Package 'edgar'

June 4, 2017

Type Package

Title Platform for EDGAR Filing Management

Version 1.0.9

Date 2017-06-04

Author Gunratan Lonare <lonare.gunratan@gmail.com>, Bharat Patil <bharatspatil@gmail.com>

Maintainer Gunratan Lonare <lonare.gunratan@gmail.com>

Depends R (>= 3.1)

Imports R.utils,tm,XML,wordcloud,RColorBrewer,ggplot2

Description In the USA, firms file different forms with the U.S. Securities and Exchange Commission (SEC) through EDGAR (Electronic Data Gathering, Analysis, and Retrieval system). EDGAR automated system collects all the different necessary filings and make it publicly available. Secondly, it validates collected filings, then perform indexing and accepting of these submitted forms. Investors, regulators, and researchers often require these forms for various purposes. This package helps in data gathering, management, and visualization in this regard. It downloads SEC EDGAR quarterly master indexes, daily master indexes, and filings from SEC.org site and perform sentiment analysis of these filing.

License GPL-2

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-06-04 20:51:24 UTC

R topics documented:

`getDailyMaster` *Retrieves daily master index.*

Description

`getDailyMaster` retrieves daily master index from US SEC site.

Usage

```
getDailyMaster(input.date)
```

Arguments

`input.date`	in character format 'mm/dd/YYYY'.

Details

getDailyMaster function takes date as an input parameter from user, download master index for that particular date from https://www.sec.gov/Archives/edgar/daily-index/ site. It strips headers and converts this daily filing information in dataframe format. Function creates new directory 'Daily Index' into working directory to save these downloaded daily master index files.

Value

Function returns filing information for the input date.

Examples

```
## Not run:

daily.filing.info <- getDailyMaster('08/09/2016')

## End(Not run)
```

getFilingInfo *Retrieves filing information of company.*

Description

getFilingInfo retrieves filing information of company based on company name.

Usage

```
getFilingInfo(firm.name, filing.year)
```

Arguments

firm.name	Desired full name or partial name of company.
filing.year	Filing year in integer.

Details

getFilingInfo function takes company name and filing year as input parameters from user, and gives filing information for that company. Master index file must be present for given year before applying this function which can be downloaded using getMasterIndex function.

Value

Function returns dataframe with filing information.

Examples

```
## Not run:

info <- getFilingInfo('United Technologies', 1994)

## End(Not run)
```

getFilings *Retrieves EDGAR filings from SEC site.*

Description

getFilings retrieves EDGAR filings for the specified CIK, form-type, and year mentioned in function parameters.

Usage

```
getFilings(year, cik.no, form.type)
```

Arguments

year	an integer specifies year for which EDGAR filings are to be downloaded.
cik.no	an integer containing specific CIK number for which EDGAR filings are to be downloaded. cik.no = 'ALL' if required to download for all CIK's.
form.type	character string containing specific Form type to be downloaded. form.type = 'ALL' if required to download all form-types.

Details

getFilings function takes year, form-type, and CIK as an input. Working directory must contains master index (in Rda) files for input year which can be downloaded using getMasterIndex function. Function creates new directory 'Edgar filings' to store all downloaded filings. Please note, for all other functions in this package needs to locate the same working directory.

Value

Function downloads EDGAR filings and returns download status in dataframe.

Examples

```
## Not run:

report <- getFilings(1994, 100030, 'ALL')
## download all filings filed by the firm with CIK=100030 in the year 1994.
## Generates download report in dataframe.

report <- getFilings(2006, 1000180, '10-K')
## download '10-K' filings filed by the firm with CIK=1000180 in the year 2006.
## Generates download report in dataframe.

## End(Not run)
```

---

getMasterIndex *Retrieves quarterly master index.*

Description

getMasterIndex retrieves the quarterly master index from US SEC site.

Usage

```
getMasterIndex(year.array)
```

Arguments

year.array	year in integer or integer array containing years for which master index are to be downloaded.

Details

getMasterIndex function takes filing year as an input parameter from user, download quarterly master index from https://www.sec.gov/Archives/edgar/full-index/. It strips the headers, converts into dataframe, and merges such quarterly dataframes into yearly dataframes and stored it in Rda format. Function creates new directory 'Master Index' into working directory to save these Rda Master Index. Please note, for all other functions in this package needs to locate the same working directory to access these Rda master index files.

Value

Function retrieves quarterly master index files from https://www.sec.gov/Archives/edgar/full-index/ site and returns download status dataframe.

Examples

```
## Not run:

report <- getMasterIndex(1995)
## Downloads quarterly master index files for the year 1995 and stores into yearly
## 1995master.Rda file. It returns download report in dataframe format.

report <- getMasterIndex(c(1994, 2006, 2014))
## Download quarterly master index files for the years 1994, 1995, 2006 and stores into
## different {year}master.Rda files. It returns download report in dataframe format.

## End(Not run)
```

---

getSentimentCount *Parse sentiment words from EDGAR filing.*

Description

`getSentimentCount` get sentiment words count from filing.

Usage

```
getSentimentCount(word.frq, words.list)
```

Arguments

`word.frq`	Word frequency dataframe created using getWordfrquency function.
`words.list`	Word list as a sentiment dictionary.

Details

getSentimentCount function takes words frequency dataframe as an input from getWordfrquency function. It compares these words with the input dictionary and parse matched words with their frequencies.

Value

Function returns sentiment words frequency dataframe.

Examples

```
## Not run:

words.list <- scan(system.file('data/negwords.txt', package = 'edgar'), what='character')
## User can apply any desired user defined dictionary other than
## default dictionaries from this package in txt format with each word in separate line.

senti.words <- getSentimentCount(word.frq, words.list)

## End(Not run)
```

---

getWordcloud *Creates wordcloud of words from EDGAR filing.*

Description

getWordcloud creates wordcloud of words from filing.

Usage

```
getWordcloud(word.frq, words.list)
```

Arguments

word.frq	Word frequency dataframe created using getWordfrquency function.
words.list	Word list as a sentiment dictionary.

Details

getWordcloud function takes words frequency dataframe as an input from getWordfrquency function. It compares these words with the input dictionary and generates wordcloud using match words.

Value

Function creates wordcloud.

Examples

```
## Not run:

words.list <- scan(system.file('data/negwords.txt', package = 'edgar'), what='character')
## User can apply any desired user defined dictionary other than
## default dictionaries from this package.

getWordcloud(word.frq, words.list)

## End(Not run)
```

---

getWordfrquency *Creates words frequency dataframe from EDGAR filing.*

Description

getWordfrquency creates word frequency dataframe using filing text.

Usage

```
getWordfrquency(filepath)
```

Arguments

filepath	Path of downloaded filing.

Details

getWordfrquency function takes path of filing which can be downloaded using getFilings function. Function cleans text from filing and creates words frequency dataframe. This dataframe is used in getWordcloud, getWordHistogram, and getSentimentCount functions.

Value

Function returns words frequency dataframe.

Examples

```
## Not run:

word.frq <- getWordfrquency('Edgar Filings/1000180_10-K_2006/1000180_10-K_2006-03-15.txt')

## End(Not run)
```

`getWordHistogram` *Creates histogram of most frequent words in EDGAR filing.*

Description

`getWordHistogram` creates histogram of most frequent words in filing.

Usage

```
getWordHistogram(word.frq, words.list)
```

Arguments

`word.frq`	Word frequency dataframe created using getWordfrquency function.
`words.list`	Word list as a sentiment dictionary.

Details

getWordHistogram function takes words frequency dataframe as an input from getWordfrquency function. It compares these words with input dictionary and generates histogram of 15 most frequent matched words with their frequencies.

Value

Function creates histogram.

Examples

```
## Not run:

words.list <- scan(system.file('data/negwords.txt', package = 'edgar'), what='character')
## User can apply any desired user defined dictionary other than
## default dictionaries from this package.

getWordHistogram(word.frq, words.list)

## End(Not run)
```

mergeCompustat *Connect sentiment count of multiple cik's 10-K filing with Compustat data.*

Description

mergeCompustat merge sentiment count of 10-K filing with Compustat data.

Usage

```
mergeCompustat (cik.no, filing.yr, words.list, compustat.data)
```

Arguments

cik.no	cik number.
filing.yr	10-K filing year.
words.list	sentiment dictionary in list format.
compustat.data	Compustat dataframe.

Details

mergeCompustat function takes cik number, filing year, sentiment dictionary, and Compustat data. It downloads 10-K filings for desired cik and filing year present in an input Compustat dataset in new directory "Edgar filings/Compustat filings". The function also counts sentiment words from 10-K's and append those counts with new column named "sentiment.count" to Compustat dataframe. Working directory must contain 'Master Index' directory which contains master Rda files for specified filing year. This master index can be downloaded using getMasterIndex function. Please note that Compustat data must contains 'cik' column, and 'datadate' column in 'mm/dd/yyyy' format.

Value

Compustat data with sentiment.count column for desired cik and filing year.

Examples

```
## Not run:

## User needs to input Compustat data in dataframe format.
compustat.data <- read.csv('compustat_data.csv')

## User can apply any desired user defined dictionary
## other than default dictionaries from this package.
words.list <- scan(system.file('data/negwords.txt', package = 'edgar')
                   , what='character')

## For single cik
res <- mergeCompustat( 2098, 2014, words.list, compustat.data)
```

```
## User can provide list of different CIK's.
cik.no <- c(1750,6201,2098)
res <- mergeCompustat( cik.no, 2014, words.list, compustat.data)


## End(Not run)
```

---

negwords *Dictionary of negative words*

Description

Dataset contains negative words for sentiment analysis of EDGAR filings suggested by Loughran and McDonald's financial sentiment dictionaries.

Format

The format is: $ WORDS: Factor w/ 2329 levels "abandon","abandoned",..: 1 2 ...

Source

Loughran and McDonald Financial Sentiment Dictionaries http://www3.nd.edu/~mcdonald/Word_Lists.html

Examples

```
## Not run:

neg.words<-read.csv(system.file("data/negwords.csv", package = "edgar"))
str(neg.words)

## End(Not run)
```

---

poswords *Dictionary of positive words*

Description

Dataset contains positive words for sentiment analysis of EDGAR filings suggested by Loughran and McDonald's financial sentiment dictionaries.

Format

The format is: $ WORDS: Factor w/ 2005 levels "abound","abounds",..: 1 2 ...

Source

Loughran and McDonald Financial Sentiment Dictionaries http://www3.nd.edu/~mcdonald/Word_Lists.html

Examples

```
## Not run:
pos.words<-read.csv(system.file("data/poswords.csv", package = "edgar"))
str(pos.words)

## End(Not run)
```

Index