使用Bag-of-Words对示例文本进行特征向量化：

from sklearn.feature\_extraction.text import CountVectorizer

import numpy as np

sent1='The cat is walking in the bedroom.'

sent2='A dog was running across the kitchen.'

count\_vec=CountVectorizer()

sentences=[sent1,sent2]

print(count\_vec.fit\_transform(sentences).toarray())

print(count\_vec.get\_feature\_names())

输出会得到有省略号的一组数，看着很不爽

np.set\_printoptions(threshold=np.inf)

通过这条指令可以让省略号的数显示出来。最后输出的是特征向量化的表示和向量各个维度的特征含义。

然后用NLTK对文本进行语言学分析

Import nltk

tokens\_1=nltk.word\_tokenize(sent1)

print(tokens\_1)

对句子进行词汇分割和正规化，有些情况如aren’t和n’t；或者是I’m要分割I和m

后面看了如何寻找各个词汇最原始的词根，还有标注啥的，我NLTK的东西还没下完，就不写了。

nltk.download()

这个是真的慢啊。

再就是词向量（Word2Vec）技术，学习了一个简单的代码：

from sklearn.datasets import fetch\_20newsgroups

news=fetch\_20newsgroups(subset=’all’)

X,y=news.data,news.target

from bs4 import BeautifulSoup

import nltk,re

def news\_to\_sentences(news):

news\_text=BeautifulSoup(news).get\_text()

tokenizer=nltk.data.load(‘tokenizers/punkt/english.pickle’)

raw\_sentences=tokenizer.tokenize(new\_text)

sentences=[]

for sent in raw\_sentences:

sentences.append(re.sub(‘[^a-zA-Z]’,’’,sent.lower().strip()).split())

return sentences

sentences=[]

for x in X:

sentences+=news\_to\_sentences(x)

from genism.models import word2vec

num\_features=300

min\_word\_count=20

num\_workers=2

context=5

downsampling=le-3

from genism.models inport word2vec

module=word2vec.Wordvec(sentences,workers=num\_workers,\size=num\_features,min\_count=min\_word\_count,\window=context,sample=downsampling)

model.init\_sims(replace=true)

model.most\_similar(‘morning’)

model.most\_similar(‘email’)

这个程序先是导入了20类新闻文本抓取器并下载数据，然后定义一个函数，将每条新闻中句子逐一玻璃，并返回一个句子的列表，然后用他们训练，运用word2vec训练词向量模型，最后寻找训练文本中与morning和email最相关的10个词汇。