Topic Modeling for Divergent Datasets

Background of topic modeling:

Topic modeling a method to perform unsupervised classification of collected documents and those like clustering on numeric data, which finds natural groups of items even when we're not sure what we're looking for. In text mining, sources often have collections of various documents, such as blog posts/ news articles, that we would like to divide into different groups to understand them separately. It is also a probabilistic generative model which has been using widely in the field of data science which has a specific focus on text mining and information retrieval in latest years. From the time this model was initially proposed, it has been receiving a lot of attention and increased widespread interest in researchers in various research fields.

The beginning of a topic model is LSI - Latent Semantic Indexing, which served as the foundation for the development of a topic model. However, L S I was not a probabilistic model. Therefore, it is not considered as an authentic topic model. Based on L S I, PLSA - Probabilistic Latent Semantic Analysis, is proposed by scholar Hofmann and PLSA is considered as a genuine topic model. After PLSA, LDA - Latent Dirichlet Allocation which was proposed by Statistician Blei et al in 2003 is considered as even more comprehensive probabilistic generative model and is considered as the extension of Probabilistic Latent Semantic Analysis. Currently, there is increasing number of probabilistic models which are based on Latent Dirichlet Allocation through combination with specific tasks. However, all the above-mentioned topic models have firstly been introduced in the text analysis community for unsupervised topic discovery in a volume of documents.

Because of its dominance in analysis of large-scale document collections. In recent years, we have been witnessing exponential growth of data in every sector. These situations also position a countless challenge, namely how to extract hidden knowledge and relations from generated data and generated data sets. As mentioned above, topic models have emerged as an effective method for discovering useful structure in collections.

Therefore, a growing number of researchers are beginning to integrate topic models into various kinds of data and data sets, not only document collections. In these studies, we find that topic models act as more than a classification or clustering approach. Therefore, topic models were recently shown to be a powerful tool for every sector and study. Existing readings on topic modeling in diversified fields data are analyzed from distinct points of view, and then the glitches and predictions are discussed.

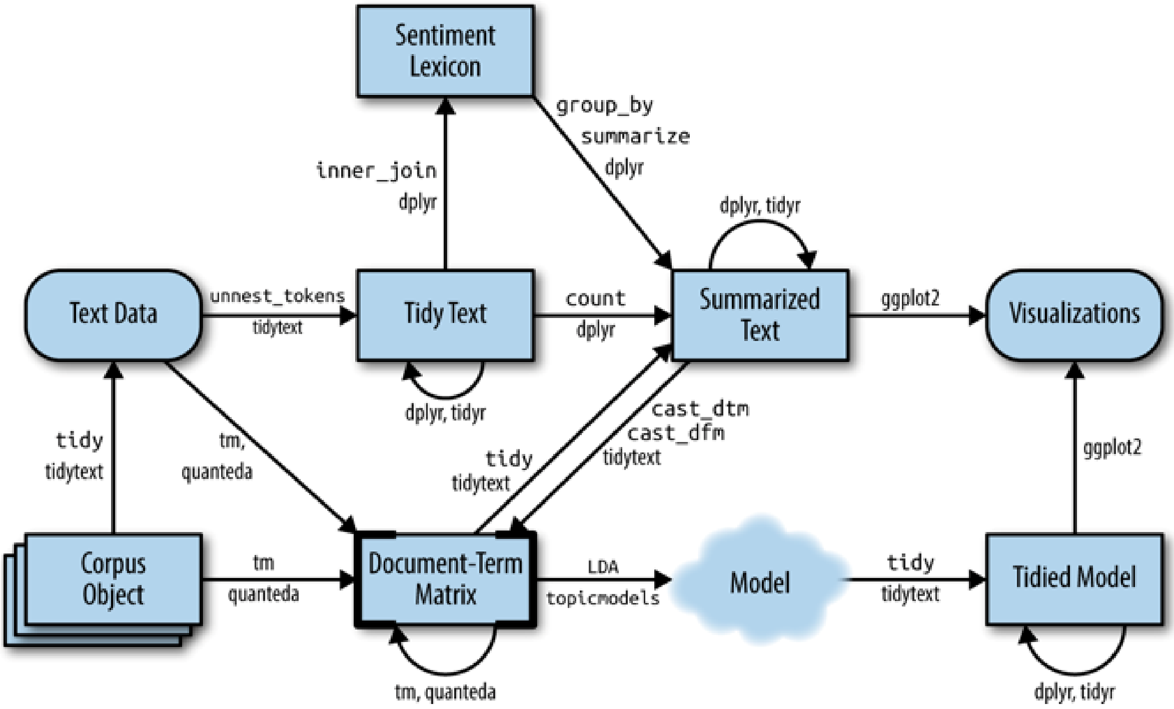


Figure 1: A flowchart of a text analysis that incorporates topic modeling.

Statement of the Problem

- Exploring Various topic modelling techniques
- Predicting the best model and the characteristic features

For the analysis we need to collect various types of data sets like short text, long text, social media, legal data, news data, numerical data. The data can be in any type of format. At first, we need to identify if it is a second-hand data or the firsthand data. Streaming data can be obtained by various open-source tools like twitter API integrating with Kafka or spark streaming. We can also use selenium to scrape the data. Then we need to do Data cleaning. This is the key phase on the data analysis. We can do exploratory data analysis and analyze the data set. Once after we are ready with the correct data, we can use various topic modelling techniques to predict which model works for which kind of Dataset.

Review of Literature

Topic modeling is a text mining technique that identifies patterns of word co-occurrence through a corpus of documents; these patterns are then thought of as secret "topics" that exist in the corpus. Topic modeling has been used to help with information retrieval, document classification, and exploratory text processing on broad corpora of texts.

Although computers may apply topic modeling algorithms to large corpora of text, a good topic model still requires a researcher to prepare the corpus for processing and make decisions about the algorithm's parameters, such as how many topics should be inferred and how certain topics should be distributed throughout the documents. This literature review answers these questions by first introducing the idea of topic modeling, then surveying a few studies that have used topic models, and finally looking at how those studies, as well as other related literature, explore the decisions that a topic model consumer must make.

Objective of Study

The project aims to find out abstract topics that occur in a collection of documents. We will be able to find out hidden sematic structures in a text body and compare the performance of various models on same type of datasets.

Data collection and Preprocessing:

Types of Data Collection:

- Web Scraping
- Secondhand data available in web

Web Scraping of collected data:

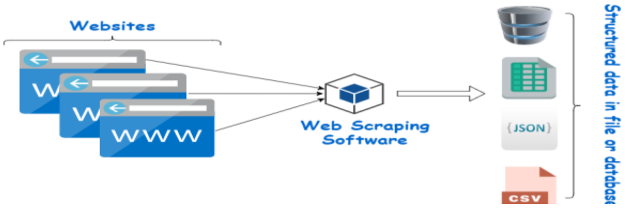


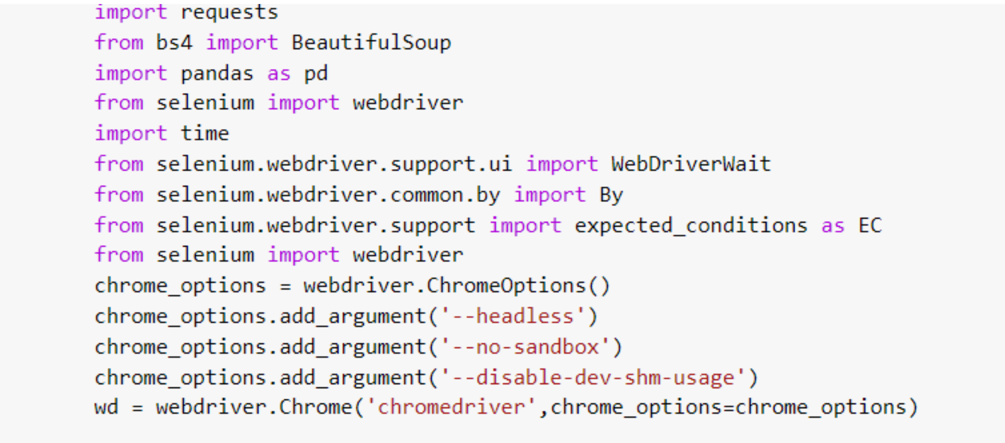
Figure 2

Data mining or gathering data is a very important step in data science life cycle. According to business requirements, anyone may require gathering data from different sources like Databases, APIs, logs, SAP servers, online repositories, or web. Tools for web scraping like Selenium has ability to scrape huge volume of data such as images and text in primitive time.

Web Scrapping is also called as “Spidering” or “Crawling” as the technique to get data automatically from any online source usually from any website. Web Scrapping is an easy way to get a huge volume of data in a very short time frame, it adds stress to the server where the source is hosted. This is also one of the main reasons why many websites donot allow scraping all on their individual website. However, as long as it does not disrupt the primary function of the online source, it is widely acceptable.

Web scraping can help data extract a huge amount of data about products, people, customers, stock markets etc. We can utilize the collected data from different websites such as Job portals, social media channels, e-commerce portal, to understand customer’s employee attrition behavior, customer’s sentiments, buying patterns, and the so on. Most famous libraries or frameworks that were used in Python for Web – Scrapping are Beautiful Soup, Scrappy & Selenium. In this project, we are using Web-scrapping using Selenium in Python.

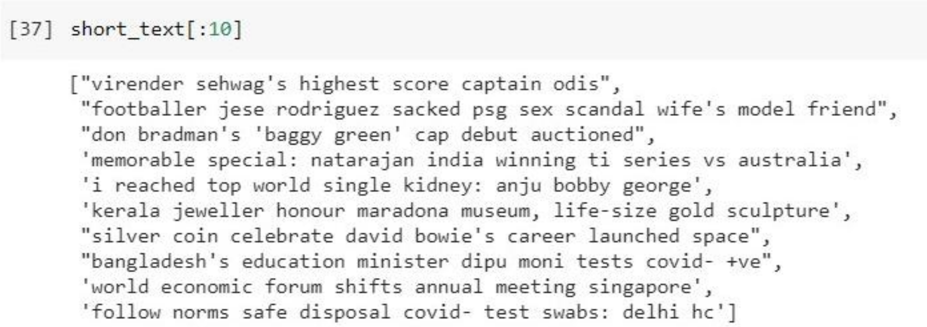
1. Short-text: Short-text is taken from headlines collected from short news website “In shorts”. We extracted the news headlines and context of News using web scraping by the libraries Beautiful soup and selenium, the site has a load more button which required clicking the button automatically if the button is detected while scraping, this is done using selenium. We first initialised selenium chrome driver with required version of chrome and then wrote the code such that the chrome driver triggers the button when the load more button is identified.

   ```
   import requests
   from bs4 import BeautifulSoup
   import pandas as pd
   from selenium import webdriver
   import time
   from selenium.webdriver.support.ui import WebDriverWait
   from selenium.webdriver.common.by import By
   from selenium.webdriver.support import expected_conditions as EC
   from selenium import webdriver
   chrome_options = webdriver.ChromeOptions()
   chrome_options.add_argument('--headless')
   chrome_options.add_argument('--no-sandbox')
   chrome_options.add_argument('--disable-dev-shm-usage')
   wd = webdriver.Chrome('chromedriver',chrome_options=chrome_options)
   ```

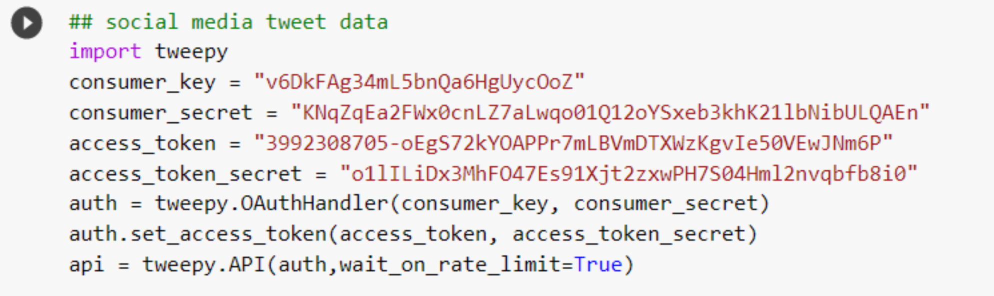
   In the above code, we can see that a chrome driver object is initialised.

   Short-text output:

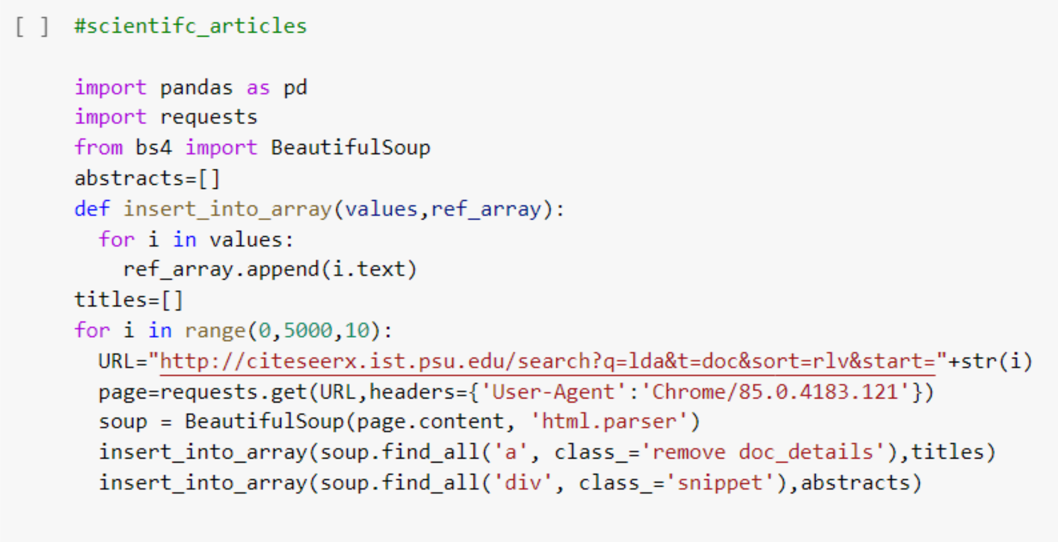
   ```
   [37] short_text[:10]

   ["virender sehwag's highest score captain odis",
    "footballer jese rodriguez sacked psg sex scandal wife's model friend",
    "don bradman's 'baggy green' cap debut auctioned",
    'memorable special: natarajan india winning ti series vs australia',
    'i reached top world single kidney: anju bobby george',
    'kerala jeweller honour maradona museum, life-size gold sculpture',
    "silver coin celebrate david bowie's career launched space",
    "bangladesh's education minister dipu moni tests covid- +ve",
    'world economic forum shifts annual meeting singapore',
    'follow norms safe disposal covid- test swabs: delhi hc']
   ```

2. Social media data: We scraped twitter using ‘Tweepy’ API. This library requires authentication with twitter developer console such that we request for the keys for the API in twitter developer console and after the permission from Twitter console, we use API keys to access data from Twitter. This API can scrape Time, User name and Tweet text from the Tweepy API object.



```
## social media tweet data
import tweepy
consumer_key = "v6DkFAg34mL5bnQa6HgUycOoZ"
consumer_secret = "KNqZqEa2FWx0cnLZ7aLwqo01Q12oYSxeb3khK21lbNibULQAEn"
access_token = "3992308705-oEgS72kYOAPPr7mLBVmDTXWzKgvIe50VEwJNm6P"
access_token_secret = "o1lILiDx3MhFO47Es91Xjt2zxwPH7S04Hml2nvqbfb8i0"
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth,wait_on_rate_limit=True)
```

3. Scientific journals: Scientific journals data is scrapped form site “citeseerx” data on the topic LDA. We scrapped the data using Beautifulsoup library. WE extracted data from citeseerx and used Abstracts in the data for out analysis.



```
[ ]  #scientifc_articles

     import pandas as pd
     import requests
     from bs4 import BeautifulSoup
     abstracts=[]
     def insert_into_array(values,ref_array):
       for i in values:
         ref_array.append(i.text)
     titles=[]
     for i in range(0,5000,10):
       URL="http://citeseerx.ist.psu.edu/search?q=lda&t=doc&sort=rlv&start="+str(i)
       page=requests.get(URL,headers={'User-Agent':'Chrome/85.0.4183.121'})
       soup = BeautifulSoup(page.content, 'html.parser')
       insert_into_array(soup.find_all('a', class_='remove doc_details'),titles)
       insert_into_array(soup.find_all('div', class_='snippet'),abstracts)
```

Web scrapping in all the above extraction using Beautiful Soup works by first extracting HTML code of the website, then parsing the HTML and then finding the required data using HTML tags and separating it.

Secondhand Data:

Figure 3: Second hand and primary data

Secondhand data set is readily available form various sources which are stored in databases. The data sources can be encyclopedia, Kaggle, any websites which collect data. The secondhand data set available in Kaggle are widely used for data analytics. Common sources of secondary data for social science include censuses, information collected by government departments, organizational records and data that was originally collected for other research purposes.

While secondary data is associated with quantitative databases, analysis focused on verbal or visual materials created for another purpose, is a legitimate avenue for the qualitative researcher. One could go as far as claim that qualitative secondary data analysis “can be understood, not so much as the analysis of pre-existing data; rather as involving a process of re-contextualizing, and re-constructing, data.

Secondary data can be obtained from different sources:

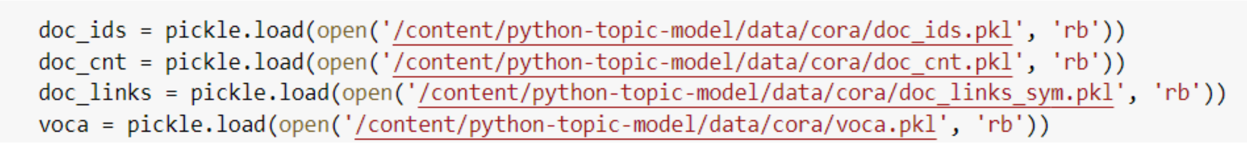
- Information, collected through censuses or government departments like housing, social security, electoral statistics, tax records
- internet searches or libraries
- GPS, remote sensing
- km progress reports

The data can be readily available in any format like csv, text, xls, json, pickle, html.

Some of the Datasets, we used in this project are:
https://www.kaggle.com/notlucasp/financial-news-headlines?select=guardian_headlines.csv
https://www.kaggle.com/tboyle10/medicaltranscriptions

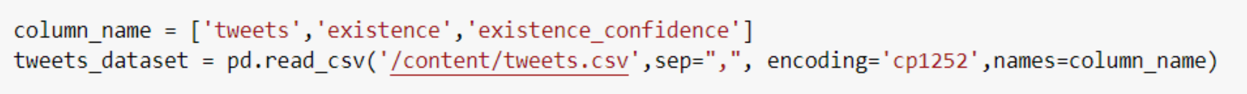
Code Snippets of Data Reading:

1) Read the Pickle file:



```
doc_ids = pickle.load(open('/content/python-topic-model/data/cora/doc_ids.pkl', 'rb'))
doc_cnt = pickle.load(open('/content/python-topic-model/data/cora/doc_cnt.pkl', 'rb'))
doc_links = pickle.load(open('/content/python-topic-model/data/cora/doc_links_sym.pkl', 'rb'))
voca = pickle.load(open('/content/python-topic-model/data/cora/voca.pkl', 'rb'))
```

2) Reading the csv files:



```
column_name = ['tweets','existence','existence_confidence']
tweets_dataset = pd.read_csv('/content/tweets.csv',sep=",", encoding='cp1252',names=column_name)
```

In addition to reading the data from the folders in local we can load the data set which resides in cloud like dropbox, google drive etc. We need to fetch the access tokens from the respective open-source API’ and use the access token to read the data set from the respective URL.

```
def __init__(self, id, params):
                super(Dropbox_Dropper, self).__init__(id, params)
                try:
                        self.access_token = params["access_token"]
                except KeyError as k: # if config parameters are missing
                        logging.error("Dropxbox: Error while trying to initialize notifier, it seems there
is a config parameter missing: %s" % k)
                        self.corrupted = True
                        return

                try:
                        self.dbx = dropbox.Dropbox(self.access_token)
                except Exception as e:
                        logging.error("Dropbox: Error while connecting to Dropbox service: %s" % e)
                        self.corrupted = True
                        return

                self.data_dir = "/var/tmp/secpi/alarms/" #change this maybe?

                logging.info("Dropbox initialized")
```

Data Cleaning:

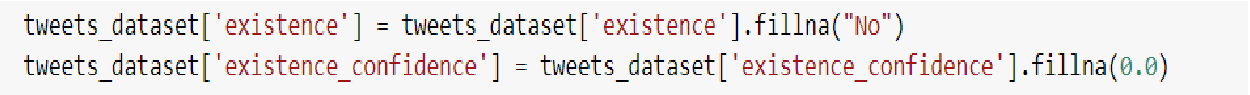
Figure 4

Data Cleaning plays a very crucial role in analysis. The data set which we collected is not in the format which is required for analysis. Data quality is a main issue in quality information management. Data quality problems occur anywhere in information systems. These problems are solved by data cleaning. Data cleaning is a process used to determine inaccurate, incomplete, or unreasonable data and then improve the quality through correcting of detected errors and omissions.

Generally, data cleaning reduces errors and improves the data quality. Correcting errors in data and eliminating bad records can be a time consuming and tedious process but it cannot be ignored. Data mining is a key technique for data cleaning. Data mining is a technique for discovery interesting information in data. Data quality mining is a recent approach applying data mining techniques to identify and recover data quality problems in large databases. Data mining automatically extract hidden and intrinsic information from the collections of data. Data mining has various techniques that are suitable for data cleaning. In this paper we discuss three major data mining methods, namely functional dependency mining, association rule mining and Bagging SVMs for data cleaning.

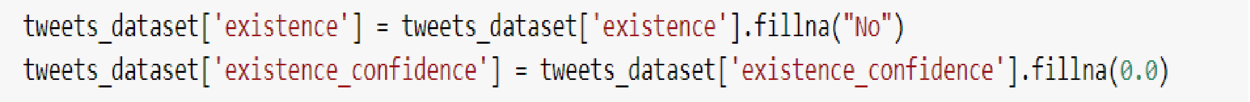
Steps involved in Data Cleaning:

- Replace null values



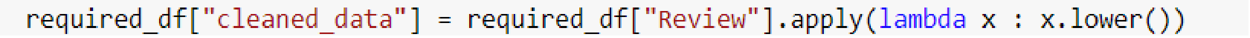
```
tweets_dataset['existence'] = tweets_dataset['existence'].fillna("No")
tweets_dataset['existence_confidence'] = tweets_dataset['existence_confidence'].fillna(0.0)
```

- Eliminate outliers
- Replace missing Values



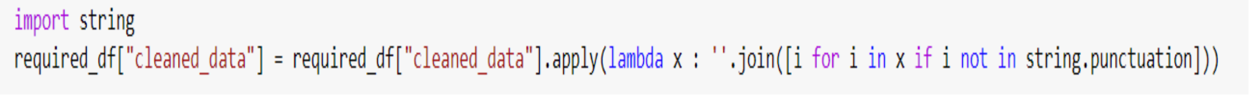
```
tweets_dataset['existence'] = tweets_dataset['existence'].fillna("No")
tweets_dataset['existence_confidence'] = tweets_dataset['existence_confidence'].fillna(0.0)
```

- Convert the text to lower case



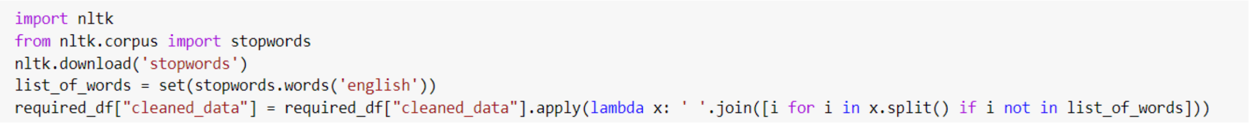
```
required_df["cleaned_data"] = required_df["Review"].apply(lambda x : x.lower())
```

- Remove the punctuations



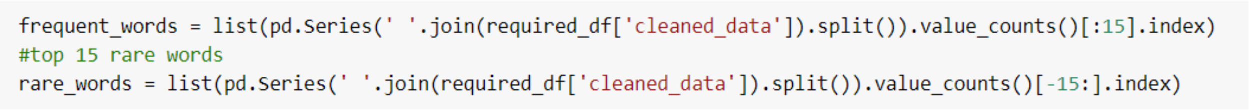
```
import string
required_df["cleaned_data"] = required_df["cleaned_data"].apply(lambda x : ''.join([i for i in x if i not in string.punctuation]))
```

- Eliminate the stop words

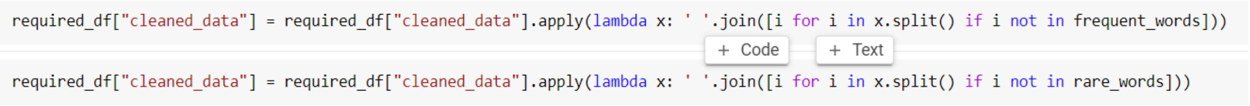


```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
list_of_words = set(stopwords.words('english'))
required_df["cleaned_data"] = required_df["cleaned_data"].apply(lambda x: ' '.join([i for i in x.split() if i not in list_of_words]))
```

- Remove the frequent and rare words in the data set



```
frequent_words = list(pd.Series(' '.join(required_df['cleaned_data']).split()).value_counts()[:15].index)
#top 15 rare words
rare_words = list(pd.Series(' '.join(required_df['cleaned_data']).split()).value_counts()[-15:].index)
```



```
required_df["cleaned_data"] = required_df["cleaned_data"].apply(lambda x: ' '.join([i for i in x.split() if i not in frequent_words]))
```

```
required_df["cleaned_data"] = required_df["cleaned_data"].apply(lambda x: ' '.join([i for i in x.split() if i not in rare_words]))
```

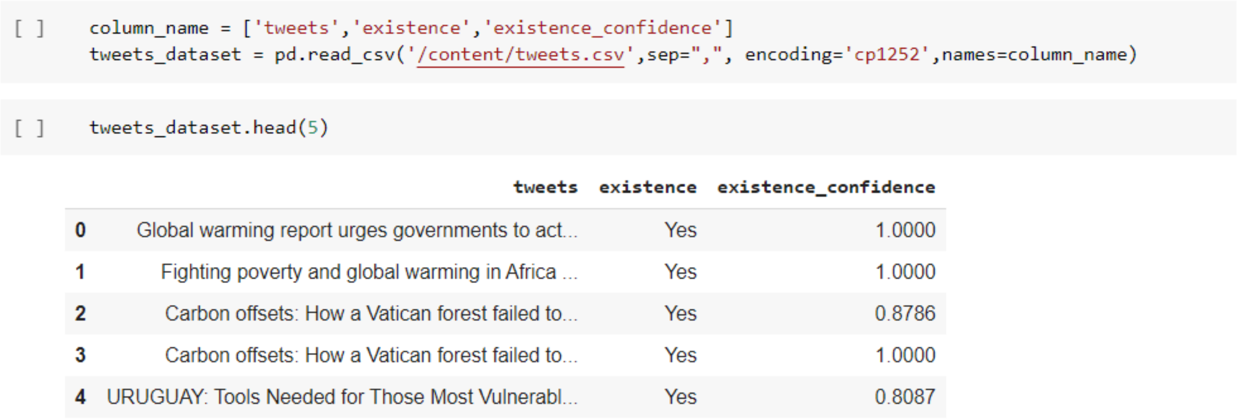
- There are times where we need to be stemming and lemmatization

Data Analysis Topic Modelling Techniques:

1. Latent Dirichlet Allocation (LDA)

Topic Modeling is a technique to extract the hidden topics from large volumes of text. Latent Dirichlet Allocation is one the popular and traditional algorithm for topic modeling. LDA's approach to topic modeling is it considers each document as a collection of topics in a certain proportion, it allows sets of observations to be explained in groups that contain similar data. LDA is being implemented from genism package. Algorithm basically runs by assuming a number of topics, and then it runs through the document and rearranges the topic distribution within the document.

Text Cleaning:

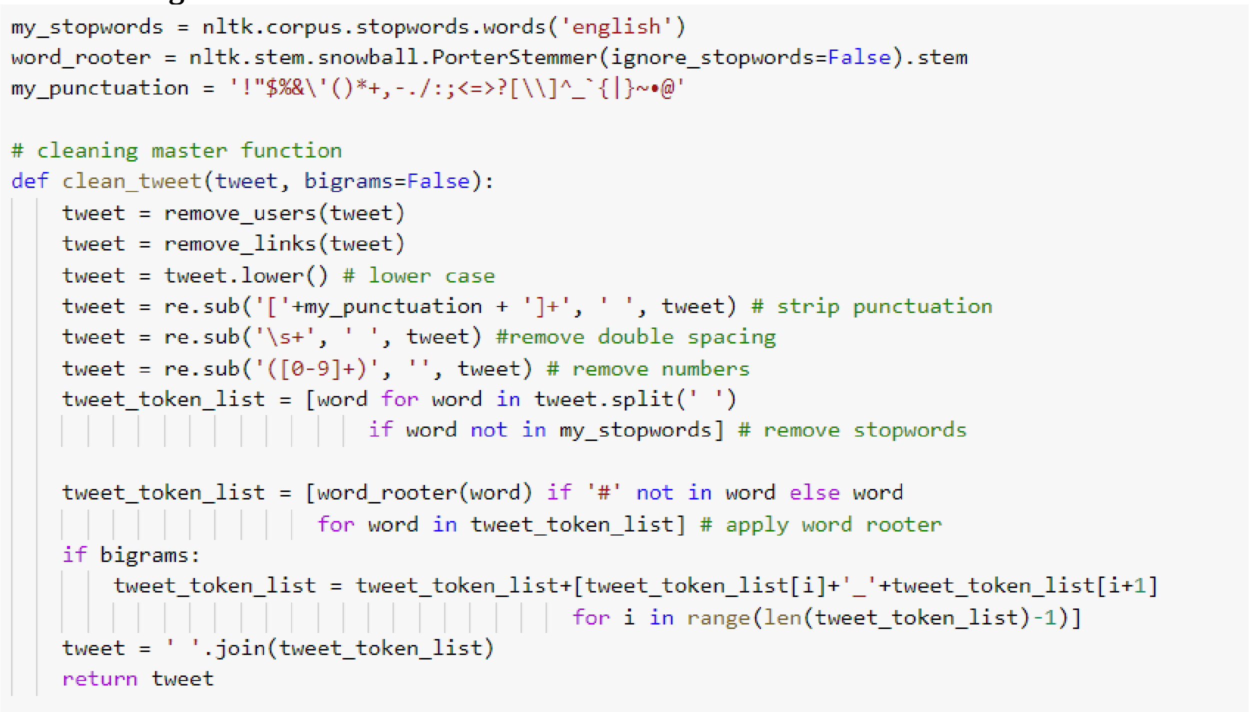


```
[ ]   column_name = ['tweets','existence','existence_confidence']
      tweets_dataset = pd.read_csv('/content/tweets.csv',sep=",", encoding='cp1252',names=column_name)
```

```
[ ]   tweets_dataset.head(5)
```

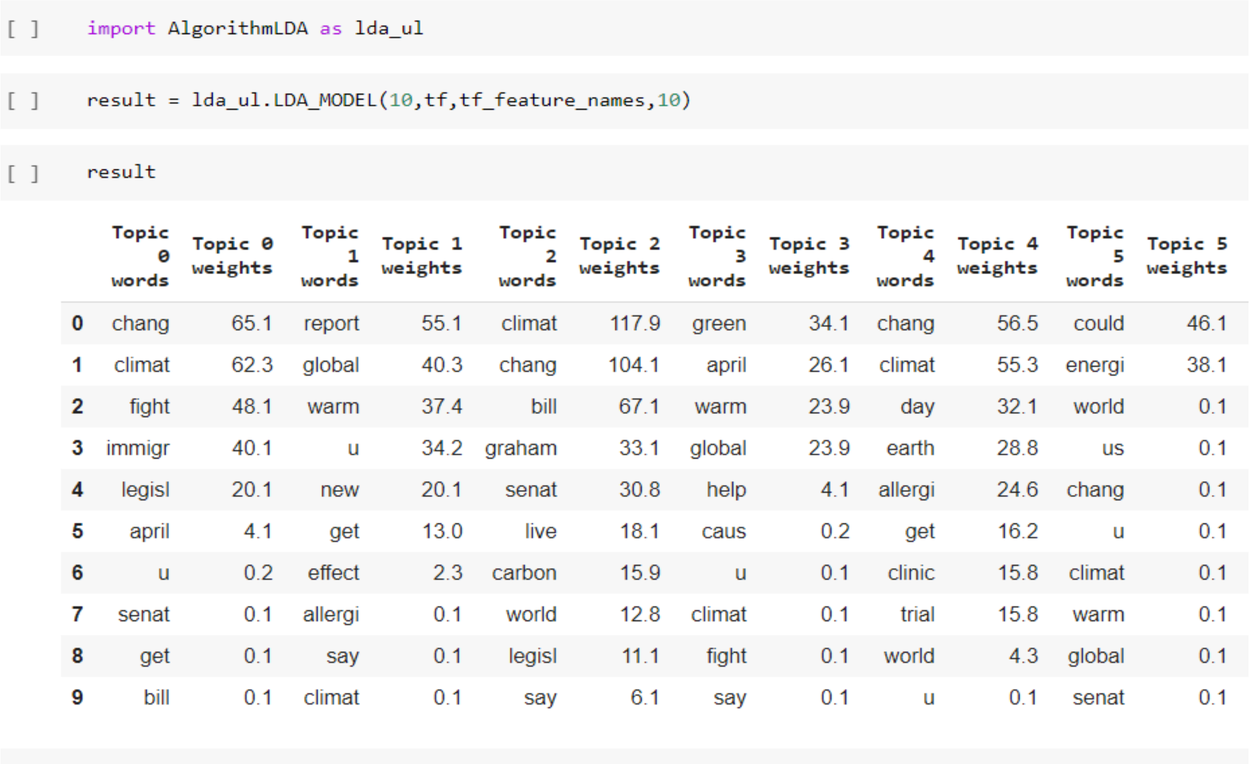
	tweets	existence	existence_confidence
0	Global warming report urges governments to act...	Yes	1.0000
1	Fighting poverty and global warming in Africa ...	Yes	1.0000
2	Carbon offsets: How a Vatican forest failed to...	Yes	0.8786
3	Carbon offsets: How a Vatican forest failed to...	Yes	1.0000
4	URUGUAY: Tools Needed for Those Most Vulnerabl...	Yes	0.8087

Text cleaning code:



```
my_stopwords = nltk.corpus.stopwords.words('english')
word_rooter = nltk.stem.snowball.PorterStemmer(ignore_stopwords=False).stem
my_punctuation = '!"$%&\'()*+,-./:;<=>?[\\]^_`{|}~•@'

# cleaning master function
def clean_tweet(tweet, bigrams=False):
    tweet = remove_users(tweet)
    tweet = remove_links(tweet)
    tweet = tweet.lower() # lower case
    tweet = re.sub('['+my_punctuation + ']+', ' ', tweet) # strip punctuation
    tweet = re.sub('\s+', ' ', tweet) #remove double spacing
    tweet = re.sub('([0-9]+)', '', tweet) # remove numbers
    tweet_token_list = [word for word in tweet.split(' ')
                            if word not in my_stopwords] # remove stopwords

    tweet_token_list = [word_rooter(word) if '#' not in word else word
                        for word in tweet_token_list] # apply word rooter
    if bigrams:
        tweet_token_list = tweet_token_list+[tweet_token_list[i]+'_'+tweet_token_list[i+1]
                                            for i in range(len(tweet_token_list)-1)]
    tweet = ' '.join(tweet_token_list)
    return tweet
```

Result after running the model:



```
[ ]  import AlgorithmLDA as lda_ul
```

```
[ ]  result = lda_ul.LDA_MODEL(10,tf,tf_feature_names,10)
```

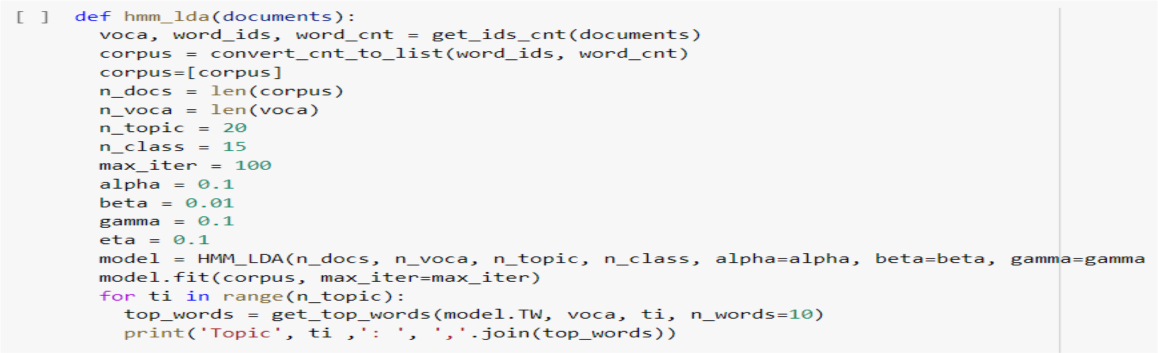
```
[ ]  result
```

	Topic 0 words	Topic 0 weights	Topic 1 words	Topic 1 weights	Topic 2 words	Topic 2 weights	Topic 3 words	Topic 3 weights	Topic 4 words	Topic 4 weights	Topic 5 words	Topic 5 weights
0	chang	65.1	report	55.1	climat	117.9	green	34.1	chang	56.5	could	46.1
1	climat	62.3	global	40.3	chang	104.1	april	26.1	climat	55.3	energi	38.1
2	fight	48.1	warm	37.4	bill	67.1	warm	23.9	day	32.1	world	0.1
3	immigr	40.1	u	34.2	graham	33.1	global	23.9	earth	28.8	us	0.1
4	legisl	20.1	new	20.1	senat	30.8	help	4.1	allergi	24.6	chang	0.1
5	april	4.1	get	13.0	live	18.1	caus	0.2	get	16.2	u	0.1
6	u	0.2	effect	2.3	carbon	15.9	u	0.1	clinic	15.8	climat	0.1
7	senat	0.1	allergi	0.1	world	12.8	climat	0.1	trial	15.8	warm	0.1
8	get	0.1	say	0.1	legisl	11.1	fight	0.1	world	4.3	global	0.1
9	bill	0.1	climat	0.1	say	6.1	say	0.1	u	0.1	senat	0.1

2.HMM-LDA (Hidden Markov model with Latent Dirichlet allocation):

Hidden Markov models are some probabilistic graphical models that are used to predict a sequence of hidden unknown variables from a set of observed variables. An LDA is a statistical model that allows observations to explained by unobserved groups, this explains similar parts of data.

An HMM-LDA model combines both LDA and HMM models to identify hidden syntactic words from topic dependent words without requiring any labelled data. Unobserved groups that are required by LDA is provided by HMM which is like an end-to-end system. As LDA is a generative model it is easy to combine with different models and words of the model are exhibited by product and mixture of the model. HMM-LDA takes words and makes them into cluster, for words distribution that vary between the documents.

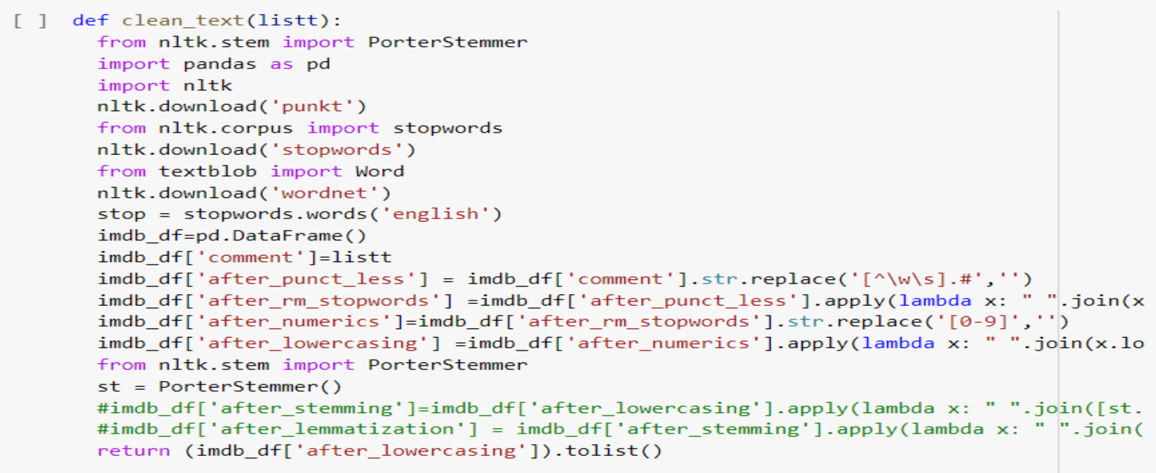


```
[ ]  def hmm_lda(documents):
       voca, word_ids, word_cnt = get_ids_cnt(documents)
       corpus = convert_cnt_to_list(word_ids, word_cnt)
       corpus=[corpus]
       n_docs = len(corpus)
       n_voca = len(voca)
       n_topic = 20
       n_class = 15
       max_iter = 100
       alpha = 0.1
       beta = 0.01
       gamma = 0.1
       eta = 0.1
       model = HMM_LDA(n_docs, n_voca, n_topic, n_class, alpha=alpha, beta=beta, gamma=gamma
       model.fit(corpus, max_iter=max_iter)
       for ti in range(n_topic):
         top_words = get_top_words(model.TW, voca, ti, n_words=10)
         print('Topic', ti ,': ', ','.join(top_words))
```

In the above code snippet, we can see the step by step process to implement the HMM-LDA model, first we will take the input documents and convert the documents into text corpus and vocabulary, then we will train these corpus and vocabular with the desired parameters of number of topics, classes and documents required are passed to the model.

The first step for any process is cleaning text, as HMM-LDA generates word embeddings without contextual representations, it is important to clean the text free form the punctuation, stop words and generating lemma etc.., to get the desired output.

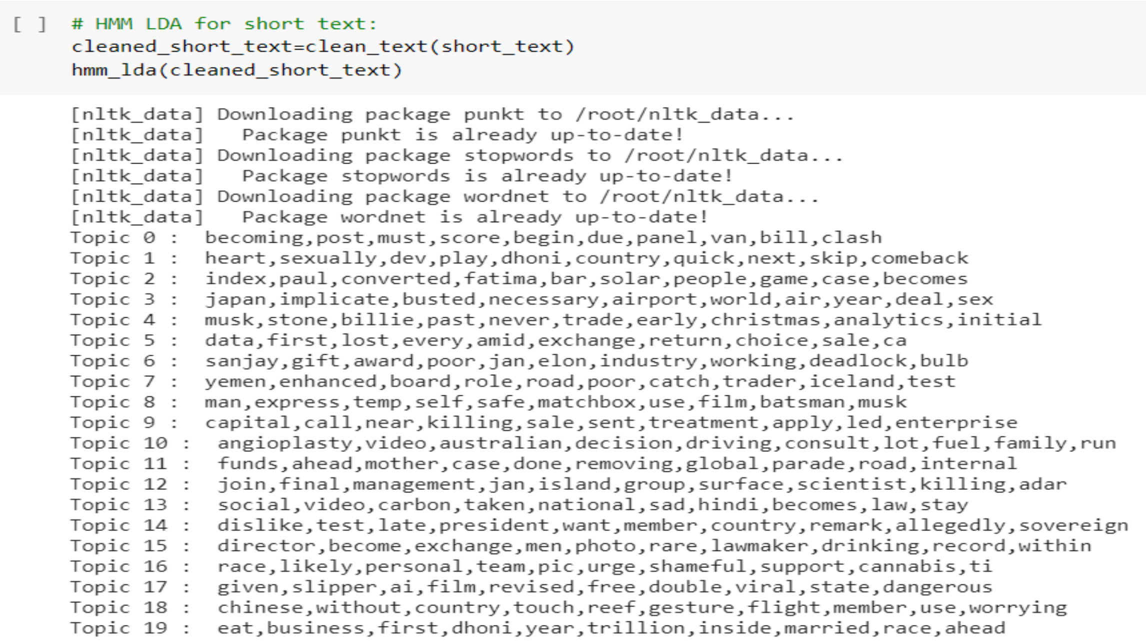
Text cleaning:



```
[ ]  def clean_text(listt):
       from nltk.stem import PorterStemmer
       import pandas as pd
       import nltk
       nltk.download('punkt')
       from nltk.corpus import stopwords
       nltk.download('stopwords')
       from textblob import Word
       nltk.download('wordnet')
       stop = stopwords.words('english')
       imdb_df=pd.DataFrame()
       imdb_df['comment']=listt
       imdb_df['after_punct_less'] = imdb_df['comment'].str.replace('[^\w\s].#','')
       imdb_df['after_rm_stopwords'] =imdb_df['after_punct_less'].apply(lambda x: " ".join(x
       imdb_df['after_numerics']=imdb_df['after_rm_stopwords'].str.replace('[0-9]','')
       imdb_df['after_lowercasing'] =imdb_df['after_numerics'].apply(lambda x: " ".join(x.lo
       from nltk.stem import PorterStemmer
       st = PorterStemmer()
       #imdb_df['after_stemming']=imdb_df['after_lowercasing'].apply(lambda x: " ".join([st.
       #imdb_df['after_lemmatization'] = imdb_df['after_stemming'].apply(lambda x: " ".join(
       return (imdb_df['after_lowercasing']).tolist()
```

<u>Topic modelling using HMM-LDA on different kinds of datasets:</u>

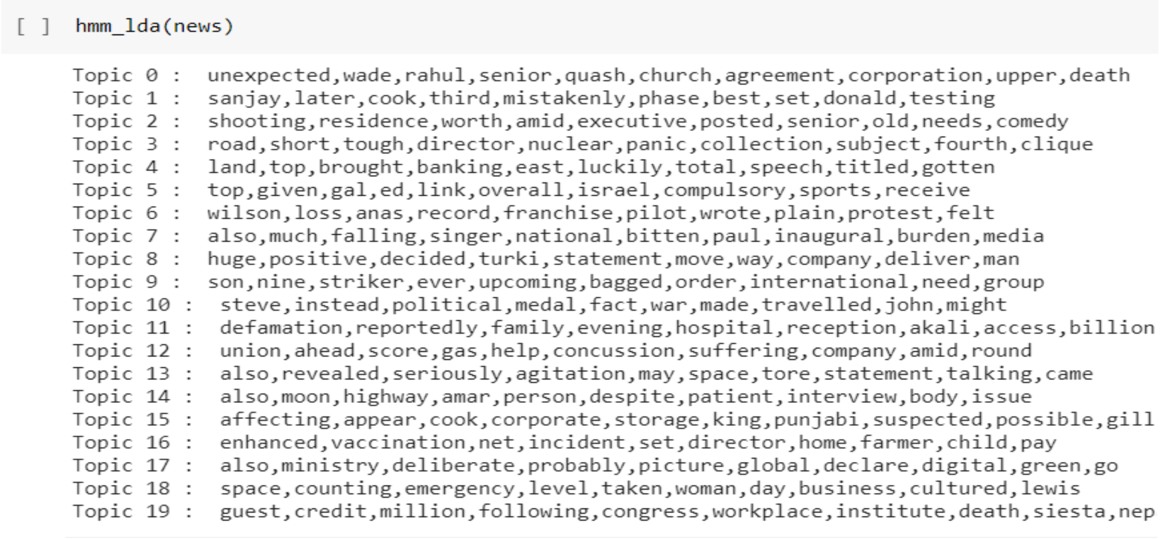
a) **<u>HMM-LDA on short text data:</u>** short text data is text whose length is not very long, limited to a few words or tokens.



```
[ ]  # HMM LDA for short text:
     cleaned_short_text=clean_text(short_text)
     hmm_lda(cleaned_short_text)

     [nltk_data] Downloading package punkt to /root/nltk_data...
     [nltk_data]   Package punkt is already up-to-date!
     [nltk_data] Downloading package stopwords to /root/nltk_data...
     [nltk_data]   Package stopwords is already up-to-date!
     [nltk_data] Downloading package wordnet to /root/nltk_data...
     [nltk_data]   Package wordnet is already up-to-date!
     Topic 0 :  becoming,post,must,score,begin,due,panel,van,bill,clash
     Topic 1 :  heart,sexually,dev,play,dhoni,country,quick,next,skip,comeback
     Topic 2 :  index,paul,converted,fatima,bar,solar,people,game,case,becomes
     Topic 3 :  japan,implicate,busted,necessary,airport,world,air,year,deal,sex
     Topic 4 :  musk,stone,billie,past,never,trade,early,christmas,analytics,initial
     Topic 5 :  data,first,lost,every,amid,exchange,return,choice,sale,ca
     Topic 6 :  sanjay,gift,award,poor,jan,elon,industry,working,deadlock,bulb
     Topic 7 :  yemen,enhanced,board,role,road,poor,catch,trader,iceland,test
     Topic 8 :  man,express,temp,self,safe,matchbox,use,film,batsman,musk
     Topic 9 :  capital,call,near,killing,sale,sent,treatment,apply,led,enterprise
     Topic 10 :  angioplasty,video,australian,decision,driving,consult,lot,fuel,family,run
     Topic 11 :  funds,ahead,mother,case,done,removing,global,parade,road,internal
     Topic 12 :  join,final,management,jan,island,group,surface,scientist,killing,adar
     Topic 13 :  social,video,carbon,taken,national,sad,hindi,becomes,law,stay
     Topic 14 :  dislike,test,late,president,want,member,country,remark,allegedly,sovereign
     Topic 15 :  director,become,exchange,men,photo,rare,lawmaker,drinking,record,within
     Topic 16 :  race,likely,personal,team,pic,urge,shameful,support,cannabis,ti
     Topic 17 :  given,slipper,ai,film,revised,free,double,viral,state,dangerous
     Topic 18 :  chinese,without,country,touch,reef,gesture,flight,member,use,worrying
     Topic 19 :  eat,business,first,dhoni,year,trillion,inside,married,race,ahead
```

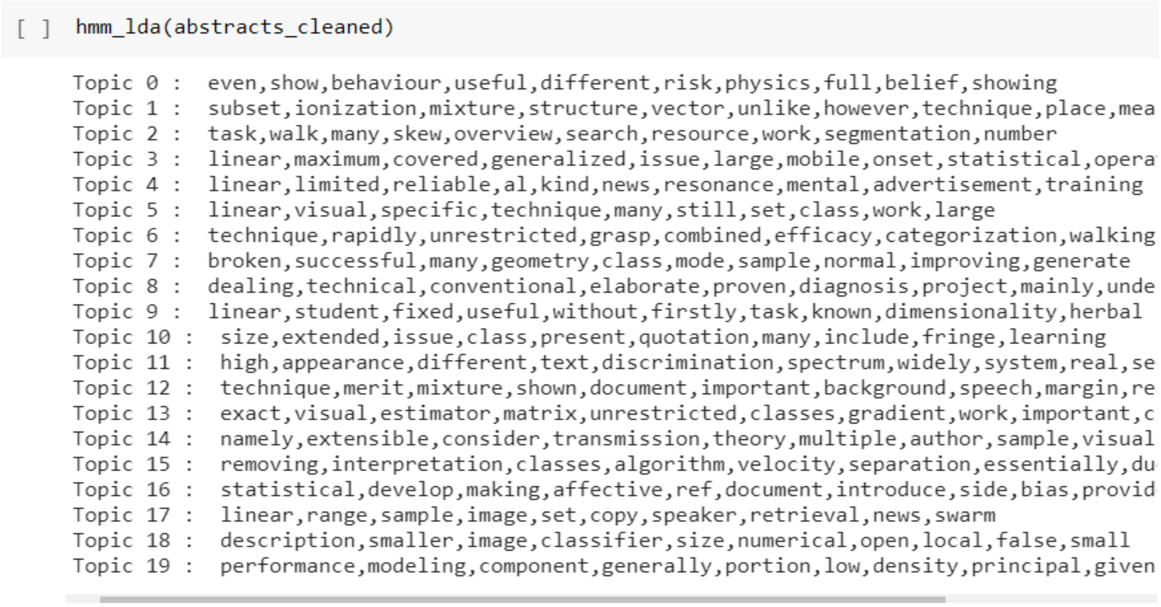
Mentioned above are the results of the topic modelling using HMM-LDA when applied on short text data, this data is of just 1000 documents and are short text.
In the TOPIC-1: Dhoni plays cricket in next comeback, makes some sense and in the remaining topics, words are not that similar with each other in the same topic. When compared to BERT it is poorly performing as it requires more amount of data.

<u>b). HMM-LDA on news data</u>: The news data is scrapped form “In-shorts” website which news not exceeding 60 words, it is longer than short text data.



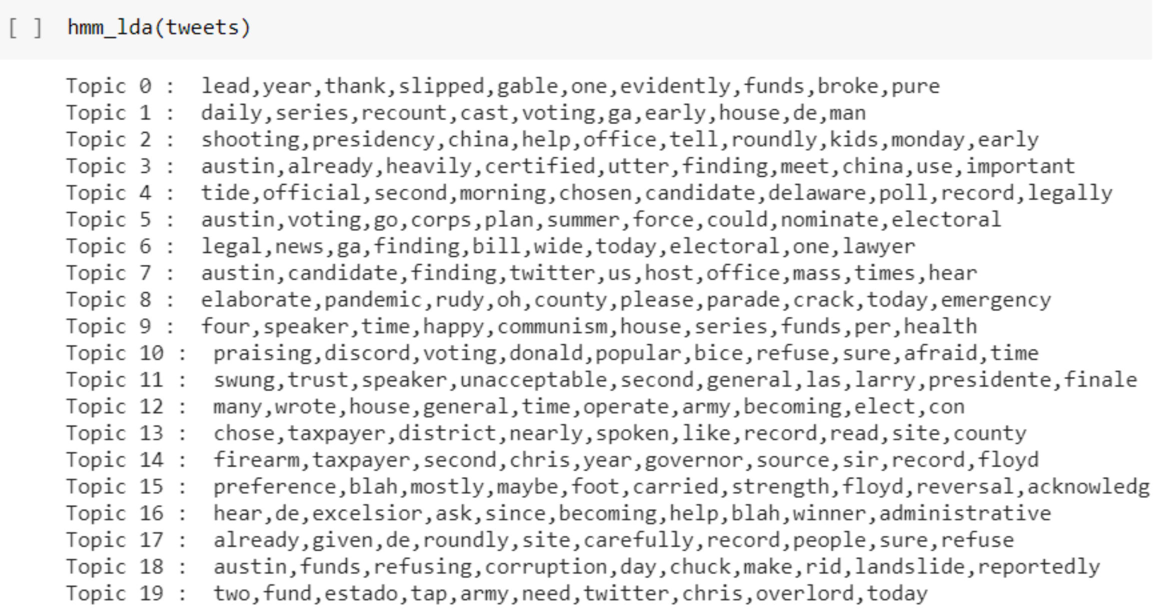
```
[ ]  hmm_lda(news)

     Topic 0 :  unexpected,wade,rahul,senior,quash,church,agreement,corporation,upper,death
     Topic 1 :  sanjay,later,cook,third,mistakenly,phase,best,set,donald,testing
     Topic 2 :  shooting,residence,worth,amid,executive,posted,senior,old,needs,comedy
     Topic 3 :  road,short,tough,director,nuclear,panic,collection,subject,fourth,clique
     Topic 4 :  land,top,brought,banking,east,luckily,total,speech,titled,gotten
     Topic 5 :  top,given,gal,ed,link,overall,israel,compulsory,sports,receive
     Topic 6 :  wilson,loss,anas,record,franchise,pilot,wrote,plain,protest,felt
     Topic 7 :  also,much,falling,singer,national,bitten,paul,inaugural,burden,media
     Topic 8 :  huge,positive,decided,turki,statement,move,way,company,deliver,man
     Topic 9 :  son,nine,striker,ever,upcoming,bagged,order,international,need,group
     Topic 10 :  steve,instead,political,medal,fact,war,made,travelled,john,might
     Topic 11 :  defamation,reportedly,family,evening,hospital,reception,akali,access,billion
     Topic 12 :  union,ahead,score,gas,help,concussion,suffering,company,amid,round
     Topic 13 :  also,revealed,seriously,agitation,may,space,tore,statement,talking,came
     Topic 14 :  also,moon,highway,amar,person,despite,patient,interview,body,issue
     Topic 15 :  affecting,appear,cook,corporate,storage,king,punjabi,suspected,possible,gill
     Topic 16 :  enhanced,vaccination,net,incident,set,director,home,farmer,child,pay
     Topic 17 :  also,ministry,deliberate,probably,picture,global,declare,digital,green,go
     Topic 18 :  space,counting,emergency,level,taken,woman,day,business,cultured,lewis
     Topic 19 :  guest,credit,million,following,congress,workplace,institute,death,siesta,nep
```

<u>c). HMM-LDA on scientific journals data</u>: The data for this scrapped from scientific journals website ‘citeseerx’ which has a collection of journals, we scrapped all the abstracts and titles on the topic LDA.



```
[ ]  hmm_lda(abstracts_cleaned)

     Topic 0 :  even,show,behaviour,useful,different,risk,physics,full,belief,showing
     Topic 1 :  subset,ionization,mixture,structure,vector,unlike,however,technique,place,mea
     Topic 2 :  task,walk,many,skew,overview,search,resource,work,segmentation,number
     Topic 3 :  linear,maximum,covered,generalized,issue,large,mobile,onset,statistical,opera
     Topic 4 :  linear,limited,reliable,al,kind,news,resonance,mental,advertisement,training
     Topic 5 :  linear,visual,specific,technique,many,still,set,class,work,large
     Topic 6 :  technique,rapidly,unrestricted,grasp,combined,efficacy,categorization,walking
     Topic 7 :  broken,successful,many,geometry,class,mode,sample,normal,improving,generate
     Topic 8 :  dealing,technical,conventional,elaborate,proven,diagnosis,project,mainly,unde
     Topic 9 :  linear,student,fixed,useful,without,firstly,task,known,dimensionality,herbal
     Topic 10 :  size,extended,issue,class,present,quotation,many,include,fringe,learning
     Topic 11 :  high,appearance,different,text,discrimination,spectrum,widely,system,real,se
     Topic 12 :  technique,merit,mixture,shown,document,important,background,speech,margin,re
     Topic 13 :  exact,visual,estimator,matrix,unrestricted,classes,gradient,work,important,c
     Topic 14 :  namely,extensible,consider,transmission,theory,multiple,author,sample,visual
     Topic 15 :  removing,interpretation,classes,algorithm,velocity,separation,essentially,du
     Topic 16 :  statistical,develop,making,affective,ref,document,introduce,side,bias,provid
     Topic 17 :  linear,range,sample,image,set,copy,speaker,retrieval,news,swarm
     Topic 18 :  description,smaller,image,classifier,size,numerical,open,local,false,small
     Topic 19 :  performance,modeling,component,generally,portion,low,density,principal,given
```

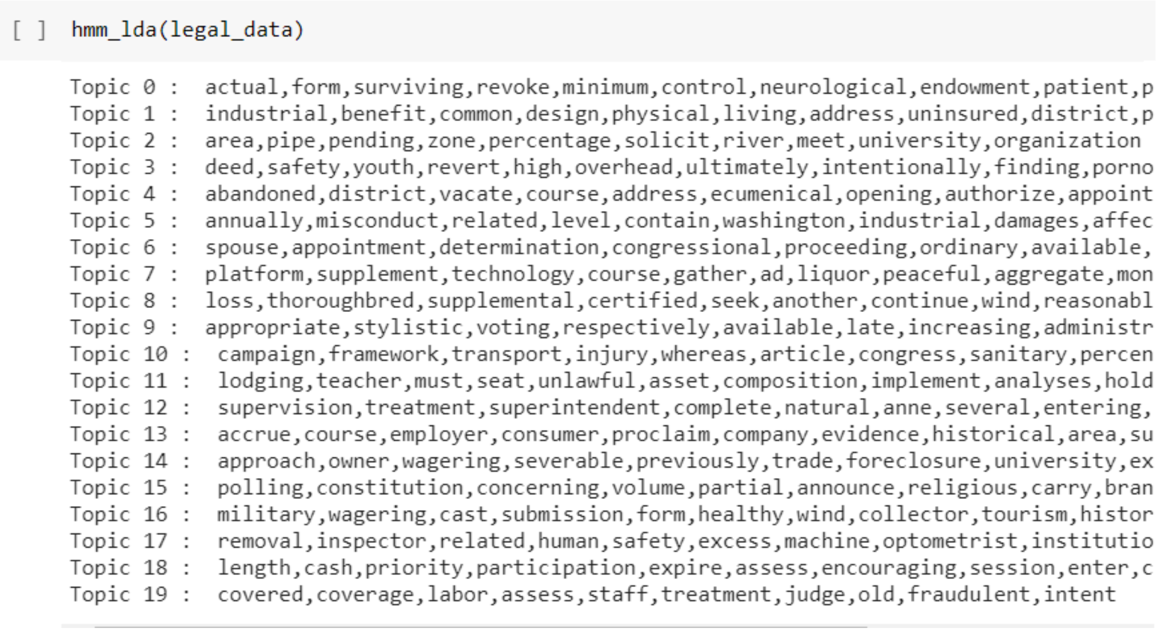
<u>d). HMM-LDA on social media data</u>: This is Twitter data extracted from the twitter using tweepy API, this API requires authentication to Twitter developer console, the data for this is collected based on the keyword search 'Joe Biden', this scrapes all the data in twitter matching with the given keyword.



```
[ ]  hmm_lda(tweets)

     Topic 0 :  lead,year,thank,slipped,gable,one,evidently,funds,broke,pure
     Topic 1 :  daily,series,recount,cast,voting,ga,early,house,de,man
     Topic 2 :  shooting,presidency,china,help,office,tell,roundly,kids,monday,early
     Topic 3 :  austin,already,heavily,certified,utter,finding,meet,china,use,important
     Topic 4 :  tide,official,second,morning,chosen,candidate,delaware,poll,record,legally
     Topic 5 :  austin,voting,go,corps,plan,summer,force,could,nominate,electoral
     Topic 6 :  legal,news,ga,finding,bill,wide,today,electoral,one,lawyer
     Topic 7 :  austin,candidate,finding,twitter,us,host,office,mass,times,hear
     Topic 8 :  elaborate,pandemic,rudy,oh,county,please,parade,crack,today,emergency
     Topic 9 :  four,speaker,time,happy,communism,house,series,funds,per,health
     Topic 10 :  praising,discord,voting,donald,popular,bice,refuse,sure,afraid,time
     Topic 11 :  swung,trust,speaker,unacceptable,second,general,las,larry,presidente,finale
     Topic 12 :  many,wrote,house,general,time,operate,army,becoming,elect,con
     Topic 13 :  chose,taxpayer,district,nearly,spoken,like,record,read,site,county
     Topic 14 :  firearm,taxpayer,second,chris,year,governor,source,sir,record,floyd
     Topic 15 :  preference,blah,mostly,maybe,foot,carried,strength,floyd,reversal,acknowledg
     Topic 16 :  hear,de,excelsior,ask,since,becoming,help,blah,winner,administrative
     Topic 17 :  already,given,de,roundly,site,carefully,record,people,sure,refuse
     Topic 18 :  austin,funds,refusing,corruption,day,chuck,make,rid,landslide,reportedly
     Topic 19 :  two,fund,estado,tap,army,need,twitter,chris,overlord,today
```

On social media text HMM-LDA seems to work well, the outputs is good, all the words in the Topic are similar.

<u>e). HMM-LDA on legal data</u>: This data is scrapped from the website Maryland legislative assembling on the bills passed, this website has text in the legal context, we aim to see the performance of HMM-LDA on the legal text which are not so common.



```
[ ]  hmm_lda(legal_data)

     Topic 0 :  actual,form,surviving,revoke,minimum,control,neurological,endowment,patient,p
     Topic 1 :  industrial,benefit,common,design,physical,living,address,uninsured,district,p
     Topic 2 :  area,pipe,pending,zone,percentage,solicit,river,meet,university,organization
     Topic 3 :  deed,safety,youth,revert,high,overhead,ultimately,intentionally,finding,porno
     Topic 4 :  abandoned,district,vacate,course,address,ecumenical,opening,authorize,appoint
     Topic 5 :  annually,misconduct,related,level,contain,washington,industrial,damages,affec
     Topic 6 :  spouse,appointment,determination,congressional,proceeding,ordinary,available,
     Topic 7 :  platform,supplement,technology,course,gather,ad,liquor,peaceful,aggregate,mon
     Topic 8 :  loss,thoroughbred,supplemental,certified,seek,another,continue,wind,reasonabl
     Topic 9 :  appropriate,stylistic,voting,respectively,available,late,increasing,administr
     Topic 10 :  campaign,framework,transport,injury,whereas,article,congress,sanitary,percen
     Topic 11 :  lodging,teacher,must,seat,unlawful,asset,composition,implement,analyses,hold
     Topic 12 :  supervision,treatment,superintendent,complete,natural,anne,several,entering,
     Topic 13 :  accrue,course,employer,consumer,proclaim,company,evidence,historical,area,su
     Topic 14 :  approach,owner,wagering,severable,previously,trade,foreclosure,university,ex
     Topic 15 :  polling,constitution,concerning,volume,partial,announce,religious,carry,bran
     Topic 16 :  military,wagering,cast,submission,form,healthy,wind,collector,tourism,histor
     Topic 17 :  removal,inspector,related,human,safety,excess,machine,optometrist,institutio
     Topic 18 :  length,cash,priority,participation,expire,assess,encouraging,session,enter,c
     Topic 19 :  covered,coverage,labor,assess,staff,treatment,judge,old,fraudulent,intent
```

All the above tests are performed on datasets with rows not greater than 1000 rows or documents, may be this makes HMM-LDA performance not good compared to BERT which is trained on a huge corpus of data. This proves that HMM-LDA should be performed on large amounts of data.

3.BERT (Bidirectional Encoder Representations from Transformers):

BERT is a brainchild of google which is based on application of bidirectional training of Transformers to language modelling. This is based on a belief that the language model which is bidirectionally trained that if from left to right and right to left will have a deeper sense of language context than single direction language models. BERT is trained on a neural network model called as transformer by giving data from Wikipedia (2500 million words) and book corpus (800 million) which is enormous thus the accuracy. It makes use of Transformer, a mechanism that learns contextual relations between words in a text. Bert uses masked language modelling which makes 15% of words in each sequence replaced with a mask token, it predicts original values of the mask token (classification layer) by multiplying the output vectors by the embedding matrix, transforming them into vocabulary dimension and calculating the probability of each word in vocabulary SoftMax. Bert is used in classification tasks, Question answering tasks and named entity recognition tasks.

In this project we used different datasets of different categories like of short text, long text, scientific journals, social media, legal documents dataset, we are going to perform topic modeling using BERT on this dataset to evaluate BERT performance on different kinds of data.

Datasets used in this analysis are extracted from:

1. Short-text: Inshorts(A news aggregator website,'https://inshorts.com/en/read')
2. Long-text: Inshorts
3. Scientific journal dataset: citeseerX(Took scientific journals on LDA.
4. Social media data: Twitter (Tweets having keyword 'JoeBiden')
5. Legal documents data: Maryland general assembly website

Steps to perform Topic modeling using BERT:

1. Data processing: Before we apply word embeddings, we clean the data, which is removing punctuations, lower casing, removing stop words, stemming etc.., and convert the raw data in to individual documents which is a list containing individual paragraphs.

2. Word Embeddings: Word Embeddings means to convert the documents to numerical data to make analysis easy and to yield better performance in various tasks performed in NLP. Here we used sentence-transformers package to generate embeddings from Text data. We here use Distilibert algorithm, which is a distilled version of BERT, this is known for its

balance between speed and performance and is also lighter and cheaper. This package even supports multi-lingual models.

3. Dimensionality reduction: we need to perform clustering to cluster all the topics that are similar, before performing clustering, we need to lower the dimensions of data as many algorithms handle high dimensionality very poorly. We here use UMAP algorithm (Uniform manifold approximation and projection) for dimensionality reduction which is the best in the space as it keeps a portion of the high dimensional local structure in lower dimensionality and works for non-linear data.

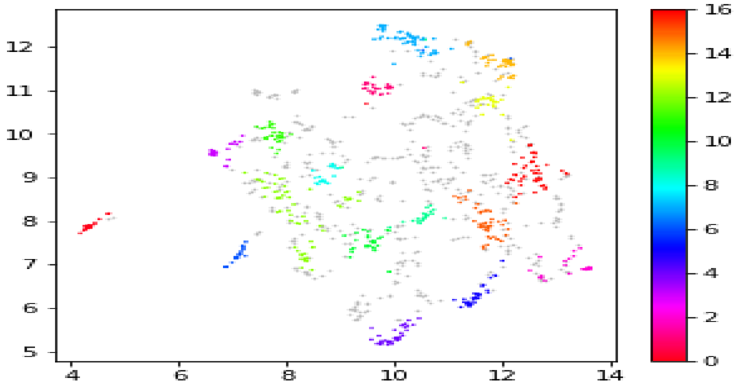
4. Clustering: After performing dimensionality reduction, we perform clustering to cluster the documents using hdbscan which is a density-based algorithm that works well with UMAP. DBSCAN is a spatial clustering of application with noise algorithm which clusters data by density.

5. Topic creation: To generate the topics from the clustered documents, we apply TF-IDF algorithm which generates term frequencies which are frequencies generated for each word 't' extracted from class 'I' and divided by total number of words 'w'. Here instead of applying this TF-IDF to every single document we apply to each cluster of documents considering that cluster as single document. The resultant topic gives the important words in the topic per each cluster.

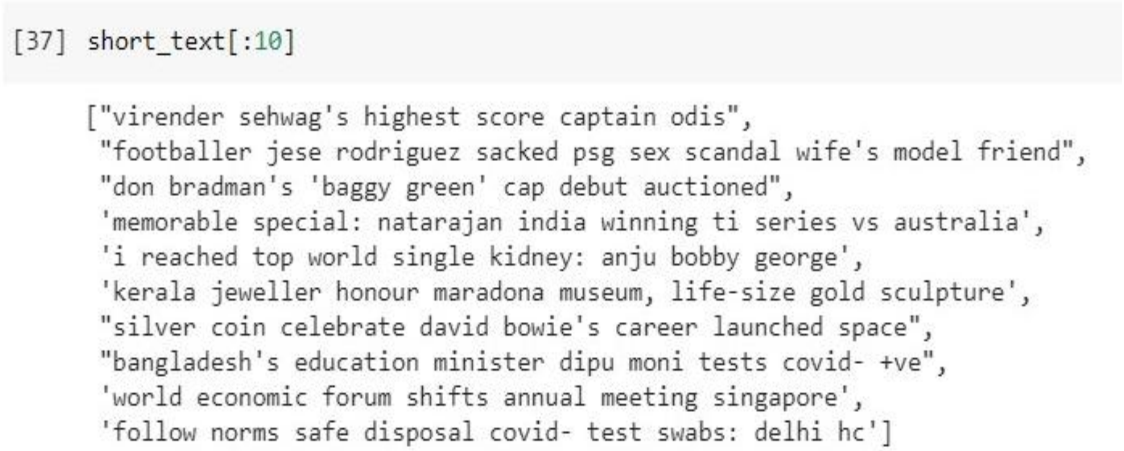
6. Topic representation: As there are many topics generated and many words generated per topic in cluster, we sort the topic and words and display the top important words and topics based on TF-IDF scores.

If there are many topic generated we tweak the parameters of HDBSCAN such that we will get few clusters by its minimum cluster size.

<u>a) BERT on short data:</u>

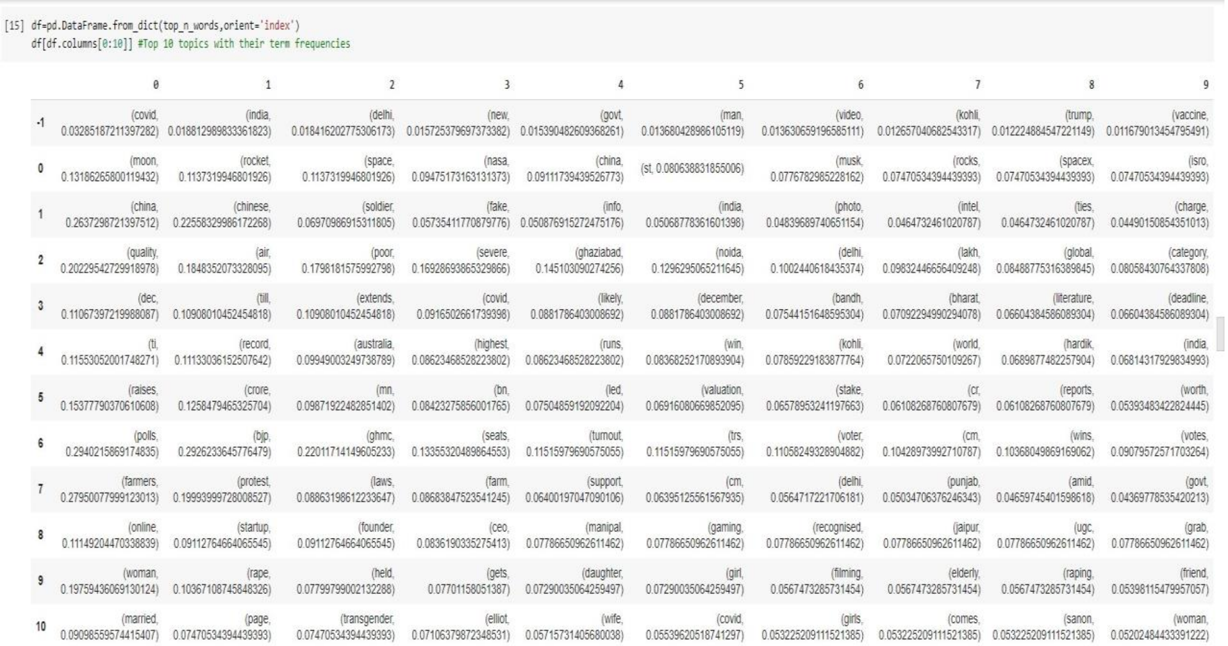


This is the sample documents given for the short text analysis:



```
[37] short_text[:10]

     ["virender sehwag's highest score captain odis",
      "footballer jese rodriguez sacked psg sex scandal wife's model friend",
      "don bradman's 'baggy green' cap debut auctioned",
      'memorable special: natarajan india winning ti series vs australia',
      'i reached top world single kidney: anju bobby george',
      'kerala jeweller honour maradona museum, life-size gold sculpture',
      "silver coin celebrate david bowie's career launched space",
      "bangladesh's education minister dipu moni tests covid- +ve",
      'world economic forum shifts annual meeting singapore',
      'follow norms safe disposal covid- test swabs: delhi hc']
```

After performing BERT Topic modeling:



```
[15] df=pd.DataFrame.from_dict(top_n_words,orient='index')
     df[df.columns[0:10]] #Top 10 topics with their term frequencies
```

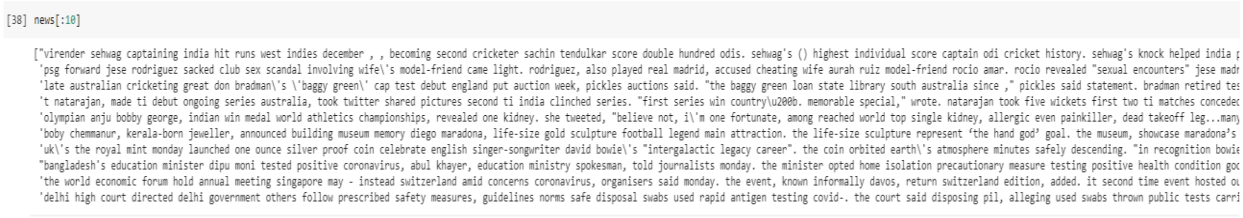
	0	1	2	3	4	5	6	7	8	9
-1	(covid, 0.03285187211397282)	(india, 0.018812989833361823)	(delhi, 0.018416202775306173)	(new, 0.015725379697373382)	(govt, 0.015390482609368261)	(man, 0.013680428986105119)	(video, 0.013630659196585111)	(kohli, 0.012657040682543317)	(trump, 0.012224884547221149)	(vaccine, 0.011679013454795491)
0	(moon, 0.13186265800119432)	(rocket, 0.1137319946801926)	(space, 0.1137319946801926)	(nasa, 0.09475173163131373)	(china, 0.09111739439526773)	(st, 0.080638831855006)	(musk, 0.0776782985228162)	(rocks, 0.07470534394439393)	(spacex, 0.07470534394439393)	(isro, 0.07470534394439393)
1	(china, 0.2637298721397512)	(chinese, 0.22558329986172268)	(soldier, 0.06970986915311805)	(fake, 0.05735411770879776)	(info, 0.050876915272475176)	(india, 0.05068778361601398)	(photo, 0.04839689740651154)	(intel, 0.0464732461020787)	(ties, 0.0464732461020787)	(charge, 0.04490150854351013)
2	(quality, 0.20229542729918978)	(air, 0.1848352073328095)	(poor, 0.1798181575992798)	(severe, 0.16928693865329866)	(ghaziabad, 0.145103090274256)	(noida, 0.1296295065211645)	(delhi, 0.1002440618435374)	(lakh, 0.09832446656409248)	(global, 0.08488775316389845)	(category, 0.08058430764337808)
3	(dec, 0.11067397219988087)	(till, 0.10908010452454818)	(extends, 0.10908010452454818)	(covid, 0.0916502661739398)	(likely, 0.0881786403008692)	(december, 0.0881786403008692)	(bandh, 0.07544151648595304)	(bharat, 0.07092294990294078)	(literature, 0.06604384586089304)	(deadline, 0.06604384586089304)
4	(ti, 0.11553052001748271)	(record, 0.11133036152507642)	(australia, 0.09949003249738789)	(highest, 0.08623468528223802)	(runs, 0.08623468528223802)	(win, 0.08368252170893904)	(kohli, 0.07859229183877764)	(world, 0.0722065750109267)	(hardik, 0.0689877482257904)	(india, 0.06814317929834993)
5	(raises, 0.15377790370610608)	(crore, 0.1258479465325704)	(mn, 0.09871922482851402)	(bn, 0.08423275856001765)	(led, 0.07504859192092204)	(valuation, 0.06916080669852095)	(stake, 0.06578953241197663)	(cr, 0.06108268760807679)	(reports, 0.06108268760807679)	(worth, 0.05393483422824445)
6	(polls, 0.2940215869174835)	(bjp, 0.2926233645776479)	(ghmc, 0.22011714149605233)	(seats, 0.13355320489864553)	(turnout, 0.11515979690575055)	(trs, 0.11515979690575055)	(voter, 0.11058249328904882)	(cm, 0.10428973992710787)	(wins, 0.10368049869169062)	(votes, 0.09079572571703264)
7	(farmers, 0.27950077999123013)	(protest, 0.19993999728008527)	(laws, 0.08863198612233647)	(farm, 0.08683847523541245)	(support, 0.06400197047090106)	(cm, 0.06395125561567935)	(delhi, 0.0564717221706181)	(punjab, 0.05034706376246343)	(amid, 0.04659745401598618)	(govt, 0.04369778535420213)
8	(online, 0.11149204470338839)	(startup, 0.09112764664065545)	(founder, 0.09112764664065545)	(ceo, 0.0836190335275413)	(manipal, 0.07786650962611462)	(gaming, 0.07786650962611462)	(recognised, 0.07786650962611462)	(jaipur, 0.07786650962611462)	(ugc, 0.07786650962611462)	(grab, 0.07786650962611462)
9	(woman, 0.19759436069130124)	(rape, 0.10367108745848326)	(held, 0.07799799002132288)	(gets, 0.07701158051387)	(daughter, 0.07290035064259497)	(girl, 0.07290035064259497)	(filming, 0.0567473285731454)	(elderly, 0.0567473285731454)	(raping, 0.0567473285731454)	(friend, 0.05398115479957057)
10	(married, 0.09098559574415407)	(page, 0.07470534394439393)	(transgender, 0.07470534394439393)	(elliot, 0.07106379872348531)	(wife, 0.05715731405680038)	(covid, 0.05539620518741297)	(girls, 0.053225209111521385)	(comes, 0.053225209111521385)	(sanon, 0.053225209111521385)	(woman, 0.05202484433391222)

The data consists news headlines of different categories, above are the TOP-10 topics from the documents.

Topic-0: NASA and Musk’s SpaceX rocket to bring rocks from the moon.
Topic-1: India intel shows fake Chinese soldier photo.
Topic-2: New Delhi, Ghaziabad of India have poor air-quality as per global index.
Topic-3: India extends lock-down to December due to COVID.
Topic-4: Kohli, Hardik made highest record runs in India-Australia match.

BERT on short-text data really worked well in clustering and generating topic of words alike. As it made use of TF-IDF vectorization it ranked words according to the score in identifying important words in topics.

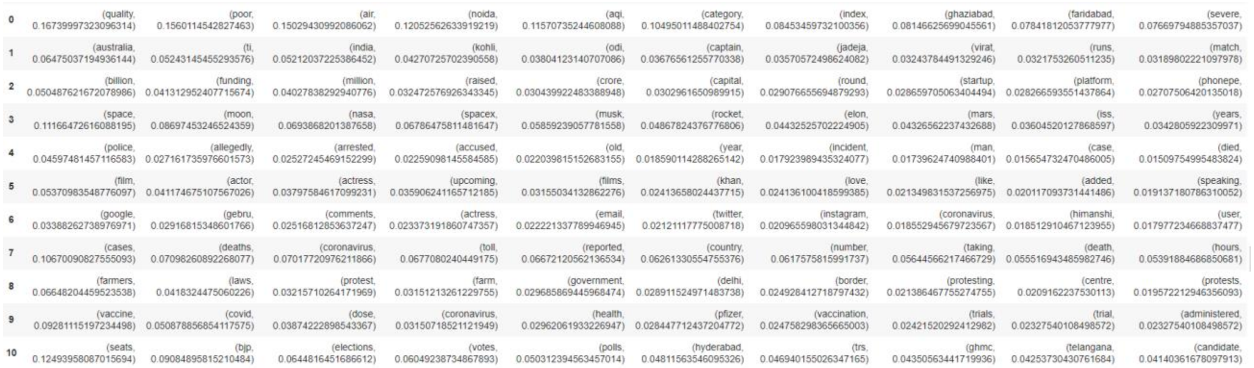
b) BERT on news-data:



```
[38] news[:10]

["virender sehwag captaining india hit runs west indies december , , becoming second cricketer sachin tendulkar score double hundred odis. sehwag's () highest individual score captain odi cricket history. sehwag's knock helped india p
 'psg forward jese rodriguez sacked club sex scandal involving wife\'s model-friend came light. rodriguez, also played real madrid, accused cheating wife aurah ruiz model-friend rocio amar. rocio revealed "sexual encounters" jese madr
 'late australian cricketing great don bradman\'s \'baggy green\' cap test debut england put auction week, pickles auctions said. "the baggy green loan state library south australia since ," pickles said statement. bradman retired tes
 't natarajan, made ti debut ongoing series australia, took twitter shared pictures second ti india clinched series. "first series win country\u200b. memorable special," wrote. natarajan took five wickets first two ti matches concedec
 'olympian anju bobby george, indian win medal world athletics championships, revealed one kidney. she tweeted, "believe not, i\'m one fortunate, among reached world top single kidney, allergic even painkiller, dead takeoff leg...many
 'boby chemmanur, kerala-born jeweller, announced building museum memory diego maradona, life-size gold sculpture football legend main attraction. the life-size sculpture represent ‘the hand god’ goal. the museum, showcase maradona’s
 'uk\'s the royal mint monday launched one ounce silver proof coin celebrate english singer-songwriter david bowie\'s "intergalactic legacy career". the coin orbited earth\'s atmosphere minutes safely descending. "in recognition bowie
 "bangladesh's education minister dipu moni tested positive coronavirus, abul khayer, education ministry spokesman, told journalists monday. the minister opted home isolation precautionary measure testing positive health condition goc
 'the world economic forum hold annual meeting singapore may - instead switzerland amid concerns coronavirus, organisers said monday. the event, known informally davos, return switzerland edition, added. it second time event hosted ou
 'delhi high court directed delhi government others follow prescribed safety measures, guidelines norms safe disposal swabs used rapid antigen testing covid-. the court said disposing pil, alleging used swabs thrown public tests carri
```

This data is scrapped from News website ‘InShorts’, which contains news from different categories like Cricket, Political, Sports, Startups etc..,

Result after applying BERT:



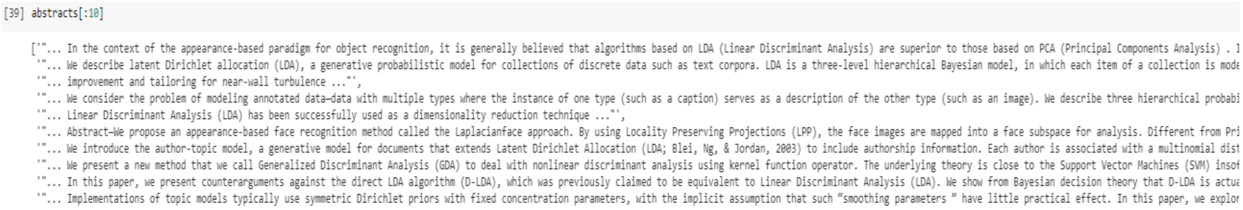
0	(quality, 0.16739997323096314)	(poor, 0.1560114542827463)	(air, 0.15029430992086062)	(noida, 0.12052562633919219)	(aqi, 0.11570735244608088)	(category, 0.10495011488402754)	(index, 0.08453459732100356)	(ghaziabad, 0.08146625699045561)	(faridabad, 0.07841812053777977)	(severe, 0.07669794885357037)
1	(australia, 0.06475037194936144)	(ti, 0.05243145455293576)	(india, 0.05212037225386452)	(kohli, 0.04270725702390558)	(odi, 0.03804123140707086)	(captain, 0.03676561255770338)	(jadeja, 0.03570572498624082)	(virat, 0.03243784491329246)	(runs, 0.0321753260511235)	(match, 0.03189802221097978)
2	(billion, 0.050487621672078986)	(funding, 0.041312952407715674)	(million, 0.04027838292940776)	(raised, 0.032472576926343345)	(crore, 0.030439922483388948)	(capital, 0.0302961650989915)	(round, 0.029076655694879293)	(startup, 0.028659705063404494)	(platform, 0.028266593551437864)	(phonepe, 0.02707506420135018)
3	(space, 0.11166472616088195)	(moon, 0.08697453246524359)	(nasa, 0.0693868201387658)	(spacex, 0.06786475811481647)	(musk, 0.05859239057781558)	(rocket, 0.04867824376776806)	(elon, 0.04432525702224905)	(mars, 0.04326562237432688)	(iss, 0.03604520127868597)	(years, 0.0342805922309971)
4	(police, 0.04597481457116583)	(allegedly, 0.027161735976601573)	(arrested, 0.02527245469152299)	(accused, 0.02259098145584585)	(old, 0.022039815152683155)	(year, 0.018590114288265142)	(incident, 0.017923989435324077)	(man, 0.01739624740988401)	(case, 0.015654732470486005)	(died, 0.01509754995483824)
5	(film, 0.05370983548776097)	(actor, 0.041174675107567026)	(actress, 0.03797584617099231)	(upcoming, 0.035906241165712185)	(films, 0.03155034132862276)	(khan, 0.02413658024437715)	(love, 0.02413610041859385)	(like, 0.021349831537256975)	(added, 0.020117093731441486)	(speaking, 0.019137180786310052)
6	(google, 0.03388262738976971)	(gebru, 0.02916815348601766)	(comments, 0.02516812853637247)	(actress, 0.023373191860747357)	(email, 0.022221337789946945)	(twitter, 0.02121117775008718)	(instagram, 0.020965598031344842)	(coronavirus, 0.018552945679723567)	(himanshi, 0.018512910467123955)	(user, 0.017977234668837477)
7	(cases, 0.10670090827555093)	(deaths, 0.07098260892268077)	(coronavirus, 0.07017720976211866)	(toll, 0.0677080240449175)	(reported, 0.06672120562136534)	(country, 0.06261330554755376)	(number, 0.0617575815991737)	(taking, 0.056445662174667729)	(death, 0.055516943485982746)	(hours, 0.05391884686850681)
8	(farmers, 0.06648204459523538)	(laws, 0.0418324475060226)	(protest, 0.03215710264171969)	(farm, 0.03151213261229755)	(government, 0.029685869445968474)	(delhi, 0.028911524971483738)	(border, 0.024928412718797432)	(protesting, 0.021386467755274755)	(centre, 0.0209162237530113)	(protests, 0.019572212946356093)
9	(vaccine, 0.09281115197234498)	(covid, 0.050878856854117575)	(dose, 0.03874222898543367)	(coronavirus, 0.03150718521121949)	(health, 0.02962061933226947)	(pfizer, 0.028447712437204772)	(vaccination, 0.024758298365665003)	(trials, 0.02421520292412982)	(trial, 0.023275401084985 72)	(administered, 0.023275401084985 72)
10	(seats, 0.12493958087015694)	(bjp, 0.09084895815210484)	(elections, 0.0644816451686612)	(votes, 0.06049238734867893)	(polls, 0.050312394563457014)	(hyderabad, 0.04811563546095326)	(trs, 0.04694015502634 7165)	(ghmc, 0.0435056344171 9936)	(telangana, 0.04253730430761684)	(candidate, 0.04140361678097913)

Topic-0: Poor air quality in Faridabad, Ghaziabad, Noida as per aqi(General news)
Topic-1: captain Virat kohli and Jadega runs India in ODI against Australia. (Cricket)
Topic-2: phonpe platform is a stratup from india that raised funding in millions and billions. (Startup).
Topic-3: NASA, Elon musk’s SpaceX, ISRO are planning to reach space for years. (Space)

Above are the news made according to the Topics, these words actually clustered with their occurrence together which is based on term frequencies of TF-IDF.

Application: BERT can be applied in News data to classify or categorize news according to category, as mentioned above. This accurately groups all the news according to the category.

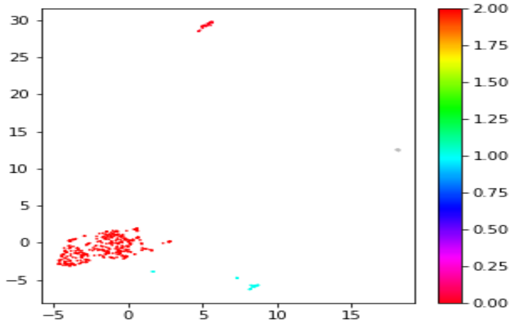
c) BERT on Scientific journals:



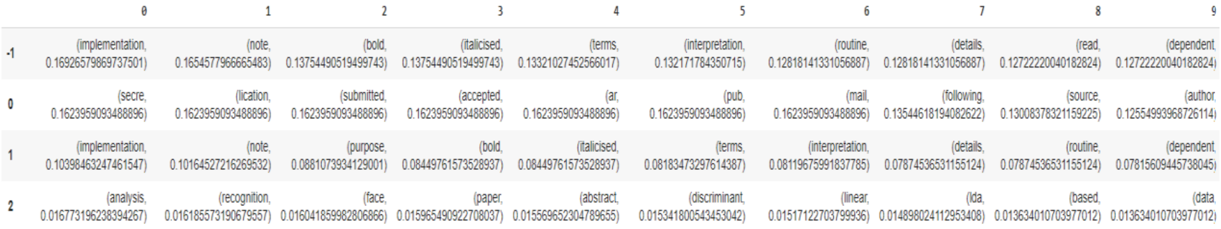
```
[39] abstracts[:10]

    ['"... In the context of the appearance-based paradigm for object recognition, it is generally believed that algorithms based on LDA (Linear Discriminant Analysis) are superior to those based on PCA (Principal Components Analysis) . I
     '"... We describe latent Dirichlet allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in which each item of a collection is mode
     '"... improvement and tailoring for near-wall turbulence ..."',
     '"... We consider the problem of modeling annotated data—data with multiple types where the instance of one type (such as a caption) serves as a description of the other type (such as an image). We describe three hierarchical probabi
     '"... Linear Discriminant Analysis (LDA) has been successfully used as a dimensionality reduction technique ..."',
     '"... Abstract—We propose an appearance-based face recognition method called the Laplacianface approach. By using Locality Preserving Projections (LPP), the face images are mapped into a face subspace for analysis. Different from Pri
     '"... We introduce the author-topic model, a generative model for documents that extends Latent Dirichlet Allocation (LDA; Blei, Ng, & Jordan, 2003) to include authorship information. Each author is associated with a multinomial dist
     '"... We present a new method that we call Generalized Discriminant Analysis (GDA) to deal with nonlinear discriminant analysis using kernel function operator. The underlying theory is close to the Support Vector Machines (SVM) insof
     '"... In this paper, we present counterarguments against the direct LDA algorithm (D-LDA), which was previously claimed to be equivalent to Linear Discriminant Analysis (LDA). We show from Bayesian decision theory that D-LDA is actua
     '"... Implementations of topic models typically use symmetric Dirichlet priors with fixed concentration parameters, with the implicit assumption that such "smoothing parameters " have little practical effect. In this paper, we explor
```

This is the data collected for citeseerX website which are papers written on the topic LDA.

Result of BERT:



The above graph shows the distribution of different clusters formed from the data, as we can see density of only one cluster is high this is because, the document contains data only on LDA.



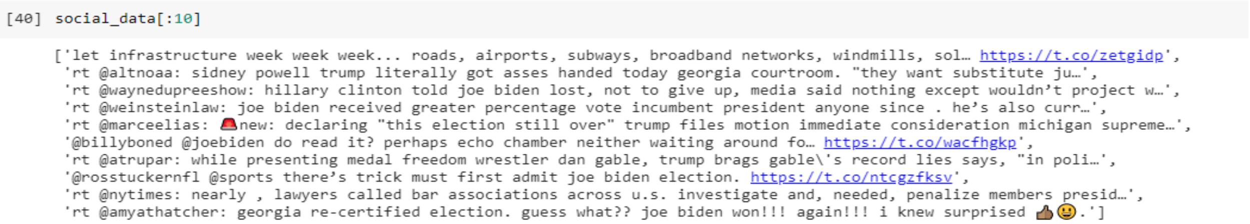
	0	1	2	3	4	5	6	7	8	9
-1	(implementation, 0.16926579869737501)	(note, 0.1654577966665483)	(bold, 0.13754490519499743)	(italicised, 0.13754490519499743)	(terms, 0.13321027452566017)	(interpretation, 0.132171784350715)	(routine, 0.1281814133105687)	(details, 0.1281814133105687)	(read, 0.12722220040182824)	(dependent, 0.12722220040182824)
0	(secre, 0.1623959093488896)	(lication, 0.1623959093488896)	(submitted, 0.1623959093488896)	(accepted, 0.1623959093488896)	(ar, 0.1623959093488896)	(pub, 0.1623959093488896)	(mail, 0.1623959093488896)	(following, 0.13544618194082622)	(source, 0.13008378321159225)	(author, 0.12554993968726114)
1	(implementation, 0.10398463247461547)	(note, 0.10164527216269532)	(purpose, 0.0881073934129001)	(bold, 0.08449761573528937)	(italicised, 0.08449761573528937)	(terms, 0.08183473297614387)	(interpretation, 0.08119675991837785)	(details, 0.07874536531155124)	(routine, 0.07874536531155124)	(dependent, 0.07815609445738045)
2	(analysis, 0.016773196238394267)	(recognition, 0.016185573190679557)	(face, 0.016041859982806866)	(paper, 0.015965490922708037)	(abstract, 0.015569652304789655)	(discriminant, 0.015341800543453042)	(linear, 0.015171227037999936)	(lda, 0.014898024112953408)	(based, 0.013634010703977012)	(data, 0.013634010703977012)

BERT topic modeling on scientific journals of similar topic yielded just three clusters with only one cluster having about 90% of values, this may be because several scientific journals use same context and words , so several words occur in many documents. As our model use TF-IDF vectorization topics are arranged according to frequencies.

Application: BERT is used to cluster different scientific journals based on the topic. Disadvantage: BERT is not that good in identifying topics from a scientific journals this is because it is based on tf-idf values, and also the topics formed are the words that occur more frequently like implementation, analysis, recognition etc.., which are not much of use.

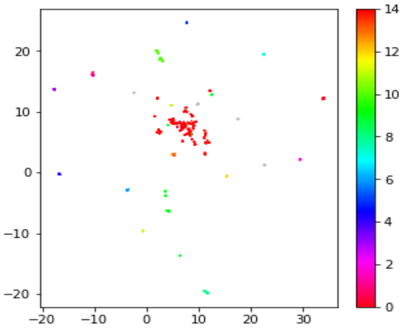
d) BERT on Social media data (Twitter):

Data used is scrapped from twitter based on keyword 'Joe Biden'.



```
[40] social_data[:10]

     ['let infrastructure week week week... roads, airports, subways, broadband networks, windmills, sol… https://t.co/zetgidp',
      'rt @altnoaa: sidney powell trump literally got asses handed today georgia courtroom. "they want substitute ju…',
      'rt @waynedupreeshow: hillary clinton told joe biden lost, not to give up, media said nothing except wouldn't project w…',
      'rt @weinsteinlaw: joe biden received greater percentage vote incumbent president anyone since . he's also curr…',
      'rt @marceelias: 🚨new: declaring "this election still over" trump files motion immediate consideration michigan supreme…',
      '@billyboned @joebiden do read it? perhaps echo chamber neither waiting around fo… https://t.co/wacfhgkp',
      'rt @atrupar: while presenting medal freedom wrestler dan gable, trump brags gable\'s record lies says, "in poli…',
      '@rosstuckernfl @sports there's trick must first admit joe biden election. https://t.co/ntcgzfksv',
      'rt @nytimes: nearly , lawyers called bar associations across u.s. investigate and, needed, penalize members presid…',
      'rt @amyathatcher: georgia re-certified election. guess what?? joe biden won!!! again!!! i knew surprised 👍😀.']
```

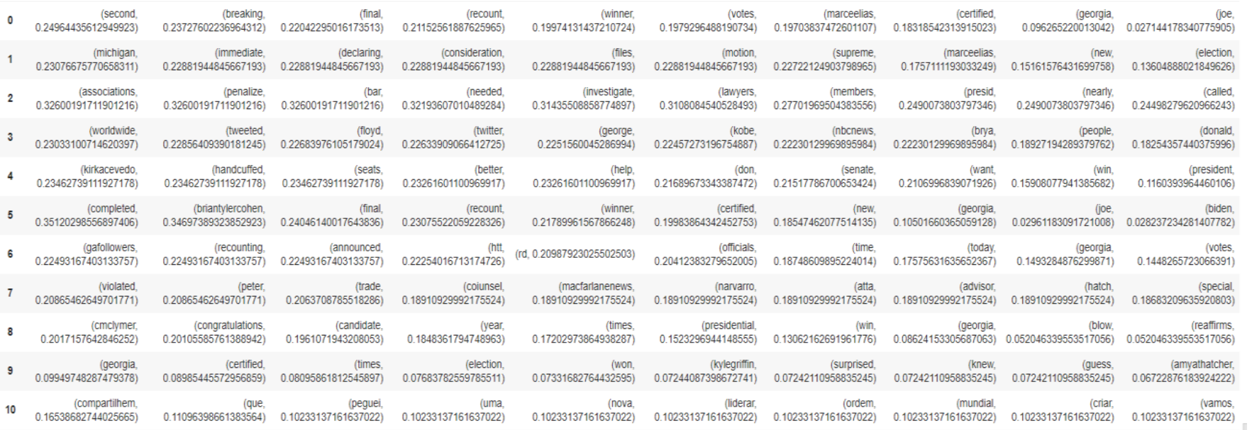
Results after applying BERT:



```
    Topic  Size
15     14   442
10      9    71
11     10    63
2       1    51
1       0    48
9       8    43
4       3    40
0      -1    35
5       4    35
14     13    31
7       6    28
12     11    25
8       7    24
13     12    23
6       5    21
3       2    20
```

As we can see above the cluster formed are about 15, but most of the words are in the cluster 15.

Topics:

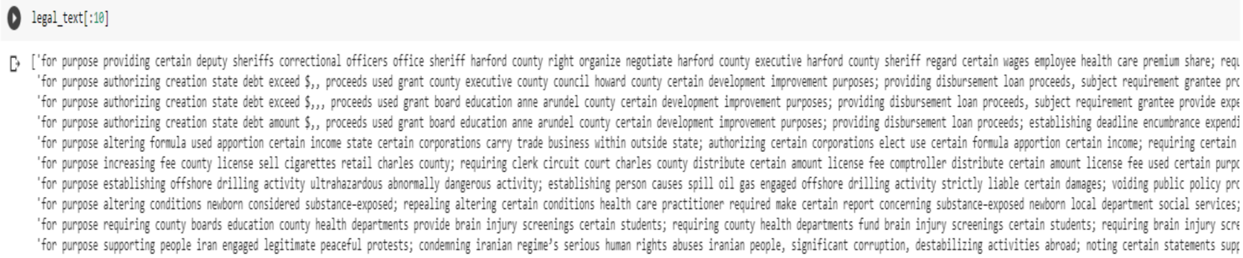


0	(second, 0.24964435612949923)	(breaking, 0.23727602236964312)	(final, 0.22042295016173513)	(recount, 0.21152561887625965)	(winner, 0.19974131437210724)	(votes, 0.1979296488190734)	(marceelias, 0.19703837472601107)	(certified, 0.18318542313915023)	(georgia, 0.096265220013042)	(joe, 0.027144178340775905)
1	(michigan, 0.23076675770658311)	(immediate, 0.22881944845667193)	(declaring, 0.22881944845667193)	(consideration, 0.22881944845667193)	(files, 0.22881944845667193)	(motion, 0.22881944845667193)	(supreme, 0.22722124903798965)	(marceelias, 0.1757111193033249)	(new, 0.15161576431699758)	(election, 0.13604888021849626)
2	(associations, 0.32600191711901216)	(penalize, 0.32600191711901216)	(bar, 0.32600191711901216)	(needed, 0.32193607010489284)	(investigate, 0.31435508858774897)	(lawyers, 0.3108084540528493)	(members, 0.27701969504383556)	(presid, 0.2490073803797346)	(nearly, 0.2490073803797346)	(called, 0.24498279620966243)
3	(worldwide, 0.23033100714620397)	(tweeted, 0.22856409390181245)	(floyd, 0.22683976105179024)	(twitter, 0.22633909066412725)	(george, 0.2251560045286994)	(kobe, 0.22457273196754887)	(nbcnews, 0.22230129969895984)	(brya, 0.22230129969895984)	(people, 0.18927194289379762)	(donald, 0.18254357440375996)
4	(kirkacevedo, 0.23462739111927178)	(handcuffed, 0.23462739111927178)	(seats, 0.23462739111927178)	(better, 0.23261601100969917)	(help, 0.23261601100969917)	(don, 0.21689673343387472)	(senate, 0.21517786700653424)	(want, 0.2106996839071926)	(win, 0.15908077941385682)	(president, 0.1160393964460106)
5	(completed, 0.35120298556897406)	(briantylercohen, 0.34697389323852923)	(final, 0.24046140017643836)	(recount, 0.23075522059228326)	(winner, 0.21789961567866248)	(certified, 0.19983864342452753)	(new, 0.18547462077514135)	(georgia, 0.10501660365059128)	(joe, 0.02961183091721008)	(biden, 0.028237234281407782)
6	(gafollowers, 0.22493167403133757)	(recounting, 0.22493167403133757)	(announced, 0.22493167403133757)	(htt, 0.22254016713174726)	(rd, 0.20987923025502503)	(officials, 0.20412383279652005)	(time, 0.18748609895224014)	(today, 0.17575631635652367)	(georgia, 0.1493284876299871)	(votes, 0.1448265723066391)
7	(violated, 0.20865462649701771)	(peter, 0.20865462649701771)	(trade, 0.2063708785518286)	(coiunsel, 0.18910929992175524)	(macfarlanenews, 0.18910929992175524)	(narvarro, 0.18910929992175524)	(atta, 0.18910929992175524)	(advisor, 0.18910929992175524)	(hatch, 0.18910929992175524)	(special, 0.18683209635920803)
8	(cmclymer, 0.2017157642846252)	(congratulations, 0.20105585761388942)	(candidate, 0.1961071943208053)	(year, 0.1848361794748963)	(times, 0.17202973864938287)	(presidential, 0.1523296944148555)	(win, 0.13062162691961776)	(georgia, 0.08624153305687063)	(blow, 0.052046339553517056)	(reaffirms, 0.052046339553517056)
9	(georgia, 0.09949748287479378)	(certified, 0.08985445572956859)	(times, 0.08095861812545897)	(election, 0.07683782559785511)	(won, 0.07331682764432595)	(kylegriffin, 0.07244087398672741)	(surprised, 0.07242110958835245)	(knew, 0.07242110958835245)	(guess, 0.07242110958835245)	(amyathatcher, 0.06722876183924222)
10	(compartilhem, 0.16538682744025665)	(que, 0.11096398661383564)	(peguei, 0.10233137161637022)	(uma, 0.10233137161637022)	(nova, 0.10233137161637022)	(liderar, 0.10233137161637022)	(ordem, 0.10233137161637022)	(mundial, 0.10233137161637022)	(criar, 0.10233137161637022)	(vamos, 0.10233137161637022)

Topic-0: Georgia recount votes after declaring winner joe
Topic-1: After declaring Michigan votes an immediate declaration of a motion is filed.
Topic-2:Bar need to investigate and penalize lawyers.

BERT on social media data works good, but as it is unorganized and may be tweeted by individuals with some biased words, meaning and topics tend to change. As social media is place with a slight censorship there may appear words with several biasing elements, so it not that good to use BERT on social media.

e) BERT on legal data:



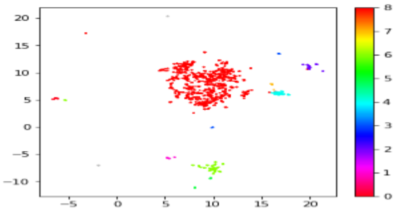
```
legal_text[:10]
```

```
['for purpose providing certain deputy sheriffs correctional officers office sheriff harford county right organize negotiate harford county executive harford county sheriff regard certain wages employee health care premium share; requ
 'for purpose authorizing creation state debt exceed $,, proceeds used grant county executive county council howard county certain development improvement purposes; providing disbursement loan proceeds, subject requirement grantee prc
 'for purpose authorizing creation state debt exceed $,,, proceeds used grant board education anne arundel county certain development improvement purposes; providing disbursement loan proceeds, subject requirement grantee provide expe
 'for purpose authorizing creation state debt amount $,, proceeds used grant board education anne arundel county certain development improvement purposes; providing disbursement loan proceeds; establishing deadline encumbrance expendi
 'for purpose altering formula used apportion certain income state certain corporations carry trade business within outside state; authorizing certain corporations elect use certain formula apportion certain income; requiring certain
 'for purpose increasing fee county license sell cigarettes retail charles county; requiring clerk circuit court charles county distribute certain amount license fee comptroller distribute certain amount license fee used certain purpc
 'for purpose establishing offshore drilling activity ultrahazardous abnormally dangerous activity; establishing person causes spill oil gas engaged offshore drilling activity strictly liable certain damages; voiding public policy prc
 'for purpose altering conditions newborn considered substance-exposed; repealing altering certain conditions health care practitioner required make certain report concerning substance-exposed newborn local department social services;
 'for purpose requiring county boards education county health departments provide brain injury screenings certain students; requiring county health departments fund brain injury screenings certain students; requiring brain injury scre
 'for purpose supporting people iran engaged legitimate peaceful protests; condemning iranian regime’s serious human rights abuses iranian people, significant corruption, destabilizing activities abroad; noting certain statements supp
```

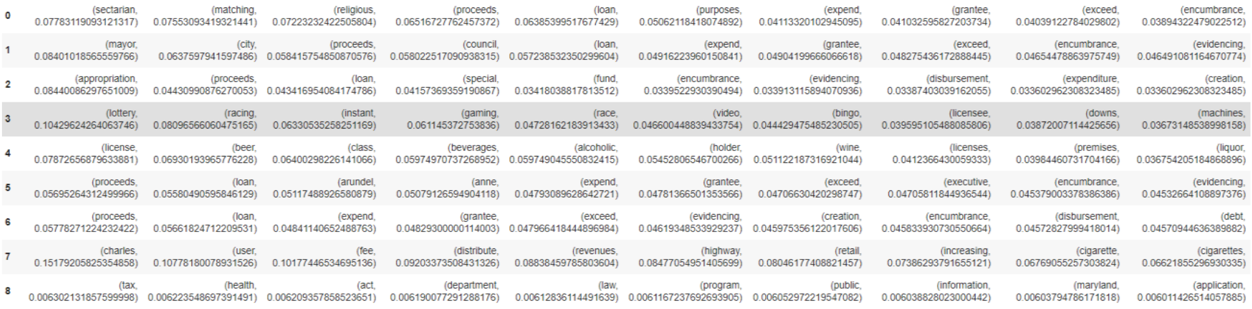
This data is scrapped from Maryland general assembly website which has details about the proposed bills.

Results:

```
   Topic  Size
9      8  2617
7      6   226
5      4   143
3      2    94
1      0    45
4      3    38
2      1    32
6      5    27
8      7    16
0     -1    12
```



Topic clusters:



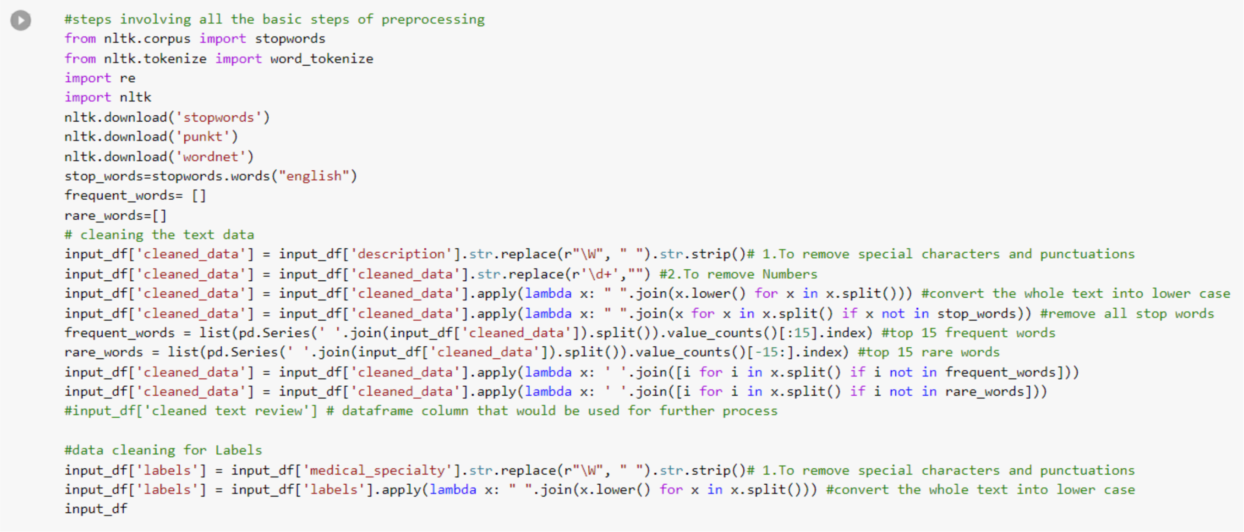
0	(sectarian, 0.07783119093121317)	(matching, 0.07553093419321441)	(religious, 0.07223232422505804)	(proceeds, 0.06516727762457372)	(loan, 0.06385399517677429)	(purposes, 0.05062118418074892)	(expend, 0.04113320102945095)	(grantee, 0.041032595827203734)	(exceed, 0.04039122784029802)	(encumbrance, 0.03894322479022512)
1	(mayor, 0.08401018565559766)	(city, 0.0637597941597486)	(proceeds, 0.058415754850870576)	(council, 0.058022517090938315)	(loan, 0.057238532350299604)	(expend, 0.04916223960150841)	(grantee, 0.04904199666066618)	(exceed, 0.04827543617288845)	(encumbrance, 0.04654478863975749)	(evidencing, 0.04649108116467077 4)
2	(appropriation, 0.08440086297651009)	(proceeds, 0.04430990876270053)	(loan, 0.043416954084174786)	(special, 0.04157369359190867)	(fund, 0.03418038817813512)	(encumbrance, 0.03395229303 90494)	(evidencing, 0.033913115894070936)	(disbursement, 0.033874030391620 55)	(expenditure, 0.033602962308323485)	(creation, 0.033602962308323485)
3	(lottery, 0.10429624264063746)	(racing, 0.08096566060475165)	(instant, 0.06330535258251169)	(gaming, 0.06114537275383 6)	(race, 0.04728162183913433)	(video, 0.046600448839433754)	(bingo, 0.044429475485230505)	(licensee, 0.039595105488085806)	(downs, 0.03872007114425656)	(machines, 0.03673148538998158)
4	(license, 0.07872656879633881)	(beer, 0.06930193965776228)	(class, 0.06400298226141066)	(beverages, 0.05974970737268952)	(alcoholic, 0.059749045550832415)	(holder, 0.05452806546700266)	(wine, 0.051122187316921044)	(licenses, 0.041236643005 9333)	(premises, 0.03984460731704166)	(liquor, 0.036754205184868896)
5	(proceeds, 0.05695264312499966)	(loan, 0.05580490595846129)	(arundel, 0.05117488926580879)	(anne, 0.05079126594904118)	(expend, 0.04793089628642721)	(grantee, 0.04781366501353566)	(exceed, 0.04706630420298747)	(executive, 0.04705811844936544)	(encumbrance, 0.04537900337838 6386)	(evidencing, 0.04532664108897376)
6	(proceeds, 0.05778271224232422)	(loan, 0.05661824712209531)	(expend, 0.04841140652488763)	(grantee, 0.04829300000114003)	(exceed, 0.047966418444896984)	(evidencing, 0.04619348533929237)	(creation, 0.04597535612201 7606)	(encumbrance, 0.045833930730550664)	(disbursement, 0.04572827999418014)	(debt, 0.04570944636389882)
7	(charles, 0.15179205825354858)	(user, 0.10778180078931526)	(fee, 0.10177446534695136)	(distribute, 0.09203373508431326)	(revenues, 0.08838459785803604)	(highway, 0.08477054951405699)	(retail, 0.08046177408821457)	(increasing, 0.07386293791655121)	(cigarette, 0.06769055257303824)	(cigarettes, 0.06621855296930335)
8	(tax, 0.006302131857599998)	(health, 0.006223548697391491)	(act, 0.006209357858523651)	(department, 0.006190077291288176)	(law, 0.006128361144916 39)	(program, 0.006116723769269390 5)	(public, 0.006052972219547082)	(information, 0.006038828023000442)	(maryland, 0.00603794786171818)	(application, 0.006011426514057885)

Topic-1: City mayor proceeds with council to encumbrance of loan.
Topic-2: Lotter, gaming machines need license.
Topic-3: Beverage, alcoholic and beer premises need licenses.

4.Labelled Latent Dirichlet Allocation (LLDA):

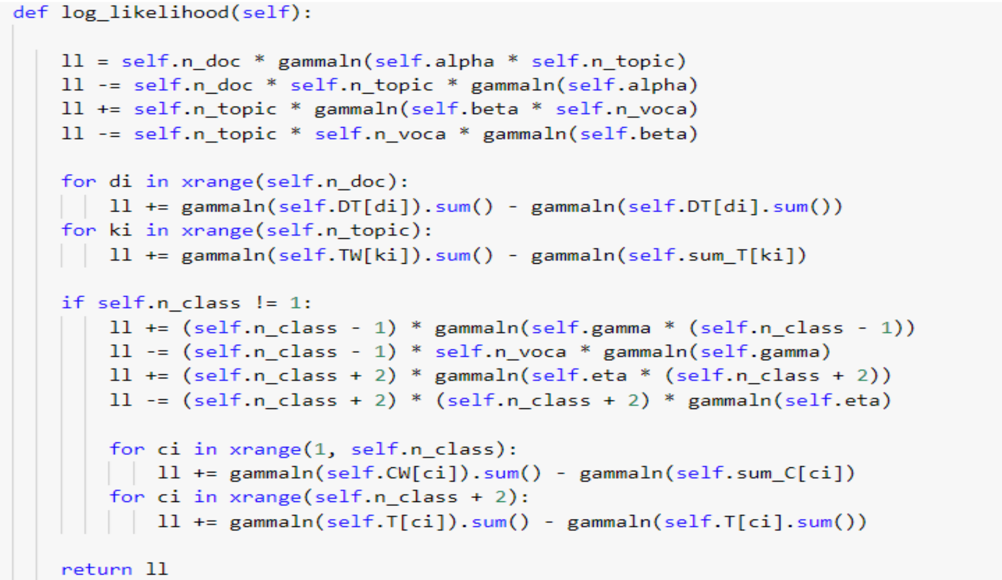
We have seen topic modelling approaches before, however LLDA method tends to over focus on the pre- assigned labels and it almost ignores lost labels and common semantics. Labeled LDA is a topic model that constrains Latent Dirichlet Allocation by defining a one-to-one correspondence between LDA's latent topics and user tags. Like Latent Dirichlet Allocation, Labeled LDA models each document as a mixture of underlying topics and generates each word from one topic. Unlike LDA, L-LDA incorporates supervision by simply constraining the topic model to use only those topics that correspond to a document's (observed) label set. Labeled LDA's credit attribution mechanism can be used to augment the view of a single document with rich contextual information about the document's tags. Thus, in this context we need to have a dataset, that's pre labeled.

Text Cleaning:



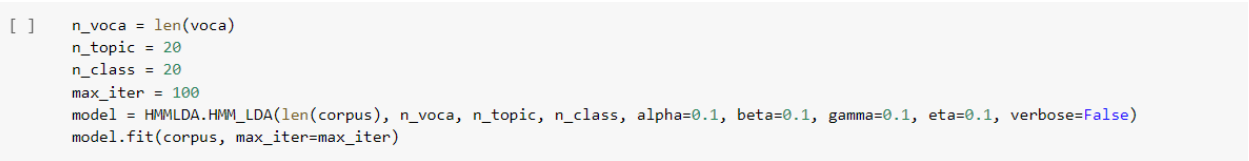
```
#steps involving all the basic steps of preprocessing
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import re
import nltk
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
stop_words=stopwords.words("english")
frequent_words= []
rare_words=[]
# cleaning the text data
input_df['cleaned_data'] = input_df['description'].str.replace(r"\W", " ").str.strip()# 1.To remove special characters and punctuations
input_df['cleaned_data'] = input_df['cleaned_data'].str.replace(r'\d+',"") #2.To remove Numbers
input_df['cleaned_data'] = input_df['cleaned_data'].apply(lambda x: " ".join(x.lower() for x in x.split())) #convert the whole text into lower case
input_df['cleaned_data'] = input_df['cleaned_data'].apply(lambda x: " ".join(x for x in x.split() if x not in stop_words)) #remove all stop words
frequent_words = list(pd.Series(' '.join(input_df['cleaned_data']).split()).value_counts()[:15].index) #top 15 frequent words
rare_words = list(pd.Series(' '.join(input_df['cleaned_data']).split()).value_counts()[-15:].index) #top 15 rare words
input_df['cleaned_data'] = input_df['cleaned_data'].apply(lambda x: ' '.join([i for i in x.split() if i not in frequent_words]))
input_df['cleaned_data'] = input_df['cleaned_data'].apply(lambda x: ' '.join([i for i in x.split() if i not in rare_words]))
#input_df['cleaned text review'] # dataframe column that would be used for further process

#data cleaning for Labels
input_df['labels'] = input_df['medical_specialty'].str.replace(r"\W", " ").str.strip()# 1.To remove special characters and punctuations
input_df['labels'] = input_df['labels'].apply(lambda x: " ".join(x.lower() for x in x.split())) #convert the whole text into lower case
input_df
```

Likelihood

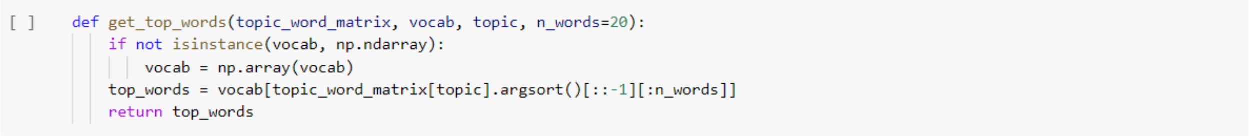


```
def log_likelihood(self):

    ll = self.n_doc * gammaln(self.alpha * self.n_topic)
    ll -= self.n_doc * self.n_topic * gammaln(self.alpha)
    ll += self.n_topic * gammaln(self.beta * self.n_voca)
    ll -= self.n_topic * self.n_voca * gammaln(self.beta)

    for di in xrange(self.n_doc):
        ll += gammaln(self.DT[di]).sum() - gammaln(self.DT[di].sum())
    for ki in xrange(self.n_topic):
        ll += gammaln(self.TW[ki]).sum() - gammaln(self.sum_T[ki])

    if self.n_class != 1:
        ll += (self.n_class - 1) * gammaln(self.gamma * (self.n_class - 1))
        ll -= (self.n_class - 1) * self.n_voca * gammaln(self.gamma)
        ll += (self.n_class + 2) * gammaln(self.eta * (self.n_class + 2))
        ll -= (self.n_class + 2) * (self.n_class + 2) * gammaln(self.eta)

        for ci in xrange(1, self.n_class):
            ll += gammaln(self.CW[ci]).sum() - gammaln(self.sum_C[ci])
        for ci in xrange(self.n_class + 2):
            ll += gammaln(self.T[ci]).sum() - gammaln(self.T[ci].sum())

    return ll
```

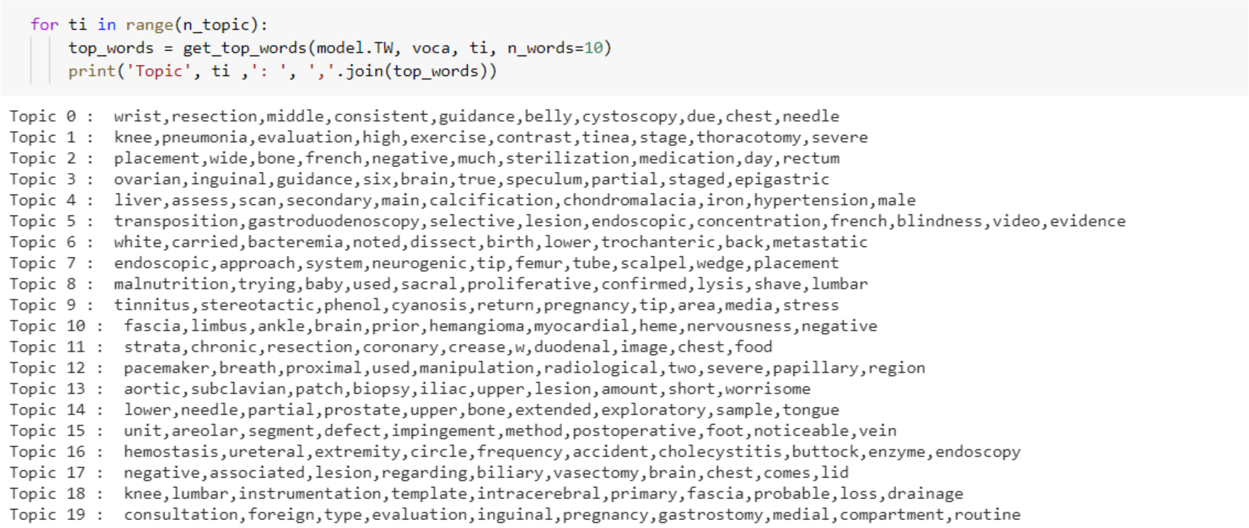
Result



```
[ ]   n_voca = len(voca)
      n_topic = 20
      n_class = 20
      max_iter = 100
      model = HMMLDA.HMM_LDA(len(corpus), n_voca, n_topic, n_class, alpha=0.1, beta=0.1, gamma=0.1, eta=0.1, verbose=False)
      model.fit(corpus, max_iter=max_iter)
```



```
[ ]   def get_top_words(topic_word_matrix, vocab, topic, n_words=20):
          if not isinstance(vocab, np.ndarray):
              vocab = np.array(vocab)
          top_words = vocab[topic_word_matrix[topic].argsort()[::-1][:n_words]]
          return top_words
```



```
for ti in range(n_topic):
    top_words = get_top_words(model.TW, voca, ti, n_words=10)
    print('Topic', ti ,': ', ','.join(top_words))
```

```
Topic 0 :  wrist,resection,middle,consistent,guidance,belly,cystoscopy,due,chest,needle
Topic 1 :  knee,pneumonia,evaluation,high,exercise,contrast,tinea,stage,thoracotomy,severe
Topic 2 :  placement,wide,bone,french,negative,much,sterilization,medication,day,rectum
Topic 3 :  ovarian,inguinal,guidance,six,brain,true,speculum,partial,staged,epigastric
Topic 4 :  liver,assess,scan,secondary,main,calcification,chondromalacia,iron,hypertension,male
Topic 5 :  transposition,gastroduodenoscopy,selective,lesion,endoscopic,concentration,french,blindness,video,evidence
Topic 6 :  white,carried,bacteremia,noted,dissect,birth,lower,trochanteric,back,metastatic
Topic 7 :  endoscopic,approach,system,neurogenic,tip,femur,tube,scalpel,wedge,placement
Topic 8 :  malnutrition,trying,baby,used,sacral,proliferative,confirmed,lysis,shave,lumbar
Topic 9 :  tinnitus,stereotactic,phenol,cyanosis,return,pregnancy,tip,area,media,stress
Topic 10 :  fascia,limbus,ankle,brain,prior,hemangioma,myocardial,heme,nervousness,negative
Topic 11 :  strata,chronic,resection,coronary,crease,w,duodenal,image,chest,food
Topic 12 :  pacemaker,breath,proximal,used,manipulation,radiological,two,severe,papillary,region
Topic 13 :  aortic,subclavian,patch,biopsy,iliac,upper,lesion,amount,short,worrisome
Topic 14 :  lower,needle,partial,prostate,upper,bone,extended,exploratory,sample,tongue
Topic 15 :  unit,areolar,segment,defect,impingement,method,postoperative,foot,noticeable,vein
Topic 16 :  hemostasis,ureteral,extremity,circle,frequency,accident,cholecystitis,buttock,enzyme,endoscopy
Topic 17 :  negative,associated,lesion,regarding,biliary,vasectomy,brain,chest,comes,lid
Topic 18 :  knee,lumbar,instrumentation,template,intracerebral,primary,fascia,probable,loss,drainage
Topic 19 :  consultation,foreign,type,evaluation,inguinal,pregnancy,gastrostomy,medial,compartment,routine
```

5. BITERM TOPIC MODEL

The Biterm Topic Model (BTM) is a word co-occurrence-based topic model that learns topics by modeling word-word co-occurrences patterns. BTM models the Biterm occurrences in a corpus (unlike LDA models which model the word occurrences in a document). It's a generative model. In the generation procedure, a biterm is generated by drawing two words independently from a same topic z. In other words, the distribution of a biterm b=(wi,wj) is defined as: P(b) = ∑_k{P(wi|z)*P(wj|z)*P(z)} where k is the number of topics you want to extract. Estimation of the topic model is done with the Gibbs sampling algorithm. Where estimates are provided for P(w|k)=phi and P(z)=theta.
Function and arguments: BTM(data, k, alpha , beta ,iter, window , background, trace ,biterms, detailed)

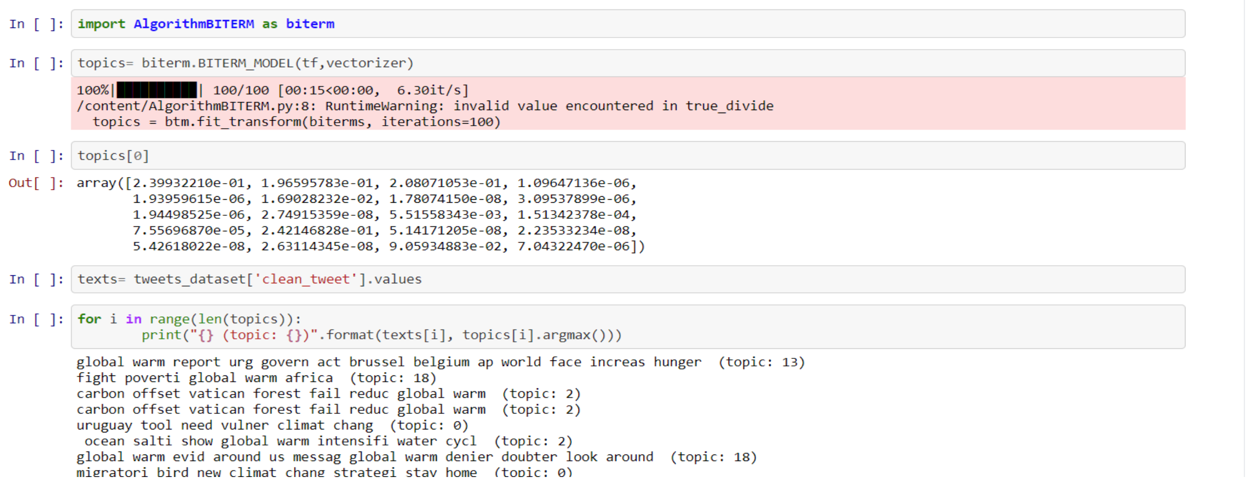
Value: an object of class BTM contains

- model: a pointer to the C++ BTM model
- K: the number of topics
- W: the number of tokens in the data
- alpha: the symmetric dirichlet prior probability of a topic P(z)
- beta: the symmetric dirichlet prior probability of a word given the topic P(w|z)

- iter: the number of iterations of Gibbs sampling
- background: indicator if the first topic is set to the background topic that equals the empirical word distribution.
- theta: a vector with the topic probability p(z) which is determined by the overall proportions of biterms in it
- phi: a matrix of dimension W x K with one row for each token in the data. This matrix contains the probability of the token given the topic P(w|z). the row names of the matrix indicate the token w
- vocab: a data Frame with columns token and freq indicating the frequency of occurrence of the tokens in data. Only provided in case argument detailed is set to TRUE
- biterms: the result of a call to terms with type set to biterms, containing all the biterms used in the model. Only provided in case argument detailed is set to TRU

Advantages:
The major advantages of BTM are that
1) BTM explicitly models the word co-occurrence patterns to enhance the topic learning
2) BTM uses the aggregated patterns in the whole corpus for learning topics to solve the problem of sparse word co-occurrence patterns at document-level.
Biterms is best suited for short text, spanning about a sentence

Results

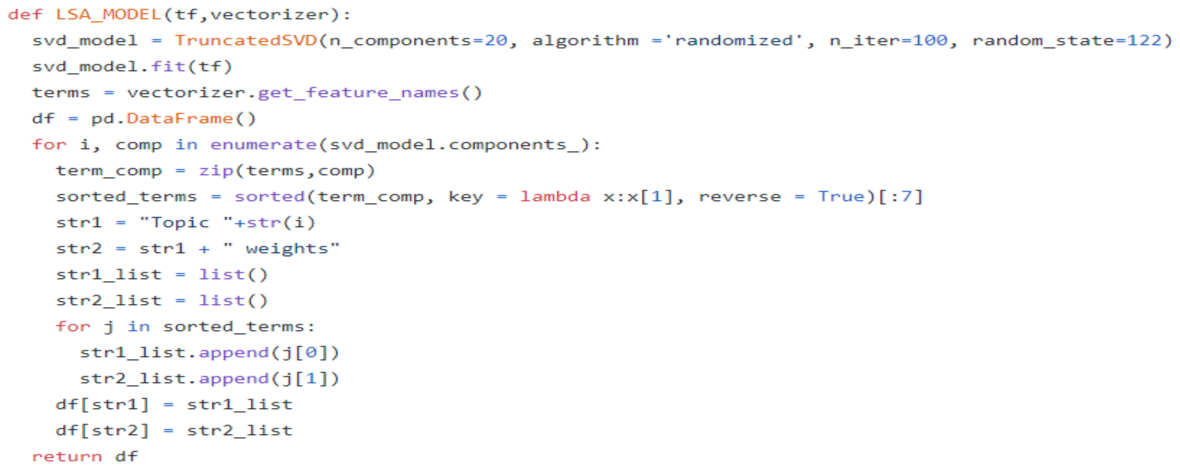


```
In [ ]: import AlgorithmBITERM as biterm

In [ ]: topics= biterm.BITERM_MODEL(tf,vectorizer)

        100%|██████████| 100/100 [00:15<00:00,  6.30it/s]
        /content/AlgorithmBITERM.py:8: RuntimeWarning: invalid value encountered in true_divide
          topics = btm.fit_transform(biterms, iterations=100)

In [ ]: topics[0]

Out[ ]: array([2.39932210e-01, 1.96595783e-01, 2.08071053e-01, 1.09647136e-06,
               1.93959615e-06, 1.69028232e-02, 1.78074150e-08, 3.09537899e-06,
               1.94498525e-06, 2.74915359e-08, 5.51558343e-03, 1.51342378e-04,
               7.55696870e-05, 2.42146828e-01, 5.14171205e-08, 2.23533234e-08,
               5.42618022e-08, 2.63114345e-08, 9.05934883e-02, 7.04322470e-06])

In [ ]: texts= tweets_dataset['clean_tweet'].values

In [ ]: for i in range(len(topics)):
                print("{} (topic: {})".format(texts[i], topics[i].argmax()))

        global warm report urg govern act brussel belgium ap world face increas hunger  (topic: 13)
        fight poverti global warm africa  (topic: 18)
        carbon offset vatican forest fail reduc global warm  (topic: 2)
        carbon offset vatican forest fail reduc global warm  (topic: 2)
        uruguay tool need vulner climat chang  (topic: 0)
         ocean salti show global warm intensifi water cycl  (topic: 2)
        global warm evid around us messag global warm denier doubter look around  (topic: 18)
        migratori bird new climat chang strategi stay home  (topic: 0)
```

6.LATENT SEMANTIC ANALYSIS (LSA):

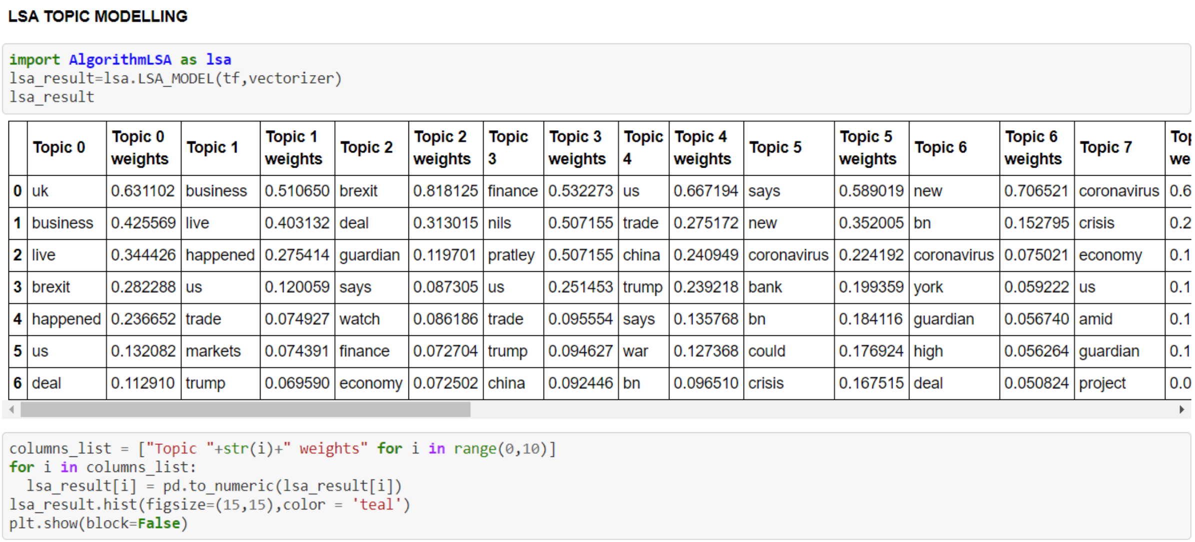
Latent Semantic Analysis (LSA) is one of the most theological techniques of topic modeling. LSA produces a set of concepts related to documents and the terms they contain to analyze the relationship among them. The basic idea process idea of LSA is to generate a document-term matrix with our in-scope documents and included terms and decompose into document-topic and topic-term matrices. For available ‘m’ documents and ‘n’ terms we shall create ‘m x n’ document-term matrix where, a row represents a document, and a column represents a word from that document. Through tf-idf, we assign a weight for a particular term ‘x’ in a particular document ‘y’. Most frequently occurring in a document but infrequently occurring term in the corpus would be allocated a large weight. The dimensionality reduction is achieved by truncated singular value decomposition (SVD). We express document vectors and term vectors in terms of topics. Once we have document vectors and term vectors, we can evaluate similarity metrics such as similarity among documents, terms and among document-terms by applying measures like cosine similarity. It is an efficient technique and quick to use.

Algorithm Implementation:



```
def LSA_MODEL(tf,vectorizer):
  svd_model = TruncatedSVD(n_components=20, algorithm ='randomized', n_iter=100, random_state=122)
  svd_model.fit(tf)
  terms = vectorizer.get_feature_names()
  df = pd.DataFrame()
  for i, comp in enumerate(svd_model.components_):
    term_comp = zip(terms,comp)
    sorted_terms = sorted(term_comp, key = lambda x:x[1], reverse = True)[:7]
    str1 = "Topic "+str(i)
    str2 = str1 + " weights"
    str1_list = list()
    str2_list = list()
    for j in sorted_terms:
      str1_list.append(j[0])
      str2_list.append(j[1])
    df[str1] = str1_list
    df[str2] = str2_list
  return df
```

Results

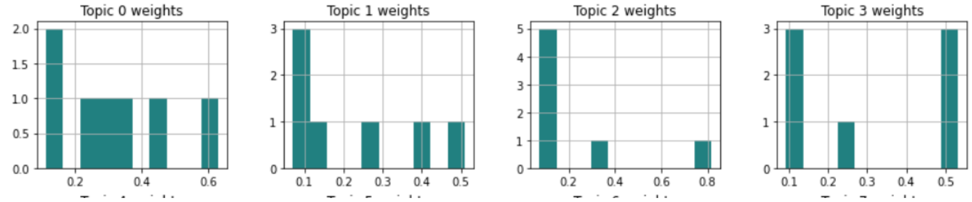


LSA TOPIC MODELLING

```
import AlgorithmLSA as lsa
lsa_result=lsa.LSA_MODEL(tf,vectorizer)
lsa_result
```

	Topic 0	Topic 0 weights	Topic 1	Topic 1 weights	Topic 2	Topic 2 weights	Topic 3	Topic 3 weights	Topic 4	Topic 4 weights	Topic 5	Topic 5 weights	Topic 6	Topic 6 weights	Topic 7	Top we
0	uk	0.631102	business	0.510650	brexit	0.818125	finance	0.532273	us	0.667194	says	0.589019	new	0.706521	coronavirus	0.6
1	business	0.425569	live	0.403132	deal	0.313015	nils	0.507155	trade	0.275172	new	0.352005	bn	0.152795	crisis	0.2
2	live	0.344426	happened	0.275414	guardian	0.119701	pratley	0.507155	china	0.240949	coronavirus	0.224192	coronavirus	0.075021	economy	0.1
3	brexit	0.282288	us	0.120059	says	0.087305	us	0.251453	trump	0.239218	bank	0.199359	york	0.059222	us	0.1
4	happened	0.236652	trade	0.074927	watch	0.086186	trade	0.095554	says	0.135768	bn	0.184116	guardian	0.056740	amid	0.1
5	us	0.132082	markets	0.074391	finance	0.072704	trump	0.094627	war	0.127368	could	0.176924	high	0.056264	guardian	0.1
6	deal	0.112910	trump	0.069590	economy	0.072502	china	0.092446	bn	0.096510	crisis	0.167515	deal	0.050824	project	0.0

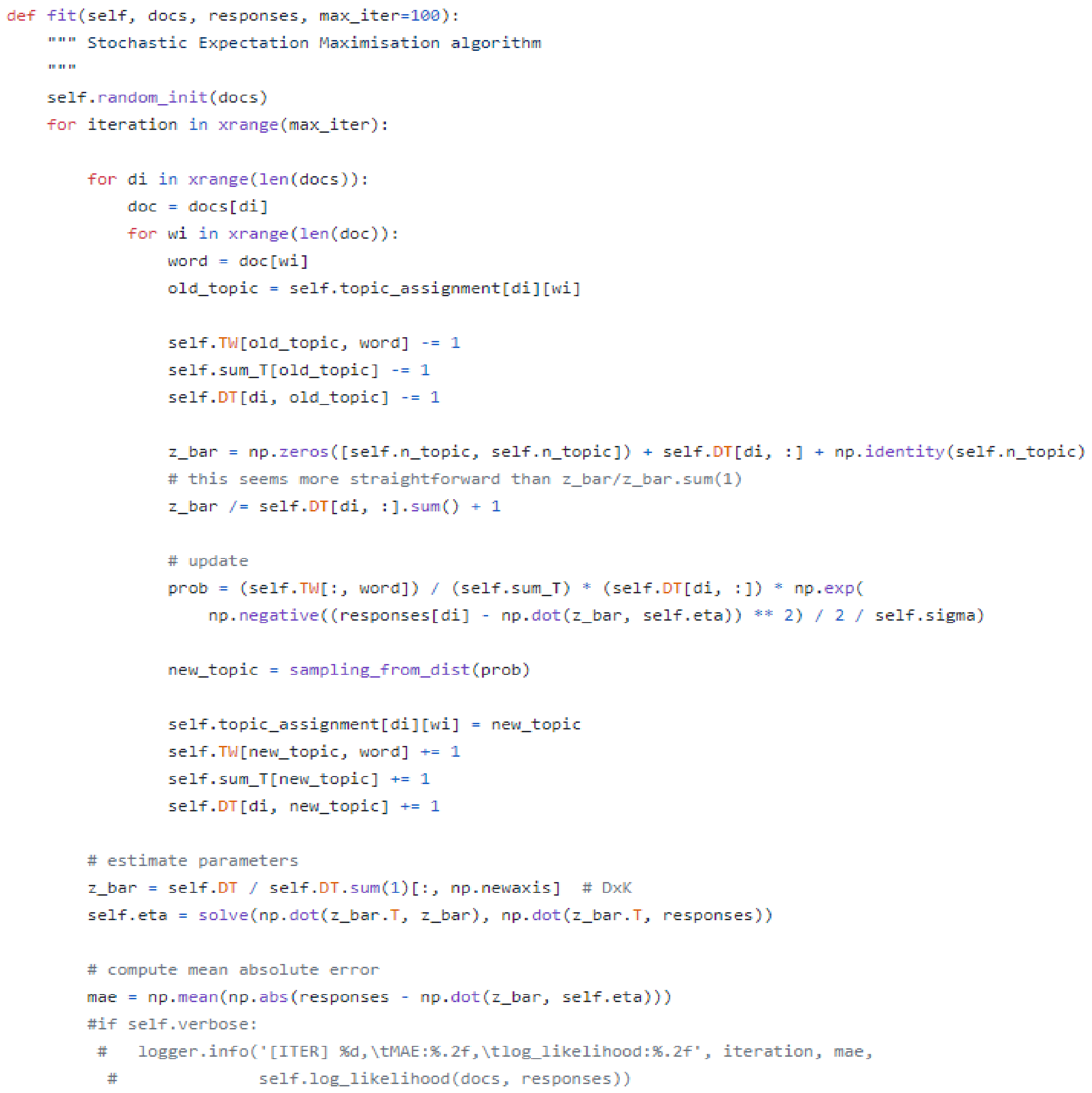
```
columns_list = ["Topic "+str(i)+" weights" for i in range(0,10)]
for i in columns_list:
  lsa_result[i] = pd.to_numeric(lsa_result[i])
lsa_result.hist(figsize=(15,15),color = 'teal')
plt.show(block=False)
```



7. SUPERVISED LATENT DIRICHLET ALLOCATION (SLDA):

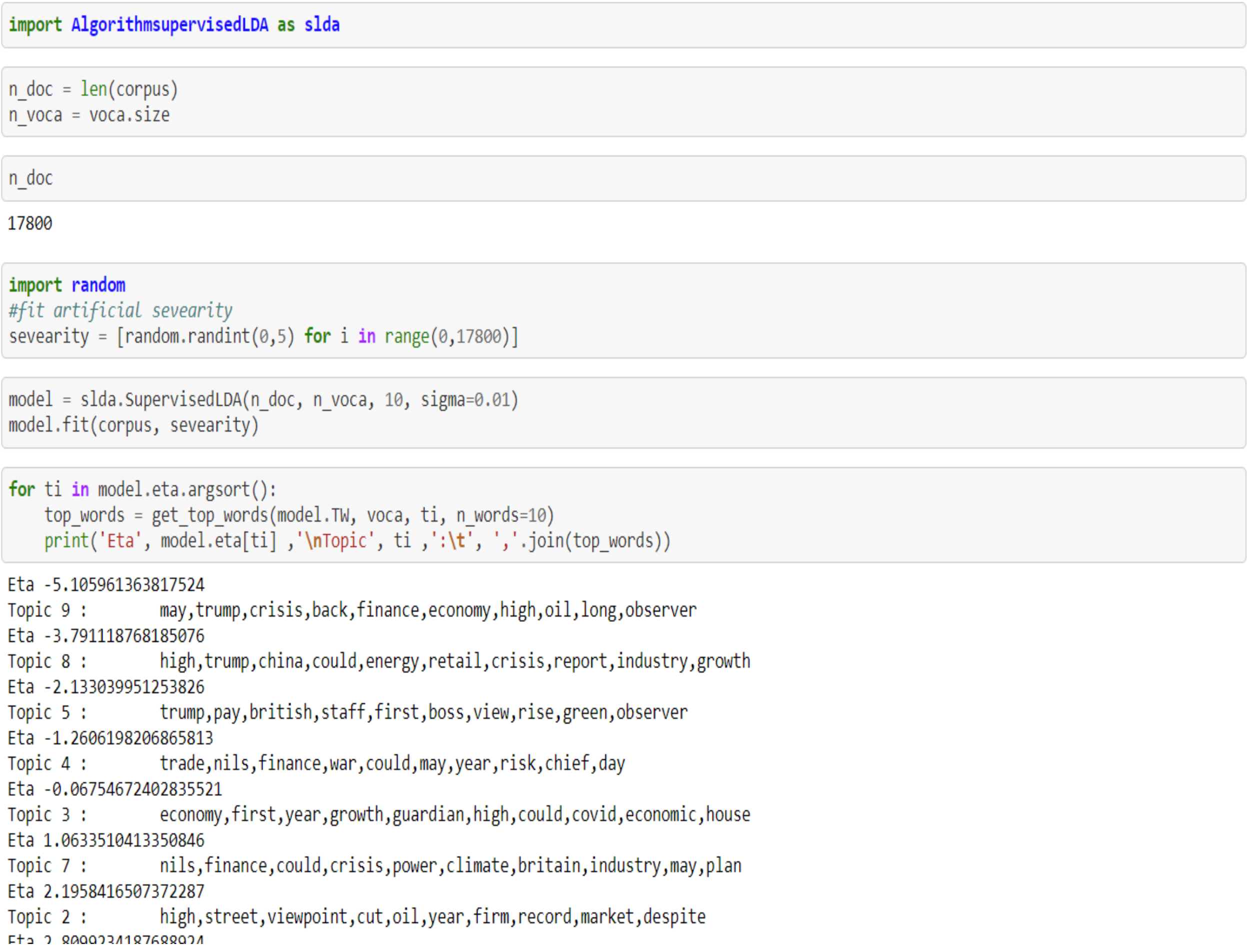
Supervised Latent Dirichlet Allocation (SLDA) is dedicated to model the terms in the documents. SLDA is a statistical model of labelled documents where each document is paired with a response. Through SLDA, we infer topic structure using a fitted model and then form its prediction. Prediction here is assessed through cross validation. Through SLDA, we can predict an observed univariate outcome and fit a topic model simultaneously. It adds a regression model above and beyond the LDA. We split the data to create a training data set with observed words and response variables. Through this, we can learn the global topics (beta) and regression coefficients (eta) for predicting the response variable (Y) in addition to topic proportions for each document (theta). To make predictions of Y given the learned beta and eta, we can define a new model where we do not observe Y and use the previously learned beta and eta.

Algorithm:



```
def fit(self, docs, responses, max_iter=100):
    """ Stochastic Expectation Maximisation algorithm
    """
    self.random_init(docs)
    for iteration in xrange(max_iter):

        for di in xrange(len(docs)):
            doc = docs[di]
            for wi in xrange(len(doc)):
                word = doc[wi]
                old_topic = self.topic_assignment[di][wi]

                self.TW[old_topic, word] -= 1
                self.sum_T[old_topic] -= 1
                self.DT[di, old_topic] -= 1

                z_bar = np.zeros([self.n_topic, self.n_topic]) + self.DT[di, :] + np.identity(self.n_topic)
                # this seems more straightforward than z_bar/z_bar.sum(1)
                z_bar /= self.DT[di, :].sum() + 1

                # update
                prob = (self.TW[:, word]) / (self.sum_T) * (self.DT[di, :]) * np.exp(
                    np.negative((responses[di] - np.dot(z_bar, self.eta)) ** 2) / 2 / self.sigma)

                new_topic = sampling_from_dist(prob)

                self.topic_assignment[di][wi] = new_topic
                self.TW[new_topic, word] += 1
                self.sum_T[new_topic] += 1
                self.DT[di, new_topic] += 1

        # estimate parameters
        z_bar = self.DT / self.DT.sum(1)[:, np.newaxis]  # DxK
        self.eta = solve(np.dot(z_bar.T, z_bar), np.dot(z_bar.T, responses))

        # compute mean absolute error
        mae = np.mean(np.abs(responses - np.dot(z_bar, self.eta)))
        #if self.verbose:
         #   logger.info('[ITER] %d,\tMAE:%.2f,\tlog_likelihood:%.2f', iteration, mae,
          #              self.log_likelihood(docs, responses))
```

Results:



```
import AlgorithmsupervisedLDA as slda
```

```
n_doc = len(corpus)
n_voca = voca.size
```

```
n_doc
```

```
17800
```

```
import random
#fit artificial seavarity
seavarity = [random.randint(0,5) for i in range(0,17800)]
```

```
model = slda.SupervisedLDA(n_doc, n_voca, 10, sigma=0.01)
model.fit(corpus, seavarity)
```

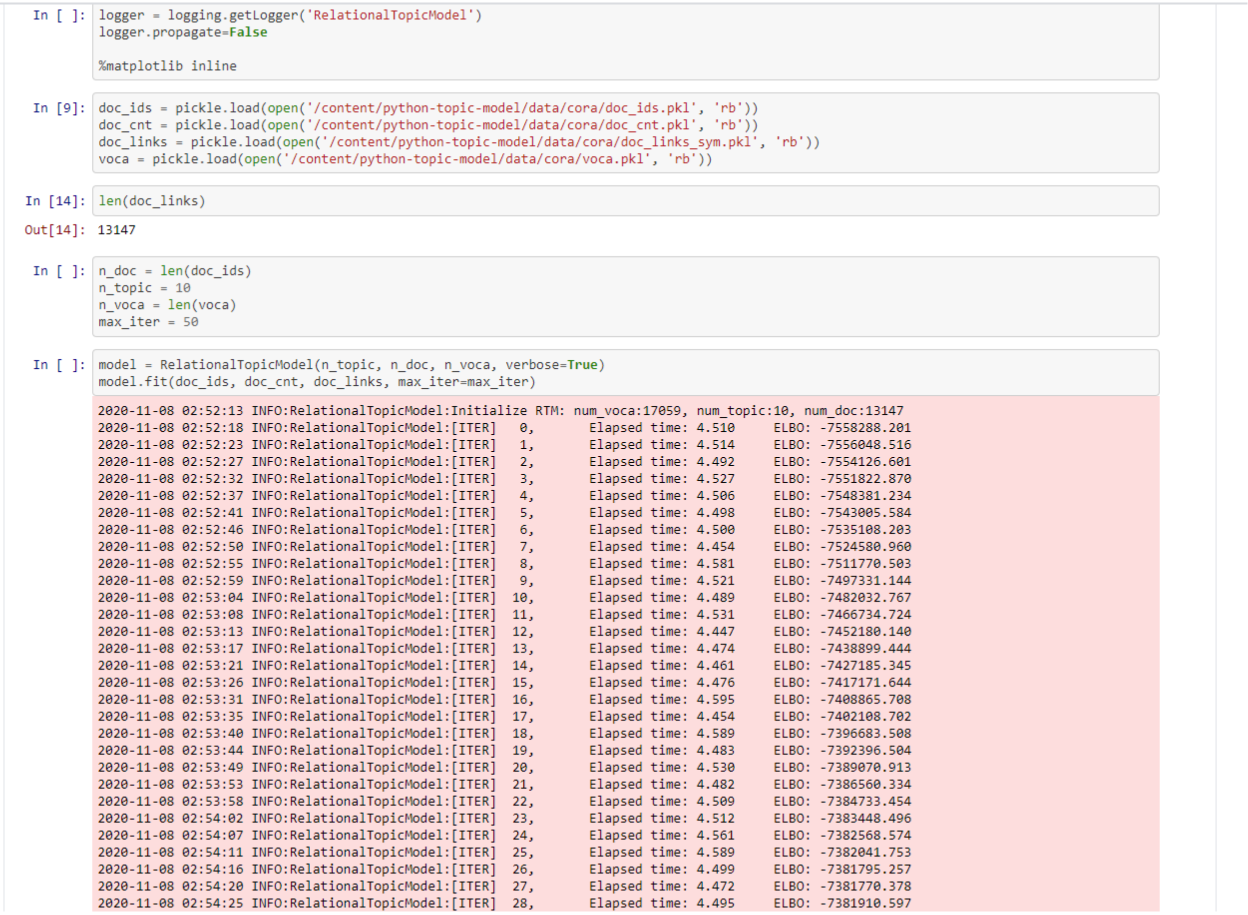
```
for ti in model.eta.argsort():
    top_words = get_top_words(model.TW, voca, ti, n_words=10)
    print('Eta', model.eta[ti] ,'\nTopic', ti ,':\t', ','.join(top_words))
```

```
Eta -5.105961363817524
Topic 9 :	may,trump,crisis,back,finance,economy,high,oil,long,observer
Eta -3.791118768185076
Topic 8 :	high,trump,china,could,energy,retail,crisis,report,industry,growth
Eta -2.133039951253826
Topic 5 :	trump,pay,british,staff,first,boss,view,rise,green,observer
Eta -1.2606198206865813
Topic 4 :	trade,nils,finance,war,could,may,year,risk,chief,day
Eta -0.06754672402835521
Topic 3 :	economy,first,year,growth,guardian,high,could,covid,economic,house
Eta 1.0633510413350846
Topic 7 :	nils,finance,could,crisis,power,climate,britain,industry,may,plan
Eta 2.1958416507372287
Topic 2 :	high,street,viewpoint,cut,oil,year,firm,record,market,despite
Eta 2.8099234187688924
```

8. RELATIONAL TOPIC MODELLING:

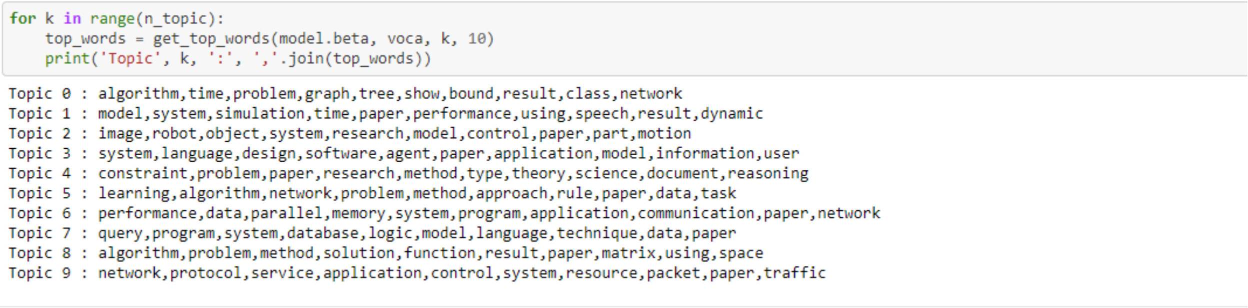
RTM is a model of data composed of documents, which are collection of terms and the links between the documents. It is based on LDA and hence each document is first generated from topics. The links between documents are then modeled as binary variables, one for each pair of documents. These are distributed according to a distribution that depends on the topics used to generate each of the constituent documents. In this way, the content of the documents is statistically connected to the link structure between them. Through relational topic modeling (RTM), we try to identify a relation between the documents we are working on. We model documents and identify the links between these documents and the contents within them. Relational topic modeling can be used to summarize a network of documents, predict the words within these documents and predict the links among these documents. With a fitted model, new data related predictions are made. The RTM is a new probabilistic generative model of documents and links between them. The RTM is used to analyze linked corpora such as citation networks, linked web pages, and social networks with user profiles.

Algorithm Execution:



```
In [ ]: logger = logging.getLogger('RelationalTopicModel')
        logger.propagate=False

        %matplotlib inline

In [9]: doc_ids = pickle.load(open('/content/python-topic-model/data/cora/doc_ids.pkl', 'rb'))
        doc_cnt = pickle.load(open('/content/python-topic-model/data/cora/doc_cnt.pkl', 'rb'))
        doc_links = pickle.load(open('/content/python-topic-model/data/cora/doc_links_sym.pkl', 'rb'))
        voca = pickle.load(open('/content/python-topic-model/data/cora/voca.pkl', 'rb'))

In [14]: len(doc_links)

Out[14]: 13147

In [ ]: n_doc = len(doc_ids)
        n_topic = 10
        n_voca = len(voca)
        max_iter = 50

In [ ]: model = RelationalTopicModel(n_topic, n_doc, n_voca, verbose=True)
        model.fit(doc_ids, doc_cnt, doc_links, max_iter=max_iter)

        2020-11-08 02:52:13 INFO:RelationalTopicModel:Initialize RTM: num_voca:17059, num_topic:10, num_doc:13147
        2020-11-08 02:52:18 INFO:RelationalTopicModel:[ITER]   0,       Elapsed time: 4.510     ELBO: -7558288.201
        2020-11-08 02:52:23 INFO:RelationalTopicModel:[ITER]   1,       Elapsed time: 4.514     ELBO: -7556048.516
        2020-11-08 02:52:27 INFO:RelationalTopicModel:[ITER]   2,       Elapsed time: 4.492     ELBO: -7554126.601
        2020-11-08 02:52:32 INFO:RelationalTopicModel:[ITER]   3,       Elapsed time: 4.527     ELBO: -7551822.870
        2020-11-08 02:52:37 INFO:RelationalTopicModel:[ITER]   4,       Elapsed time: 4.506     ELBO: -7548381.234
        2020-11-08 02:52:41 INFO:RelationalTopicModel:[ITER]   5,       Elapsed time: 4.498     ELBO: -7543005.584
        2020-11-08 02:52:46 INFO:RelationalTopicModel:[ITER]   6,       Elapsed time: 4.500     ELBO: -7535108.203
        2020-11-08 02:52:50 INFO:RelationalTopicModel:[ITER]   7,       Elapsed time: 4.454     ELBO: -7524580.960
        2020-11-08 02:52:55 INFO:RelationalTopicModel:[ITER]   8,       Elapsed time: 4.581     ELBO: -7511770.503
        2020-11-08 02:52:59 INFO:RelationalTopicModel:[ITER]   9,       Elapsed time: 4.521     ELBO: -7497331.144
        2020-11-08 02:53:04 INFO:RelationalTopicModel:[ITER]  10,       Elapsed time: 4.489     ELBO: -7482032.767
        2020-11-08 02:53:08 INFO:RelationalTopicModel:[ITER]  11,       Elapsed time: 4.531     ELBO: -7466734.724
        2020-11-08 02:53:13 INFO:RelationalTopicModel:[ITER]  12,       Elapsed time: 4.447     ELBO: -7452180.140
        2020-11-08 02:53:17 INFO:RelationalTopicModel:[ITER]  13,       Elapsed time: 4.474     ELBO: -7438899.444
        2020-11-08 02:53:21 INFO:RelationalTopicModel:[ITER]  14,       Elapsed time: 4.461     ELBO: -7427185.345
        2020-11-08 02:53:26 INFO:RelationalTopicModel:[ITER]  15,       Elapsed time: 4.476     ELBO: -7417171.644
        2020-11-08 02:53:31 INFO:RelationalTopicModel:[ITER]  16,       Elapsed time: 4.595     ELBO: -7408865.708
        2020-11-08 02:53:35 INFO:RelationalTopicModel:[ITER]  17,       Elapsed time: 4.454     ELBO: -7402108.702
        2020-11-08 02:53:40 INFO:RelationalTopicModel:[ITER]  18,       Elapsed time: 4.589     ELBO: -7396683.508
        2020-11-08 02:53:44 INFO:RelationalTopicModel:[ITER]  19,       Elapsed time: 4.483     ELBO: -7392396.504
        2020-11-08 02:53:49 INFO:RelationalTopicModel:[ITER]  20,       Elapsed time: 4.530     ELBO: -7389070.913
        2020-11-08 02:53:53 INFO:RelationalTopicModel:[ITER]  21,       Elapsed time: 4.482     ELBO: -7386560.334
        2020-11-08 02:53:58 INFO:RelationalTopicModel:[ITER]  22,       Elapsed time: 4.509     ELBO: -7384733.454
        2020-11-08 02:54:02 INFO:RelationalTopicModel:[ITER]  23,       Elapsed time: 4.512     ELBO: -7383448.496
        2020-11-08 02:54:07 INFO:RelationalTopicModel:[ITER]  24,       Elapsed time: 4.561     ELBO: -7382568.574
        2020-11-08 02:54:11 INFO:RelationalTopicModel:[ITER]  25,       Elapsed time: 4.589     ELBO: -7382041.753
        2020-11-08 02:54:16 INFO:RelationalTopicModel:[ITER]  26,       Elapsed time: 4.499     ELBO: -7381795.257
        2020-11-08 02:54:20 INFO:RelationalTopicModel:[ITER]  27,       Elapsed time: 4.472     ELBO: -7381770.378
        2020-11-08 02:54:25 INFO:RelationalTopicModel:[ITER]  28,       Elapsed time: 4.495     ELBO: -7381910.597
```

Results:

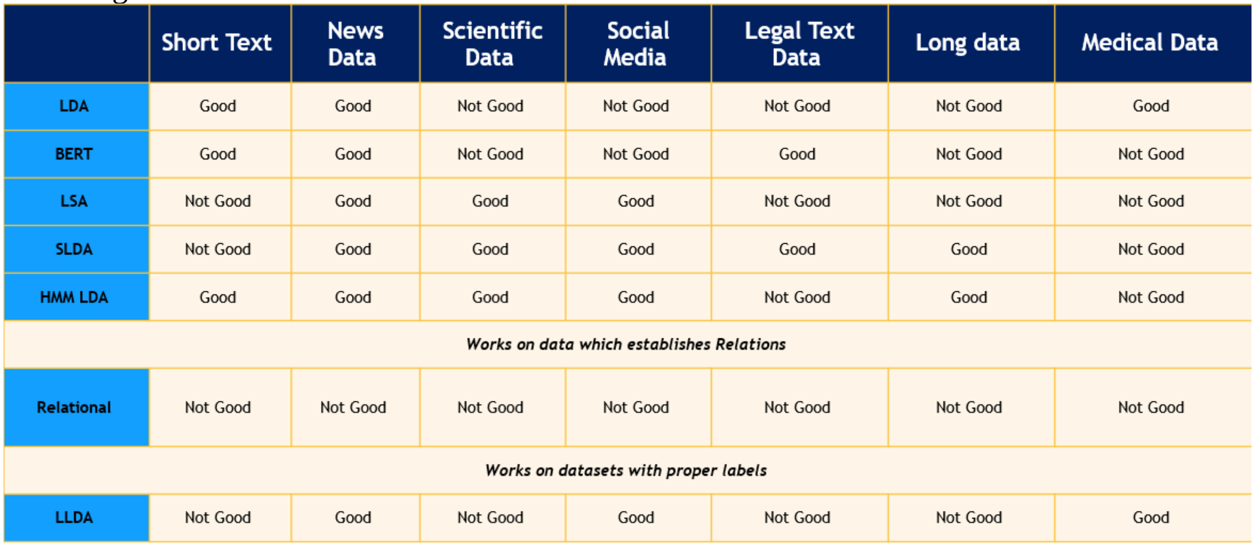


```
for k in range(n_topic):
    top_words = get_top_words(model.beta, voca, k, 10)
    print('Topic', k, ':', ','.join(top_words))

Topic 0 : algorithm,time,problem,graph,tree,show,bound,result,class,network
Topic 1 : model,system,simulation,time,paper,performance,using,speech,result,dynamic
Topic 2 : image,robot,object,system,research,model,control,paper,part,motion
Topic 3 : system,language,design,software,agent,paper,application,model,information,user
Topic 4 : constraint,problem,paper,research,method,type,theory,science,document,reasoning
Topic 5 : learning,algorithm,network,problem,method,approach,rule,paper,data,task
Topic 6 : performance,data,parallel,memory,system,program,application,communication,paper,network
Topic 7 : query,program,system,database,logic,model,language,technique,data,paper
Topic 8 : algorithm,problem,method,solution,function,result,paper,matrix,using,space
Topic 9 : network,protocol,service,application,control,system,resource,packet,paper,traffic
```

Repository: https://github.com/rajitakolla/INFO5082

Conclusion:

We ran the models for various data sets and tried to figure out which type of data best suits for each type of topic modelling. Due to the technical limitations, we can process only 500 records and evaluated the time complexity of algorithm. From our observations we can figure out the following



	Short Text	News Data	Scientific Data	Social Media	Legal Text Data	Long data	Medical Data
LDA	Good	Good	Not Good	Not Good	Not Good	Not Good	Good
BERT	Good	Good	Not Good	Not Good	Good	Not Good	Not Good
LSA	Not Good	Good	Good	Good	Not Good	Not Good	Not Good
SLDA	Not Good	Good	Good	Good	Good	Good	Not Good
HMM LDA	Good	Good	Good	Good	Not Good	Good	Not Good
Works on data which establishes Relations							
Relational	Not Good	Not Good	Not Good	Not Good	Not Good	Not Good	Not Good
Works on datasets with proper labels							
LLDA	Not Good	Good	Not Good	Good	Not Good	Not Good	Good

Limitations:

We analyzed the performance basing on the time complexity and the coherence score. But we need to build a quantitative measure for evaluating the metrics

Acknowledgement:

We would like to express our special thanks of gratitude to Prof. **Ding** for providing his invaluable guidance, comments, and suggestions throughout the course of the project. **We would like to continue working on the project and design quantitative measures to provide more generic insights to suggest best model**

References:

https://msaxton.github.io/topic-model-best-practices/project_overview.html
tidytextmining.com,
https://github.com/JoeZJH/Labeled-LDA-Python
https://github.com/markoarnauto/biterm
https://github.com/dongwookim-ml/python-topic-model
https://github.com/dongwookim-ml/python-topic-model/tree/master/notebook
https://github.com/dongwookim-ml/python-topic-model/tree/master/notebook
https://github.com/cemoody/lda2vec
https://arxiv.org/abs/1704.06879
https://github.com/MaartenGr/BERTopic