SVM Wrapper

October 19, 2014

`plot.svm` *plot*

Description

Draws choosen dimenstions from a dataset on 2D plot. NOTE: This function will be change to a package default function.

Usage

```
plot(svm)
```

Format

NULL

`dataset.X` *dataset.X*

Description

Prints dataset stored in a SVM object, without the labels.

Usage

```
dataset.X(svm)
```

Arguments

`object`	SVM object.

Format

NULL

dataset.Y *dataset.Y*

Description

Prints lables stored in a SVM object.

Usage

```
dataset.Y(svm)
```

Arguments

object	SVM object.

Format

NULL

rcpp_hello_world *Simple function using Rcpp*

Description

Simple function using Rcpp

Usage

```
rcpp_hello_world()
```

Examples

```
## Not run:
rcpp_hello_world()

## End(Not run)
```

load.dataset *load.dataset*

Description

Loads provided dataset to a SVM object, right now lables need to be in the last column.

Usage

```
load.dataset(svm, x)
```

Arguments

object	SVM object to which examples will be loaded.
x	Labeled dataset with lables in the last column.

Format

NULL

params.svm *params*

Description

Changes SVM objects parameters to provided, not every parameter needs to be given.

Usage

```
params(svm, object, lib, kernel, preprocess, C, gamma, coef0, degree, cache_size, shrinking, probabil
```

Arguments

lib	Desired SVM Library, avialable are: libsvm.
kernel	Kernel type, avialable are: linear, poly, rbf, sigmoid.
prep	Preprocess method, avialable are: none, 2e.
C	Cost/Complexity parameter.
gamma	Gamma parameter for poly, rbf and sigmoid kernels.
coef0	Coef0 for poly and sigmoid kernels.
degree	Degree for poly kernel.
shrinking	Whether to use shrinking heuristics.
probability	Whether to train a model for probability estimates .
cache_size	Cache size.

Format

NULL

Examples

```
params(svm, kernel="linear", preproces="none", C=10000)
```

---

params.svm.cache_size *params.cache_size*

Description

Changes or prints (if a argument is not provided) cache_size (in MB) in a SVM object, note: requires retraining.

Usage

```
params.cache_size(svm)
params.cache_size(svm,x)
```

Arguments

object	SVM object.
x	New cache size.

Format

NULL

Examples

```
params.cache_size(svm, 100)
```

---

params.svm.C *params.C*

Description

Changes or prints (if a argument is not provided) Cost parameter in a SVM object, note: requires retraining. Aviable are: none, 2e

Usage

```
params.C(svm)
params.C(svm,x)
```

Arguments

object	SVM object.
x	New cost.

Format

NULL

Examples

```
params.C(svm, 1000)
```

---

params.svm.coef0 *params.coef0*

Description

Changes or prints (if a argument is not provided) coef0 parameter in a SVM object, note: requires retraining.

Usage

```
params.coef0(svm)
params.coef0(svm,x)
```

Arguments

object	SVM object.
x	coef0.

Format

NULL

Examples

```
params.coef0(svm, 0)
```

params.svm.degree *params.degree*

Description

Changes or prints (if a argument is not provided) degree in a SVM object, note: requires retraining.

Usage

```
params.degree(svm)
params.degree(svm,x)
```

Arguments

object	SVM object.
x	New degree.

Format

NULL

Examples

```
params.degree(svm, 4)
```

params.svm.gamma *params.gamma*

Description

Changes or prints (if a argument is not provided) gamma parameter in a SVM object, note: requires retraining.

Usage

```
params.gamma(svm)
params.gamma(svm,x)
```

Arguments

object	SVM object.
x	New gamma.

Format

NULL

Examples

```
params.gamma(svm, 0.01)
```

`params.svm.kernel` *params.kernel*

Description

Changes or prints (if a argument is not provided)kernel in a SVM object, note: requires retraining. Aviable are: libsvm.

Usage

```
params.kernel(svm)
params.kernel(svm,x)
```

Arguments

`object`	SVM object.
`x`	New kernel.

Format

`NULL`

Examples

```
params.kernel(svm, "rbf")
```

`params.svm.lib` *params.lib*

Description

Changes or prints (if a argument is not provided) library in a SVM object, note: requires retraining.

Usage

```
params.lib(svm)
params.lib(svm,x)
```

Arguments

`object`	SVM object.
`x`	New library.

Format

NULL

Examples

```
params.lib(svm, "libsvm")
```

---

params.svm.prep *params.prep*

Description

Changes or prints (if a argument is not provided) preprocess type in a SVM object, note: requires retraining. Aviable are: linear, sigmoid, poly, rbf.

Usage

```
params.prep(svm)
params.prep(svm,x)
```

Arguments

object	SVM object.
x	New preprocess type.

Format

NULL

Examples

```
params.prep(svm, "2e")
```

---

params.svm.probability *params.probability*

Description

Changes or prints (if a argument is not provided) if a SVM object is used for probability estimates , note: requires retraining.

Usage

```
params.probability(svm)
params.probability(svm,x)
```

Arguments

object	SVM object.
x	TRUE/FALSE.

Format

NULL

Examples

```
params.probability(svm, TRUE)
```

---

params.svm.shrinking *params.shrinking*

Description

Changes or prints (if a argument is not provided) if shrinking heurestics are used in a SVM object, note: requires retraining.

Usage

```
params.shrinking(svm)
params.shrinking(svm,x)
```

Arguments

object	SVM object.
x	TRUE/FALSE.

Format

NULL

Examples

```
params.shrinking(svm, TRUE)
```

predict.svm *Predict*

Description

Returns predicted classes for provided test examples, note that provided model needs to be trained first before any prediction can be made.

Usage

```
predict(svm, x)
```

Arguments

object	Trained SVM object.
x	unlabeled data, note that each entry needs to be the same dimentionality as training examples.

Format

NULL

print.svm *print*

Description

Prints short summary of the SVM object and its parameters.

Usage

```
print(svm)
```

Arguments

object	SVM object

Format

NULL

SVM *SVM*

Description

Create SVM model object. If any parameter will be omitted a default value will be used

Usage

```
SVM
```

Arguments

x	Dataset without labels
y	Labels
lib	Desired SVM Library, avialable are: libsvm
kernel	Kernel type, avialable are: linear, poly, rbf, sigmoid
prep	Preprocess method, avialable are: none, 2e
mclass	Multiclass wariant, avialable are: none
C	Cost/Complexity parameter
gamma	Gamma parameter for poly, rbf and sigmoid kernels
coef0	Coef0 for poly and sigmoid kernels
degree	Degree for poly kernel
shrinking	Whether to use shrinking heuristics
probability	Whether to train a model for probability estimates
cache_size	Cache size
tol	Tolerance of termination criterion

Format

```
NULL
```

Value

SVM model object

Examples

```
svm <- SVM(lib = "libsvm", kernel = "linear", C = 1, gamma = 0.01, coef0 = 0,degree = 3)
```

`train.svm` *Train SVM model*

Description

Trains provided svm object, using its parameters and dataset.

Usage

```
train(svm)
```

Arguments

`object`	Trained SVM object which will be used for prediction.

Format

`NULL`

Index