TOPIC MODELING ON ASSOCIATED PRESS ARTICLES

Siddhesh Tiwari

UNIVERSITY OF ILLINOIS AT CHICAGO

UIN: 657796780

Contents

INTRODUCTION

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents[1]. Topic models are a suite of algorithms that uncover the hidden thematic structure in document collections. These algorithms help us develop new ways to search, browse and summarize large archives of texts. It provide a simple way to analyze large volumes of unlabeled text. A "topic" consists of a cluster of words that frequently occur together. Using contextual clues, topic models can connect words with similar meanings and distinguish between uses of words with multiple meanings. Topic modeling is a form of text mining, a way of identifying patterns in a corpus. You take your corpus and run it through a tool which groups words across the corpus into ‘topics’.

There are various algorithms that can be used for Topic Modeling like:

- Explicit semantic analysis
- Latent semantic analysis
- Latent Dirichlet allocation
- Hierarchical Dirichlet process
- Non-negative matrix factorization

Topic Modeling is being used extensively in area of digital media to organize different articles, news, blogs etc. under proper topic for easy use by consumer. My attempt in this project was to understand and replicate how this task is done by publisher agencies by using Topic Modeling.

[1] https://en.wikipedia.org/wiki/Topic_model

PROJECT OVERVIEW

Dataset

The project deals with building a Topic Model on Associated Press articles. Associated Press is world's oldest and largest news gathering organization. Data set is in form of a corpus of 2246 documents in a single text file. Articles in corpus belong to various domain like Business, Banking, Politics, Education, War etc. giving a nice mix of topics to train the model.

Data set - http://www.cs.princeton.edu/~blei/lda-c/

Tools & Algorithm

- **Python** - a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

- **Gensim**
  - An open-source vector space modeling and topic modeling toolkit, implemented in the Python programming language. It uses NumPy, SciPy and optionally Cython for performance. It is specifically intended for handling large text collections, using efficient online, incremental algorithms.
  - Gensim includes implementations of tf-idf, random projections, word2vec and document2vec algorithms, hierarchical Dirichlet processes (HDP), latent semantic analysis (LSA) and latent Dirichlet allocation (LDA), including distributed parallel versions.

- **pyLDAvis -** Python library to visualize topics generated from LDA.

- **Latent Dirichlet Allocation -** In natural language processing, Latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. In LDA, each document may be viewed as a mixture of various topics. This is similar to probabilistic latent semantic analysis (pLSA), except that in LDA the topic distribution is assumed to have a Dirichlet prior. In practice, this results in more reasonable mixtures of topics in a document. It has been noted, however, that the pLSA model is equivalent to the LDA model under a uniform Dirichlet prior distribution.[2]

[2] https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

IMPLEMENTATION

Pre-Processing

- Divided single text file containing the data into a corpus of 2246 text documents.
- Remove Non-ASCII (eg: ¶,®,§) values from all the documents
- Tokenize and remove stopwords from each document and store in a list containing each document in form of tokens.
- Remove tokens in each document with length less than 5 and frequency less than 2.
- Create a dictionary of unique tokens
- Create a document-term matrix representing document and frequency of occurrence of the unique tokens in each document.
- Divide data into training and testing set.

Modeling & Evaluation

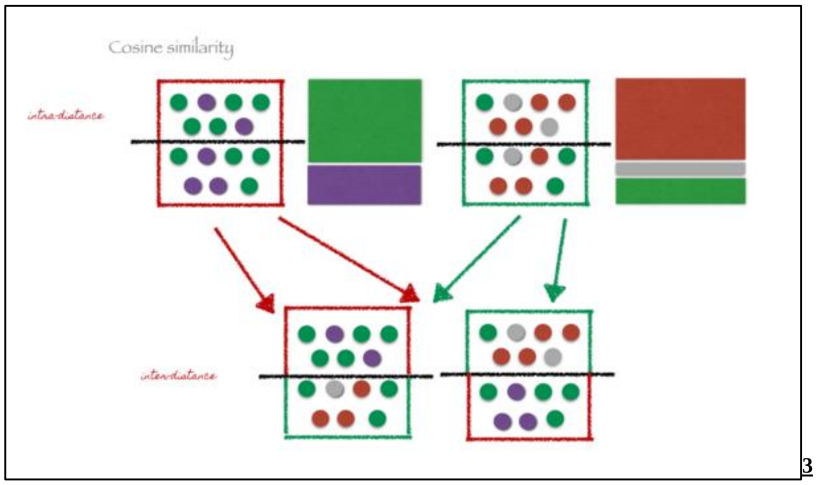
Gensim library contains implementation of Latent Dirichlet allocation (LDA). So I had to feed the LDA implementation with required input and train the model for different set of inputs and then evaluate different models.

The input that LDA required were:

- Corpus representing the document-term matrix
- Dictionary of unique tokens
- Number of Topics
- Number of iterations

After modeling the training set the model was evaluated on test set using following approach:

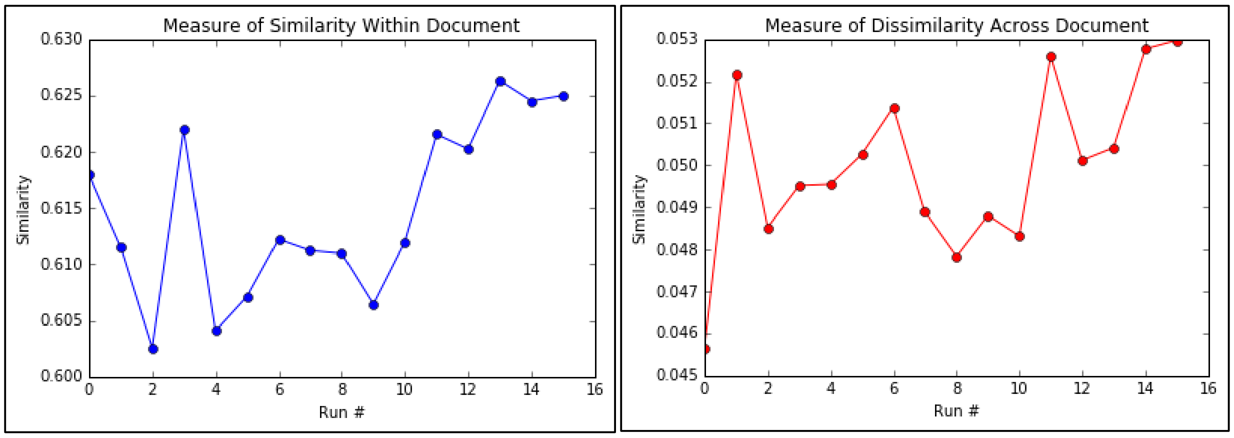
1. Split each testing documents into two parts.
2. Compute average cosine similarity between halves of same documents with respect to topics in each present in each halves. This Metric should be higher.
3. Compute average cosine similarity between halves of different documents. This metric should be small as different documents will have different distribution topics.



[3]

[3] http://chdoig.github.io/pygotham-topic-modeling/#/3/15

I trained 16 models on different combination of inputs and evaluated these models based on above mentioned approach. Following are the results obtained after evaluation:



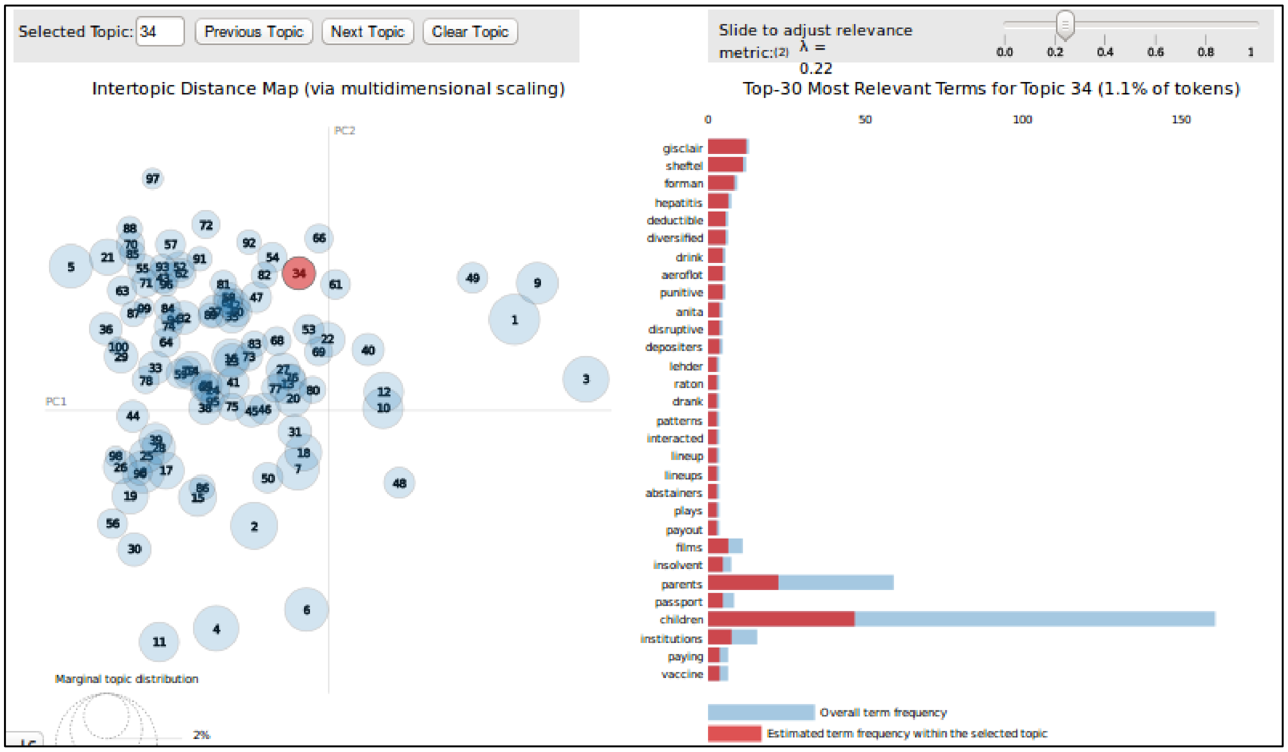
Model Evaluation

Based on analysis from above evaluation it can be seen that model #14 gives the best performance by having the highest topic similarity within document and a low topic dissimilarity across documents.

<u>Visualization</u>

To visualize the best model pyLDAvis package was used whose sole purpose is to visualize LDA models. The input that this package requires are:

- Topic Term Distribution – This is distribution of terms within a document.
- Document Topic Distribution – This is distribution of various topics within each document.
- Document Length, Dictionary, Document Term Frequency



Selected Topic: 34

Slide to adjust relevance metric:(2) λ = 0.22

Topic Model Visualization

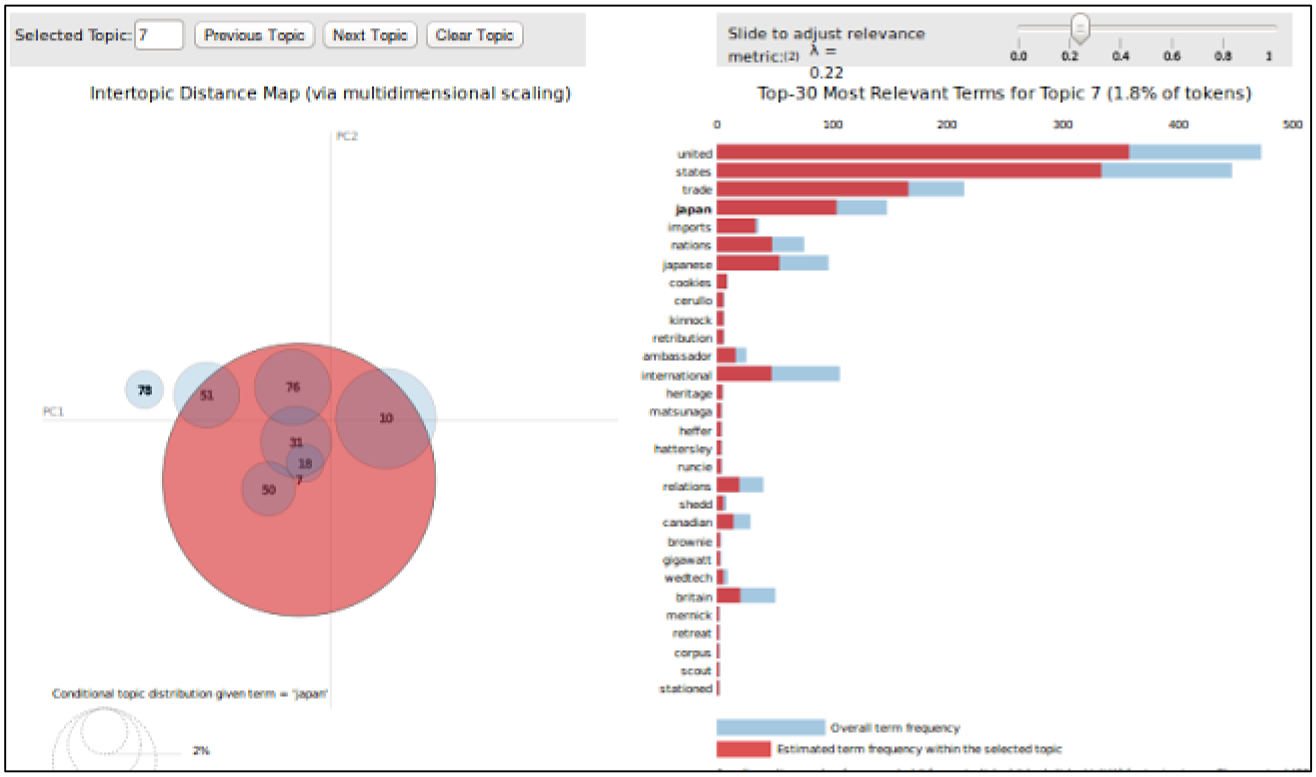
The visualization has 4 parts:

1. **Scatterplot**
   - ➢ Shows number of topics and distance between them. Topics which are cluttered on the top-left corner are very similar whereas topics on top right corner are different. Distance is proportional to similarity between topics.
   - ➢ The radius of topic circle gives the prevalence of that topic within the corpus.

2. **Frequency Distribution –** After clicking a topic from scatterplot we get its term frequency distribution.
   - ➢ Red bar gives the frequency of term within topic.
   - ➢ Blue bar gives the frequency of term in the corpus.

3. **Lamda Slider(λ) –** λ is ratio of Frequency of term within topic/Frequency of term in corpus. Sliding the λ level gives the relevant words specific within the topic. If you slide λ close to 1 we will get most frequent terms in the topic whereas if we move it close to 0 we will get terms very specific to that topic.

4. **Term Topic Prevalence –** By clicking a particular term from the frequency distribution we can get the topic in which this term is most prevalent as shown below:



CONCLUSION

I believe that the work that I have conducted above can be used successfully to extract topics from textual data available at publishing agencies and online reading apps to organize their data.

This work can be used for following application in future work:

- Topic similarity among documents can be used to organize documents in news portals, online library and research paper publishers.
- Topic modeling can be used to build recommendation engine for online magazines by using topic similarity between documents read by users.