# Package 'TextWiller'

October 28, 2015

**Type** Package

**Title** Collection of functions for text mining, specially devoted to the italian language

**Version** 1.0

**Date** 2013-12-19

**Author**
(associazionerospo.org) Dario Solari, Andrea Sciandra, Marco Rinaldo, Matteo Redaelli, Livio Finos (con contributi di Marco Rinaldo, Maddalena Branca, Federico Ferraccioli).

**Maintainer** dario solari <dario.solari@gmail.com>

**Depends** R (>= 2.10), stringr, twitteR, RCurl, SnowballC

**Description** Collection of functions for text mining, specially devoted to the italian language.

**License** GPL (>= 2)

## R topics documented:



---

`TextWiller-package` *Collection of functions for text mining, specially devoted to the italian language*

---

### Description

Collection of functions for text mining, specially devoted to the italian language

**Details**

| | |
|---|---|
| Package: | TextWiller |
| Type: | Package |
| Version: | 1.0 |
| Date: | 2013-12-19 |
| License: | GPL (>= 2) |

## Author(s)

Dario Solari, Andrea Sciandra, Marco Rinaldo, Matteo Redaelli, Livio Finos (con contributi di Marco Rinaldo, Maddalena Branca, Federico Ferraccioli).

Maintainer: dario solari <dario.solari@gmail.com>

## Examples

```
 ## Not run: # install.packages("devtools") # if you don't already have it.
library(devtools)
install_github("TextWiller", "livioivil")
library(TextWiller)

## End(Not run)

normalizzaTesti(c('ciao bella!','www.associazionerospo.org','noooo, che grandeeeeee!!!!!','mitticooo', 'mai pos

sentiment(c("ciao bella!","farabutto!","fofi sei figo!"))

classificaUtenti(c('livio','alessandra'))

#extract short urls and get the long ones
## Not run: urls=urlExtract("Influenza Vaccination | ONS - Oncology Nursing Society http://t.co/924sRKGBU9 See Al

#extract users:
## Not run: extract("@livio: #ciao","@\w+")
```

---

`classificaUtenti` *Associa i nomi in names ai valori indicati da vocabolarioNomiPropri*

---

## Description

Associa i nomi in names ai valori indicati da un vocabolario. ad esempio vocabolarioNomiPropri assegna il genere e `data(vocabolarioLuoghi)` l'area geografica (vedi esempio)

## Usage

```
classificaUtenti(names, vocabolario = NULL, ifManyUseFirst = TRUE, NAasExtraLevel = FALSE)
```

### Arguments

| | |
|---|---|
| names | vettore di nomi |
| vocabolario | data.frame di una colonna con la classificazione da associare. I rownames(vocabolario) devono essere unici (sono i nomi unici su cui viene fatto il controllo). il vocabolario fornito da noi e' data(vocabolarioNomiPropri). ATTENZIONE, nel vocabolario usare solo lower-case e non usare mai "NA" (mentre "na" e' valido). |
| ifManyUseFirst | TRUE by default. Nel caso di molteplici classificazioni, assegna alla prima categoria di unique(vocabolario$categoria). |
| NAasExtraLevel | gli NA diventano una categoria a parte. |

### Details

vedi esempio sotto.

Per il data(vocabolarioLuoghi) abbiamo escluso i paesi Re (800 abitanti, Nord-ovest) e Lu (1200 abitanti, Nord-ovest) perche' in conflitto con le sigle delle province.

### Value

caccia fuori un named vector con elementi dalla colonna categoria del data.frame vocabolario. Per vocabolario=vocabolarioNomiPropri le modalita' sono c('masc','femm','ente')

### Author(s)

Livio, Andrea Mamprin, Dario Solari

### Examples

```
 ## Not run: data(vocabolarioNomiPropri)
 ## Not run: str(vocabolarioNomiPropri)
classificaUtenti(c('livio','alessandra'))
data(vocabolarioLuoghi)
classificaUtenti(c('Bosa','Pordenone, Italy'),vocabolarioLuoghi)
```

---

## extract *Estrazione di regular expression (e quindi users, hashtag) e shorturl*

---

### Description

patternExtract estrae i pattern contenuti in in testo. urlExtract estrae e converte gli shorturl contenuti in testo in url. shorturl2url sostituisce gli shorturl contenuti in testo in url.

### Usage

```
shorturl2url(testo,id=names(testo))
urlExtract(testo, id = names(testo))
patternExtract(testo, pattern="@( *)\\w+", id = names(testo))
```

### Arguments

| | |
|---|---|
| testo | Vettore (eventualmente con nomi) di testi contenenti shorturl. |
| pattern | stringa di testo da cercare ed estrarre. "@\\w+" (default) estrae i riferimenti ad uno user nei tweets. "#\\w+" estrae gli hashtag. |
| id | se testo e' un vettore con nomi, questi vengono presi come id. In caso contrario, gli id sono numeri progressivi da 1 a length(testo) |

### Value

patternExtract restituisce un data.frame con colonne:id, pattern

urlExtract restituisce un data.frame con colonne:id, shorturl e url

### Author(s)

Dario Solari, Livio Finos

### Examples

```
 ## Not run:
 testo=c("Influenza Vaccination | ONS - Oncology Nursing Society http://t.co/924sRKGBU9 See All http://t.co/dbtPJ
 shorturl2url(testo,id=names(testo))
urls=urlExtract(testo)
patternExtract(c("@luca @paolo: buon giorno!", "@matteo: a te!"), pattern="@\w+")

## End(Not run)
```

---

## normalizzaTesti *Varie funzioni di normalizzazione del testo*

---

### Description

Varie funzioni di normalizzazione del testo

### Usage

```
normalizzaTesti(testo, tolower = TRUE, normalizzahtml = TRUE,
  normalizzacaratteri = TRUE, normalizzaemote = TRUE,
  normalizzapunteggiatura = TRUE, normalizzaslang = TRUE,
  fixed = TRUE, perl = TRUE,
  preprocessingEncoding=TRUE, encoding="UTF-8", sub = "",
  contaStringhe = c("\\?","\\!","@","#",
  "(\u20AC|euro)","(\$|dollar)", "SUPPRESSEDTEXT"),
  suppressInvalidTexts=TRUE,
  verbatim=TRUE, remove = TRUE)

preprocessingEncoding(testo, encoding="UTF-8",
                          suppressInvalidTexts=TRUE, verbatim = TRUE, sub = "")
```

```
normalizzacaratteri(testo, fixed = TRUE)
normalizzapunteggiatura(testo, removeUnderscore = TRUE, perl = TRUE, fixed = TRUE)
normalizzaslang(testo, perl = TRUE)
normalizzahtml(testo, perl = TRUE, fixed = TRUE)
normalizzaemote(testo, perl = TRUE)
tryTolower(testo, ifErrorReturnText = FALSE)
removeStopwords(testo, stopwords = itastopwords)
data(itastopwords)
```

## Arguments

| | |
|---|---|
| testo | character vector of texts |
| tolower | TRUE by default |
| normalizzahtml | TRUE by default |
| normalizzacaratteri | TRUE by default |
| normalizzaemote | TRUE by default |
| normalizzapunteggiatura | TRUE by default |
| normalizzaslang | TRUE by default |
| fixed | vedi base:gsub. Preferibilmente non usare l'opzione. |
| perl | vedi base:gsub. Preferibilmente non usare l'opzione. |
| preprocessingEncoding | logical |
| encoding | "UTF-8" default. Se FALSE evita la conversione. |
| sub | character string. If not NA it is used to replace any non-convertible bytes in the input. See also parameter sub in function iconv. |
| contaStringhe | stringhe da contare nei documenti. Default: c("\\?","\\!","#","@", "(\|euro)","(\$\|dollar)","$ |
| suppressInvalidTexts | Sostituisce con "SUPPRESSEDTEXT" le stringhe con mutibyte non valida (che produrrebbero verosimilmente errori nelle successive normalizzazioni). Default TRUE. |
| removeUnderscore | rimuovere gli underscore? |
| ifErrorReturnText | what to return for tests with a wrong encoding. |
| stopwords | Lista di parole da escludere dall'analisi. A list of words to be excluded from the process. itastopwords by default. |
| verbatim | Mostra statitiche durante il processo. Default TRUE |
| remove | TRUE by default |

### Details

itastopwords e' una lista di stopwords italiane.

### Value

Per normalizzaTesti l'output e' il vettore di testi normalizzati. La tabella dei conteggi specificati in contaStringhe e' assegnato come tabella counts tra gli attributes del vettore stesso.

Per tutte le altre funzioni, l'output e' un vector della stessa lunghezza di testo ma con testi normalizzati.

### Author(s)

Dario Solari, Livio Finos, Maddalena Branca

### Examples

```
testoNorm <- normalizzaTesti(c('ciao bella!','www.associazionerospo.org','noooo, che grandeeeeee!!!!!','mittico
testoNorm
attr(testoNorm,"counts")
```

---

RTHound *RTHound*

---

### Description

Identifies the most frequent retweets through hierarchical clustering on Levenshtein distance (dissimilarity) matrix.

### Usage

```
RTHound(testo, S = 500, L = 100,
                hclust.dist = 100, hclust.method = "complete",
                showTopN=5, dist="levenshtein", verbatim=TRUE)
```

### Arguments

| | |
|---|---|
| testo | Tweets or generic texts vector. |
| S | Number of tweets (or texts) for each subset. 500 by deafault. |
| L | Number of tweets (or texts) belonging to the previous subset to embed in subset analysis. 100 by default. |
| hclust.dist | Numeric scalar with height where the trees should be cut. 100 by deafault. |
| hclust.method | The agglomeration method to be used. This should be (an unambiguous abbreviation of) one of "ward", "single", "complete", "average", "mcquitty", "median" or "centroid". "complete" by default. |
| showTopN | Number of most frequent retweets to show. 5 by deafault. |
| dist | "levenshtein" is the default. "profile" is the other - quicker - accepted value. |
| verbatim | logical |

## Details

RTHound divides testo in subsets of length S (from the second subset also incorporates L tweets of the previous subset); calculate a dissimilarity matrix based on Levenshtein distance for each subsets and clusterize tweets throught hierarchical clustering algorithm.

## Value

RTHound replaces the tweets belong to the same cluster with the oldest, identifying them as retweets, and returns a list of the most frequent retweets (top).

## Author(s)

Federico Ferraccioli, Livio Finos

## See Also

hclust

## Examples

```
## Not run:
testo=c(
"RT @LAVonlus: Tre miti da sfatare sulla #vivisezione. Le risposte  ai luoghi comuni della sperimentazione animal
"Tre miti da sfatare sulla #vivisezione. Le risposte  ai luoghi comuni della sperimentazione animale  http://t.co
"RT @LAVonlus: Tre miti da sfatare sulla #vivisezione. Le risposte  ai luoghi comuni della sperimentazione animal
"RT @orianoPER: La #sperimentazioneanimale è inutile perché non predittiva per la specie umana. MEDICI ANTI #VIVI
"La #sperimentazioneanimale è inutile perché non predittiva per la specie umana. MEDICI ANTI #VIVISEZIONE- LIMAV
"RT @orianoPER: La #ricerca in #Medicina con #sperimentazioneanimale non e' predittiva per la specie umana. MEDIC
"RT @HuffPostItalia: Il Governo italiano non fermi la sperimentazione animale. Intervista a Elena Cattaneo http:
"RT @HuffPostItalia: \"Il Governo italiano non fermi la sperimentazione animale\". Intervista a Elena Cattaneo h
"\"Il Governo italiano non fermi la sperimentazione animale\". Intervista a Elena Cattaneo http://t.co/q1dm430a9
"RT @orianoPER: @EnricoLetta LA #VIVISEZIONE NON SERVE: PAROLA DI GLAXO-APTUIT http://t.co/mtsHJjDIvu #StopVivis

testo=RTHound(testo, S = 3, L = 1,
                hclust.dist = 100, hclust.method = "complete",
                showTopN=3)


## End(Not run)
```

---

# sentiment *Performs sentiment analysis*

---

## Description

Assegna una sentiment per ogni testo in text

### Usage

```
sentiment(text, algorithm="Maddalena",
          vocabularies= NULL, ...)
data(vocabolariMadda)
```

### Arguments

| | |
|---|---|
| text | descr . |
| algorithm | descr . |
| vocabularies | vocabolariMadda by default. |

### Details

aggiungere dettagli qui

### Value

l'output etc

### Author(s)

Maddalena Branca, Livio Finos

### Examples

```
sentiment(c("ciao bella", "ciao", "good","casa", "farabutto!"))
```

---

## TimeStamp *Funzioni di gestione delle date*

---

### Description

Funzioni di gestione delle date

### Usage

```
fixTimeStamp(db, campoData = "created", timeRange = NULL)
selezionaIntervalloTimeStamp(db, timeRange = range(db$created) , campoData = "created")
```

### Arguments

| | |
|---|---|
| db | data.frame contenente i tweets. |
| timeRange | due valori di tipo data indicanti inizio e fine. |
| campoData | "created" e "ts" sono due campi data del db estratto da dump_twitter.R |

## Details

aggiungere dettagli qui

## Value

l'output e' db "aggiustato"

## Author(s)

Dario Solari, Livio Finos

## Examples

```
## Not run: TW=fixTimeStamp(TW)
## Not run: TW=selezionaIntervallo(TW,as.POSIXct(c("2013-12-27 17:54:42 CET", "2013-12-27 22:33:38 CET")))
## Not run: TW$created.round <- as.POSIXct(round(t$created,"hour"))
```

# Index