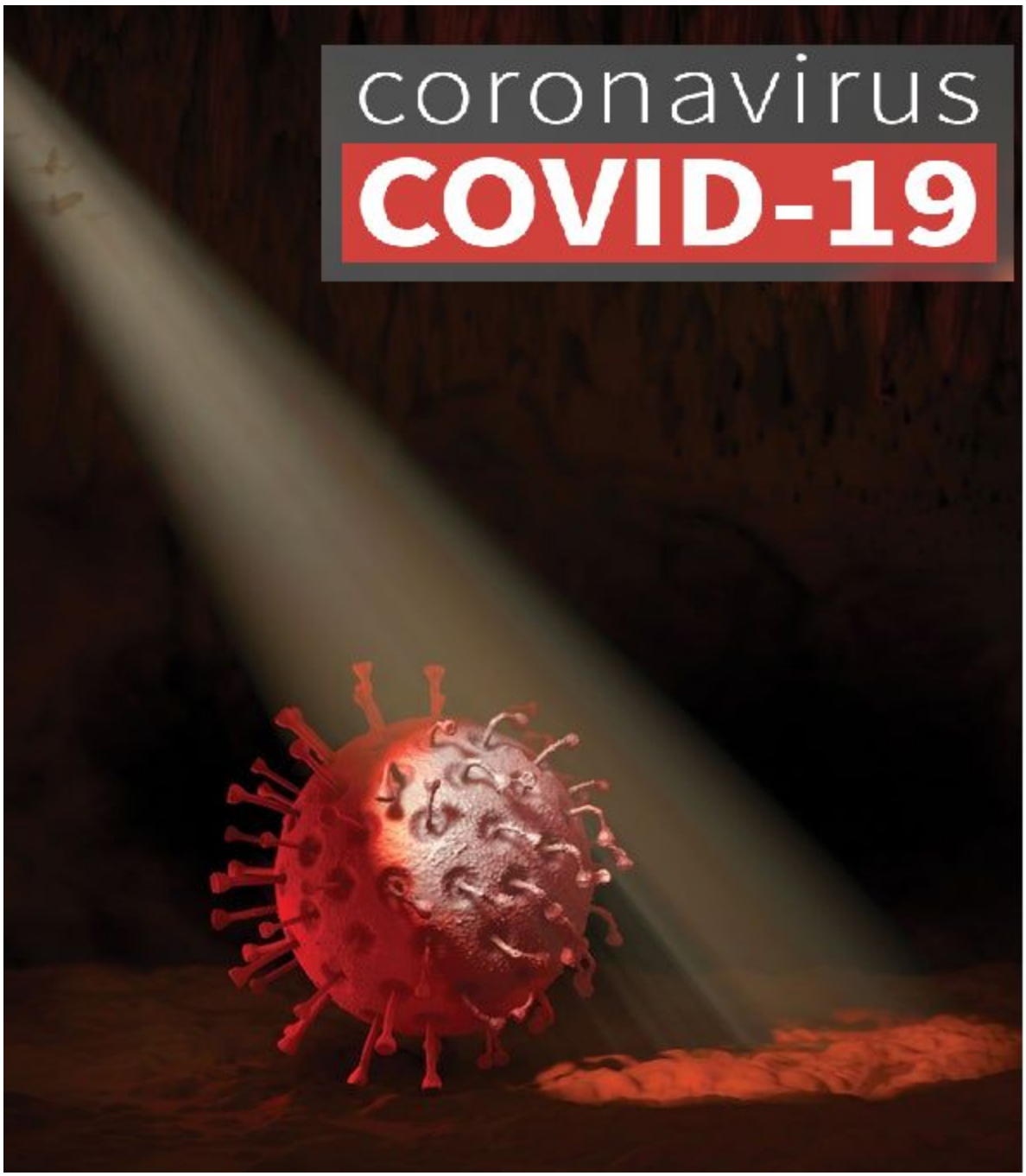
Capstone Project 2 - Final Project Report



Credit: Richard Borge

Literature Clustering of Covid-19 Related News Articles using NLP

Asha shravanthi Pidathala
Springboard Capstone Project 2
Jan 6th 2020 Cohort

Personal Statement

The World Health Organization declared that COVID-19 was a global pandemic, indicating significant global spread of an infectious disease. Coronaviruses are susceptible to mutation and recombination and are therefore highly diverse. The corona-like appearance of coronaviruses is caused by so-called spike glycoproteins, or peplomers, which are necessary for the viruses to enter host cells. There are about 40 different varieties and they mainly infect human and non-human mammals and birds. The virus that causes COVID-19 is thought to have originated in bats and then spread to snakes and pangolins and hence to humans, perhaps by contamination of meat from wild animals, as sold in China's meat markets. COVID-19 outbreak was first reported in Wuhan, China and has spread to more than 50 countries. WHO declared COVID-19 as a Public Health Emergency of International Concern (PHEIC) on March 11, 2020. Naturally, a rising infectious disease involves fast spreading, endangering the health of large numbers of people, and thus requires immediate actions to prevent the disease at the community level. Lot of people around the world are trying to find as much information and news on this topic to stay up to date and also help society with their research and developments. With this project we hope to cluster all the research/news articles related to covid-19 to simplify the search of related articles. Articles of highly similar topics will share a label and we can find meaning in the clusters by doing topic modeling to find keywords based on specific topics.

Dataset

The web is full of data. You will find it in different shapes and formats; simple tabular sheets, excel files, large and unstructured NoSql databases. The variety of content available in the form of texts, tweets, images, comments, vidoes, and news headlines is overwhelming and scraping the data from the web is not too reliable especially when it involves multiple urls at once. When disaster strikes, it is critically important to provide the most relevant information to first responders and the general public. During a disaster, people are inundated with a barrage of news sources, resulting in an environment of confusing and misinformation.

In this project, we have used **NewsAPI** to collect all the articles related to covid-19 and create the dataset for analysis. This project will show how to easily collect newsfeed from various data sources and apply machine learning algorithms on them to automatically extract the hidden topics. The NewsAPI is a simple and easy to use REST API that returns JSON metadata for headlines/articles currently published from over 30,000 news sources and blogs. The NewsAPI covers top publications including: ABC

News, Reuters, Associated Press, BBC News, Business Insider, CNN, Buzzfeed, Fox, ESPN, and so much more. The API also covers a huge range of countries around the world. The NewsAPI also has a free version available for all developers working on personal projects which can let you search articles upto a month old (limit 100 articles per day) and make 500 requests per day. In this project, we have scoured to find 3 different api keys to get as many articles as possible. Also over the course of the project, we could extract two sets of raw data again to make my dataset as big as possible.

Data Extraction

All the essential libraries were installed and imported necessary for the project. We will be using python with the necessary packages used in data science like Numpy, Pandas, Scipy or Scikit-learn. For the purpose of this project we also had to download external packages like:

- ➢ tqdm (a progress bar python utility)
- ➢ nltk (for natural language processing)
- ➢ bokeh (for interactive data viz)
- ➢ gensim
- ➢ pyldavis (python package to visualize lda topics)

To connect to the Newsapi service first we need to create an account at https://newsapi.org/register to get a key. Then we will have to put our key in the code and run the script. We managed to get three api keys using different email addresses and we can get only upto 30 days worth of articles with 100 articles maximum per day. A function was defined to extract news articles from the last 30 days. Using the function and from/to date lists, we extracted all news articles related to covid or coronavirus from each day. Using the parameters, we extracted all english news articles from all sources by setting the page size to maximum i.e; 100. Since we were extracting data using three different api keys, we set the ‘sort_by’ parameter different for each request i.e; to relevancy, publishedAt, and popularity to avoid duplicates. NewsAPI will return a JSON file and by exploring the documentation on the NewsAPI website and using .keys() function, we see that the data is arranged in three sections - dict_keys(['status', 'totalResults', 'articles']). We are interested in the ‘articles’ as this is where all our data is, so we normalize the json file and convert the ‘articles’ into a dataframe for further analysis and ease of manipulation. Three different data frames from three different requests were concatenated and saved as ‘raw_articles’. Over the course of this project we could extract two sets of raw data with a gap of 15 days, which we later merged to make my final raw dataset consisting of 18000 rows and 9 columns. The columns we get as part of the NewsAPI request are:

- ❏ source.id-The identifier id for the source this article came from.
- ❏ source.name-A display name for the source this article came from.
- ❏ author-The author of the article
- ❏ title-The headline or title of the article.
- ❏ description-A description or snippet from the article.
- ❏ url-The direct URL to the article.
- ❏ urlToImage-The URL to a relevant image for the article.
- ❏ publishedAt-The date and time that the article was published, in UTC (+000)
- ❏ content-The unformatted content of the article, where available. This is truncated to 200 chars.

18000 rows equals to 18000 articles that were extracted from the API. Let's have fun with the data !!!

Data Cleaning

Since the data was extracted from a NewsAPI, there was not much cleaning required as everything was very well organized and informative. We started with changing the datetime format to something more readable of the 'publishedAt' column. Counted the number of duplicate articles in the dataset based on the url subset. Since there was some overlap in the days when we extracted the articles, there was a good chunk of duplicate articles (6279 duplicates). After removing the duplicates we end up with a 11721 x 9 dataset. Further exploring the dataset for null values, we see that there are a lot of columns with some null values. However the content and description columns are the most important as we would be getting the majority of our texts for analysis from those columns. So we dropped the null value rows from those columns using dropna(). Now the shape of our final dataset is (11161,9).

For some feature engineering and EDA, we added the word count and unique words count columns for both content, description columns. This gave a rough estimate on how many words we have to play with for our topic modeling (which is not much as NewsAPI truncates the content column to 200 characters). NewsAPI provides 'sources' endpoint that allows you to map each data source to its category. Using the source.name column we requested and matched the source name to its category from the list of sources and added a 'category' to the dataframe. This will help visualize distribution of articles in our dataset. After cleaning the dataset and adding all the helpful columns, the final dataset was saved as 'processed_articles'. Now that the data has been collected, following analysis will be performed in this project:

- ❏ Have a look at the dataset and inspect it
- ❏ Apply some preprocessing on the texts: tokenization, tf-idf

- ❏ Cluster the articles using two different algorithms (Kmeans and LDA)
- ❏ Visualize the clusters using Bokeh and pyldavis

Data Storytelling

Further explored the processed dataset using .describe() which showed that the words counts of each column are very similar to their unique word counts. Even though there is not much text provided by the NewsAPI, this can be really helpful having such tight and good word content. Trying to visualize the distribution of articles based on category revealed that more than half the dataset had news articles from sources that were not listed in the newsapi sources endpoint.

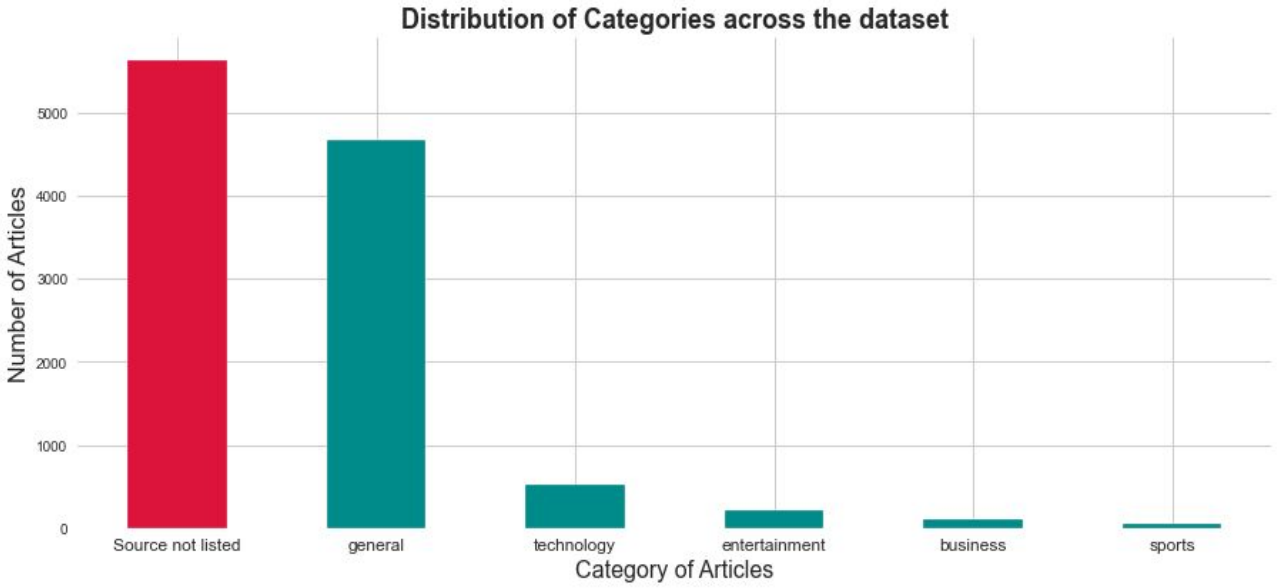


Figure 1: Distribution of articles based on category

As evident from the bar plot, most of the sources for the articles were not listed as part of the NewsAPI sources endpoint. Many mixed topics are included in the "general" category which is the largest category for our dataset. This only gives us a high level idea of our dataset, a very superficial classification of the news. It does not tell us the underlying topics or the most relevant news per each category. To analyze the articles deeper, we will have to process the descriptions and contents of each article since the words in the articles naturally carry more meaning.

The description column gives a snippet or description of the article and the content column gives the unformatted content of the article truncated to 200 characters. Using the word counts for content and description columns let’s check their distribution. From

figure 2, we can see the distribution of the word counts in both description and content columns which are our columns of interest for further analysis.

The description column has a wider range of word counts anywhere from 10 to 45 words. The description column length and word count is dependent on article to article as in what snippet from the article is provided by the NewsApi. Whereas the content column is already truncated to 200 characters by NewsApi and hence we see a narrow range of word counts anywhere between 30 to 40 words. The interesting and useful part is that the unique words count column for both description and content are almost similar to their respective word count columns. This means even though we have limited content due to restrictions of the free version of NewsApi, most of the content provided is useful which will help a lot in our further analysis of articles.

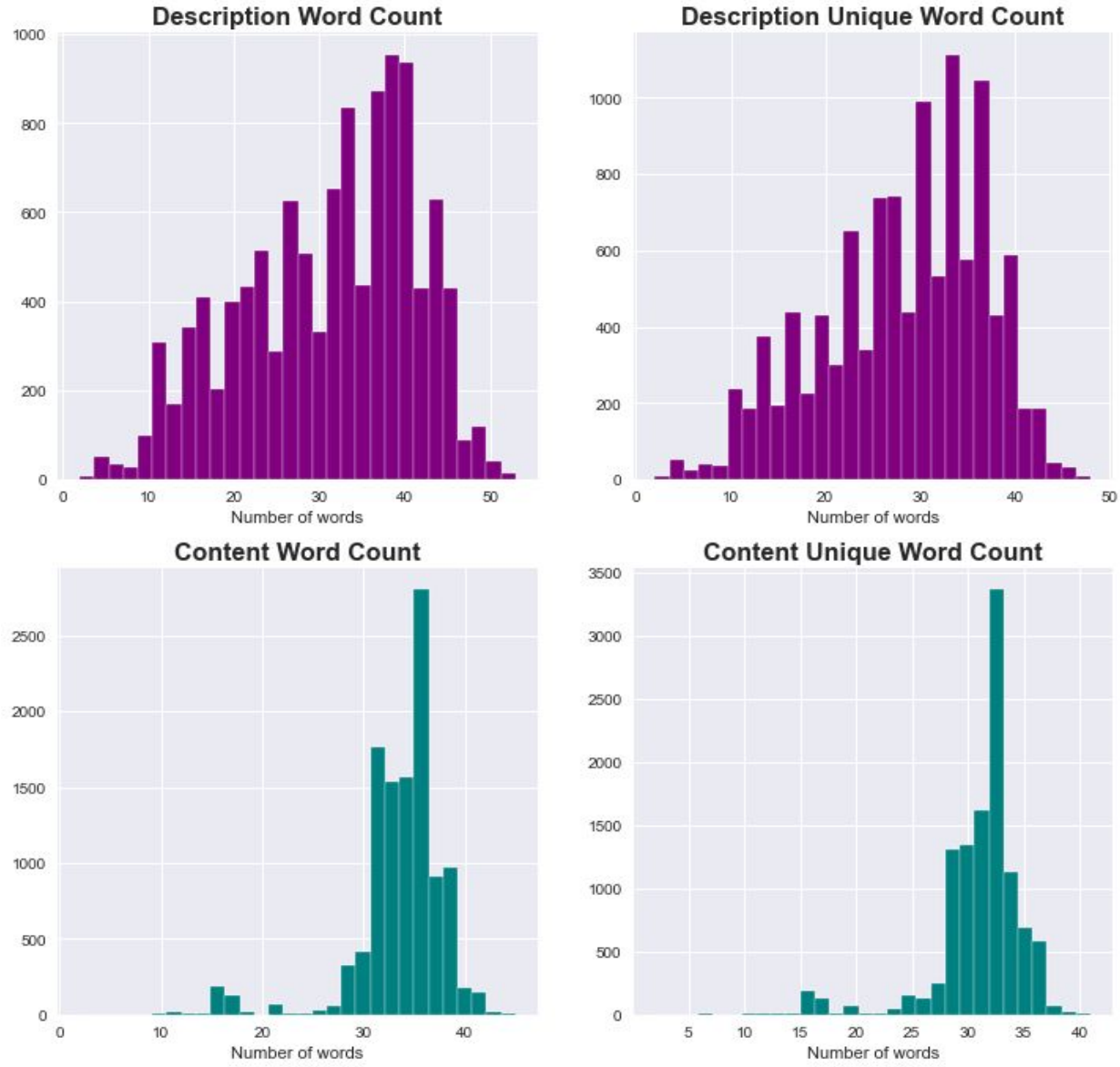


Figure 2: Distribution of the word counts for content and description columns

To make a little more sense of the data from the news and see some of the commonly occuring words in the dataset, we move onto WordClouds.

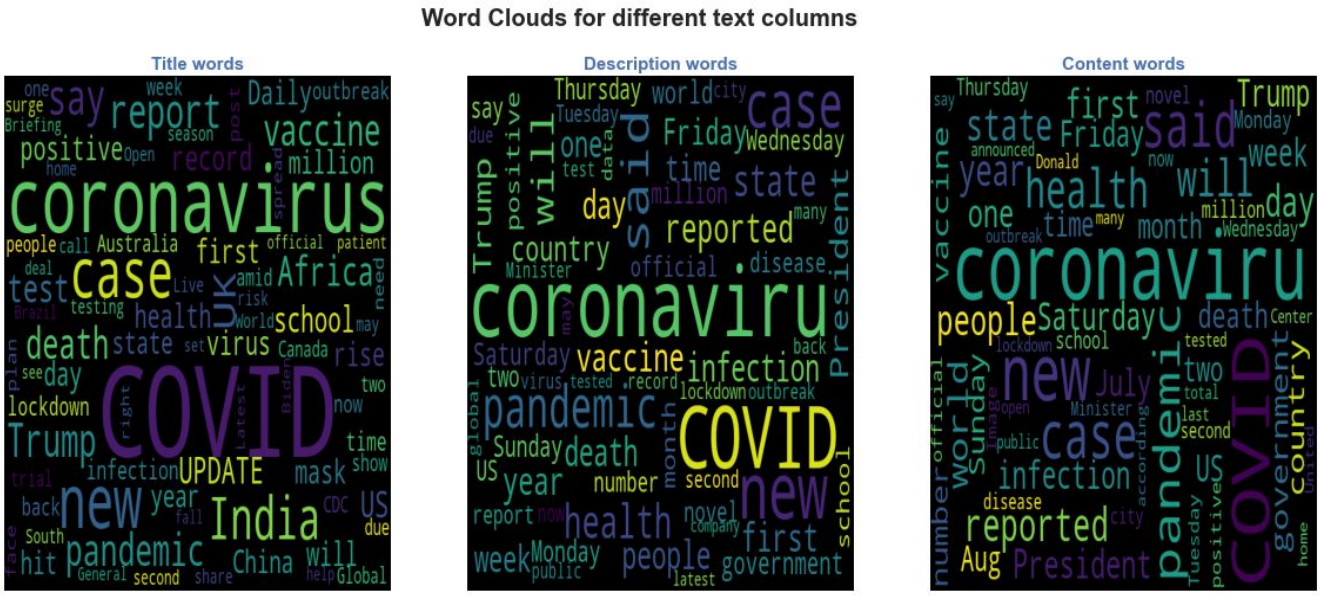


Figure 3: WordClouds for title, description, and content columns text

From the above WordClouds of different text columns from the dataset we see that as expected all articles are related to covid or coronavirus or pandemic. We can also see names of different countries in the title indicating articles covering world updates. Also possible articles about vaccines and political leaders. Now that we have an idea of what kind of articles we have in the dataset, let's dig deeper by some text pre-processing steps.

Data Pre-processing and Tokenization

All the essential libraries pertaining to nltk and sklearn feature extraction were imported. Since we do not have full articles provided by the NewsAPI and to make maximum use of the text columns we had, we combined the content, title, and description columns into one string and added it as ‘combined_text’ column which will serve as our full article text in preparation for tokenization. Building a tokenizer will break the combined_text into sentences and then break the sentences into tokens, also remove punctuations and stopwords. We downloaded the nltk’s stopwords list and also prepared a list of my own stopwords which included some unimportant words that would have not helped in topic modeling. Especially since all articles are related covid, we included the words like covid, coronavirus, and pandemic as part of my stopwords to reduce the impact of those when we extract keywords from the articles and do topic modeling. Additionally we added a couple of more cleaning functions like any words of 2 or less characters to be

removed as it would not help much in understanding the articles. Also removing any non ascii characters standardize the text (eg: i'm -- i am). This will make the tokenization process more efficient. Finally we defined a tokenize function where we integrated all the functions created for cleaning. By applying the tokenize function using a map method to the combined_text column, we created a new column 'tokens' that contained all the keywords for each article. Following is what the tokenization looks like for the first few rows of combined_text column:

combined_text: Image copyrightReutersImage caption
Abhishek (L) said his wife and daughter would self-isolate at home
Three generations of a high-profile Bollywood family have tested positive for Covid-19, offic… [+3432 chars]Coronavirus: Three generations of Bollywood Bachchan family infectedActress Aishwarya Rai Bachchan, her father-in-law, husband and daughter test positive for Covid-19.
tokens: ['abhishek', 'said', 'wife', 'daughter', 'would', 'self', 'isolate', 'home', 'three', 'generations', 'high', 'profile', 'bollywood', 'family', 'tested', 'positive', 'offic', 'three', 'generations', 'bollywood', 'bachchan', 'family', 'infectedactress', 'aishwarya', 'rai', 'bachchan', 'father', 'law', 'husband', 'daughter', 'test', 'positive']

combined_text: Image copyrightAFPImage caption
This is the second ban on alcohol sales since South Africa's outbreak began
South Africa has introduced new restrictions, including another ban on alcohol sales, to… [+2226 chars]Coronavirus: South Africa bans alcohol sales again to combat Covid-19It is one of several restrictions introduced by President Ramaphosa amid rising infection rates.
tokens: ['second', 'ban', 'alcohol', 'sales', 'since', 'south', 'africa', 'outbreak', 'began', 'south', 'africa', 'introduced', 'new', 'restrictions', 'including', 'another', 'ban', 'alcohol', 'sales', 'south', 'africa', 'bans', 'alcohol', 'sales', 'combat', 'one', 'several', 'restrictions', 'introduced', 'president', 'ramaphosa', 'amid', 'rising', 'infection', 'rates']

combined_text: Florida broke the nations covid-19 single-day case record on Sunday, reporting 15,299 new infections, the most new cases ever reported by a state during the pandemic. The news underscores the raging … [+2909 chars]Florida Records 15,000 New Covid-19 Cases in a Day, the Most for Any State During the PandemicFlorida broke the nation's covid-19 single-day case record on Sunday, reporting 15,299 new infections, the most new cases ever reported by a state during the pandemic. The news underscores the raging state of the coronavirus crisis in the U.S., proving once a…
tokens: ['florida', 'broke', 'nations', 'single', 'day', 'case', 'record', 'sunday', 'reporting', 'new', 'infections', 'new', 'cases', 'ever', 'reported', 'state', 'news', 'underscores', 'raging', 'florida', 'records', 'new', 'cases', 'day', 'state', 'pandemicflorida', 'broke', 'nation', 'single', 'day', 'case', 'record', 'sunday', 'reporting', 'new', 'infections', 'new', 'cases', 'ever', 'reported', 'state', 'news', 'underscores', 'raging', 'state', 'crisis', 'proving']

Keywords are the most relevant words that are associated with a particular piece of content. The text below shows the top 10 keywords from the tokens column relevant to each category.

category : sports
top 10 keywords: [('players', 22), ('transfer', 19), ('league', 17), ('season', 15), ('open', 15), ('talk', 15), ('first', 13), ('team', 12), ('nfl', 12), ('positive', 11)]

category : general
top 10 keywords: [('cases', 2634), ('new', 2618), ('said', 2413), ('reported', 1263), ('health', 1185), ('infections', 962), ('country', 873), ('saturday', 871), ('friday', 786), ('government', 781)]

category : technology
top 10 keywords: [('new', 219), ('company', 165), ('startup', 120), ('today', 104), ('one', 103), ('year', 86), ('announced', 83), ('world', 82), ('people', 81), ('first', 80)]

category : entertainment
top 10 keywords: [('trump', 81), ('new', 64), ('face', 49), ('like', 45), ('masks', 43), ('many', 41), ('one', 38), ('mask', 35), ('world', 34), ('would', 34)]

category : business
top 10 keywords: [('new', 66), ('cases', 58), ('people', 45), ('trump', 45), ('reported', 38), ('vaccine', 33), ('said', 29), ('news', 27), ('president', 26), ('healthcare', 25)]

category : Source not listed
top 10 keywords: [('new', 2502), ('cases', 1636), ('health', 1139), ('news', 1116), ('trump', 868), ('people', 863), ('first', 808), ('one', 739), ('said', 714), ('vaccine', 634)]

Looking at the above lists of top 10 keywords, we can formulate some hypotheses. Like the sports category has articles relating to nlf, transfer, players and league etc. Technology category has articles related to covid covering startups and companies etc., whereas the business category has trump, healthcare keywords etc. pointing in the right direction. Other generic categories include articles about vaccines, health, and cases etc.. Even after pre-processing the text and removing the stop words, we still see a lot of generic words like year, new, us, and people that don't carry much meaning. So we will be using tf-idf as our next step to prevent a similar situation.

Text Pre-processing and tf-idf

Tf-idf stands for term frequency-inverse document frequency. The tf-idf a of a term t in a document d is proportional to the number of times the word t appears in the document d but is also offset by the frequency of the term t in the collection of the documents of the

corpus. It's a numerical statistic intended to reflect how important a word is to a document or a corpus (a collection of documents). In this project, words correspond to tokens and documents correspond to combined_texts. A corpus is therefore a collection of combined_texts. tf-idf acts therefore as a weighting scheme to extract relevant words in a document. Some words appear more frequently in general and don't especially carry a meaning.

Computing the tfidf matrix is done using the TfidfVectorizer method from scikit-learn. After applying the fit_transform() method on the tokens columns, we can print the shape of the tfidf_matrix (11161, 13799) where the number of rows is total numbers of articles/documents and number of columns is the total number of unique terms (tokens) across the documents. Creating a dictionary mapping the tokens to their tf-idf scores we can visualize the distribution of the tf-idf scores through a histogram.

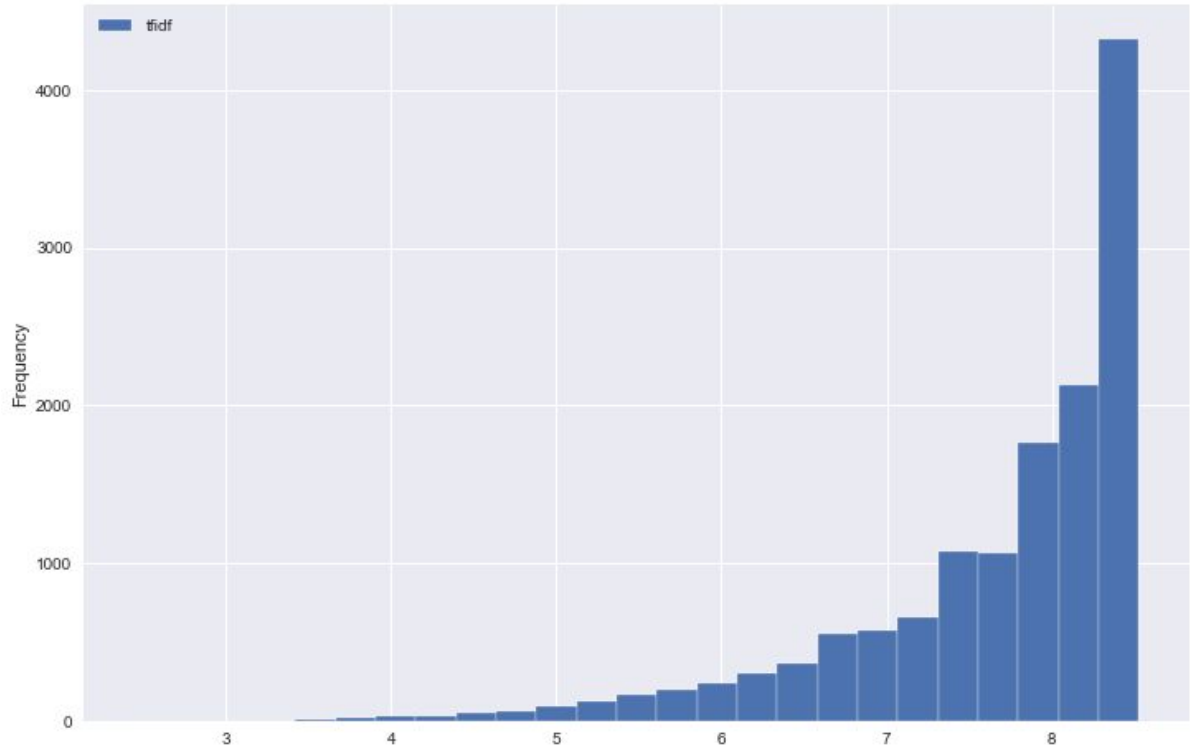
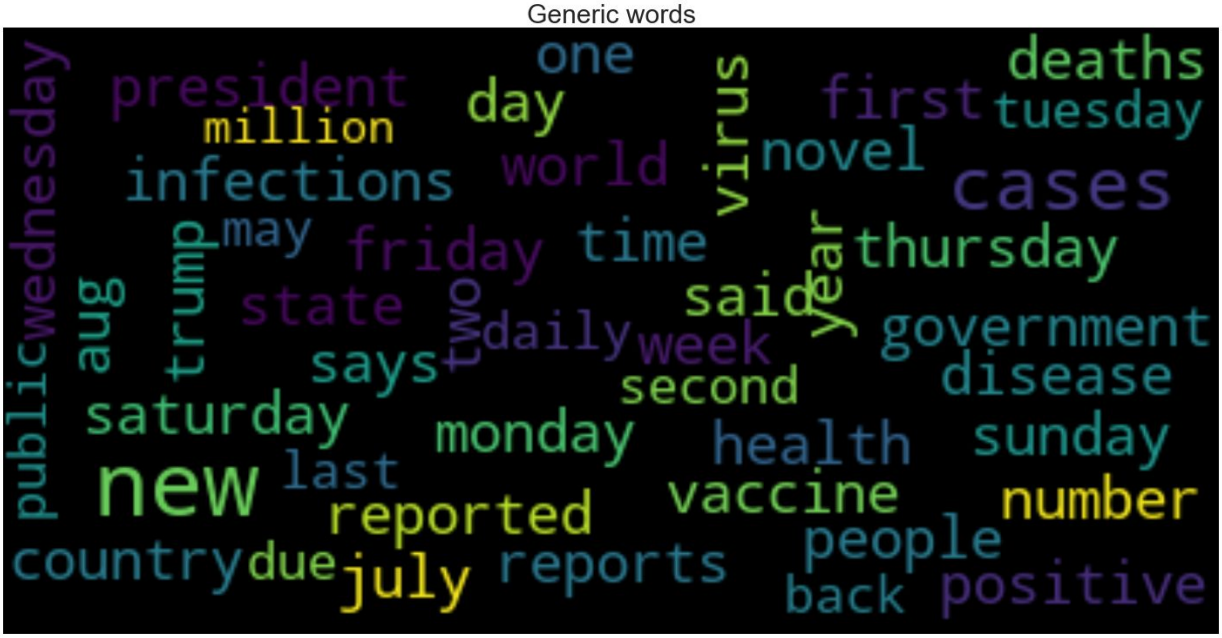


Figure 4: tf-idf scores of all tokens

The tokens with lowest tf-idf scores will give us very generic words and those with highest scores will be the important words that carry more meaning in outlining the underlying topics. Below are the WordClouds for the 50 tokens with lowest (generic words) and highest (Important words) scores. From figure 5 we can see that the lowest tf-idf scores give us very generic words as these are the most common words found

across all combined_text. Words like people, day, week, cases, deaths, virus etc. got a low score as a penalty for not being relevant.

Generic words



Important words

Figure 5: WordClouds from tf-idf scores

KMeans Clustering

As we have noticed before from our tfidf_matrix shape, the documents have close to 15000 features i.e; each document has close to 15000 dimensions. By applying

Principle Component Analysis (PCA) to our vectorized data we can reduce the dimensions while still keeping 95% variance. By keeping the large number of dimensions with PCA, you don't destroy much of the information but hopefully will remove some noise/outliers from the data, and make the clustering problem easier for k-means. Using fit_transform() we reduced the vectorized data and saved it as X_reduced. This will only be used for k-means and later in the project t-SNE will still use the original feature vector X that was generated through tf-idf on the NLP processed text.

Given the number of clusters, k, k-means will categorize each vector by taking the mean distance to a randomly initialized centroid. The centroids are updated iteratively. To find the best k value for k-means we'll look at the distortion and silhouette score at different k values. Distortion computes the sum of squared distances from each point to its assigned center. When distortion is plotted against k there will be a k value after which decreases in distortion are minimal. This number is usually determined by trying out different values until the result satisfies a given metrics (silhouette score or distortion). The optimal number is the one that maximizes the silhouette score and minimizes the distortion.

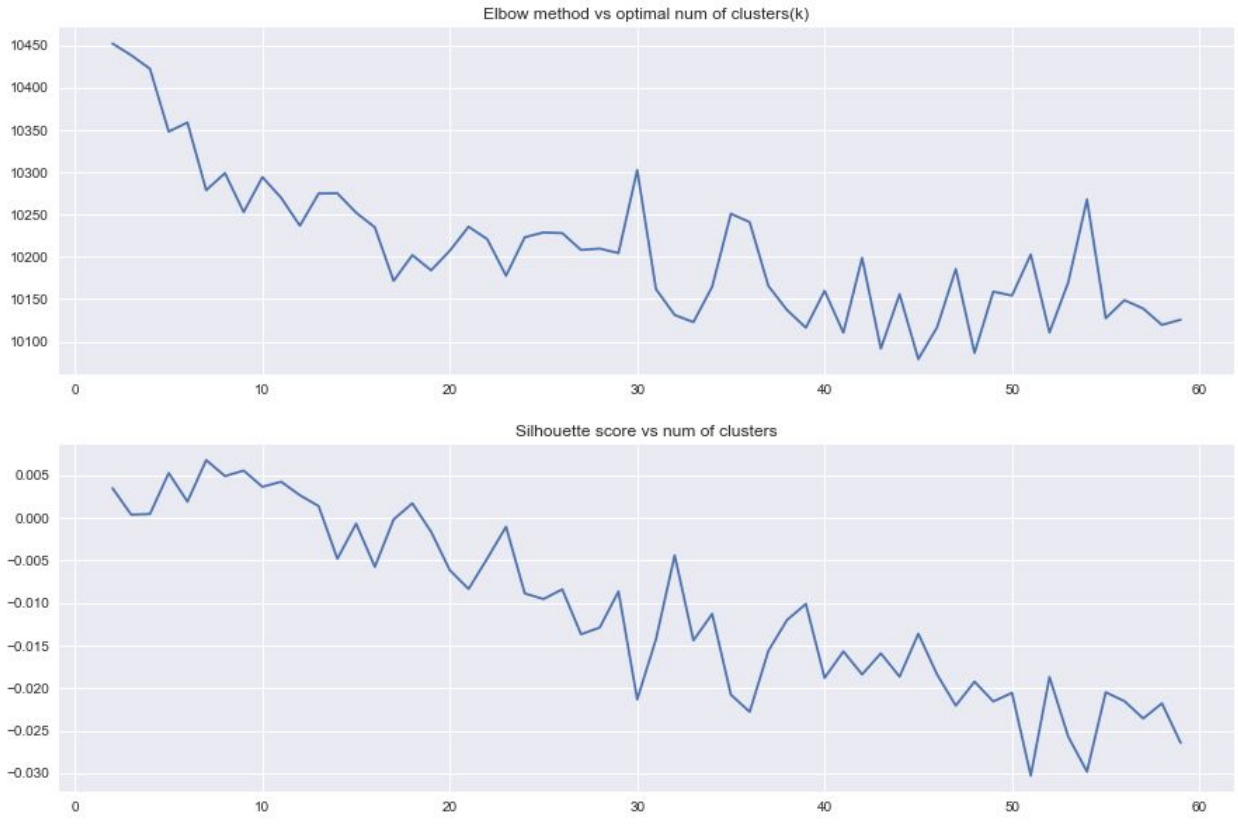


Figure 6: Elbow method and Silhouette score for a range of k values

From the above plot we can see that k value of 20 will be a good choice.The decrease in distortion is not as significant after 20 clusters and taking the inertias and silhouette score in consideration, we will use k=20 in this project. Using MiniBatchKMeans and fitting the model to the X_reduced dataset, we predicted the clusters for the dataset. Looking up the hot keywords that describe each cluster, it is very evident that there are a lot of similar keywords appearing in multiple clusters indicating that the clusters are not well separated.

	keyword_0	keyword_1	keyword_2	keyword_3	keyword_4	keyword_5	keyword_6	keyword_7	keyword_8	keyword_9
cluster_0	advocates	aggregate	abc news	announced tuesday	active	agent	adds new	across europe	adopted euro	ami
cluster_1	aaron	abhishek bachchan	abhishek	abbott	abuse	abandoned	abc news	accelerated	according official	aberdeen
cluster_2	abbott	abc news	abbott laboratories	abattoir	abating	abrupt	abhishek	abandon	ability	able
cluster_3	abbott	aberdeen	abating	abattoir	ability	abhishek bachchan	abrupt	abandon	abroad	ababa
cluster_4	aaron	absolutely	abroad	abbott	abuja	abandoned	abrupt	acce	abhishek bachchan	abruptly
cluster_5	aaron	absolutely	abrupt	abroad	abuja	abu	acceptance	activated	active case	abu dhabi
cluster_6	abbott laboratories	asuncion	arts	camp	ahead new	agency said	biotech firm	billions	avoid public	aug took
cluster_7	abandoned	abbott	aberdeen	abuse	ability	abroad	absent	accept	abu	accepting
cluster_8	ability	acce	adding	adds details	acute respiratory	absent	among younger	alone	bag	aaron
cluster_9	ababa	abandoned	abc news	abbott	ability	able	acc	alex castro	according research	american airlines
cluster_10	aaron	aboard	across nation	activity	accelerated	across canada	across several	according new	abrupt	absent
cluster_11	abbott	abc news	accepted	accelerating	accidental	acceptance	access information	americans	addis ababa	act
cluster_12	according new	accept	accidental	abruptly	according state	according statement	adams	afford	administered	accepted

Figure 7: Keywords from kmeans clusters

Since all the articles are based on covid or coronavirus, it is difficult to differentiate topics based on the top 10 keywords and we see that some keywords repeat in most of the clusters. However there is still some clustering that is noticable like:
Cluster 0 : Based on international news covering europe and euro;
Cluster 18: Academia probably schools and universities;
Cluster 6: Abbott Laboratories, biotech industry
Apart from these looks like the Abhishek bacchan and the Bacchan family covid case was covered by various sources and see that commonly across all clusters. Descriptions may in reality be characterized by a "mixture" of topics. To get better separation of articles based on topics, LDA will be used further. To plot our clusters, we did dimensionality reduction using t-SNE. Using t-SNE we can reduce our high dimensional features vector to 2 dimensions as x,y coordinates and plot the clusters based on combined_text. Plotting the TSNE fit dataset, we can visualize the clusters.

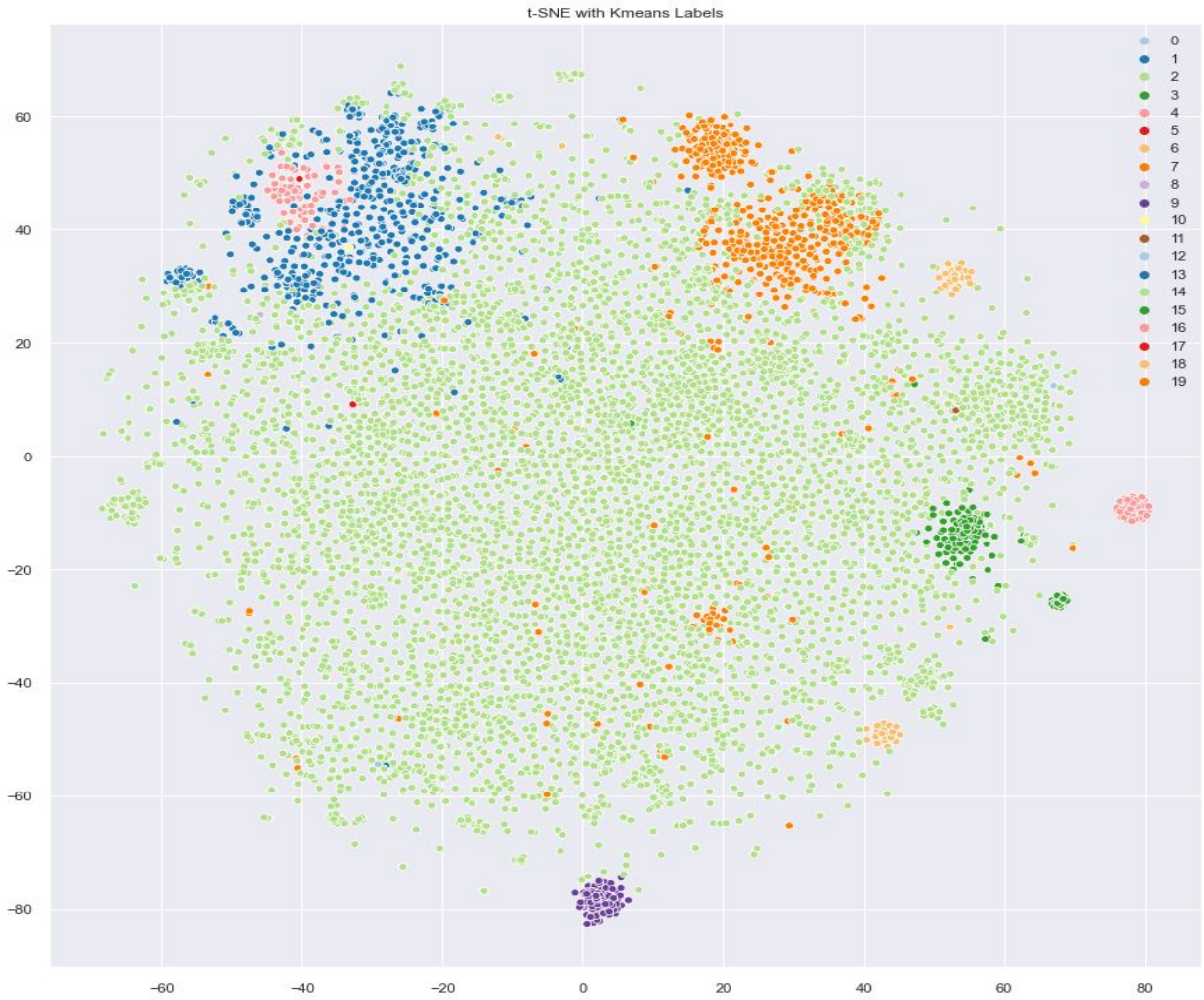


Figure 8: 2D scatter plot of articles showing the kmeans generated clusters

The labeled plot gives better insight into how the articles are grouped. The location of each article on the plot was determined by t-SNE while the label-color was determined by k-means. If we look at the plot where t-SNE has grouped articles into clusters, it shows that k-means was uniform in labeling that cluster (same color groups). This behavior shows that structure within the data can be observed and measured to some extent. However there are still a large number of articles that are spread out on the plot. This is a result of t-SNE and k-means finding different connections in the higher dimensional data. The topics of these articles often intersect so it is hard to cleanly separate them, especially in our case since all the articles are related to covid and many sources could cover similar news pertaining to a single topic. We can further use LDA for topic modeling to find the most significant words in each cluster.

LDA

LDA stands for Latent Dirichlet Allocation. LDA importantly is based on two things, each document in a corpus is a weighted combination of several topics and each topic has its collection of representative keywords. The main hyperparameter that we have to correctly identify in the LDA models is the number of topics. Just like we did with Kmeans, we will select this optimal number using some metrics. Given the number of topics, LDA starts shuffling the topic distribution in each document and the word distribution in each topic until the final results shows a high segregation of topics.

To perform LDA, we have used gensim to create a corpus and a dictionary from the tokens column of the processed dataset. Started with defining the LDA model in function that takes up the number of topics as a parameter, and then defining a metric to assess a good topic model: the coherence score. Also defined a function to display topics and corresponding keywords. Like Kmeans, we vary the number of topics in an interval and pick the number that optimizes the coherence score.

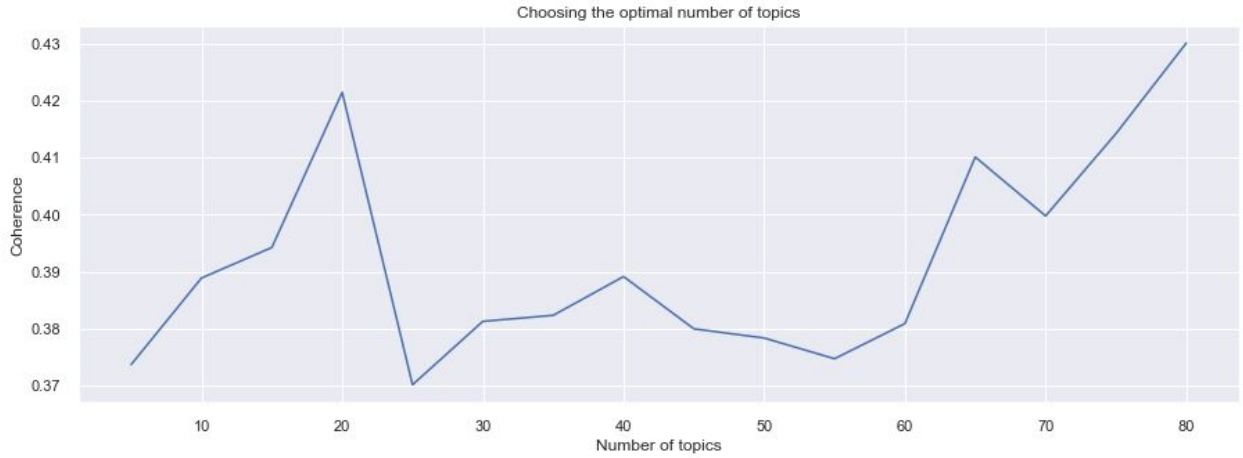


Figure 9: Coherence score for a range of topic numbers

We are quite consistent with the KMeans approach where we chose 20 clusters based on elbow method and silhouette score. 20 seems to be a good number of topics, since it's a value after which the coherence stops increasing rapidly. Now using the functions defined to generate the LDA model, let’s display the 20 topics and its associated keywords. From figure 10, it is quite evident that LDA did a very good job separating the articles based on topic. These clusters/topics look more distinct from one another like:
Topic 1: Tech industry;
Topic 4: Travel restrictions and lockdown;
Topic 5: Politics, Trump;
Topic 7: Sports;

Topic 10: Schools and academia;
Topic 13: German travel;
Topic 19: General covid updates.

	keyword_1	keyword_2	keyword_3	keyword_4	keyword_5	keyword_6	keyword_7	keyword_8	keyword_9	keyword_10
topic_0	health	public	new	people	officials	zealand	england	general	testing	says
topic_1	company	new	one	today	years	apple	time	facebook	last	year
topic_2	first	aug	make	cost	take	turn	power	years	update	half
topic_3	shares	find	australia	virus	stocks	reopen	economy	fresh	closed	could
topic_4	said	world	cases	million	new	cdc	restrictions	country	friday	lockdown
topic_5	trump	president	president_donald	night	cnn	administration	white_house	would	donald_trump	news
topic_6	open	api	park	accused	slow	set	face	little	process	cloud
topic_7	year	season	next	series	play	first	team	due	fans	players
topic_8	democratic	international	security	vote	research	work	voters	like	california	sign
topic_9	disease	sars_cov	patients	business	study	small	buy	government	people	treatment
topic_10	school	students	convention	back	women	way	return	person	online	fall
topic_11	first	test	million	biden	positive	joe_biden	said	tests	campaign	course
topic_12	children	people	schools	one	kids	family	home	many	hospital	say
topic_13	travel	germany	virtual	german	face_masks	people	mask	berlin	conference	stop
topic_14	oil	trade	questions	court	welcome	title	tax	manager	green	difficult
topic_15	new	state	said	cases	city	country	day	prime_minister	outbreak	lockdown
topic_16	university	california	researchers	show	beat	meet	program	texas	also	major
topic_17	vaccine	news	vaccines	potential	canada	companies	china	trials	company	deal
topic_18	said	right	call	need_know	democrats	party	chief	would	parties	economic
topic_19	new	cases	reported	deaths	health	daily	infections	reports	confirmed	day

Figure 10: Topics and associated keywords generated using LDA

Now that we know 20 topics is a good choice and LDA does a good job separating the topics; let's go ahead and build a document/topic matrix and use Bokeh plot to visualize.

LDA outputs a distribution of topics for each document. We'll assume that a document's topic is the one with the highest probability. Using TSNE to make a 2D plot of the topic clusters, we generated a scatter plot using Bokeh. From figure 11, we can see that LDA performed better than KMeans in modeling the clusters around topics. With Bokeh, we assumed that a document's topic is the one that has the highest probability. But this way we might have lost information as we did not consider the topic distribution/mixture as discussed before.

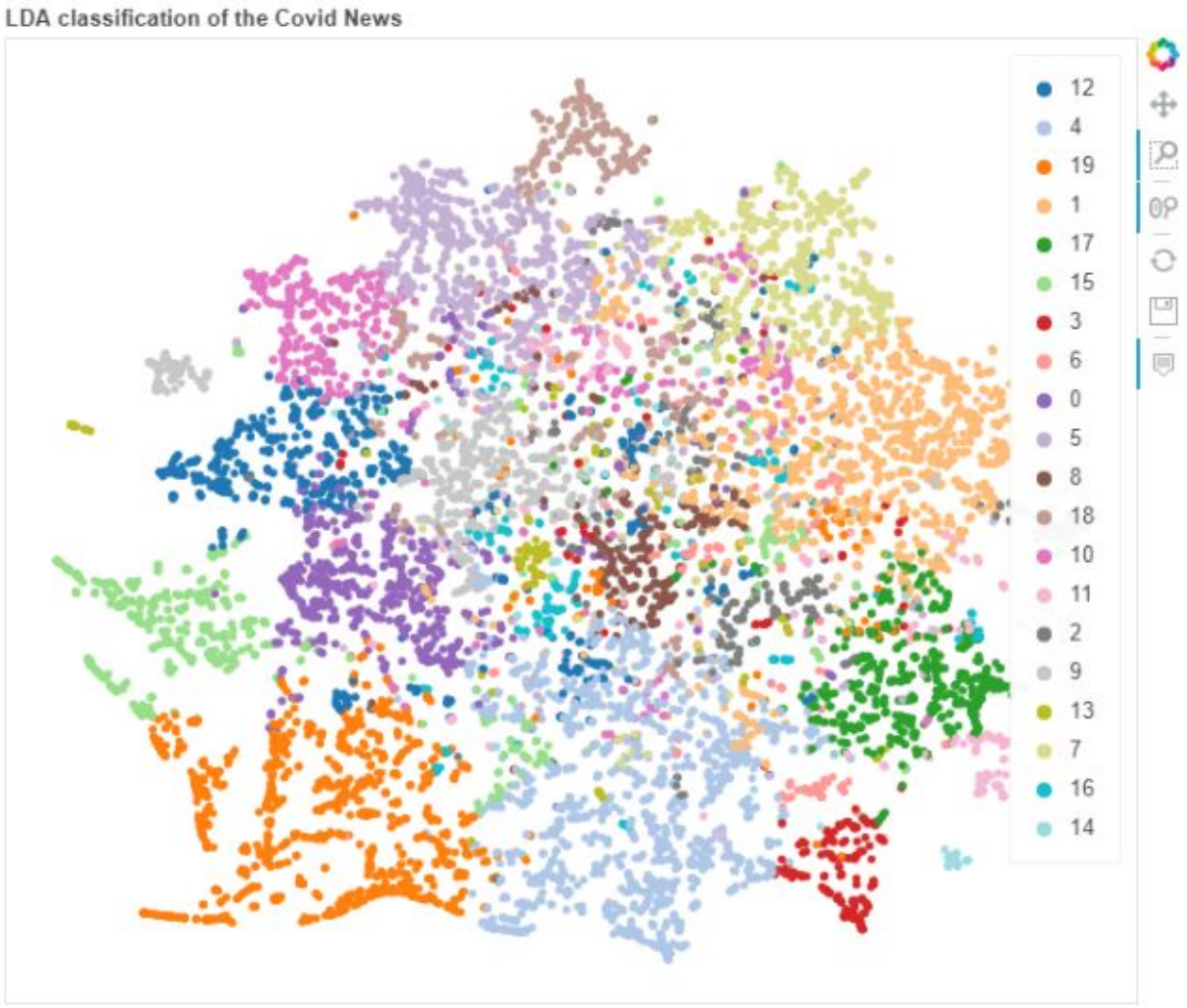


Figure 11: Bokeh scatter plot of topic clusters

Hopefully pyLDAvis (visualization package) will help us solve this problem of topic mixtures. Using gensim we generated an interactive plot to visually see what topics does each cluster entail.

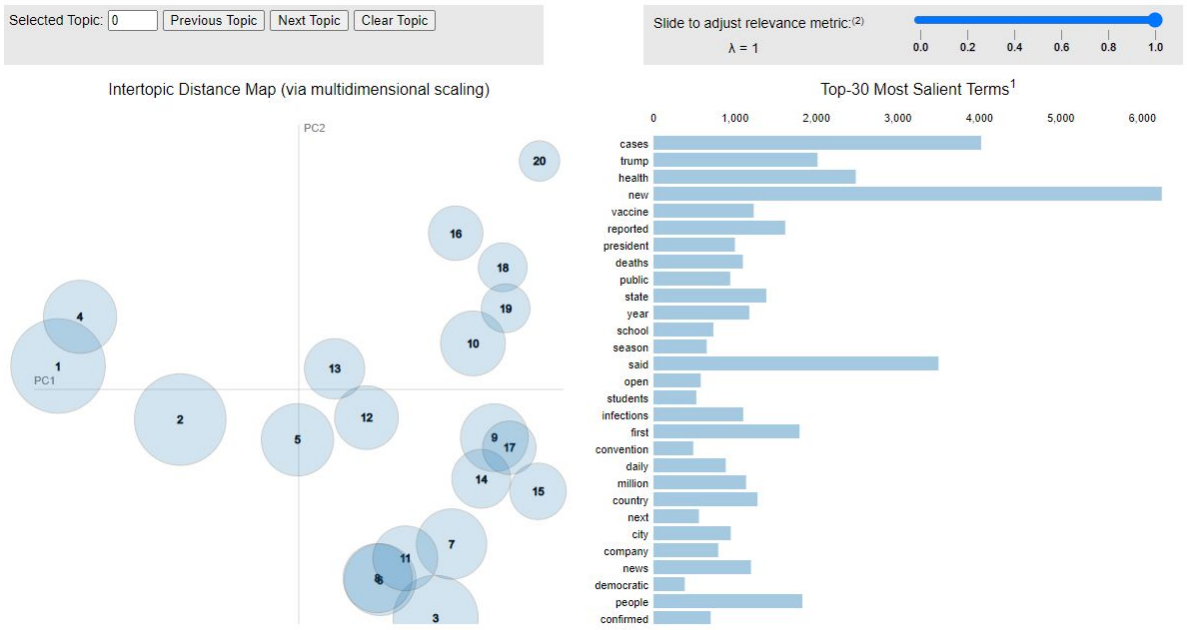


Figure 11: pyLDAvis plot of the LDA prepared model

Conclusion

This project is an attempt to cluster published articles on COVID-19 for the ease of finding relevant information quickly. We also reduced the dimensionality of the dataset for visualization purposes. The dataset of articles was obtained from NewsApi. Clustering and dimensionality allowed for an interactive scatter plot of article topics related to covid-19 news which were grouped together based on similar themes. This helps people to quickly find material related to a central topic. The clustering of the data was done through k-means on a pre-processed, vectorized version of the article's text. As k-means simply split the data into clusters, topic modeling through LDA was performed to identify keywords. KMeans did not give distinct clusters and many keywords were common in a lot of clusters. LDA was the right choice to extract the keywords and topics associated with those. The topics were distinct and were useful in revealing patterns. Visualizing these patterns using Bokeh and pyLDAvis is a feast for the eyes!!!