Soheila Behrooznia 974110

NLP course => NLTK- ch03

November 2019

#6

Describe the class of strings matched by the following regular expressions.

a. [a-zA-Z]+

b. [A-Z][a-z]*

c. p[aeiou]{,2}t

d. \d+(\.\d+)?

e. ([^aeiou][aeiou][^aeiou])*

f. \w+|[^\w\s]+

Test your answers using nltk.re_show().

Answer:

The selected sentences of this part belongs to the introduction of AI lab part of IASBS website.

“Artificial Intelligence (AI) is a branch of Computer Science concerned with making computers behave like humans. The term was coined in 1956 by John McCarthy at the Massachusetts Institute of Technology. Artificial intelligence aims to understand the nature of intelligence and to engineer systems that exhibit such intelligence by utilizing vision, language, and knowledge. The resultant knowledge has diverse applications particularly for designing and developing humanoid robots.The objective of the AI lab is to identify and develop technologies to enable computers to learn, think, see, and understand the scenes like humans, communicate with human beings, and self-monitor/protect their systems. More specifically, we focus on:Machine Learning Computer VisionNatural Language Processing Computer Security and Malware Detection”

a=>We have at least one of these characters

b=>We have A-Z but we may have not a-z

c=> {n,m} means: repeat the previous word at least n times and at most m times. If we put [aeiou] as X the output should be: pt, pXt, pXXt.

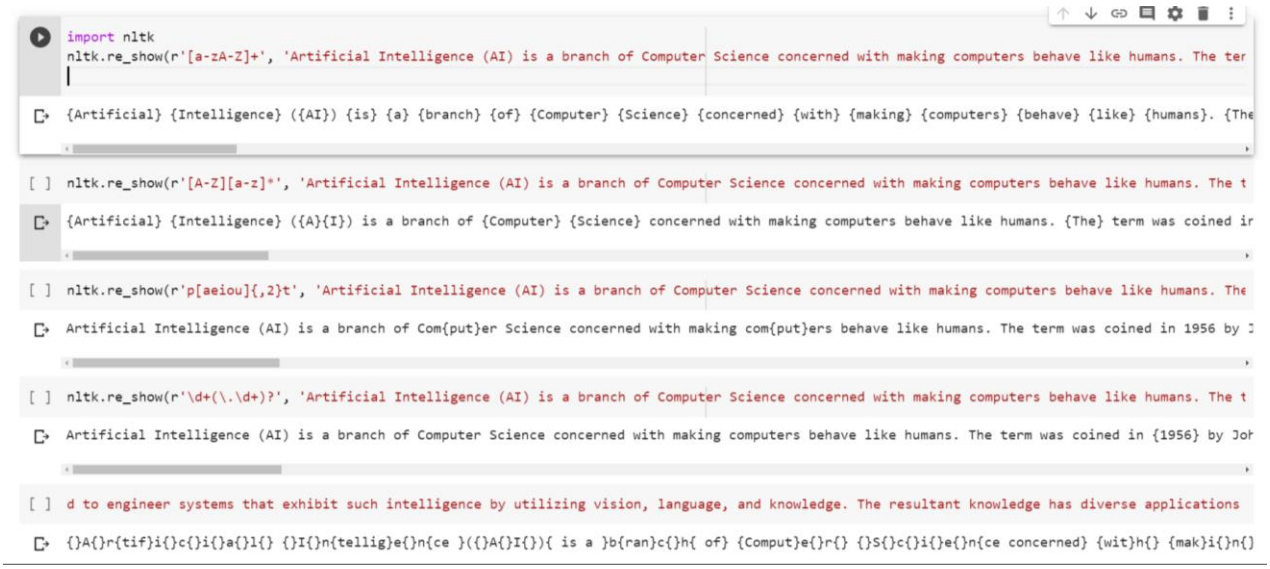
Note: each character of X doesn’t repeat or comes 1!

d=> \d is any digit (also Persian numbers)

at first we have a digit. The float sign can be put in middle. And the ? shows us that the parentheses is optional.

e=> It means “constant +vowel+ constant”

f=> \w is any word character and \s matches a space, a tab or line break. A word is produced then for separating this word from other words in sentence we use ^\w. It means we can’t produce a word and \s shows us that we should separate this word from other words in the sentence by a space or break line.



#8

Write a utility function that takes a URL as its argument, and returns the contents of the URL, with all HTML markup removed.

Use from urllib import request and *then request.urlopen('http://nltk.org/').read().decode('utf8')* to access the contents of the URL.

Answer:

The urllib.request module defines functions and classes which help in opening URLs (mostly HTTP) in a complex world — basic and digest authentication, redirections, cookies and more.

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

=>Both of these libraries are needed to fetch a URL.

Then we use utility function which uses “url” as its input. Then we should decode this url.

```
from urllib import request
from bs4 import BeautifulSoup

def utility(url):
  link = url
  html = request.urlopen(url).read().decode('utf8')
  raw = BeautifulSoup(html).get_text()
  print(raw)
utility('http://nltk.org')
```

```
along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning,
wrappers for industrial-strength NLP libraries,
and an active discussion forum.
Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation
NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike.
NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.
NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,”
and “an amazing library to play with natural language.”
Natural Language Processing with Python provides a practical
introduction to programming for language processing.
Written by the creators of NLTK, it guides the reader through the fundamentals
of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure,
and more.
The online version of the book has been been updated for Python 3 and NLTK 3.
(The original Python 2 version is still available at http://nltk.org/book_1ed.)

Some simple things you can do with NLTK¶
Tokenize and tag some text:
>>> import nltk
>>> sentence = """At eight o'clock on Thursday morning
```

#14

Create a variable words containing a list of words. Experiment with *words.sort()* and *sorted(words)*. What is the difference?

Answer:

words.sort() modify the original variable words, and it doesn’t have output but sorted(words) output is a sorted list:

```
[25]
    words = ['Soheila', 'Nastaran', 'IASBS', 'music','best']
    sorted(words)
```

```
['IASBS', 'Nastaran', 'Soheila', 'best', 'music']
```

```
output=words.sort()
print(output)
```

```
None
```

#21

Write a function *unknown()* that takes a URL as its argument, and returns a list of unknown words that occur on that webpage. In order to do this, extract all substrings consisting of lowercase letters (using *re.findall()*) and remove any items from this set that occur in the Words Corpus (nltk.corpus.words). Try to categorize these words manually and discuss your findings.

Answer:

The pprint module provides a capability to “pretty-print” arbitrary Python data structures in a form which can be used as input to the interpreter. If the formatted structures include objects which are not fundamental Python types, the representation may not be loadable.

The simplest way to use the module is through the pprint() function. pprint() formats an object and writes it to the data stream passed in as an argument (or sys.stdout by default).

The random module provides access to functions that support many operations.It allows you to generate random numbers.

The algorithm is mentioned in the question. We should define *unknown* function with a URL as its input.

#39

Read the Wikipedia entry on *Soundex*. Implement this algorithm in Python.

Answer:

Soundex is an algorithm based on phonetic and pronunciation for indexing names. It is developed in the early 1900s by Robert C. Russell and Margaret King Odell. Soundex attempts to find similar names or homophones using phonetic notation. The program retains letters according to detailed equations, to match individual names for the purposes of large volume research.

The Soundex code for a name consists of a letter followed by three numerical digits: the letter is the first letter of the name, and the digits encode the remaining consonants. Consonants at a similar place of articulation share the same digit so, for example, the labial consonants B, F, P, and V are each encoded as the number 1.

code	letter	description
1	B.F,P,V	Labial
2	C,G,J,K,Q,S,X,Z	Gutterals and sibilants
3	D,T	Dental
4	L	Long liquid
5	M,N	Nasal
6	R	Short liquid
SKIP	Other letters are skipped	-

The correct value can be found as follows:

Step 1.Retain the first letter of the name and drop all other occurrences of a, e, i, o, u, y, h, w.

Step 2.Replace consonants with digits as follows (after the first letter) => according to table

Step 3.If two or more letters with the same number are adjacent in the original name (before step 1), only retain the first letter; also two letters with the same number separated by 'h' or 'w' are coded as a single number, whereas such letters separated by a vowel are coded twice. This rule also applies to the first letter.

Step 4. If you have too few letters in your word that you can't assign three numbers, append with zeros until there are three numbers. If you have more than 3 letters, just retain the first 3 numbers.

**** CODES are attached ****