass</code> , <code>svms</code> , or <code>svms_multiclass</code> object.
<code>x</code>	Numeric matrix or data frame of predictors.
<code>y</code>	True class labels.

Value

A list with `table`, a `table(truth, predicted)` confusion matrix, and `accuracy`, the overall classification accuracy.

See Also

Other multiclass: [predict.svms_multiclass\(\)](#), [predict.tsvm_multiclass\(\)](#), [print.svms_multiclass\(\)](#), [print.tsvm_multiclass\(\)](#)

Examples

```

set.seed(50)
x <- rbind(
  matrix(rnorm(20, -2, 0.2), ncol = 2),
  matrix(rnorm(20, 0, 0.2), ncol = 2),
  matrix(rnorm(20, 2, 0.2), ncol = 2)
)
y <- factor(rep(c("a", "b", "c"), each = 10))
fit <- tsvm(x, y, kernel = "linear")
confusion(fit, x, y)

```

cv_tsvm

*Cross-Validate Twin-SVM Hyperparameters***Description**

Runs k-fold cross-validation over c1, c2, and optionally gamma.

Usage

```
cv_tsvm(x, y, c1_grid, c2_grid, gamma_grid = NULL, k = 5, ...)
```

Arguments

x	Numeric matrix or data frame of predictors.
y	Two-class response.
c1_grid, c2_grid	Positive numeric vectors.
gamma_grid	Optional positive numeric vector. If NULL, gamma is left at the tsvm() default.
k	Number of folds.
...	Additional arguments passed to tsvm().

Value

A cv_tsvm object with best_params and results.

Examples

```

set.seed(10)
dat <- gen_moons(40, noise = 0.1)
cv <- cv_tsvm(dat$x, dat$y, c1_grid = c(0.1, 1), c2_grid = c(0.1, 1), k = 3)
cv$best_params

```

gen_circles	<i>Generate Concentric Circles Data</i>
-------------	---

Description

Generate Concentric Circles Data

Usage

```
gen_circles(n, noise = 0.05)
```

Arguments

n	Number of observations.
noise	Standard deviation of Gaussian noise.

Value

A list with numeric matrix x and factor y.

Examples

```
dat <- gen_circles(20, noise = 0.05)
str(dat)
```

gen_moons	<i>Generate Two-Moons Data</i>
-----------	--------------------------------

Description

Generate Two-Moons Data

Usage

```
gen_moons(n, noise = 0.2)
```

Arguments

n	Number of observations.
noise	Standard deviation of Gaussian noise.

Value

A list with numeric matrix x and factor y.

Examples

```
dat <- gen_moons(20, noise = 0.1)
str(dat)
```

`kernel_lift`*Compatibility Alias for the RBF Kernel Lift Plot*

Description

`kernel_lift()` is kept for existing code. New code should call [lift_plot\(\)](#).

Usage

```
kernel_lift(x, y, gamma = 1, center = NULL)
```

Arguments

<code>x</code>	Numeric two-column matrix or data frame.
<code>y</code>	Two-class response.
<code>gamma</code>	Positive RBF scale.
<code>center</code>	Optional numeric length-two center for the RBF bump. If <code>NULL</code> , the centroid of the inner class is used.

Value

A `ggplot` object.

See Also

Other visualization: [compare_methods\(\)](#), [lift_plot\(\)](#), [lift_plotly\(\)](#)

Examples

```
set.seed(22)
dat <- gen_circles(60, noise = 0.04)
kernel_lift(dat$x, dat$y, gamma = 1)
```

lift_plot	<i>Static RBF Kernel Lift Plot</i>
-----------	------------------------------------

Description

Maps two-dimensional data to a third RBF height and draws a static oblique projection. The height is $\exp(-\gamma * ||x - center||^2)$, so points near the center rise while points farther away stay low. A translucent plane is drawn halfway between the two class mean heights.

Usage

```
lift_plot(x, y, gamma = 1, center = NULL)
```

Arguments

x	Numeric two-column matrix or data frame.
y	Two-class response.
gamma	Positive RBF scale.
center	Optional numeric length-two center for the RBF bump. If NULL, the centroid of the inner class is used.

Value

A ggplot object.

See Also

Other visualization: [compare_methods\(\)](#), [kernel_lift\(\)](#), [lift_plotly\(\)](#)

Examples

```
set.seed(20)
dat <- gen_circles(80, noise = 0.04)
lift_plot(dat$x, dat$y, gamma = 1)
```

lift_plotly	<i>Interactive RBF Kernel Lift Plot</i>
-------------	---

Description

Draws the same RBF lift as [lift_plot\(\)](#) as a rotatable plotly 3D chart. This function is optional; the package does not require plotly to build or check.

Usage

```
lift_plotly(x, y, gamma = 1, center = NULL)
```

Arguments

x	Numeric two-column matrix or data frame.
y	Two-class response.
gamma	Positive RBF scale.
center	Optional numeric length-two center for the RBF bump. If NULL, the centroid of the inner class is used.

Value

A plotly object.

See Also

Other visualization: [compare_methods\(\)](#), [kernel_lift\(\)](#), [lift_plot\(\)](#)

Examples

```
set.seed(21)
dat <- gen_circles(80, noise = 0.04)
if (requireNamespace("plotly", quietly = TRUE)) {
  lift_plotly(dat$x, dat$y, gamma = 1)
}
```

morph_boundary

Animate a Decision Boundary Across a Hyperparameter Range

Description

Refits a model over a sequence of hyperparameter values and returns a gganimate object showing the boundary change.

Usage

```
morph_boundary(
  x,
  y,
  param = c("gamma", "cost", "c1"),
  range,
  model = c("tsvm", "svms"),
  n = 30,
  ...
)
```

Arguments

x	Numeric two-column matrix or data frame.
y	Two-class response.
param	Hyperparameter to vary.
range	Numeric length-two range for the hyperparameter.
model	Model family: "tsvm" or "svms".
n	Number of frames.
...	Additional arguments passed to the model fit function.

Value

A gganim object.

Examples

```
if (interactive()) {
  dat <- gen_moons(40)
  morph_boundary(dat$x, dat$y, param = "gamma", range = c(0.5, 2), n = 4)
}
```

plot.cv_tsvm

Plot Twin-SVM Cross-Validation Results

Description

Plot Twin-SVM Cross-Validation Results

Usage

```
## S3 method for class 'cv_tsvm'
plot(x, ...)
```

Arguments

x	A cv_tsvm object.
...	Unused.

Value

A ggplot object.

Examples

```
set.seed(11)
dat <- gen_moons(40, noise = 0.1)
cv <- cv_tsvm(dat$x, dat$y, c1_grid = c(0.1, 1), c2_grid = c(0.1, 1), k = 3)
plot(cv)
```

plot.svms *Plot a Standard SVM Decision Boundary*

Description

Plot a Standard SVM Decision Boundary

Usage

```
## S3 method for class 'svms'  
plot(x, ...)
```

Arguments

x	A fitted svms object.
...	Unused.

Value

A ggplot object.

Examples

```
dat <- gen_moons(40)  
fit <- svms(dat$x, dat$y, kernel = "linear")  
plot(fit)
```

plot.tsvm *Plot a Twin-SVM Decision Boundary*

Description

Plot a Twin-SVM Decision Boundary

Usage

```
## S3 method for class 'tsvm'  
plot(x, ...)
```

Arguments

x	A fitted tsvm object.
...	Unused.

Value

A ggplot object.

Examples

```
dat <- gen_moons(40)
fit <- tsvm(dat$x, dat$y)
plot(fit)
```

predict.svms

Predict from a Standard SVM

Description

Predict from a Standard SVM

Usage

```
## S3 method for class 'svms'
predict(object, newdata, decision.values = FALSE, ...)
```

Arguments

object	A fitted svms object.
newdata	Numeric matrix or data frame.
decision.values	If TRUE, return raw decision values instead of class labels.
...	Unused.

Value

A factor of predicted classes, or a numeric vector when decision.values = TRUE.

Examples

```
set.seed(2)
dat <- gen_moons(30)
fit <- svms(dat$x, dat$y)
predict(fit, dat$x, decision.values = TRUE)
```

`predict.svms_multiclass`*Predict from a Multiclass Standard SVM*

Description

Predicts from a one-vs-one multiclass standard SVM. Each binary model votes for one class. Ties are resolved deterministically by choosing the class that appears first in the training factor level order.

Usage

```
## S3 method for class 'svms_multiclass'
predict(object, newdata, type = c("class", "votes"), ...)
```

Arguments

<code>object</code>	A fitted <code>svms_multiclass</code> object.
<code>newdata</code>	Numeric matrix or data frame.
<code>type</code>	Output type. "class" returns predicted class labels; "votes" returns the vote matrix.
<code>...</code>	Unused. <code>decision.values</code> is not supported for multiclass objects because OVO decision values do not have a single unambiguous scale.

Value

A factor of predicted classes, or an integer vote matrix when `type = "votes"`.

See Also

Other multiclass: [confusion\(\)](#), [predict.tsvm_multiclass\(\)](#), [print.svms_multiclass\(\)](#), [print.tsvm_multiclass\(\)](#)

Examples

```
set.seed(45)
x <- rbind(
  matrix(rnorm(8, -2, 0.2), ncol = 2),
  matrix(rnorm(8, 0, 0.2), ncol = 2),
  matrix(rnorm(8, 2, 0.2), ncol = 2)
)
y <- factor(rep(c("a", "b", "c"), each = 4))
fit <- svms(x, y, kernel = "linear", max_passes = 2, max_iter = 100)
predict(fit, x[1:3, , drop = FALSE])
```

```
predict.tsvm_multiclass
```

Predict from a Multiclass Twin SVM

Description

Predicts from a one-vs-one multiclass twin SVM. Each binary model votes for one class. Ties are resolved deterministically by choosing the class that appears first in the training factor level order.

Usage

```
## S3 method for class 'tsvm_multiclass'
predict(object, newdata, type = c("class", "votes"), ...)
```

Arguments

object	A fitted tsvm_multiclass object.
newdata	Numeric matrix or data frame.
type	Output type. "class" returns predicted class labels; "votes" returns the vote matrix.
...	Unused. decision.values is not supported for multiclass objects because OVO decision values do not have a single unambiguous scale.

Value

A factor of predicted classes, or an integer vote matrix when type = "votes".

See Also

Other multiclass: [confusion\(\)](#), [predict.svms_multiclass\(\)](#), [print.svms_multiclass\(\)](#), [print.tsvm_multiclass\(\)](#)

Examples

```
set.seed(40)
x <- rbind(
  matrix(rnorm(20, -2, 0.2), ncol = 2),
  matrix(rnorm(20, 0, 0.2), ncol = 2),
  matrix(rnorm(20, 2, 0.2), ncol = 2)
)
y <- factor(rep(c("a", "b", "c"), each = 10))
fit <- tsvm(x, y, kernel = "linear")
predict(fit, x[1:3, , drop = FALSE])
```

print.svms_multiclass *Print a Multiclass Standard SVM*

Description

Print a Multiclass Standard SVM

Usage

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

Arguments

x	A fitted svms_multiclass object.
...	Unused.

Value

The input object, invisibly.

See Also

Other multiclass: [confusion\(\)](#), [predict.svms_multiclass\(\)](#), [predict.tsvm_multiclass\(\)](#), [print.tsvm_multiclass\(\)](#)

Examples

```
set.seed(46)  
x <- rbind(  
  matrix(rnorm(8, -2, 0.2), ncol = 2),  
  matrix(rnorm(8, 0, 0.2), ncol = 2),  
  matrix(rnorm(8, 2, 0.2), ncol = 2)  
)  
y <- factor(rep(c("a", "b", "c"), each = 4))  
print(svms(x, y, kernel = "linear", max_passes = 2, max_iter = 100))
```

print.tsvm_multiclass *Print a Multiclass Twin SVM*

Description

Print a Multiclass Twin SVM

Usage

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

Arguments

x A fitted tsvm_multiclass object.
... Unused.

Value

The input object, invisibly.

See Also

Other multiclass: [confusion\(\)](#), [predict.svms_multiclass\(\)](#), [predict.tsvm_multiclass\(\)](#), [print.svms_multiclass\(\)](#)

Examples

```
set.seed(41)  
x <- rbind(  
  matrix(rnorm(20, -2, 0.2), ncol = 2),  
  matrix(rnorm(20, 0, 0.2), ncol = 2),  
  matrix(rnorm(20, 2, 0.2), ncol = 2)  
)  
y <- factor(rep(c("a", "b", "c"), each = 10))  
print(tsvm(x, y, kernel = "linear"))
```

svms

Fit a Standard C-SVC Support Vector Machine

Description

Fits a C-SVC support vector machine with a Platt SMO solver. With two classes this uses the validated binary path. With three or more classes, the function fits one binary SVM for each class pair and predicts by majority vote. Multiclass ties are resolved by choosing the class that appears first in the factor level order.

Usage

```
svms(  
  x,  
  y,  
  kernel = c("linear", "rbf", "poly"),  
  cost = 1,  
  gamma = NULL,  
  degree = 3,
```

```

coef0 = 1,
tol = 0.001,
max_passes = 10L,
max_iter = 10000L
)

```

Arguments

x	Numeric matrix or data frame of predictors.
y	Response with at least two classes. In the binary path, level 1 is the negative class and level 2 is the positive class.
kernel	Kernel name: "linear", "rbf", or "poly".
cost	Positive C-SVC cost parameter.
gamma	Kernel scale. Defaults to 1 / ncol(x).
degree	Polynomial degree.
coef0	Polynomial offset.
tol	SMO tolerance.
max_passes	Maximum consecutive passes without alpha changes.
max_iter	Maximum SMO iterations.

Value

A fitted svms object for two classes, or a svms_multiclass object for three or more classes.

Examples

```

set.seed(1)
dat <- gen_moons(40, noise = 0.1)
fit <- svms(dat$x, dat$y, kernel = "linear", cost = 1)
predict(fit, dat$x[1:3, ])

```

tsvm

Fit a Twin Support Vector Machine

Description

Fits a twin support vector machine. With two classes this uses the validated binary twin-SVM path: level 1 of y is class B, level 2 is class A, plane 1 is close to class A, and plane 2 is close to class B. With three or more classes, the function fits one binary twin SVM for each class pair and predicts by majority vote. Multiclass ties are resolved by choosing the class that appears first in the factor level order.

Usage

```

tsvm(
  x,
  y,
  method = c("ls", "twin"),
  kernel = c("linear", "rbf", "poly"),
  c1 = 1,
  c2 = 1,
  gamma = NULL,
  degree = 3,
  coef0 = 1,
  eps = 1e-06
)

```

Arguments

x	Numeric matrix or data frame of predictors.
y	Response with at least two classes.
method	Twin-SVM method. "ls" fits least-squares twin SVM; "twin" fits the original box-constrained dual formulation.
kernel	Kernel name.
c1, c2	Positive regularization parameters.
gamma	Kernel scale. Defaults to 1 / ncol(x).
degree	Polynomial degree.
coef0	Polynomial offset.
eps	Ridge term added to every linear solve.

Value

A fitted tsvm object for two classes, or a tsvm_multiclass object for three or more classes.

References

Jayadeva, Khemchandani, R., and Chandra, S. (2007). Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5), 905-910.

Kumar, M. A. and Gopal, M. (2009). Least squares twin support vector machines for pattern classification. *Expert Systems with Applications*, 36(4), 7535-7543.

Examples

```

set.seed(3)
dat <- gen_moons(50, noise = 0.05)
fit <- tsvm(dat$x, dat$y)
predict(fit, dat$x[1:4, ])

```

`twomoons`*Example Two-Moons Data*

Description

A small two-class two-moons data set for examples and tests.

Usage

```
data(twomoons)
```

Format

A data frame with 120 rows and 3 variables:

x1 First coordinate.

x2 Second coordinate.

y Two-class factor with levels B and A.

Examples

```
data(twomoons)
str(twomoons)
```

Index

- * **datasets**
 - twomoons, 19
- * **multiclass**
 - confusion, 4
 - predict.svms_multiclass, 13
 - predict.tsvm_multiclass, 14
 - print.svms_multiclass, 15
 - print.tsvm_multiclass, 15
- * **visualization**
 - compare_methods, 3
 - kernel_lift, 7
 - lift_plot, 8
 - lift_plotly, 8

coef.tsvm, 3

compare_methods, 3

compare_methods(), 7–9

confusion, 4

confusion(), 13–16

cv_tsvm, 5

gen_circles, 6

gen_moons, 6

kernel_lift, 7

kernel_lift(), 4, 8, 9

lift_plot, 8

lift_plot(), 4, 7–9

lift_plotly, 8

lift_plotly(), 4, 7, 8

morph_boundary, 9

plot.cv_tsvm, 10

plot.svms, 11

plot.tsvm, 11

predict.svms, 12

predict.svms_multiclass, 13

predict.svms_multiclass(), 4, 14–16

predict.tsvm_multiclass, 14

predict.tsvm_multiclass(), 4, 13, 15, 16

print.svms_multiclass, 15

print.svms_multiclass(), 4, 13, 14, 16

print.tsvm_multiclass, 15

print.tsvm_multiclass(), 4, 13–15

svms, 16

tsvm, 17

tsvm(), 5

twinsvm (twinsvm-package), 2

twinsvm-package, 2

twomoons, 19