Package ‘classifly’

October 12, 2022

Title Explore Classification Models in High Dimensions
Version 0.4.1
Author Hadley Wickham <h.wickham@gmail.com>
Maintainer Hadley Wickham <h.wickham@gmail.com>
Description Given $p$-dimensional training data containing $d$ groups (the design space), a classification algorithm (classifier) predicts which group new data belongs to. Generally the input to these algorithms is high dimensional, and the boundaries between groups will be high dimensional and perhaps curvilinear or multi-faceted. This package implements methods for understanding the division of space between the groups.
License MIT + file LICENSE
URL http://had.co.nz/classifly
Imports class, plyr, stats
Suggests e1071, MASS, rpart
Encoding UTF-8
LazyData true
RoxygenNote 7.2.0
NeedsCompilation no
Repository CRAN
Date/Publication 2022-05-20 06:10:02 UTC

R topics documented:

advantage ................................................................. 2
classifly ............................................................... 2
extend ................................................................. 4
generate_classification_data ........................................ 5
generate_data ....................................................... 6
knnf ................................................................. 6
olives ............................................................... 7
Description

This is used to identify the boundaries between classification regions. Points with low (close to 0) advantage are likely to be near boundaries.

Usage

```r
advantage(post)
```

Arguments

- `post`: matrix of posterior probabilities

---

classify

Classify provides a convenient method to fit a classification function and then explore the results in the original high dimensional space.

Description

This is a convenient function to fit a classification function and then explore the results using GGobi. You can also do this in two separate steps using the classification function and then `explore`.

Usage

```r
classify(
    data,
    model,
    classifier,
    ...
    n = 10000,
    method = "nonaligned",
    type = "range"
)
```
Arguments

data  Data set use for classification
model Classification formula, usually of the form response ~ predictors
classifier Function to use for the classification, eg. lda
... Other arguments passed to classification function. For example, if you use svm you need to use probability = TRUE so that posterior probabilities can be retrieved.
n  Number of points to simulate. To maintain the illusion of a filled solid this needs to increase with dimension. 10,000 points seems adequate for up to four or five dimensions, but if you have more predictors than that, you will need to increase this number.
method method to simulate points: grid, random or nonaligned (default). See simvar for more details on the methods used.
type type of scaling to apply to data. Defaults to common range. See rescaler for more details.

Details

By default in GGobi, points that are not on the boundary (ie. that have an advantage greater than the 5 to brush mode and choose include shadowed points from the brush menu on the plot window. You can then brush them yourself to explore how the certainty of classification varies throughout the space

Special notes:

- You should make sure the response variable is a factor
- For SVM, make sure to include probability = TRUE in the arguments to classifly

See Also

explore, http://had.co.nz/classifly

Examples

data(kyphosis, package = "rpart")
library(MASS)
classify(kyphosis, Kyphosis ~ . , lda)
classify(kyphosis, Kyphosis ~ . , qda)
classify(kyphosis, Kyphosis ~ . , glm, family="binomial")
classify(kyphosis, Kyphosis ~ . , knnf, k=3)

library(rpart)
classify(kyphosis, Kyphosis ~ . , rpart)

if (require("e1071")) {
classify(kyphosis, Kyphosis ~ . , svm, probability=TRUE)
classify(kyphosis, Kyphosis ~ . , svm, probability=TRUE, kernel="linear")}
explore(kyphosis, Kyphosis ~ ., best.svm, probability=TRUE, 
   kernel="linear")

# Also can use explore directly
bsvm <- best.svm(Species~., data = iris, gamma = 2^(-1:1), 
   cost = 2^(2:+ 4), probability=TRUE)
explot(bsvm, iris)
}

explore

### Default method for exploring objects

**Description**

The default method currently works for classification functions.

**Usage**

```r
explore(model, data, n = 10000, method = "nonaligned", advantage = TRUE, ...)
```

**Arguments**

- `model` : classification object
- `data` : data set used with classifier
- `n` : number of points to generate when searching for boundaries
- `method` : method to generate points, see `generate_data`
- `advantage` : only display boundaries
- `...` : other arguments not currently used

**Details**

It generates a data set filling the design space, finds class boundaries (if desired) and then displays in a new ggobi instance.

**Value**

A `invisible` data frame of class `classifly` that contains all the simulated and true data. This can be saved and then printed later to open with rggobi.

**See Also**

`generate_classification_data`, `http://had.co.nz/classifly`
generate_classification_data

Examples

```r
if (require("e1071")) {
  bsvm <- best.svm(Species~., data = iris, gamma = 2^(-1:1),
                  cost = 2^2:4, probability=TRUE)
  explore(bsvm, iris)
}
```

generate_classification_data

*Generate classification data.*

Description

Given a model, this function generates points within the range of the data, classifies them, and attempts to locate boundaries by looking at advantage.

Usage

```r
generate_classification_data(model, data, n, method, advantage)
```

Arguments

- `model`: classification model
- `data`: data set used in model
- `n`: number of points to generate
- `method`: method to use, currently either grid (an evenly spaced grid), random (uniform random distribution across cube), or nonaligned (grid + some random perturbation)
- `advantage`: if TRUE, compute advantage, otherwise don’t

Details

If posterior probabilities of classification are available, then the `advantage` will be calculated directly. If not, knn is used calculate the advantage based on the number of neighbouring points that share the same classification. Because knn is $O(n^2)$ this method is rather slow for large (>20,000 say) data sets.

By default, the boundary points are identified as those below the 5th-percentile for advantage.

Value

data.frame of classified data
### generate_data

*Generate new data from a data frame.*

**Description**

This method generates new data that fills the range of the supplied datasets.

**Usage**

```r
generate_data(data, n = 10000, method = "grid")
```

**Arguments**

- `data`: data frame
- `n`: desired number of new observations
- `method`: method to use, see `simvar`

### knnf

*A wrapper function for `knn` to allow use with classify.*

**Description**

A wrapper function for `knn` to allow use with classify.

**Usage**

```r
knnf(formula, data, k = 2)
```

**Arguments**

- `formula`: classification formula
- `data`: training data set
- `k`: number of neighbours to use
Description

The olive oil data consists of the percentage composition of 8 fatty acids (palmitic, palmitoleic, stearic, oleic, linoleic, linolenic, arachidic, eicosenoic) found in the lipid fraction of 572 Italian olive oils. There are 9 collection areas, 4 from southern Italy (North and South Apulia, Calabria, Sicily), two from Sardinia (Inland and Coastal) and 3 from northern Italy (Umbria, East and West Liguria).

Format

A data frame with 244 rows and 7 variables

References


posterior

Extract posterior group probabilities

Description

Every classification method seems to provide a slightly different way of retrieving the posterior probability of group membership. This function provides a common interface to all of them

Usage

posterior(model, data)

Arguments

model model object

data data set used in model
**simvar**

*Simulate observations from a vector*

**Description**

Given a vector of data this function will simulate data that could have come from that vector.

**Usage**

```r
simvar(x, n = 10, method = "grid")
```

**Arguments**

- `x`: data vector
- `n`: desired number of points (will not always be achieved)

**Details**

There are three methods to choose from:

- nonaligned (default): grid + some random perturbation
- grid: grid of evenly spaced observations. If a factor, all levels in a factor will be used, regardless of `n`
- random: a random uniform sample from the range of the variable

**variables**

*Extract predictor and response variables for a model object.*

**Description**

Due to the way that most model objects are stored, you also need to supply the data set you used with the original data set. It currently doesn’t support models fitted without using a data argument.

**Usage**

```r
variables(model)
```

**Arguments**

- `model`: model object

**Value**

list containing response and predictor variables
Index

* attribute
  variables, 8
* classif
  advantage, 2
  knnf, 6
* datagen
  generate_classification_data, 5
  generate_data, 6
  simvar, 8
* dynamic
  classifly, 2
  advantage, 2, 5
  classifly, 2
  explore, 2, 3, 4
  generate_classification_data, 4, 5
  generate_data, 4, 6
  invisible, 4
  knn, 5, 6
  knnf, 6
  lda, 3
  olives, 7
  package-classifly(classifly), 2
  posterior, 7
  rescaler, 3
  simvar, 3, 6, 8
  svm, 3
  variables, 8