Topic Modeling in NLP using Latent Dirichlet Allocation (LDA) model

ITCS 5156 - Applied Machine Learning

Prof. Minwoo Lee

Pranjali Mehta
(801255574)

October 12, 2022

Contents

1 Introduction

Topic modeling is a simple way to capture the sense of what a document or a collection of documents is about. Documents are any coherent collection of words, which could be as short as a tweet or as large as an encyclopedia. Topic modeling, in a summary, is a text-mining methodology for identifying topics in documents. Topic modeling is frequently used as a first step in analyzing textual data to acquire a sense of the text's content. This is especially relevant when abstracts/summaries are unavailable, and the text is too voluminous to manually examine within the time span provided.

1.1 Problem Statement

In a world full of documents, it's important to have a mechanism to segregate them with proper Topic assignment. Topic modelling looks to combine topics into a single, understandable structure. It's about grouping topics into broader concepts that make sense for a particular business or issue. Topic Models are very useful for the purpose for document clustering, organizing large blocks of textual data, information retrieval from unstructured text and feature selection. For Example – New York Times are using topic models to boost their user – article recommendation engines. Various professionals are using topic models for recruitment industries where they aim to extract latent features of job descriptions and map them to right candidates. They are being used to organize large datasets of emails, customer reviews, and user social media profiles. To analyze what are the most relevant words for each of the topics and how many broad classifications of topics can be done on given dataset, how closely the topics are interlinked.

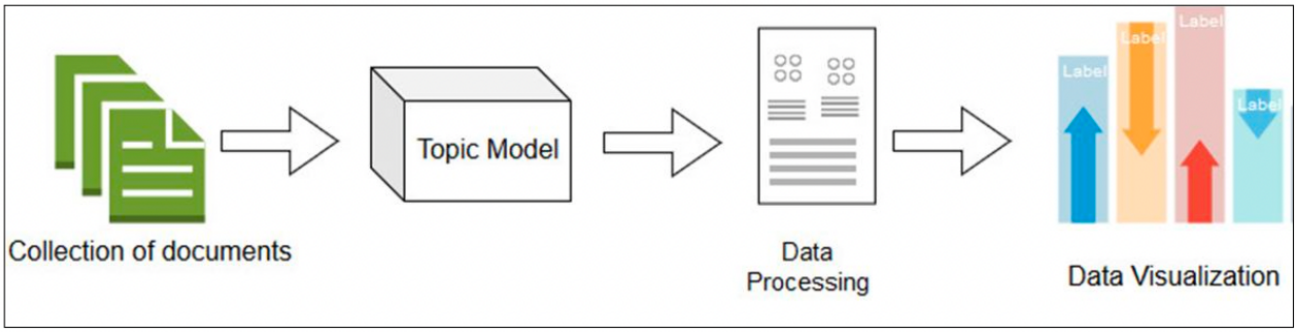

Figure 1. Topic Modeling

1.2 Motivation

Topic modeling automatically discover the topics in a collection of documents. This kind of technique can be helpful to segregate a chunk of un-annotated documents. We can make sense out of undefined data and perform analysis on them. Topic modeling can prove to be useful when we have messed up information. We can also use topic modeling to assign more than one topic

to a document such if it has information which is a combination of more than one domain. We can use the probability scores being assigned by the LDA model to make such decisions.

1.3 Approach

Topic Modeling using Latent Dirichlet Allocation model (LDA) in NLP: LDA method picks up a collection of words from a given document to assign them as topic of that document. LDA starts with assigning any random k number of words as a topic to the document and generate n such topics. After that, let's say we have n topics with k words each, and each word has an associated randomly assigned probability for that topic. The next step for LDA is to iterate over these probabilities and improve them in such a way that we can maximize the probability that we can generate our original documents, using these topics. Before we apply LDA, we need to ensure that our dataset is processed using natural language processing (NLP) for which we will be using the techniques such as lemmatization, stemming of words, removal of stop words etc.

2 Related Work

We have a couple of frequently used existing methods for Topic Modeling, these methods are Latent Semantic Analysis, Latent Dirichlet Allocation, non-negative matrix factorization, random projection, and principal component analysis. When two textual datasets were selected to evaluate the performance of included topic modeling methods based on the topic quality and some standard statistical evaluation metrics, like recall, precision, F-score, and topic coherence. As a result, Latent Dirichlet Allocation and non-negative matrix factorization methods delivered more meaningful extracted topics and obtained good results.

2.1 NLP Topic Modeling Methods and Tools

Topic modeling is an important area of data (text) mining. A topic model is a probabilistic model that finds out the main topics in a collection of data called as corpus. The basic concept is to consider the data or documents as a collection of topics in the topic model, and each topic is considered as a probability distribution of the words. Each topic in itself is viewed as a mixture of words, and each document can be viewed as a collection of topics with different probabilities depending on the frequency that those terms appear.

Topic Modelling is distinctive of rule-based text mining strategies that make use of regular expressions or dictionary-based keyword searching routines. It is an unsupervised approach used for detecting the mass of words (called "topics") in large clusters of texts.

Topic modeling for information retrieval these days has gained importance and has proved good performance in several tasks. It facilitates understanding, organizing, and summarizing huge text

datasets. Various topic modeling approaches, including Latent Semantic Analysis (LSA), Probabilistic Latent Semantic Analysis (PLSA) and Latent Dirichlet Allocation (LDA) can be used. Several tools are available for performing NLP. Most used are - Natural Language Toolkit (NLTK), Gensim and Mallet.

2.2 LDA Topic Modeling and Qualitative Analysis on Large Datasets

In 2021, Gaurav Nanda, Kerrie A. Douglas, David R. Waller, Hillary E. Merzdorf, and Dan Goldwasser, presented their findings after analyzing large collections of Open-Ended Feedback from MOOCs (Massive Online Courses) Learners using LDA Topic Modeling.

In this study, they identified the most significant aspects of the MOOC learning experience from learners' perspective by mining their post course open-ended survey responses using LDA topic models and qualitative analysis. The significant pedagogical aspects were instructor, content, social interaction, course-load, assessments, and feedback, and the significant technical aspects included detailed information about the course, language-related support, usability of the platform, accessibility of learning material, support related to low Internet speed in developing countries, and pricing of the certificate.

The methodology used in this study is replicable for exploratory analysis on large textual data and can be used in research for similar tasks such as analyzing open-ended feedback or discussion forums. It is also important to note that while the LDA topic model provides a good starting point for exploratory data analysis, its results need to be interpreted using qualitative analysis to obtain meaningful conclusions.

3 Method

LDA, is a probabilistic model that is the most popular Topic Modeling algorithm in real-life applications to extract topics from document collections since it provides accurate results and can be trained online. Corpus is organized as a random mixture of latent topics in the LDA model, and the topic refers to a word distribution. Also, LDA is a generative unsupervised statistical algorithm for extracting thematic information (topics) of a collection of documents within the Bayesian statistical paradigm. The LDA model assumes that each document is made up of various topics, where each topic is a probability distribution over words. A significant advantage of using the LDA model is that topics can be inferred from a given collection without input from any prior knowledge.

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 every chunk of text we feed into it will contain words that are somehow related. Hence, 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.

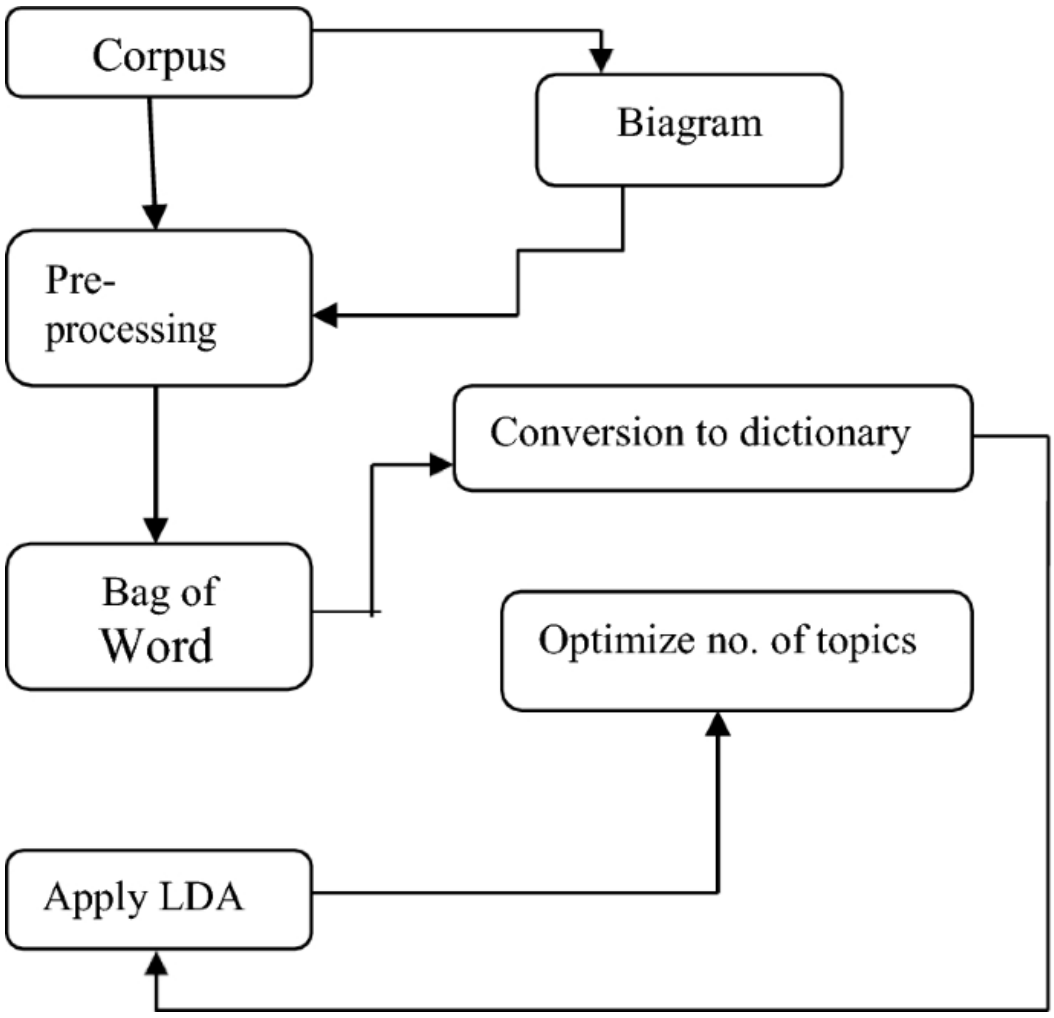

Figure 2. Topic Modeling Flow

3.1 Dataset

For my project I took the dataset from Kaggle. It includes News stories of 4 categories. Size of training set: 7,628 records Size of test set: 2,748 records.

Features:

- STORY: A part of the main content of the article to be published as a piece of news.
- SECTION: The genre/category the STORY falls in.

There are four distinct sections where each story may fall in to. The Sections are labelled as follows: Politics: 0 Technology: 1 Entertainment: 2 Business: 3.

3.2 Data Preprocessing

Before using LDA model on my dataset I performed the following operations on data to pre-process it. Steps followed:

1. Tokenization: Split the text into sentences and the sentences into words. Lowercase the words and remove punctuation.
2. Stop Words Removal: Words that have fewer than 3 characters are removed. All stop words are removed.
3. Lemmatization: Words are lemmatized - words in third person are changed to first person and verbs in past and future tenses are changed into present.
4. Stemming: Words are stemmed - words are reduced to their root form.

```
'''
Function to perform the pre processing steps on the entire dataset.
'''
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
```

Figure 3. Data Pre-Processing

```
'''
Previewing a random story from train data after preprocessing.
'''
story_num = random.randint(0,train_data_df.shape[0])
story_sample = train_data_df['STORY'][story_num]
print("Original Story: ")
words = []
for word in story_sample.split(' '):
    words.append(word)
print(words)
print("\n\nTokenized and lemmatized Story: ")
print(preprocess(story_sample))
```

```
Original Story:
['Facebook', 'claims', 'that', 'over', '700', 'million', 'people', 'play', 'games,', 'watch', 'gaming', 'videos', 'or', 'engage', 'in', 'gaming', 'groups', '

Tokenized and lemmatized Story:
['facebook', 'claim', 'million', 'peopl', 'play', 'game', 'watch', 'game', 'video', 'engag', 'game', 'group', 'platform', 'game', 'help', 'connect', 'techcru
```

Figure 4. Preview of test sample after pre-processing of data

3.3 Bag of Words

This step creates a dictionary from 'processed_stories' containing the number of times a word appears in the training set. I have used genism.coropa.Dictionary() for this. Followed by some more filtering out of data. Steps Involved:

1. Create dictionary from words present in the entire training data.
2. Remove very rare and very common words from the dictionary under consideration.
3. Convert document (a list of words) into the bag-of-words format = list of (token_id, token_count) 2-tuples. Each word is assumed to be a tokenized and normalized string.

```
'''
Preview BOW for our sample preprocessed stories
'''
random_story_num = random.randint(0,train_data_df.shape[0])
bow_doc_x = bow_corpus[random_story_num]

for i in range(len(bow_doc_x)):
    print("Word {} (\"{}\") appears {} time.".format(bow_doc_x[i][0],
                                                     dictionary[bow_doc_x[i][0]],
                                                     bow_doc_x[i][1]))
```

```
Word 217 ("date") appears 1 time.
Word 237 ("launch") appears 3 time.
Word 326 ("wire") appears 1 time.
Word 435 ("iphon") appears 3 time.
Word 563 ("mint") appears 1 time.
Word 1165 ("surgic") appears 1 time.
Word 1318 ("grade") appears 1 time.
Word 1371 ("frame") appears 1 time.
Word 1698 ("aluminium") appears 1 time.
Word 1707 ("steel") appears 1 time.
Word 1754 ("cloth") appears 1 time.
Word 2340 ("xsiphon") appears 1 time.
```

Figure 5. Preview of Bag-Of-Words Corpus

3.4 Topic Modeling: Running LDA using Bag-Of-Words

I trained my LDA model using gensim.models.LdaMulticore. Some of the parameters which I tried to tweak were:

1. num_topics: is the number of requested latent topics to be extracted from the training corpus.
2. id2word: is a mapping from word ids (integers) to words (strings). It is used to determine the vocabulary size, as well as for debugging and topic printing.
3. workers: is the number of extra processes to use for parallelization. Uses all available cores by default.
4. alpha and eta: Hyperparameters affecting the sparsity of stories.
5. passes: No of training passes through the corpus.

```
[18] '''
     Training the lda model using gensim.models.LdaMulticore and saving it to 'lda_model'
     For my use case I will be chosing num_topics = 4, initially.
     '''
     lda_model =  gensim.models.LdaMulticore(bow_corpus,
                                             num_topics = 4,
                                             id2word = dictionary,
                                             passes = 10,
                                             workers = 2)
```

Figure 6. Training the LDA Model.

4 Experiments and Results

4.1 Computing Coherence Scores

Topic coherence is one of the main techniques used to estimate the number of topics. I used both UMass and c_v measure to see the coherence score of my LDA model. It measures how interpretable the topics are to humans. In this case, topics are represented as the top N words with the highest probability of belonging to that topic. Briefly, the coherence score measures how similar these words are to each other.

```
[64] from gensim.models import CoherenceModel
     # Compute Coherence Score using c_v
     coherence_model_lda = CoherenceModel(model=lda_model, texts=processed_stories, dictionary=dictionary, coherence='c_v')
     coherence_lda = coherence_model_lda.get_coherence()
     print('\nCoherence Score: ', coherence_lda)

     Coherence Score:  0.6153343082681273

     # Compute Coherence Score using u_mass
     coherence_model_lda = CoherenceModel(model=lda_model, texts=processed_stories, dictionary=dictionary, coherence='u_mass')
     coherence_lda = coherence_model_lda.get_coherence()
     print('\nCoherence Score: ', coherence_lda)

     Coherence Score:  -2.3752227789934777
```

Figure 7. Computing coherence Scores (when num_topics = 4)

4.2 Experiments to get Optimal number of Topics

In this step I worked on finding the optimal number of topics. We need to build many LDA models with different values of the number of topics (k) and pick the one that gives the highest coherence value. Choosing a 'k' that marks the end of a rapid growth of topic coherence usually offers meaningful and interpretable topics. Picking an even higher value can sometimes provide more granular sub-topics. If you see the same keywords being repeated in multiple topics, it's probably a sign that the 'k' is too large.

```
from gensim.models.ldamodel import LdaModel
def compute_coherence_values(dictionary, corpus, texts, limit, start=2, step=3):
    """
    Compute c_v coherence for various number of topics

    Parameters:
    ----------
    dictionary : Gensim dictionary
    corpus : Gensim corpus
    texts : List of input texts
    limit : Max num of topics

    Returns:
    -------
    model_list : List of LDA topic models
    coherence_values : Coherence values corresponding to the LDA model with respective number of topics
    """
    coherence_values = []
    model_list = []
    for num_topics in range(start, limit, step):
        model=LdaModel(corpus=corpus, id2word=dictionary, num_topics=num_topics)
        model_list.append(model)
        coherencemodel = CoherenceModel(model=model, texts=texts, dictionary=dictionary, coherence='c_v')
        coherence_values.append(coherencemodel.get_coherence())

    return model_list, coherence_values
```

```
[76] model_list, coherence_values = compute_coherence_values(dictionary=dictionary, corpus=bow_corpus, texts=processed_stories, start=2, limit=40, step=6)
     # Show graph
     import matplotlib.pyplot as plt
     limit=40; start=2; step=6;
     x = range(start, limit, step)
     plt.plot(x, coherence_values)
     plt.xlabel("Num Topics")
     plt.ylabel("Coherence score")
     plt.legend(("coherence_values"), loc='best')
     plt.show()
```

Figure 8. Code to compute and plot the graph for num_topics vs coherence scores.

After experimenting, I was able to come with an optimal number for count of topics, which was **7**. Following graphs show that when **k=7** we had the maximum coherence scores.

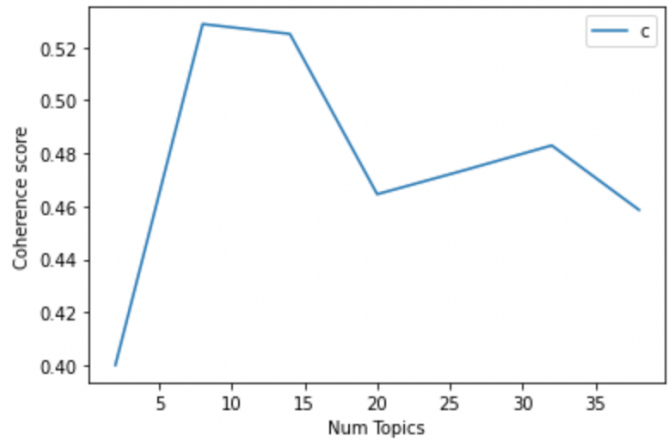

Figure 9. When coherence measure was ‘c_v’

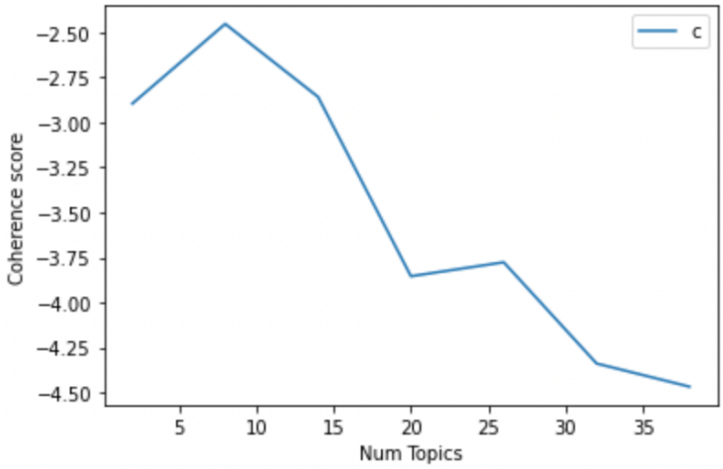

Figure 10. When coherence measure was ‘u_mass’

4.3 Classification of Topics

After using the value of num_topics as and retraining the LDA model we got the following topics with the scores of words associated with each of the topics.

```
'''
For each topic, we can see the words occuring in that topic and their relative weights.
'''
for idx, topic in lda_model.print_topics(-1):
    print("Topic: {} \nWords: {}".format(idx, topic ))
    print("\n")
```

```
Topic: 0
Words: 0.018*"seat" + 0.016*"modi" + 0.015*"minist" + 0.011*"polit" + 0.011*"leader" + 0.011*"sabha" + 0.010*"poll" + 0.009*"vote" + 0.008*"govern" + 0.008*"


Topic: 1
Words: 0.012*"trade" + 0.011*"note" + 0.010*"bank" + 0.009*"dollar" + 0.009*"quarter" + 0.009*"growth" + 0.008*"investor" + 0.008*"stock" + 0.008*"crore" + 0


Topic: 2
Words: 0.021*"film" + 0.010*"actor" + 0.009*"facebook" + 0.007*"star" + 0.006*"stori" + 0.006*"know" + 0.006*"movi" + 0.006*"media" + 0.006*"charact" + 0.005


Topic: 3
Words: 0.017*"poll" + 0.016*"phase" + 0.014*"aveng" + 0.012*"vote" + 0.011*"endgam" + 0.009*"court" + 0.008*"marvel" + 0.008*"voter" + 0.007*"turnout" + 0.00


Topic: 4
Words: 0.013*"govern" + 0.008*"data" + 0.007*"secur" + 0.007*"countri" + 0.006*"indian" + 0.006*"million" + 0.006*"servic" + 0.005*"scheme" + 0.005*"bank" +


Topic: 5
Words: 0.012*"googl" + 0.009*"featur" + 0.009*"data" + 0.008*"game" + 0.008*"devic" + 0.007*"technolog" + 0.006*"facebook" + 0.006*"servic" + 0.005*"whatsapp


Topic: 6
Words: 0.030*"smartphon" + 0.021*"appl" + 0.018*"samsung" + 0.017*"phone" + 0.014*"launch" + 0.014*"iphon" + 0.013*"galaxi" + 0.013*"display" + 0.010*"oneplu
```

Figure 11. Topics Classification and words probability scores distribution

4.4 Testing LDA Model on Test Dataset

Here is an example of a random News story from the test data which I had to see how my model assigns the topics to it. In [Figure 13.] we can see that my model assigned a score of **0.62 to Topic 6** and a score of **0.36 to Topic 5**.

```
[85] story_num = random.randint(0,test_data_df.shape[1])
     test_story = test_data_df['STORY'][story_num]
     test_story
```

```
'2019 will see gadgets like gaming smartphones and wearable medical devices lifting the user experience to a whole new level\n\n\nmint-india-wire consumer t
echnologyconsumer technology trends in New Yeartech gadgetsFoldable phonesgaming smartphoneswearable medical devicestechnology\n\n\nNew Delhi: Gadgets have
become an integral part of our lives with most of us relying on some form of factor to communicate, commute, work, be informed or entertained. Year 2019 wil
l see some gadgets lifting the user experience to a whole new level. Here's what we can expect to see:\n\n\nSmartphones with foldable screens: Foldable phon
es are finally moving from the concept stage to commercial launches. They are made up of organic light-emitting diode (OLED) panels with higher plastic subs
trates, allowing them to be bent without damage.\n\n\nUS-based display maker Royole Corp's foldable phone, FlexPai, has already arrived in select markets, w
hile Samsung's unnamed foldable phone is expected sometime next y…'
```

Figure 12. Random Sample Test Data

```
[86] # Data preprocessing step for the test story and score computation

     bow_vector = dictionary.doc2bow(preprocess(test_story))

     for index, score in sorted(lda_model[bow_vector], key=lambda tup: -1*tup[1]):
         print("Score: {}\t Topic: {}".format(score, lda_model.print_topic(index, 5)))
```

```
Score: 0.6250270009040833       Topic: 0.030*"smartphon" + 0.021*"appl" + 0.018*"samsung" + 0.017*"phone" + 0.014*"launch"
Score: 0.36808550357818604      Topic: 0.012*"googl" + 0.009*"featur" + 0.009*"data" + 0.008*"game" + 0.008*"devic"
```

Figure 13. Scores assigned to Topic 6 and Topic 5

4.5 Analysis of Topics

4.5.1 Word Count and Importance of Topic Keywords

When it comes to the keywords in the topics, the importance (weights) of the keywords matters. Along with that, how frequently the words have appeared is an important parameter too. Following plot of the word counts and the weights of each keyword in the same chart is an important visualization to understand the Topics.

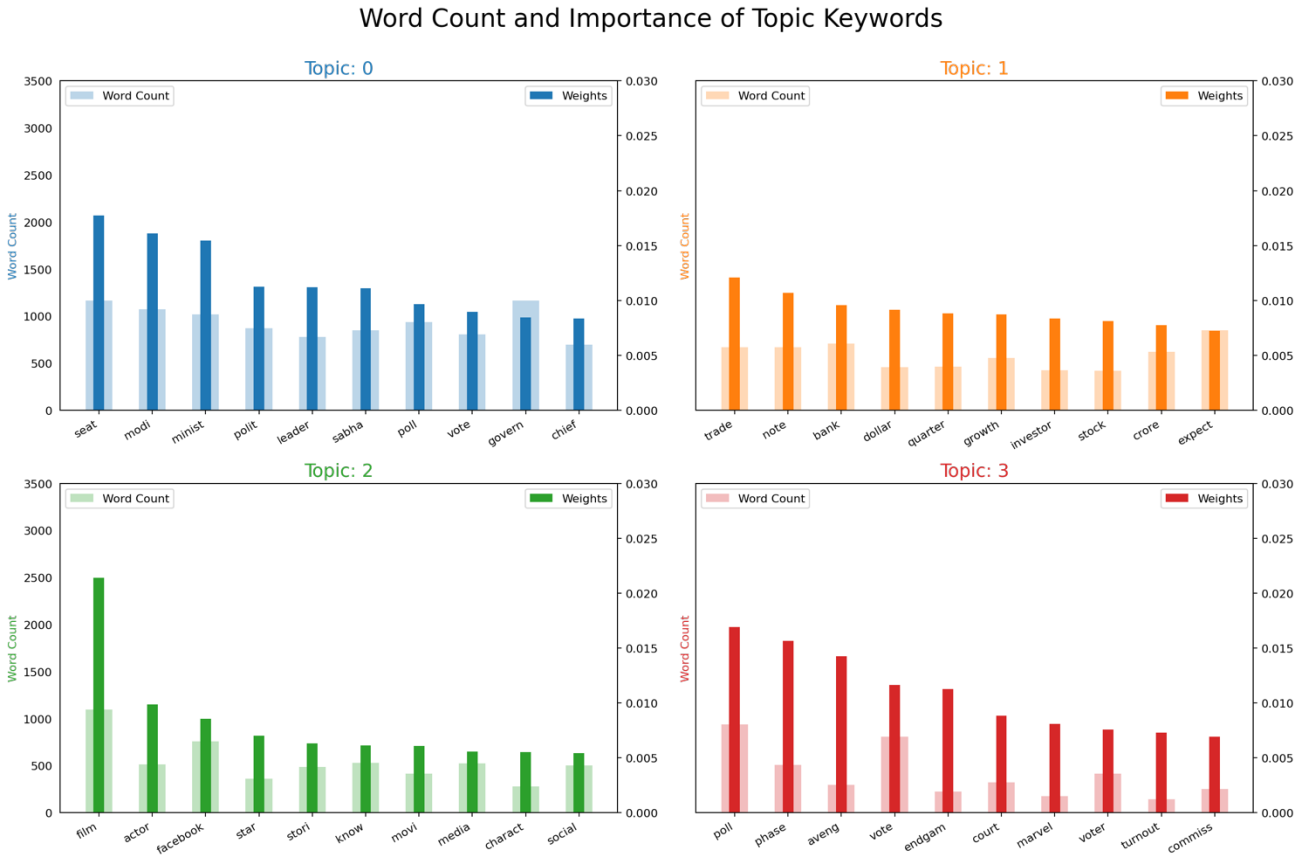

Figure 14. Plot of Word Counts and Importance of Keywords for 4 of the Topics

4.5.2 Visualization of Topics

I have used, pyLDAVis to visualize the information contained in a topic model. Below is the dashboard built on the LDA model trained. It is a dynamic dashboard and user can select the topic number to see the information accordingly. The dashboard covers a distance map showing how widely each topic is placed, second graph show the Top-30 mot relevant terms for the selected topic. We can also use the slider to adjust the hyperparameter lambda.

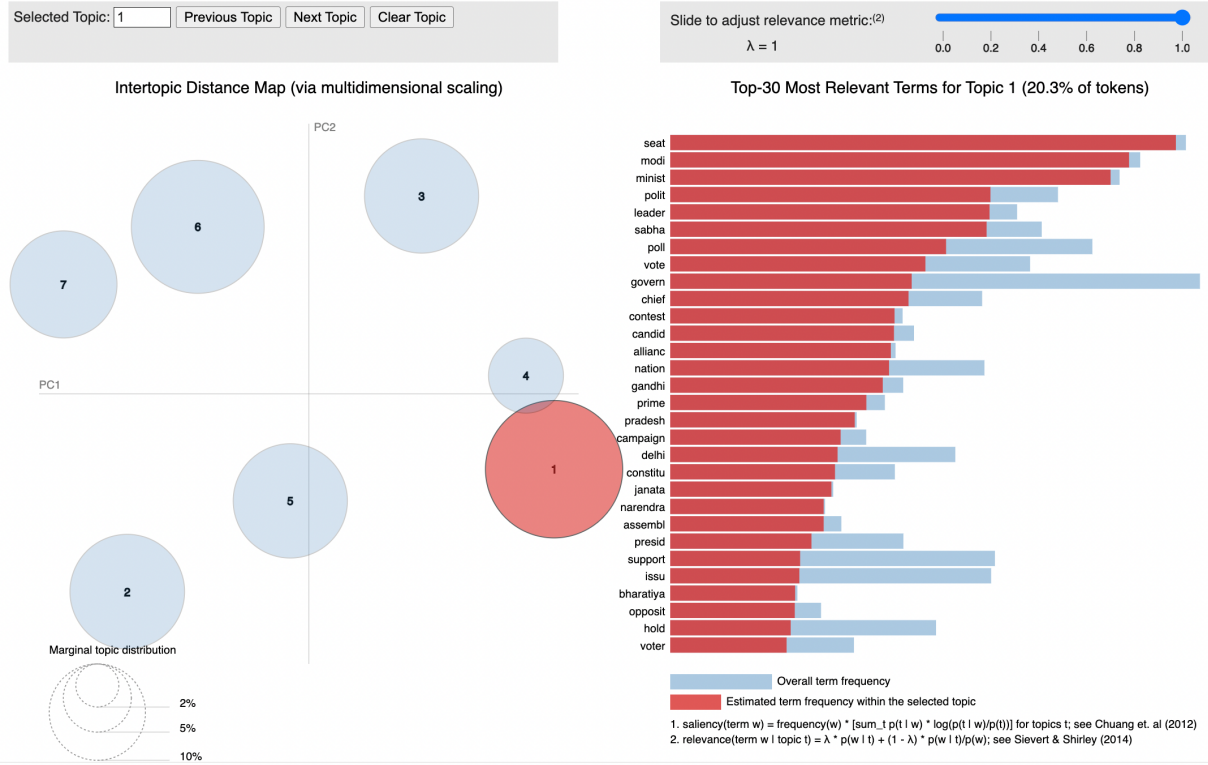

Figure 15. Visualization of Topic 1

5 Conclusion

In this project, I proposed the steps to perform Topic Modeling on a given dataset. I used Latent Dirichlet Allocation model to perform topic modeling on a News Dataset. Before inputting the data to the model, I performed various pre-processing steps on the data, followed by generating a bag-of-words data to be passed to LDA model. For evaluating the performance of the LDA model I used the measure of Coherence scores. After finding an optimal value for number of topics into which the given dataset can be classified into, I performed an analysis on the Topics using some graphs and visualizations.

6 References

[1] "A Comparative Study of NLP Topic Modeling Methods and Tools" – By Dr. Shivkumar Goel, Miss. Priyanka Devasthal. (https://citationsy.com/archives/q?doi=10.22214/ijraset.2019.6334)

[2] "Topic Modeling in Natural Language Processing" – By Subhashini Gupta, Prof. Grishma Sharma. (https://www.analyticsvidhya.com/blog/2021/05/topic-modelling-in-natural-language-processing/#:~:text=Topic%20modelling%20is%20done%20using,process%20(therefore%20called%20latent))

[3] "Analyzing Large Collections of Open-Ended Feedback From MOOC Learners Using LDA Topic Modeling and Qualitative Analysis" – By Gaurav Nanda , Kerrie A. Douglas, David R. Waller, Hillary E. Merzdorf, and Dan Goldwasser. (https://ieeexplore.ieee.org/document/9373927)
[4] https://www.machinelearningplus.com/nlp/topic-modeling-visualization-how-to-present-results-lda-models/
[5] Dataset: https://www.kaggle.com/datasets/akash14/news-category-dataset?select=Data_Train.csv
[6] Github Repo: https://github.com/pmehta16/Topic-Modeling-using-Natural-Language-Processing