Package 'iqspr'

April 24, 2017

Type Package

Title Generate Chemical Strings (SMILES) with the Inverse QSPR Model

Version 1.1

Date 2017-4-23

Author Hisaki Ikebata

Maintainer Ryo Yoshida <yoshidar@ism.ac.jp>

Description Generate chemical structures possibly satisfying desired properties
using the inverse QSPR model. It has three reference classes. ENgram is a class for learning the grammar
structure of existing chemical strings using an extended N-gram model. QSPRpred contains a simple Bayes regression model to predict properties from structures. SmcChem is a class of the generator of
chemical strings from the Inverse-
QSPR model. This class has ENgram and QSPRpred class objects inside.
The generator is implemented by the Sequential Monte Carlo sampler.

License Artistic-2.0

SystemRequirements OpenBabel (>= 2.3.1) with headers.
http://openbabel.org

Suggests testthat

Depends R (>= 2.10), rcdk

Imports methods

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-04-24 06:02:52 UTC

R topics documented:

`ENgram-class` *Extended N-gram model for leraning SMILES strings*

Description

Extended N-gram model for leraning SMILES strings

Methods

`initialize(smis = NULL, order = NULL)` Initialize the extend N-gram model with SMILES strings smis and numeric value order

`update_mat(data)` update the model with additional SMILES strings data

Examples

```
data(trainedSMI)
data(engram_5k)  #same as run => engram <- ENgram$new(trainedSMI, order=10)
#-----arguments
#smis: character vector, SMILES string set for training an extended N-gram model
#order: numeric, value representing the maximum order in modified N-gram model
#-----arguments
```

engram_5k *ENgram object trained with 5000 SMILES strings*

Description

ENgram object trained with 5000 SMILES strings

Usage

```
data("engram_5k")
```

Examples

```
data(engram_5k)
```

Esmi-class *Extend smi class*

Description

Extend smi class

genENgram *generate SMILES strings from extended N-gram model*

Description

generate SMILES strings from extended N-gram model

Usage

```
genENgram(nsmis, engram, order, gentype = "ML", crange = c(10, 100))
```

Arguments

nsmis	number of generating SMILES strings
engram	ENgram object
order	n in ENgram model
gentype	Back-off procedure with "ML" option and Neaser-Nay smoothing with "KN" option
crange	range in the length of output SMILES strings

Examples

```
data(engram_5k)
smiles <- genENgram(4, engram_5k, 10)
viewstr(smiles)
```

`get_hiscores` *get chemical structures with high QSPR score from SmcChem object*

Description

get chemical structures with high QSPR scores from the SmcChem object after excluding similar structures

Usage

```
get_hiscores(smchem, nsmi = 50, exsim = 0.8)
```

Arguments

`smchem`	SmcChem class object
`nsmi`	maximum number of SMILES strings obtained
`exsim`	excluding structures that are similar with already chosen structures in that the Tanimoto coefficient >= exsim

Examples

```
data(engram_5k)
data(qsprpred_EG_5k)
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3)
smcexec(smchem,10)
res <- get_hiscores(smchem, exsim=0.8)
viewstr(res[1:4, 1])
```

`get_score` *get QSPR scores for structures in the SmcChem object*

Description

get QSPR scores for structures in the SmcChem object

Usage

```
get_score(smchem)
```

Arguments

smchem	SmcChem object

Examples

```
data(engram_5k)
data(qsprpred_EG_5k)
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3)
scores <- matrix(0, 25, 5)
for(i in 1:5){
smchem$smcexec(1)
scores[,i] <- get_score(smchem)
boxplot(scores)
}
```

---

get_smiles *get SMILES strings from the SmcChem object*

Description

get SMILES strings from the SmcChem object

Usage

```
get_smiles(smchem)
```

Arguments

smchem	SmcChem class object

Examples

```
data(engram_5k)
data(qsprpred_EG_5k)
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3)
smcexec(smchem, niter=5, preorder=0, nview=4)
get_smiles(smchem)
```

iqspr *iqspr*

Description

Generate chemical structures from Inverse-QSPR model

Examples

```
#sample data
data(qspr.data)
idx <- sample(nrow(qspr.data), 5000)
smis <- paste(qspr.data[idx,1])
y <- qspr.data[idx,c(2,5)]

#learning a pattern of chemical strings
data(trainedSMI)
data(engram_5k)  #same as run => engram <- ENgram$new(trainedSMI, order=10)

#learning QSPR model
data(qsprpred_EG_5k)
#same as run => qsprpred <- QSPRpred$new(smis=smis, y=as.matrix(y), v_fpnames="graph")

#set target range
qsprpred_EG_5k$ymin <- c(200, 1.5)
qsprpred_EG_5k$ymax <- c(350, 2.5)

#getting chemical strings from the Inverse-QSPR model
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3, decay=0.95)
smchem$smcexec(niter=5, preorder=0, nview=4)
#if OpenBabel (>= 2.3.1) is installed, you can use reordering for better mixing as
#smchem$smcexec(niter=100, preorder=0.2, nview=4)
#see http://openbabel.org

#check
smiles <- get_smiles(smchem)
predict(qsprpred_EG_5k, smiles[1:5])
```

predict *predict properties with forward QSPR model*

Description

predict properties with forward QSPR model

Usage

```
predict(qsprpred, smis)
```

Arguments

qsprpred	QSPRpred class object
smis	SMILES strings

Examples

```
data(qsprpred_EG_5k)
predict(qsprpred_EG_5k, c("c1ccccc1O", "c1ccccc1F"))
```

---

qspr.data *sample dataset for QSPR modeling*

Description

This is sample dataset for QSPR modeling. The first column is SMILES strings, and the rest 4 columns are properties.

Usage

```
data("qspr.data")
```

Examples

```
data(qspr.data)
smis <- paste(qspr.data[,1])
ty <- qspr.data[,c(2,5)]
```

QSPRpred-class *QSPRpredictor class*

Description

QSPR model construction using the Bayesian linear regression model.

Fields

`ymin` vector representing minimal value in each target property

`ymax` vector representing maximum value in each target property

`descriptor` function transforming SMILES strings into numeric vector (matrix)

`fpnames` name of fingerprint used in a predictor

Methods

`initialize(smis = NULL, y = NULL, v_ymin = NULL, v_ymax = NULL, v_descriptor = NULL, v_fpnames = NU` initialize the QSPR predictor

`inverse_predx(smis = NULL, temp = 1)` Inverse QSPR prediction of input SMILES strings smis for target property range

`qspr_predx(smis = NULL)` QSPR prediction of input SMILES strings smis

`set_targety(v_ymin, v_ymax)` set the target propety range

Examples

```
data(qspr.data)
smis <- paste(qspr.data[,1])
ty <- qspr.data[,c(2,5)]
trainidx <- sample(1:nrow(qspr.data), 5000)
testidx <- (1:nrow(qspr.data))[-trainidx][1:100]

data(qsprpred_EG_5k)
## same as run => qsprpred_EG_5k <-

#-----arguments
#smis: SMILES string set (character vector) for training
#y: property sets (matrix) for training
#v_ymin: minimum value of target properties
#v_ymax: maximum value of target properties
#v_descriptor: function transforming SMILES strings to feature matrix
#v_fpnames: character vectors indicating finger print used in the descriptor
#w0: matrix representing prior mean of coeffients in linear regression model
#V0_inv: matrix representing the prior variance of coeffients in linear regression model
#a0: numeric value representing the location parameter in gamma prior
#b0: numeric value representing the shape parameter in gamma prior
#------

predictions <- qsprpred_EG_5k$qspr_predx(smis[testidx])
par(mfrow=c(1,2))
plot(predictions[[1]][1,], ty[testidx,1])
plot(predictions[[1]][2,], ty[testidx,2])

#computing the probability which the properties of test structures is in the target range
# set the minimal values in 2-d target properties
qsprpred_EG_5k$ymin <- c(100, 4)
# set the maximum values in 2-d target properties
qsprpred_EG_5k$ymin <- c(200, 5.5)
# method inverse_predx returns the probability that input SMILES has property in target range
qsprpred_EG_5k$inverse_predx("c1ccccc1O")
```

`qsprpred_EG_5k` *QSPRpred object trained with 5000 SMILES strings and properties (Internal energy and HOMO-LUMO gap) dataset*

Description

QSPRpred object trained with 5000 SMILES strings and properties (Internal energy and HOMO-LUMO gap) dataset

Usage

```
data("engram_5k")
```

Examples

```
data(engram_5k)
```

SmcChem-class *SMC chemical generator class*

Description

SMILES generator with Sequence Monte Calro sampler

Fields

`qsprpred` QSPRpred object

`engram` ENgram object

`m` numeric value representing the order of extended N-gram model

`v_ESSth` numeric value representing the threshold which resample is done when ESS is less than 100 * v_ESSth

`v_decay` numeric value decaying temparature for the target distribution in SMC sampler

Methods

`get_hiscores(nsmi = 100, exsim = 0.8)` get chemical structures with high QSPR score from SmcChem object (same as get_hiscores function)

`get_smiles()` get SMILES strings from the SmcChem object (same as get_smiles function)

`initialize(smis = NULL, v_engram = NULL, v_qsprpred = NULL, v_m = NULL, temp = 1, ESSth = 0.5, lamb` Initialize the SMC chemical generator with initial SMILES strings smis, ENgram class object v_engram and QSPRpred class object v_qsprpred

`smcexec(niter, nsteps = 5, preorder = 0, nview = 0)` modify chemical structures with niter SMC updates

`viewstr(idx)` view 2D structures from SMILES string vector with index idx (same as viewstr function)

Examples

```
#sample data
data(qspr.data)
idx <- sample(nrow(qspr.data), 5000)
smis <- paste(qspr.data[idx,1])
y <- qspr.data[idx,c(2,5)]

#learning a pattern of chemical strings
data(trainedSMI)
data(engram_5k)  #same as run => engram <- ENgram$new(trainedSMI, order=10)

#learning QSPR model
data(qsprpred_EG_5k)
#same as run => qsprpred <- QSPRpred$new(smis=smis, y=as.matrix(y), v_fpnames="graph")

#set target range
qsprpred_EG_5k$ymin <- c(200, 1.5)
qsprpred_EG_5k$ymax <- c(350, 2.5)

#getting chemical strings from the Inverse-QSPR model
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3)
#-----arguments
#smis: initial SMILES strings in SMC sampler
#v_qsprpred: QSPRpred object
#v_engram: ENgram object
#v_m: numeric value representing the order of modified N-gram model
#ESSth: numeric resampling threshold at 100 * v_ESSth
#temp: numeric, annealing parameter in SMC sampler
#lambda: numeric
#decay: numeric, decaying rate for temparature
#-----arguments

smchem$smcexec(niter=5, preorder=0, nview=4)
#if OpenBabel (>= 2.3.1) is installed, you can use reordering for better mixing as
#smchem$smcexec(niter=100, preorder=0.2, nview=4)
#see http://openbabel.org

#check
smiles <- get_smiles(smchem)
predict(qsprpred_EG_5k, smiles[1:5])
```

`smcexec` *modify chemical structures with SMC*

Description

modify chemical structures with SMC

Usage

```
smcexec(smchem, niter, nsteps = 5, preorder = 0, nview = 0)
```

Arguments

smchem	SmcChem object
niter	number of iterations
nsteps	total number of letters extending and contracting in a iteration
preorder	probability for reordering SMILES
nview	number of structures shown after each iteration (if zero, structures are not shown)

Examples

```
data(engram_5k)
data(qsprpred_EG_5k)
smchem <- SmcChem$new(smis = rep("c1ccccc1O", 25), v_qsprpred=qsprpred_EG_5k,
                      v_engram=engram_5k,temp=3)
smcexec(smchem, niter=5, preorder=0, nview=4)
#if OpenBabel (>= 2.3.1) is installed, you can use reordering for better mixing as
#smcexec(smchem, niter=100, preorder=0.2, nview=4)
#see http://openbabel.org
```

---

trainedSMI *sample dataset for learning with Extended N-gram model*

Description

sample dataset for learning with Extended N-gram model 5000 SMILES strings are included.

Usage

```
data("qspr.data")
```

Examples

```
data(trainedSMI)
trainedSMI[1:10]
```

viewstr *view 2D structures from SMILES string vector*

Description

view 2D structures from SMILES string vector

Usage

```
viewstr(smis)
```

Arguments

smis	SMILES string vector

Examples

```
viewstr(c("c1ccc2ccc3c(NCCN(C)C)cc(nc3c2c1)", "c1ccc2ccc3c(NCCN(CC)CCCl)cc(nc3c2c1)",
 "c1ccc2ccc3c(NC(CC)CC)cc(nc3c2c1)", "c1ccc2ccc3c(c2c1)ncc(c3NCCNCC=CCCCC)"))
```

Index