Midterm Project Report

Jason Booth Alexander Duffy Mary Forde

Abstract

The 20 newsgroups dataset includes almost 20,000 documents split roughly evenly among 20 different newsgroups each corresponding to a different topic (Figure 1). This project involved investigating and visualizing the dataset using machine learning and other tools. Several embeddings of the corpus were created such as bag-of-words, tfidf, and doc2vec. These embedding were used to create an LDA model and clustering was used to validate results. Overall it was found that the documents were able to be clustered by topic and that preprocessing with Spacy produced slightly better results than preprocessing with NLTK.

Methods

The project was separated into a frontend and backend portion to enable quick development and easy communication between members via APIs. The backend was responsible for preprocessing, creation of embeddings, and training of the machine learning models. The frontend was responsible for creating visualizations of statistics and results. All functionality was separated into separate scripts so that a single file could pull in the modules needed and define a pipeline for each run. A main focus was to make everything extremely modular and not resource intensive. This was accomplished by using good object-oriented practices and allowing each step to be saved to or read from an external file using pickle. This allows the team to fix bugs and create new visualizations without having to wait to train a model or preprocess a huge corpus each time, which greatly increased development time. This generalized and decoupled framework is usable as a library for other datasets, and is scalable for much larger corpuses.

The backend used methods from NLTK and Spacy for preprocessing the corpus. These are two industry leading nlp libraries that offer similar functionality and are used across research and industry (Omran & Treude, 2017). Spacy offers a slightly more intuitive interface and API, but both have large amounts of community support and were able to be integrated quickly into the project. An abstract base class was created for tokenization to allow for the easy creation of two embedding spaces to use in future steps.

A major consideration for the backend was performance. To ensure that the project would be scalable, Apache Spark was used during the tokenization step. Tokenization is embarrassingly parallel, meaning that it is able to be done with no interaction between cores or nodes. Apache spark is a framework that uses functional programming to abstract away the computation model (Apache, n.d.). Therefore the tokenization and preprocessing steps are able to be run on a single machine or across several thousands of nodes with no changes to the code. All models were trained over multiple cores to further increase the speed of the training step.

Embeddings were used to convert the textual data into a numerical representation to be analyzed. K-means was used as a validation step for the embeddings and trained representations of the corpus. K-means attempts to find unlabelled clusters in the data. If the representations reflect the corpus well, the clusters should reflect the labelled topics in the dataset. The fit of the clusters is assessed using a normalized mutual information score, which is a measure of the mutual dependence between two variables. Scores range from 0 to 1 with 1 representing complete accuracy.

The individuals assigned to the frontend were able to use synthetic data to develop their visualizations, and later integrate into the backend to use the real data. This enabled fast development even at the beginning of the project when there were no results to visualize. All visualizations are able to read in data from a file and prevent the retraining of models for each graph produced.

The statistics of the dataset were visualized in several different ways. Boxplots of the number of words and number of characters in a document (Figure 2). Also a plot of word frequency vs inverse document frequency was created to visualize the difference between these two metrics (Figure 3).

Doc2vec was visualized in several ways. The first visualization involved taking the vector representations of documents outputted by the doc2vec model and using PCA to move them to two dimensions and plot them based on similarity (figure 8). The second visualization involved inputting a document that would then be preprocessed in an identical manner as the documents in the corpus and returning the 10 most similar documents already in the corpus (Figure 9). Another visualization was created by training doc2vec by topic rather than by document. This involves labelling each document by topic only, and results in 20 document vectors with each corresponding to a topic. A visualization of the cosine similarity between these vectors enables a visualization of the similarity between topics across all documents (Figure 10).

The group then trained LDA models. In order to visualize the resulting information, there were four methods that each were used four times (figures 4-7) , one for each model trained: a word cloud, word weight bar graph, pyLDAvis, and finally a t-SNE visualization. The word cloud takes the LDA model's top ten most relevant topic words by word weight for each topic, and sizes the words based on that weighting. The word weight graph also shows the top ten words and their weighting but provides the numbers in a bar graph. The pyLDAvis visualization takes the LDA model topics, plots their relative distances, and interactively shows the term frequency for terms in each topic and overall.

The group used k-means clustering to further visualize and understand the models that had been trained. The clusters were visualized by reducing dimensionality through PCA and graphing the results (Figure 11). The K-means analysis also included visualizing SSE vs cluster number. This method is based on the elbow method to determine the optimal number of clusters for a dataset (Figure 12). Lastly a visualization of the nmi scores is used to analyze the overall results (Figure 13, 14).

Results

It's clear from Figure 13 and Figure 14 that the most accurate method the group used was to preprocess with NLTK then use BoW and an LDA model. NLTK has better or equal accuracy across all processing types. BoW and LDA model combination do best across both kinds of preprocessing.

The embeddings were able to discriminate between topics and create a clean separation of cluster using t-SNE. Word cloud and word weight visualizations were able to approximately discern topics that seem to correspond to the true categories. The elbow method was inconclusive in determining the best number of clusters, although a dip near 6 corresponds to the aggregated number of topics in 20-newsgroups. Overall the results confirm that the analysis was performed correctly and that the documents can be analyzed by topic.

Citations

Apache Spark™ - Unified Analytics Engine for Big Data. (n.d.). Retrieved from https://spark.apache.org/

Omran, F. N., & Treude, C. (2017). Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments. *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*. doi:10.1109/msr.2017.42

Code

Please see: https://github.com/jaybooth4/DataVizMidterm

Appendix

All visualizations except for the word clouds are interactive and included in the results folder of the submission.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Figure 1: the 20 different newsgroups http://qwone.com/~jason/20Newsgroups/

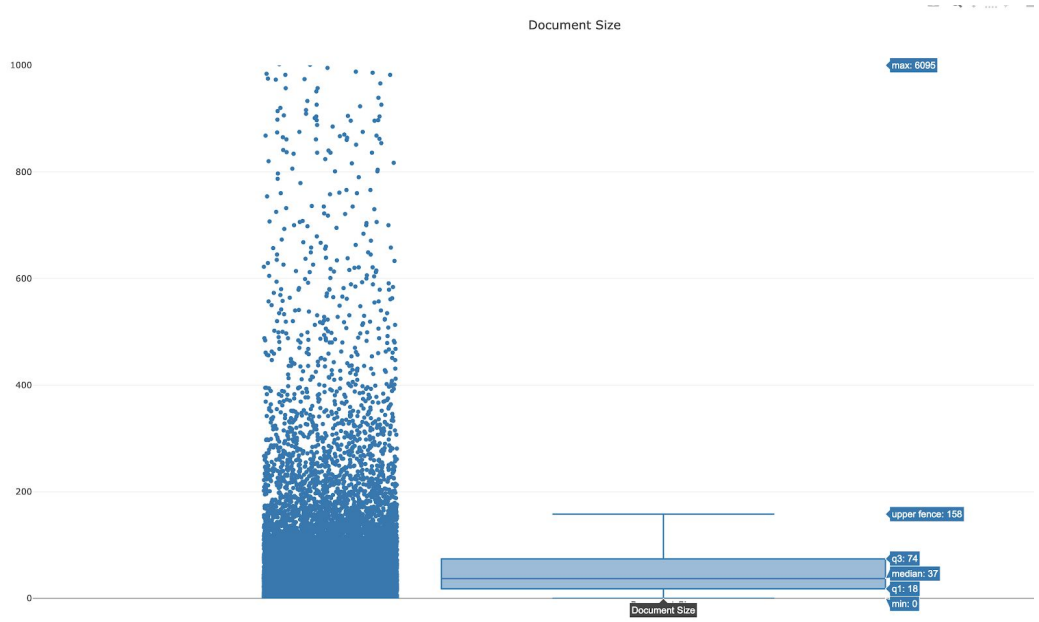

Figure 2: Visualization of document length across all documents in the corpus.

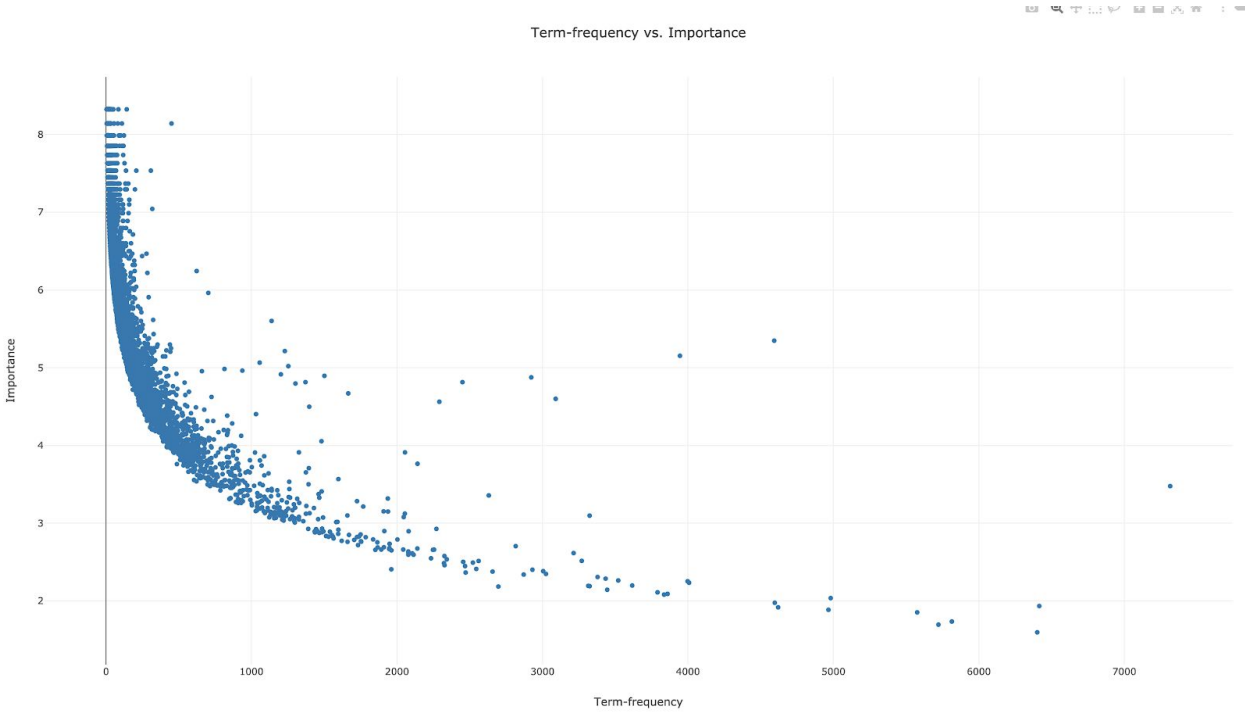

Figure 3: Visualization of term frequency vs importance

Figure 4: LDA model visualization - Spacy & BoW - Word Cloud

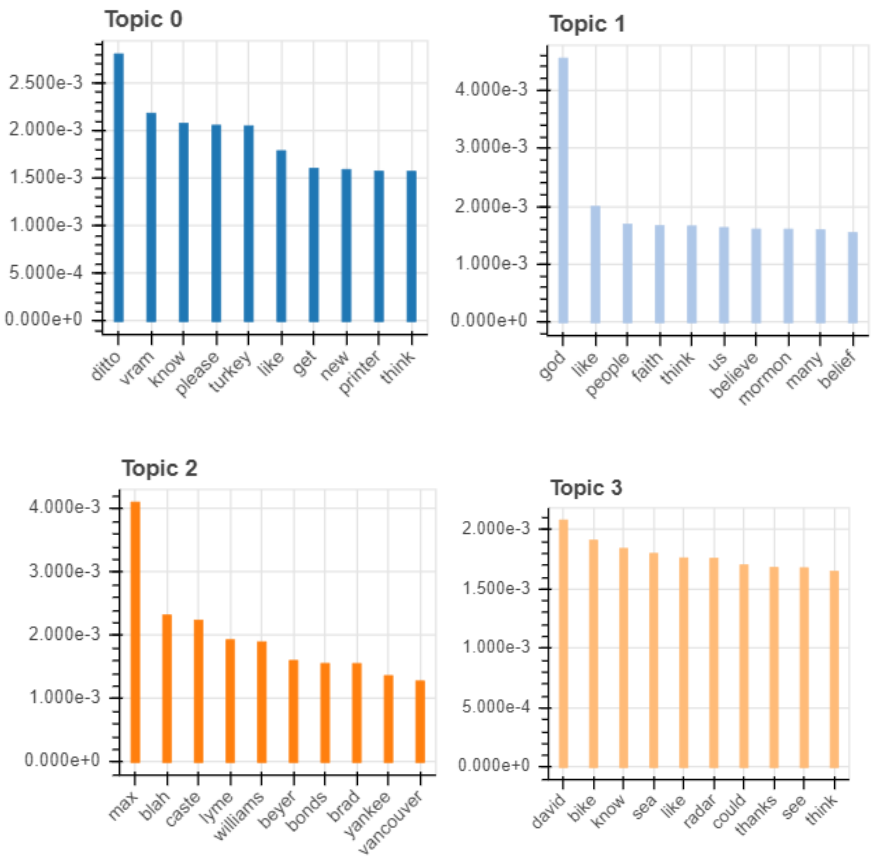

Figure 5: LDA model visualization – NLTK & TFIDF – Word Weight Bar Graph

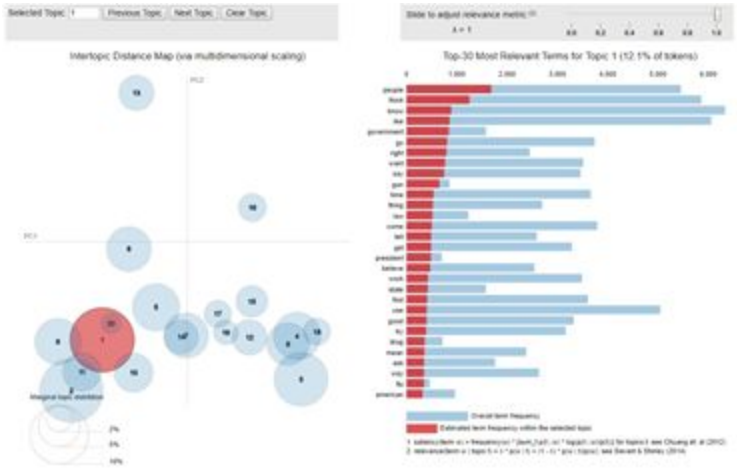

Figure 6: LDA model visualization – Spacy & BoW – pyLDAvis

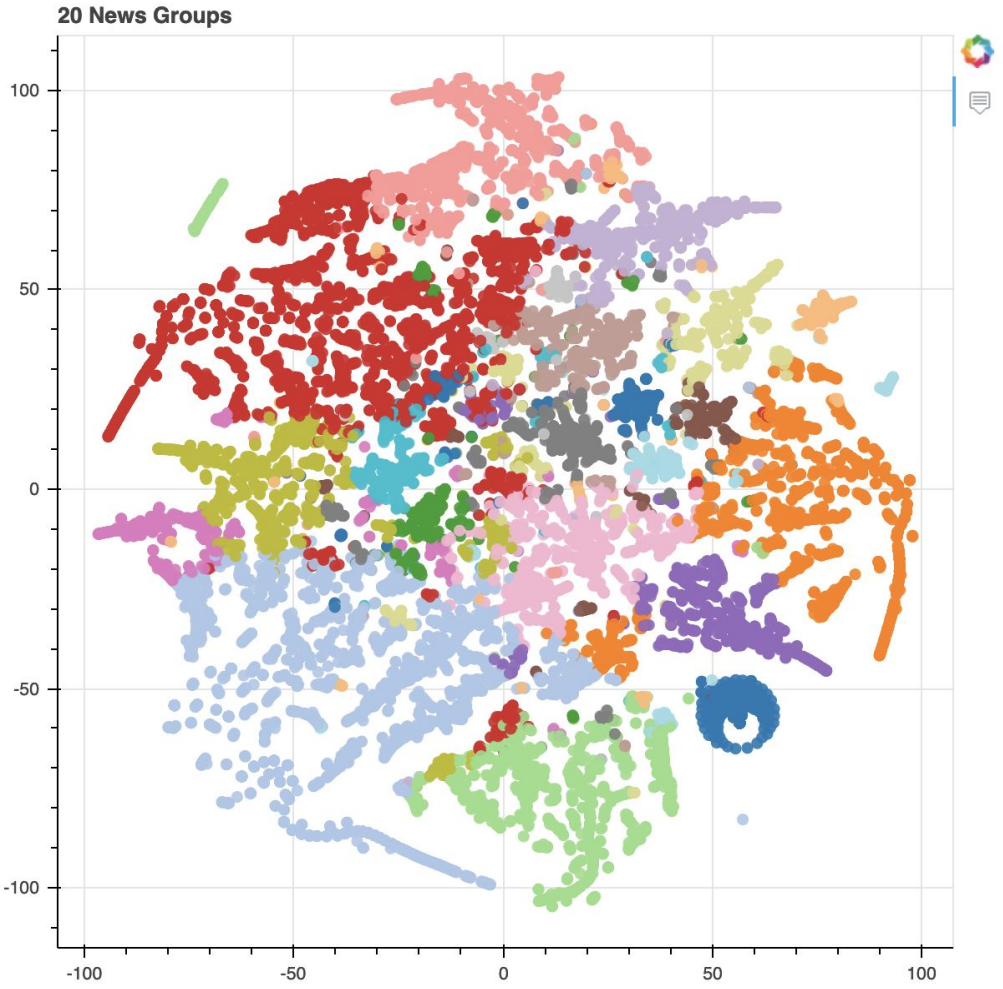

Figure 7: LDA model visualization – NLTK & BoW – t-SNE

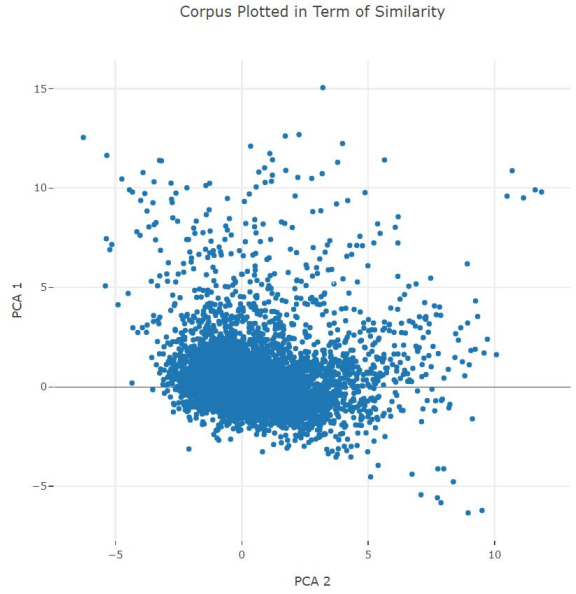

Figure 8: Doc2Vec visualization – NLTK – Corpus Similarity Scatter Plot

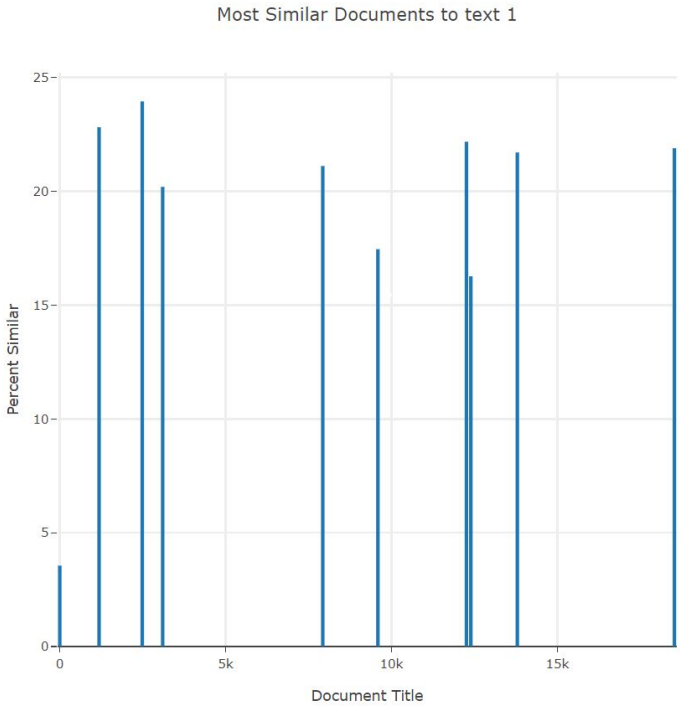

Figure 9: Doc2Vec visualization – Spacy– Most Similar Documents

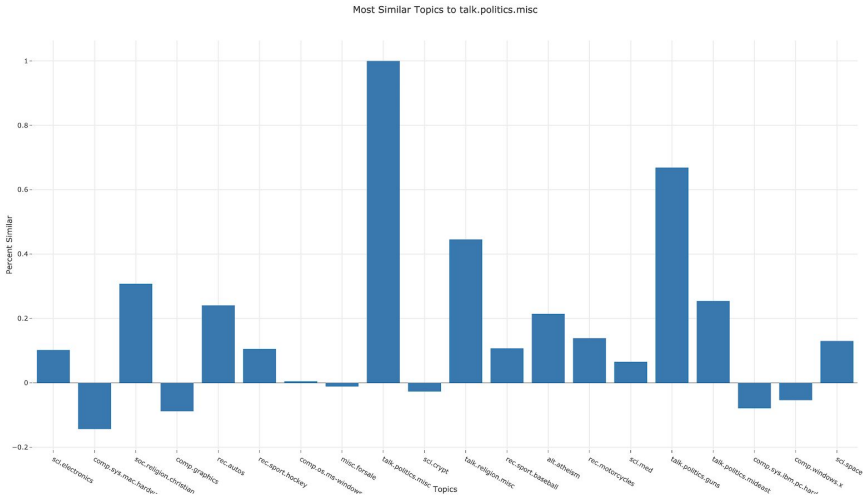

Figure 10: Similarity of topics to talk.politics.misc

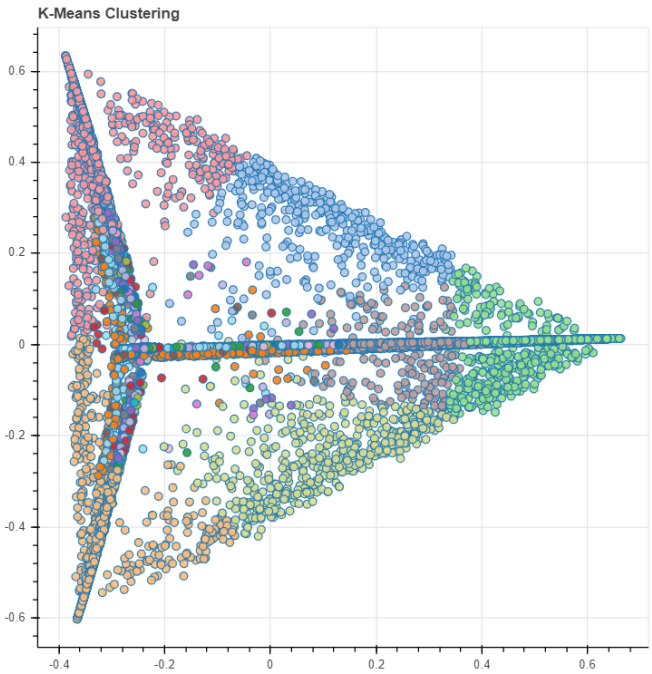

Figure 11: K-means clustering visualization – LDA + Spacy + TFIDF

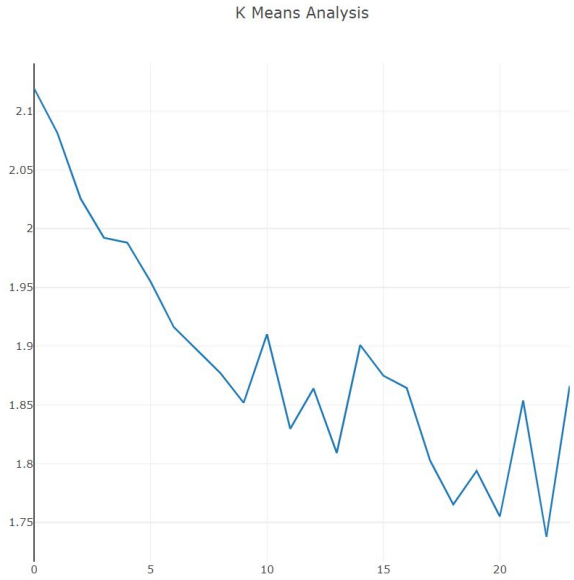

Figure 12: K-means visualization – SSE vs. cluster number

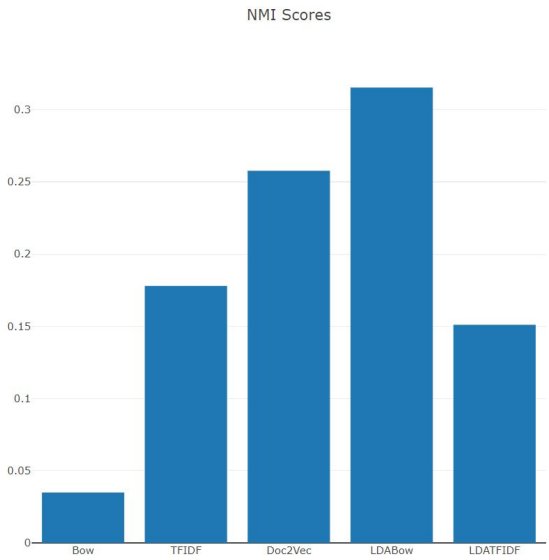

Figure 13: NLTK NMI Scores

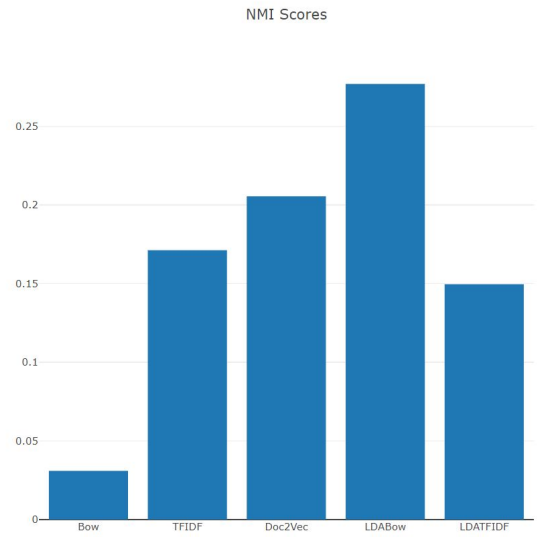

Figure 14: Spacy NMI Score