Package 'topicmodels'

July 1, 2015

Type Package

Title Topic Models

Version 0.2-2

Description Provides an interface to the C code for Latent Dirichlet
Allocation (LDA) models and Correlated Topics Models
(CTM) by David M. Blei and co-authors and the C++ code
for fitting LDA models using Gibbs sampling by Xuan-Hieu
Phan and co-authors.

Depends R (>= 2.15.0)

Imports stats4, methods, modeltools, slam, tm (>= 0.6)

Suggests lasso2, lattice, lda, OAIHarvester, SnowballC, XML,
corpus.JSS.papers

Additional_repositories http://datacube.wu.ac.at

SystemRequirements GNU Scientific Library version >= 1.8

License GPL-2

Encoding UTF-8

LazyLoad yes

NeedsCompilation yes

Author Bettina Grün [aut, cre],
Kurt Hornik [aut]

Maintainer Bettina Grün <Bettina.Gruen@jku.at>

Repository CRAN

Date/Publication 2015-07-01 16:06:34

R topics documented:

Index   **15**

AssociatedPress   *Associated Press data*

Description

Associated Press data from the First Text Retrieval Conference (TREC-1) 1992.

Usage

```
data("AssociatedPress")
```

Format

The data set is an object of class "DocumentTermMatrix" provided by package **tm**. It is a document-term matrix which contains the term frequency of 10473 terms in 2246 documents.

Source

Accompanying material to the source code for fitting LDA models provided by David M. Blei and co-authors. Downloaded from: <http://www.cs.columbia.edu/~blei/>

References

D. Harman (1992) Overview of the first text retrieval conference (TREC-1). In Proceedings of the First Text Retrieval Conference (TREC-1), 1–20.

---

build_graph *Construct the adjacency matrix for a topic graph*

Description

The lasso is used to determine which edges are present in a topic graph. The original R code was written by David M. Blei and co-authors and is available together with the C code for fitting the CTM.

Usage

```
build_graph(x, lambda, and = TRUE)
```

Arguments

x	Object of class "CTM".
lambda	Numeric in $[0, 1]$ indicating the relative bound on the L1-norm of the parameters.
and	Logical; if TRUE the graph is computed by taking the intersection of the neighbors, otherwise the union is determined.

Value

Returns an adjacency matrix for the topics versus topics graph.

Author(s)

David M. Blei, modified by Bettina Gruen.

References

Blei D.M., Lafferty J.D. (2007). A Correlated Topic Model of Science. *The Annals of Applied Statistics*, **1**(1), 17–35.

CTM *Correlated Topic Model*

Description

Estimate a CTM model using for example the VEM algorithm.

Usage

```
CTM(x, k, method = "VEM", control = NULL, model = NULL, ...)
```

Arguments

x	Object of class `"DocumentTermMatrix"` with term-frequency weighting or an object coercible to a `"simple_triplet_matrix"` with integer entries.
k	Integer; number of topics.
method	The method to be used for fitting; currently only `method = "VEM"` is supported.
control	A named list of the control parameters for estimation or an object of class `"CTM_VEMcontrol"`.
model	Object of class `"CTM"` for initialization.
...	Currently not used.

Details

The C code for CTM from David M. Blei and co-authors is used to estimate and fit a correlated topic model.

Value

CTM() returns an object of class `"CTM"`.

Author(s)

Bettina Gruen

References

Blei D.M., Lafferty J.D. (2007). A Correlated Topic Model of Science. *The Annals of Applied Statistics*, **1**(1), 17–35.

See Also

`"CTM_VEMcontrol"`

Examples

```
data("AssociatedPress", package = "topicmodels")
ctm <- CTM(AssociatedPress[1:20,], k = 2)
```

---

distHellinger *Compute Hellinger distance*

Description

The Hellinger distance between the rows of two data matrices are determined or if the second argument is missing between the rows of one data matrix.

Usage

```
## Default S3 method:
distHellinger(x, y, ...)
## S3 method for class 'simple_triplet_matrix'
distHellinger(x, y, ...)
```

Arguments

x   A data matrix.

y   A data matrix.

...   Currently not used.

Value

A matrix containing the distances.

Author(s)

Bettina Gruen

LDA *Latent Dirichlet Allocation*

Description

Estimate a LDA model using for example the VEM algorithm or Gibbs Sampling.

Usage

```
LDA(x, k, method = "VEM", control = NULL, model = NULL, ...)
```

Arguments

x	Object of class "DocumentTermMatrix" with term-frequency weighting or an object coercible to a "simple_triplet_matrix" with integer entries.
k	Integer; number of topics.
method	The method to be used for fitting; currently method = "VEM" or method= "Gibbs" are supported.
control	A named list of the control parameters for estimation or an object of class "LDAcontrol".
model	Object of class "LDA" for initialization.
...	Optional arguments. For method = "Gibbs" an additional argument seedwords can be specified as a matrix or an object of class "simple_triplet_matrix"; the default is NULL.

Details

The C code for LDA from David M. Blei and co-authors is used to estimate and fit a latent dirichlet allocation model with the VEM algorithm. For Gibbs Sampling the C++ code from Xuan-Hieu Phan and co-authors is used.

When Gibbs sampling is used for fitting the model, seed words with their additional weights for the prior parameters can be specified in order to be able to fit seeded topic models.

Value

LDA() returns an object of class "LDA".

Author(s)

Bettina Gruen

References

Blei D.M., Ng A.Y., Jordan M.I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, **3**, 993–1022.

Phan X.H., Nguyen L.M., Horguchi S. (2008). Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In Proceedings of the 17th International World Wide Web Conference (WWW 2008), pages 91–100, Beijing, China.

Lu, B., Ott, M., Cardie, C., Tsou, B.K. (2011). Multi-aspect Sentiment Analysis with Topic Models. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, pages 81–88.

See Also

"LDAcontrol"

Examples

```
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
lda_inf <- posterior(lda, AssociatedPress[21:30,])
```

---

ldaformat2dtm	*Transform data from and for use with the* **lda** *package*

Description

Data from the **lda** package is transformed to a document-term matrix. This data format can be used to fit topic models using package **topicmodels**.

Data in form of a document-term matrix is transformed to the LDA format used by package **lda**.

Usage

```
ldaformat2dtm(documents, vocab, omit_empty = TRUE)
dtm2ldaformat(x, omit_empty = TRUE)
```

Arguments

documents	A list where each entry corresponds to a document; for each document the number of terms occurring in the document are stored in a matrix with two rows such that in each column the first entry corresponds to the vocabulary id of the term and the second entry to the number of times this term occurred in the document.
vocab	A "character" vector of the terms in the vocabulary.
x	An object of class "DocumentTermMatrix" as defined in package **tm**.
omit_empty	A logical indicating if empty documents should be removed when converting the objects. By default empty documents are removed.

Value

An object of class "DocumentTermMatrix" is returned by ldaformat2dtm() and a list with components "documents" and "vocab" by dtm2ldaformat().

Author(s)

Bettina Gruen

Examples

```
if (require("lda")) {
  data("cora.documents", package = "lda")
  data("cora.vocab", package = "lda")
  dtm <- ldaformat2dtm(cora.documents, cora.vocab)
  cora <- dtm2ldaformat(dtm)
  all.equal(cora, list(documents = cora.documents,
                       vocab = cora.vocab))
}
```

---

logLik-methods   *Methods for Function logLik*

Description

Compute the log-likelihood.

Methods

object = TopicModel: Compute the log-likelihood of a "TopicModel" object. For "VEM" objects the sum of the log-likelihood of all documents given the parameters for the topic distribution and for the word distribution of each topic is approximated using the variational parameters and underestimates the log-likelihood by the Kullback-Leibler divergence between the variational posterior probability and the true posterior probability.

object = Gibbs_list: Compute the log-likelihoods of the "TopicModel" objects contained in the "Gibbs_list" object.

---

perplexity   *Methods for Function perplexity*

Description

Determine the perplexity of a fitted model.

Usage

```
perplexity(object, newdata, ...)

## S4 method for signature 'VEM,simple_triplet_matrix'
perplexity(object, newdata, control, ...)

## S4 method for signature 'Gibbs,simple_triplet_matrix'
perplexity(object, newdata, control, use_theta = TRUE,
estimate_theta = TRUE, ...)
```

```
## S4 method for signature 'Gibbs_list,simple_triplet_matrix'
perplexity(object, newdata, control, use_theta  = TRUE,
estimate_theta = TRUE, ...)
```

Arguments

object	Object of class "TopicModel" or "Gibbs_list".
newdata	If missing, the perplexity for the data to which the model was fitted is determined. For objects fitted using Gibbs sampling newdata needs to be specified.
control	If missing, the control of the fitted model is used with suitable changes of the relevant parameters (see **Details**).
use_theta	Object of class "logical". If TRUE the estimated topic distributions for the documents are used. Otherwise equal weights are assigned to the topics for each document.
estimate_theta	Object of class "logical". If FALSE the data provided is assumed to be the same as the data used for fitting the model. The topic distributions therefore do not need to be estimated and the data in newdata is used for weighting the term-document occurrences.
...	Further arguments passed to the different methods.

Details

The specified control is modified to ensure that (1) estimate.beta=FALSE and (2) nstart=1.

For "Gibbs_list" objects the control is further modified to have (1) iter=thin and (2) best=TRUE and the model is fitted to the new data with this control for each available iteration. The perplexity is then determined by averaging over the same number of iterations.

If a list is supplied as object, it is assumed that it consists of several models which were fitted using different starting configurations.

Value

A numeric value.

Author(s)

Bettina Gruen

References

Blei D.M., Ng A.Y., Jordan M.I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, **3**, 993–1022.

Griffiths T.L., Steyvers, M. (2004). Finding Scientific Topics. *Proceedings of the National Academy of Sciences of the United States of America*, **101**, Suppl. 1, 5228–5235.

Newman D., Asuncion A., Smyth P., Welling M. (2009). Distributed Algorithms for Topic Models. *Journal of Machine Learning Research*, **10**, 1801–1828.

---

posterior-methods   *Determine posterior probabilities*

Description

Determine the posterior probabilities of the topics for each document and of the terms for each topic for a fitted topic model.

Usage

```
## S4 method for signature 'TopicModel,missing'
posterior(object, newdata, ...)
## S4 method for signature 'TopicModel,ANY'
posterior(object, newdata, control = list(), ...)
```

Arguments

object	An object of class "TopicModel".
newdata	If missing the posteriors for the original observations are returned.
control	A named list of the control parameters for estimation or a suitable control object.
...	Currently not used.

Author(s)

Bettina Gruen

---

terms_and_topics   *Extract most likely terms or topics.*

Description

Function to extract the most likely terms for each topic or the most likely topics for each document.

Usage

```
## S4 method for signature 'TopicModel'
terms(x, k, threshold, ...)
## S4 method for signature 'TopicModel'
topics(x, k, threshold, ...)
```

Arguments

x	Object of class "TopicModel".
k	The maximum number of terms/topics returned. By default set to 1 if no threshold is given.
threshold	Only the terms/topics which are more likely than the threshold are returned.
...	Further arguments passed to sapply.

Value

A list or matrix containing the most likely terms for each topic or the most likely topics for each document.

Author(s)

Bettina Gruen

See Also

posterior-methods

TopicModel-class   *Virtual class "TopicModel"*

Description

Fitted topic model.

Objects from the Class

Objects of class "LDA" are returned by [LDA](LDA)() and of class "CTM" by [CTM](CTM)().

Slots

Class "TopicModel" contains

call: Object of class "call".

Dim: Object of class "integer"; number of documents and terms.

control: Object of class "TopicModelcontrol"; options used for estimating the topic model.

k: Object of class "integer"; number of topics.

terms: Vector containing the term names.

documents: Vector containing the document names.

beta: Object of class "matrix"; logarithmized parameters of the word distribution for each topic.

gamma: Object of class "matrix"; parameters of the posterior topic distribution for each document.

iter: Object of class "integer"; the number of iterations made.

logLiks: Object of class "numeric"; the vector of kept intermediate log-likelihood values of the corpus. See loglikelihood how the log-likelihood is determined.

n: Object of class "integer"; number of words in the data used.

wordassignments: Object of class "simple_triplet_matrix"; most probable topic for each observed word in each document.

Class "VEM" contains

loglikelihood: Object of class "numeric"; the log-likelihood of each document given the parameters for the topic distribution and for the word distribution of each topic is approximated using the variational parameters and underestimates the log-likelihood by the Kullback-Leibler divergence between the variational posterior probability and the true posterior probability.

Class "LDA" extends class "TopicModel" and has the additional slots

loglikelihood: Object of class "numeric"; the posterior likelihood of the corpus conditional on the topic assignments is returned.

alpha: Object of class "numeric"; parameter of the Dirichlet distribution for topics over documents.

Class "LDA_Gibbs" extends class "LDA" and has the additional slots

seed: Either NULL or object of class "simple_triplet_matrix"; parameter for the prior distribution of the word distribution for topics if seeded.

z: Object of class "integer"; topic assignments of words ordered by terms with suitable repetition within documents.

Class "CTM" extends class "TopicModel" and has the additional slots

mu: Object of class "numeric"; mean of the topic distribution on the logit scale.

Sigma: Object of class "matrix"; variance-covariance matrix of topics on the logit scale.

Class "CTM_VEM" extends classes "CTM" and "VEM" and has the additional slots

nusqared: Object of class "matrix"; variance of the variational distribution on the parameter mu.

Author(s)

Bettina Gruen

TopicModelcontrol-class

Different classes for controlling the estimation of topic models

---

Description

Classes to control the estimation of topic models which are inheriting from the virtual base class
`"TopicModelcontrol"`.

Objects from the Class

Objects can be created from named lists.

Slots

Class `"TopicModelcontrol"` contains

seed: Object of class `"integer"`; used to set the seed in the external code for VEM estimation and
    to call `set.seed` for Gibbs sampling. For Gibbs sampling it can also be set to NA (default) to
    avoid changing the seed of the random number generator in the model fitting call.

verbose: Object of class `"integer"`. If a positive integer, then the progress is reported every
    verbose iterations. If 0 (default), no output is generated during model fitting.

save: Object of class `"integer"`. If a positive integer the estimated model is saved all verbose
    iterations. If 0 (default), no output is generated during model fitting.

prefix: Object of class `"character"`; path indicating where to save the intermediate results.

nstart: Object of class `"integer"`. Number of repeated random starts.

best: Object of class `"logical"`; if TRUE only the model with the maximum (posterior) likelihood
    is returned, by default equals TRUE.

keep: Object of class `"integer"`; if a positive integer, the log-likelihood is saved every keep iter-
    ations.

estimate.beta: Object of class `"logical"`; controls if beta, the term distribution of the topics, is
    fixed, by default equals TRUE.

Class `"VEMcontrol"` contains

var: Object of class `"OPTcontrol"`; controls the variational inference for a single document, by
    default `iter.max` equals 500 and `tol` 10^{-6}.

em: Object of class `"OPTcontrol"`; controls the variational EM algorithm, by default `iter.max`
    equals 1000 and `tol` 10^{-4}.

initialize: Object of class `"character"`; one of `"random"`, `"seeded"` and `"model"`, by default
    equals `"random"`.

Class `"LDAcontrol"` extends class `"TopicModelcontrol"` and has the additional slots

alpha: Object of class `"numeric"`; initial value for alpha.

Class "LDA_VEMcontrol" extends classes "LDAcontrol" and "VEMcontrol" and has the additional slots

estimate.alpha: Object of class "logical"; indicates if the parameter alpha is fixed a-priori or estimated, by default equals TRUE.

Class "LDA_Gibbscontrol" extends classes "LDAcontrol" and has the additional slots

delta: Object of class "numeric"; initial value for delta, by default equals 0.1.

iter: Object of class "integer"; number of Gibbs iterations, by default equals 2000.

thin: Object of class "integer"; number of omitted in-between Gibbs iterations, by default equals iter.

burnin: Object of class "integer"; number of omitted Gibbs iterations at beginning, by default equals 0.

burnin: Object of class "integer"; number of omitted Gibbs iterations at beginning, by default equals 0.

initialize: Object of class "character"; one of "random", "beta" and "z", by default equals "random".

Class "CTM_VEMcontrol" extends classes "TopicModelcontrol" and "VEMcontrol" and has the additional slots

cg: Object of class "OPTcontrol"; controls the conjugate gradient iterations in fitting the variational mean and variance per document, by default iter.max equals 500 and tol 10^-5.

Class "OPTcontrol" contains

iter.max: Object of class "integer"; maximum number of iterations.

tol: Object of class "numeric"; tolerance for convergence check.

Author(s)

Bettina Gruen

Index