**TOPIC MODELING**

**Introduction**

Topic Modelling is an unsupervised machine learning techniques where in it can scan large set of documents, detecting words and phrases inside it and automatically clusters the words groups without a predefined set of labels. If you feed the model data, it will give you different sets of words, and each set of words describes the topic.

e.g., Identifying the topics of a set of customer reviews by detecting patterns and recurring words.

Consider following sentence: “The nice thing about Eventbrite is that it's **free to use** as long as you're not **charging** for the event. There is a **fee** if you are **charging** for the event – 2.5**% plus a $0.99 transaction fee**.” By identifying words and expressions such as free to use, fee, charging, 2.5% plus 99 cents transaction fee, topic modeling can group this review with other reviews that talk about similar things (these may or may not be about pricing).

**Working of Topic Modelling**

* Topic Modeling will involve counting words and grouping similar patterns of the word to infer topics inside the unstructured data.
* Detecting word frequency and distance between words, topic model will cluster the feedback that is similar and words and expressions that appear most often.

It is a process of dividing the corpus of documents into

1. A list of the topics covered by the documents in the corpus.
2. Several sets of documents from the corpus grouped by the topics they cover.

Assumption is that every document consists of a statistical distribution of topics obtained by summing all the distributions for all the topics covered. It figures out   
which topics are present in the document of the corpus and how strong that presence is.

**BERTopic:** It is a topic modeling technique that leverages transformers and c-TF-IDF to create dense clusters allowing for easily interpretable topics while keeping the important words in the topic descriptions.

It is guided, semi-supervised, hierarchical, and dynamic topic modeling.

BERTopic generates the topic representations through three steps.

1. Each document is converted to its embedding representation using a pre-trained language model.
2. Before clustering these embeddings, the dimensionality of the resulting embeddings is reduced to optimize the clustering process.
3. From the cluster of documents, topic representations are extracted using a custom class-based variation of TF-IDF.

i) Document Embeddings: In BERTopic documents are embedded to create the representations in vector space that can be compared semantically. Here Sentence BERT (SBERT) framework is used. This framework allows users to convert sentences and paragraphs to dense vector representations using pre-trained language models. These embeddings are used to cluster semantically similar documents and not directly used in generating the topics.

ii) Document clustering: UMAP (Uniform Manifold Approximation and Projection) is used to reduce the dimensionality of document embeddings. The reduced embeddings are clustering used HDBSCAN. It is an extension of DBSCAN which finds clusters of varying densities by converting DBSCAN into a hierarchical clustering algorithm.

HDBSCAN models clusters using a soft-clustering approach allowing noise to be modeled as outliers. It prevents unrelated documents to be assigned to any cluster and improves topic representations.

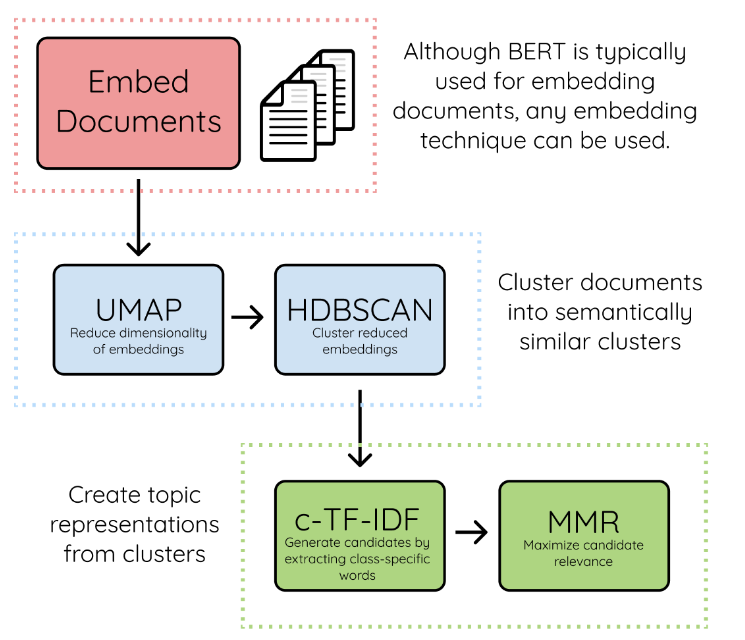
iii)Topic Representation: These are modeled based on the documents in each cluster where each cluster will be assigned one topic. For each topic, we would like to know what makes one topic based on its cluster-word distribution. We can modify TF-IDF (Term Frequency – Inverse Document Frequency) which is a measure to represent the importance of a word to a document such that it allows for a representation of a term’s importance to a topic instead.

Where the term frequency models the frequency of term t in document d. The inverse document frequency measures how much information a term provides to a document and is calculated by taking the logarithm of the number of documents in a corpus N divided by the total number of documents that contain t.

The procedure is then generalized to clusters of documents. All the documents in a cluster are treated as a single document by concatenating the documents.

Where the term frequency models the frequency of term t in a class c or in this instance. The class c is the collection of documents concatenated into a single document for each cluster. Then, the inverse document frequency is replaced by the inverse class frequency to measure how much information a term provides to a class. It is calculated by taking the logarithm of the average number of words per class A divided by the frequency of term t across all classes. This class-based TF-IDF procedure models the importance of words in clusters instead of individual documents. This allows us to generate topic-word distributions for each cluster of documents

**Algorithm**



The algorithm contains 3 stages:

1. **Embed documents**: Extract document embeddings with BERT or any other embedding technique. The document embeddings are created from a set of documents using sentence-transformers. These models are pre-trained for many languages and are great for creating either document- or sentence-embeddings. In BERTopic, we can choose any sentence-transformers model but there are two models that are set as defaults:

* all-MiniLM-L6-v2
* Paraphrase-multilingual-MiniLM-L12-v2

The first is an English language model trained specifically for semantic similarity tasks. The second model is very similar to the first with one major difference is that the multilingual models work for 50+ languages. This model is a bit larger than the first and is only selected if you select any language other than English.

1. **Cluster Documents**: UMAP is used to reduce the dimensionality of embeddings. HDBSCAN is used to cluster reduced embeddings and create clusters of semantically similar documents. Typically, clustering algorithms have difficulty clustering data in high dimensional space. First, we need to reduce the dimensionality of the embeddings that we generated. As a default, we use UMAP since it preserves both the local and global structure of embeddings. Then, we use a density-based clustering technique, HDBSCAN, to create our clusters and identify outliers where possible.
2. **Create topic representation**: Extract and reduce topics with c-TF-IDF. Improve coherence and diversity of words with Maximal Marginal Relevance.

* When you apply TF-IDF as usual on a set of documents, we are comparing the importance of words between documents. The more important words are within a cluster, the more it is representative of that topic. If we extract the most important words per cluster, we get descriptions of **topics**. This model is called class-based TF-IDF.
* Each cluster is converted to a single document instead of a set of documents. Then, we extract the frequency of word x in class c, where c refers to the cluster we created before. This results in our class-based tf representation.
* Then, we take the logarithm of one plus the average number of words per class A divided by the frequency of word x across all classes. We add plus one within the logarithm to force values to be positive. This results in our class-based idf representation.
* We then multiply tf with idf to get the importance score per word in each class.

## **Maximal Marginal Relevance Coherence: After generating the c-TF-IDF representations, we have a set of words which describe a collection of documents. Here many of the words describe a similar topic but some words will overfit the documents. To improve the coherence of words, Maximal Marginal Relevance was used to find the most coherent words which results in removing the words which do not contribute to a topic. This technique is used to diversify the words in the topic representation.**

**Dataset Description**

The transcripts data of different companies like Cisco, Wells Fargo, Intuit, UHG, ThermoFisher etc. was collected from docoh.com website for one quarter Q3 of 2022. It contains the names of the participants along with their designations. Also, it has call transcript which is the conversation between the participants and the operator.

**Methodology**

1. We have considered transcripts data of 5 companies (Wells Fargo, Cisco, Intuit, UHG and ThermoFisher) from docoh.com and names of the officials (CEO, CFO, VP etc.) from different companies.
2. Initially using BERTopic package we extracted the topics and visualized it for different datasets separately.
3. Transformers like Roberta base, FinBERT, the **all-**\* models etc. were used for optimization. Also, Countvectorizer was considered to remove stop words in the transcript.
4. LDA (Latent Dirichlet Allocation) algorithm was used to check for better results.
5. At last, the transcript data from all the 5 files were combined in a single text file (Five\_companies\_transcript.txt) and step 1 to 4 were repeated.

**Used Transformers**

1. Roberta-base: It is a pre-trained model on English language using a masked language modeling (MLM). RoBERTa is a transformer model pretrained on a large corpus of English data in a self-supervised manner. The model was pretrained on the raw texts only with an automatic process to generate the inputs and labels from those texts. It was pretrained with the Masked Language Modelling (MLM). The model randomly masks 15% of the words in the input, then runs the entire masked sentence through the model and predicts the masked words. The model learns an inner representation of the English language that can be used to extract useful for downstream tasks.
2. FinBERT: It is a pre-trained NLP model to analyze sentiment of financial text. It is built by training the BERT language model in the finance domain and fine-tuning it for financial sentiment classification.
3. all-mpnet-base-v2: All-round model tuned for many use-cases. It is trained on a large and diverse dataset of over 1 billion training pairs. This is a sentence-transformers model. It maps sentences & paragraphs to a 768-dimensional dense vector space and can be used for tasks like clustering or semantic search.
4. all-distilroberta-v1: All-round model tuned for many use-cases. Trained on a large and diverse dataset of over 1 billion training pairs. This is a sentence-transformers model. It maps sentences & paragraphs to a 768-dimensional dense vector space and can be used for tasks like clustering or semantic search.
5. all-MiniLM-L12-v2: All-round model tuned for many use-cases. Trained on a large and diverse dataset of over 1 billion training pairs. This is a sentence-transformers model. It maps sentences & paragraphs to a 384-dimensional dense vector space and can be used for tasks like clustering or semantic search.
6. all-MiniLM-L6-v2: All-round model tuned for many use-cases. Trained on a large and diverse dataset of over 1 billion training pairs. This is a sentence-transformers model. It maps sentences & paragraphs to a 384-dimensional dense vector space and can be used for tasks like clustering or semantic search.

**LDA Algorithm**

Latent Dirichlet Allocation (LDA) algorithm is an example of topic model and is used to classify text in a document to a particular topic. It helps to build a topic per document model and words per topic model which called as Dirichlet distributions.

The aim of LDA is to find topics a document belongs to, based on the words in it.

There are 2 parts in LDA:

* The words that belong to a document, which we already we know.
* The aim of LDA is to find topics a document belongs to, based on the words in it.

The two methods of LDA are using 1) Gensim, 2) sklearn

1) **Various steps in LDA using gensim package**:

i) After importing the required libraries, we will compile all the documents into one list to have the corpus.

ii) Text preprocessing

* Convert the text into lowercases.
* Split the text into words.
* Remove the stop loss words.
* Remove the punctuation, symbols, and special characters.
* Normalize the word.

iii) For gensim: Using gensim for Document Term Matrix (DTM), we don’t need to explicitly create the DTM matrix from scratch. The gensim library has an internal mechanism to create the DTM. The only requirement for the gensim package is that we need to pass the cleaned data in the form of tokenized words.

Creating the Document Term Matrix:

* We use gensim package for Document Term Matrix (DTM). So, we don’t have to create DTM manually. The data which is cleaned must be passed in the form of tokenized words.
* Convert the corpus (list of documents) into a document-term Matrix. Bag of Words (BOW) vectorizer is being used.

iv) We pass the vectorized corpus to the LDA model.

Implementing the LDA Algorithm:

* Creating the object for LDA model using gensim library.
* Running and training the LDA model on the document term matrix.
* Extract the topics from the Corpus.
* Assign the topics to the documents.

### **Parameters for LDA model in gensim**

Following are the important parameters for LDA for implementing in the gensim package:

1. The corpus or the document-term matrix to be passed to the model. It is called as doc\_term\_matrix.
2. **Number of Topics:**num\_topics is the number of topics we want to extract from the corpus.
3. **id2word**: It is the mapping from word indices to words. Each of the words has an index that is present in the dictionary.
4. **Number of Iterations:**Tt is represented by **Passes**in Python**.**Referred to as ‘epochs’. Passes control how often we want to train the model on the entire corpus for convergence.
5. **Chunksize**: It is the number of documents to be used in each training chunk. The chunksize controls how many documents can be processed at one time in the training algorithm.
6. **LDA’s model parameters:**

* **Alpha:** is the document-topic density
* **Beta:** is the topic word density
* For, the higher values of alpha —> the documents will be composed of more topics.
* The lower values of alpha —> returns documents with fewer topics.

Similarly, for the values of Beta:

* The higher beta —> has a greater number of words in each topic,
* The lower value of beta —> topics contains few words.

It is better to keep alpha and beta parameters as ‘auto’ because the model is automatically learning these two parameters.

2) **Various steps in LDA using sklearn package**:

i) After importing the required libraries, we will compile all the documents into one list to have the corpus.

ii) Text preprocessing as in previous method

iii) Converting the text into numerical representation. The clean preprocessed corpus is converted into array. Here count vectorizer is used. For sklearn: Use either the Count vectorizer or TF-IDF vectorizer to transform the Document Term Matrix (DTM) into numerical arrays.

iv) We pass the vectorized corpus to the LDA model.

Implementing the LDA Algorithm:

* Pass the corpus (document-term matrix) to the model.
* Retrieve the topics.
* Annotate the topics of the documents.

### **Parameters for LDA model in sklearn**

The arguments used in the sklearn package are:

1. The corpus or the document-term matrix to be passed to the model.
2. **Number of Topics:**n\_components is the number of topics to find from the corpus.
3. **The number of maximum iterations:** max\_iter: It is the number of maximum iterations allowed for the LDA algorithm to converge.

**Sub-Models:**

**Dimensionality Reduction Techniques**

* In BERTopic we use UMAP as it can capture both the local and global high-dimensional space in lower dimensions. UMAP model parameter allows for a variety of dimensionality reduction models. fit(x) is a function that can be used to fit the model. transform(x) function transforms the input to a lower dimensional size.

1. **UMAP (Uniform Manifold Approximation and Mapping):** It is dimension reduction technique that can be used for visualization. It is also for general non-linear dimension reduction. The algorithm is founded on three assumptions about the data

* The data is uniformly distributed on Riemannian manifold.
* The Riemannian metric is locally constant.
* The manifold is locally connected.

1. **PCA (Principal Component Analysis):** Sometimes UMAP may take a while to fit on embeddings. When we have the embeddings of millions of documents which we want to reduce the dimensionality, there is a trick to speed up the process. We can initialize the UMAP with rescaled PCA embeddings.  It can be faster to train and to perform inference with. To use PCA, we can simply import it from sklearn and pass it to the umap\_model parameter

**Clustering Algorithms:**

After reducing the dimensionality of the input embeddings, we need to cluster them into groups of similar embeddings to extract our topics. The more we perform clustering, the more accurate will be the topic representations. HDBSCAN is used in BERTopic since it is capturing the structures with different densities. The class will have the following attributes:

1. . fit(X): A function that can be used to fit the model.
2. . predict(X): A predict function that transforms the input to cluster labels.
3. . labels\_: The labels after fitting the model.
4. **HDBSCAN**: It extends DBSCAN by converting it into a hierarchical clustering algorithm, and then using a technique to extract a flat clustering based in the stability of clusters.

**Steps:**

1. Transform the space according to the density/sparsity.
2. Build the minimum spanning tree of the distance weighted graph.
3. Construct a cluster hierarchy of connected components.
4. Condense the cluster hierarchy based on minimum cluster size.
5. Extract the stable clusters from the condensed tree.
6. **K-Means**: It allows us to select how many clusters you will like and forces every single point to be present in a cluster. Therefore, no outliers will be created.

Disadvantage is that it we force every single point in a cluster, it will mean that the cluster is highly likely to contain noise which disturbs the topic representations. When using vectorizer\_model=CountVectorizer(stop\_words="english"), it will improve the topic representation.

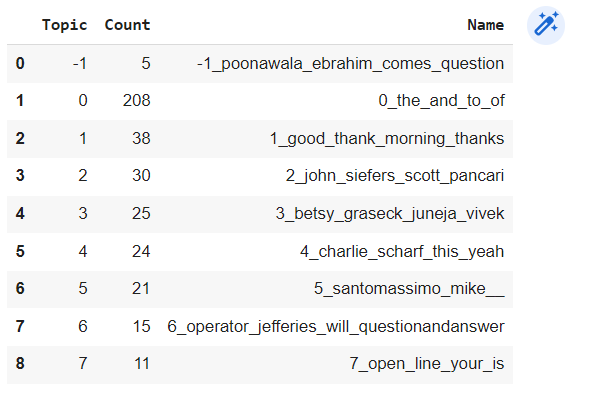
**Count Vectorizer**: CountVectorizer and c-TF-IDF together are responsible for creating the topic representations and flexible in parameter tuning. We can pass CountVectorizer before and after training the topic model. Passing it before training allows us to minimize the size of the resulting c-TF-IDF matrix. Passing it after training allows us to fine-tune the topic representations by using update\_topics().

**Contextualized Topic Modelling (CTM)**:  **CTMs** are a family of topic models that combine the expressive power of BERT embeddings with the unsupervised capabilities of topic models to get topics out of documents.

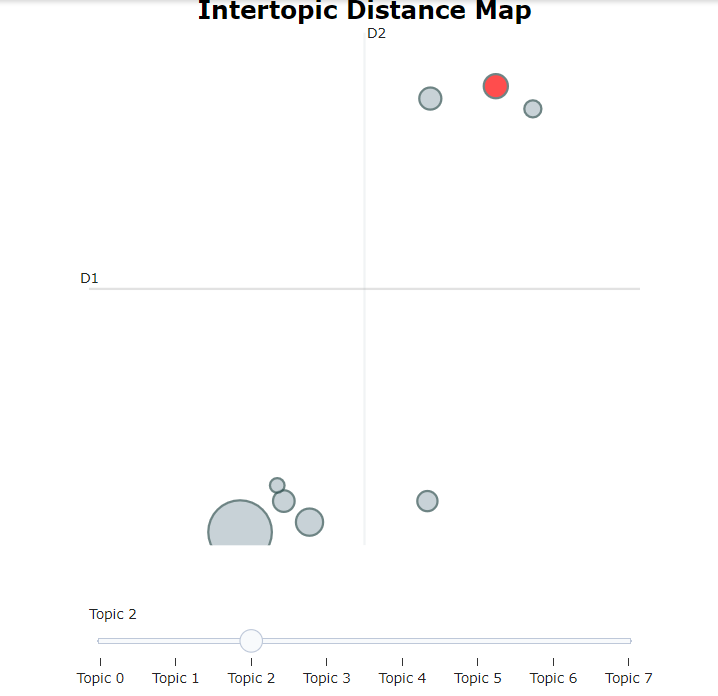
In CTMs we have two models. CombinedTM and ZeroShotTM. CTMs work better when the size of the bag of words **has been restricted to number of terms**. This is because we have a neural model that reconstructs the input bag of word. In CombinedTM we project the contextualized embedding to the vocab space. If you give a contextual model like BERT preprocessed text, it might be difficult to get out a good representation. We use the preprocessed text for the bag of word creating and use the NOT preprocessed text for BERT embeddings. CombinedTM combines contextual embeddings with the good old bag of words to make more coherent topics. ZeroShotTM is the perfect topic model for task in which you might have missing words in the test data and if trained with multilingual embeddings, it inherits the property of being a multilingual topic model. The advantage is we can use different embeddings for CTMs. So, when a new embedding method comes out you can use it in the code and improve the results.

**BERTopic Topic Extraction Results**

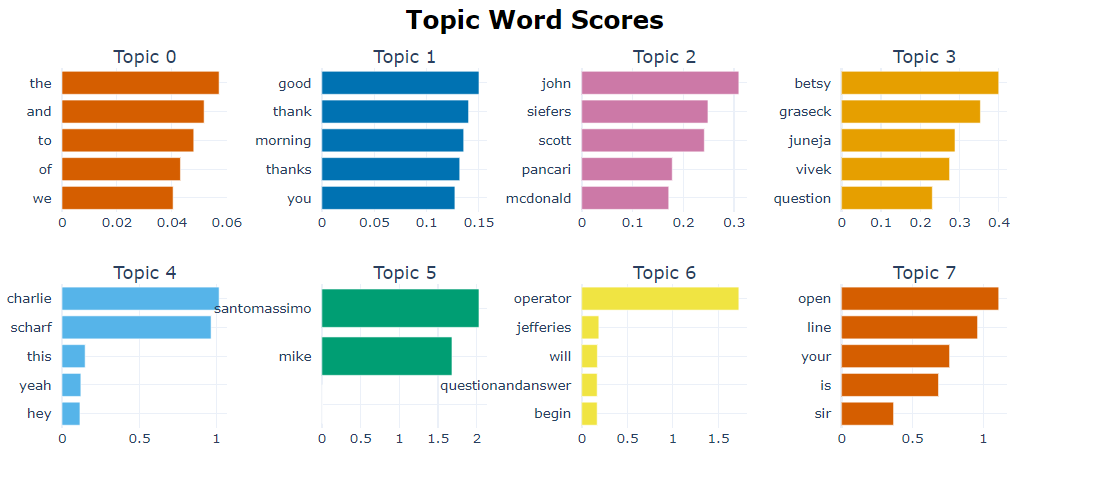
1. **Wells Fargo dataset**
2. **Getting the topics**

****

1. **Visualizing the topics**

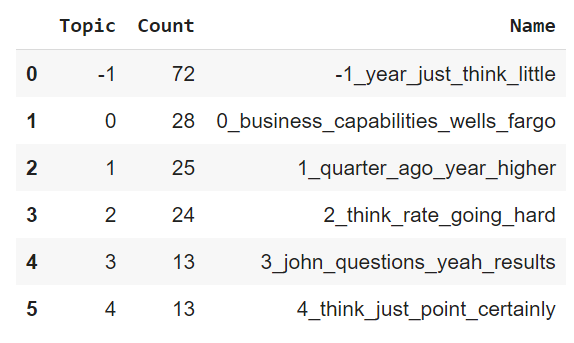
****

1. **Visualizing the topics using bar chart**

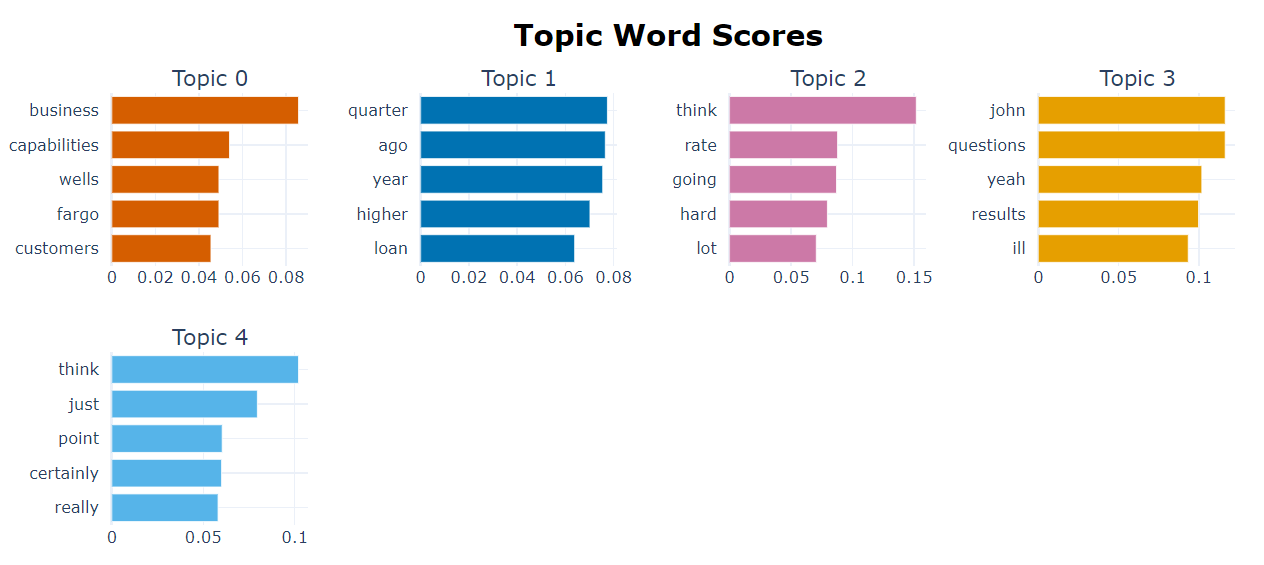
****

**Using the count vectorizer and the conversations from different officials**

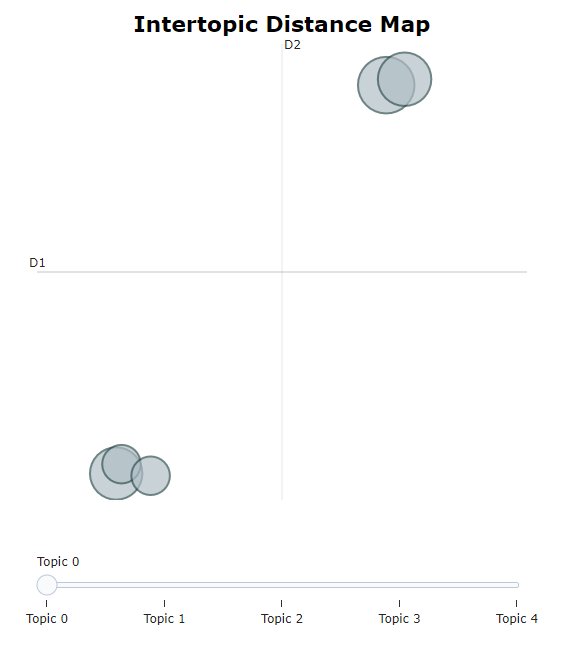
1. **Getting the topics**

****

1. **Visualizing the topics using bar chart**

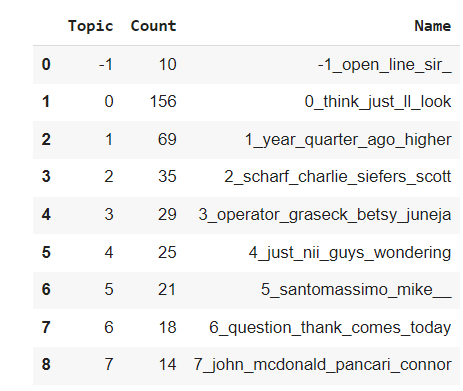
****

**c) Visualizing the topics**

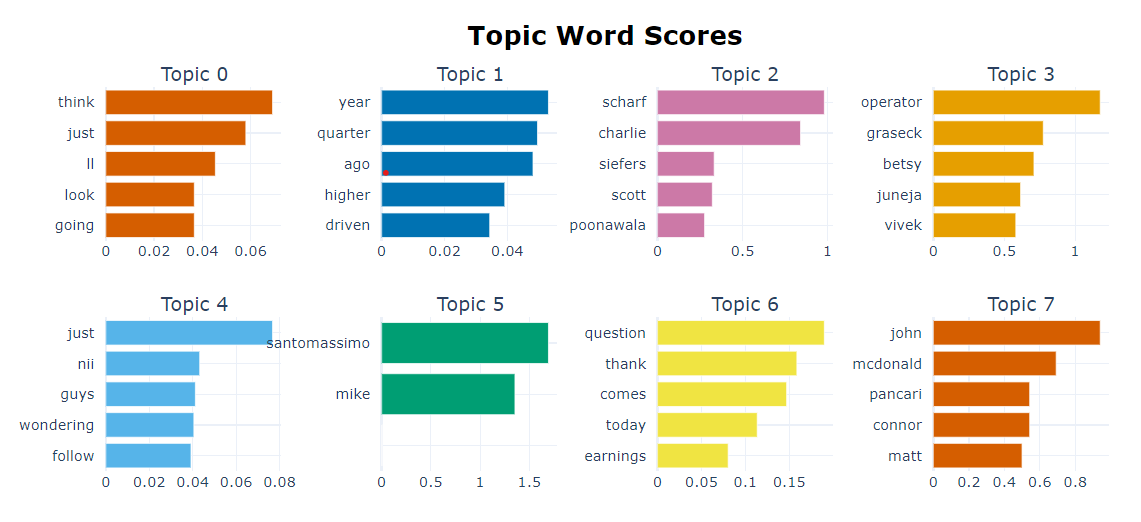
****

**Using Transformers:**

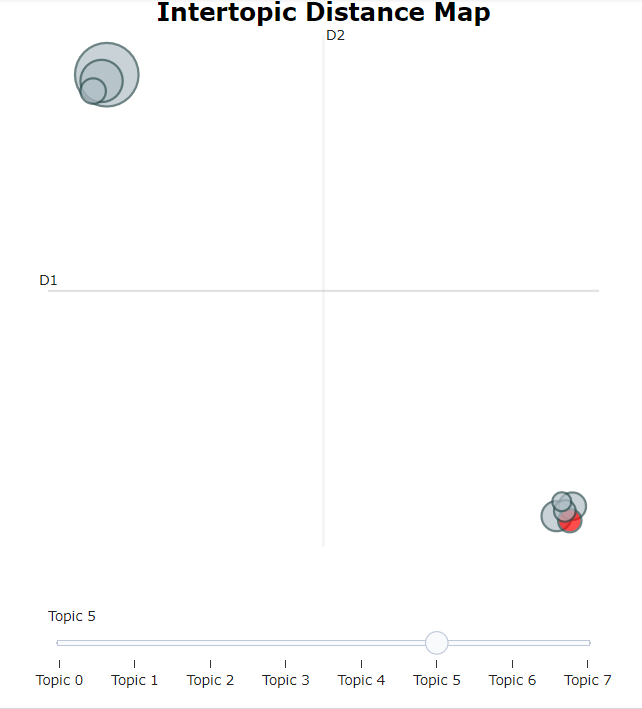
1. **Roberta-base**
2. **Getting the topics**

****

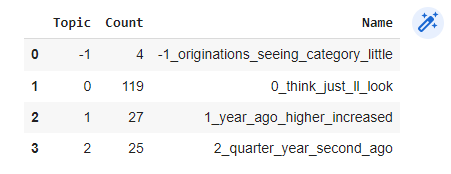
1. **Visualizing the topics**

****

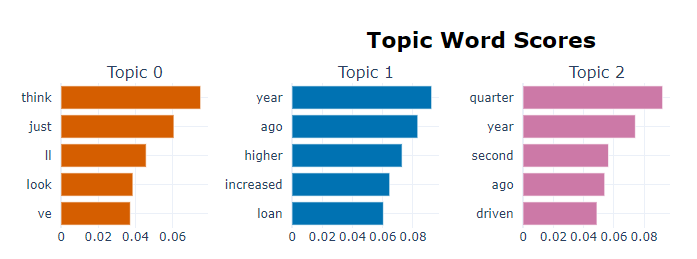
1. **Visualizing the topics using bar chart**

****

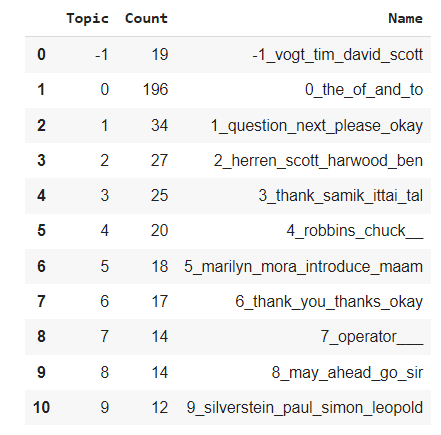
1. **Finbert**
2. **Getting the topics**

****

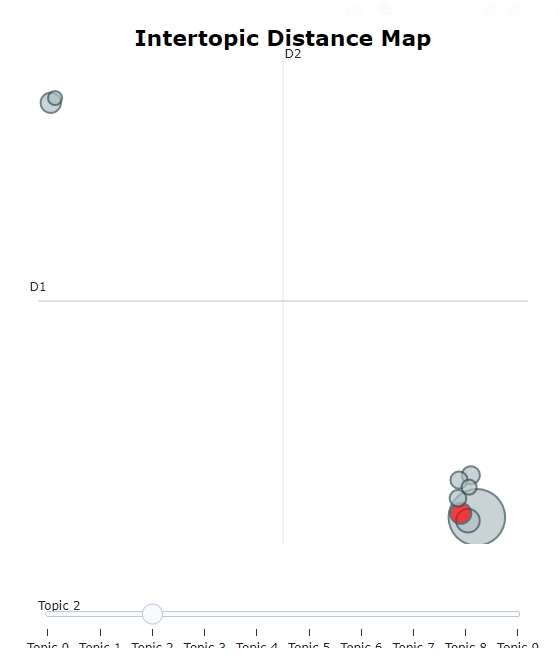
1. **Visualizing the topics using bar chart**

****

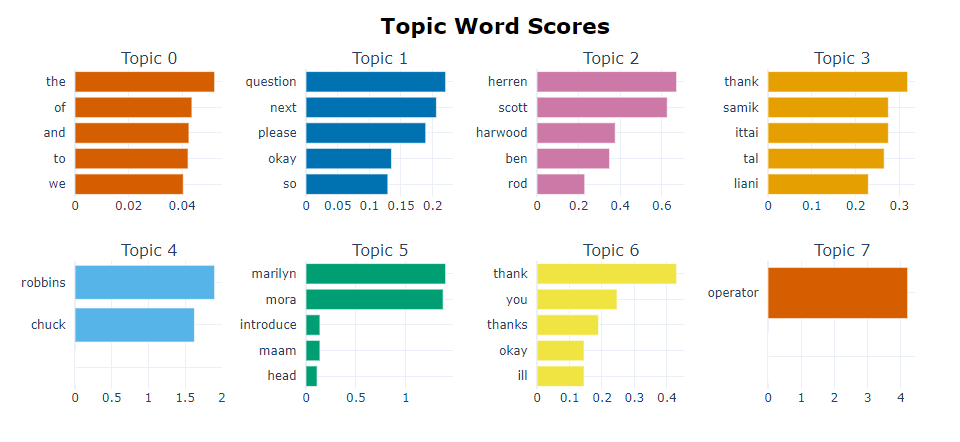
1. **Cisco dataset**
2. **Getting the topics**

****

**ii) Visualizing the topics**

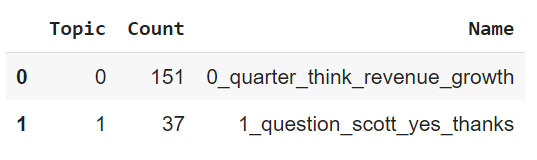
****

**iii) Visualizing the topics using bar chart**

****

**Using the count vectorizer and the conversations from different officials**

1. **Getting the topics**

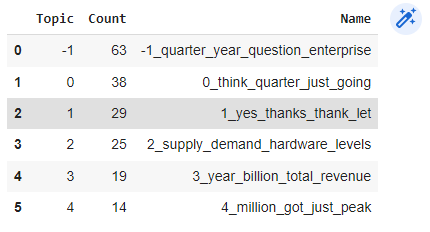
****

1. **Visualizing the topics using bar chart**

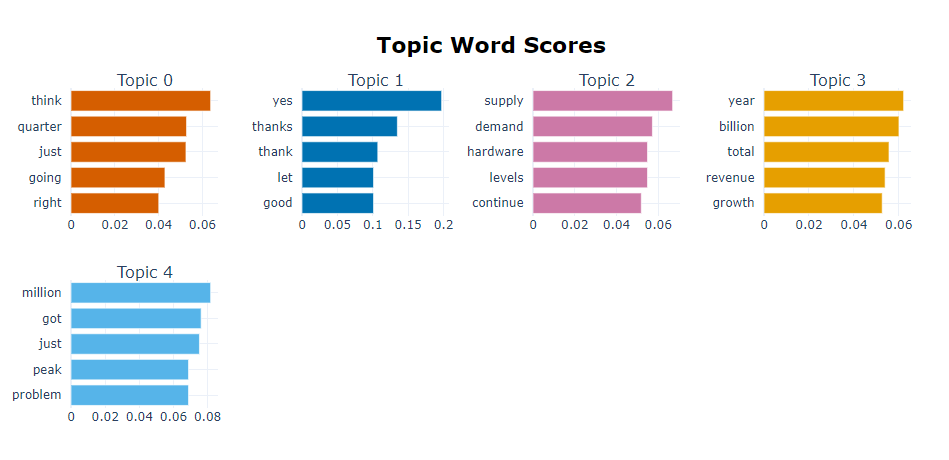
****

**Using transformers:**

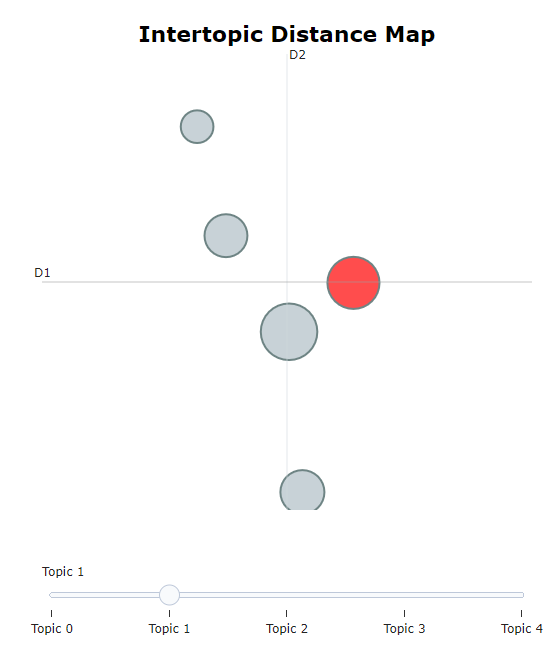
1. **Roberta base**
2. **Getting the topics**

****

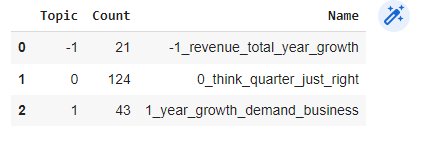
1. **Visualizing the topics using bar chart**

****

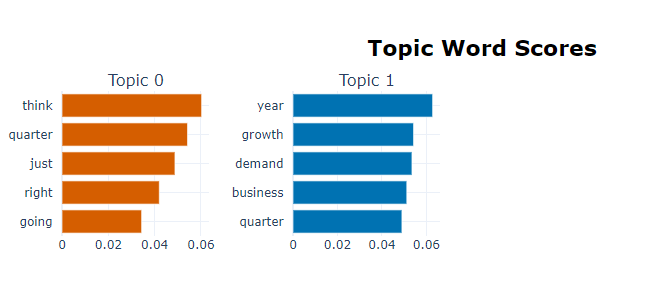
1. **Visualize the topics**

****

1. **Finbert**
2. **Getting the topics**

****

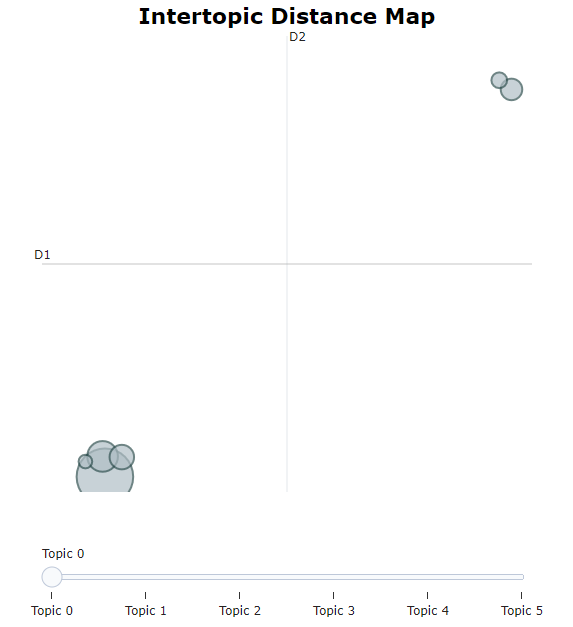
1. **Visualizing the topics using bar chart**

****

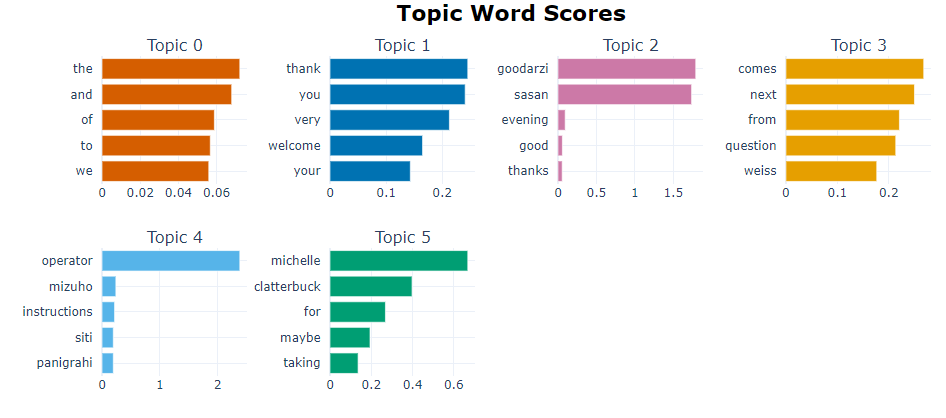
1. **Intuit dataset**
2. **Getting the topics**

****

1. **Visualizing the topics**

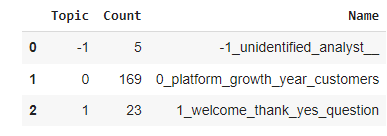
****

1. **Visualizing the topics using bar chart**

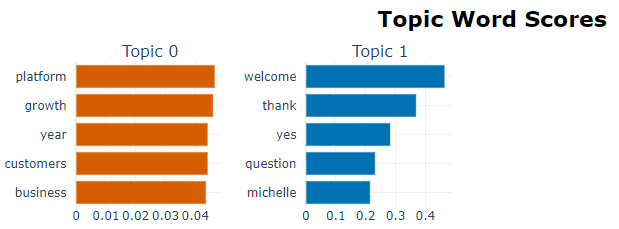
****

**Using the count vectorizer and the conversations from different officials**

1. **Getting the topics**

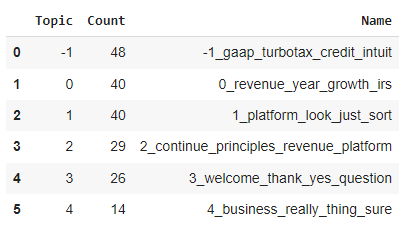
****

1. **Visualizing the topics using bar chart**

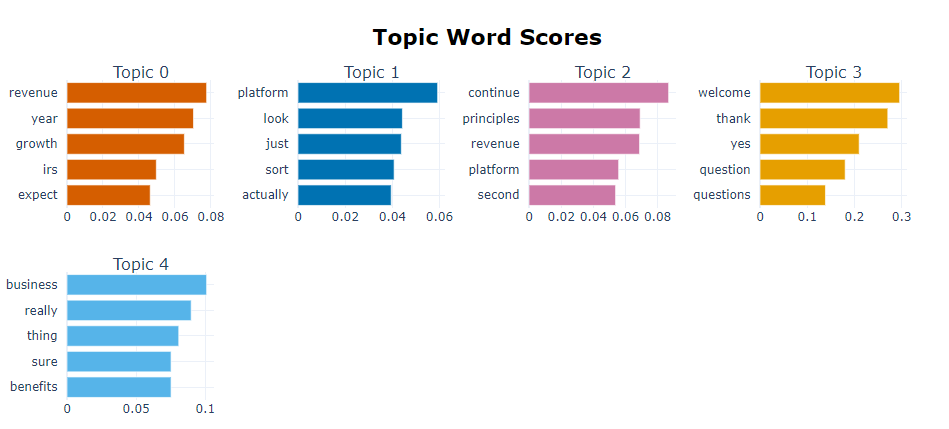
****

**Using transformers:**

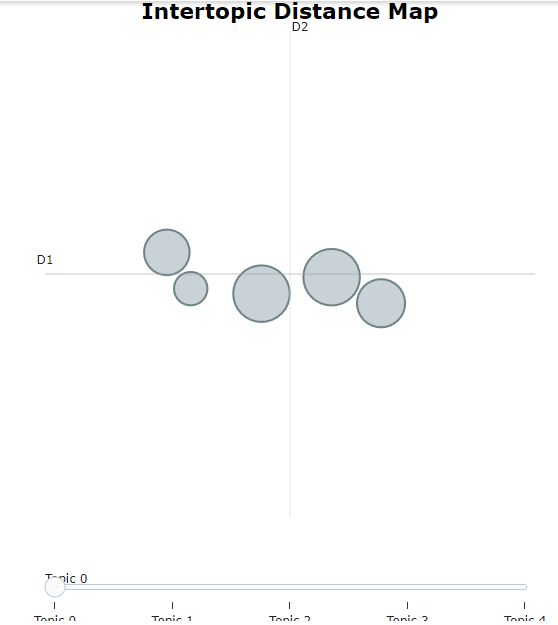
1. **Roberta-base**
2. **Getting the topics**

****

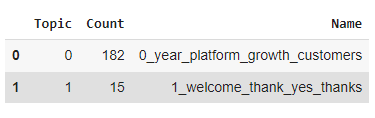
1. **Visualizing the topics using bar chart**

****

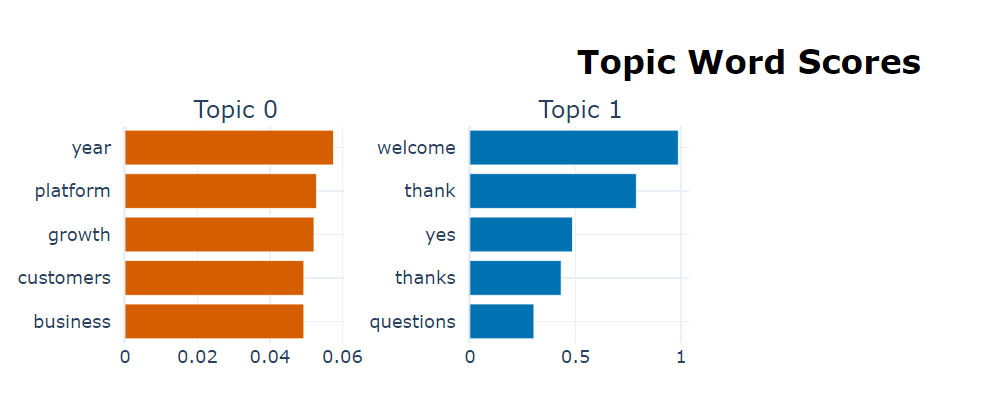
1. **Visualizing the topics**

****

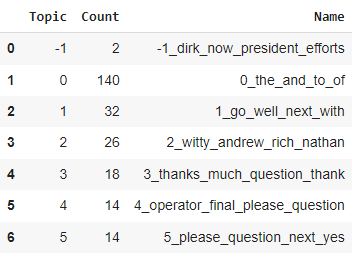
1. **Finbert**
2. **Getting the topics**

****

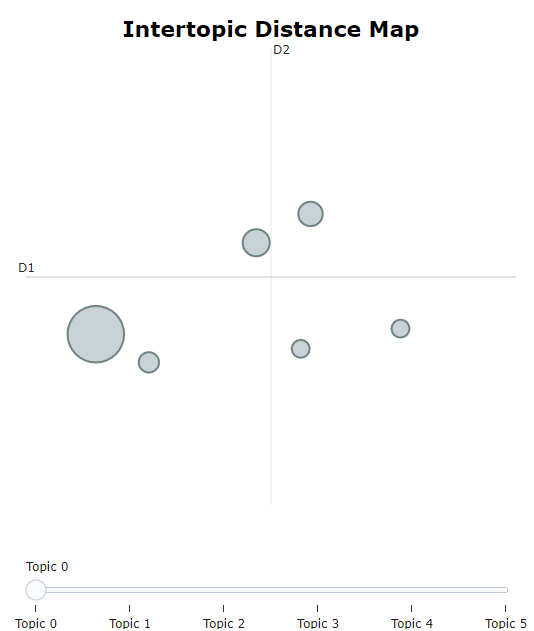
1. **Visualizing the topics using bar chart**

****

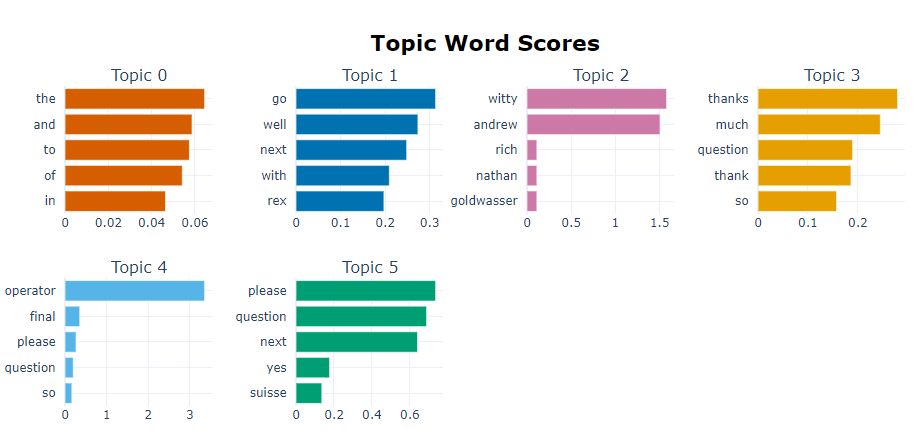
1. **UHG dataset**
2. **Getting the topics**

****

1. **Visualizing the topics**

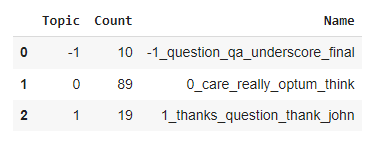
****

1. **Visualizing the topics using bar chart**

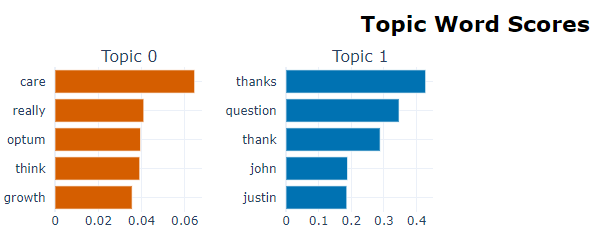
****

**Using the count vectorizer and the conversations from different officials**

1. **Getting the topics**

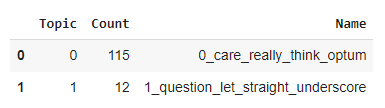
****

1. **Visualizing the topics using bar chart**

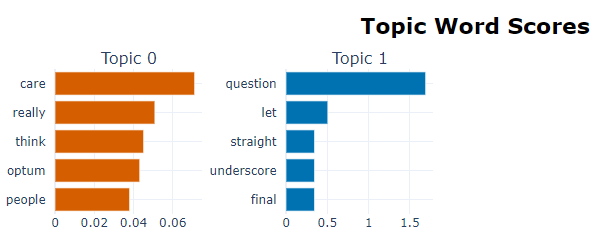
****

**Using transformers:**

1. **Roberta-base**
2. **Getting the topics**

****

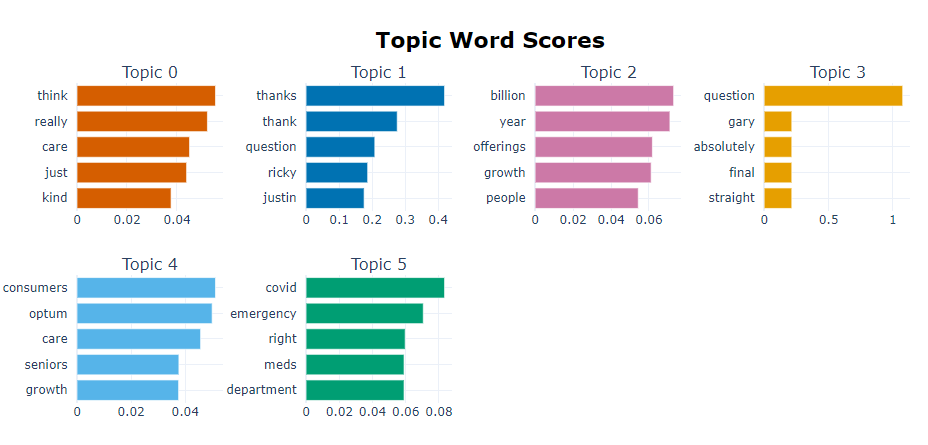
1. **Visualizing the topics using bar chart**

****

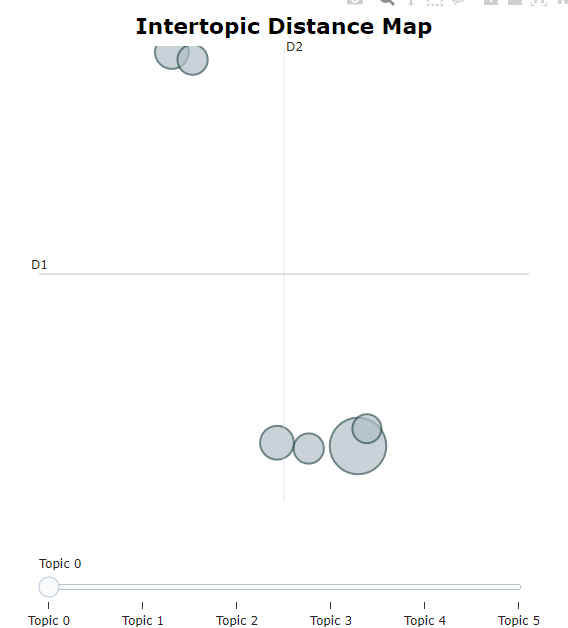
1. **Finbert**
2. **Getting the topics**

****

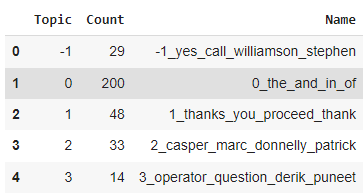
1. **Visualizing the topics using bar chart**

****

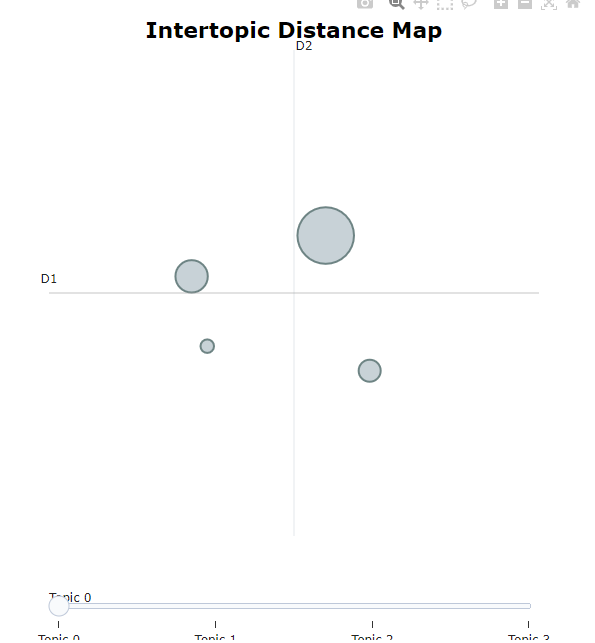
1. **Visualizing the topics**

****

1. **ThermoFisher dataset**
2. **Getting the topics**

****

1. **Visualizing the topics**

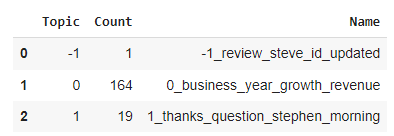
****

1. **Visualizing the topics using bar chart**

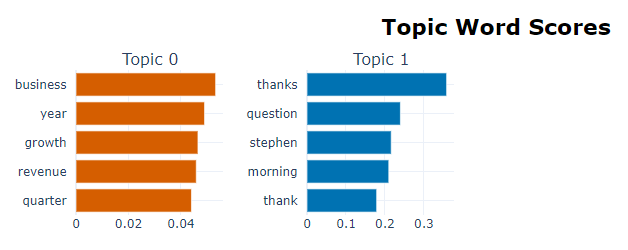
****

**Using the count vectorizer and the conversations from different officials**

1. **Getting the topics**

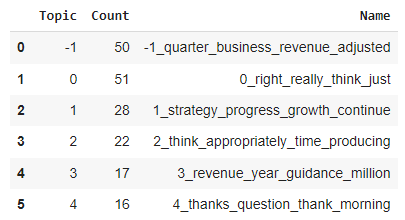
****

1. **Visualizing the topics using bar chart**

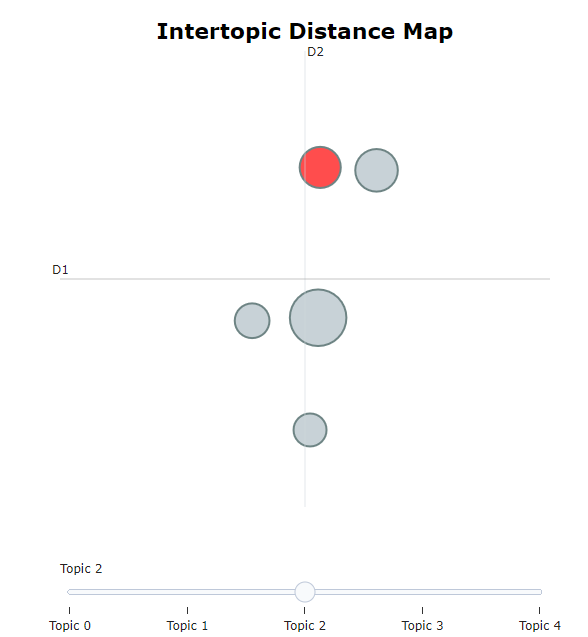
****

**Using transformers:**

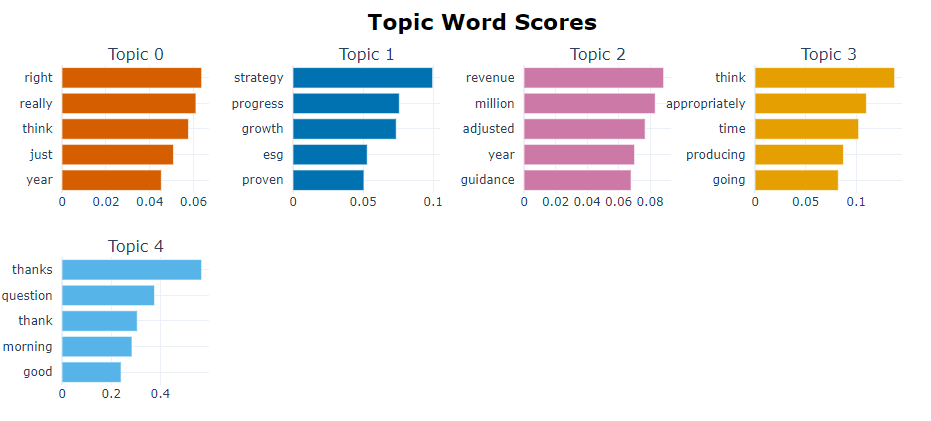
1. **Roberta-base**
2. **Getting the topics**

****

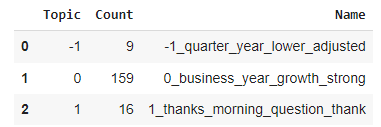
1. **Visualizing the topic**

****

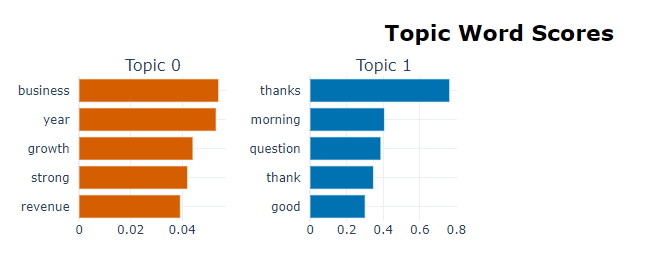
1. **Visualizing the topics using bar chart**

****

1. **Finbert**
2. **Getting the topics**

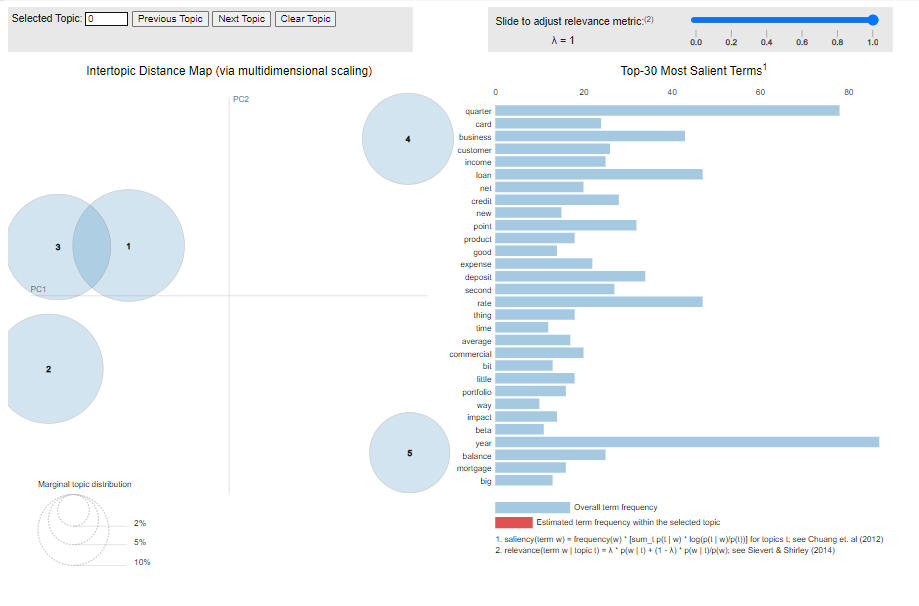
****

1. **Visualizing the topics using bar chart**

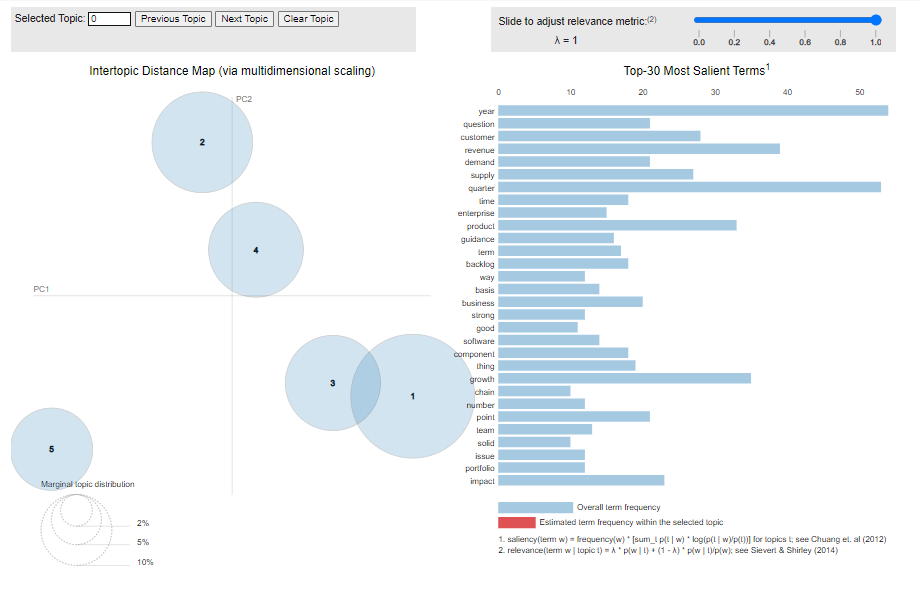
****

**LDA (Latent Dirichlet Allocation) Algorithm Visualization:**

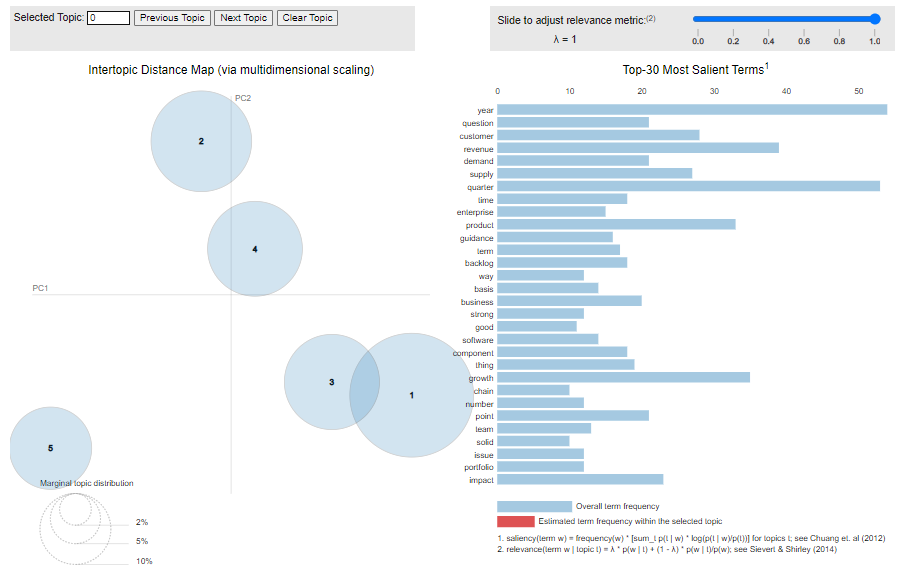
1. **Wells Fargo dataset**

****

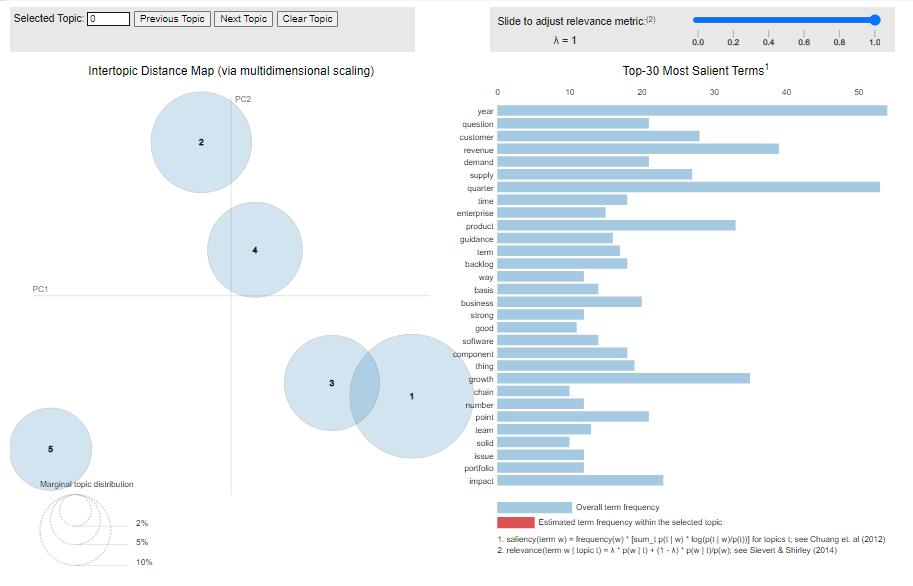
1. **Cisco dataset**

****

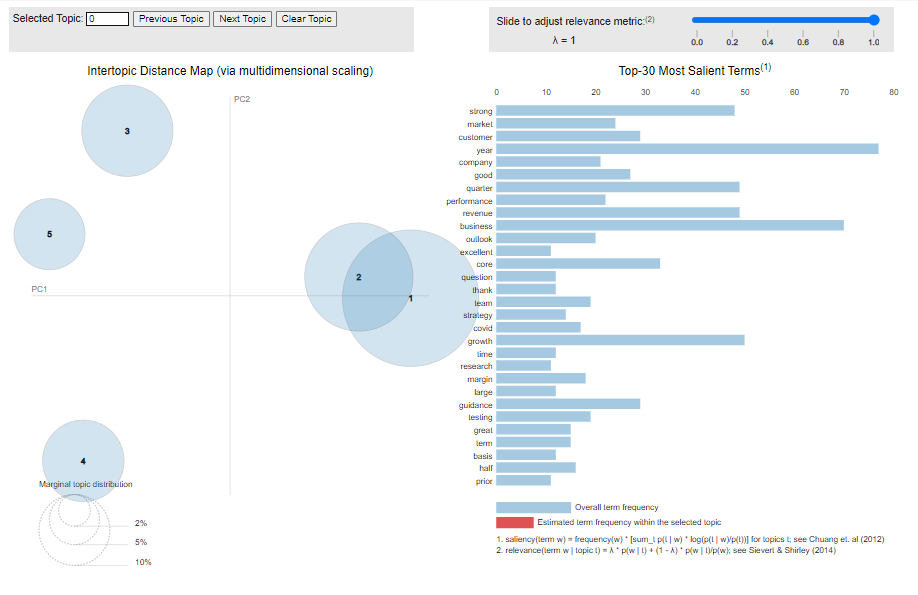
1. **Intuit dataset**

****

1. **UHG dataset**

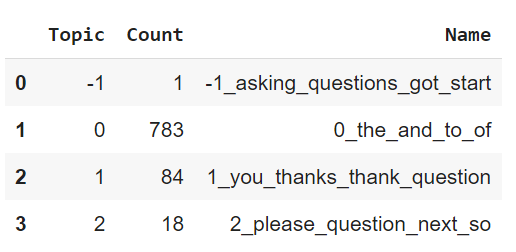


1. **ThermoFisher dataset**

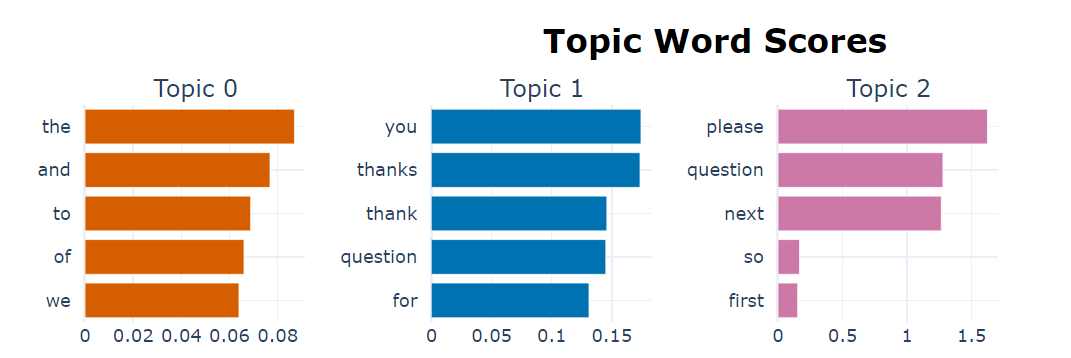


**Results of Combination of Entire Dataset**

1. **Getting the topics**

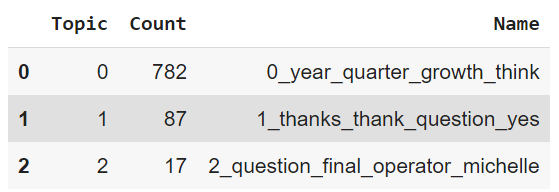


1. **Visualizing the barchart**

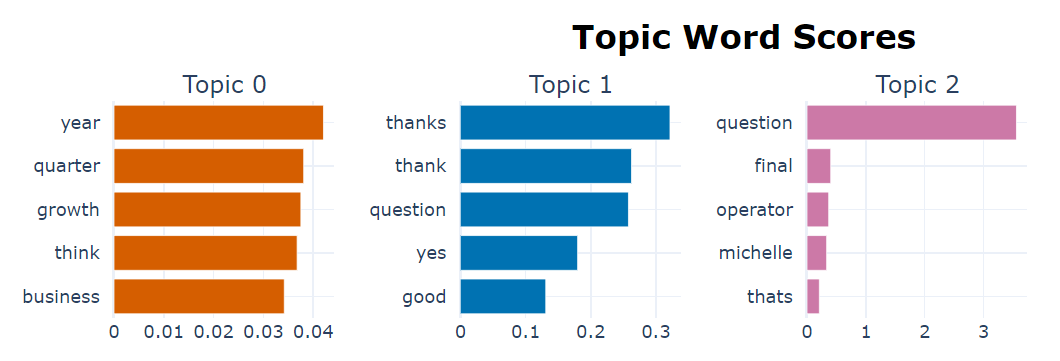


**Using the count vectorizer**

1. **Getting the topics**

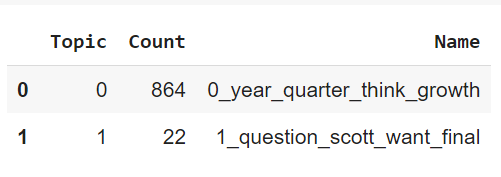
****

1. **Visualizing the topics using bar chart**

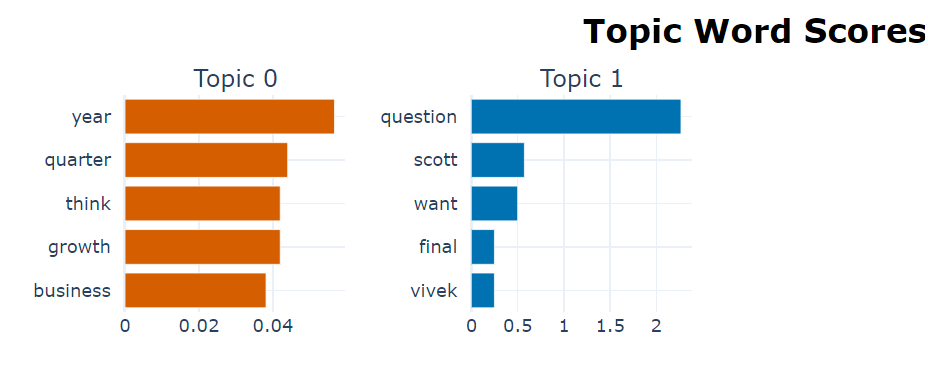
****

**Using transformers**

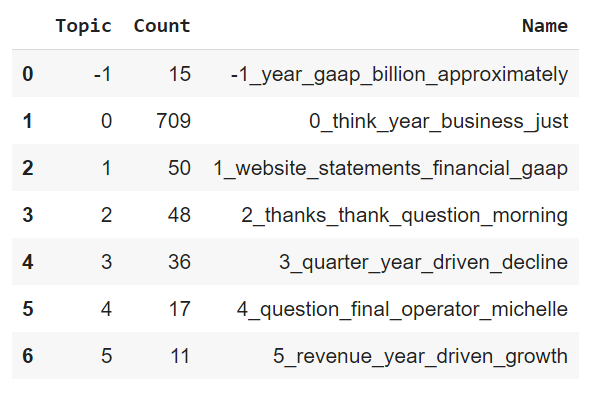
1. **Roberta-base**
2. **Getting the topics**



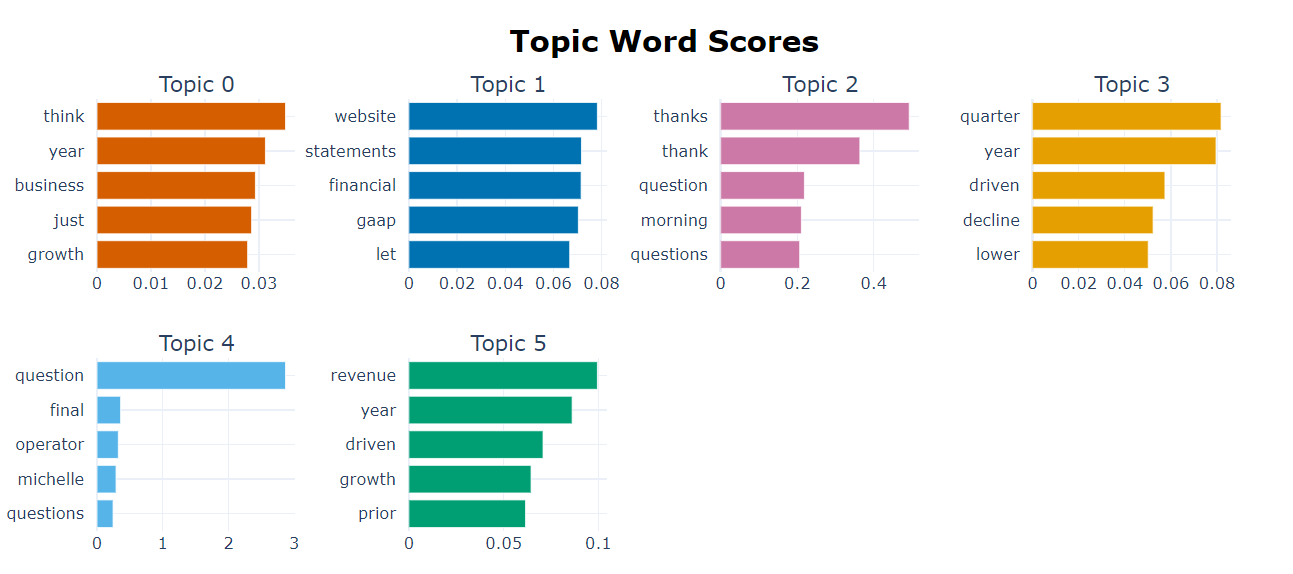
1. **Visualizing the topics using bar chart**



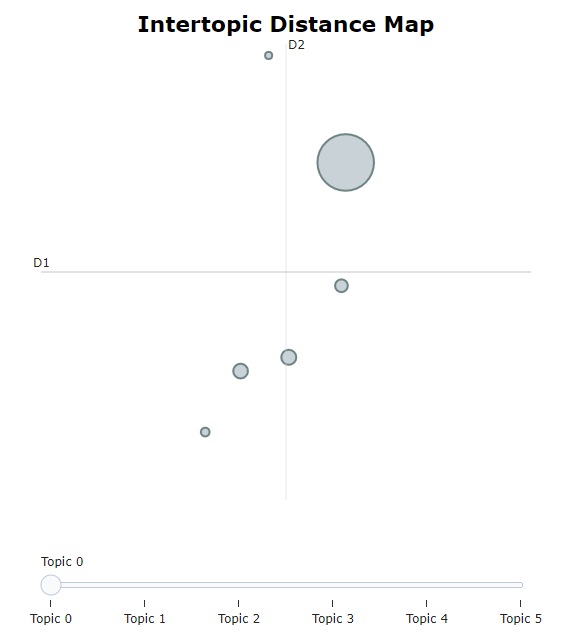
1. **Finbert**
2. **Getting the topics**



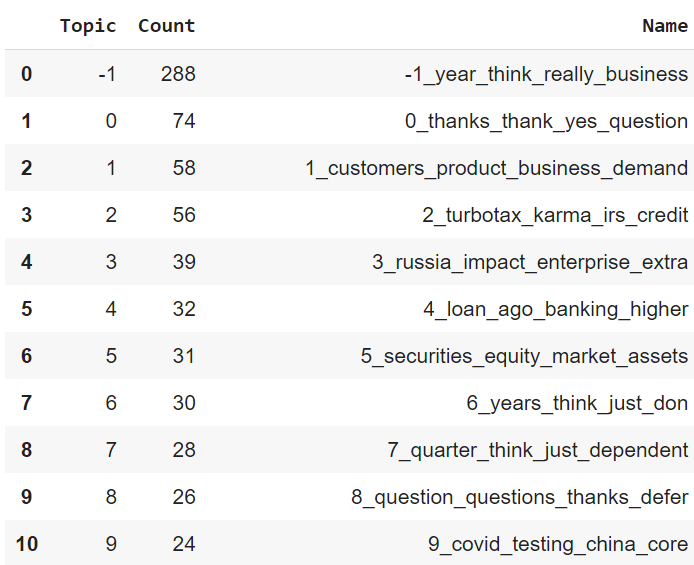
1. **Visualizing the topics using bar chart**

****

**c) Visualizing the topics**

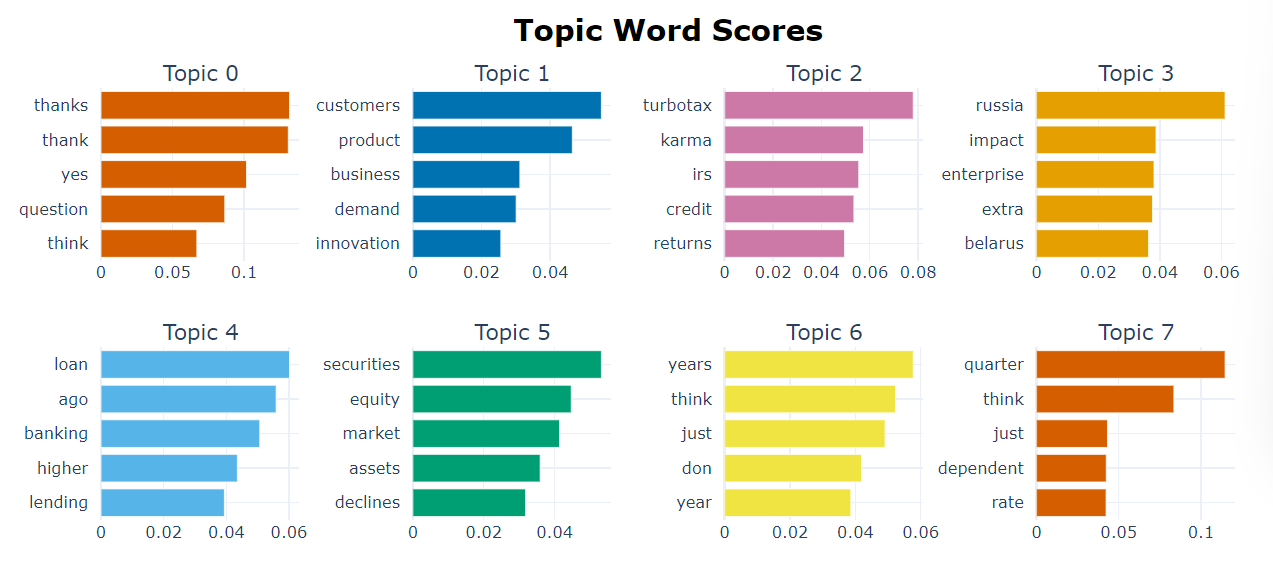
****

1. **all-MiniLM-L6-v2**
2. **Getting the topics**

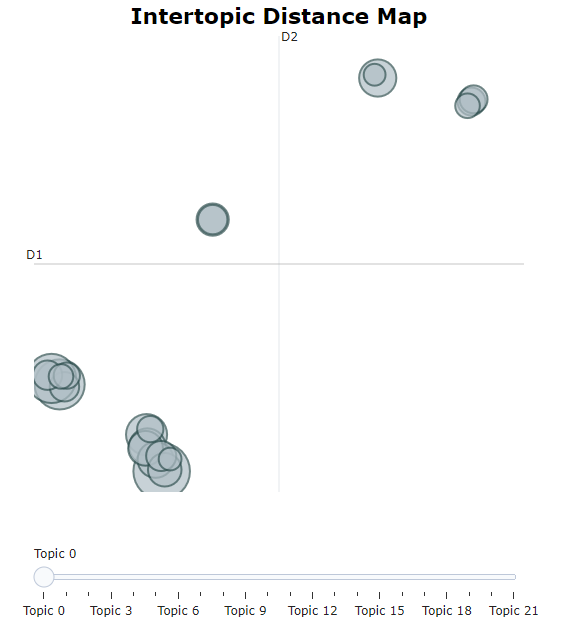
****

****

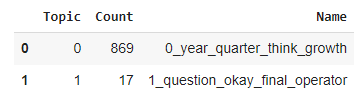
1. **Visualizing the topics using bar chart**

****

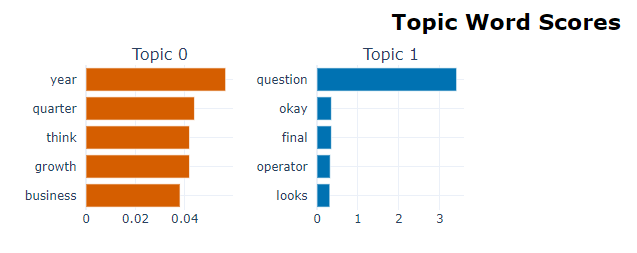
1. **Visualizing the topics**

****

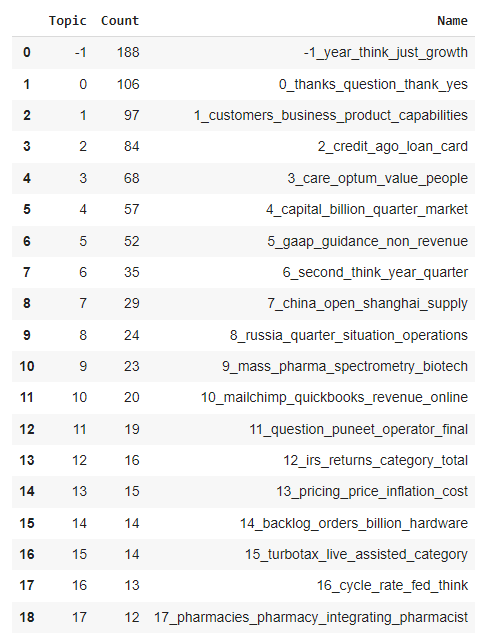
1. **all-mpnet-base-v2**
2. **Getting the topics**

****

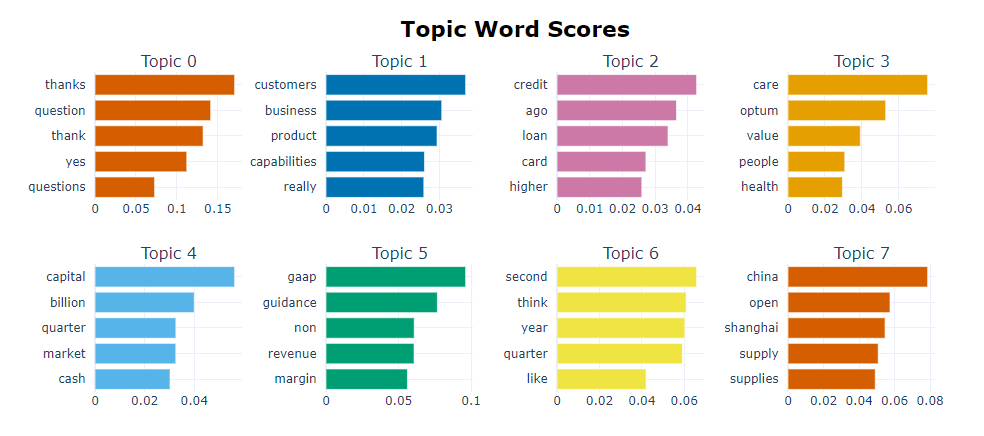
1. **Visualizing the topics using bar chart**

****

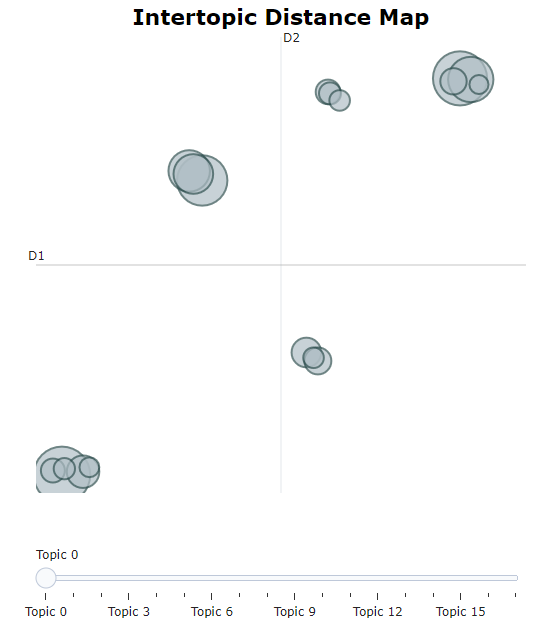
1. **all-distilroberta-v1**
2. **Getting the topics**

****

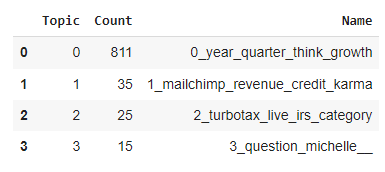
1. **Visualizing the topics using bar chart**

****

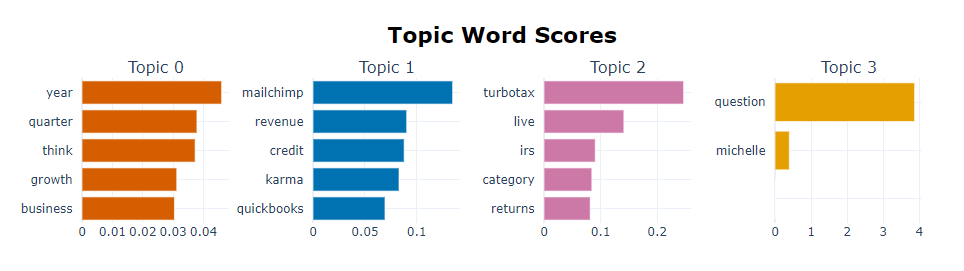
1. **Visualizing the topics**

****

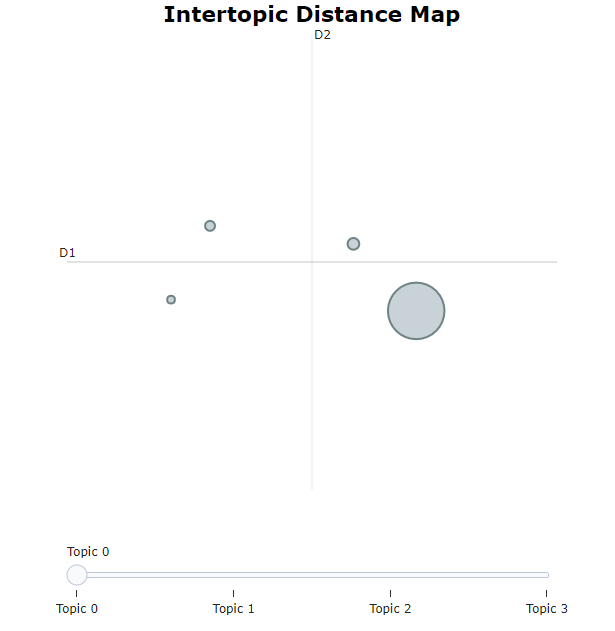
1. **all-MiniLM-L12-v2**
2. **Getting the topics**

****

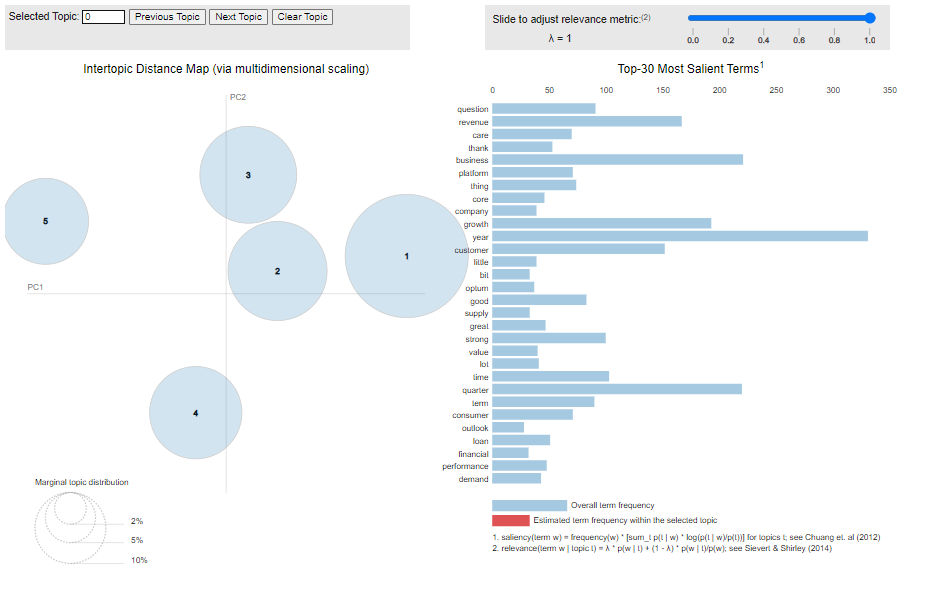
1. **Visualizing the topics using bar chart**

****

1. **Visualizing the topics**

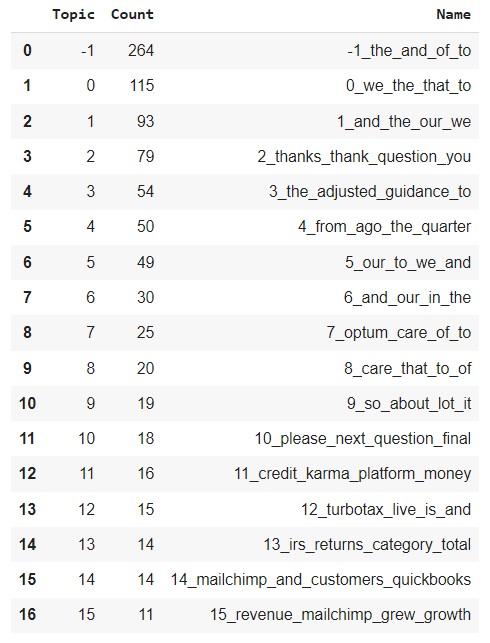
****

**LDA (Latent Dirichlet Allocation) Algorithm Visualization**

****

**Using Dimensionality Reduction Techniques**

1. **UMAP (Uniform Manifold Approximation and Projection)**
2. **Getting the topics**

****

1. **Visualizing the topics**

****

1. **Visualizing the topics using bar chart**

****

1. **PCA (Principal Component Analysis)**
2. **Getting the topics**

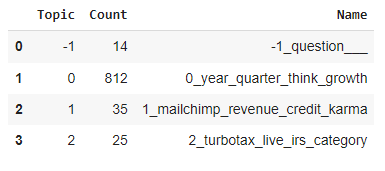
****

1. **Visualizing the topics using bar chart**

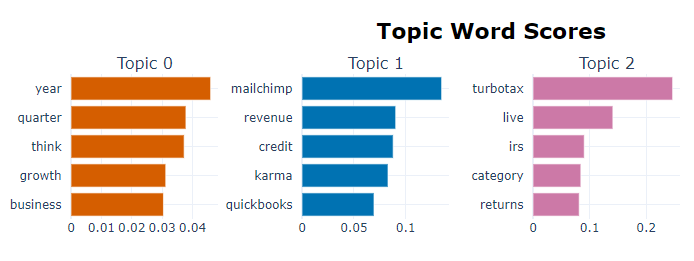
****

**Clustering Algorithms**

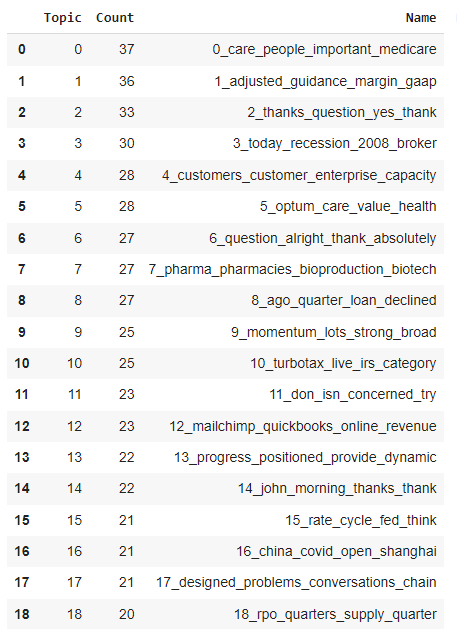
1. **HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) Algorithm**
2. **Getting the topics**

****

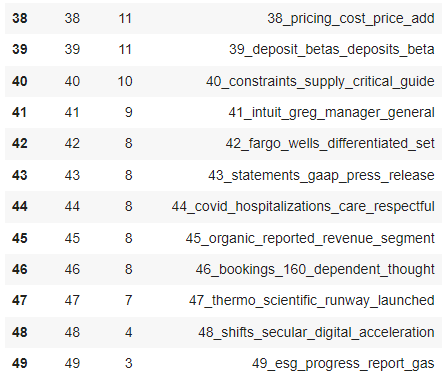
1. **Visualizing the topics using bar chart**

****

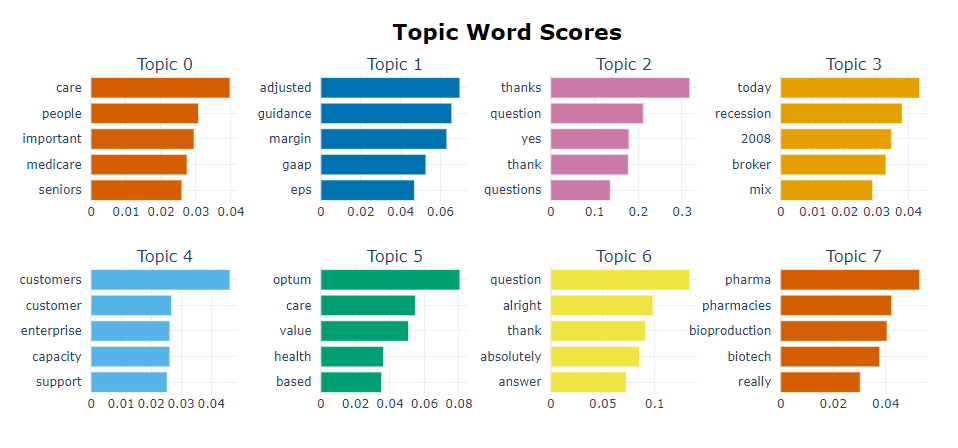
1. **K-Means Clustering Algorithm**
2. **Getting the topics**

****

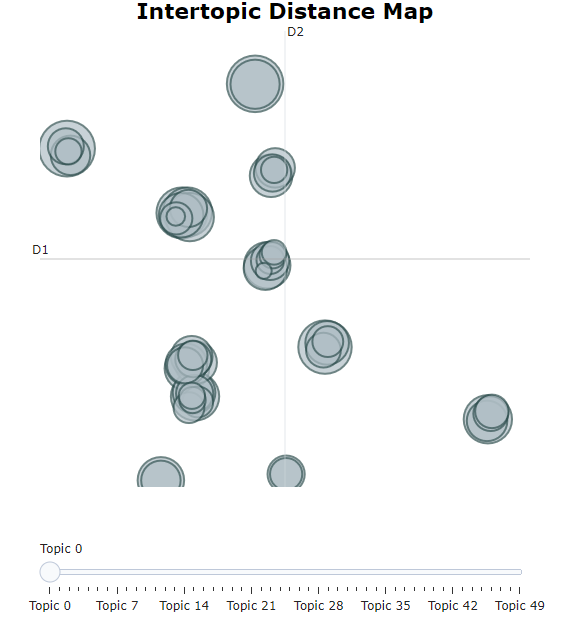
****

****

1. **Visualizing the topics using bar chart**

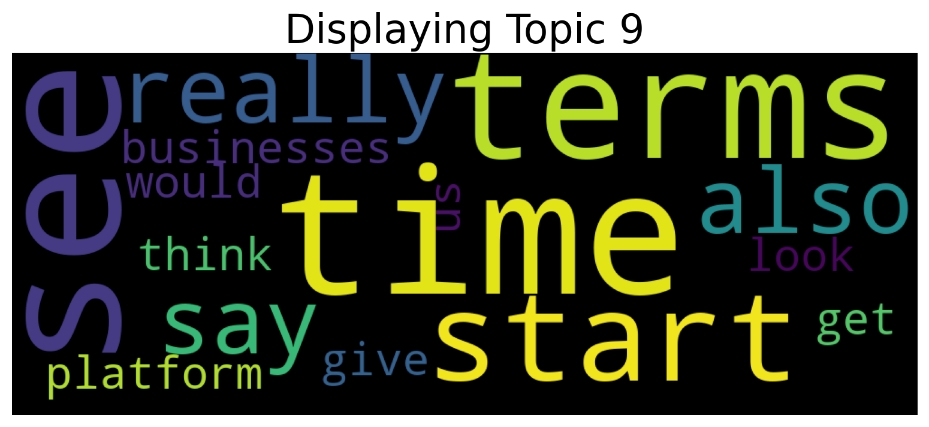
****

1. **Visualizing the topics**

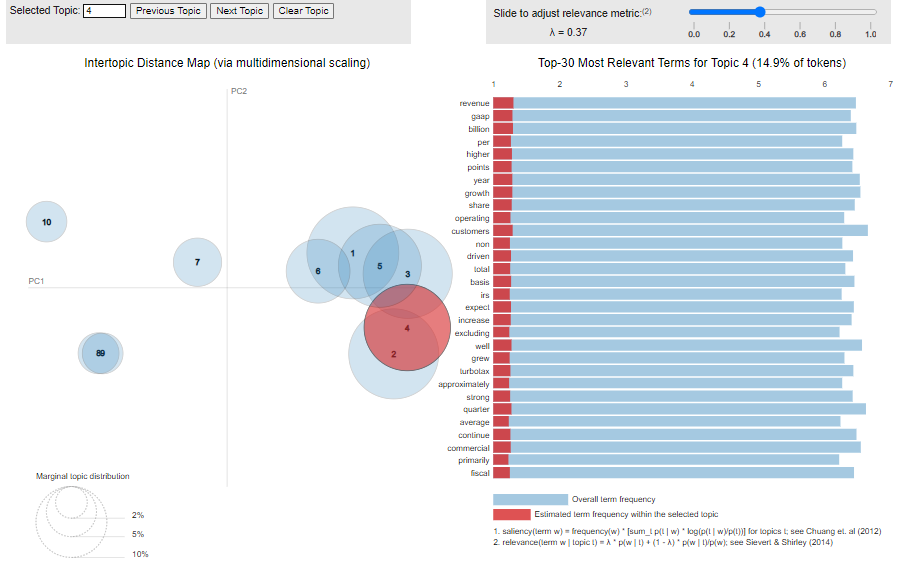
****

**Contextualized Topic Modeling Topic Extraction Results**

* 1. **Using Finbert Transformer**
     1. **Displaying topics using wordcloud**

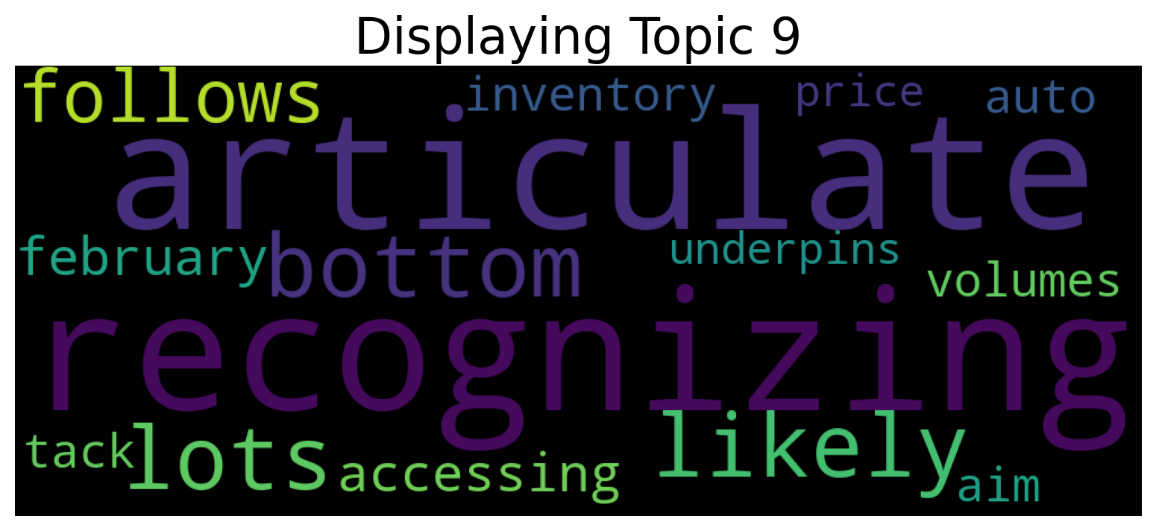
****

* + 1. **Visualizing the topics using PyLDAvis**

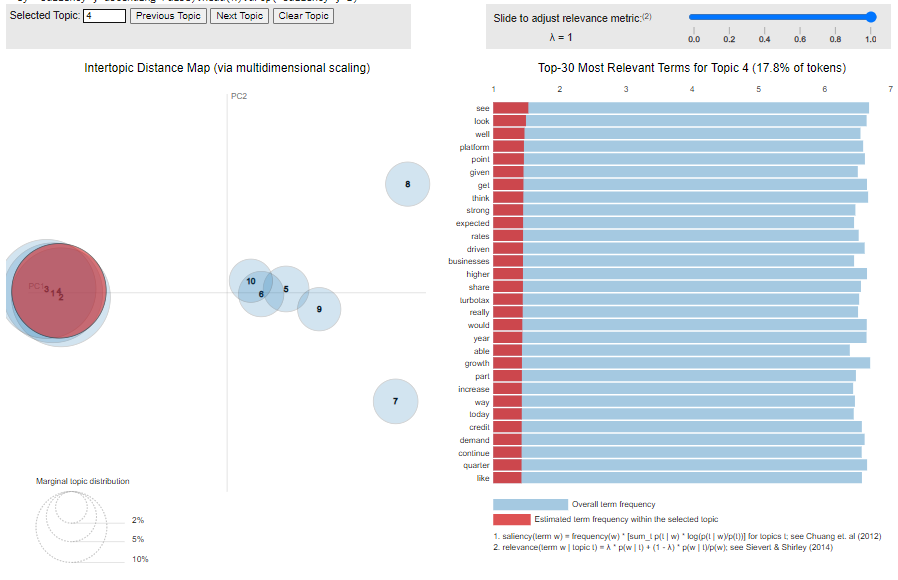
****

## **Using all-mpnet-base-v2 Transformer**

* + 1. **Displaying topics using wordcloud**

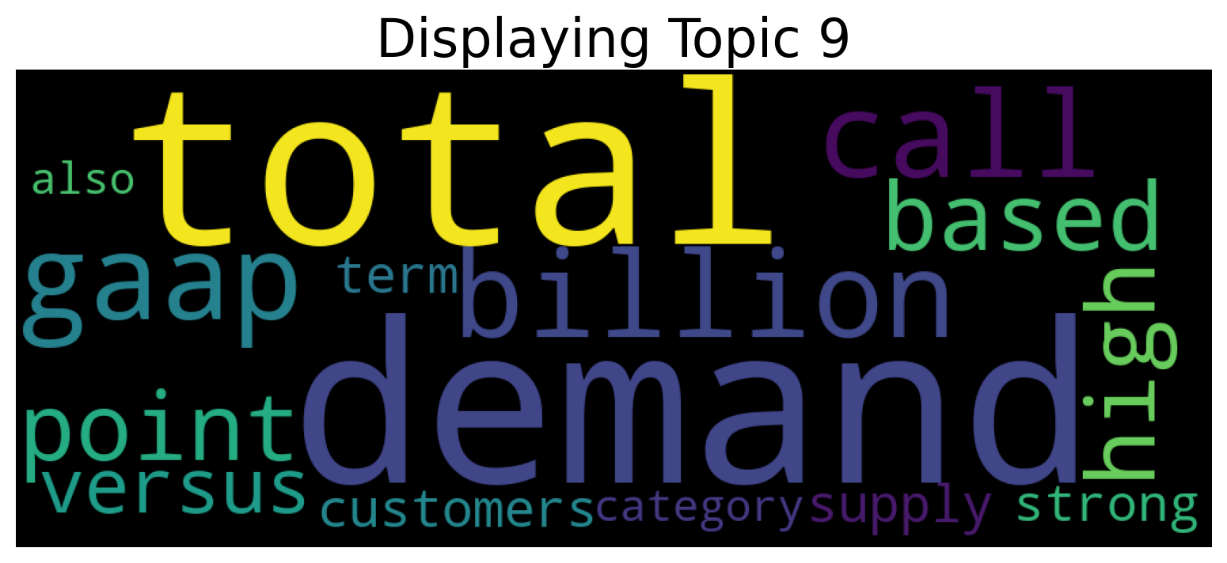
****

* + 1. **Visualizing the topics using PyLDAvis**

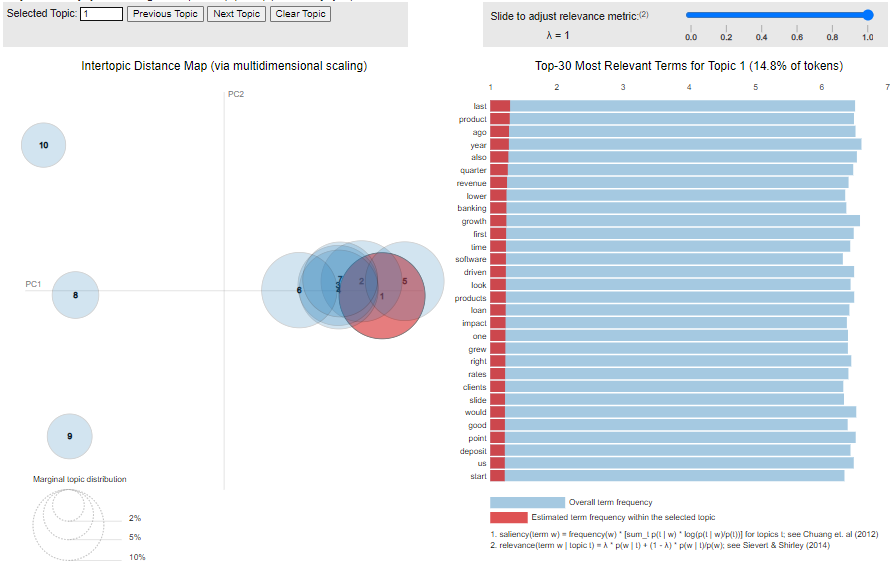
****

## **Using all-distilroberta-v1 Transformer**

* + 1. **Displaying topics using wordcloud**

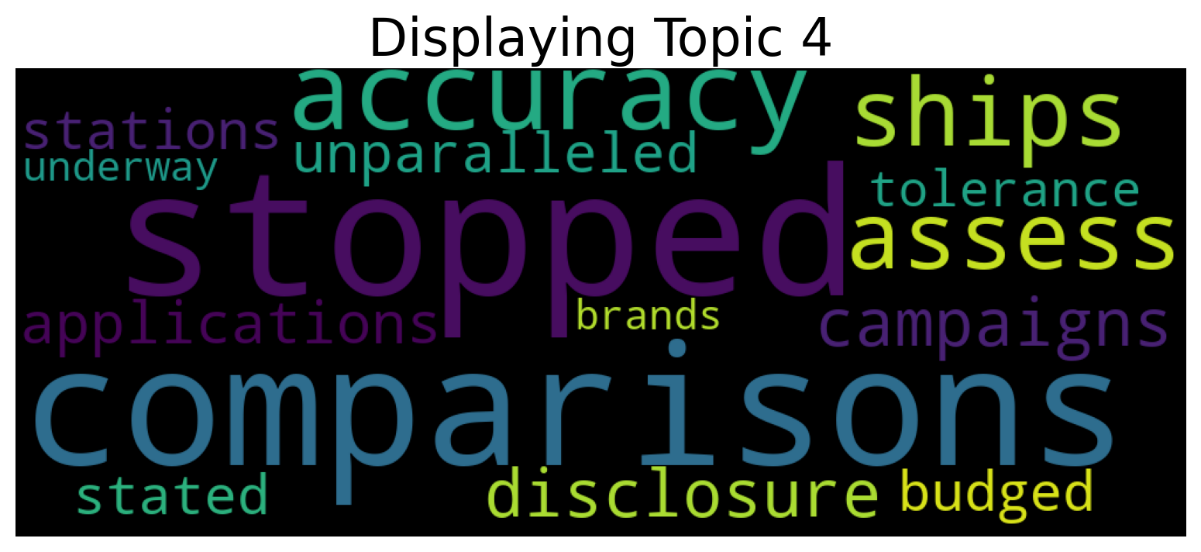
****

* + 1. **Visualizing the topics using PyLDAvis**

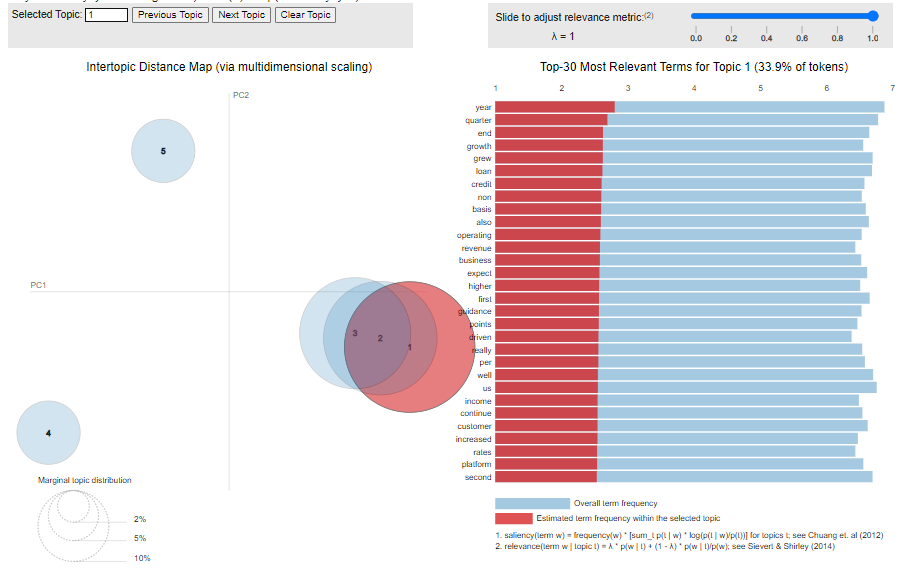
****

## **Using Roberta-base Transformer**

* + 1. **Displaying topics using wordcloud**

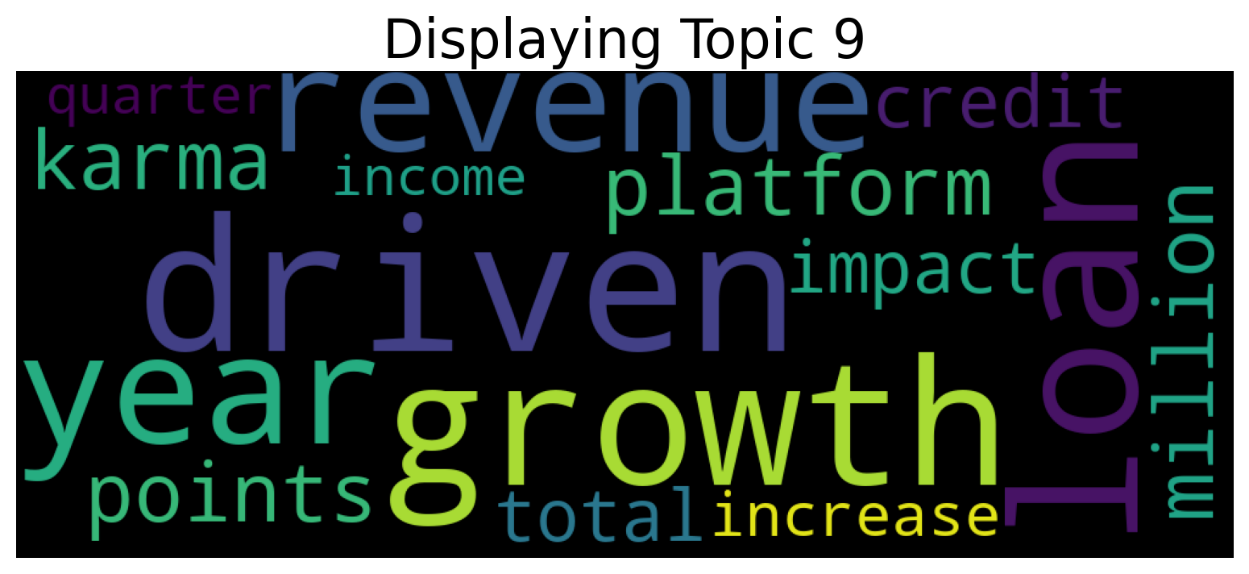
****

* + 1. **Visualizing the topics using PyLDAvis**

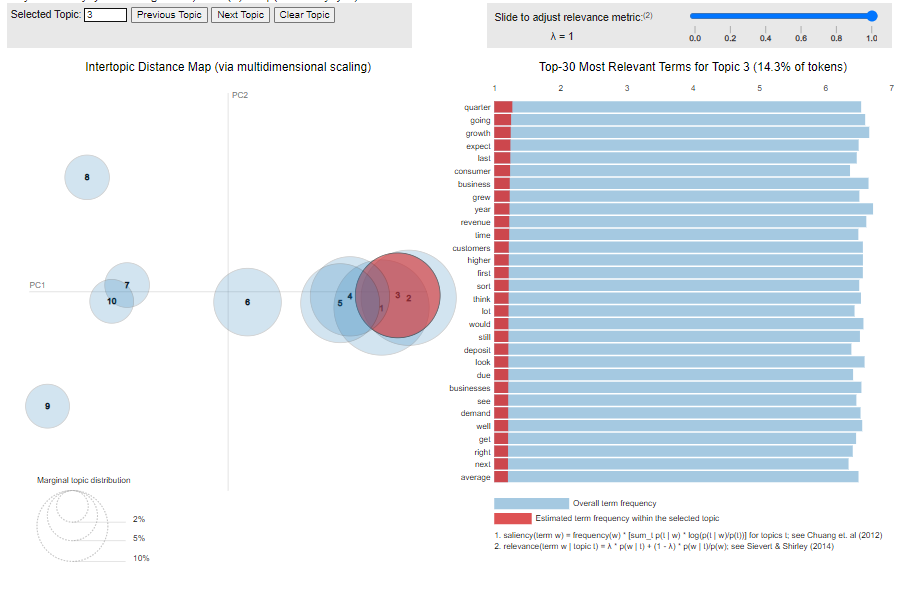
****

## **Using all-MiniLM-L6-v2 Transformer**

* + 1. **Displaying topics using wordcloud**

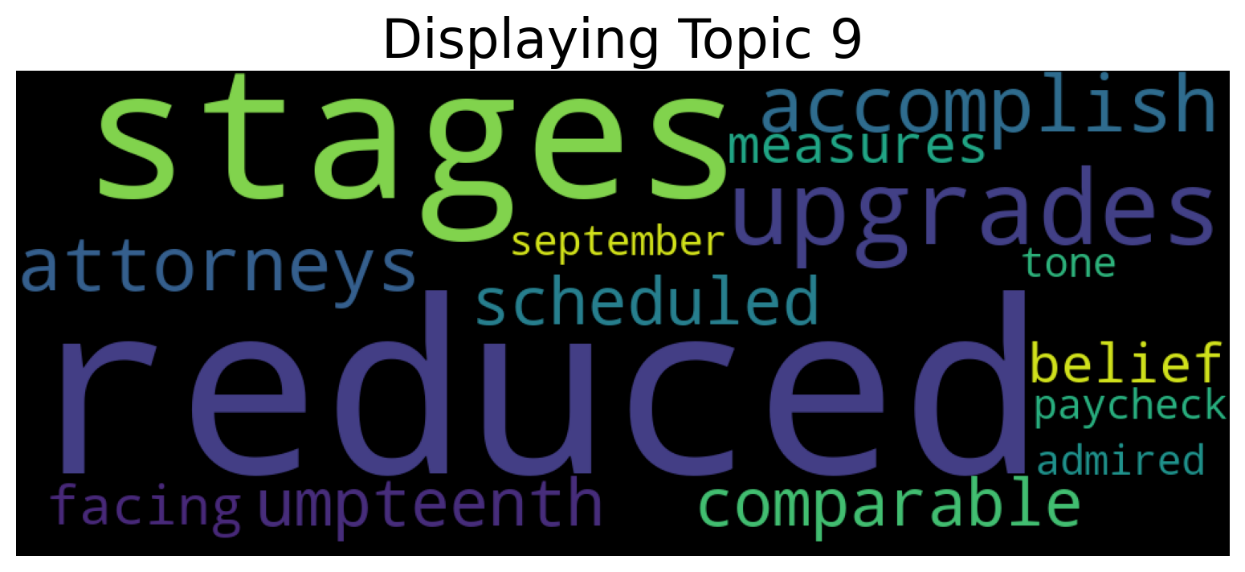
****

* + 1. **Visualizing the topics using PyLDAvis**

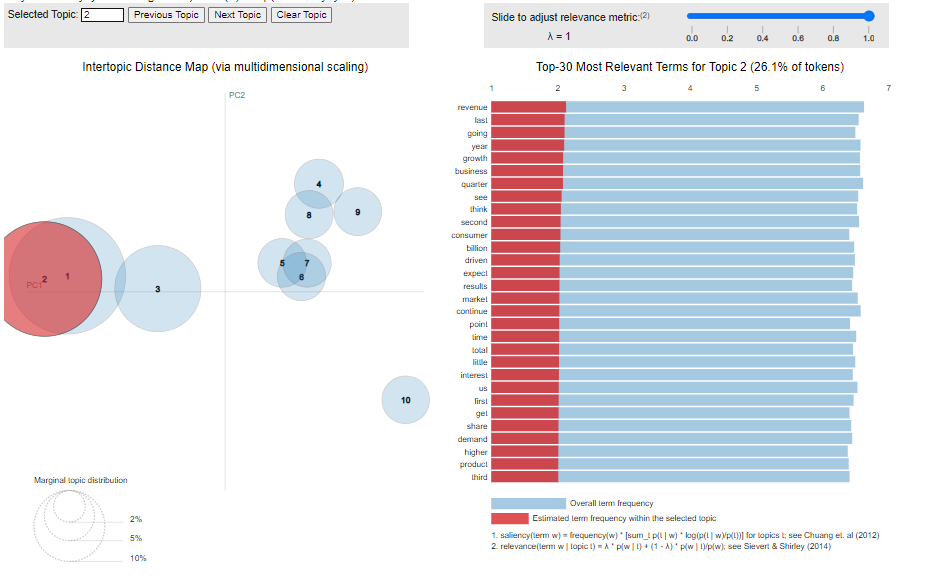
****

## **Using all-MiniLM-L12-v2 Transformer**

* + 1. **Displaying topics using wordcloud**

****

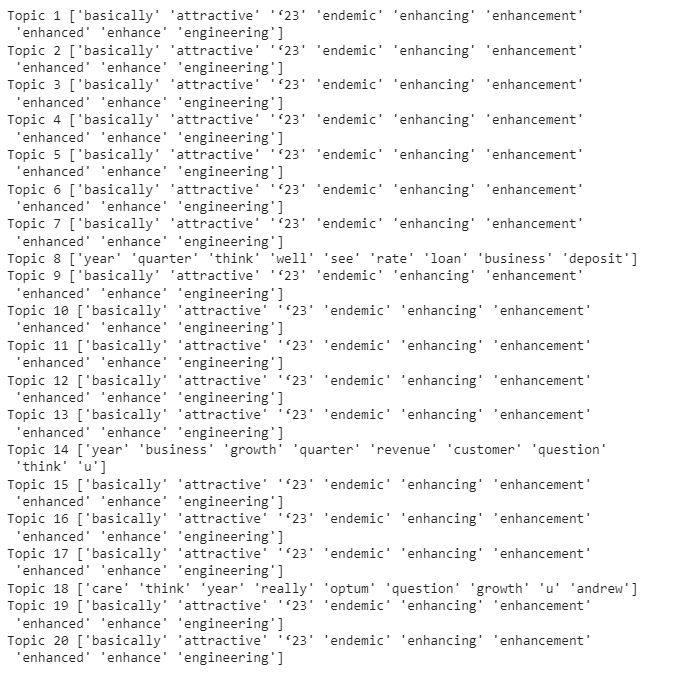
* + 1. **Visualizing the topics using PyLDAvis**



**Sklearn LDA Topic Extraction Results**

**i) Retrieve the topics**

a) Using Count Vectorizer



b) TF-IDF Vectorizer

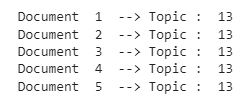


**ii) Annotate the topic documents**

a) Using Count Vectorizer



b) Using TF-IDF Vectorizer

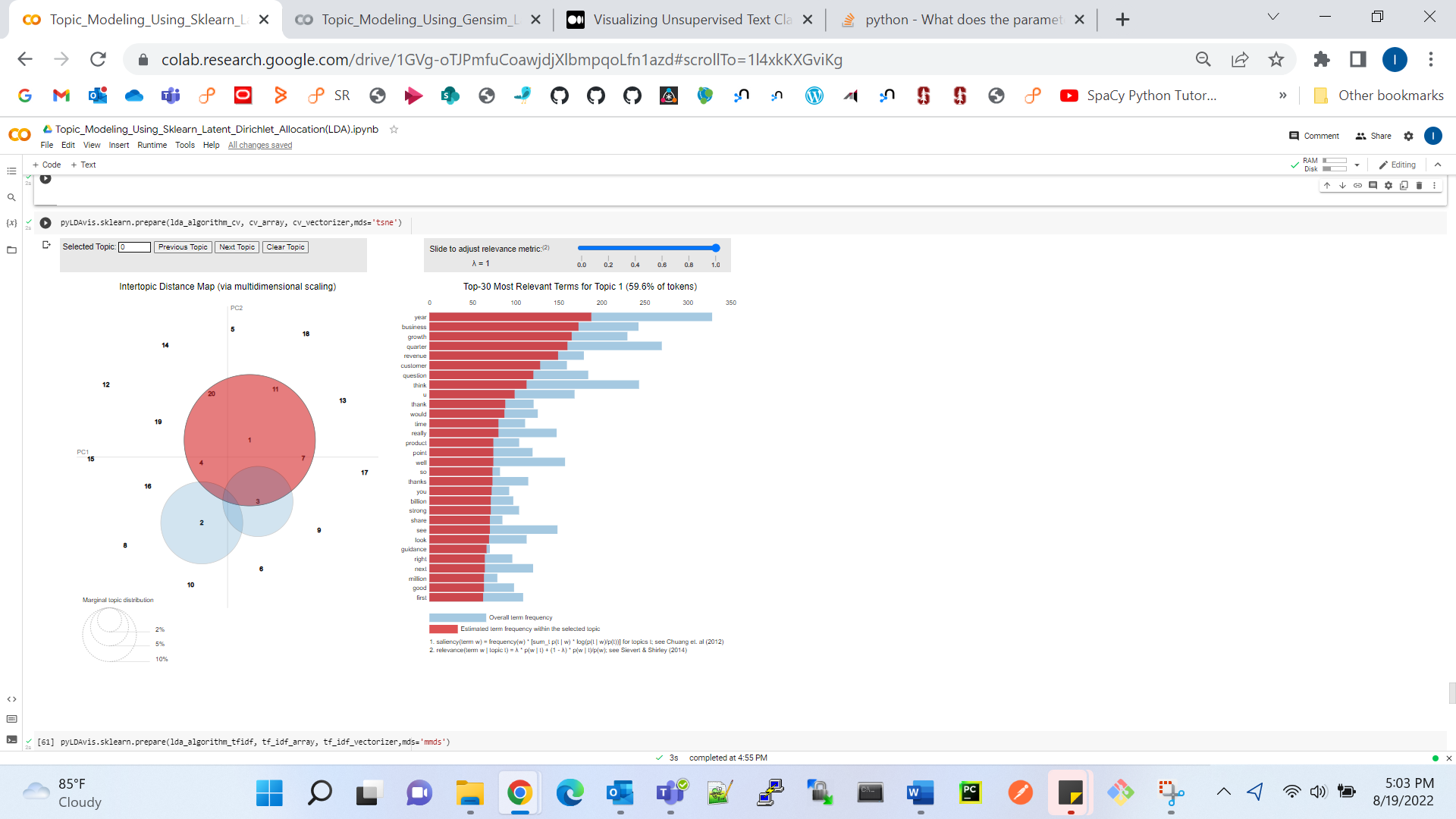


**iii) Using PyLDAvis for Visualization**

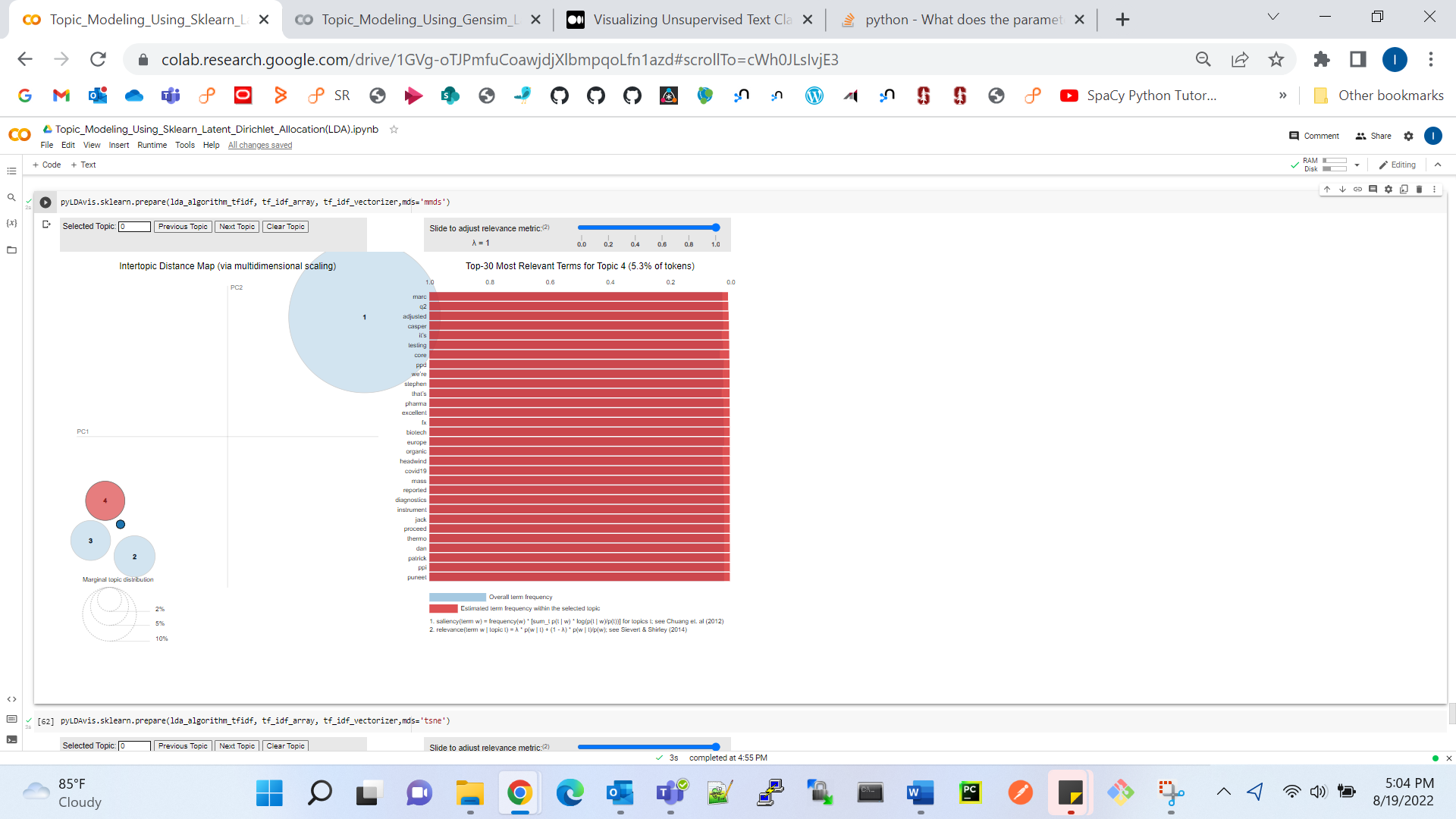
a) Using Count Vectorizer and MMDS (Metric Multidimensional Scaling)



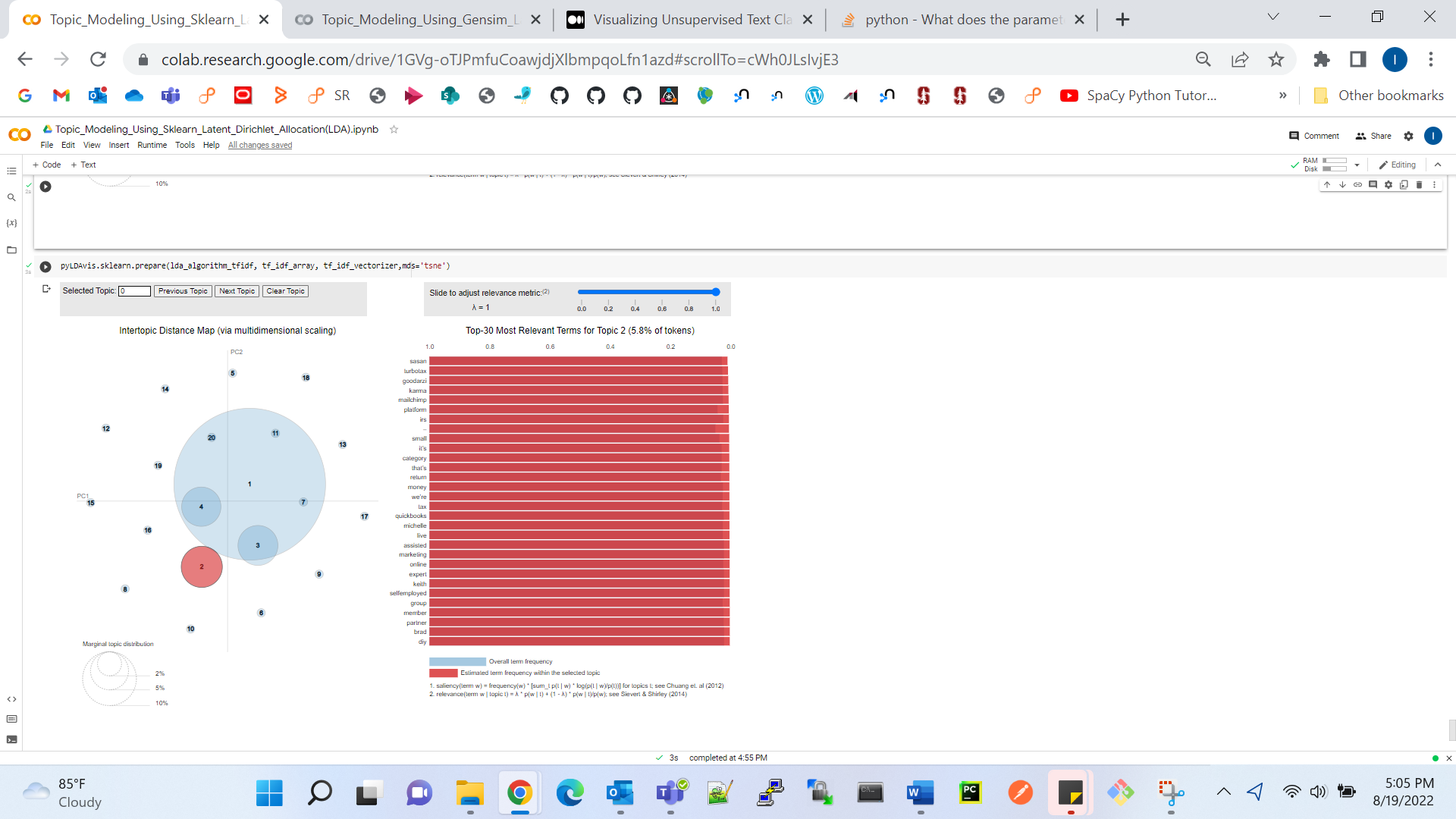
b) Using Count Vectorizer and TSNE (t-distributed stochastic neighbor embedding)



c) Using TF-IDF Vectorizer and MMDS



d) Using TF-IDF Vectorizer and TSNE



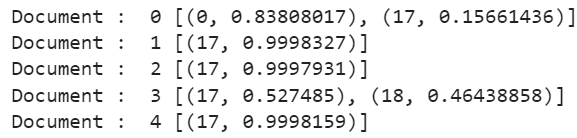
**Gensim LDA Topic Extraction Results**

### **i) Extracting Topics from the Corpus**

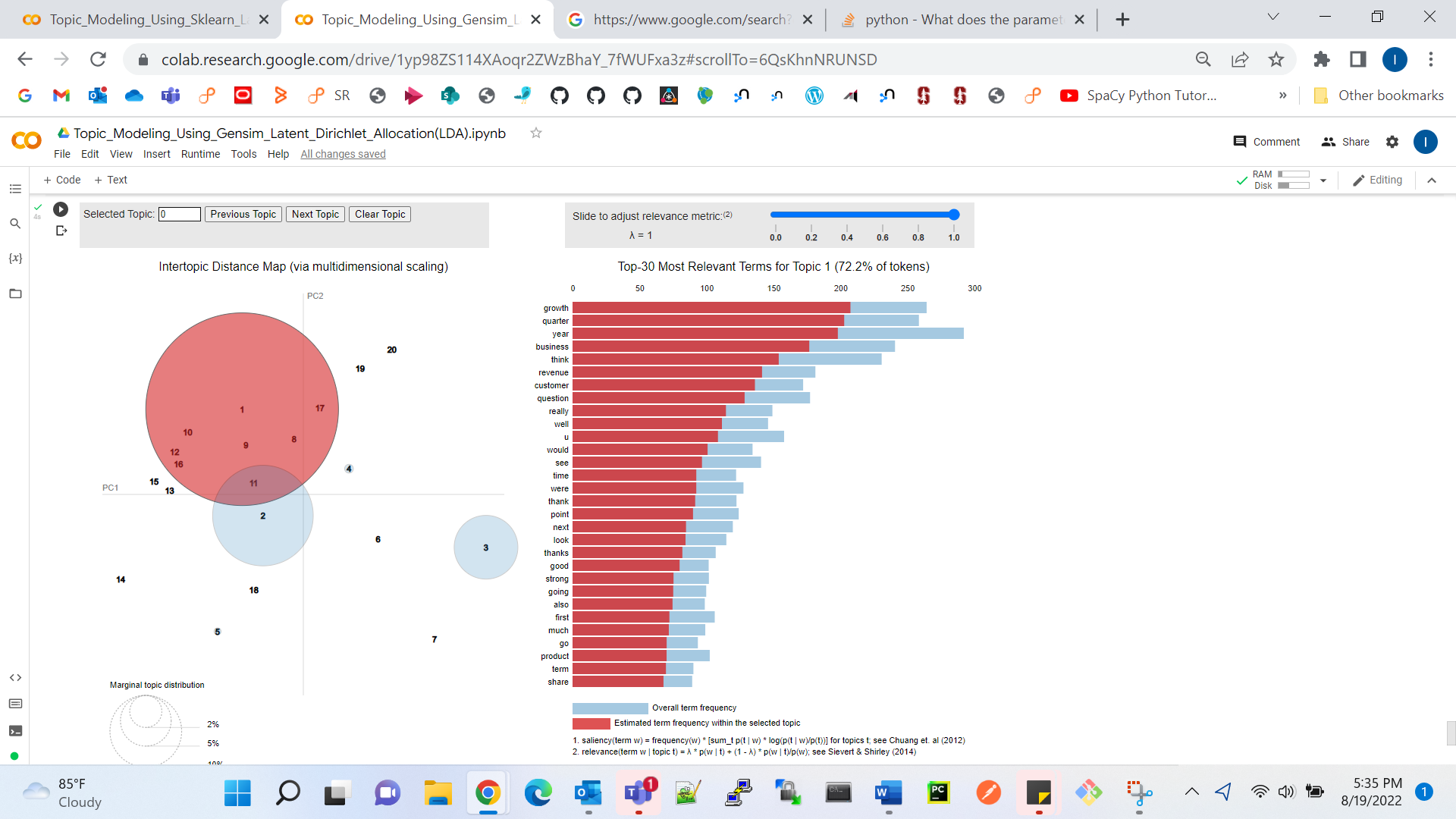




### **ii) Assigning the topics to the documents**



**iii)** **Using PyLDAvis for Visualization**



**References:**

* + 1. Docoh Dataset: <https://docoh.com/>
    2. BERTopic: <https://maartengr.github.io/BERTopic/index.html>
    3. <https://arxiv.org/pdf/2203.05794.pdf>
    4. <https://maartengr.github.io/BERTopic/getting_started/parameter%20tuning/parametertuning.html>
    5. <https://towardsdatascience.com/topic-modeling-with-bert-779f7db187e6>
    6. <https://towardsdatascience.com/interactive-topic-modeling-with-bertopic-1ea55e7d73d8>
    7. <https://maartengr.github.io/BERTopic/getting_started/visualization/visualization.html#visualize-topic-hierarchy>
    8. Algorithm: <https://maartengr.github.io/BERTopic/algorithm/algorithm.html>
    9. Dimensionality Reduction: <https://maartengr.github.io/BERTopic/getting_started/dim_reduction/dim_reduction.html>
    10. Clustering: <https://maartengr.github.io/BERTopic/getting_started/clustering/clustering.html>
    11. LDA: <https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2>
    12. <https://github.com/sethns/Latent-Dirichlet-Allocation-LDA-/blob/main/Topic%20Modeling%20_%20Extracting%20Topics_%20Using%20Genism.ipynb>
    13. <https://github.com/sethns/Latent-Dirichlet-Allocation-LDA-/blob/main/Topic%20Modeling%20_%20Extracting%20Topics_%20Using%20Sklearn.ipynb>
    14. UMAP: <https://umap-learn.readthedocs.io/en/latest/>
    15. CTM: <https://contextualized-topic-models.readthedocs.io/en/latest/introduction.html>
    16. <https://github.com/MilaNLProc/contextualized-topic-models/blob/master/docs/introduction.rst>