Package 'CatEncoders'

September 11, 2016

Type Package

Title Encoders for Categorical Variables

Version 0.1.0

Author nl zhang

Maintainer nl zhang <setseed2016@gmail.com>

Description This package contins some commonly used categorical variable encoders, such as LabelEncoder and OneHotEncoder. The package is inspired by the encoders implemented in python sklearn.preprocessing package.

License GPL-2 | GPL-3

LazyData TRUE

Imports Matrix (>= 1.2-6), data.table (>= 1.9.6), methods

RoxygenNote 5.0.1

R topics documented:

`inverse.transform` *inverse.transform transforms an integer vector back to the original vector*

Description

inverse.transform transforms an integer vector back to the original vector

Usage

```
inverse.transform(enc, z)

## S4 method for signature 'LabelEncoder,numeric'
inverse.transform(enc, z)
```

Arguments

enc	A fitted LabelEncoder
z	A vector of integers

Value

A vector of characters, factors or numerics.

Examples

```
# character vector y
y <- c('a','d','e',NA)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,c('d','d',NA,'f'))
print(z)
inverse.transform(lenc,z)

# factor vector y
y <- factor(c('a','d','e',NA),exclude=NULL)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,factor(c('a','d',NA,'f')))
inverse.transform(lenc,z)

# numeric vector y
set.seed(123)
y <- c(1:10,NA)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
newy <- sample(c(1:10,NA),5)
print(newy)
z <-transform(lenc,newy)
inverse.transform(lenc, z)
```

`LabelEncoder-class` *An S4 class to represent a LabelEncoder.*

Description

An S4 class to represent a LabelEncoder.

Slots

`type` A character to denote the input type, either character, factor or numeric

`mapping` A data.frame to store the mapping table

`LabelEncoder.Character-class` *An S4 class to represent a LabelEncoder with character input.*

Description

An S4 class to represent a LabelEncoder with character input.

Slots

`classes` A character vector to store the unique values of classes

`LabelEncoder.Factor-class` *An S4 class to represent a LabelEncoder with factor input.*

Description

An S4 class to represent a LabelEncoder with factor input.

Slots

`classes` A factor vector to store the unique values of classes

LabelEncoder.fit *LabelEncoder.fit fits a LabelEncoder object*

Description

LabelEncoder.fit fits a LabelEncoder object

Usage

```
LabelEncoder.fit(y)
```

Arguments

y	A vector of characters, factors, or numerics, which can include NA as well

Value

Returns an object of S4 class LabelEncoder.

Examples

```
# factor y
y <- factor(c('a','d','e',NA),exclude=NULL)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,factor(c('d','d',NA,'f')))
print(z)

# character y
y <- c('a','d','e',NA)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,c('d','d',NA,'f'))
print(z)

# numeric y
set.seed(123)
y <- sample(c(1:10,NA),5)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <-transform(lenc,sample(c(1:10,NA),5))
print(z)
```

LabelEncoder.Numeric-class *An S4 class to represent a LabelEncoder with numeric input.*

Description

An S4 class to represent a LabelEncoder with numeric input.

Slots

`classes` A numeric vector to store the unique values of classes

`OneHotEncoder-class` *An S4 class to represent a OneHotEncoder*

Description

An S4 class to represent a OneHotEncoder

Slots

`n_columns` An integer value to store the number of columns of input data

`n_values` A numeric vector to store the number of unique values in each column of input data

`column_encoders` A list that stores the LabelEncoder for each column of input data

---

`OneHotEncoder.fit` *OneHotEncoder.fit fits an OneHotEncoder object*

Description

OneHotEncoder.fit fits an OneHotEncoder object

Usage

```
OneHotEncoder.fit(X)
```

Arguments

`X`	A matrix or data.frame, which can include NA

Value

Returns an object of S4 class OneHotEncoder

Examples

```
# matrix input
X1 <- matrix(c(0, 1, 0, 1, 0, 1, 2, 0, 3, 0, 1, 2),c(4,3),byrow=FALSE)
oenc <- OneHotEncoder.fit(X1)
z <- transform(oenc,X1,sparse=TRUE)
# return a sparse matrix
print(z)

# data.frame
X2 <- cbind(data.frame(X1),X4=c('a','b','d',NA),X5=factor(c(1,2,3,1)))
oenc <- OneHotEncoder.fit(X2)
z <- transform(oenc,X2,sparse=FALSE)
# return a dense matrix
print(z)
```

`transform` *transform transforms a new data set using the fitted encoder*

Description

transform transforms a new data set using the fitted encoder

Usage

```
transform(enc, ...)

## S4 method for signature 'LabelEncoder.Numeric'
transform(enc, y)

## S4 method for signature 'LabelEncoder.Character'
transform(enc, y)

## S4 method for signature 'LabelEncoder.Factor'
transform(enc, y)

## S4 method for signature 'OneHotEncoder'
transform(enc, X, sparse = TRUE,
  new.feature.error = TRUE)
```

Arguments

`enc`	A fitted encoder, i.e., LabelEncoder or OneHotEncoder
`...`	Additional argument list
`y`	A vector of character, factor or numeric values
`X`	A data.frame or matrix
`sparse`	If TRUE then return a sparse matrix, default = TRUE
`new.feature.error`	If TRUE then throw an error for new feature values; otherwise the new feature values are ignored, default = TRUE

Value

If enc is an OneHotEncoder, the returned value is a sparse or dense matrix. If enc is a LabelEncoder, the returned value is a vector.

Examples

```
# matrix X
X1 <- matrix(c(0, 1, 0, 1, 0, 1, 2, 0, 3, 0, 1, 2),c(4,3),byrow=FALSE)
oenc <- OneHotEncoder.fit(X1)
z <- transform(oenc,X1,sparse=TRUE)
# return a sparse matrix
print(z)

# data.frame X
X2 <- cbind(data.frame(X1),X4=c('a','b','d',NA),X5=factor(c(1,2,3,1)))
```

```
oenc <- OneHotEncoder.fit(X2)
z <- transform(oenc,X2,sparse=FALSE)
# return a dense matrix
print(z)

# factor vector y
y <- factor(c('a','d','e',NA),exclude=NULL)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,factor(c('d','d',NA,'f')))
print(z)

# character vector y
y <- c('a','d','e',NA)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <- transform(lenc,c('d','d',NA,'f'))
print(z)

# numeric vector y
set.seed(123)
y <- sample(c(1:10,NA),5)
lenc <- LabelEncoder.fit(y)
# new values are transformed to NA
z <-transform(lenc,sample(c(1:10,NA),5))
print(z)
```

Index