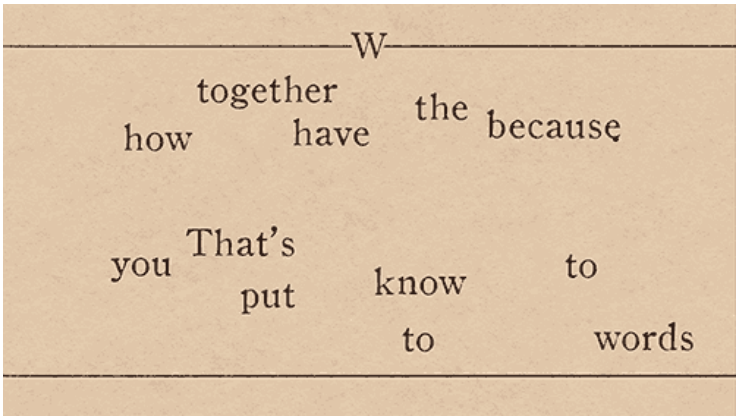
NLP: Extracting the main topics from your dataset using LDA in minutes

Priya Dwivedi Follow
Aug 22, 2018 · 5 min read

Doing cool things with data!



Power of NLP

I recently started learning about Latent Dirichlet Allocation (LDA) for topic modelling and was amazed at how powerful it can be and at the same time quick to run. Topic Modelling is the task of using unsupervised learning to extract the main topics (represented as a set of words) that occur in a collection of documents.

I tested the algorithm on 20 Newsgroup data set which has thousands of news articles from many sections of a news report. In this data set I knew the main news topics before hand and could verify that LDA was correctly identifying them.

The code is quite simply and fast to run. You can find it on Github. I encourage you to pull it and try it.

About LDA

LDA is used to classify text in a document to a particular topic. It builds a topic per document model and words per topic model, modeled as Dirichlet distributions.

- Each document is modeled as a multinomial distribution of topics and each topic is modeled as a multinomial distribution of words.
- LDA assumes that the every chunk of text we feed into it will contain words that are somehow related. Therefore choosing the right corpus of data is crucial.

- It also assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution.

To learn more about LDA please check out this link.

Data set Used

The data set I used is the 20Newsgroup data set. It is available under sklearn data sets and can be easily downloaded as

```
from sklearn.datasets import fetch_20newsgroups
newsgroups_train = fetch_20newsgroups(subset='train', shuffle =
True)
newsgroups_test = fetch_20newsgroups(subset='test', shuffle = True)
```

This data set has the news already grouped into key topics. Which you can get by

```
print(list(newsgroups_train.target_names))
```

There are 20 targets in the data set — *'alt.atheism',*
'comp.graphics',
'comp.os.ms-windows.misc',
'comp.sys.ibm.pc.hardware',
'comp.sys.mac.hardware',
'comp.windows.x',
'misc.forsale',
'rec.autos',
'rec.motorcycles',
'rec.sport.baseball',
'rec.sport.hockey',
'sci.crypt',
'sci.electronics',
'sci.med',
'sci.space',
'soc.religion.christian',
'talk.politics.guns',
'talk.politics.mideast',
'talk.politics.misc',
'talk.religion.misc

Looking visually we can say that this data set has a few broad topics like:

- Science
- Politics
- Sports
- Religion

- Technology etc

Extracting Topics using LDA in Python

1. **Preprocessing the raw text**

This involves the following:

- **Tokenization**: Split the text into sentences and the sentences into words. Lowercase the words and remove punctuation.
- Words that have fewer than 3 characters are removed.
- All **stopwords** are removed.
- Words are **lemmatized** — words in third person are changed to first person and verbs in past and future tenses are changed into present.
- Words are **stemmed** — words are reduced to their root form.

We use the NLTK and gensim libraries to perform the preprocessing

```
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text,
pos='v'))

# Tokenize and lemmatize
def preprocess(text):
    result=[]
    for token in gensim.utils.simple_preprocess(text) :
        if token not in gensim.parsing.preprocessing.STOPWORDS and
len(token) > 3:
            result.append(lemmatize_stemming(token))

    return result
```

The resulting text looks like this:

```
Original document:
['This', 'disk', 'has', 'failed', 'many', 'times.', 'I', 'would',
'like', 'to', 'get', 'it', 'replaced.']


Tokenized and lemmatized document:
['disk', 'fail', 'time', 'like', 'replac']
```

2. Converting text to bag of words

Prior to topic modelling, we convert the tokenized and lemmatized text to a bag of words — which you can think of as a dictionary where the key is the word and value is the number of times that word occurs in the entire corpus.

```
dictionary = gensim.corpora.Dictionary(processed_docs)
```

We can further filter words that occur very few times or occur very frequently.

Now for each pre-processed document we use the dictionary object just created to convert that document into a bag of words. i.e for each document we create a dictionary reporting how many words and how many times those words appear.

```
bow_corpus = [dictionary.doc2bow(doc) for doc in processed_docs]
```

The results look like:

```
Word 453 ("exampl") appears 1 time.
Word 476 ("jew") appears 1 time.
Word 480 ("lead") appears 1 time.
Word 482 ("littl") appears 3 time.
Word 520 ("wors") appears 2 time.
Word 721 ("keith") appears 3 time.
Word 732 ("punish") appears 1 time.
Word 803 ("california") appears 1 time.
Word 859 ("institut") appears 1 time.
```

3. Running LDA

This is actually quite simple as we can use the gensim LDA model. We need to specify how many topics are there in the data set. Lets say we start with 8 unique topics. Num of passes is the number of training passes over the document.

```
lda_model =  gensim.models.LdaMulticore(bow_corpus,
                                        num_topics = 8,
                                        id2word = dictionary,
                                        passes = 10,
                                        workers = 2)
```

Results and interpreting them

That's it! The model is built. Now let's interpret it and see if results make sense.

The output from the model is a 8 topics each categorized by a series of words. LDA model doesn't give a topic name to those words and it is for us humans to interpret them. See below sample output from the model and how "I" have assigned potential topics to these words.

```
Topic 1: Possibly Graphics Cards
Words: "drive" , "sale" , "driver" , *"wire" , "card" , "graphic" ,
"price" , "appl" ,"softwar", "monitor"

Topic 2: Possibly Space
Words: "space","nasa" , "drive" , "scsi" , "orbit" , "launch"
,"data" ,"control" , "earth" ,"moon"

Topic 3: Possibly Sports
Words: "game" , "team" , "play" , "player" , "hockey" , season" ,
"pitt" , "score" , "leagu" , "pittsburgh"
```

```
Topic 4: Possibly Politics
Words: "armenian" , "public" , "govern" , "turkish", "columbia" ,
"nation", "presid" , "turk" , "american", "group"

Topic 5: Possibly Gun Violence
Words: "kill" , "bike", "live" , "leav" , "weapon" , "happen" ,
*"gun", "crime" , "car" , "hand"
```

Check out the github code to look at all the topics and play with the model to increase decrease the number of topics.

Observations:

- The model did impressively well in extracting the unique topics in the data set which we can confirm given we know the target names
- The model runs very quickly. I could extract topics from data set in minutes
- It does assume that there are distinct topics in the data set. So if the data set is a bunch of random tweets than the model results may not be as interpretable.

Improvements for the future

I am very intrigued by this post on Guided LDA and would love to try it out.

I have my own deep learning consultancy and love to work on interesting problems. I have helped many startups deploy innovative AI based solutions. Check us out at — http://deeplearninganalytics.org/.

You can also see my other writings at: https://medium.com/@priya.dwivedi

If you have a project that we can collaborate on, then please contact me through my website or at info@deeplearninganalytics.org

References

- A big thanks to Udacity and particularly their NLP nanodegree for making learning fun!
- Paper on LDA

Data Science | Artificial Intelligence | NLP | Machine Learning | Topic Modeling