0105_topic_modelling

December 18, 2018

1 Topic modelling (dividing documents into topic groups) with Gensim

Gensim is a library that can sort documents into groups. It is an 'unsupervised' method, meaning that documents do not need to be pre-labbeled.

Here we will use gensim to group titles or keywords from PubMed scientific paper references.

Gensim is not part of the standard Anaconda Python installation, but it may be installed from the command line with:

conda install gensim

If you are not using an Anaconda installation of Python then you can install with pip:

pip install gensim

1.1 Import libraries

```
In [1]: import pandas as pd
        import gensim
        import nltk
        from nltk.corpus import stopwords
```

1.2 Load data

Now we will load our data (the script below loads from a local copy of the imdb movie review database, but instructions are also given for downloading from the internet).

In this example we will use a portion of a large dataset of pubmed medical paper titles and keywords. The full data set may be downloaded from the link below (1.2GB download).

https://www.kaggle.com/hsrobo/titlebased-semantic-subject-indexing#pubmed.csv

The code section below downloads a 50k subset of the full data. It will download and save locally.

```
In [2]: ## LOAD 50k DATA SET FROM INTERNET

        file_location = 'https://gitlab.com/michaelallen1966/1804_python_healthcare_wordpress'
            '/raw/master/jupyter_notebooks/pubmed_50k.csv'
        data = pd.read_csv(file_location)
        # save to current directory
        data.to_csv('pubmed_50k.csv', index=False)
```

```
        # If you already have the data locally, you may load with:
        # data = pd.read_csv('pubmed_50k.csv')
```

1.3 Clean data

We will clean data by applying the following steps.

- convert all text to lower case
- divide strings/sentences into individual words ('tokenize')
- remove non-text words
- remove 'stop words' (commonly occurring words that have little value in model)

In the example here we will take the keywords (called #labels' for each paper)

```
In [3]: stops = set(stopwords.words("english"))

        # Define function to clean text
        def pre_process_text(X):
            cleaned_X = []
            for raw_text in X:
                    # Convert to lower case
                text = raw_text.lower()

                # Tokenize
                tokens = nltk.word_tokenize(text)

                # Keep only words (removes punctuation + numbers)
                token_words = [w for w in tokens if w.isalpha()]

                # Remove stop words
                meaningful_words = [w for w in token_words if not w in stops]

                cleaned_X.append(meaningful_words)
            return cleaned_X


        # Clean text
        raw_text = list(data['labels'])
        processed_text = pre_process_text(raw_text)
```

1.4 Create topic model

The following will create our topic model. We will divide the references into 50 different topic areas. This may take a few minutes.

```
In [4]: dictionary = gensim.corpora.Dictionary(processed_text)
        corpus = [dictionary.doc2bow(text) for text in processed_text]
        model = gensim.models.LdaModel(corpus=corpus,
```

```
                                        id2word=dictionary,
                                        num_topics=50,
                                        passes=10)
```

1.5 Show topics

```
In [5]: top_topics = model.top_topics(corpus)
```

When we look at the first topic, we see that keywords are largely related to molecular biology.

```
In [6]: # Print the keywords for the first topic
        from pprint import pprint # makes the output easier to read
        pprint(top_topics[0])
```

```
([(0.14238602, 'sequence'),
  (0.07095096, 'molecular'),
  (0.068985716, 'acid'),
  (0.06632707, 'dna'),
  (0.052034967, 'amino'),
  (0.045958135, 'data'),
  (0.03165856, 'proteins'),
  (0.03090581, 'base'),
  (0.02128076, 'rna'),
  (0.018465547, 'genetic'),
  (0.01786064, 'bacterial'),
  (0.017836036, 'viral'),
  (0.014751778, 'animals'),
  (0.013531377, 'nucleic'),
  (0.013407393, 'genes'),
  (0.013192138, 'analysis'),
  (0.012144415, 'humans'),
  (0.011643986, 'cloning'),
  (0.011388958, 'phylogeny'),
  (0.011024448, 'protein')],
 -1.8900328727623419)
```

If we look at another topic (topic 10) we see keywords that are associated with cardiac surgical procedures.

```
In [10]: pprint(top_topics[10])
```

```
([(0.056376483, 'outcome'),
  (0.0558772, 'treatment'),
  (0.05120605, 'humans'),
  (0.03075589, 'surgical'),
  (0.030704997, 'complications'),
  (0.028222634, 'postoperative'),
  (0.026755776, 'coronary'),
```

```
  (0.02651835, 'tomography'),
  (0.026010627, 'heart'),
  (0.023422625, 'male'),
  (0.022875749, 'computed'),
  (0.021913974, 'studies'),
  (0.02180667, 'procedures'),
  (0.019811377, 'myocardial'),
  (0.01757028, 'cardiac'),
  (0.015987962, 'artery'),
  (0.015796969, 'female'),
  (0.013579166, 'prosthesis'),
  (0.012545248, 'valve'),
  (0.01180034, 'history')],
 -3.1752669709698886)
```

1.6 Show topics present in each document

Each document may contain one or more topics. The first paper is highlighted as containing topics 4, 8, 19 and 40.

```
In [14]: model[corpus[0]]
```

```
Out[14]: [(4, 0.0967338), (8, 0.27275458), (19, 0.4578644), (40, 0.09598056)]
```