Package: kknn (via r-universe)

January 20, 2025

Title Weighted k-Nearest Neighbors

Version 1.3.3

Description Weighted k-Nearest Neighbors for Classification, Regression and Clustering.

Depends R (>= 2.10)

Imports igraph (>= 1.0), Matrix, stats, graphics

Suggests tinytest

ByteCompile TRUE

License GPL (>= 2)

NeedsCompilation yes

URL https://github.com/KlausVigo/kknn

BugReports https://github.com/KlausVigo/kknn/issues

Config/pak/sysreqs libglpk-dev libxml2-dev

Repository https://klausvigo.r-universe.dev

RemoteUrl https://github.com/klausvigo/kknn

RemoteRef HEAD

RemoteSha 3becbeb277627ccb4d9e1dc96fb9c3a4d4af5933

Contents

kknn-package *Weighted k-Nearest Neighbors Classification and Clustering*

Description

Weighted k-Nearest Neighbors Classification, Regression and spectral Clustering

The complete list of functions can be displayed with `library(help = kknn)`.

Author(s)

Klaus Schliep
Klaus Hechenbichler

Maintainer: Klaus Schliep <klaus.schliep@gmail.com>

References

Hechenbichler K. and Schliep K.P. (2004) *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*, Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich (https://doi.org/10.5282/ubm/epub.1769)

Hechenbichler K. (2005) *Ensemble-Techniken und ordinale Klassifikation*, PhD-thesis

contr.dummy *Contrast Matrices*

Description

Returns a matrix of contrasts.

Usage

```
contr.dummy(n, contrasts = TRUE)
contr.ordinal(n, contrasts = TRUE)
contr.metric(n, contrasts = TRUE)
```

Arguments

n	A vector containing levels of a factor, or the number of levels.
contrasts	A logical value indicating whether contrasts should be computed.

Details

`contr.dummy` is standard dummy-coding, `contr.metric` has the same effect like `as.numeric` (makes sense of course only for ordered variables). `contr.ordinal` computes contrasts for ordinal variables.

Value

A matrix with *n* rows and *n-1* columns for `contr.ordinal`, a matrix with *n* rows and *n* columns for `contr.dummy` and a vector of length *n* for `contr.metric`.

Author(s)

Klaus P. Schliep <klaus.schliep@gmail.com>

References

Hechenbichler K. and Schliep K.P. (2004) *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*, Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich (https://doi.org/10.5282/ubm/epub.1769)

See Also

`contrasts`, `contr.poly` and `contr.sdif`

Examples

```
contr.metric(5)
contr.ordinal(5)
contr.dummy(5)
```

---

glass *Glass Identification Database*

Description

A data frame with 214 observations, where the problem is to predict the type of glass in terms of their oxide content (i.e. Na, Fe, K, etc). The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence... if it is correctly identified!

Usage

```
data(glass)
```

Format

A data frame with 214 observations on the following 11 variables.

Id Id number.

RI Refractive index.

Na Sodium (unit measurement: weight percent in corresponding oxide, as are attributes 4-10).

Mg Magnesium.

Al Aluminum.

Si Silicon.

K Potassium.

Ca Calcium.

Ba Barium.

Fe Iron.

Type Type of glass: (class attribute)

1 building windows float processed
2 building windows non float processed
3 vehicle windows float processed
4 vehicle windows non float processed (none in this database)
5 containers
6 tableware
7 headlamps

Source

- Creator: B. German, Central Research Establishment, Home Office Forensic Science Service, Aldermaston, Reading, Berkshire RG7 4PN
- Donor: Vina Spiehler, Ph.D., DABFT, Diagnostic Products Corporation

The data have been taken from the UCI Machine Learning Database Repository
http://www.ics.uci.edu/~mlearn/MLRepository.html
and were converted to R format by <klaus.schliep@gmail.com>.

Examples

```
data(glass)
str(glass)
```

`ionosphere` *Johns Hopkins University Ionosphere Database*

Description

This radar data was collected by a system in Goose Bay, Labrador. This system consists of a phased array of 16 high-frequency antennas with a total transmitted power on the order of 6.4 kilowatts. See the paper for more details. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere.

Received signals were processed using an autocorrelation function whose arguments are the time of a pulse and the pulse number. There were 17 pulse numbers for the Goose Bay system. Instances in this database are described by 2 attributes per pulse number, corresponding to the complex values returned by the function resulting from the complex electromagnetic signal.

Usage

```
data(ionosphere)
```

Format

A data frame with 351 observations on the following 35 variables. The first 34 continuous covariables are used for the prediction. The 35th attribute is either g ("good") or b ("bad") according to the definition summarized above. This is a binary classification task.

Source

Vince Sigillito (vgs@aplcen.apl.jhu.edu), Space Physics Group, Applied Physics Laboratory, Johns Hopkins University, Johns Hopkins Road, Laurel, MD 20723

The data have been taken from the UCI Machine Learning Database Repository
http://www.ics.uci.edu/~mlearn/MLRepository.html
and were converted to R format by <klaus.schliep@gmail.com >.

Examples

```
data(ionosphere)
```

---

kknn *Weighted k-Nearest Neighbor Classifier*

Description

Performs k-nearest neighbor classification of a test set using a training set. For each row of the test set, the k nearest training set vectors (according to Minkowski distance) are found, and the classification is done via the maximum of summed kernel densities. In addition even ordinal and continuous variables can be predicted.

Usage

```
kknn(formula = formula(train), train, test, na.action = na.omit(),
 k = 7, distance = 2, kernel = "optimal", ykernel = NULL, scale=TRUE,
 contrasts = c('unordered' = "contr.dummy", ordered = "contr.ordinal"))
kknn.dist(learn, valid, k = 10, distance = 2)
```

Arguments

formula	A formula object.
train	Matrix or data frame of training set cases.
test	Matrix or data frame of test set cases.
learn	Matrix or data frame of training set cases.
valid	Matrix or data frame of test set cases.
na.action	A function which indicates what should happen when the data contain 'NA's.
k	Number of neighbors considered.
distance	Parameter of Minkowski distance.

kernel	Kernel to use. Possible choices are "rectangular" (which is standard unweighted knn), "triangular", "epanechnikov" (or beta(2,2)), "biweight" (or beta(3,3)), "triweight" (or beta(4,4)), "cos", "inv", "gaussian", "rank" and "optimal".
ykernel	Window width of an y-kernel, especially for prediction of ordinal classes.
scale	logical, scale variable to have equal sd.
contrasts	A vector containing the 'unordered' and 'ordered' contrasts to use.

Details

This nearest neighbor method expands knn in several directions. First it can be used not only for classification, but also for regression and ordinal classification. Second it uses kernel functions to weight the neighbors according to their distances. In fact, not only kernel functions but every monotonic decreasing function $f(x) \forall x > 0$ will work fine.

The number of neighbours used for the "optimal" kernel should be $[(2(d+4)/(d+2))^(d/(d+4))k]$, where k is the number that would be used for unweighted knn classification, i.e. kernel="rectangular". This factor $(2(d+4)/(d+2))^(d/(d+4))$ is between 1.2 and 2 (see Samworth (2012) for more details).

Value

kknn returns a list-object of class kknn including the components

fitted.values	Vector of predictions.
CL	Matrix of classes of the k nearest neighbors.
W	Matrix of weights of the k nearest neighbors.
D	Matrix of distances of the k nearest neighbors.
C	Matrix of indices of the k nearest neighbors.
prob	Matrix of predicted class probabilities.
response	Type of response variable, one of *continuous*, *nominal* or *ordinal*.
distance	Parameter of Minkowski distance.
call	The matched call.
terms	The 'terms' object used.

Author(s)

Klaus P. Schliep <klaus.schliep@gmail.com>
Klaus Hechenbichler

References

Hechenbichler K. and Schliep K.P. (2004) *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*, Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich (https://doi.org/10.5282/ubm/epub.1769)

Hechenbichler K. (2005) *Ensemble-Techniken und ordinale Klassifikation*, PhD-thesis

Samworth, R.J. (2012) *Optimal weighted nearest neighbour classifiers.* Annals of Statistics, 40, 2733-2763. (avaialble from http://www.statslab.cam.ac.uk/~rjs57/Research.html)

See Also

train.kknn

Examples

```
library(kknn)

data(iris)
m <- dim(iris)[1]
val <- sample(1:m, size = round(m/3), replace = FALSE,
 prob = rep(1/m, m))
iris.learn <- iris[-val,]
iris.valid <- iris[val,]
iris.kknn <- kknn(Species~., iris.learn, iris.valid, distance = 1,
 kernel = "triangular")
summary(iris.kknn)
fit <- fitted(iris.kknn)
table(iris.valid$Species, fit)
pcol <- as.character(as.numeric(iris.valid$Species))
pairs(iris.valid[1:4], pch = pcol, col = c("green3", "red")
 [(iris.valid$Species != fit)+1])

data(ionosphere)
ionosphere.learn <- ionosphere[1:200,]
ionosphere.valid <- ionosphere[-c(1:200),]
fit.kknn <- kknn(class ~ ., ionosphere.learn, ionosphere.valid)
table(ionosphere.valid$class, fit.kknn$fit)
(fit.train1 <- train.kknn(class ~ ., ionosphere.learn, kmax = 15,
 kernel = c("triangular", "rectangular", "epanechnikov", "optimal"), distance = 1))
table(predict(fit.train1, ionosphere.valid), ionosphere.valid$class)
(fit.train2 <- train.kknn(class ~ ., ionosphere.learn, kmax = 15,
 kernel = c("triangular", "rectangular", "epanechnikov", "optimal"), distance = 2))
table(predict(fit.train2, ionosphere.valid), ionosphere.valid$class)
```

---

kknn-deprecated *Deprecated Functions in Package kknn*

Description

These functions are provided for compatibility with older versions of R only, and may be defunct as soon as of the next release.

Usage

```
simulation(formula, data, runs = 10, train = TRUE, k = 11, ...)
```

Arguments

formula	A formula object.
data	Matrix or data frame.
runs	Number of crossvalidation runs.
train	A logical value. If TRUE the training procedure for selecting optimal values of k and kernel is performed.
k	Number or maximal number of neighbors considered, dependent of choice for train.
...	Further arguments passed to or from other methods.

Value

A matrix, containing the mean and variance of the misclassification error, the absolute and the squared distances.

References

Hechenbichler K. and Schliep K.P. (2004) *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*, Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich (https://doi.org/10.5282/ubm/epub.1769)

See Also

Defunct and Deprecated

---

miete *Munich Rent Standard Database (1994)*

Description

Many german cities compose so-called rent standards to make a decision making instrument available to tenants, landlords, renting advisory boards and experts. The rent standards are used in particular for the determination of the local comparative rent (i.e. net rent as a function of household size, equipment, year of construction, etc.). For the composition of the rent standards, a representative random sample is drawn from all relevant households, and the interesting data are determined by interviewers by means of questionnaires. The dataset contains the data of 1082 households interviewed for the munich rent standard 1994.

Usage

```
data(miete)
```

Format

A data frame with 1082 observations on the following 18 variables.

nm Net rent in DM.

wfl Floor space in sqm.

bj Year of construction.

bad0 Bathroom in apartment?
- 1 : no
- 0 : yes

zh Central heating?
- 1 : yes
- 0 : no

ww0 Hot water supply?
- 1 : no
- 0 : yes

badkach Tiled bathroom?
- 1 : yes
- 0 : no

fenster Window type:
- 1 : plain windows
- 0 : state-of-the-art windows

kueche Kitchen type
- 1 : well equipped kitchen
- 0 : plain kitchen

mvdauer Lease duration in years.

bjkat Age category of the building (bj categorized)
- 1 : built before 1919
- 2 : built between 1919 and 1948
- 3 : built between 1949 and 1965
- 4 : built between 1966 and 1977
- 5 : built between 1978 and 1983
- 6 : built after 1983

wflkat Floor space category (wfl categorized):
- 1 : less than 50 sqm
- 2 : between 51 sqm and 80 sqm
- 3 : at least 81 sqm

nmqm Net rent per sqm.

rooms Number of rooms in household.

nmkat Net rent category (nm categorized):
- 1 : less than 500 DM
- 2 : between 500 DM and 675 DM
- 3 : between 675 DM and 850 DM
- 4 : between 850 DM and 1150 DM
- 5 : at least 1150 DM

adr Address type:
1 : bad
2 : average
3 : good

wohn Residential type:
1 : bad
2 : average
3 : good

Source

Fahrmeir, L., Kuenstler, R., Pigeot, I. und Tutz, G. (1997): *Statistik: der Weg zur Datenanalyse*, Springer, Berlin. http://www.stat.uni-muenchen.de/service/datenarchiv

The data were converted to R format by <klaus.schliep@gmail.com>.

Examples

```
data(miete)
str(miete)
```

---

specClust *Spectral Clustering*

Description

Spectral clustering based on k-nearest neighbor graph.

Usage

```
specClust(data, centers=NULL, nn = 7, method = "symmetric", gmax=NULL, ...)
## S3 method for class 'specClust'
plot(x, ...)
```

Arguments

data	Matrix or data frame.
centers	number of clusters to estimate, if NULL the number is chosen automatical.
nn	Number of neighbors considered.
method	Normalisation of the Laplacian ("none", "symmetric" or "random-walk").
gmax	maximal number of connected components.
x	an object of class specClust
...	Further arguments passed to or from other methods.

Details

specClust alllows to estimate several popular spectral clustering algorithms, for an overview see von Luxburg (2007).

The Laplacian is constructed from a from nearest neighbors and there are several kernels available. The eigenvalues and eigenvectors are computed using the binding in igraph to arpack. This should ensure that this algorithm is also feasable for larger datasets as the the the distances used have dimension n*m, where n is the number of observations and m the number of nearest neighbors. The Laplacian is sparse and has roughly n*m elements and only k eigenvectors are computed, where k is the number of centers.

Value

specClust returns a kmeans object or in case of k being a vector a list of kmeans objects.

Author(s)

Klaus P. Schliep <klaus.schliep@gmail.com>

References

U. von Luxburg (2007) A tutorial on spectral clustering, *Stat Comput*, **17**, 395–416

Ng, A., Jordan, M., Weiss, Y. (2002) On spectral clustering: analysis and an algorithm. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) *Advances in Neural Information Processing Systems*, **14**, 849–856. MIT Press, Cambridge

Lihi Zelnik-Manor and P. Perona (2004) Self-Tuning Spectral Clustering, *Eighteenth Annual Conference on Neural Information Processing Systems, (NIPS)*

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22 (8)**, 888–905

See Also

kknn, arpack, kmeans

Examples

```
data(iris)
cl <- specClust(iris[,1:4], 3, nn=5)
pcol <- as.character(as.numeric(iris$Species))
pairs(iris[1:4], pch = pcol, col = c("green", "red", "blue")[cl$cluster])
table(iris[,5], cl$cluster)
```

`train.kknn` *Training kknn*

Description

Training of kknn method via leave-one-out (`train.kknn`) or k-fold (`cv.kknn`) crossvalidation.

Usage

```
train.kknn(formula, data, kmax = 11, ks = NULL, distance = 2, kernel = "optimal",
 ykernel = NULL, scale = TRUE, contrasts = c('unordered' = "contr.dummy",
 ordered = "contr.ordinal"), ...)
cv.kknn(formula, data, kcv = 10, ...)
```

Arguments

`formula`	A formula object.
`data`	Matrix or data frame.
`kmax`	Maximum number of k, if `ks` is not specified.
`ks`	A vector specifying values of k. If not null, this takes precedence over `kmax`.
`distance`	Parameter of Minkowski distance.
`kernel`	Kernel to use. Possible choices are "rectangular" (which is standard unweighted knn), "triangular", "epanechnikov" (or beta(2,2)), "biweight" (or beta(3,3)), "triweight" (or beta(4,4)), "cos", "inv", "gaussian" and "optimal".
`ykernel`	Window width of an y-kernel, especially for prediction of ordinal classes.
`scale`	logical, scale variable to have equal sd.
`contrasts`	A vector containing the 'unordered' and 'ordered' contrasts to use.
`...`	Further arguments passed to or from other methods.
`kcv`	Number of partitions for k-fold cross validation.

Details

`train.kknn` performs leave-one-out crossvalidation and is computatioanlly very efficient. `cv.kknn` performs k-fold crossvalidation and is generally slower and does not yet contain the test of different models yet.

Value

`train.kknn` returns a list-object of class `train.kknn` including the components.

`MISCLASS`	Matrix of misclassification errors.
`MEAN.ABS`	Matrix of mean absolute errors.
`MEAN.SQU`	Matrix of mean squared errors.
`fitted.values`	List of predictions for all combinations of kernel and k.

best.parameters	List containing the best parameter value for kernel and k.
response	Type of response variable, one of *continuous*, *nominal* or *ordinal*.
distance	Parameter of Minkowski distance.
call	The matched call.
terms	The 'terms' object used.

Author(s)

Klaus P. Schliep <klaus.schliep@gmail.com>

References

Hechenbichler K. and Schliep K.P. (2004) *Weighted k-Nearest-Neighbor Techniques and Ordinal Classification*, Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich (https://doi.org/10.5282/ubm/epub.1769)

Hechenbichler K. (2005) *Ensemble-Techniken und ordinale Klassifikation*, PhD-thesis

Samworth, R.J. (2012) *Optimal weighted nearest neighbour classifiers.* Annals of Statistics, 40, 2733-2763. (avaialble from http://www.statslab.cam.ac.uk/~rjs57/Research.html)

See Also

kknn and simulation

Examples

```
library(kknn)
## Not run:
data(miete)
(train.con <- train.kknn(nmqm ~ wfl + bjkat + zh, data = miete,
 kmax = 25, kernel = c("rectangular", "triangular", "epanechnikov",
 "gaussian", "rank", "optimal")))
plot(train.con)
(train.ord <- train.kknn(wflkat ~ nm + bjkat + zh, miete, kmax = 25,
  kernel = c("rectangular", "triangular", "epanechnikov", "gaussian",
  "rank", "optimal")))
plot(train.ord)
(train.nom <- train.kknn(zh ~ wfl + bjkat + nmqm, miete, kmax = 25,
 kernel = c("rectangular", "triangular", "epanechnikov", "gaussian",
 "rank", "optimal")))
plot(train.nom)

## End(Not run)
data(glass)
glass <- glass[,-1]
(fit.glass1 <- train.kknn(Type ~ ., glass, kmax = 15, kernel =
 c("triangular", "rectangular", "epanechnikov", "optimal"), distance = 1))
(fit.glass2 <- train.kknn(Type ~ ., glass, kmax = 15, kernel =
 c("triangular", "rectangular", "epanechnikov", "optimal"), distance = 2))
plot(fit.glass1)
```

```
plot(fit.glass2)
```

Index