Linguistic Analyzer Documentation

Release 2.0

Paul Brown, Tyler Blanton

Mar 14, 2018

Contents:

CHAPTER 1

Keyword module

class Keyword.**Keyword**(*nWord=''*, *nType=0*, *nSal=0*, *nFreq=0*, *nKeyscore=0*)

Bases: object

summary: Stores a specific keyword and it's associated information. The constructor accepts the word, type, salience, frequency and keyscore.

classmethod issimilar(*passedWord*)

summary: determines if the passed keyword is similar to (or exactly the same as) the main word in the class

Parameters **passedWord** (*str*) – word

Returns boolean value of True or False

Return type bool

similarwordfrequency()

Returns the frequency of a similar word in a document

Return type int

wordfrequency()

Returns the frequency value of a word

Return type int

CHAPTER 2

KeywordList module

class KeywordList.**KeywordList**

Bases: object

Summary: A list that contains keywords. The list also contains unique keyword value, keyword score, yules k score, yules score and a document score.

calculateavgscores()

Summary: calculates a document's average score values.

Returns void

existsinlist(*keyword_name*)

Summary: searches through the list of keywords and sees if any keywords shares the same Keyword.word.

Parameters **keyword_name** (*str*) – The keyword

Returns returns true if a keyword with keyword_name as Keyword.word exists in the list. False otherwise.

Return type bool

getavgkeywordscore()

Summary: returns document's average keyword score.

Returns average keyword score

Return type int

getdocumentscore()

Summary: Returns document's score.

Returns document score

Return type int

getindexofword(*keyword_name*)

Summary: returns index of a Keyword in the list of Keywords

Parameters **key_name** (*str*) – keyword

Returns keyword index

Return type int

getkeywordscore()
Summary: returns document's keyword score.

Returns keyword score of document

Return type int

getyulesiscore()
Summary: returns document's Yule's i score.

Returns Yule's I score

Return type int

getyuleskscore()
Summary: returns document's Yule's k score.

Returns Yules K score

Rytpe int

insertkeyword(*keyword*)
Summary: inserts new Keyword into Keyword list

Parameters **keyword** (Keyword) – an instance of the class keyword

Returns void

CHAPTER 3

functionsv1 package

3.1 common_functions module

common_functions.**changefileextension**(*regfilename*)
Changes the file name string from .pdf to .txt.

Parameters **regfilename** (*str*) – name of regulatory file

Returns string with .pdf file extension

Return type str

common_functions.**cleantext**(*text_list*)
Removes special characters from text

Parameters **text_list** (*List[str]*) – a text string

Returns text_list with no special chars

Return type List[str]

common_functions.**createkeywordfromgoogleapientity**(*entity*, *file_text*)
Creates a Keyword from a single entity that is returned by the google API

Parameters

- **entity** (*Entity*) – Google API response entity object
- **file_text** (*List[str]*) – entire text of file

Returns Populated Keyword object

Return type *Keyword*

common_functions.**extractkeywordfromtxt**(*filename*)
This function will extract keyword information from .txt file and place into KeywordList object

Parameters **file** (*str*) – location of .txt file

Returns keyword list in file

Return type *KeywordList*

common_functions.**extractmicrosoftdocxtext**(*file*, *testdownload_folder=None*)
Extracts text from any ".docx" document and returns it.

Parameters

- **file** (*fileStorage*) – the file to save
- **testdownload_folder** (*str*) – Specific download folder is necessary

Returns file's text

Return type List[str]

common_functions.**extractpdftext**(*file*, *testdownload_folder=None*, *RegDoc=False*)
Extracts Text from PDF document referenced in given file argument

Parameters

- **file** (*fileStorage*) – the PDF file to extract text from
- **testdownload_folder** (*str*) – specific download folder if necessary
- **RegDoc** (*bool*) – flag specifying whether this is a user doc or a regulatory doc

Returns file's text

Return type List[str]

common_functions.**generatebubblecsv**(*kw_list*, *reg_kw_list*)
Creates a new csv file with all the keywords

Parameters

- **kw_list** (KeywordList) – list of doc keywords
- **reg_kw_list** (KeywordList) – list of reg doc keywords

Returns void

common_functions.**geterrorpage**(*errtext='Unknown Error'*)
Populates error message with proper response and returns html

Parameters **errtext** (*str*) – text of error

Returns html page with error displayed

Return type str

common_functions.**getregulatorydoctext**(*filename*)
Looks in the RegulatoryDocuments folder for the file with the given file name and return's its text as a list of string

Parameters **filename** (*str*) – name of regulatory file without file ending on it

Returns list of strings of length 1024 containing text of file

Return type List[str]

common_functions.**getscorepage**(*kw_list*, *reg_kw_list*, *userdocwordcount*, *filename*, *regfilename*)
Returns html page that is populated with proper calculated Keyword, Comparison, and Yule's scores.

Parameters

- **kw_list** (KeywordList) – list of user document's Keyword objects
- **reg_kw_list** (KeywordList) – list of regulatory document's Keywords

Returns html page with scores displayed

Return type str

common_functions.**getwordfrequency**(*word*, *file_text*)
Determines frequency of the given word in the file's text

Parameters

- **word** (*str*) – Word to find frequency of
- **filetext** (*List[str]*) – list of string containing entire text of file

Returns frequency of word parameter in text

Return type int

common_functions.**homeCount**()
Initializes variables for logging session

Returns void

common_functions.**interpretexistingfile**(*regfilename*)
Parses, identifies keywords and analyzes content of chosen regulatory file document is being compares against.

Parameters **regfilename** (*str*) – name of regulatory file

Returns list of analyzed Keyword objects

Return type *KeywordList*

common_functions.**interpretfile**(*file*, *localuploadfolder*)
Parses uploaded file's text, identifies keywords, analyzes keywords, and returns a list of Keyword Objects

Parameters

- **file** (*fileStorage*) – file to be interpreted
- **localuploadfolder** (*str*) – Place to temporary store file so it can be read from

Returns list of file's Keywords

Return type *KeywordList*

common_functions.**kwhighestfrequencies**(*keyword_list*, *numtopkws=10*)
Returns the top 10 most frequent Keywords in the user's uploaded file

Parameters **keyword_list** (KeywordList) – List of Keyword objects

Returns Keywords with highest frequencies

Return type List[*Keyword*]

common_functions.**kwhighestkeyscores**(*keyword_list*)
Returns ten Keywords with the highest Keyword scores

Parameters **keyword_list** (KeywordList) – list of Keyword objects

Returns list of top keyword scores

Return type List[*Keyword*]

common_functions.**longstringtostringlist**(*longstring*, *strsize*)
This functions splits a long string "longstring" into strings of size "strsize" and returns a list of those strings.

Parameters

- **longstring** (*string*) – text of file

- **strsize** (*int*) – requested length of each string in created list of strings

Returns file text

Return type List[str]

common_functions.**outputkeywordtotext**(*keylist*, *download_folder='Documents/Keywords.txt'*)
This function will write Keywords from an analyzed document to a .txt file

Parameters **keylist** (KeywordList) – list of document keywords

Returns void

common_functions.**plotkeywordfrequency**(*keyword_list1*, *keyword_list2*, *doc1name='doc1'*, *doc2name='doc2'*)
Plots keyword score of most frequently used keywords. Saves graph to "/Downloads" folder

Parameters

- **keyword_list1** (KeywordList) – user document keywords
- **keyword_list2** (KeywordList) – regulatory document keywords
- **doc1name** (*str*) – name of user document
- **doc2name** (*str*) – name of regulatory document

Returns void

common_functions.**plotkeywordsalience**(*keyword_list1*, *keyword_list2*, *doc1name='doc1'*, *doc2name='doc2'*)
Plots salience of most frequently used keywords. Pulls KWs from list1, compares against list2

Parameters

- **keyword_list1** (KeywordList) – user KeywordList
- **keyword_list2** (KeywordList) – regulatory KeywordList
- **doc1name** (*str*) – user document name
- **doc2name** (*str*) – regulatory document name

Returns void

common_functions.**plotkeywordscores**(*keyword_list1*, *keyword_list2*, *doc1name='doc1'*, *doc2name='doc2'*)
Plots keyword score of most frequently used keywords. Pulls KWs from list1, compares against list2

Parameters

- **keyword_list1** (KeywordList) – user KeywordList
- **keyword_list2** (KeywordList) – regulatory KeywordList
- **doc1name** (*str*) – user document name
- **doc2name** (*str*) – regulatory document name

Returns void

common_functions.**printStringList**(*textList*)
Helper function that prints a list of strings

Parameters **textList** (*List[str]*) – a text string

Returns void

common_functions.**printanalytics**(*filename*, *regfilename*, *keywordlist*, *regkeywordlist*, *calctime*)
prints the data passed in te argument to the ever-increasing file that contains data analytics information

Parameters **printstr** (*str*) – string to output to file

Returns void

common_functions.**savefile**(*file*, *download_folder=None*)
Save's given file to /Downloads folder"

Parameters

- **file** (*fileStorage*) – the file to save
- **download_folder** (*str*) – specific download folder if necessary

Returns void

common_functions.**splitintosize**(*file_text*)
This function splits a list of keywords of any length into a lit of keywords eachof length specified by NUM_SEND_CHARS in 'applicationconfig.json'

Parameters **file_text** (*list*) – list of document's words

Return list file_text

common_functions.**stringlisttolonglongstring**(*string_list*)
Helper function to turn list of string into one long long string

Parameters **string_list** (*List[str]*) – a string of text

Returns file's text

Return type long string

common_functions.**writeToConfig**(*key*, *value*)
Writes value into applicationconfig.json file

Parameters

- **key** – key
- **value** – value

Returns none

3.2 analyze_functions module

analyze_functions.**calculatecomparisonscore**(*kw_list*, *reg_kw_list*)

Summary: Compares the calculated scores of the two documents and generates value based on that comparison

Parameters

- **kw_list** (KeywordList) – list of Keywords
- **reg_kw_list** (KeywordList) – list of Keywords

Returns comparison score of two documents

Return type float

analyze_functions.**calculatekeywordscore**(*kw_list*, *file_text*, *kw*)
Summary: calculate a keyword score for a single keyword

Parameters

- **kw_list** (KeywordList) – all keywords
- **file_text** (*list[str]*) – file's entire text
- **kw** (Keyword) – keyword

Returns keyword score

Return type float

analyze_functions.**calculatescores**(*kw_list*, *file_text*)
Summary: Calculate Yule's k and i scores, and keywords scores for a given document

Parameters

- **kw_list** (KeywordList) – list of Keywords
- **file_text** (*List[string]*) – Text of file

Returns void

analyze_functions.**calculateyulesscore**(*file_text*)
Summary: calculates Yule's K scores for givven keyword argument

Parameters **file_text** (*list[str]*) – plain text of document

Returns Yules score of text file

Return type float

analyze_functions.**declarelogger**()
Summary: Declares logger for the current session.

analyze_functions.**identifykeywords**(*file_text*)
Summary: Calls the Google NLP API to extract Keyword information from text

Parameters **file_text** (*str*) – text of document

Returns KeywordList object

Return type *KeywordList*

analyze_functions.**tokenize**(*tokenStr*)
Summary: Splits up string into individual tokens.

Parameters **tokenStr** (*str*) – a string of words

Returns tokens

Return type list

CHAPTER 4

analyze module

analyze.**analyzeText**(*fileText*)

Parameters **fileText** (*str*) – text of fileText

Returns file text

Return type str

analyze.**checkSimilarity**(*fileText*)

Parameters **fileText** (*str*) – text of file

Returns pass or fail

Return type bool

analyze.**createObjects**(*fileText*)

Parameters **fileText** (*str*) – text of file

Returns pass or fail

Return type bool

analyze.**scrapeText**(*fileText*)

Parameters **fileText** (*str*) – text of file

Returns pass or fail

Return type bool

CHAPTER 5

application module

application.**analyze**()
Receives uploaded document and comparison document choice and executes logic to compare them.

Returns Information regarding the uploaded document's similarity to regulatory document

Return type html

application.**bubbletest**()
Page for testing

Returns Test page

Return type html

application.**comparisoninfo**()
Comparison Information

Returns graph html page that describes the Linguistic Analyzer's Comparison Score

Return type html

application.**getapplicationconfig**()

Returns json application config file

Returns applicationconfig.json

Return type json file

application.**getbackgroundimg**()
Returns png image of file at

Returns graph

Return type png

application.**getbackgroundwordsimg**()
Returns png image of a graph of words background

Returns graph

Return type png

application.**getcsvkeywords**()
Returns csvkeywords.csv

Returns csvkeywords keyword file

Return type csv

application.**getdocumentationhome**()
Returns index page nested in Documentation/_build/html which is the home page for our Sphinx-generated documentation

Returns html text

application.**getkwfreeqimage**()
Returns Keyword frequency graph

Returns graph

Return type png

application.**getkwsalienceimage**()
Returns png image of a graph of top salience keywords

Returns graph

Return type png

application.**getkwscoresimage**()
Returns png image of a graph of keyword scores

Returns graph

Return type png

application.**getlinguisticanalyzerlog**()
Returns LinguisticAnalyzer.log

Returns log file

Return type log

application.**getregdockws**()
Returns Reg_Keywords.txt

Returns regulatory doc keyword file

Return type txt

application.**gettestkeywords**()
Returns test_keywords.csv

Returns test_keywords doc keyword file

Return type csv

application.**getuserdockws**()
Returns Keywords.txt

Returns keyword file

Return type txt

application.**indexjs**()
Page for testing

Returns Test page

Return type html

application.**keywordbubblechart**()
Returns bubble chart html page

Returns bubble chart html page

Return type html

application.**main**()
Home page of the Linguistic Analyzer API

Returns Home page

Return type html

application.**newregdoc**()
Adds new regulatory document

Returns none

Return type none

application.**project**()
Returns an html page containing details about the Linguistic Analyzer project.

Returns Home page

Return type html

application.**resource_path**(*relative_path*)
Summary: Function to determine correct file path of directories for use within an IDE or executable.

Parameters **relative_path** (*str*) – the path of a directory relative to a local environment

Returns base_path in relation to executable environment and relative_path of local environment

Return type string

application.**reusablebubble**()
Page for testing

Returns Test page

Return type html

application.**reusablebubblejs**()
Page for testing

Returns Test page

Return type html

application.**yulesinfo**()
Yule's Info

Returns Page that describes Yule's k and Yule's i algorithms

Return type html

CHAPTER 6

unit_tests package

6.1 test_analyze module

class unit_tests.test_analyze.**TestAnalyze**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_analyze()
Summary: Tests the Analyze() function

6.2 test_extractmicrosoftdocxtext module

class unit_tests.test_extractmicrosoftdocxtext.**TestExtractmicrosoftdocxtext**(*methodName='runTe*
Bases: unittest.case.TestCase

test_extractmicrosoftdocxtext()
Summary: Tests the extractmicrosoftdoctet() function

6.3 test_extractpdftext module

class unit_tests.test_extractpdftext.**TestExtractpdftext**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_extractpdftext()
Summary: Tests the extractpdftext() function

6.4 test_outputkeywordtotext module

class unit_tests.test_outputkeywordtotext.**TestOutputkeywordtotext**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_outputkeywordtotext()

6.5 test_pdfanddocxarereadthesame module

class unit_tests.test_pdfanddocxarereadthesame.**TestEnsurepdfanddocxarereadthesame**(*methodName*
Bases: unittest.case.TestCase

test_ensurepdfanddocarereadthesame()
Summary: tests whether extractpdftext() and extractdocxtext() return the same exact information when given the same document in different formats

CHAPTER 7

behave_tests package

7.1 tutorial module

behave_tests.tutorial.steps.tutorial.**step_impl**(*context*)
@type context: behave.runner.Context

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index

K

L

M

N

O

P

R

S

T

U

W

Y