# Package 'gensvm'

September 11, 2020

**Version** 0.1.5

**Date** 2020-09-04

**Title** A Generalized Multiclass Support Vector Machine

**Maintainer** Gertjan van den Burg <gertjanvandenburg@gmail.com>

**Description** The GenSVM classifier is a generalized multiclass support vector machine (SVM). This classifier aims to find decision boundaries that separate the classes with as wide a margin as possible. In GenSVM, the loss function is very flexible in the way that misclassifications are penalized. This allows the user to tune the classifier to the dataset at hand and potentially obtain higher classification accuracy than alternative multiclass SVMs. Moreover, this flexibility means that GenSVM has a number of other multiclass SVMs as special cases. One of the other advantages of GenSVM is that it is trained in the primal space, allowing the use of warm starts during optimization. This means that for common tasks such as cross validation or repeated model fitting, GenSVM can be trained very quickly. Based on: G.J.J. van den Burg and P.J.F. Groenen (2018) <https://www.jmlr.org/papers/v17/14-526.html>.

**License** GPL (>= 2)

**Depends** R (>= 3.0.0)

**Classification/MSC** 62H30, 68T10

**URL** https://github.com/GjjvdBurg/RGenSVM

https://jmlr.org/papers/v17/14-526.html

**BugReports** https://github.com/GjjvdBurg/RGenSVM

**RoxygenNote** 7.1.0

**NeedsCompilation** yes

**Author** Gertjan van den Burg [aut, cre],
Patrick Groenen [ctb]

**Repository** CRAN

**Date/Publication** 2020-09-11 08:00:03 UTC

# R topics documented:



---

`gensvm-package` *GenSVM: A Generalized Multiclass Support Vector Machine*

---

## Description

The GenSVM classifier is a generalized multiclass support vector machine (SVM). This classifier aims to find decision boundaries that separate the classes with as wide a margin as possible. In GenSVM, the loss functions that measures how misclassifications are counted is very flexible. This allows the user to tune the classifier to the dataset at hand and potentially obtain higher classification accuracy. Moreover, this flexibility means that GenSVM has a number of alternative multiclass SVMs as special cases. One of the other advantages of GenSVM is that it is trained in the primal space, allowing the use of warm starts during optimization. This means that for common tasks such as cross validation or repeated model fitting, GenSVM can be trained very quickly.

## Details

This package provides functions for training the GenSVM model either as a separate model or through a cross-validated parameter grid search. In both cases the GenSVM C library is used for speed. Auxiliary functions for evaluating and using the model are also provided.

## GenSVM functions

The main GenSVM functions are:

`gensvm` Fit a GenSVM model for specific model parameters.

`gensvm.grid` Run a cross-validated grid search for GenSVM.

For the GenSVM and GenSVMGrid models the following two functions are available. When applied to a GenSVMGrid object, the function is applied to the best GenSVM model.

`plot` Plot the low-dimensional *simplex* space where the decision boundaries are fixed (for problems with 3 classes).

`predict` Predict the class labels of new data using the GenSVM model.

Moreover, for the GenSVM and GenSVMGrid models a `coef` function is defined:

`coef.gensvm` Get the coefficients of the fitted GenSVM model.

`coef.gensvm.grid` Get the parameter grid of the GenSVM grid search.

The following utility functions are also included:

`gensvm.accuracy` Compute the accuracy score between true and predicted class labels

`gensvm.maxabs.scale` Scale each column of the dataset by its maximum absolute value, preserving sparsity and mapping the data to [-1, 1]

`gensvm.train.test.split` Split a dataset into a training and testing sample

`gensvm.refit` Refit a fitted GenSVM model with slightly different parameters or on a different dataset

## Kernels in GenSVM

GenSVM can be used for both linear and nonlinear multiclass support vector machine classification. In general, linear classification will be faster but depending on the dataset higher classification performance can be achieved using a nonlinear kernel.

The following nonlinear kernels are implemented in the GenSVM package:

**RBF** The Radial Basis Function kernel is a well-known kernel function based on the Euclidean distance between objects. It is defined as

$$k(x_i, x_j) = exp(-\gamma||x_i - x_j||^2)$$

**Polynomial** A polynomial kernel can also be used in GenSVM. This kernel function is implemented very generally and therefore takes three parameters (`coef`, `gamma`, and `degree`). It is defined as:

$$k(x_i, x_j) = (\gamma x_i' x_j + coef)^{degree}$$

**Sigmoid** The sigmoid kernel is the final kernel implemented in GenSVM. This kernel has two parameters and is implemented as follows:

$$k(x_i, x_j) = \tanh(\gamma x_i' x_j + coef)$$

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, gensvm.grid

---

`coef.gensvm` *Get the coefficients of the fitted GenSVM model*

---

## Description

Returns the model coefficients of the GenSVM object

## Usage

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

## Arguments

| | |
|---|---|
| `object` | a `gensvm` object |
| `...` | further arguments are ignored |

## Value

The coefficients of the GenSVM model. This is a matrix of size $(n_{features} + 1)x(n_{classes} - 1)$. This matrix is used to project the input data to a low dimensional space using the equation: $XW + t$ where $X$ is the input matrix, $t$ is the first row of the matrix returned by this function, and $W$ is the $n_{features}x(n_{classes} - 1)$ matrix formed by the remaining rows.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, plot.gensvm, predict.gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

fit <- gensvm(x, y)
V <- coef(fit)
```

---

coef.gensvm.grid *Get the parameter grid from a GenSVM Grid object*

---

## Description

Returns the parameter grid of a gensvm.grid object.

## Usage

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

## Arguments

| | |
|---|---|
| object | a gensvm.grid object |
| ... | further arguments are ignored |

## Value

The parameter grid of the GenSVMGrid object as a data frame.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm.grid, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

grid <- gensvm.grid(x, y)
pg <- coef(grid)
```

---

`fitted.gensvm` *Show fitted labels for the GenSVM model*

---

## Description

This function shows the fitted class labels of training data using a fitted GenSVM model.

## Usage

```
## S3 method for class 'gensvm'
fitted(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | Fitted `gensvm` object |
| `...` | further arguments are passed to predict |

## Value

a vector of class labels, with the same type as the original class labels.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`plot.gensvm`, `predict.gensvm.grid`, `gensvm`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit GenSVM and compute training set predictions
fit <- gensvm(x, y)
yhat <- fitted(fit)

# compute the accuracy with gensvm.accuracy
gensvm.accuracy(y, yhat)
```

---

`fitted.gensvm.grid` *Fitted labels for the GenSVMGrid class*

---

## Description

Wrapper to get the fitted class labels from the best estimator of the fitted GenSVMGrid model. Only works if refit was enabled.

## Usage

```
## S3 method for class 'gensvm.grid'
fitted(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | A `gensvm.grid` object |
| `...` | further arguments are passed to fitted |

## Value

a vector of class labels, with the same type as the original class labels.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`plot.gensvm`, `predict.gensvm.grid`, `gensvm`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit GenSVM and compute training set predictions
fit <- gensvm(x, y)
yhat <- fitted(fit)

# compute the accuracy with gensvm.accuracy
gensvm.accuracy(y, yhat)
```

---

gensvm *Fit the GenSVM model*

---

## Description

Fits the Generalized Multiclass Support Vector Machine model with the given parameters. See the package documentation (gensvm-package) for more general information about GenSVM.

## Usage

```
gensvm(
  x,
  y,
  p = 1,
  lambda = 1e-08,
  kappa = 0,
  epsilon = 1e-06,
  weights = "unit",
  kernel = "linear",
  gamma = "auto",
  coef = 1,
  degree = 2,
  kernel.eigen.cutoff = 1e-08,
  verbose = FALSE,
  random.seed = NULL,
  max.iter = 1e+08,
  seed.V = NULL
)
```

## Arguments

| | |
|---|---|
| x | data matrix with the predictors.<br><br>Note that for SVMs categorical features should be converted to binary dummy features. This can be done with using the model.matrix function (i.e. model.matrix( ~ var -1)). |

| | |
|---|---|
| y | class labels |
| p | parameter for the L_p norm of the loss function (1.0 <= p <= 2.0) |
| lambda | regularization parameter for the loss function (lambda > 0) |
| kappa | parameter for the hinge function in the loss function (kappa > -1.0) |
| epsilon | Stopping parameter for the optimization algorithm. The optimization will stop if the relative change in the loss function is below this value. |
| weights | type or vector of instance weights to use. Options are 'unit' for unit weights and 'group' for group size correction weights (eq. 4 in the paper). Alternatively, a vector of weights can be provided. |
| kernel | the kernel type to use in the classifier. It must be one of 'linear', 'poly', 'rbf', or 'sigmoid'. See the section "Kernels in GenSVM" in gensvm-package for more info. |
| gamma | kernel parameter for the rbf, polynomial, and sigmoid kernel. If gamma is 'auto', then 1/n_features will be used. |
| coef | parameter for the polynomial and sigmoid kernel. |
| degree | parameter for the polynomial kernel |
| kernel.eigen.cutoff | Cutoff point for the reduced eigendecomposition used with kernel-GenSVM. Eigenvectors for which the ratio between their corresponding eigenvalue and the largest eigenvalue is smaller than this cutoff value will be dropped. |
| verbose | Turn on verbose output and fit progress |
| random.seed | Seed for the random number generator (useful for reproducible output) |
| max.iter | Maximum number of iterations of the optimization algorithm. |
| seed.V | Matrix to warm-start the optimization algorithm. This is typically the output of coef(fit). Note that this function will silently drop seed.V if the dimensions don't match the provided data. |

## Value

A "gensvm" S3 object is returned for which the print, predict, coef, and plot methods are available. It has the following items:

| | |
|---|---|
| call | The call that was used to construct the model. |
| p | The value of the lp norm in the loss function |
| lambda | The regularization parameter used in the model. |
| kappa | The hinge function parameter used. |
| epsilon | The stopping criterion used. |
| weights | The instance weights type used. |
| kernel | The kernel function used. |
| gamma | The value of the gamma parameter of the kernel, if applicable |
| coef | The value of the coef parameter of the kernel, if applicable |
| degree | The degree of the kernel, if applicable |

| | |
|---|---|
| `kernel.eigen.cutoff` | The cutoff value of the reduced eigendecomposition of the kernel matrix. |
| `verbose` | Whether or not the model was fitted with progress output |
| `random.seed` | The random seed used to seed the model. |
| `max.iter` | Maximum number of iterations of the algorithm. |
| `n.objects` | Number of objects in the dataset |
| `n.features` | Number of features in the dataset |
| `n.classes` | Number of classes in the dataset |
| `classes` | Array with the actual class labels |
| `V` | Coefficient matrix |
| `n.iter` | Number of iterations performed in training |
| `n.support` | Number of support vectors in the final model |
| `training.time` | Total training time |

## Note

This function returns partial results when the computation is interrupted by the user.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`coef`, `print`, `predict`, `plot`, `gensvm.grid`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit using the default parameters and show progress
fit <- gensvm(x, y, verbose=TRUE)

# fit with some changed parameters
fit <- gensvm(x, y, lambda=1e-6)

# Early stopping defined through epsilon
fit <- gensvm(x, y, epsilon=1e-3)
```

```
# Early stopping defined through max.iter
fit <- gensvm(x, y, max.iter=1000)

# Nonlinear training
fit <- gensvm(x, y, kernel='rbf', max.iter=1000)
fit <- gensvm(x, y, kernel='poly', degree=2, gamma=1.0, max.iter=1000)

# Setting the random seed and comparing results
fit <- gensvm(x, y, random.seed=123, max.iter=1000)
fit2 <- gensvm(x, y, random.seed=123, max.iter=1000)
all.equal(coef(fit), coef(fit2))
```

---

`gensvm.accuracy` *Compute the accuracy score*

---

## Description

Compute the accuracy score between the true labels and the predicted labels.

## Usage

```
gensvm.accuracy(y.true, y.pred)
```

## Arguments

| | |
|---|---|
| `y.true` | vector of true labels |
| `y.pred` | vector of predicted labels |

## Value

The accuracy as a value in the range [0.0, 1.0]

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`predict.gensvm.grid`, `predict.gensvm`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

fit <- gensvm(x, y)
gensvm.accuracy(predict(fit, x), y)
```

---

`gensvm.generate.cv.idx` *Generate a vector of cross-validation indices*

---

## Description

This function generates a vector of length `n` with values from 0 to `folds-1` to mark train and test splits.

## Usage

```
gensvm.generate.cv.idx(n, folds)
```

## Arguments

| | |
|---|---|
| `n` | the number of instances |
| `folds` | the number of cross validation folds |

## Value

an array of length `n` with values in the range [0, folds-1] indicating the test fold of each instance.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`gensvm.grid`

gensvm.grid *Cross-validated grid search for GenSVM*

## Description

This function performs a cross-validated grid search of the model parameters to find the best hyper-parameter configuration for a given dataset. This function takes advantage of GenSVM's ability to use warm starts to speed up computation. The function uses the GenSVM C library for speed.

## Usage

```
gensvm.grid(
  x,
  y,
  param.grid = "tiny",
  refit = TRUE,
  scoring = NULL,
  cv = 3,
  verbose = 0,
  return.train.score = TRUE
)
```

## Arguments

| | |
|---|---|
| x | training data matrix. We denote the size of this matrix by n_samples x n_features. |
| y | training vector of class labels of length n_samples. The number of unique labels in this vector is denoted by n_classes. |
| param.grid | String (`'tiny'`, `'small'`, or `'full'`) or data frame with parameter configurations to evaluate. Typically this is the output of `expand.grid`. For more details, see "Using a Parameter Grid" below. |
| refit | boolean variable. If true, the best model from cross validation is fitted again on the entire dataset. |
| scoring | metric to use to evaluate the classifier performance during cross validation. The metric should be an R function that takes two arguments: y_true and y_pred and that returns a float such that higher values are better. If it is NULL, the accuracy score will be used. |
| cv | the number of cross-validation folds to use or a vector with the same length as y where each unique value denotes a test split. |
| verbose | integer to indicate the level of verbosity (higher is more verbose) |
| return.train.score | whether or not to return the scores on the training splits |

## Value

A "gensvm.grid" S3 object with the following items:

| | |
|---|---|
| `call` | Call that produced this object |
| `param.grid` | Sorted version of the parameter grid used in training |
| `cv.results` | A data frame with the cross validation results |
| `best.estimator` | If refit=TRUE, this is the GenSVM model fitted with the best hyperparameter configuration, otherwise it is NULL |
| `best.score` | Mean cross-validated test score for the model with the best hyperparameter configuration |
| `best.params` | Parameter configuration that provided the highest mean cross-validated test score |
| `best.index` | Row index of the cv.results data frame that corresponds to the best hyperparameter configuration |
| `n.splits` | The number of cross-validation splits |
| `n.objects` | The number of instances in the data |
| `n.features` | The number of features of the data |
| `n.classes` | The number of classes in the data |
| `classes` | Array with the unique classes in the data |
| `total.time` | Training time for the grid search |
| `cv.idx` | Array with cross validation indices used to split the data |

## Using a Parameter Grid

To evaluate certain parameter configurations, a data frame can be supplied to the `param.grid` argument of the function. Such a data frame can easily be generated using the R function `expand.grid`, or could be created through other ways to test specific parameter configurations.

Three parameter grids are predefined:

`'tiny'` This parameter grid is generated by the function `gensvm.load.tiny.grid` and is the default parameter grid. It consists of parameter configurations that are likely to perform well on various datasets.

`'small'` This grid is generated by `gensvm.load.small.grid` and generates a data frame with 90 configurations. It is typically fast to train but contains some configurations that are unlikely to perform well. It is included for educational purposes.

`'full'` This grid loads the parameter grid as used in the GenSVM paper. It consists of 342 configurations and is generated by the `gensvm.load.full.grid` function. Note that in the GenSVM paper cross validation was done with this parameter grid, but the final training step used `epsilon=1e-8`. The `gensvm.refit` function is useful in this scenario.

When you provide your own parameter grid, beware that only certain column names are allowed in the data frame corresponding to parameters for the GenSVM model. These names are:

**p** Parameter for the lp norm. Must be in [1.0, 2.0].

**kappa** Parameter for the Huber hinge function. Must be larger than -1.

**lambda** Parameter for the regularization term. Must be larger than 0.

**weights** Instance weights specification. Allowed values are "unit" for unit weights and "group" for group-size correction weights

**epsilon** Stopping parameter for the algorithm. Must be larger than 0.

**max.iter** Maximum number of iterations of the algorithm. Must be larger than 0.

**kernel** The kernel to used, allowed values are "linear", "poly", "rbf", and "sigmoid". The default is "linear"

**coef** Parameter for the "poly" and "sigmoid" kernels. See the section "Kernels in GenSVM" in the codeinkgensvm-package page for more info.

**degree** Parameter for the "poly" kernel. See the section "Kernels in GenSVM" in the codeinkgensvm-package page for more info.

**gamma** Parameter for the "poly", "rbf", and "sigmoid" kernels. See the section "Kernels in GenSVM" in the codeinkgensvm-package page for more info.

For variables that are not present in the `param.grid` data frame the default parameter values in the `gensvm` function will be used.

Note that this function reorders the parameter grid to make the warm starts as efficient as possible, which is why the param.grid in the result will not be the same as the param.grid in the input.

## Note

1. This function returns partial results when the computation is interrupted by the user. 2. The score.time reported in the results only covers the time needed to compute the score from the predictions and true class labels. It does not include the time to compute the predictions themselves.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`predict.gensvm.grid`, `print.gensvm.grid`, `plot.gensvm.grid`, `gensvm`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]


# use the default parameter grid
grid <- gensvm.grid(x, y, verbose=TRUE)
```

```
# use a smaller parameter grid
pg <- expand.grid(p=c(1.0, 1.5, 2.0), kappa=c(-0.9, 1.0), epsilon=c(1e-3))
grid <- gensvm.grid(x, y, param.grid=pg)

# print the result
print(grid)


# Using a custom scoring function (accuracy as percentage)
acc.pct <- function(yt, yp) { return (100 * sum(yt == yp) / length(yt)) }
grid <- gensvm.grid(x, y, scoring=acc.pct)

# With RBF kernel and very verbose progress printing
pg <- expand.grid(kernel=c('rbf'), gamma=c(1e-2, 1e-1, 1, 1e1, 1e2),
                  lambda=c(1e-8, 1e-6), max.iter=c(5000))
grid <- gensvm.grid(x, y, param.grid=pg, verbose=2)
```

---

`gensvm.load.full.grid` *Load a large parameter grid for the GenSVM grid search*

---

## Description

This loads the parameter grid from the GenSVM paper. It consists of 342 configurations and is constructed from all possible combinations of the following parameter sets:

`p = c(1.0,1.5,2.0)` `lambda = 2^seq(-18,18,2)` `kappa = c(-0.9,0.5,5.0)` `weights = c('unit','group')`

## Usage

```
gensvm.load.full.grid()
```

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`gensvm.grid`, `gensvm.load.tiny.grid`, `gensvm.load.full.grid`.

## gensvm.load.small.grid *Load the small parameter grid for the GenSVM grid search*

### Description

This function loads a small parameter grid to use for the GenSVM gridsearch. It contains all possible combinations of the following parameter sets:

`p = c(1.0,1.5,2.0)` `lambda = c(1e-8,1e-6,1e-4,1e-2,1)` `kappa = c(-0.9,0.5,5.0)` `weights= c('unit','group')`

### Usage

```
gensvm.load.small.grid()
```

### Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

### References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

### See Also

gensvm.grid, gensvm.load.tiny.grid, gensvm.load.small.grid.

## gensvm.load.tiny.grid *Load a tiny parameter grid for the GenSVM grid search*

### Description

This function returns a parameter grid to use in the GenSVM grid search. This grid was obtained by analyzing the experiments done for the GenSVM paper and selecting the configurations that achieve accuracy within the 95th percentile on over 90 for a parameter search with a reasonably high chance of achieving good performance on most datasets.

Note that this grid is only tested to work well in combination with the linear kernel.

### Usage

```
gensvm.load.tiny.grid()
```

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm.grid, gensvm.load.small.grid, gensvm.load.full.grid.

---

# gensvm.maxabs.scale *Scale each column of a matrix by its maximum absolute value*

---

## Description

Scaling a dataset can greatly decrease the computation time of GenSVM. This function scales the data by dividing each column of a matrix by the maximum absolute value of that column. This preserves sparsity in the data while mapping each column to the interval [-1, 1].

Optionally a test dataset can be provided as well. In this case, the scaling will be computed on the first argument (x) and applied to the test dataset. Note that the return value is a list when this argument is supplied.

## Usage

```
gensvm.maxabs.scale(x, x.test = NULL)
```

## Arguments

| | |
|---|---|
| x | a matrix to scale |
| x.test | (optional) a test matrix to scale as well. |

## Value

if x.test=NULL a scaled matrix where the maximum value of the columns is 1 and the minimum value of the columns isn't below -1. If x.test is supplied, a list with elements x and x.test representing the scaled datasets.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, gensvm.grid, gensvm.train.test.split, gensvm-package

## Examples

```
x <- iris[, -5]

# check the min and max of the columns
apply(x, 2, min)
apply(x, 2, max)

# scale the data
x.scale <- gensvm.maxabs.scale(x)

# check again (max should be 1.0, min shouldn't be below -1)
apply(x.scale, 2, min)
apply(x.scale, 2, max)

# with a train and test dataset
split <- gensvm.train.test.split(x)
x.train <- split$x.train
x.test <- split$x.test
scaled <- gensvm.maxabs.scale(x.train, x.test)
x.train.scl <- scaled$x
x.test.scl <- scaled$x.test
```

---

# gensvm.rank.score *Compute the ranks for the numbers in a given vector*

---

## Description

This function computes the ranks for the values in an array. The highest value gets the smallest rank. Ties are broken by assigning the smallest value. The smallest rank is 1.

## Usage

```
gensvm.rank.score(x)
```

## Arguments

| | |
|---|---|
| x | array of numeric values |

## Value

array with the ranks of the values in the input array.

---

`gensvm.refit` *Train an already fitted model on new data*

---

## Description

This function can be used to train an existing model on new data or fit an existing model with slightly different parameters. It is useful for retraining without having to copy all the parameters over. One common application for this is to refit the best model found by a grid search, as illustrated in the examples.

## Usage

```
gensvm.refit(
  fit,
  x,
  y,
  p = NULL,
  lambda = NULL,
  kappa = NULL,
  epsilon = NULL,
  weights = NULL,
  kernel = NULL,
  gamma = NULL,
  coef = NULL,
  degree = NULL,
  kernel.eigen.cutoff = NULL,
  max.iter = NULL,
  verbose = NULL,
  random.seed = NULL
)
```

## Arguments

| | |
|---|---|
| `fit` | Fitted `gensvm` object |
| `x` | Data matrix of the new data |
| `y` | Label vector of the new data |
| `p` | if NULL use the value from `fit` in the new model, otherwise override with this value. |
| `lambda` | if NULL use the value from `fit` in the new model, otherwise override with this value. |
| `kappa` | if NULL use the value from `fit` in the new model, otherwise override with this value. |

`epsilon` if NULL use the value from `fit` in the new model, otherwise override with this value.

`weights` if NULL use the value from `fit` in the new model, otherwise override with this value.

`kernel` if NULL use the value from `fit` in the new model, otherwise override with this value.

`gamma` if NULL use the value from `fit` in the new model, otherwise override with this value.

`coef` if NULL use the value from `fit` in the new model, otherwise override with this value.

`degree` if NULL use the value from `fit` in the new model, otherwise override with this value.

`kernel.eigen.cutoff` if NULL use the value from `fit` in the new model, otherwise override with this value.

`max.iter` if NULL use the value from `fit` in the new model, otherwise override with this value.

`verbose` if NULL use the value from `fit` in the new model, otherwise override with this value.

`random.seed` if NULL use the value from `fit` in the new model, otherwise override with this value.

## Value

a new fitted `gensvm` model

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit a standard model and refit with slightly different parameters
```

```
fit <- gensvm(x, y)
fit2 <- gensvm.refit(fit, x, y, epsilon=1e-8)


# refit a model returned by a grid search
grid <- gensvm.grid(x, y)
fit <- gensvm.refit(fit, x, y, epsilon=1e-8)


# refit on different data
idx <- runif(nrow(x)) > 0.5
x1 <- x[idx, ]
x2 <- x[!idx, ]
y1 <- y[idx]
y2 <- y[!idx]

fit1 <- gensvm(x1, y1)
fit2 <- gensvm.refit(fit1, x2, y2)
```

---

`gensvm.train.test.split` *Create a train/test split of a dataset*

---

## Description

Often it is desirable to split a dataset into a training and testing sample. This function is included in GenSVM to make it easy to do so. The function is inspired by a similar function in Scikit-Learn.

## Usage

```
gensvm.train.test.split(
  x,
  y = NULL,
  train.size = NULL,
  test.size = NULL,
  shuffle = TRUE,
  random.state = NULL,
  return.idx = FALSE
)
```

## Arguments

| | |
|---|---|
| `x` | array to split |
| `y` | another array to split (typically this is a vector) |
| `train.size` | size of the training dataset. This can be provided as float or as int. If it's a float, it should be between 0.0 and 1.0 and represents the fraction of the dataset that should be placed in the training dataset. If it's an int, it represents the exact |

| | |
|---|---|
| | number of samples in the training dataset. If it is NULL, the complement of `test.size` will be used. |
| `test.size` | size of the test dataset. Similarly to train.size both a float or an int can be supplied. If it's NULL, the complement of train.size will be used. If both train.size and test.size are NULL, a default test.size of 0.25 will be used. |
| `shuffle` | shuffle the rows or not |
| `random.state` | seed for the random number generator (int) |
| `return.idx` | whether or not to return the indices in the output |

## Value

a list with `x.train` and `x.test` splits of the `x` array provided. If `y` is provided, also `y.train` and `y.test`. If `return.idx` is TRUE, also `idx.train` and `idx.test`.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# using the default values
split <- gensvm.train.test.split(x, y)

# using the split in a GenSVM model
fit <- gensvm(split$x.train, split$y.train)
gensvm.accuracy(split$y.test, predict(fit, split$x.test))

# using attach makes the results directly available
attach(gensvm.train.test.split(x, y))
fit <- gensvm(x.train, y.train)
gensvm.accuracy(y.test, predict(fit, x.test))
```

`plot.gensvm` *Plot the simplex space of the fitted GenSVM model*

## Description

This function creates a plot of the simplex space for a fitted GenSVM model and the given data set. This function works for dataset with two or three classes. For more than 3 classes, the simplex space is too high dimensional to easily visualize.

## Usage

```
## S3 method for class 'gensvm'
plot(
  x,
  labels,
  newdata = NULL,
  with.margins = TRUE,
  with.shading = TRUE,
  with.legend = TRUE,
  center.plot = TRUE,
  xlim = NULL,
  ylim = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | A fitted `gensvm` object |
| `labels` | the labels to color points with. If this is omitted the predicted labels are used. |
| `newdata` | the dataset to plot. If this is NULL the training data is used. |
| `with.margins` | plot the margins |
| `with.shading` | show shaded areas for the class regions |
| `with.legend` | show the legend for the class labels |
| `center.plot` | ensure that the boundaries and margins are always visible in the plot |
| `xlim` | allows the user to force certain plot limits. If set, these bounds will be used for the horizontal axis. |
| `ylim` | allows the user to force certain plot limits. If set, these bounds will be used for the vertical axis and the value of center.plot will be ignored |
| `...` | further arguments are passed to the builtin plot() function |

## Value

returns the object passed as input

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

plot.gensvm.grid, predict.gensvm, gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# train the model
fit <- gensvm(x, y)

# plot the simplex space
plot(fit)

# plot and use the true colors (easier to spot misclassified samples)
plot(fit, y)

# plot only misclassified samples
x.mis <- x[predict(fit) != y, ]
y.mis.true <- y[predict(fit) != y]
plot(fit, newdata=x.mis)
plot(fit, y.mis.true, newdata=x.mis)

# plot a 2-d model
xx <- x[y %in% c('versicolor', 'virginica'), ]
yy <- y[y %in% c('versicolor', 'virginica')]
fit <- gensvm(xx, yy, kernel='rbf', max.iter=1000)
plot(fit)
```

---

# plot.gensvm.grid *Plot the simplex space of the best fitted model in the GenSVMGrid*

---

## Description

This is a wrapper which calls the plot function for the best model in the provided GenSVMGrid object. See the documentation for plot.gensvm for more information.

### Usage

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

### Arguments

| | |
|---|---|
| x | A gensvm.grid object trained with refit=TRUE |
| ... | further arguments are passed to the plot function |

### Value

returns the object passed as input

### Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

### References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

### See Also

plot.gensvm, gensvm.grid, predict.gensvm.grid, gensvm-package

### Examples

```
x <- iris[, -5]
y <- iris[, 5]

grid <- gensvm.grid(x, y)
plot(grid, x)
```

---

## predict.gensvm *Predict class labels with the GenSVM model*

---

### Description

This function predicts the class labels of new data using a fitted GenSVM model.

## Usage

```
## S3 method for class 'gensvm'
predict(object, newdata, add.rownames = FALSE, ...)
```

## Arguments

| | |
|---|---|
| object | Fitted gensvm object |
| newdata | Matrix of new data for which predictions need to be made. |
| add.rownames | add the rownames from the training data to the predictions |
| ... | further arguments are ignored |

## Value

a vector of class labels, with the same type as the original class labels.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

plot.gensvm, predict.gensvm.grid, gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# create a training and test sample
attach(gensvm.train.test.split(x, y))
fit <- gensvm(x.train, y.train)

# predict the class labels of the test sample
y.test.pred <- predict(fit, x.test)

# compute the accuracy with gensvm.accuracy
gensvm.accuracy(y.test, y.test.pred)
```

predict.gensvm.grid *Predict class labels from the GenSVMGrid class*

## Description

Predict class labels using the best model from a grid search. After doing a grid search with the gensvm.grid function, this function can be used to make predictions of class labels. It uses the best GenSVM model found during the grid search to do the predictions. Note that this model is only available if refit=TRUE was specified in the gensvm.grid call (the default).

## Usage

```
## S3 method for class 'gensvm.grid'
predict(object, newdata, ...)
```

## Arguments

| | |
|---|---|
| object | A gensvm.grid object trained with refit=TRUE |
| newdata | Matrix of new values for x for which predictions need to be computed. |
| ... | further arguments are passed to predict.gensvm() |

## Value

a vector of class labels, with the same type as the original class labels provided to gensvm.grid()

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

gensvm, predict.gensvm.grid, plot.gensvm, gensvm-package

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# run a grid search
grid <- gensvm.grid(x, y)
```

```
# predict training sample
y.hat <- predict(grid, x)
```

---

`print.gensvm` *Print the fitted GenSVM model*

---

## Description

Prints a short description of the fitted GenSVM model

## Usage

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

## Arguments

| | |
|---|---|
| `x` | A `gensvm` object to print |
| `...` | further arguments are ignored |

## Value

returns the object passed as input. This can be useful for chaining operations on a fit object.

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`gensvm`, `predict.gensvm`, `plot.gensvm`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit and print the model
fit <- gensvm(x, y)
print(fit)

# (advanced) use the fact that print returns the fitted model
fit <- gensvm(x, y)
predict(print(fit), x)
```

---

`print.gensvm.grid` *Print the fitted GenSVMGrid model*

---

## Description

Prints the summary of the fitted GenSVMGrid model

## Usage

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

## Arguments

| | |
|---|---|
| `x` | a `gensvm.grid` object to print |
| `...` | further arguments are ignored |

## Value

returns the object passed as input

## Author(s)

Gerrit J.J. van den Burg, Patrick J.F. Groenen
Maintainer: Gerrit J.J. van den Burg <gertjanvandenburg@gmail.com>

## References

Van den Burg, G.J.J. and Groenen, P.J.F. (2016). *GenSVM: A Generalized Multiclass Support Vector Machine*, Journal of Machine Learning Research, 17(225):1–42. URL https://jmlr.org/papers/v17/14-526.html.

## See Also

`gensvm.grid`, `predict.gensvm.grid`, `plot.gensvm.grid`, `gensvm.grid`, `gensvm-package`

## Examples

```
x <- iris[, -5]
y <- iris[, 5]

# fit a grid search and print the resulting object
grid <- gensvm.grid(x, y)
print(grid)
```

# Index