Cleaning Text for Natural Language Processing Tasks in Machine Learning in Python

ieva.rocks/2016/08/07/cleaning-text-for-nlp

ieva

August 7, 2016

Often when I work with text I need it to be clean. That is to remove gibberish or symbols/words I don't need and to make all letters lowercase.

For example, a "dirty" line of text:

```
text = ['This is dirty TEXT: A phone number +001234561234, moNey 3.333, some date like
09.08.2016 and weird Čárákterš.']
```

Using Python2.7:

1) Read the line from list:

```
for line in text:
    # do something with line
```

or read from file:

```
with open('file.txt', 'r') as f:
    for line in f:
        # do something with line
```

2) Decode the line to utf8 from a string of bytes to work with special symbols:

```
line = line.decode('utf8')
```

3) Remove the symbols you don't need. With replace() you can stack as many replace operations as you want.

```
line = line.replace('+', ' ').replace('.', ' ').replace(',', ' ').replace(':', ' ')
```

4) Remove numbers. Here you can use regex \d+. Because dots have already been removed we only need to check for whole numbers.

```
line = re.sub("(^|\W)\d+($|\W)", " ", line)
```

This regex matches the start of line ^ or whitespace, digits, end of line $ or whitespace to a space.

Alternatively you can just check if a word evaluates to a number by a simple function – is_digit() attempts to turn a string into int. If it succeeds, then the function returns true.

```
def is_digit(word):
    try:
        int(word)
        return True
    except ValueError:
        return False
```

Use this function on each word in the line by splitting the line on space with line.split(). New line array will hold only those words that are not numbers. At the end the array is joined together to a string.

```
new_line = []
for word in line.split():
    if not is_digit(word):
        new_line.append()
line = " ".join(new_line)
```

5) Now only lowercase and special characters remain. As lowercase only supports Latin letters, the special characters need to be turned to Latin. This can be done using Transliterate Python package or by hand. Here is a simple transliteration dictionary made from lists of character pairs:

```
cedilla2latin = [[u'Á', u'A'], [u'á', u'a'], [u'Č', u'C'], [u'č', u'c'], [u'Š', u'S'], [u'š', u's']]
tr = dict([(a[0], a[1]) for (a) in cedilla2latin])
```

In this way you can have multiple simbols to stand for one special symbol (like German [u'ä', u'ae']).
With the dictionary I can recreate letters in Latin:

```
def transliterate(line):
    new_line = ""
        for letter in line:
            if letter in tr:
                new_line += tr[letter]
            else:
                new_line += letter
    return new_line
```

And call the transliterate function:

```
line = transliterate(line)
```

6) After clearing away unnecessary symbols, finally I can lowercase the line:

```
line = line.lower()
```

And finally the line is reduced to simple Latin characters.

```
print line
>> this is dirty text a phone number money some date like and weird carakters
```

If you need to retain some numbers or check for other fields then go ahead and write more specific regexes. However, regexes in Python use backtracking that makes them n-squared in terms of speed. This can slow you down especially if you are working with millions of lines.

As a side note a more general cleaning method that leaves only Latin characters can be to check for the ASCII value of each letter with ord().

```
def get_latin(line):
    return ' '.join(''.join([i if ord(i) >=65 and ord(i) <=90 or  ord(i) >= 97 and ord(i) <= 122 else
' ' for i in line]).split())
```

Full code of the above description is available below or here:

```
# -*- coding: utf-8 -*-
# Ieva Zarina, 2016, licensed under the Apache 2.0 licnece

import re

def is_digit(word):
try:
int(word)
return True
except ValueError:
return False

cedilla2latin = [[u'Á', u'A'], [u'á', u'a'], [u'Č', u'C'], [u'č', u'c'], [u'Š', u'S'], [u'š', u's']]
tr = dict([(a[0], a[1]) for (a) in cedilla2latin])

def transliterate(line):
new_line = ""
for letter in line:
if letter in tr:
new_line += tr[letter]
else:
```

```
new_line += letter
return new_line

text = ['This is dirty TEXT: A phone number +001234561234, moNey 3.333, some date like 09.08.2016 and weird Čárákterš.']

for line in text:
# decode line to worrk with utf8 symbols
line = line.decode('utf8')
line = line.replace('+', ' ').replace('.', ' ').replace(',', ' ').replace(':', ' ')
# remove digits with regex
line = re.sub("(^|\W)\d+($|\W)", " ", line)
# OR remove digits with casting to int
new_line = []
for word in line.split():
if not is_digit(word):
new_line.append(word)
line = " ".join(new_line)
# transliterate to Latin characters
line = transliterate(line)
line = line.lower()
print line
```

view raw text_cleaning.py hosted with by GitHub