import re

import nltk

from nltk.stem.porter import PorterStemmer

def get\_tokens(stem):

tokens = nltk.word\_tokenize(stem)

return tokens

def do\_stemming(filtered):

stemmed = []

for f in filtered:

stemmed.append(PorterStemmer().stem(f)) # не используется? снесите нах

return stemmed

class Porter:

PERFECTIVEGROUND = re.compile(u"((ив|ивши|ившись|ыв|ывши|ывшись)|((?<=[ая])(в|вши|вшись)))$")

REFLEXIVE = re.compile(u"(с[яь])$")

ADJECTIVE = re.compile(u"(ее|ие|ые|ое|ими|ыми|ей|ий|ый|ой|ем|им|ым|ом|его|ого|ему|ому|их|ых|ую|юю|ая|яя|ою|ею)$")

PARTICIPLE = re.compile(u"((ивш|ывш|ующ)|((?<=[ая])(ем|нн|вш|ющ|щ)))$")

VERB = re.compile(u"((ила|ыла|ена|ейте|уйте|ите|или|ыли|ей|уй|ил|ыл|им|ым|ен|ило|ыло|ено|ят|ует|уют|ит|ыт|ены|ить|ыть|ишь|ую|ю)|((?<=[ая])(ла|на|ете|йте|ли|й|л|ем|н|ло|но|ет|ют|ны|ть|ешь|нно)))$")

NOUN = re.compile(u"(а|ев|ов|ие|ье|е|иями|ями|ами|еи|ии|и|ией|ей|ой|ий|й|иям|ям|ием|ем|ам|ом|о|у|ах|иях|ях|ы|ь|ию|ью|ю|ия|ья|я)$")

RVRE = re.compile(u"^(.\*?[аеиоуыэюя])(.\*)$")

DERIVATIONAL = re.compile(u".\*[^аеиоуыэюя]+[аеиоуыэюя].\*ость?$")

DER = re.compile(u"ость?$")

SUPERLATIVE = re.compile(u"(ейше|ейш)$")

I = re.compile(u"и$")

P = re.compile(u"ь$")

NN = re.compile(u"нн$")

@staticmethod # здесь не хватало staticmethod'а

def stem(word):

word = word.lower()

word = word.replace(u'ё', u'е')

m = re.match(Porter.RVRE, word)

if m and m.groups():

pre = m.group(1)

rv = m.group(2)

temp = Porter.PERFECTIVEGROUND.sub('', rv, 1)

if temp != rv:

rv = temp

rv = Porter.REFLEXIVE.sub('', rv, 1)

temp = Porter.ADJECTIVE.sub('', rv, 1)

if temp != rv:

rv = temp

rv = Porter.PARTICIPLE.sub('', rv, 1)

else:

temp = Porter.VERB.sub('', rv, 1)

if temp == rv:

rv = Porter.NOUN.sub('', rv, 1)

else:

rv = temp

rv = Porter.I.sub('', rv, 1)

if re.match(Porter.DERIVATIONAL, rv):

rv = Porter.DER.sub('', rv, 1)

temp = Porter.P.sub('', rv, 1)

if temp == rv:

rv = Porter.SUPERLATIVE.sub('', rv, 1)

rv = Porter.NN.sub(u'н', rv, 1)

else:

rv = temp

word = pre+rv

return word

# импортируем Pandas и Numpy

import re

import pandas as pd

import numpy as np

df = pd.read\_csv('X\_train.csv')

pd.set\_option('display.max\_columns', 20)

pd.set\_option('display.max\_rows', 1000)

cols = list(df.columns.values)

df\_nes=df[['sku', 'categoryLevel2Id' , 'brandId', 'userName', 'reting' , 'comment' , 'commentNegative', 'commentPositive']]

#df\_nes.head(n=100)

df\_nes\_learn=df\_nes.head(n=10000)

df\_nes\_test=df\_nes.tail(n=5550)

#df\_nes\_test.head()

#df\_nes\_learn.head()

abs=['first\_lane']

for row in list(df\_nes\_learn.itertuples(index=True, name='Pandas')):

#for row in df\_nes\_learn.itertuples(index=True, name='Pandas'):

tokens = get\_tokens(getattr(row,'comment'))

for token in tokens:

if re.match(r'\w+', token):

stemmed = Porter.stem(token)

for x in abs:

if x != stemmed:

abs.append(stemmed)

# print(token, '->', stemmed)

# а можно сразу так:

#stems = [Porter.stem(token) for token in tokens if re.match(r'\w+', token)]

#print('оригинальный текст:')

#print(row.comment)

#print()

#print('стемы:')

#print(' '.join(stems))

#print('\n' + '-' \* 60 + '\n')

# print(' ')

print(abs)