Brody_Vogel_HW_6

October 11, 2018

The models performed very similarly on the toy data I provided. The 'documents' I fed in were really just sentences about my cats and my roommate, Daniel. The only difference I noticed is that, when I asked for 3 topics, LSI gave me one topic mostly about the words 'cats' and 'cat', one topic about 'cat' and 'daniel', and a third about the verb 'likes'. But, LDA gave me one topic mostly about 'daniel' and 'cats', one topic about 'cat', 'cats', and 'likes', and a final topic about 'cats' first and then 'daniel'.

So the methods are performing similarly - as one would expect - on this simple text. The results are a little different, though.

In [54]:
```
# Brody Vogel Homework 6

# import everything I might need
from gensim import models, corpora
from nltk import word_tokenize
from nltk.corpus import gutenberg, stopwords
from gensim.test.utils import common_dictionary, common_corpus
from collections import defaultdict
import string
```

In [83]:
```
# create a very simple custom corpus from these sentences
    # each sentence is a 'document' of terms
texts = ['This sentence is about the two cats that live with my roommate, Daniel, and
        'I like the two cats; one cat is Eddie, the other is Sollie; Daniel also loves
        'One cat likes hair ties; the other cat likes straws.',
        'Daniel likes books, and the cats.',
        'The two cats are pretty clean.',
        'My roommate, Daniel, is also neat.',
        'The cats also do a good job of getting rid of bugs, which Daniel and I appre
        'The cat, Eddie, eats a lot; the cat, Sollie, is always running around.',
        'The cats sleep with Daniel in his room.',
        'Daniel does not seem to mind.',
        'I like my roommate and my cat.']

# get rid of the stopwords and punctuation
stop = stopwords.words('english') + list(string.punctuation)
texts = [[word for word in word_tokenize(sent.lower())
          if word not in stop] for sent in texts]
```

In [84]:
```
# create a dictionary for the words in the sentences
dictionary = corpora.Dictionary(texts)

# here is the actual corpus
corpus = [dictionary.doc2bow(text) for text in texts]

# the models should figure out the corpus is about Daniel and my cats
```

In [95]:
```
# train the LSI model
lsi_model = models.LsiModel(corpus, id2word = dictionary, num_topics = 3)

# apply the model to my corpus
corpus_lsi = lsi_model[corpus]

# get three topics; the first two turn out to be mostly about 'cat', 'cats' and 'dani
#while the third picks up the verb 'likes'
print(lsi_model.print_topics(3))

print('\n')

# this tells us how much the three topic vectors affect each sentence
    # nothing surprising here
    # the first few words of each sentence are printed after
    # the topic scores from that sentence
num = 0
for doc in corpus_lsi:
    print(doc, texts[num][0:3])
    num += 1
```

```
[(0, '0.509*"cats" + 0.437*"cat" + 0.392*"daniel" + 0.229*"also" + 0.224*"two" + 0.203*"one" +


[(0, 1.3584070928842404), (1, -1.0580045299504826), (2, 0.03746592206934092)] ['sentence', 'two
[(0, 3.2051269280045926), (1, -0.4629290736303886), (2, -0.6405458832509097)] ['like', 'two',
[(0, 1.6565768409115638), (1, 2.304825256568126), (2, 1.956380949275553)] ['one', 'cat', 'likes
[(0, 1.1349802474583361), (1, -0.4536032789536676), (2, 0.7909974671604512)] ['daniel', 'likes
[(0, 0.8003642009630354), (1, -0.562410019783367), (2, -0.129047373856441)] ['two', 'cats', 'pi
[(0, 0.7729030946628296), (1, -0.6419340220859222), (2, 0.12237342404390332)] ['roommate', 'dan
[(0, 1.5091990780038596), (1, -1.4459441595794253), (2, 0.3699511984722875)] ['cats', 'also',
[(0, 1.613994060075505), (1, 2.073532970292444), (2, -1.7404998712834787)] ['cat', 'eddie', 'ea
[(0, 0.9839798985466577), (1, -0.8069918615335178), (2, 0.1527409949476248)] ['cats', 'sleep',
[(0, 0.42808081475279447), (1, -0.4042967372697582), (2, 0.16236249465492059)] ['daniel', 'seen
[(0, 0.7202351073565365), (1, 0.5090375257153075), (2, -0.13439331028257595)] ['like', 'roommat
```

In [94]:
```
# this is the same concept, but with LDA instead of LSI

# train the model
```

```
        lda_model = models.LdaModel(corpus, id2word = dictionary, num_topics = 3)

        # apply the model to my corpus
        corpus_lda = lda_model[corpus]

        # get the topics
        print(lda_model.print_topics(3))

        print('\n')

        # look at the effect of each topic vector in each sentence
            # the first few words of each sentence are printed after
            # the topic scores from that sentence
        num = 0
        for doc in corpus_lda:
            print(doc, texts[num][0:3])
            num += 1
```

```
[(0, '0.122*"daniel" + 0.093*"cats" + 0.054*"also" + 0.047*"roommate" + 0.046*"two" + 0.046*"li
```

```
[(0, 0.061913602), (1, 0.88477534), (2, 0.053311054)] ['sentence', 'two', 'cats']
[(0, 0.03437651), (1, 0.93398434), (2, 0.031639133)] ['like', 'two', 'cats']
[(0, 0.038552683), (1, 0.92407566), (2, 0.037371617)] ['one', 'cat', 'likes']
[(0, 0.85371745), (1, 0.074345276), (2, 0.0719373)] ['daniel', 'likes', 'books']
[(0, 0.8565641), (1, 0.0725786), (2, 0.0708573)] ['two', 'cats', 'pretty']
[(0, 0.8554741), (1, 0.07270508), (2, 0.07182078)] ['roommate', 'daniel', 'also']
[(0, 0.0357294), (1, 0.03431831), (2, 0.92995226)] ['cats', 'also', 'good']
[(0, 0.033759885), (1, 0.93256396), (2, 0.0336761)] ['cat', 'eddie', 'eats']
[(0, 0.07528676), (1, 0.070092365), (2, 0.8546208)] ['cats', 'sleep', 'daniel']
[(0, 0.82778704), (1, 0.085130565), (2, 0.08708243)] ['daniel', 'seem', 'mind']
[(0, 0.08777013), (1, 0.8282673), (2, 0.08396263)] ['like', 'roommate', 'cat']
```