CS F469 INFORMATION RETRIEVAL

(First semester 2019-2020)

Assignment-1

Documentation

**Group Members:**

Jaju Vedant Vinod - 2016A3PS0303H

Tulika Manek – 2017A7PS0147H

Pavan Chakradhar Reddy T – 2017A7PS0189H

Tanuja Kirthi D – 2017A7PS0044H

**Functions/Classes/Modules:**

CorpusProcessing:

**1) RegexpTokenizer in module nltk.tokenize.regexp object:**

class RegexpTokenizer(nltk.tokenize.api.TokenizerI)

| RegexpTokenizer(pattern, gaps=False, discard\_empty=True, flags=<RegexFlag.UNICODE|DOTALL|MULTILINE: 56>)

|

| A tokenizer that splits a string using a regular expression, which

| matches either the tokens or the separators between tokens.

|

| >>> tokenizer = RegexpTokenizer('\w+|\$[\d\.]+|\S+')

|

| :type pattern: str

| :param pattern: The pattern used to build this tokenizer.

| (This pattern must not contain capturing parentheses;

| Use non-capturing parentheses, e.g. (?:...), instead)

| :type gaps: bool

| :param gaps: True if this tokenizer's pattern should be used

| to find separators between tokens; False if this

| tokenizer's pattern should be used to find the tokens

| themselves.

| :type discard\_empty: bool

| :param discard\_empty: True if any empty tokens `''`

| generated by the tokenizer should be discarded. Empty

| tokens can only be generated if `\_gaps == True`.

| :type flags: int

| :param flags: The regexp flags used to compile this

| tokenizer's pattern. By default, the following flags are

| used: `re.UNICODE | re.MULTILINE | re.DOTALL`.

|

| Method resolution order:

| RegexpTokenizer

| nltk.tokenize.api.TokenizerI

| builtins.object

|

| Methods defined here:

|

| \_\_init\_\_(self, pattern, gaps=False, discard\_empty=True, flags=<RegexFlag.UNICODE|DOTALL|MULTILINE: 56>)

| Initialize self. See help(type(self)) for accurate signature.

|

| \_\_repr\_\_(self)

| Return repr(self).

|

| \_\_unicode\_\_ = \_\_str\_\_(self, /)

|

| span\_tokenize(self, text)

| Identify the tokens using integer offsets ``(start\_i, end\_i)``,

| where ``s[start\_i:end\_i]`` is the corresponding token.

|

| :rtype: iter(tuple(int, int))

|

| tokenize(self, text)

| Return a tokenized copy of \*s\*.

|

| :rtype: list of str

|

| unicode\_repr = \_\_repr\_\_(self)

|

| ----------------------------------------------------------------------

| Data and other attributes defined here:

|

| \_\_abstractmethods\_\_ = frozenset()

|

| ----------------------------------------------------------------------

| Methods inherited from nltk.tokenize.api.TokenizerI:

|

| span\_tokenize\_sents(self, strings)

| Apply ``self.span\_tokenize()`` to each element of ``strings``. I.e.:

|

| return [self.span\_tokenize(s) for s in strings]

|

| :rtype: iter(list(tuple(int, int)))

|

| tokenize\_sents(self, strings)

| Apply ``self.tokenize()`` to each element of ``strings``. I.e.:

|

| return [self.tokenize(s) for s in strings]

|

| :rtype: list(list(str))

|

| ----------------------------------------------------------------------

| Data descriptors inherited from nltk.tokenize.api.TokenizerI:

|

| \_\_dict\_\_

| dictionary for instance variables (if defined)

|

| \_\_weakref\_\_

| list of weak references to the object (if defined)

**2) PorterStemmer in module nltk.stem.porter object:**

class PorterStemmer(nltk.stem.api.StemmerI)

| PorterStemmer(mode='NLTK\_EXTENSIONS')

|

| A word stemmer based on the Porter stemming algorithm.

|

| Porter, M. "An algorithm for suffix stripping."

| Program 14.3 (1980): 130-137.

|

| See http://www.tartarus.org/~martin/PorterStemmer/ for the homepage

| of the algorithm.

|

| Martin Porter has endorsed several modifications to the Porter

| algorithm since writing his original paper, and those extensions are

| included in the implementations on his website. Additionally, others

| have proposed further improvements to the algorithm, including NLTK

| contributors. There are thus three modes that can be selected by

| passing the appropriate constant to the class constructor's `mode`

| attribute:

|

| PorterStemmer.ORIGINAL\_ALGORITHM

| - Implementation that is faithful to the original paper.

|

| Note that Martin Porter has deprecated this version of the

| algorithm. Martin distributes implementations of the Porter

| Stemmer in many languages, hosted at:

|

| http://www.tartarus.org/~martin/PorterStemmer/

|

| and all of these implementations include his extensions. He

| strongly recommends against using the original, published

| version of the algorithm; only use this mode if you clearly

| understand why you are choosing to do so.

|

| PorterStemmer.MARTIN\_EXTENSIONS

| - Implementation that only uses the modifications to the

| algorithm that are included in the implementations on Martin

| Porter's website. He has declared Porter frozen, so the

| behaviour of those implementations should never change.

|

| PorterStemmer.NLTK\_EXTENSIONS (default)

| - Implementation that includes further improvements devised by

| NLTK contributors or taken from other modified implementations

| found on the web.

|

| For the best stemming, you should use the default NLTK\_EXTENSIONS

| version. However, if you need to get the same results as either the

| original algorithm or one of Martin Porter's hosted versions for

| compatibility with an existing implementation or dataset, you can use

| one of the other modes instead.

|

| Method resolution order:

| PorterStemmer

| nltk.stem.api.StemmerI

| builtins.object

|

| Methods defined here:

|

| \_\_init\_\_(self, mode='NLTK\_EXTENSIONS')

| Initialize self. See help(type(self)) for accurate signature.

|

| \_\_repr\_\_(self)

| Return repr(self).

|

| \_\_unicode\_\_ = \_\_str\_\_(self, /)

|

| stem(self, word)

| Strip affixes from the token and return the stem.

|

| :param token: The token that should be stemmed.

| :type token: str

|

| unicode\_repr = \_\_repr\_\_(self)

|

| ----------------------------------------------------------------------

| Data and other attributes defined here:

|

| MARTIN\_EXTENSIONS = 'MARTIN\_EXTENSIONS'

|

| NLTK\_EXTENSIONS = 'NLTK\_EXTENSIONS'

|

| ORIGINAL\_ALGORITHM = 'ORIGINAL\_ALGORITHM'

|

| \_\_abstractmethods\_\_ = frozenset()

|

| ----------------------------------------------------------------------

| Data descriptors inherited from nltk.stem.api.StemmerI:

|

| \_\_dict\_\_

| dictionary for instance variables (if defined)

|

| \_\_weakref\_\_

| list of weak references to the object (if defined)

**3) Built-in function getcwd in module os:**

getcwd() – Return a Unicode string representing the current working directory.

**4) Module glob:**

NAME

glob - Filename globbing utility.

FUNCTIONS

escape(pathname)

Escape all special characters.

glob(pathname, \*, recursive=False)

Return a list of paths matching a pathname pattern.

The pattern may contain simple shell-style wildcards a la

fnmatch. However, unlike fnmatch, filenames starting with a

dot are special cases that are not matched by '\*' and '?'

patterns.

If recursive is true, the pattern '\*\*' will match any files and

zero or more directories and subdirectories.

iglob(pathname, \*, recursive=False)

Return an iterator which yields the paths matching a pathname pattern.

The pattern may contain simple shell-style wildcards a la

fnmatch. However, unlike fnmatch, filenames starting with a

dot are special cases that are not matched by '\*' and '?'

patterns.

If recursive is true, the pattern '\*\*' will match any files and

zero or more directories and subdirectories.

DATA

\_\_all\_\_ = ['glob', 'iglob', 'escape']

**5) Method tokenize in module nltk.tokenize.regexp:**

tokenize(text) method of nltk.tokenize.regexp.RegexpTokenizer instance

Return a tokenized copy of \*s\*.

:rtype: list of str

**6) Method stem in module nltk.stem.porter:**

stem(word) method of nltk.stem.porter.PorterStemmer instance

Strip affixes from the token and return the stem.

:param token: The token that should be stemmed.

:type token: str

**7) Function deepcopy in module copy:**

deepcopy(x, memo=None, \_nil=[])

Deep copy operation on arbitrary Python objects.

**8) Method words in module nltk.corpus.reader.wordlist:**

words(fileids=None, ignore\_lines\_startswith='\n') method of nltk.corpus.reader.wordlist.WordListCorpusReader instance

**9) DataFrame in module pandas.core.frame object:**

class DataFrame(pandas.core.generic.NDFrame)

| DataFrame(data=None, index=None, columns=None, dtype=None, copy=False)

|

| Two-dimensional size-mutable, potentially heterogeneous tabular data

| structure with labeled axes (rows and columns). Arithmetic operations

| align on both row and column labels. Can be thought of as a dict-like

| container for Series objects. The primary pandas data structure.

|

| Parameters

| ----------

| data : ndarray (structured or homogeneous), Iterable, dict, or DataFrame

| Dict can contain Series, arrays, constants, or list-like objects

|

| .. versionchanged :: 0.23.0

| If data is a dict, column order follows insertion-order for

| Python 3.6 and later.

|

| .. versionchanged :: 0.25.0

| If data is a list of dicts, column order follows insertion-order

| Python 3.6 and later.

|

| index : Index or array-like

| Index to use for resulting frame. Will default to RangeIndex if

| no indexing information part of input data and no index provided

| columns : Index or array-like

| Column labels to use for resulting frame. Will default to

| RangeIndex (0, 1, 2, ..., n) if no column labels are provided

| dtype : dtype, default None

| Data type to force. Only a single dtype is allowed. If None, infer

| copy : boolean, default False

| Copy data from inputs. Only affects DataFrame / 2d ndarray input

|

|

QueryProcessing:

**1) Method tokenize in module nltk.tokenize.regexp:**

tokenize(text) method of nltk.tokenize.regexp.RegexpTokenizer instance

Return a tokenized copy of \*s\*.

:rtype: list of str

**2) Function deepcopy in module copy:**

deepcopy(x, memo=None, \_nil=[])

Deep copy operation on arbitrary Python objects.

**3) Method stem in module nltk.stem.porter:**

stem(word) method of nltk.stem.porter.PorterStemmer instance

Strip affixes from the token and return the stem.

:param token: The token that should be stemmed.

:type token: str

**4) Class enumerate in module builtins:**

class enumerate(object)

| enumerate(iterable, start=0)

|

| Return an enumerate object.

|

| iterable

| an object supporting iteration

|

| The enumerate object yields pairs containing a count (from start, which

| defaults to zero) and a value yielded by the iterable argument.

|

| enumerate is useful for obtaining an indexed list:

| (0, seq[0]), (1, seq[1]), (2, seq[2]), ...

|

| Methods defined here:

|

| \_\_getattribute\_\_(self, name, /)

| Return getattr(self, name).

|

| \_\_iter\_\_(self, /)

| Implement iter(self).

|

| \_\_next\_\_(self, /)

| Implement next(self).

|

| \_\_reduce\_\_(...)

| Return state information for pickling.

|

| ----------------------------------------------------------------------

| Static methods defined here:

|

| \_\_new\_\_(\*args, \*\*kwargs) from builtins.type

| Create and return a new object. See help(type) for accurate signature.

**5) Matmul in numpy module - ufunc object:**

matmul = class ufunc(builtins.object)

| Functions that operate element by element on whole arrays.

|

| To see the documentation for a specific ufunc, use `info`. For

| example, ``np.info(np.sin)``. Because ufuncs are written in C

| (for speed) and linked into Python with NumPy's ufunc facility,

| Python's help() function finds this page whenever help() is called

| on a ufunc.

|

| A detailed explanation of ufuncs can be found in the docs for :ref:`ufuncs`.

|

| Calling ufuncs:

| ===============

|

| op(\*x[, out], where=True, \*\*kwargs)

| Apply `op` to the arguments `\*x` elementwise, broadcasting the arguments.

|

| The broadcasting rules are:

|

| \* Dimensions of length 1 may be prepended to either array.

| \* Arrays may be repeated along dimensions of length 1.

|

| Parameters

| ----------

| \*x : array\_like

| Input arrays.

| out : ndarray, None, or tuple of ndarray and None, optional

| Alternate array object(s) in which to put the result; if provided, it

| must have a shape that the inputs broadcast to. A tuple of arrays

| (possible only as a keyword argument) must have length equal to the

| number of outputs; use `None` for uninitialized outputs to be

| allocated by the ufunc.

| where : array\_like, optional

| This condition is broadcast over the input. At locations where the

| condition is True, the `out` array will be set to the ufunc result.

| Elsewhere, the `out` array will retain its original value.

| Note that if an uninitialized `out` array is created via the default

| ``out=None``, locations within it where the condition is False will

| remain uninitialized.

| \*\*kwargs

| For other keyword-only arguments, see the :ref:`ufunc docs <ufuncs.kwargs>`.

|

| Returns

| -------

| r : ndarray or tuple of ndarray

| `r` will have the shape that the arrays in `x` broadcast to; if `out` is

| provided, it will be returned. If not, `r` will be allocated and

| may contain uninitialized values. If the function has more than one

| output, then the result will be a tuple of arrays.

**6) Built-in function sorted in module builtins:**

sorted(iterable, /, \*, key=None, reverse=False)

Return a new list containing all items from the iterable in ascending order.

A custom key function can be supplied to customize the sort order, and the

reverse flag can be set to request the result in descending order.