Package 'labelrank'

November 22, 2015

Type Package

Title Predicting Rankings of Labels

Version 0.1

Date 2015-11-20

Depends R (>= 2.10)

Imports pdist

Description
An implementation of distance-based ranking algorithms to predict rankings of labels. Two common algorithms are included: the naive Bayes and the nearest neighbor algorithms.

Suggests knitr

VignetteBuilder knitr

RoxygenNote 5.0.1

LazyData true

License GPL-3

NeedsCompilation no

Author Artur Aiguzhinov [cre],
Carlos Soares [aut]

Maintainer Artur Aiguzhinov <aiguzhinov@gmail.com>

Repository CRAN

Date/Publication 2015-11-22 11:48:10

R topics documented:

---

kNN *Nearest neighbor*

Description

An auxiliary function to find the nearest neighbors from the distance matrix

Usage

```
kNN(model, k)
```

Arguments

model	nearest neighbor ranking model
k	number of the nearest neighbors to consider

Details

This function is applied to find the nearest neighbors in the distance matrix.

Value

a vector of length of model

---

lr.nom *Synthetic data for label ranking experiments.*

Description

Synthetic data for label ranking experiments.

Usage

```
lr.nom
```

Format

A matrix of 20 discrete values and 2 columns

lr.num *Synthetic data for label ranking experiments.*

Description

Synthetic data for label ranking experiments.

Usage

```
lr.num
```

Format

A matrix of 20 numeric values and 2 columns

model_nbr *A naive Bayes label ranking model*

Description

This is an auxiliary function to build a necessary inputs to predict rankings.

Usage

```
model_nbr(x, y, n = 1)
```

Arguments

x	is n x p matrix of n observations and p training attributes and can have continuous or nominal values.
y	is n x j matrix of training rankings (permutations).
n	is a parameter of 'memory'; that is, how fast past gets forgotten. (see details of time_weights).

Value

a list of size two: prior and conditional label ranking probabilities.

nb_rank *Predicting label rankings based on the naive Bayes ranking model*

Description

This function predicts the rankings given prior and conditional probabilities obtained from `model_nbr`

Usage

```
nb_rank(x, y, new.x, n = 1)
```

Arguments

`x`	is `n x p` matrix of `n` observations and `p` training attributes and can have continuous or nominal values.
`y`	is `n x j` matrix of label rankings
`new.x`	is a vector of new attributes
`n`	is a parameter of 'memory'; that is, how fast past gets forgotten. (see details of time_weights).

Details

This function predicts a ranking for `test.x` attributes. It initially builds a model for naive Bayes algorithm that calculates priors and conditional label ranking probabilities and then use them to predict rankings. The attributes can be nominal or continuous data.

Value

a numeric vector of ranking

Examples

```
train.x <- lr.nom[1:16,]
test.x <- lr.nom[17,]
predrank <- nb_rank(train.x,y,test.x,n=1)
```

nn_rank *Predicting rankings using the nearest neighbor algorithm*

Description

This function makes prediction of rankings based on the nearest neighbor

Usage

```
nn_rank(train.x, y, test.x, n = 1, k = 3)
```

Arguments

train.x	is matrix of numeric attributes in training sample
y	is matrix of training rankings
test.x	is a vector of new numeric attributes for which to predict rankings
n	is a parameter of 'memory' of how fast the past rankings gets forgotten. (see details of time_weights). By default, n=1 which means that a label ranking problem does not have timing effect.
k	is the number of the nearest neighbors to consider (default k=3)

Details

A function predicts the rankings based on the euclidean distance between train and test attributes.

Value

a vector of predicted ranking for attribute test.x

Examples

```
train.x <- lr.num[1:16,]
test.x <- lr.num[17,]
ranking <- nn_rank(train.x, y, test.x, n=1,k=3)
```

time_weights *Weights for timing*

Description

This function calculates the diminishing weights for label ranking probabilities in case of timing nature of rankings.

Usage

```
time_weights(x, n)
```

Arguments

x	a scalar of timing periods.
n	is a parameter of 'memory' of how fast the past gets forgotten.

Details

Sometimes rankings have a 'timing' component (for example, weekly sport teams standing) and a recent event can be more important than the past. The model can take advantage of this difference in importance by weighting the ranking probabilities. The weights are calculated using an exponential function . In case of n=1, weights are a unitary vector of length n; thus, no time nature in rankings.

Value

a vector of values.

y *Permutations to be used as rankings*

Description

Permutations to be used as rankings

Usage

```
y
```

Format

A matrix of 20 rows and 3 columns

Index