Package 'RTextTools'

August 19, 2011

Type Package

Title Automatic Text Classification via Supervised Learning

Version 1.3

Date 2011-08-22

Author Timothy P. Jurka, Loren Collingwood, Amber E. Boydstun, Emiliano Grossman, Wouter van Atteveldt

Maintainer Timothy P. Jurka <tpjurka@ucdavis.edu>

Depends R (>= 2.13.0), methods, SparseM, randomForest, tree, nnet, tm, e1071, ipred, caTools, maxent, RODBC, glmnet, Rstem

Description
RTextTools is a machine learning package for automatic text classification that makes it simple for novice users to get started with machine learning, while allowing experienced users to easily experiment with different settings and algorithm combinations. The package includes nine algorithms for ensemble classification (SVM, SLDA, BOOSTING, BAGGING, RF, GLMNET, TREE, NNET, MAXENT), comprehensive analytics, and thorough documentation.

License GPL-3

URL http://www.rtexttools.com/

LazyLoad yes

R topics documented:

`RTextTools-package` *RTextTools Machine Learning*

Description

RTextTools is a machine learning package for automatic text classification that makes it simple for novice users to get started with machine learning, while allowing experienced users to easily experiment with different settings and algorithm combinations. The package includes nine algorithms for ensemble classification (SVM, SLDA, BOOSTING, BAGGING, RF, GLMNET, TREE, NNET, MAXENT), comprehensive analytics, and thorough documentation.

Details

Package:	RTextTools
Type:	Package
Version:	1.3
Date:	2011-08-22
License:	GPL-3
LazyLoad:	yes

Using RTextTools can be broken down into five simple steps. First, read your data into R as a data frame using the included `read_data` function or any other method. Next, create the document term matrix from your textual documents using `create_matrix`, and create a container of these sparse matrices and labels with `create_corpus`. This object will then be input to both `train_model` and `classify_model`, which respectively train and classify the textual data. Alternatively, you may use `train_models` and `classify_models` to train and classify using multiple algorithms at once. You may use `print_algorithms` to see a list of available algorithms. Last, use `create_analytics` to analyze the results and determine accuracy rates as well as to prepare the ensemble agreement.

Author(s)

Timothy P. Jurka, Loren Collingwood, Amber E. Boydstun, Emiliano Grossman, Wouter van Atteveldt

Maintainer: <tpjurka@ucdavis.edu>

Examples

```
# LOAD THE RTextTools LIBARY
library(RTextTools)


# READ THE CSV DATA
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")


# [OPTIONAL] SUBSET YOUR DATA TO GET A RANDOM SAMPLE
data <- data[sample(1:3100,size=1000,replace=FALSE),]


# CREATE A TERM-DOCUMENT MATRIX THAT REPRESENTS WORD FREQUENCIES IN EACH DOCUMENT
# WE WILL TRAIN ON THE Title and Subject COLUMNS
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=TRUE, weighting=weightTfIdf)


# CREATE A CORPUS THAT IS SPLIT INTO A TRAINING SET AND A TESTING SET
# WE WILL BE USING Topic.Code AS THE CODE COLUMN. WE DEFINE A 750
# ARTICLE TRAINING SET AND A 250 ARTICLE TESTING SET.
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:750, testSize=751:1000,
virgin=FALSE)


# THERE ARE TWO METHODS OF TRAINING AND CLASSIFYING DATA.
# ONE WAY IS TO DO THEM AS A BATCH (SEVERAL ALGORITHMS AT ONCE)
models <- train_models(corpus, algorithms=c("GLMNET","MAXENT","SVM"))
results <- classify_models(corpus, models)


# ANOTHER WAY IS TO DO THEM ONE BY ONE.
glmnet_model <- train_model(corpus,"GLMNET")
maxent_model <- train_model(corpus,"MAXENT")
svm_model <- train_model(corpus,"SVM")

glmnet_results <- classify_model(corpus,glmnet_model)
maxent_results <- classify_model(corpus,maxent_model)
svm_results <- classify_model(corpus,svm_model)

# USE print_algorithms() TO SEE ALL AVAILABLE ALGORITHMS.
print_algorithms()


# VIEW THE RESULTS BY CREATING ANALYTICS
# IF YOU USED OPTION 1, YOU CAN GENERATE ANALYTICS USING
analytics <- create_analytics(corpus, results)

# IF YOU USED OPTION 2, YOU CAN GENERATE ANALYTICS USING:
analytics <- create_analytics(corpus,cbind(svm_results,maxent_results))

# RESULTS WILL BE REPORTED BACK IN THE analytics VARIABLE.
# analytics@algorithm_summary: SUMMARY OF PRECISION, RECALL, F-SCORES, AND
#  ACCURACY SORTED BY TOPIC CODE FOR EACH ALGORITHM
# analytics@label_summary: SUMMARY OF LABEL (e.g. TOPIC) ACCURACY
```

```
# analytics@document_summary: RAW SUMMARY OF ALL DATA AND SCORING
# analytics@ensemble_summary: SUMMARY OF ENSEMBLE PRECISION/COVERAGE.
#  USES THE n VARIABLE PASSED INTO create_analytics()

head(analytics@algorithm_summary)
head(analytics@label_summary)
head(analytics@document_summary)
head(analytics@ensemble_summary)

# WRITE OUT THE DATA TO A CSV
write.csv(analytics@algorithm_summary,"SampleData_AlgorithmSummary.csv")
write.csv(analytics@label_summary,"SampleData_LabelSummary.csv")
write.csv(analytics@document_summary,"SampleData_DocumentSummary.csv")
write.csv(analytics@ensemble_summary,"SampleData_EnsembleSummary.csv")
```

`analytics_container-class`

an S4 class containing the analytics for a classified set of documents.

Description

An `S4 class` containing the analytics for a classified set of documents. This includes a label summary, document summary, ensemble summary, and algorithm summary. This class is returned if `virgin=FALSE` in `create_corpus`.

Objects from the Class

Objects could in principle be created by calls of the form `new("analytics_container", ...)`. The preferred form is to have them created via a call to `create_analytics`.

Slots

`label_summary` Object of class `"data.frame"`: stores the analytics for each label, including the percent coded accurately and how much overcoding occurred

`document_summary` Object of class `"data.frame"`: stores the analytics for each document, including all available raw data associated with the learning process

`algorithm_summary` Object of class `"data.frame"`: stores precision, recall, and F-score statistics for each algorithm, broken down by label

`ensemble_summary` Object of class `"matrix"`: stores the accuracy and coverage for an n-algorithm ensemble scoring

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
analytics <- create_analytics(corpus, results)

analytics@label_summary
analytics@document_summary
analytics@algorithm_summary
analytics@ensemble_summary
```

---

analytics_container_virgin-class

an S4 class containing the analytics for a classified set of documents.

Description

An `S4 class` containing the analytics for a classified set of documents. This includes a label summary and a document summary. This class is returned if `virgin=TRUE` in `create_corpus`.

Objects from the Class

Objects could in principle be created by calls of the form `new("analytics_container", ...)`. The preferred form is to have them created via a call to `create_analytics`.

Slots

`label_summary` Object of class `"data.frame"`: stores the analytics for each label, including how many documents were classified with each label

`document_summary` Object of class `"data.frame"`: stores the analytics for each document, including all available raw data associated with the learning process

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=TRUE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
```

```
analytics <- create_analytics(corpus, results)

analytics@label_summary
analytics@document_summary
```

`classify_model` *makes predictions from a train_model() object.*

Description

Uses a trained model from the `train_model` function to classify new data.

Usage

```
classify_model(corpus, model, s=0.01, ...)
```

Arguments

`corpus`	Class of type `matrix_container-class` generated by the `create_corpus` function.
`model`	Slot for trained SVM, SLDA, boosting, bagging, RandomForests, glmnet, decision tree, neural network, or maximum entropy model generated by `train_model`.
`s`	Penalty parameter lambda for **glmnet** classification.
`...`	Additional parameters to be passed into the `predict` function of any algorithm.

Details

Only one model may be passed in at a time for classification. See `train_models` and `classify_models` to train and classify using multiple algorithms.

Value

Returns a `data.frame` of predicted codes and probabilities for the specified algorithm.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
maxent_model <- train_model(corpus,"MAXENT")
svm_model <- train_model(corpus,"SVM")
maxent_results <- classify_model(corpus,maxent_model)
svm_results <- classify_model(corpus,svm_model)
```

classify_models *makes predictions from a train_models() object.*

Description

Uses a trained model from the train_models function to classify new data.

Usage

```
classify_models(corpus, models, ...)
```

Arguments

corpus	Class of type matrix_container-class generated by the create_corpus function.
models	List of models to be used for classification generated by train_models.
...	Other parameters to be passed on to classify_model.

Details

Use the list returned by train_models to use multiple models for classification.

Author(s)

Wouter Van Atteveldt <wouter@vanatteveldt.com>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
```

create_analytics *creates an object of class analytics given classification results.*

Description

Takes the results from functions classify_model or classify_models and computes various statistics to help interpret the data.

Usage

```
create_analytics(corpus, classification_results, b=1, threshold=NULL)
```

Arguments

corpus	Class of type matrix_container-class generated by the create_corpus function.
classification_results	A cbind() of result objects returned by classify_model, or the object returned by classify_models.
b	b-value for generating precision, recall, and F-scores statistics.
threshold	The number of algorithms greater than or equal to this threshold that agree on the same topic. For example, a threshold value of 3 will search for those documents where 3 or more algorithms agreed.

Value

Object of class analytics_container_virgin-class or analytics_container-class has either two or four slots respectively, depending on whether the virgin flag is set to TRUE or FALSE in create_corpus. They can be accessed using the @ operator for S4 classes (e.g. analytics@document_summary).

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Loren Collingwood <lorenc2@uw.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
analytics <- create_analytics(corpus, results)
```

---

create_corpus *creates a corpus for training, classifying, and analyzing documents.*

Description

Given a DocumentTermMatrix from the **tm** package and corresponding document labels, creates a corpus of class matrix_container-class that can be used for training and classification (i.e. train_model, train_models, classify_model, classify_models)

Usage

```
create_corpus(matrix, labels, trainSize, testSize, virgin)
```

Arguments

matrix	A document-term matrix of class DocumentTermMatrix or TermDocumentMatrix from the **tm** package, or generated by create_matrix.
labels	A factor or vector of labels corresponding to each document in the matrix.
trainSize	A range (e.g. 1:1000) specifying the number of documents to use for training the models.
testSize	A range (e.g. 1:1000) specifying the number of documents to use for classification.
virgin	A logical (TRUE or FALSE) specifying whether to treat the classification data as virgin data or not.

Value

A corpus of class matrix_container-class that can be passed into other functions such as train_model, train_models, classify_model, classify_models, wizard_train_classify, and create_analytics.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Loren Collingwood <lorenc2@uw.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
```

---

create_ensembleSummary

creates a summary with ensemble coverage and precision.

Description

Creates a summary with ensemble coverage and precision values for an ensemble greater than the threshold specified.

Usage

```
create_ensembleSummary(document_summary, threshold)
```

Arguments

document_summary	The document_summary slot from the analytics_container-class generated by create_analytics.
threshold	The number of algorithms greater than or equal to this threshold that agree on the same topic. For example, a threshold value of 3 will search for those documents where 3 or more algorithms agreed.

Details

This summary is created in the create_analytics function. Note that a threshold value of 3 will return ensemble coverage and precision statistics for topic codes that had 3 or more (i.e. >=3) algorithms agree on the same topic code.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
analytics <- create_analytics(corpus, results)
ensemble <- create_ensembleSummary(analytics@document_summary,2)
ensemble
```

---

create_matrix *creates a document-term matrix to be passed into create_corpus().*

Description

Creates an object of class DocumentTermMatrix from **tm** that can be used in the create_corpus function.

Usage

```
create_matrix(textColumns, language = "en", minDocFreq = 1,
minWordLength = 3, removeNumbers = FALSE, removePunctuation = TRUE,
removeSparseTerms = 0, removeStopwords = TRUE, selectFreqTerms = 0,
stemWords = FALSE, stripWhitespace = TRUE, toLower = TRUE,
weighting = weightTf)
```

Arguments

textColumns	Either character vector (e.g. data$Title) or a cbind() of columns to use for training the algorithms (e.g. cbind(data$Title,data$Subject)).
language	The language to be used for stemming the text data.
minDocFreq	The minimum number of times a word should appear in a document for it to be included in the matrix. See package **tm** for more details.
minWordLength	The minimum number of letters a word should contain to be included in the matrix. See package **tm** for more details.

removeNumbers
: A logical parameter to specify whether to remove numbers.

removePunctuation
: A logical parameter to specify whether to remove punctuation.

removeSparseTerms
: See package **tm** for more details.

removeStopwords
: A logical parameter to specify whether to remove stopwords using the language specified in language.

selectFreqTerms
: Select the N most frequent terms in each document to use for training.

stemWords
: A logical parameter to specify whether to stem words using the language specified in language.

stripWhitespace
: A logical parameter to specify whether to strip whitespace.

toLower
: A logical parameter to specify whether to make all text lowercase.

weighting
: Either weightTf or weightTfIdf. See package **tm** for more details.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Loren Collingwood <lorenc2@uw.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
```

---

create_precisionRecallSummary

creates a summary with precision, recall, and F1 scores.

Description

Creates a summary with precision, recall, and F1 scores for each algorithm broken down by unique label.

Usage

```
create_precisionRecallSummary(corpus, classification_results, b_value = 1)
```

Arguments

corpus
: Class of type matrix_container-class generated by the create_corpus function.

classification_results
: A cbind() of result objects returned by classify_model, or the object returned by classify_models.

b_value
: b-value for generating precision, recall, and F-scores statistics.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
precision_recall_f1 <- create_precisionRecallSummary(corpus, results)
```

---

create_scoreSummary

creates a summary with the best label for each document.

Description

Creates a summary with the best label for each document, determined by highest algorithm certainty, and highest consensus (i.e. most number of algorithms agreed).

Usage

```
create_scoreSummary(corpus, classification_results)
```

Arguments

corpus	Class of type matrix_container-class generated by the create_corpus function.
classification_results	A cbind() of result objects returned by classify_model, or the object returned by classify_models.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Loren Collingwood <lorenc2@uw.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
score_summary <- create_scoreSummary(corpus, results)
```

`cross_validate` *used for cross-validation of various algorithms.*

Description

Performs n-fold cross-validation of specified algorithm.

Usage

```
cross_validate(corpus, nfold, algorithm = c("SVM", "SLDA", "BOOSTING",
"BAGGING", "RF", "GLMNET", "TREE", "NNET", "MAXENT"), seed = NA,
method = "C-classification", cross = 0, cost = 100, kernel = "radial",
maxitboost = 100, maxitglm = 500, size = 1, maxitnnet = 1000, MaxNWts = 10000,
rang = 0.1, decay = 5e-04, ntree = 200, feature_cutoff = 0, gaussian_prior = 0,
inequality_constraints = 0)
```

Arguments

`corpus`	Class of type `matrix_container-class` generated by the `create_corpus` function.
`nfold`	Number of folds to perform for cross-validation.
`algorithm`	A string specifying which algorithm to use. Use `print_algorithms` to see a list of options.
`seed`	Random seed number used to replicate cross-validation results.
`method`	Method parameter for SVM implentation. See **e1071** documentation for more details.
`cross`	Cross parameter for SVM implentation. See **e1071** documentation for more details.
`cost`	Cost parameter for SVM implentation. See **e1071** documentation for more details.
`kernel`	Kernel parameter for SVM implentation. See **e1071** documentation for more details.
`maxitboost`	Maximum iterations parameter for boosting implentation. See **caTools** documentation for more details.
`maxitglm`	Maximum iterations parameter for glmnet implentation. See **glmnet** documentation for more details.
`size`	Size parameter for neural networks implentation. See **nnet** documentation for more details.
`maxitnnet`	Maximum iterations for neural networks implentation. See **nnet** documentation for more details.
`MaxNWts`	Maximum number of weights parameter for neural networks implentation. See **nnet** documentation for more details.
`rang`	Range parameter for neural networks implentation. See **nnet** documentation for more details.
`decay`	Decay parameter for neural networks implentation. See **nnet** documentation for more details.

`ntree` Number of trees parameter for RandomForests implentation. See **randomForest** documentation for more details.

`feature_cutoff` Feature cutoff parameter for maximum entropy implementation. See **maxent** documentation for more details.

`gaussian_prior` Guassian prior parameter for maximum entropy implementation. See **maxent** documentation for more details.

`inequality_constraints` Inequality constraints parameter for maximum entropy implementation. See **maxent** documentation for more details.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
svm <- cross_validate(corpus,2,algorithm="SVM")
maxent <- cross_validate(corpus,2,algorithm="MAXENT")
```

---

matrix_container-class

an S4 class containing the training and classification matrices.

Description

An S4 class containing all information necessary to train, classify, and generate analytics for a dataset.

Objects from the Class

Objects could in principle be created by calls of the form `new("matrix_container", ...)`. The preferred form is to have them created via a call to `create_corpus`.

Slots

`training_matrix` Object of class `"matrix.csr"`: stores the training set of the `DocumentTermMatrix` created by `create_matrix`

`training_codes` Object of class `"factor"`: stores the training labels for each document in the `training_matrix` slot of `matrix_container-class`

`classification_matrix` Object of class `"matrix.csr"`: stores the classification set of the `DocumentTermMatrix` created by `create_matrix`

`testing_codes` Object of class `"factor"`: if `virgin=FALSE`, stores the labels for each document in classification_matrix

column_names Object of class "vector": stores the column names of the DocumentTermMatrix created by create_matrix

virgin Object of class "logical": boolean specifying whether the classification set is virgin data (TRUE) or not (FALSE).

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)

corpus@training_matrix
corpus@training_codes
corpus@classification_matrix
corpus@testing_codes
corpus@column_names
corpus@virgin
```

NYTimes *a sample dataset containing labeled headlines from The New York Times.*

Description

A sample dataset containing labeled headlines from The New York Times, compiled by Professor Amber E. Boydstun at the University of California, Davis.

Usage

```
data(NYTimes)
```

Format

A data.frame containing five columns.

1. Article_ID - A unique identifier for the headline from The New York Times.
2. Date - The date the headline appeared in The New York Times.
3. Title - The headline as it appeared in The New York Times.
4. Subject - A manually classified subject of the headline.
5. Topic.Code - A manually labeled topic code corresponding to the subject.

Source

http://www.amberboydstun.com/

Examples

```
# READ THE CSV
data <- read.csv(system.file("data/NYTimes.csv.gz",package="RTextTools"))
# ALTERNATIVELY, USE THE data() FUNCTION
data(NYTimes)
```

---

print_algorithms *prints available algorithms for train_model() and train_models().*

Description

An informative function that displays options for the `algorithms` parameter in `train_model` and `train_models`.

Usage

```
print_algorithms()
```

Value

Prints a list of available algorithms.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
print_algorithms()
```

---

read_data *reads data from files into an R data frame.*

Description

Reads data from several types of data storage types into an R data frame.

Usage

```
read_data(filename, tablename = NULL, type = c("csv", "tab", "accdb", "mdb"),
...)
```

Arguments

`filename`	Character string of the name of the file, include path if the file is not located in the working directory.
`tablename`	Microsoft Access database only. The table name in the database.
`type`	Character vector specifying the file type. Options include `"csv"`, `"tab"`, `"accdb"`, `"mdb"` to denote .csv files, text files, or Access databases.
`...`	Other arguments passed to read_data.

Value

An `data.frame` object is returned with the contents of the file.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
```

---

recall_accuracy *calculates the recall accuracy of the classified data.*

Description

Given the true labels to compare to the labels predicted by the algorithms, calculates the recall accuracy of each algorithm.

Usage

```
recall_accuracy(true_labels, predicted_labels)
```

Arguments

`true_labels`	A vector containing the true labels, or known values for each document in the classification set.
`predicted_labels`	A vector containing the predicted labels, or classified values for each document in the classification set.

Author(s)

Loren Collingwood <lorenc2@uw.edu>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
results <- classify_models(corpus, models)
analytics <- create_analytics(corpus, results)
recall_accuracy(analytics@document_summary$MANUAL_CODE,
analytics@document_summary$GLMNET_LABEL)
recall_accuracy(analytics@document_summary$MANUAL_CODE,
analytics@document_summary$MAXENTROPY_LABEL)
recall_accuracy(analytics@document_summary$MANUAL_CODE,
analytics@document_summary$SVM_LABEL)
```

`train_model` *makes a model object using the specified algorithm.*

Description

Creates a trained model using the specified algorithm.

Usage

```
train_model(corpus, algorithm=c("SVM","SLDA","BOOSTING","BAGGING",
"RF","GLMNET","TREE","NNET","MAXENT"), method = "C-classification",
cross = 0, cost = 100, kernel = "radial", maxitboost = 100,
maxitglm = 10^5, size = 1, maxitnnet = 1000, MaxNWts = 10000,
rang = 0.1, decay = 5e-04, trace=FALSE, ntree = 200,
feature_cutoff = 0, gaussian_prior = 0, inequality_constraints = 0,
...)
```

Arguments

`corpus`	Class of type `matrix_container-class` generated by the `create_corpus` function.
`algorithm`	Character vector (i.e. a string) specifying which algorithm to use. Use `print_algorithms` to see a list of options.
`method`	Method parameter for SVM implentation. See **e1071** documentation for more details.
`cross`	Cross parameter for SVM implentation. See **e1071** documentation for more details.
`cost`	Cost parameter for SVM implentation. See **e1071** documentation for more details.
`kernel`	Kernel parameter for SVM implentation. See **e1071** documentation for more details.
`maxitboost`	Maximum iterations parameter for boosting implentation. See **caTools** documentation for more details.
`maxitglm`	Maximum iterations parameter for glmnet implentation. See **glmnet** documentation for more details.
`size`	Size parameter for neural networks implentation. See **nnet** documentation for more details.
`maxitnnet`	Maximum iterations for neural networks implentation. See **nnet** documentation for more details.
`MaxNWts`	Maximum number of weights parameter for neural networks implentation. See **nnet** documentation for more details.
`rang`	Range parameter for neural networks implentation. See **nnet** documentation for more details.
`decay`	Decay parameter for neural networks implentation. See **nnet** documentation for more details.
`trace`	Trace parameter for neural networks implentation. See **nnet** documentation for more details.

`ntree` Number of trees parameter for RandomForests implentation. See **randomForest** documentation for more details.

`feature_cutoff` Feature cutoff parameter for maximum entropy implementation. See **maxent** documentation for more details.

`gaussian_prior` Guassian prior parameter for maximum entropy implementation. See **maxent** documentation for more details.

`inequality_constraints` Inequality constraints parameter for maximum entropy implementation. See **maxent** documentation for more details.

`...` Additional arguments to be passed on to algorithm function calls.

Details

Only one algorithm may be selected for training. See `train_models` and `classify_models` to train and classify using multiple algorithms.

Value

Returns a trained model that can be subsequently used in `classify_model` to classify new data.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Loren Collingwood <lorenc2@uw.edu>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
maxent_model <- train_model(corpus,"MAXENT")
svm_model <- train_model(corpus,"SVM")
```

`train_models` *makes a model object using the specified algorithms.*

Description

Creates a trained model using the specified algorithms.

Usage

```
train_models(corpus, algorithms, ...)
```

Arguments

corpus	Class of type matrix_container-class generated by the create_corpus function.
algorithms	List of algorithms as a character vector (e.g. c("SVM","MAXENT")).
...	Other parameters to be passed on to train_model.

Details

Calls the train_model function for each algorithm you list.

Value

Returns a list of trained models that can be subsequently used in classify_models to classify new data.

Author(s)

Wouter Van Atteveldt <wouter@vanatteveldt.com>

Examples

```
library(RTextTools)
data <- read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),type="csv")
data <- data[sample(1:3100,size=100,replace=FALSE),]
matrix <- create_matrix(cbind(data$Title,data$Subject), language="english",
removeNumbers=TRUE, stemWords=FALSE, weighting=weightTfIdf)
corpus <- create_corpus(matrix,data$Topic.Code,trainSize=1:75, testSize=76:100,
virgin=FALSE)
models <- train_models(corpus, algorithms=c("MAXENT","SVM"))
```

USCongress — *a sample dataset containing labeled bills from the United State Congress.*

Description

A sample dataset containing labeled bills from the United States Congress, compiled by Professor John D. Wilkerson at the University of Washington, Seattle and E. Scott Adler at the University of Colorado, Boulder.

Usage

```
data(USCongress)
```

Format

A `data.frame` containing five columns.

1. `ID` - A unique identifier for the bill.
2. `cong` - The session of congress that the bill first appeared in.
3. `billnum` - The number of the bill as it appears in the congressional docket.
4. `h_or_sen` - A field specifying whether the bill was introduced in the House (HR) or the Senate (S).
5. `major` - A manually labeled topic code corresponding to the subject of the bill.

Source

http://www.congressionalbills.org/

Examples

```
# READ THE CSV
data <- read.csv(system.file("data/USCongress.csv.gz",package="RTextTools"))
# ALTERNATIVELY, USE THE data() FUNCTION
data(USCongress)
```

---

wizard_read_data *a simplified function for reading data from files.*

Description

A simple interface for reading in data from files and creating a corpus all in one step.

Usage

```
wizard_read_data(filename, tablename = NULL, filetype = "csv",
virgin=FALSE, textColumns, codeColumn, trainSize, testSize, ...)
```

Arguments

`filename`	Character string of the name of the file, include path if the file is not located in the working directory.
`tablename`	Microsoft Access database only. The table name in the database.
`filetype`	Character vector specifying the file type. Options include `"csv"`, `"tab"`, `"accdb"`, `"mdb"` to denote .csv files, text files, or Access databases.
`virgin`	A logical (`TRUE` or `FALSE`) specifying whether to treat the classification data as virgin data or not. Defaults to `FALSE`, specifying that classification data is not virgin data.
`textColumns`	The a `cbind()` of column(s) to use for training the algorithms (e.g. `cbind(data$Title)`).
`codeColumn`	A factor or vector of labels corresponding to each document in the matrix.
`trainSize`	A range (e.g. `1:1000`) specifying the number of documents to use for training the models.
`testSize`	A range (e.g. `1001:2000`) specifying the number of documents to use for classification.
`...`	Other parameters to be passed on to `create_matrix`.

Value

A corpus of class matrix_container-class that can be passed into other functions such as train_model, train_models, classify_model, classify_models, wizard_train_classify, and create_analytics.

Author(s)

Wouter Van Atteveldt <wouter@vanatteveldt.com>, Timothy P. Jurka <tpjurka@ucdavis.edu>

Examples

```
library(RTextTools)
corpus <- wizard_read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),
textColumns=c("Title","Subject"), codeColumn="Topic.Code", trainSize=75,
testSize=25, virgin=FALSE)
```

---

wizard_train_classify

a simplified function for training and classifying data.

Description

A simple interface for training and classifying data using the internal train_model and classify_model commands, and returning a results data.frame ready for use in create_analytics.

Usage

```
wizard_train_classify(corpus, algorithms, ...)
```

Arguments

corpus	Class of type matrix_container-class generated by the create_corpus function.
algorithms	List of algorithms as a character vector (e.g. c("SVM","MAXENT")).
...	Other parameters to be passed on to train_model.

Value

A data.frame containing the results of the classification. Pass into create_analytics to generate detailed analytics.

Author(s)

Timothy P. Jurka <tpjurka@ucdavis.edu>, Wouter Van Atteveldt <wouter@vanatteveldt.com>

Examples

```
library(RTextTools)
corpus <- wizard_read_data(system.file("data/NYTimes.csv.gz",package="RTextTools"),
textColumns=c("Title","Subject"), codeColumn="Topic.Code", trainSize=75,
testSize=25, virgin=FALSE)
results <- wizard_train_classify(corpus, c("SVM", "MAXENT"))
```

Index