University of Bologna

Project Work
Natural Language Processing

Automatic Detection of Depression on COVID-19 Tweets

Eleonora Mancini, Eleonora Misino

September 12, 2021

Abstract

The purpose of this project is the analysis of labeling strategies aimed at identifying depression phenomena among users' tweets. The tweets used in this analysis refer to a specific period of the COVID19 pandemic. In particular, the objective is to try to understand if the strategies studied allow to identify evident phenomena of depression among users during the pandemic period. 4 different strategies were developed and analyzed. It was not possible to arrive at a robust solution, but this project highlights some interesting aspects that could be the starting point for a more in-depth analysis.

Contents

1 Exploratory Data Analysis

1.1 A first look at data

We work with COVID19 Tweets dataset, which contains around 178K English tweets collected using Twitter API and a Python script. The tweets were collected by defining a query for `#covid19` hashtag, and cover the period from 25/7/2020 to 30/08/2020.

The dataset contains 13 features (Figures 1), and only 4 of them have missing values (Figure 2): `source`, `user_description`, `user_location` and `hashtags`. The former is useless in our analysis, while the remaining three features require to be explored more in depth.

1. `user_name [str]`
   The name of the user, as they've defined it. Not necessarily a person's name. Typically capped at 50 characters, but subject to change.
2. `user_location [str]`
   Nullable. The user-defined location for this account's profile. Not necessarily a location, nor machine-parseable. This field will occasionally be fuzzily interpreted by the Search service.
3. `user_description [str]`
   Nullable. The user-defined UTF-8 string describing their account.
4. `user_created [str]`
   The UTC datetime that the user account was created on Twitter.
5. `user_followers [int]`
   The number of followers this account currently has.
6. `user_friends [int]`
   The number of users this account is following (AKA their "followings").
7. `user_favourites [str]`
   The number of Tweets this user has liked in the account's lifetime.
8. `user_verified [bool]`
   When true, indicates that the user has a verified account.
9. `date [str]`
   The UTC datetime that the tweet was created.
10. `text [str]`
    Text of the tweet.
11. `hashtags [list]`
    List of hashtags contained in the tweet.
12. `source [str]`
    Source of the tweet.
13. `is_retweet [Bool]`
    When true, indicates that the tweet has been retweeted.

Figure 1: Dataset features.

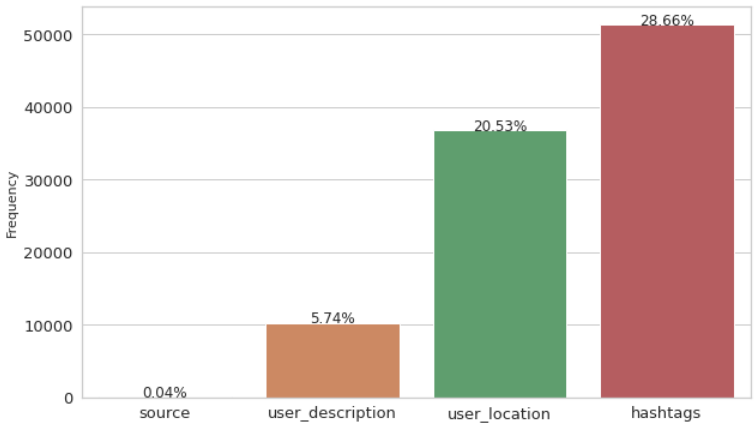


Figure 2: Dataset missing values.

1.2 Locations, Hashtags and User Names

Locations

Around 20% of locations are missing and we have nearly $30K$ different values for this feature.
As we can see in Figure 18, the most common location is India, which represents at least 20% of the whole user locations. Moreover, some locations are quite creative (*no e-pass to cross borders, astroworld, 816 ways to say midwest, Stuck in the Middle*, ...) and there are many inconsistencies: "New Delhi, India", "New Delhi", "London", "London, England", "UK", "United Kingdom", "USA", "U.S.A", ...
Thus, to safely work with this feature we should prune out any fictional location and solve all the inconsistencies, which is out of this project scope.

Hashtags

This feature has nearly 29% of missing values and more than $52K$ unique values. Since tweets were collected searching for covid19, the frequency chart displayed in Figure 19 is not unexpected: around 90% of hashtags consists of different ways to write the search key (COVID19, Covid19, covid19), and the remaining 10% contains the search key plus other subject related hashtags (pandemic, coronavirus, vaccine,...).

User name

As we can see in Figure 20 the most frequent names seem not to belong to physical people, but to media, bots and private companies. We confirmed our hypothesis by taking a look at some users' description:

- GlobalPandemic.NET:

  Breaking News Critical Information to SURVIVE the Coronavirus Outbreak/Civil Unrest

- Coronavirus Updates:

  COVID-19 (Coronavirus) Latest News Statistics. We post updates 24/7 as they come in, all stats are from official sources. Coronavirus COVID19

- covidnews.ch:

  Coronavirus Statistics, Information News

- Open Letters:

  Copies of real letters delivered to the President, Congress, Governors, and State Legislators. Send 'resist' as a Direct Message to @resistbot to write yours.

- Blood Donors India:

  Focused on matching blood donors with those in need. No followers here, only contributors. Spread the word, help save lives. We save 8/day.

Since our task is to analyze people's emotions, we are not interested in texts tweeted by media, bots or private companies. Thus, we tried to filter out these undesired tweets by both manually removing specific users' tweets and searching for users whose description contains at least one of the following words:

- 'news', 'newspaper', 'newspapers', 'media', 'journal', 'journals', 'web sources', 'bot', 'bots', 'updates', 'headlines', 'official government'.

This approach allowed us to prune out around $39K$ undesired tweets (Figure 21).

2 Preprocessing

After carrying out an exploratory analysis of the data, we focus on cleaning up the tweets in question.
We have to define two types of preprocessing aimed at the different labeling techniques that we are going to implement (please refer to *Section* 4 for more details); in particular:

1. the first preprocessing pipeline will be used to extract the `clean_text` to be used in **TWINT-labeling** and **NRClex-labeling**.
2. the second preprocessing pipeline will be used to extract the `clean_text` to be exploited in **VADER-labeling**.

2.1 Pipeline A

The first pipeline is characterized by the following operations:

- **remove format** (bold, italic, and so on)
- **lower text**: transform given text in lower case
- **remove mentions**: removes mentions from the text by finding them through "`@`".
- **remove urls**: removes external links from the text by finding them through "`http(s)`".
- **transform emoji**: transforms emoji in the corresponding word.
- **remove digits**: removes any digits from the text
- **remove punctuation**: replaces any punctuation symbol by a space.
- **apostrophe**: Substitute "‘" with standard apostrophe "'".
- **expand contractions**: transform contracted words into their standard form.
- **lemmatize**: returns the input text lemmatized through `WordNetLemmatizer`.
- **remove stopwords**: removes english stopwords.

2.2 Pipeline B

The second pipeline is characterized by the following operations:

- **remove format** (the same as *Pipeline A*)
- **remove mentions** (the same as *Pipeline A*)
- **remove urls** (the same as *Pipeline A*)
- **apostrophe** (the same as *Pipeline A*)
- **expand contractions - Vader**: Since VADER [1] works well with slang, we do not transform the slang in the text (`slang = False`).
- **strip text**: removes spaces at the beginning and at the end of the string
- **hashtags**: replace hashtag special character with white space
- **trailing**: replace trailing characters with white space.

Text Before Preproccesing	**Clean Text After Pipeline A**	**Clean Text After Pipeline B**
@diane3443 @wdunlap @realDonaldTrump Trump nev...\	trump never claimed wa hoax claim effort	Trump never once claimed COVID19 was a hoax. ...
@brookbanktv The one gift #COVID19 has give me..	one gift give appreciation simple thing always...	The one gift COVID19 has give me is an apprec...
25 July : Media Bulletin on Novel #CoronaVirus...	july medium bulletin novel	25 July : Media Bulletin on Novel CoronaVirus...

Table 1: Examples of tweet texts before and after the two types of preprocessing.

2.3 Notes about the Pipeline B

As can be seen from the lists in the previous sections, in the second pipeline (compared to the first):

- we do not remove the punctuation
- we do not lower the text
- we expand the contractions without replacing the slang
- we do not lemmatize
- we do not transform emojis and emoticons
- we do not remove the stopwords
- since we do not remove the punctuation, we have to manually eliminate spurious characters referring to hashtags and trailing.

We need to use this pipeline because **VADER** [1] assigns a polarity in reference to the following as well:

- Punctuation
- Capitalization
- Conjunctions
- Preceding Tri-gram
- Emojis, Slangs, and Emoticons

Please, refer to the *Section* 4.2 for furhter details.

2.4 Notes about Stopwords

We have included among the stopwords some very frequent words which were either not very significant (such as *ha* or *amp* -aint my problem-) or strictly related to the context of the tweets of our case study (such as *covid*, *coronavirus* etc.), and so not very significant too.
Thus, we add to NLTK English stopwords the following terms:

- covid
- co
- coronavirus
- corona

- virus
- new
- case
- cases
- coronavirusupdates
- ha
- amp

The stopwords in *Pipeline B* have not been removed since *VADER* bases its analysis also on bigrams and trigrams, and therefore it considers also the stopwords to evaluate the possible negativity of a sentence; for example, the sentence "not so happy" has a negative polarity due to presence of the stopword "not".

2.5 Notes about Duplicates Removal

We realized, from an analysis of the clean text, that there were many duplicates and we decided to leave a single copy for each text in order not to generate outliers in the Unsupervised Analysis (KMeans algorithms for Clustering). Please refer to *Section* 6 for furhter details.

2.6 WordClouds

We are going to show the `WordClouds` referred to our tweets, before and after the cleaning carried out in the *Exploratory Data Analysis* and *Preprocessing* sections. We also show the Word Cloud of clean texts through Pipeline B (for VADER-labeling). In the end we present a detailed analysis between the various Word Clouds.

Figure 3: WordClouds visual comparison.

2.6.1 WordClouds Analysis

In this section we present a detailed analysis of the Word Clouds that we generated in different phases of manipulation of the dataset.
In particular, as can be seen from the image shown at the end of the previous section, the 4 phases that interest our comparison are:

1. After EDA - Before PreProcessing
2. After PreProcessing - Before Duplicates Removal
3. After PreProcessing - After Duplicates Removal - Pipeline A
4. After PreProcessing - After Duplicates Removal - Pipeline B (VADER)

1. After EDA - Before PreProcessing

- As expected, the **most frequent terms are related to COVID19** topic('COVID19', 'CO', 'pandemic', 'coronaviru', 'case', 'death', 'mask',...).
- The term `https` has a high frequency, since many tweets contain external links.
- Moreover, the importance of "Trump" and "realDonaldTrump" in the WordCloud, may suggest us the presence of other topics.

2. After PreProcessing - Before Duplicates Removal

- Here the situation is different than in the first scenario. Since we removed the words listed in the customized stopwords list (mainly related to COVID19 topic), other frequent and more informative words emerged such as 'death', 'people', 'face', 'mask', 'health'.

3. After PreProcessing - After Duplicates Removal - Pipeline A

- The main different scenario can be seen here. After the duplicates removal the frequence of the highlighted words drastically changed. This confirms that our strategy of removing duplicates to avoid outliers is correct. We further analyzed that tweets containing the word 'death' were mainly published by bot, newspapers and so on. Since the focus of our study is finding a link between depression-mental state and COVID19, we are mainly interested in tweets written by private users and not by public institutions and therefore the removal of duplicates turns out to be perfect to achieve this goal.

4. After PreProcessing - After Duplicates Removal - Pipeline B (VADER)

- Differently from phase 3, here we did not remove stopwords and words related to COVID19 (so we have some words that we had in phase 1 such as 'COVID19', 'coronoviru' and so on), but we do not have spurious words such as 'https' thanks to the cleaning process.

3 Topic Modeling: Latent Dirichlet Allocation

Topic modeling is a type of unsupervised machine learning technique for discovering the abstract "topics" that occur in a collection of documents.
In particular, this machine learning technique is capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize the set of documents.

There are several existing algorithms you can use to perform the topic modeling. The most common are:

- Latent Semantic Analysis (LSA/LSI)
- Probabilistic Latent Semantic Analysis (pLSA)
- Latent Dirichlet Allocation (LDA)

In our project we use Latent Dirichlet Allocation (LDA).

3.1 Theoretical Overview

LDA [2] is an example of topic modeling used to group the text of a document into topics.
LDA is based on 2 main principles:

1. **Every document is a mixture of topics**. We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B."
2. **Every topic is a mixture of words**. For example, we could imagine a two-topic model of American news, with one topic for "politics" and one for "entertainment." The most common words in the politics topic might be "President", "Congress", and "government", while the entertainment topic may be made up of words such as "movies", "television", and "actor". Importantly, words can be shared between topics; a word like "budget" might appear in both equally.

LDA estimates both of these aspects at the same time: it finds the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document.

Before getting into the details of the Latent Dirichlet Allocation model, let's have a look at the words that form the name of the technique:

- **Latent** indicates that the model discovers the 'yet-to-be-found' or hidden topics from the documents.
- **Dirichlet** indicates LDA's assumption that the distribution of topics in a document and the distribution of words in topics are both Dirichlet distributions.
- **Allocation** indicates the distribution of topics in the document.

LDA assumes that the documents are composed of words that help determine the topics, and therefore it maps the documents to a list of topics by assigning each word in the document to different topics. The assignment is in terms of conditional probability estimates. It is important to note that LDA ignores the order of occurrence of words and the syntactic information; it treats documents just as a collection of words or a *bag of words*.
We can describe the generative process of LDA as, given M documents, N words, and a pre-defined K topics, the model is trained to retrieve:

- *psi*, the distribution of words for each topic K
- *phi*, the distribution of topics for each document i

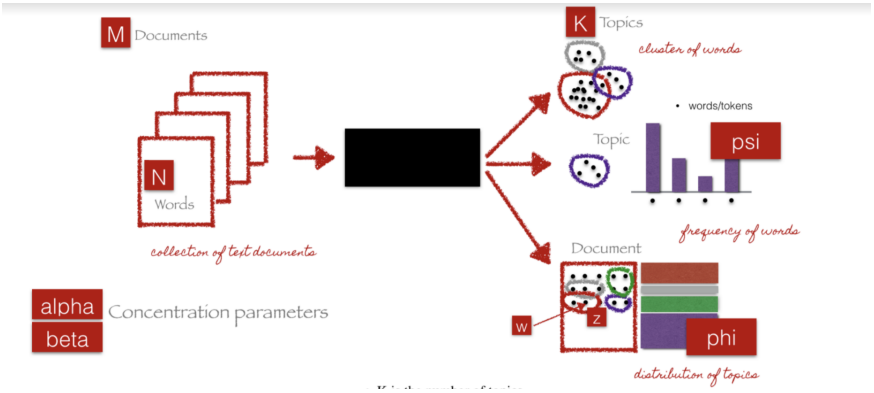


Figure 4: LDA

The main hyperparameters of LDA are:

- **Alpha parameter** is the Dirichlet prior concentration parameter that represents document-topic density; with a higher alpha, documents are assumed to be made up of more topics and result in more specific topic distribution per document.
- **Beta parameter** is the Dirichlet prior concentration parameter that represents topic-word density; with high beta, topics are assumed to made up of most of the words and result in a more specific word distribution per topic.

- **K** specifies the number of topics expected in the corpus of documents. Choosing a value for K is generally based on domain knowledge. An alternate way is to train different LDA models with different numbers of K values and compute the 'Coherence Score'. Choose the value of K for which the coherence score is highest. The latter is the strategy that we followed and that we are going to discuss.

Ideally, we would like to see a few topics in each document and few words in each of the topics. So, α and β are typically set below one.
We have left α and β to their default parameters since their fine-tuning is out of scope.

3.2 Data Preparation

To prepare data in order to be used in LDA algorithm, we need to peform some steps:

- Text Preprocessing (already described in Text Preprocessing Section). In order to correctly extract topics it is recommended to perform the following steps:
  - lower text
  - remove punctuation
  - remove stopwords
  - lemmatize
- Text Tokenization
- Convert the tokenized object into a corpus and dictionary

In order to **optimize interpretatibility** is better to identify phrases through **n-grams** and **filter noun-type structures**.

In order to identify the phrases to be input to the algorithm, we used the following n-gram structures:

- **Bigrams**, which are phrases containing 2 words e.g. 'social media'.
- **Trigrams**, which are phrases containing 3 words e.g. 'Proctor and Gamble'.

There are many ways to detect n-grams. We used **Pointwise Mutual Information (PMI)** score to identify significant bigrams and trigrams to concatenate. PMI measures how much more likely the words co-occur than if they were independent. The metric is sensitive to rare combination of words, so it is used with an occurrence frequency filter to ensure phrase relevance.

We also filtered bigrams or trigrams with the filter (noun/adj, noun), (noun/adj,all types,noun/adj) because these are common structures pointing out noun-type n-grams. This helps the LDA model better cluster topics. Nouns are most likely indicators of a

topic. For example, for the sentence 'The store is nice', we know the sentence is talking about 'store'. The other words in the sentence provide more context and explanation about the topic ('store') itself. Therefore, filtering for the noun cleans the text for words that are more interpretable in the topic model.

Moreover, in order **to have less noisy results** in the search for topics, we have decided to adopt the following strategies:

- Adding new stopwords to the original stopwords list (and used for preprocessing): we have removed very common and not very significant words for our purpose
- Removal of personal names for the same reasons as above
- We filter bigrams or trigrams with noun structures (POS-tag = **NN**). This helps the LDA model better cluster topics, as nouns are better indicators of a topic being talked about.

3.3 Running LDA

LDA requires us to specify the number of topics that exists in a corpus of text. There are several common measures that can be optimized, such as predictive likelihood, perplexity, and coherence. Much literature has indicated that maximizing coherence, particularly a measure named C_v [3], leads to better human interpretability. This measure assesses the interpretability of topics given the set of words in generated topics. Therefore, we choose this measure for the number of topics optimization.
We compute the coherence for a range of **2-14 topics**.

In order to be able to execute the algorithm in question, we took advantage of the implementation provided by *GENSIM - Topic Modelling for Humans* [4]. We have decided to leave all parameters at their default value, and to search for the best values only of the following hyperparameters (in combination with different K values):

- **chunksize** is the number of documents to be loaded into memory each time for training.
- **passes** is the number of training iterations through the entire corpus.
- **iterations** is the maximum iterations over each document to reach convergence (limiting this means that some documents may not converge in time).

In order to **adjust the hyperparameters while finding the best number of topics** we perform the following steps:

1. We set a configuration of hyperparameters (we will try many different combinations of hyperparameters)
2. We run an LDA model with different number of topics (from 2 to 14)

3. We plot the coherence graph

4. We select the *best number of topics* by looking at the coherence graph

    - the best result does not always coincide with the highest C_v, so we try multiple times to find the best result by also increasing the number of topics to reveal further sub topics. Nonetheless, if the same words start to appear across multiple topics, the number of topics is too high.

5. We plot and save the Word Clouds referred to the words per topic for each possible *best_k* that we retrieved from the coherence graph.

6. If the topics still do not make sense, we try to increase `passes` and `iterations`, while increasing `chunksize` to the extent our memory can handle. To understand how we evaluate if certain topics make sense, please refer below to the *Evaluation Method* paragraph.

    - The topic distributions for entire corpus is updated after each 'chunksize', and after each 'passes'. Increasing 'chunksize' to the extent your memory can handle will increase speed as topic distribution update is expensive. However, increasing 'chunksize' requires increasing number of 'passes' to ensure sufficient corpus topic distribution updates, especially in small corpuses. 'iterations' also needs to be high enough to ensure a good amount of documents reach convergence before moving on.

7. Once we are satisfied with the number and the quality of the topics that we obtained with a specific configuration of hyperparameters we save that configuration, compare all the WordClouds that we obtained and we go on by analyzing more in details the computed topics.

Summarizing, this is an *iterative process.*

3.3.1 Evaluation Method

There is no a scientific way to evaluate the quality of topics and it is precisely here that the **difficulty of evaluating the results** obtained with LDA lies.

Therefore, evaluating the quality of the topics and then deciding whether to continue in the search for the best hyperparameters is a **human task** and has to be executed by who is carrying out the evaluation. In particular, this type of evaluation greatly depends on the cultural background of the person who is performing the analysis.
Thus, although we were able to find a more technical strategy to evaluate the best number of topics for each configuration, we are still not able to evaluate the quality of the topics in a statistical way.

3.3.2 Summary of the trials

For each configuration (*passes, iterations, chucksize*) listed below, we extract the best number of topics k following the strategy explained in the previous sections.

- 1, 50, 10000
  - k = 10, 13
- 10, 50, 10000
  - k = 11, 13
- 10, 100, 10000
  - k = 8, 11, 13 , 14
- 10, 1000, 10000
  - k = 8, 10, 12 , 14

For each configuration we report in *Appendix* B the Coherence Graph and the Word Clouds of words-per-topic obtained with the k numbers listed above.

3.4 Exploration of LDA results

3.4.1 WordClouds Analysis

After carefully analyzing the WordClouds shown in *Appendix* B, we selected the best configuration by looking for the configuration that allows us to extract the greatest amount of significant topics.
Obviously, this type of analysis, as specified in the *Evaluation Method* section, is very subjective.
We have tried to recover some objectivity by looking at the news published in the same period of our tweets.
In light of this analysis, the best configuration is:

- passes = 10
- iterations = 100
- chunksize = 10000
- number of topics = 13

Below is the Coherence Graph that led us to choose $K = 13$ as number of topics and the Word Cloud linked to this configuration:

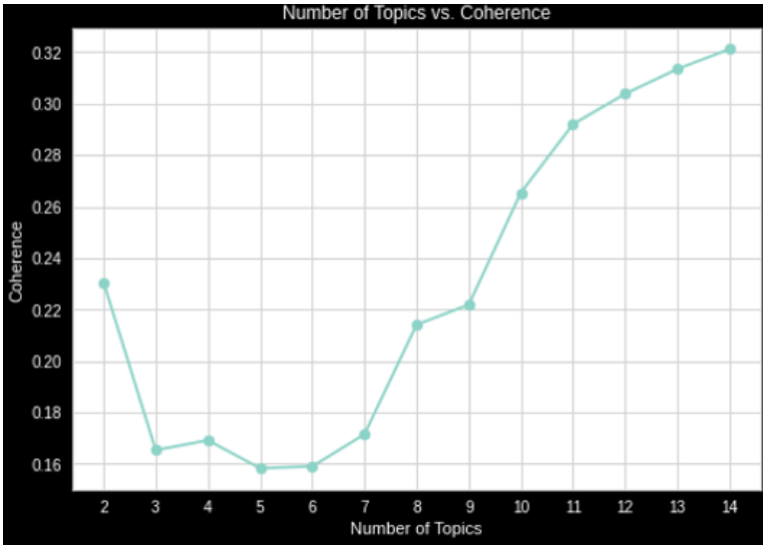


Figure 5: Coherence Graph that made us choose 13 as the best topic number.

Figure 6: WordCloud obtained with best LDA hyperparameters configuration.

Topic analysis

1. **Topic 0 - Aspects related to the COVID19 crisis**: governement, university, friend (and so friendships), use the mask
2. **Topic 1 - Number of deaths and use of the medical mask**:related to population, story in the sense of 'historical period', event, person
3. **Topic 2 - Russia's first vaccine announced in August 2020**: "Aug. 11, 2020 MOSCOW — Russia has become the first country in the world to approve a vaccine for the coronavirus."
4. **Topic 3 - Issue of osepdals/doctors and people at risk, suffering from other diseases (such as heart disease)**
5. **Topic 4 - Students, college, infection**
6. **Topic 5 - Topics related to the economy in general**
7. **Topics 6**: it is difficult to categorize Topic 6.
8. **Topic 7 - The challenge of hospitals against time. The importance of taking action, gratitude for the things done by the hospitals.**
9. **Topic 8 - School and Education of kids**: parents forced to stay home and look their children instead of going to the office. Being at home and not going at school increase safety. It is linked to the themes of teleworking and distance-learning.
10. **Topic 9 - In August, India recorded the highest number of COVID19 cases since the start of the pandemic and a very high number of deaths per day**: "India reported close to 2 million Covid19 cases in August, the highest recorded in any country during any month since the outbreak of the pandemic. The country also saw a surge in deaths from the virus, with 28,859 fatalities reported in August, a 50% jump from the previous month's toll."
11. **Topic 10 - U.S.A Elections**: trump, america, vote, election
12. **Topic 11 - Work issue: risk in the workplace, risk of losing your job**
13. **Topic 12 - Pandemic crisis shared by the whole community, worldwide impact**

3.4.2 Dominant topic and its percentage contribution in each document

We computed the dominant topic for each tweet and is percentage distribution. Below is an example:

Document_No	Dominant_Topic	Topic_Perc_Contrib	Keywords	Text
0	9.0	0.5188	day, today, week, india, death, change, quaran...	[smelled, scent, hand_sanitizers, today, past,...
1	10.0	0.2692	trump, fact, money, situation, election, vote,...	[trump, never, claimed, hoax, claim, effort]
2	7.0	0.2693	time, thing, hospital, thank, question, county...	[gift, appreciation, simple, thing]
3	9.0	0.6154	day, today, week, india, death, change, quaran...	[july, medium, bulletin, novel]
4	1.0	0.4154	death, number, state, person, face_medical_mas...	[death, continue, rise, bad, politician, busin...
5	12.0	0.2693	world, crisis, care, community, impact, woman,...	[wear, face_covering, shopping, includes, visi...
6	11.0	0.3590	health, work, news, risk, job, medium, problem...	[praying, good, health, recovery, covidpositive]
7	10.0	0.4864	trump, fact, money, situation, election, vote,...	[pope, god, prophet, sadhu, sundar, selvaraj, ...
8	8.0	0.4154	school, child, home, look, kid, education, saf...	[response, cancel, compartment, exa]
9	5.0	0.5126	help, month, business, spread, president, need...	[order, logo, graphicdesigner, logodesign, log...
10	1.0	0.3590	death, number, state, person, face_medical_mas...	[protect, real, number, climbing, fast, contin..

Figure 7: Dominant topic per tweet.

Observing the texts and the topic associated with the text, we can say that the analysis carried out by LDA turns out to be satisfactory

Furthermore, the texts broadly confirm that the topics we identified in the previous section were interpreted fairly correctly.

3.4.3 The most representative sentence for each topic

We have extracted the text that best represents each topic. It is interesting to focus on the following things:

- **Topic 11**: for this topic, which we have previously identified as "**Work issue**", it is interesting to observe that the representative text contains words such as *depression, stress, die, problem.* This makes us think, in the context of our analysis, that in the period of the pandemic there was a close correlation between the fear of losing a job and stress/depression.

- **Topic 2**: from the observation of the representative text, our interpretation of

- **Topic 2** is also correct, identified as "Russia's first vaccine announced in August 2020". There is even the word *sputnikv* which is the specific name of the vaccine in question.

- **Topic 10**: from the observation of the representative text, our interpretation of **Topic 10** is also correct, identified as "**U.S.A. elections**". It's very interesting the fact that, in the Representative Text, 4 words out of 9 are "*maga*" which means *Make America Great Again.*

- **Topic 7**: from the observation of the representative text, our interpretation of Topic 7 is also correct, identified as "The challenge of hospitals against time. The importance of taking action, gratitude for the things done by the hospitals.". In particular, there many words that coincide with "**folded_hand**", which is

the translation of the emoji which simulates the act of praying; this emoji is commonly used to express *gratitude*.

- **Topic 6**: in the previous section, by looking only at the WordClouds, we would not be able to extract a suitable topic for those words. Here, from the observation of the Representative Text, we can observe that the following words are present: *idc (i don't care), face, steam, nose, iamnotokaywiththis (i am not okay with this)*. "Face", "Steam" and "Nose" could be related to the fact that in order to prevent the spread of the coronavirus through the air emitted from nose we need to wear a mask. The combination of the concept just explained and the words "idc" and "iamnotokaywiththis" could be related to *COVID19 deniers* (people who do not accept the existence of the COVID19). Another hypothesis could be linked to the fact that at that time Netflix removed the TV series "IAmNotOkayWithThis" due to the circumstances related to COVID. News of 21 August 2020: 'The Society' 'I Am Not Okay With This' Canceled By Netflix

- **Topic 12**: also in this case it seems that the topic we previously identified is slightly wrong, as by observing the representative text there are words like "gender", "inequality", "women". Therefore, it appears that this topic is more akin to "**Gender Inequality**".

- **Topic 3**: observing the text related to this topic, words like *antigen, test, conducted, doctor* emerge; therefore it seems that the topic is slightly different from that identified by us and that it is therefore more linked to the "**Administration of rapid tests for monitoring the epidemic**".

For all other topics it is not possible to conduct a meaningful analysis as the words that have emerged do not seem very sensible.
Probably, in light of what we got here, we could have also removed words like "idc" (I don't care) or "rlr" (real life raw) in preprocessing; unfortunately these things cannot be noticed from an initial analysis.

3.4.4 Word Count of Topic Keywords

We want to understand which words have a greater weight in determining the topic. The weight associated with a word indicates how significant that word is to distinguish that topic from the others.
In particular, it is interesting to note that in some cases LDA inserts words that it considers most significant within the topic, although these are not the most frequent words. This is one of the strengths of LDA.
The images referring to this section can be found in *Appendix* C.
We analyzed the various topics by looking at both the Word Clouds and the graphs that show the weight of the words together with their occurrence in the texts and we have identified these clusters:

- **Topic 0**:"**Aspects related to the COVID19 crisis**". The most weighted word coincides with the most frequent word for this topic: "**mask**". We know that the **mask** is the symbolic element of this pandemic.

- **Topic 1**: "**Number of deaths and use of the medical mask**". **death**, **number** and **person** are both the most frequent and weighted words for this topic.

- **Topic 2**: "**Russia's first vaccine announced in August 2020**".The most weighted words are: **vaccine**, **country**, **life**.

- **Topic 3**: during a first analysis we had identified "**Issue of hospitals/doctors and people at risk, suffering from other diseases (such as heart disease)**". By looking at keywords and most representative sentence for each topic we changed it into "**Administration of rapid tests for monitoring the epidemic**". This is reasonable since the most weighted words are **test**, **report**, **support**.

- **Topic 4**:" **Students, college, infection**" is the topic identified through the Word Clouds. The most weighted words are **year**, **student**, **plan**. In reality, the words identified as keywords for this topic do not seem to be very related to each other so, even if we look at the words that have had the most weight for LDA, we are unable to clearly confirm the hypothesis regarding our interpretation of this topic.

- **Topic 5**: "**Topics related to the economy in general**". If we combine the most frequent and the most weighted word, we obtain "need help". It seems that our hypothesis for this topic is correct.

- **Topic 6**: As we said in the previous sections, is difficult to individuate a topic.

- **Topic 7**: "**The challenge of hospitals against time. The importance of taking action, gratitude for the things done by the hospitals.**" The most weighted words are **time**, **thing**, **hospital**.

- **Topic 8**: "**School and Education of kids at home**". The most weighted keywords are **school**, **child**,**home**.

- **Topic 9**: "**In August, India recorded the highest number of COVID19 cases since the start of the pandemic and a very high number of deaths per day**". The most weighted words are **day**, **today**, **week**, **india**, **death**.

- **Topic 10**: "**U.S.A Elections**". The most weighted words are **trump, fact, money, situation, election**.

- **Topic 11**: "**Work issue: risk in the workplace, risk of losing your job**". The most weighted words are **health, work, news, risk, job**.

- **Topic 12**: From the anlysis of the weights of the keywords we are unable to confirm the hypotehsis that we made in the previous section ("**Gender Inequality**").

4 Tweets Labelling

In this section we present the different strategies we decided to adopt to assign the label `depressed`/ `not_depressed` to each tweet.
In general, regardless of the strategy, we assigned the labels with the following meaning:

- **0** to indicate **not depressed** tweet
- **1** to indicate **depressed** tweet

The strategies we adopted are the following:

- **TWINT**: through the use of TWINT [5] [1] we were able to extract different datasets based on keywords related to the theme of depression.
  Based on the analysis of these datasets we were able to create what we called a `depression_list`, that we used to label the tweets. Please, see the *Section* 4.1 for further details.
- **VADER**: through the use of VADER-Sentiment Analyzer [1] we were able to label each tweet by studying the polarities assigned by VADER to each tweet. In particular, tweets with positive polarity were labeled as `not_depressed` (positive sentiment), while tweets with negative polarity were labeled as `depressed` (negative sentiment). Obviously behind this reasoning there is a basic knowledge linked to the fact that in general the condition of depression is associated with a feeling socially and globally evaluated as negative. Please, see the *Section* 4.2 for further details.
- **NRCLex**: through the use of NRCLex [6] we measured emotional affect from a body of text. Affect dictionary contains approximately 27.000 words, and is based on the National Research Council Canada (NRC) affect lexicon and the NLTK library's WordNet synonym sets. So, we were able to analyze the emotions retrieved by NRCLex in our tweets and to mark as `depressed` those tweets affected by negative top-emotions. We considered as negative top-emotions *fear* and *sadness* (because *negative* is too much general to be directly linked with the theme of depression). Please, see the *Section* 4.3 for further details.

[1] An advanced Twitter scraping tool written in Python that allows for scraping Tweets from Twitter profiles without using Twitter's API.

4.1 TWINT

Through the use of TWINT, we were able to extract different datasets based on keywords related to the theme of depression.

The choice of the period
All the datasets were extracted by taking the tweets written in the period from 01/01/2019 to 01/05/2019.
We chose to use a period prior to the evolution of COVID19 in the world, so that this research produced the most objective results possible and did not overlap with our analysis.

Pipeline
The research was done incrementally:

- We started with a word ("depressed")
- We extracted the tweets of the chosen period characterized by this word
- Through the analysis of the Word Cloud of the extracted dataset we extracted one of the most frequent words ("anxious")
- We then repeated the download of the dataset for this new word and proceeded with the same technique for the extraction of a third term and so on
- Note that we scraped only tweets written in English
- We finally extracted **5 different datasets** referring to the words *depressed*, *anxious*, *stressed*, *sad*, *lonely*.
- We applied our **Text Preprocessing - Pipeline A** to these new datasets
- We compared the Word Clouds of the 5 datasets and carried out analyzes based on the most frequent terms in and among the various datasets in order to extract the `depression_list`: a list of terms related to the theme of depression
- Finally, we used this list to analyze our tweets:
  - if the tweet contains at least one word of the `depression_list`, then it is labeled as **depressed (1)**
  - if the tweet does not contain any word of the `depression_list`, then it is labeled as **not_depressed (0)**

Additional Notes
We decided to select adjectives instead of nouns considering that when someone want to communicate something on social networks, usually refers to himself with adjectives. Examples: "I am depressed", "I feel anxious".

Depression List Creation

Here we are going to present the analysis of the most frequent words in and among Word Clouds.

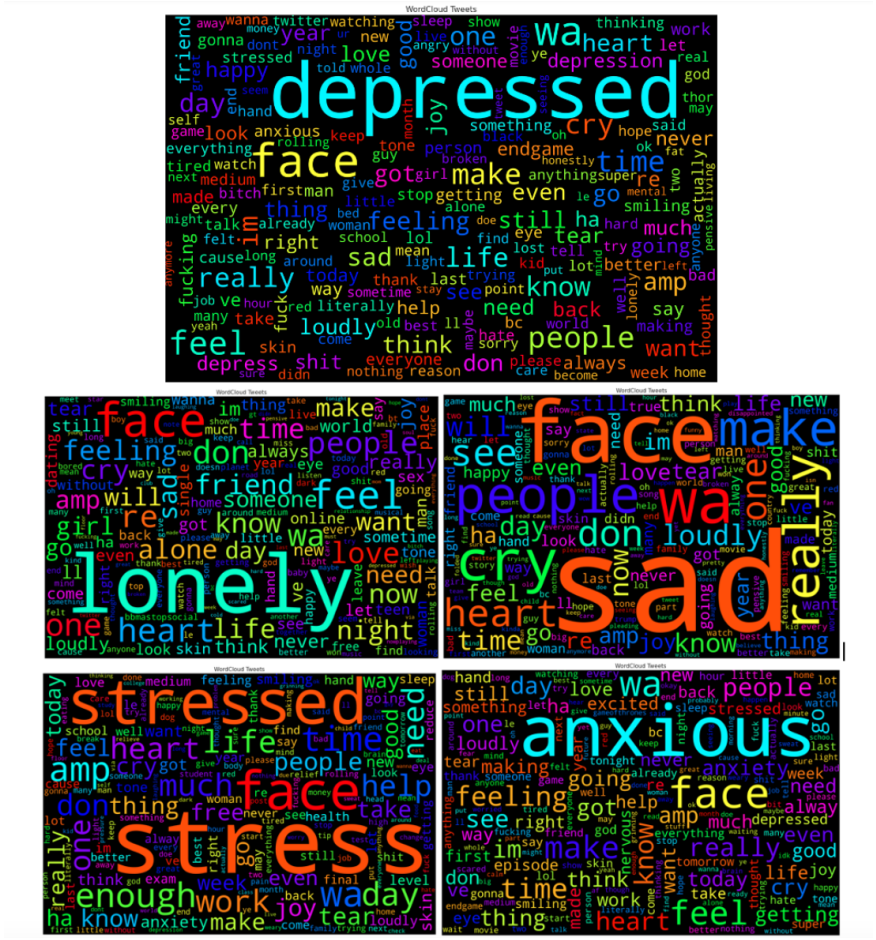


Figure 8: Word Cloud of datasets that refer to each depressed word identified.

- In the Word Clouds of *stress* and *anxious* we found the words *depressed* / *depression*, so this makes us think that these 3 words are good candidates for `depression_list`; vice versa we found this link also between *depressed* and *stressed* because both in the Word Clouds of *depressed* and *anxious* there is the word *stressed*.

- In the Word Cloud of *depressed* we found *depress* and *depression* which we added to the `depression_list`.

- In the Word Cloud of *depressed* emerged words such as *alone*, *cry*; these words also emerged in the Word Clouds of *sad* and *anxious* and this make us think that there exists a link between *sad* - *depression* - *anxious* .
- In the Word Clouds of *lonely* and *sad* we found in both the word *cry*.
- In all Word Clouds we find negative words such as *nothing*, *never*. Many scientific articles [7] correlate the use of these two terms to depression, so we added them to the `depression_list`.
- In all our Word Clouds we found the word *face* in the sense of "fronting", but we did not put it in the `depression_list` due to the unfortunate combination with "face mask"; otherwise we would have had a lot of false positives. However, if there had not been "face mask" (so if the topic of the target tweets had been different from the COVID19) we would have put it.

In conclusion, mainly there seemed to be a fairly important correlation between the words that we highlighted above, that lead to depression.

So, our `depression_list` is composed as follows:
`[depression, depressed, depress, anxious, anxiety, stress, stressed, cry, alone, lonely, never, nothing, sad]`.

4.2 VADER

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. [1]
VADER uses a combination of sentiment lexicons (which are lists of lexical features e.g., words) which are generally labelled according to their semantic orientation as either positive or negative.
VADER has been found to be quite successful when dealing with social media texts, NY Times editorials, movie reviews, and product reviews; this is because VADER not only tells about the *positivity* and *negativity* score but also tells us about how positive or negative a sentiment is.
VADER analyses sentiments primarily based on certain key points:

- **Punctuation**: The use of an exclamation mark(!), increases the magnitude of the intensity without modifying the semantic orientation. For example, "The food here is good!" is more intense than "The food here is good." and an increase in the number of (!), increases the magnitude accordingly.

- **Capitalization**: Using upper case letters to emphasize a sentiment-relevant word in the presence of other non-capitalized words, increases the magnitude of the sentiment intensity. For example, "The food here is GREAT!" conveys more intensity than "The food here is great!".

- **Degree modifiers**: Also called intensifiers, they impact the sentiment intensity by either increasing or decreasing the intensity. For example, "The service here is extremely good" is more intense than "The service here is good", whereas "The service here is marginally good" reduces the intensity.

- **Conjunctions**: Use of conjunctions like "but" signals a shift in sentiment polarity, with the sentiment of the text following the conjunction being dominant. "The food here is great, but the service is horrible" has mixed sentiment, with the latter half dictating the overall rating.

- **Preceding Tri-gram**:By examining the tri-gram preceding a sentiment-laden lexical feature, we catch nearly 90% of cases where negation flips the polarity of the text. A negated sentence would be "The food here isn't really all that great".

- **Emojis, Slangs, and Emoticons**: VADER performs very well with emojis, slangs, and acronyms in sentences.

The **Compound Score** is a metric that calculates the sum of all the lexicon ratings which were normalized between -1(most extreme negative) and +1 (most extreme positive).

Pipeline
After computing the score with VADER Sentiment Analyzer we:

- labelled as **depressed (1)** if `compound_score` < 0
- labelled as **not_depressed (0)** if `compound_score` ≥ 0

4.3 NRCLex

Through the use of NRCLex [6] we measured emotional affect from a body of text. Affect dictionary contains approximately 27,000 words, and is based on the National Research Council Canada (NRC) affect lexicon [8] and the NLTK library's WordNet synonym sets [9] .
So we were able to analyze the emotions retrieved by NRCLex in our tweets and to mark as **depressed** those tweets affected by negative top-emotions.
Let's take a look at how NRCLex-python module works:

1. Assign input text (each tweet in our case)
2. Create NRCLex object for each input text.
3. Apply methods to classify emotions.

Sr.	Method	Description
1	*emotion.words*	Return words list.
2	*emotion.sentences*	Return sentences list.
3	*emotion.affect_list*	Return affect list.
4	*emotion.affect_dict*	Return affect dictionary.
5	*emotion.raw_emotion_scores*	Return raw emotional counts.
6	*emotion.top_emotions*	Return highest emotions.
7	*emotion.affect_frequencies*	Return affect frequencies.

Figure 9: NRCLex methods for emotions classification.

4. Among all of these methods we used `top-emotions` to return the highest emotions of each tweets.
5. We iterated among the dataframe containing NRCLex results for each tweet and if `top_emotions` list of a tweet contained an emotion of our `depression_list` we marked it as **depressed (1)** (**0** otherwise).

Additional Notes - Affects
Emotional affects measured include the following:

- fear
- anger
- anticipation
- trust
- surprise
- positive
- negative
- sadness
- disgust
- joy

Additional Notes - Depression List
We considered as negative top-emotions *fear* and *sadness* (because *negative* is too much general to be directly linked with the theme of depression).

Additional Notes - Text Preprocessing
In the text preprocessing section we used the Lemmatization technique to find right matches with the words in NRC Lexicon, since NRC lexicon contains lemmatized terms.

5 Labelling Comparison

To compare the 3 different strategies we used to label our tweet we rely on **Cohen's kappa**, which measures the agreement between two raters A and B which classify N items into $|C|$ mutually exclusive categories, formally

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where p_o is the relative observed agreement among raters, and p_e is the hypothetical probability of chance agreement. The latter is computed by using the observed data to calculate the probabilities of each observer randomly seeing each category, that is

$$p_e = \frac{1}{N^2} \sum_k n_{k1} n_{k2}$$

where n_{ki} is the number of times classifier i predicted category k.

In our case study $C = \{0, 1\}$, thus

$$p_e = \tfrac{1}{N^2}(n_{0A} n_{0B} + n_{1A} n_{1B})$$

and p_o can be computed by looking at the contingency table of the two labelling strategies A and B.
Looking at *Cohen's kappa* definition, we can easily conclude that

- if our labelling strategies are in complete agreement $\Rightarrow \kappa = 1$;
- if there is no agreement other than what would be expected by chance $\Rightarrow \kappa = 0$.

By looking at Figure 10, we can clearly observe that our 3 labelling strategies are not equivalent: even though VADER [1] and NCRLex [6] agree with each other slightly more than with TWINT[5], their score is still very low.

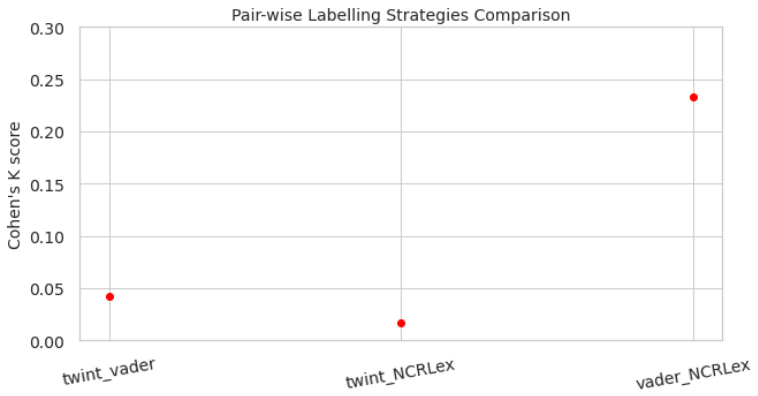


Figure 10: Labelling comparison using Cohen's kappa.

6 Unsupervised Analysis

6.1 Latent Semantic Analysis

As the vast majority of text data, our dataset heavily suffers from high-dimensionality. In order to address this problem and easily work with our data, we apply the well known **Latent Semantic Analysis (LSA)** technique on the tf-idf matrix.

Since we are dealing with sparse data, we rely on `scikit-learn TruncatedSVD` transformer, which performs linear dimensionality reduction by means of truncated **Singular Value Decomposition (SVD)**.
To define the number of components we look at the cumulative explained variance ratio (Figure 11) and we take 500 as a trade-off between the percentage of data variance explained (around 30%) and the clustering algorithm training time.

SVD decomposition is also very useful to represent high-dimensional data into a 2-dimensional space by plotting the first 2 components of the reduced matrix. In Figure 31 we report the unlabeled data represented in a 2-dimensional space, along with the same data labeled according to our 3 different labelling strategies. By visually inspecting the three labelling startegies, we can clearly see that TWINT largely differs from NCRLex and VADER for the number of tweets labelled as depressed. In fact, there are less depressed tweets according to TWINT strategy w.r.t. the other two strategies.

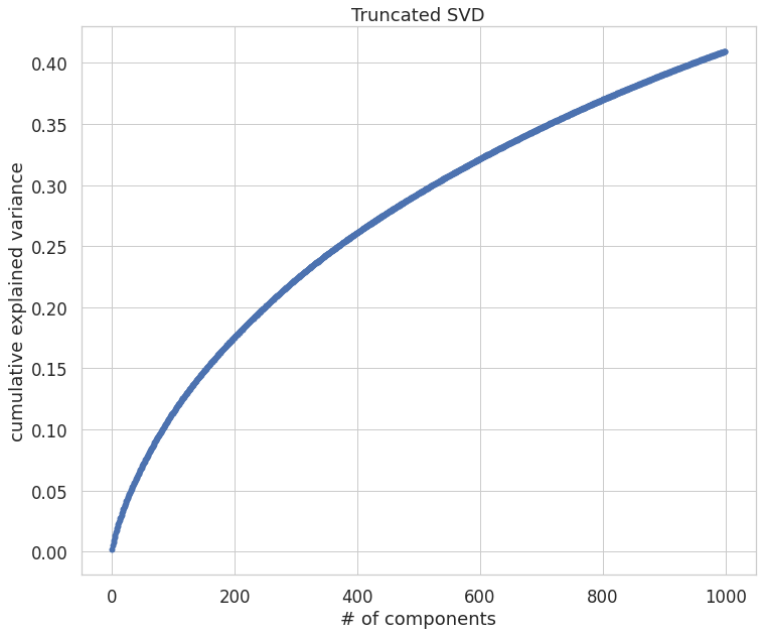


Figure 11: Cumulative explained variance ratio as function of the number of components.

6.2 Clustering

One of the most well known clustering method is the **k-Means algorithm** which clusters a set of samples X into k disjoint clusters, each described by the mean or "centroid" μ_j of the samples in the cluster.
k-Means algorithm is based on the **minimization of inertia**, or **within-cluster sum-of-squares criterion**, formally

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

Despite its simplicity and efficiency, k-Means have various drawbacks:

- it is based on the *inertia* which assumes **convex and isotropic clusters**, which is not always the case. It responds poorly to elongated clusters, or manifolds with irregular shapes.
  Moreover, *inertia* is **not a normalized metric**, thus it suffers from the "curse of dimensionality". However, the LSA we have performed should mitigate this problem and speed up the computation.

- k-Means algorithm requires to specify the number of clusters which is not always a simple task. As explained in the next section, We will try to find the best number of clusters by applying the **Elbow Method**.

- k-Means **convergence is highly dependent on the initialization of the centroids**. To avoid being stuck in a local minimum the computation is often done several times, with different initialization of the centroids.
  To speed up the process we rely on the **k-means++ initialization scheme**, which has been implemented in `sklearn.cluster.KMeans` class. With this initialization, the centroids are placed distant from each other, leading to provably better results than random initialization.

6.2.1 Elbow Method

As anticipated, we use the **Elbow Method** to select the **best number of clusters**. This method consists of running the algorithm multiple times with an increasing number of cluster choice k, and computing the sum of squared distances from each point to its assigned centroid (i.e. the **distortion**). Then, the *distortion* is plotted as a function of the number of clusters, and the best value of k is selected by identifying the point where the *distortion* begins to decrease most rapidly before the curve reached a plateau.

In Figure 12 we display the Elbow Method result for $k \in [2, 12]$: by looking at the blue curve, we can identify an elbow at both $k = 6$ and $k = 8$. In the next section we will compare these 2 values of k by performing an analysis on the two resulting sets of clusters.

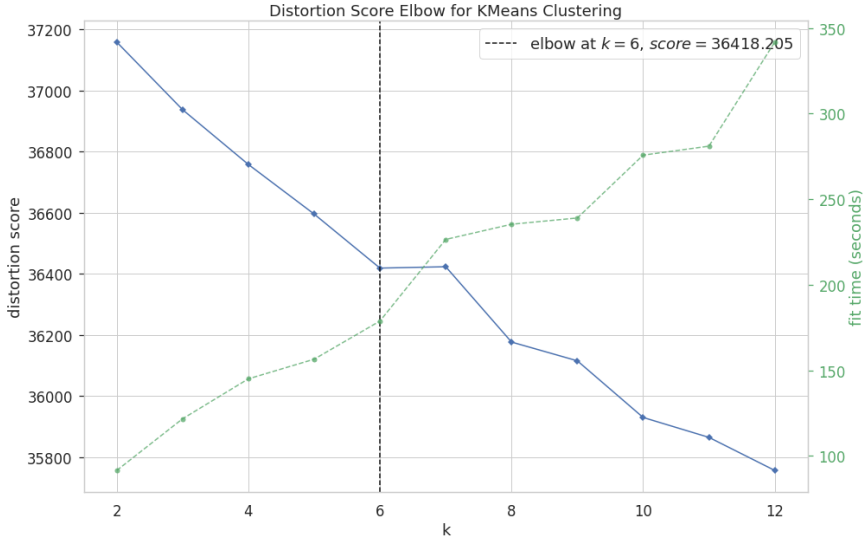
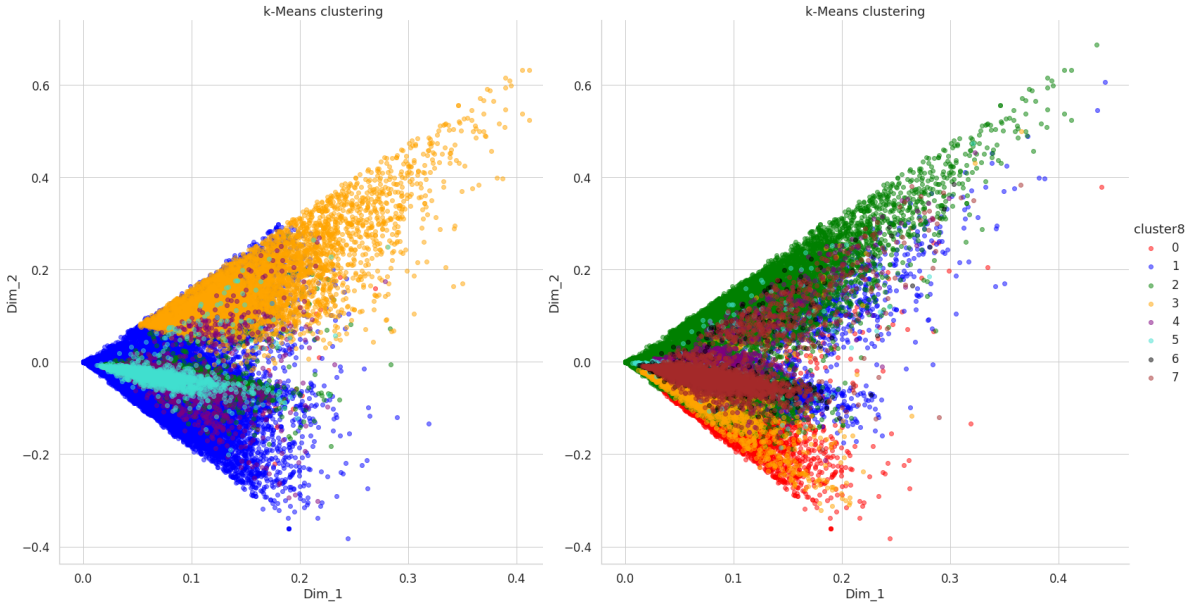


Figure 12: Distortion score (in blue) and fit time (in green) as function of the number of clusters k.

6.2.2 k-Means Clustering

As mentioned in the previous section, we compared two values of k (6 and 8) by applying the corresponding k-Means algorithm on the reduced matrix and looking at the results.
In Figure 13 we display the 2-dimensional representation of our dataset by colouring data points according to the cluster they belong to.



(a) $k = 6$ (b) $k = 8$

Figure 13: k-Means clustering with two different values of k.

As we can clearly see

- $k = 6$: **cluster 1** (blue) is by far the largest cluster, while **cluster 5** (turquoise) is the smallest;
- $k = 8$: **cluster 2** (green) is by far the largest cluster, while **cluster 5** (turquoise) is the smallest;

These observations are confirmed in the **intercluster distance maps** (Figures 14 and 15) which are 2-dimensional representations of the cluster centers obtained by performing **Multidimensional Scaling (MDS)** on k-Means output.
In the intercluster distance maps **the distance among the centroids are preserved** (i.e. the closer two centers are in the visualization, the closer they are in the original feature space). Moreover, the **clusters are sized by membership** (i.e. the number of instances that belong to each center).
It is also important to keep in mind that if two clusters overlap in the 2D space they do not necessarily overlap in the original feature space.

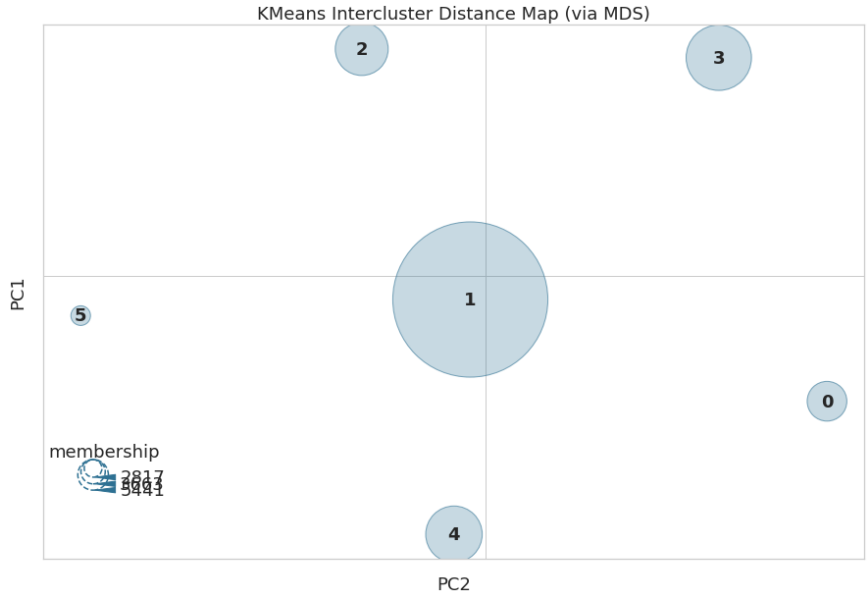


Figure 14: Intercluster distance map for $k = 6$.

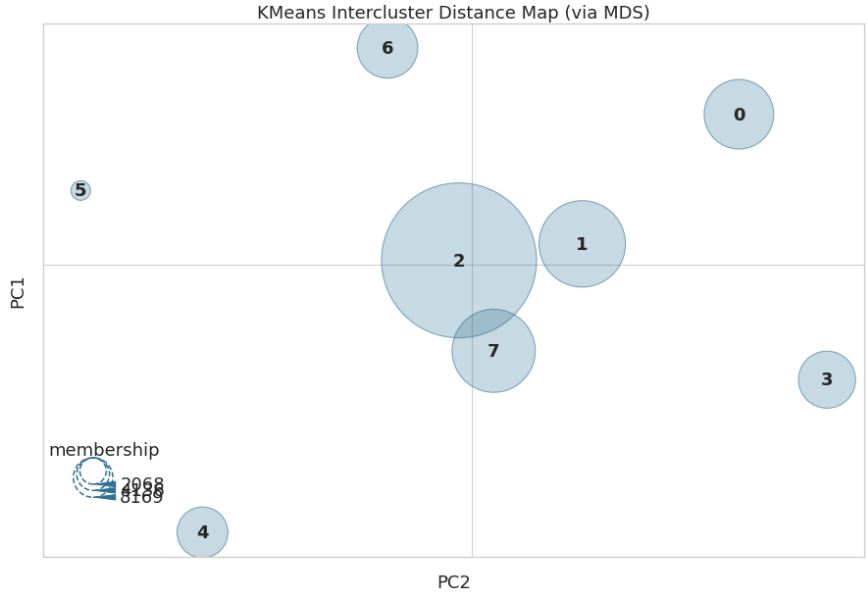


Figure 15: Intercluster distance map for $k = 8$.

6.3 Focus on single clusters

Since we would like to to get a deeper insight into the clusters of tweets found with *k-Means* algorithm, we plot them individually in Figures 32 and 33, and we display their Word Clouds in Figures 34 and 35.
We also performed a LDA on each cluster using the same configuration specified in section *LDA - Running LDA with best parameters* to find the most relevant keywords (Figures 36 - 49).

We relied on both the Word Clouds and the LDA keywords to try to identify the central topic of each cluster:

- $k = 6$
  - **Cluster 0**: the central topic is **school**, with a particular attention on kids education (Figure 36).
  - **Cluster 1**: the focus is on **daily death report** (Figure 37).
  - **Cluster 2**: here is no a clear central topic (Figure 38).
    The 10 keywords have very similar weights and are not specific of a single topic. Moreover, the phrase "counted_nov_eod_e_g_" is not so easy to decode. We suppose it refers to the USA president's election ("nov" $\rightarrow$ "November", "eod" $\rightarrow$ "end of day", thus "counted (votes) at the end of the day in November"; but We have no clue about "e_g" and it is strange that "trump" does not appear among the keywords).
  - **Cluster 3**: the central topic is **medical face mask** (Figure 39).
  - **Cluster 4**: as in Cluster 2, the keywords have very low relevance and can be used in many different topics. Most of them are temporal indications ("day", "today", "time", "year", "month", "week") and there is also the very vague noun "thing". Thus, we are not able to identify a central topic (Figure 40).
  - **Cluster 5**: the central topic is clearly the **Russian vaccine** (Figure 41).
- $k = 8$
  - **Cluster 0**: the focus is on **daily death report**, as in Cluster 1 with $k = 6$ (Figure 42).
  - **Cluster 1**: there is no a clear topic. The keywords have very low weights and do not seem related to a common topic (Figure 43).
  - **Cluster 2**: the same as in Cluster 1 (Figure 44).
  - **Cluster 3**: the keywords suggest that tweets in this cluster mainly came from **India**.
    In fact, "india" is by far the most relevant keyword and "lakh" is a unit in the Indian numbering system equal to one hundred thousand (Figure 45).

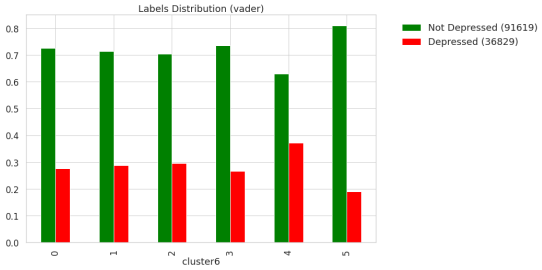
- **Cluster 4**: there is no a clear topic, but it is interesting to observe that many keywords are encoding of emoji (`folded_hand_medium_light_skin_tone`, `woman_shrugging_medium_light_skin_tone`, `backhand_index_pointing`, ...)(Figure 46).
  This suggests us that tweets of Cluster 4 contained lot emoji, and we confirm our hypothesis by visually inspect tweets text.
- **Cluster 5**: as in Clusters 1,2 and 4, we are not able to find a central topic (Figure 47).
- **Cluster 6**: the central topic is the **Russian vaccine** as in Cluster 5 of $k = 6$ (Figure 48).
- **Cluster 7**: the focus is on **health**.
  This cluster is really interesting since some of its keywords may be related to depression if we considered them together: "mental_health", "crisis", "help", "life". We will inspect this cluster more in depth in the next section.

Thus, we were able to identify a central topic for 4 out of 6 clusters for $k = 6$ and only 4 out of 8 clusters for $k = 8$.
Two topics are in common (**daily death report** and **Russian vaccine**), while we have **school** and **medical face masks** for $k = 6$, and **health** and a cluster of tweets largely coming from **India** for $k = 8$.

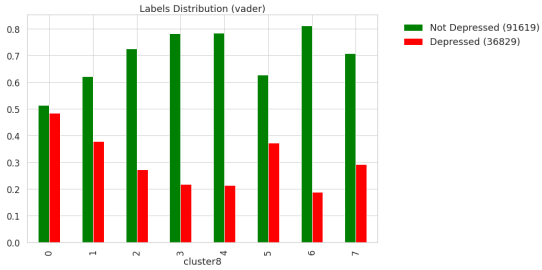
6.3.1 Labelling among clusters

Let's compare how the depressed and not depressed tweets are distributed in the clusters according to each labelling strategy for the two values of $k = 6, 8$.

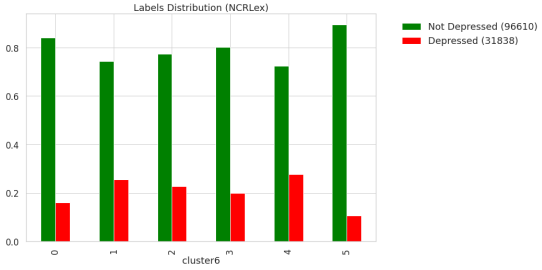
In Figure 16 we display the fraction of depressed and not depressed tweets in each cluster according to a given labelling strategy.
As expected by looking at Cohen's kappa values (Section 5), the three strategies do not agree. In particular, TWINT is very different from NCRLex and VADER for both $k = 6$ and $k = 8$.
All of the clusters have more not depressed tweets than depressed, except for clusters 0 and 7 with NCRLex strategies. If we look at the word count and importance of topic keywords for these two clusters (Figures 42 and 49), we can see that cluster 0 is dominated by the word *death*, which has a very negative connotation for NCRLex; while cluster 7 has words like *health, pandemic, mental_health, crisis, help*. Thus, it is plausible to have more depressed tweets than not depressed ones in these two clusters. Moreover, the scarcity of depressed tweets with TWINT labelling even in cluster 0 and 6 suggests that this strategy is the less correct among the three tested.



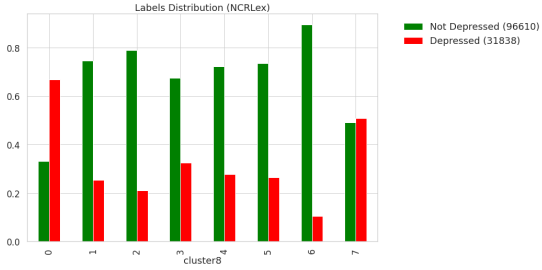
(a) VADER - k = 6



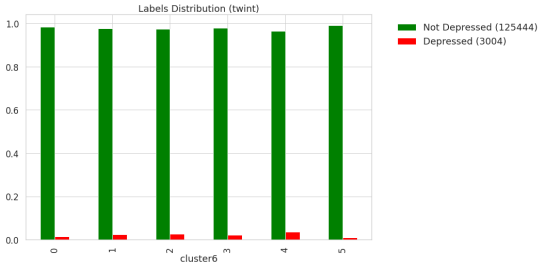
(b) VADER - k = 8



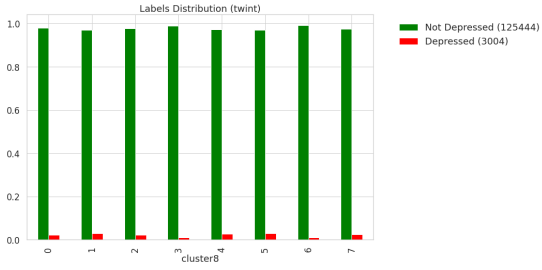
(c) NCRLex - k = 6



(d) NCRLex - k = 8



(e) TWINT - k = 6



(f) TWINT - k = 8

Figure 16: Labels distribution across clusters according to different labelling strategies.

7 CLPsych Dataset

We decided to explore a fourth labelling strategies by relying on the *2015 CLPsych Shared Task Dataset* (see [10] for more details).
To build this labelling strategy, hereinafter called *CLPsych strategy*, we started by preprocessing and extracting a list of features from tweets text. Then, we used these features to train a classifier to distinguish between depressed and not depressed tweets, and we marked each user as depressed or not depressed according to the percentage of depressed tweets that they have posted.
To measure our user-labelling strategy, we compared the labels over users with the ground truth provided in the original dataset.

In the following sections, we described the above-mentioned procedure more in details.

7.1 A first look at data

The dataset contains about 1.73 million tweets, labelled as depressed or not depressed according to whether they belong to a depressed user or not.
We decided to work with only $640K$ tweets ($320K$ per each class) to reduce the computational time.
We applied to *CLPsych* tweets the same preprocessing procedure described in Section 2. [2]
In Table 2 we report some examples of tweets *before* and *after* the preprocessing.

Original text	**Preprocessed text**
RT @lZAZj4tbdUN6alc: Why are girls so horny :)	whi girl horni
i am constantly being let down	constant let
@vRyWTyPU awee thank you!! ;) and im trying trust me	awe thank tri trust
God dammit school is soon	god dammit school soon

Table 2

7.2 Features Extraction

After data pre-processing, we feed our models with the features that reflect users' language habits in Twitter. To explore the users' linguistic usage in the tweets, we decided to look at specific features by taking inspiration from existing works on depression detection [11] [12] [10] [13].
In the following sections, we describe the feature extraction process more in detail.

[2] We slightly modified the list of stopwords by removing words related to COVID-19, and adding the term "rt" which means *Re-Tweet*, i.e. the act of taking someone's tweet and posting it as your own.

7.2.1 N-gram Features

N-gram modeling is used to examine the features from the posts. It is widely used in text mining and NLP as a feature for depression detection [14] to calculate the probability of co-occurrence of each input sentence as a unigram and bigram. For n-gram modelling we use the *term frequency-inverse document frequency* (TF-IDF) as a numeric statistic where the importance of a word with respect to each document in corpora is highlighted [15]. The main goal of its usage is to scale down the impact of empirically less informative tokens, which occur frequently to give a space for the more informative words occurring in a small fraction. The word is ranked with greater TF-IDF value if it is present in a particular post and absent in other post.

In our study, we use TF-IDF vectorizer from the scikit-learn Python library on depressed and not depressed tweets. For each class we restrict the term-document matrix to the first 100 most frequent unigrams and 100 most frequent bigrams[3]. Then, we filter out the n-grams in common to both depressed and control tweets, and we cast to integers their tf-idf values to reduce the noise and to give the same weights to those unigrams/bigrams that differ in tf-idf for less than 1 unit. Finally, we prepare a vector of fetaures by encoding each tweets with the weighted count of each pre-defined unigram and bigram. The weight assign to each occurrence of a unigram/bigram is exactly the tf-idf value of the unigram/bigram itself.

7.2.2 Language Predictors of Depression and Tweets Features

J.C.Eichstaedt et al. [16] showed that depression is marked by linguistic predictors such as increased perceptual processes, references to sadness and discrepancies or greater negative emotions.
Thus, we select a list of language predictors of depression and we count their occurrences into the original, un-preprocessed tweets by defining a custom transformer and relying on regular expressions.
Here is the list of language predictors organized in different semantic clusters:

- *self* : {I'm, I am, I was, I wasn't, my, mine, I, me}
- *feelings* :{I feel, I felt}
- *negation* : {no, not, no one, can't, cannot}
- *hopelessness* : {pointless, anyone else, need help, end, need, help}
- *sufferance* : {pain, hurt, want, die, stop, don't want, kill, mental illness, my head}

[3]Please refer to section *N-gram Features* of the notebook *CLPsych.ipynb* for the list of n-grams retrieved.

Human communication is not only verbal, but it occurs through means other than words, such as body language, gestures, and silence. The same happens when we communicate through social media platform, such as Twitter. In fact, tweets are not only made of plain words: users express their own feelings and ideas with punctuation, emoji, capital letters, etc. All these elements can be extracted and used as additional features to train our NLP classifier.
Therefore, we enhance our custom transformer with other regular expressions to extract the following list of features:

- number of words (n_words)
- number of uppercase words (n_capitals)
- number of question or exclamation marks (n_qe_marks)
- number of hashtags (n_hashtags)
- number of mentions to other Twitter accounts (n_mentions)
- number of links (n_urls)
- number of emoji (n_emoji)

Here are the ad-hoc regular expressions we use:

```
% number of words
\w+

% number of uppercase words
\b[A-Z]{2,}\b

% number of question or exclamation marks
!|\?

% number of hashtags
#\w+

% number of mentions
@\w+

% number of links
http.?:\/\/[^\s]+[\s]?

% number of emoji
_+[a-z_&]+_+
```

Regarding `n_emoji`, before counting the regex matches, we transformed each emoji into the corresponding string using the `demojize` function of module `emoji` with delimiters parameter set to `(__,__)`.
Thus, for example: Once we have extract both the language predictors and the tweets

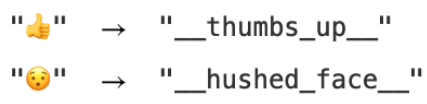


Figure 17: Example of emoji transformation.

features, we performed a *two sample Z-test* on each feature to verify whether the difference between the mean of the two populations (*depressed* and *not depressed* tweets) is statistically significant or not.
Our results show that all the features, except for *n_mentions*, are statistically significant with a confidence of 99%. Thus we keep them for our classification task, described in the next section.

7.3 Tweets Classification

The last step of our project was constituted by the training of classifiers, using as features those described in the previous paragraphs. In particular, the purpose of this section was to find the best combination of feature and classifier and validate the robustness of the strategy found, in order to understand if this could also be used on COVID19 tweets. The validation was carried out through the use of a strategy that allowed us to move from the knowledge of depressed users to the knowledge of depressed tweets. This step was made possible thanks to the use of the CLPsych dataset as ground truth, since in this dataset we know if a user is depressed or not and we have the texts of each user's tweets.

7.3.1 Search for the best strategy

In order to find the best classification strategy in terms of classifier (with its best hyperparameters) and features, different classifiers and different combinations of features were tested. The purpose of the classifiers was to predict whether the tweet in question was depressed or not depressed.
The different features passed in input to the classifiers are the following:

- Features (with "Features" we mean Ngram features + Language Predictors + Tweets Features described in the previous paragraph)
- Features + Vectorized Text (TF-IDF)
- Features + Vectorized Text (Bag of Words)
- Vectorized Text (TF-IDF)

- Vectorized Text (Bag of Words)

The different classifiers used are the following:

- Logistic Regression
- SVM
- Bernoulli Naive Bayes
- Random Forest

The implementations provided by Scikit-Learn[17] were used both for the classifiers and the Grid Search.

The hyperparameter tuning was done through the execution of a GridSearch without cross validation due to Google Colab parallelization errors.

The reference metric was the **accuracy**, which was considered reliable since particular attention was paid to making the dataset balanced.

The hyperparameters tested for each classifier were the following:

- Logistic Regression: max_iter = [100, 1000] [4]
- SVM: alpha = [0.1, 0.121, 0.142, 0.163, 0.184, 0.205, 0.226, 0.247, 0.268, 0.289, 0.310, 0.331, 0.352, 0.373, 0.394, 0.415, 0.436, 0.457, 0.478, 0.5] [5]
- Bernoulli Naive Bayes ran with its default parameters
- Random Forest was run with max_depth = 2, random_state = 0. A more accurate search could not be made due to the long run time.[6]

[4] **max_iter** = maximum number of iterations taken for the solvers to converge

[5] **alpha** = constant that multiplies the regularization term. The higher the value, the stronger the regularization. Also used to compute the learning rate when set to learning_rate is set to 'optimal'.

[6] **max_depth** = the maximum depth of the tree. **random_state** = controls both the randomness of the bootstrapping of the samples used when building trees (if bootstrap=True) and the sampling of the features to consider when looking for the best split at each node (if max_features < n_features).

	Logistic Regression	SVM	Bernoulli Naive Bayes	Random Forest
Features	accuracy = 0.565 max_iter = 1000	accuracy = 0.543 alpha=0.14210526315789473	accuracy = 0.560	accuracy = 0.569
Features + TF-IDF	accuracy = 0.583 max_iter = 1000	accuracy = 0.554 alpha = 0.1	accuracy = 0.652	accuracy = 0.559
TF-IDF	accuracy = 0.650 max_iter = 1000	accuracy = 0.502 alpha = 0.2894736842105263	accuracy = 0.649	accuracy = 0.548
Features + BoW	accuracy = 0.599 max_iter = 1000	accuracy = 0.557 alpha = 0.1631578947368421	accuracy = 0.652	accuracy = 0.557
BoW	accuracy = 0.651 max_iter = 1000	accuracy = 0.511 alpha = 0.2894736842105263	accuracy = 0.649	accuracy = 0.547

Table 3: GridSearch results. The highlighted cells represent the strategies with which the best results were achieved.

7.3.2 Considerations about the results

A more accurate analysis of the parameters could have been carried out, but this was beyond the scope of this research. In particular, we focused more on carrying out experiments to understand the trend of the results and see if it was possible to consider the features chosen as significant for our task. As the results show, it was not possible to achieve a high level of accuracy, probably due to the complexity of the task itself.
In any case, we report that the best result, as shown in the table above, was achieved through the use of the following strategies:

- Bernoulli Naive Bayes + Features + TF-IDF
- Bernoulli Naive Bayes + Features + BoW

In particular, in both cases the same level of accuracy was achieved, therefore for the subsequent analyzes it was decided to keep only one of the two strategies, that is the first of the two listed above.

7.3.3 Validation: from depressed tweets to depressed users

In order to carry out the validation phase, the strategy that was found to be the best in the previous section was used.

Prediction
After running the classifier on the training set we predicted the label that indicates the state of depression for each tweet. At this point we had the **label** associated with each tweet (CLPsych User → Tweet) **based on the user's depression status** (if the user was characterized as depressed, the CLPsychs label has been set to be equal to **1** for all tweets of that user) and the **label** associated with each tweet **based on the prediction of the classifier**.

Validation
To evaluate the validity of our approach we counted, for each group of tweets of each user, the number of predictions on the single tweets that corresponded to the real label indicating the user's status. We then set threshold values to see how accurate our classifier was in the task of carrying out the **reverse process**, that was to **pass from the knowledge of the depression state of the tweet to the knowledge of the user's depression state (gold-standard)**.

Results
From the analysis carried out it emerged that:

- with a threshold of 50%, 40.34% of users are classified correctly
- with a threshold of 60%, 12.88% of users are classified correctly
- with a threshold of 70%, 8.99% of users are classified correctly
- with a threshold of 80%, 5.83% of users are classified correctly
- with a 90% threshold, 4.74% of users are classified correctly

7.3.4 Considerations about the results

The accuracy of the classification is **too low** to be able to generalize this method to predict depression on COVID19 tweets.
Our method includes several steps and in each of them we can highlight some critical issues that may have led to a similar result:

- **Features**: we took inspiration from existing works related to the detection of depression within textual content and we built a strategy based on taking the most frequent 100 ngrams and on linguistic predictors that we built manually. It is possible that there are more effective features for carrying out this task and it would certainly be interesting to see if by changing the strategy used for the extraction of the features it is possible to observe a drastic change in the results.
- **Classifiers**: we decided to narrow our search to 4 classifiers to be able to give a rough evaluation of the problem. It could be that, without changing the features, but simply changing the model (moving for example to a neural model) we could observe a change in the results. Therefore, it could be that the classifiers chosen are not the ideal ones for carrying out this task.
- **Hyperparameters of Classifiers**: we performed the Grid Search on a small group of hyper parameters for each classifier. It could be that by analyzing further values and further hyper parameters the results change.

- **Threshold**: in order to be able to carry out the assessment, we considered a very small group of threshold values. It would be interesting to perform the analysis over a wider range and to study the precise trend of the results as the threshold varies.

- **Dataset CLPsych**: surely this is the main problem. In order to be able to perform our analysis using the dataset in question we were forced to make a very strong **assumption**: **a depressed user only posts depressed content**. It proved necessary to make this assumption because the CLPsych dataset was built with an objective opposite to ours: while we want to assign a label to the tweet, the authors of the dataset have assigned a label to the users. Therefore, it could be that by assigning the user's label to all his tweets, noise has been generated which made it impossible to carry out the correct classification task. In other words, it is not certain that depressed users only post depressed things and since our job is the "Automatic Detection of Depression" within the tweets, this noise could completely mislead our analysis. Surely therefore, this forcing made to be able to obtain a ground truth did not actually allow us to obtain a robust and reliable ground truth.

8 Conclusions and Future Works

Although we did not completely accomplish the task, this work is certainly a good starting point to find a suitable strategy for detecting depression in social media.
Besides the inner complexity of the task, one of the main critical points was the absence of a ground truth dataset to validate our labelling strategies.
Moreover, the large number of variables involved would have required a complete and exhaustive analysis to ensure the correctness of the results.
As a final remark, the lack of domain expertise forced us to make assumptions that are not always verifiable, such as the definition of the features for CLPsych dataset or the assumption made on the threshold for detecting depression through the number of depressed tweets.

It would be very interesting to go and run the type of analysis conducted in this research again taking into account the following factors:

- **Domain Experts**: it would be interesting to involve a heterogeneous group of domain experts (such as psychologists, linguists etc.) in order to be able to build more robust linguistic features by selecting those terms and expressions that may characterize subjects suffering from depression.

- **Temporal Dataset**: it would be necessary to extend our analysis to include the temporal dimension as well, since depression is a phenomenon characterized by different phases and certainly not limited to a single moment. Thus, it could be interesting to observe if among the tweets of a certain user there are also some temporal patterns in the use of terms and expressions. For example, a depressed user may posts specific contents on particular days that remind him a negative event.

- **Age/Gender Analysis**: people of different ages and genders may express themselves differently on social media. It would be therefore interesting to carry out this analysis by differentiating tweets based on age groups and gender, in order to observe whether, by creating ad-hoc linguistic predictors for each group, it is possible to produce a more robust analysis. Undoubtedly, this proposal is very ambitious as linguistic characteristics could be influenced not only by gender and age, but also by historical period and social contexts. The involvement of a heterogeneous group of domain experts is therefore also necessary here.

References

[1] Clayton J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Eytan Adar, Paul Resnick, Munmun De Choudhury, Bernie Hogan, and Alice H. Oh, editors, *ICWSM*. The AAAI Press, 2014.

[2] David Blei, Andrew Ng, and Michael Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 05 2003.

[3] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. WSDM '15, page 399–408, New York, NY, USA, 2015. Association for Computing Machinery.

[4] Gensim: topic modelling for humans, Aug 2021.

[5] OSINT Team. Twint project. https://github.com/twintproject/twint.

[6] Saif Mohammad and Peter Turney. Nrc emotion lexicon. 01 2013.

[7] Tina Donvito. 13 common words and phrases that may signal depression, Jan 2021.

[8] Nrc emotion lexicon.

[9] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*, 2002.

[10] Glen Coppersmith, Mark Dredze, Craig Harman, Kristy Hollingshead, and Margaret Mitchell. CLPsych 2015 shared task: Depression and PTSD on Twitter. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 31–39, Denver, Colorado, June 5 2015. Association for Computational Linguistics.

[11] Adonay Zenebe Resom, Maurice Anthony Flannery, Jerry Xu Assan, Yufei Gao, and Yuxin Wu. Technical report.

[12] Moin Nadeem. Identifying depression on twitter, 2016.

[13] Michael M. Tadesse, Hongfei Lin, Bo Xu, and Liang Yang. Detection of depression-related posts in reddit social media forum. *IEEE Access*, 7:44883–44893, 2019.

[14] Daniel Preoţiuc-Pietro, Johannes Eichstaedt, Gregory Park, Maarten Sap, Laura Smith, Victoria Tobolsky, H. Andrew Schwartz, and Lyle Ungar. The role of personality, age, and gender in tweeting about mental illness. In *Proceedings of*

the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality, pages 21–30, Denver, Colorado, June 5 2015. Association for Computational Linguistics.

[15] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing Management*, 24(5):513–523, 1988.

[16] Johannes C. Eichstaedt, Robert J. Smith, Raina M. Merchant, Lyle H. Ungar, Patrick Crutchley, Daniel Preoţiuc-Pietro, David A. Asch, and H. Andrew Schwartz. Facebook language predicts depression in medical records. *Proceedings of the National Academy of Sciences*, 115(44):11203–11208, 2018.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

A Figures Section 2

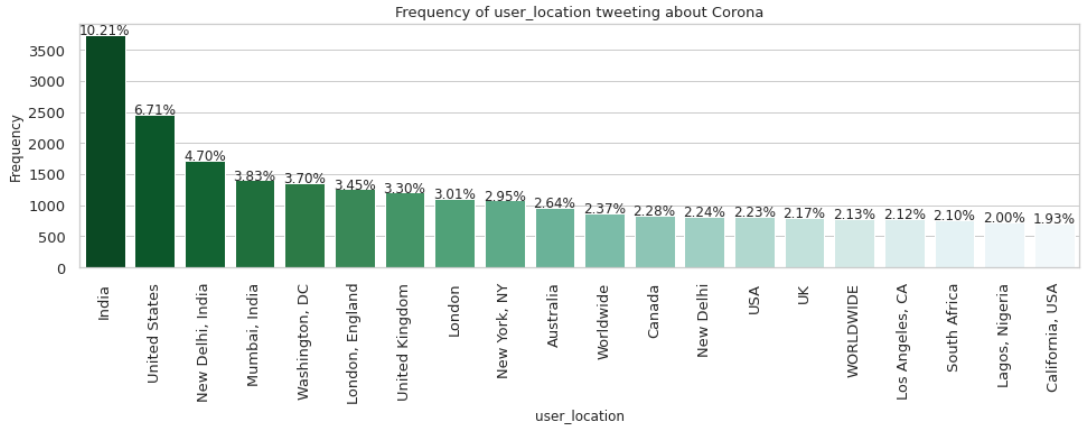


Figure 18: The top 20 most frequent values of `user_location`.

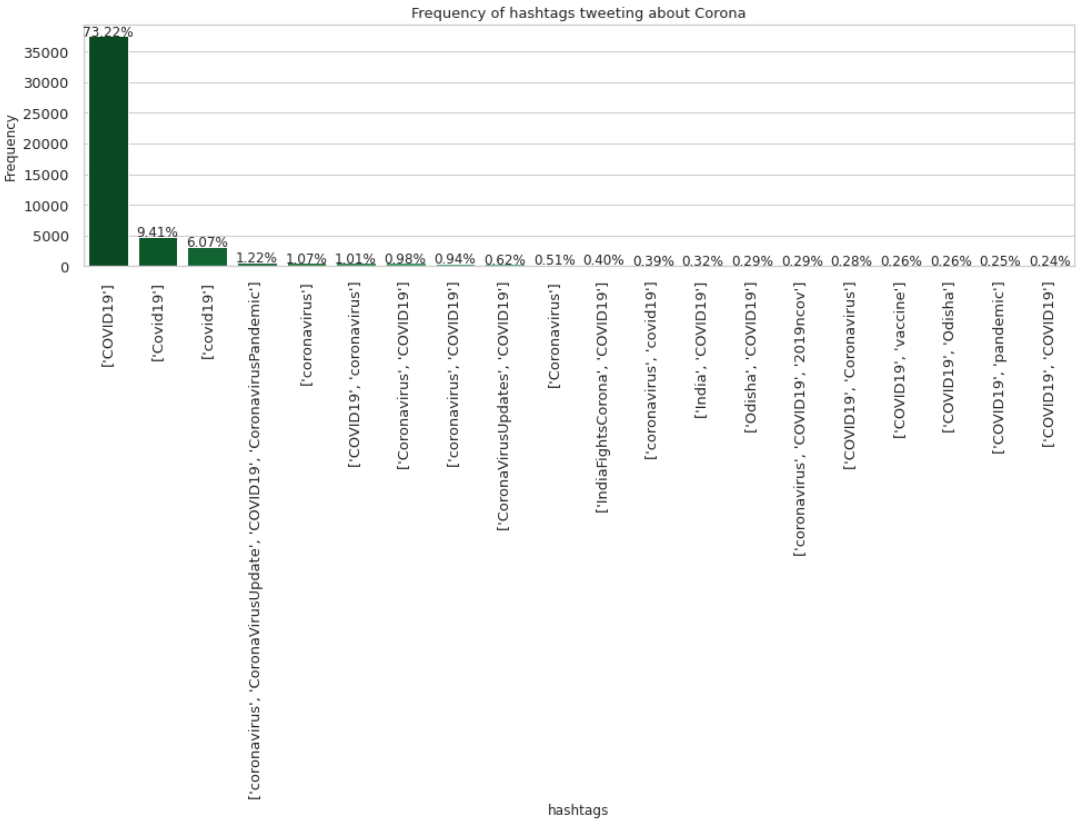


Figure 19: The top 20 most frequent values of `hashtags`.

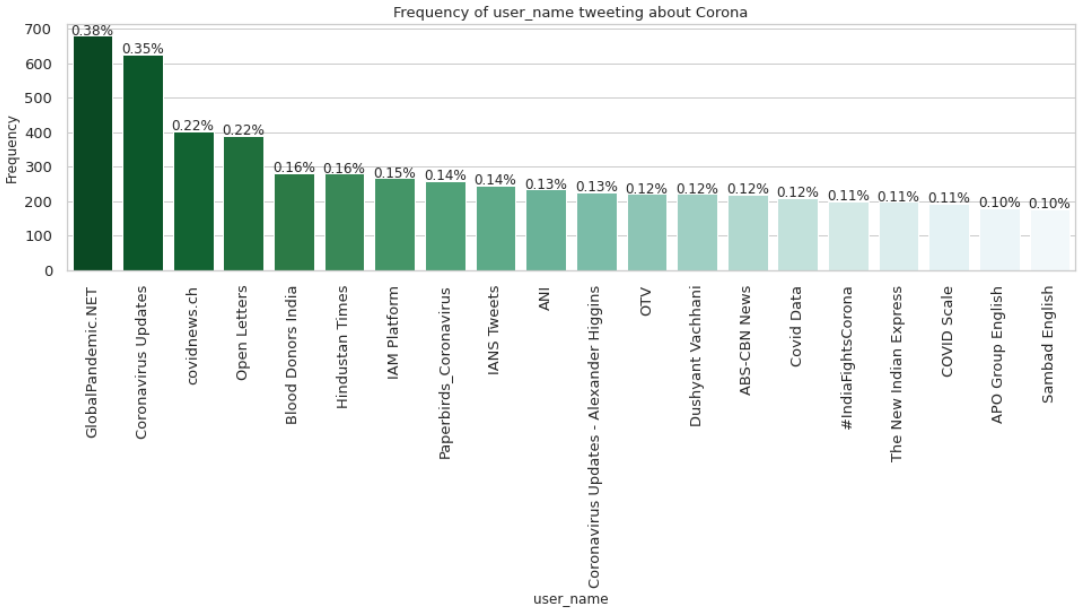


Figure 20: The top 20 most frequent values of `user_name` before filtering.

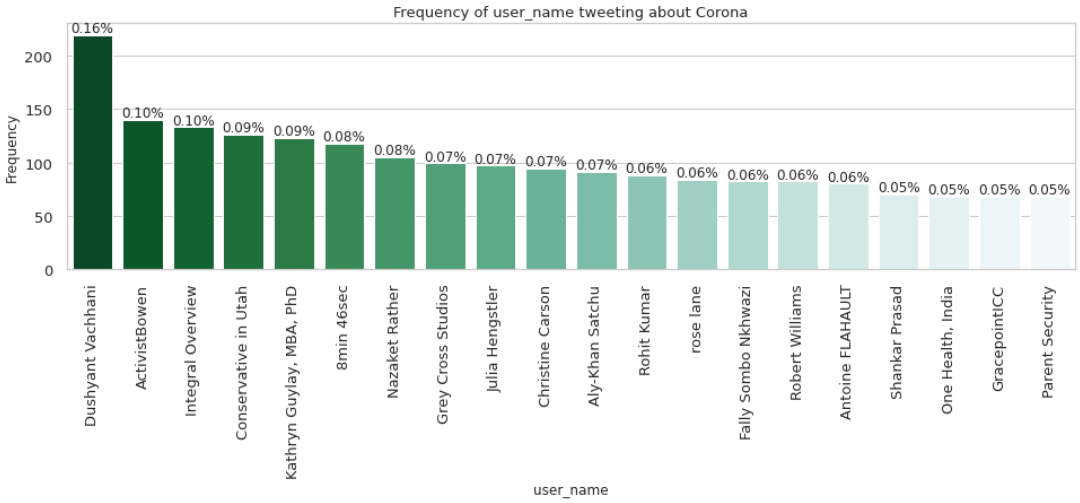


Figure 21: The top 20 most frequent values of `user_name` after filtering.

B Figures Section 3.3.2

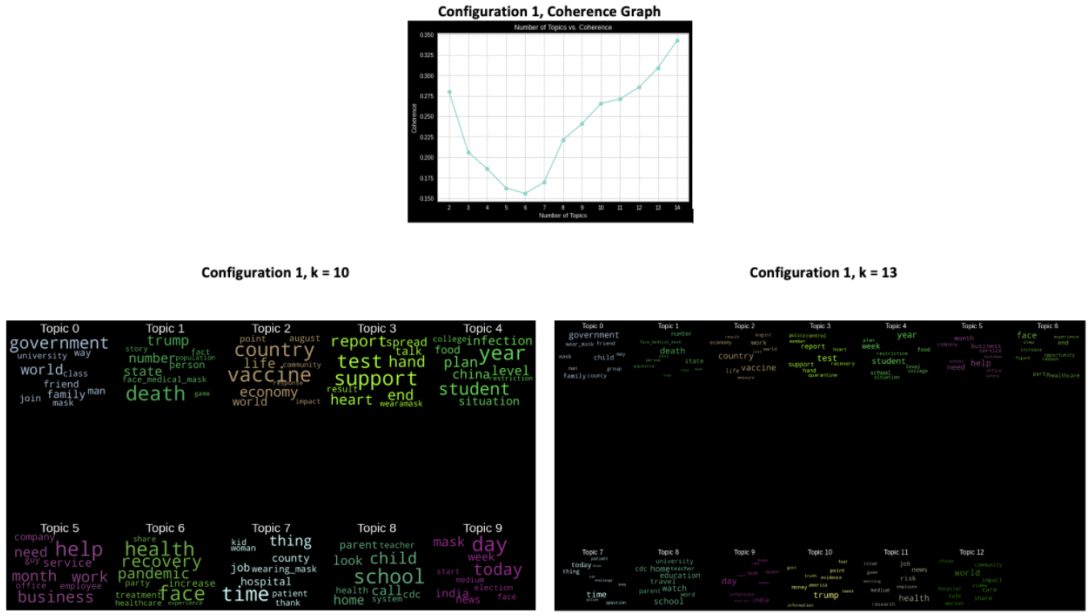
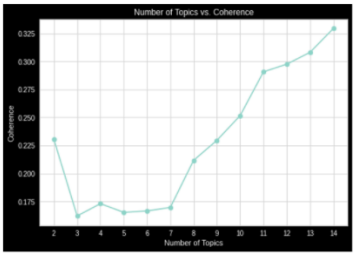


Figure 22

Configuration 2, Coherence Graph

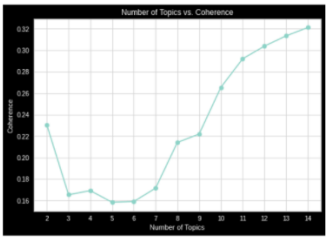


Configuration 2, k = 11

Configuration 2, k = 13

Figure 23

Configuration 3, Coherence Graph



Configuration 3, k = 8

Configuration 3, k = 11

Configuration 3, k = 13

Figure 24

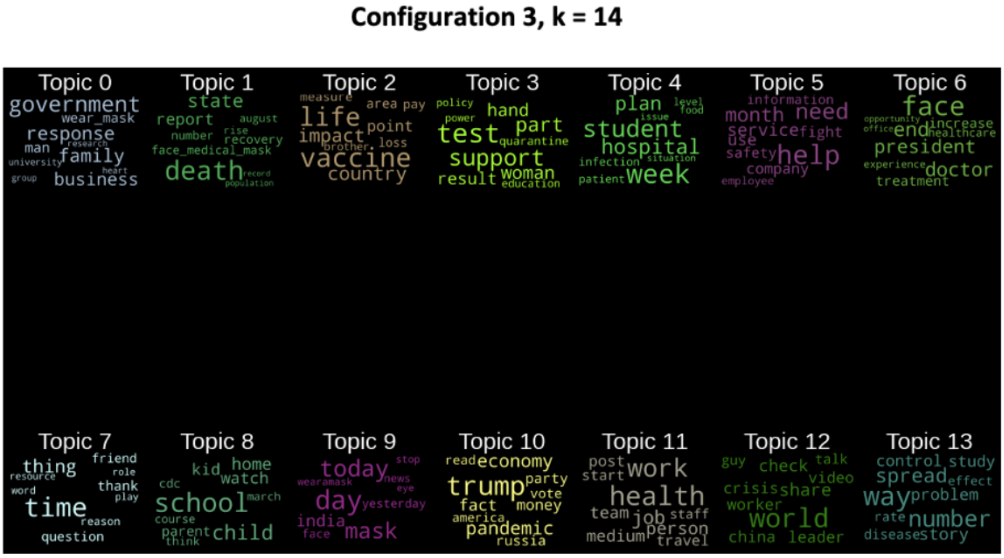


Figure 25

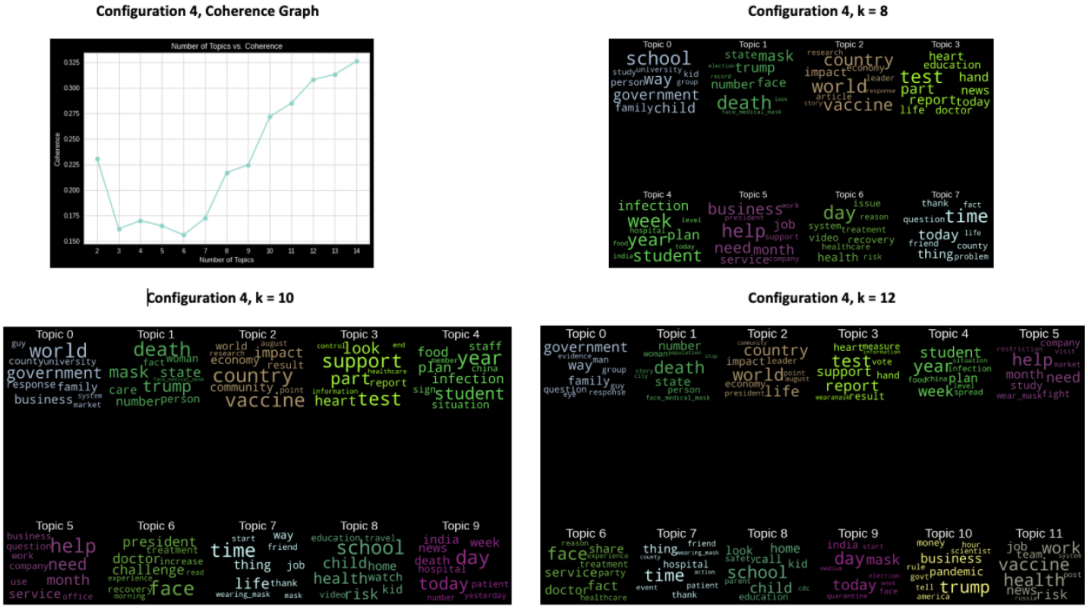


Figure 26

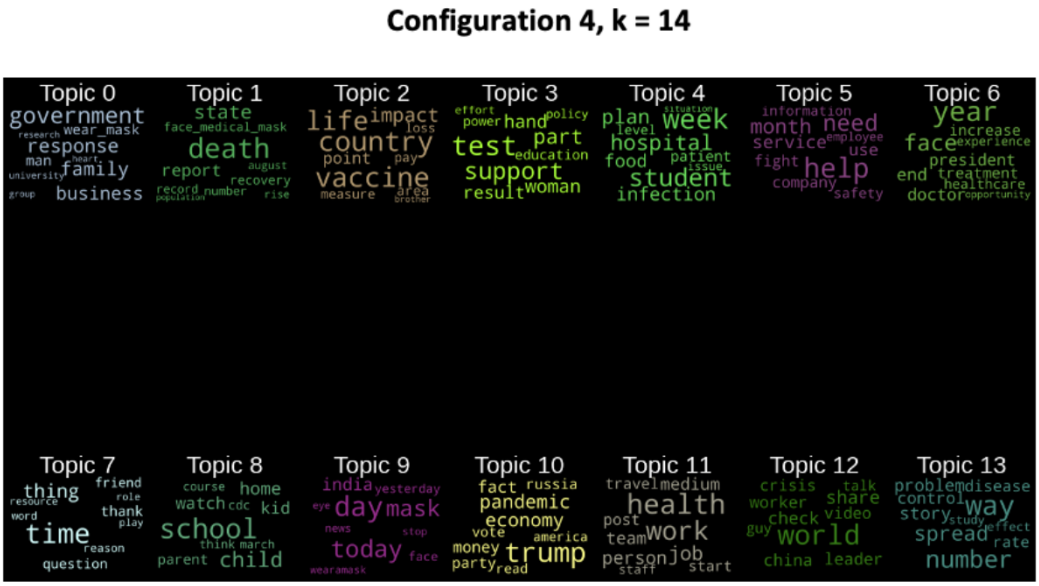


Figure 27

C Figures Section 3.4.4

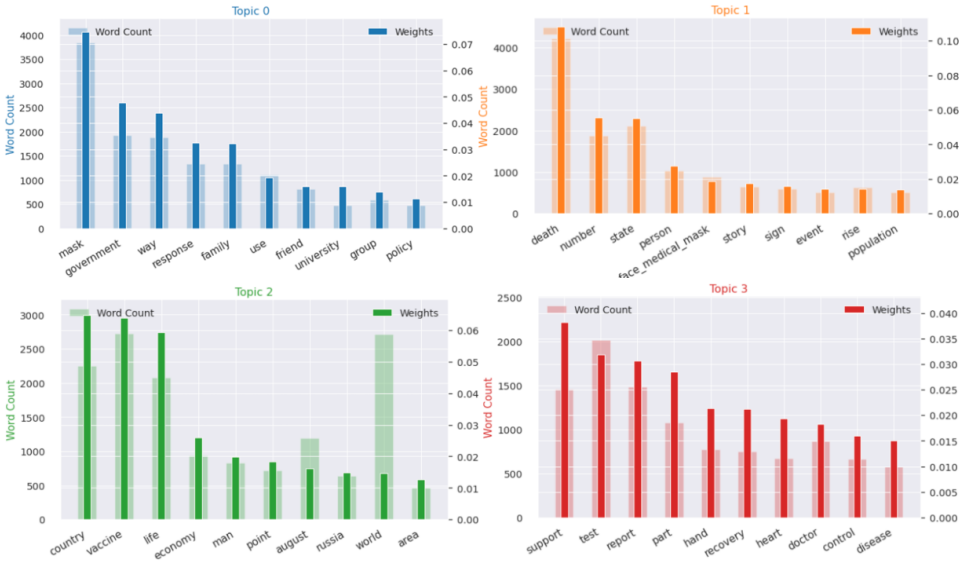


Figure 28: Word Count of Topic Keywords - 1

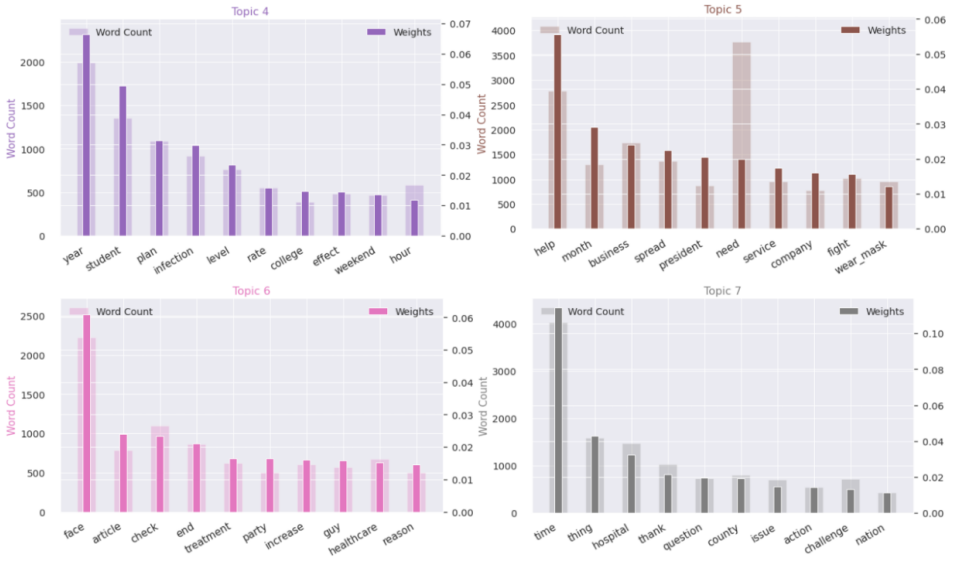


Figure 29: Word Count of Topic Keywords - 2

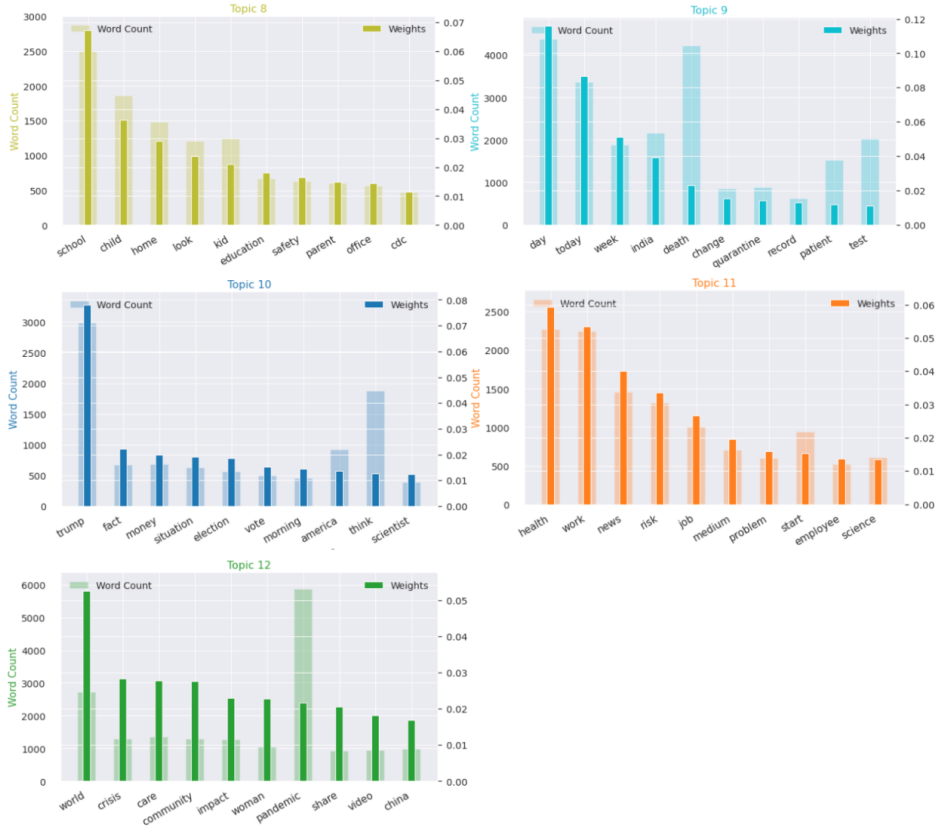
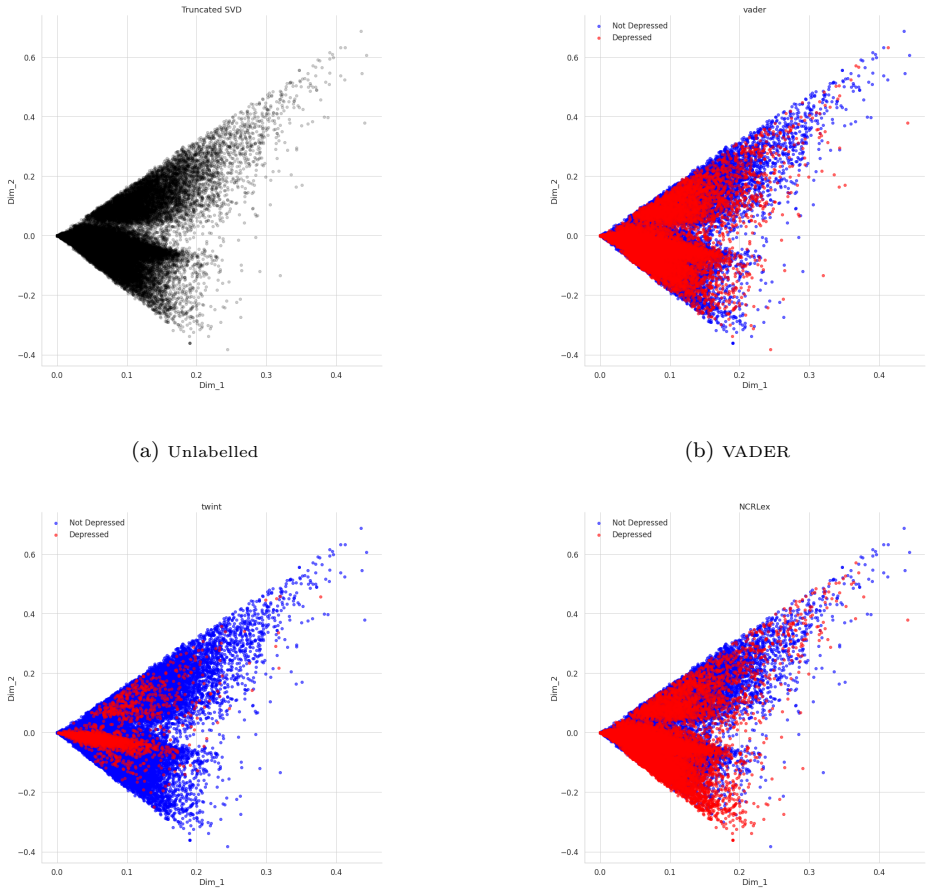


Figure 30: Word Count of Topic Keywords - 3

D Figures Section 6



(a) Unlabelled

(b) VADER

(c) TWINT

(d) NCRLex

Figure 31: 2-dimensional representation of our corpus of tweets labelled using different labelling startegies.

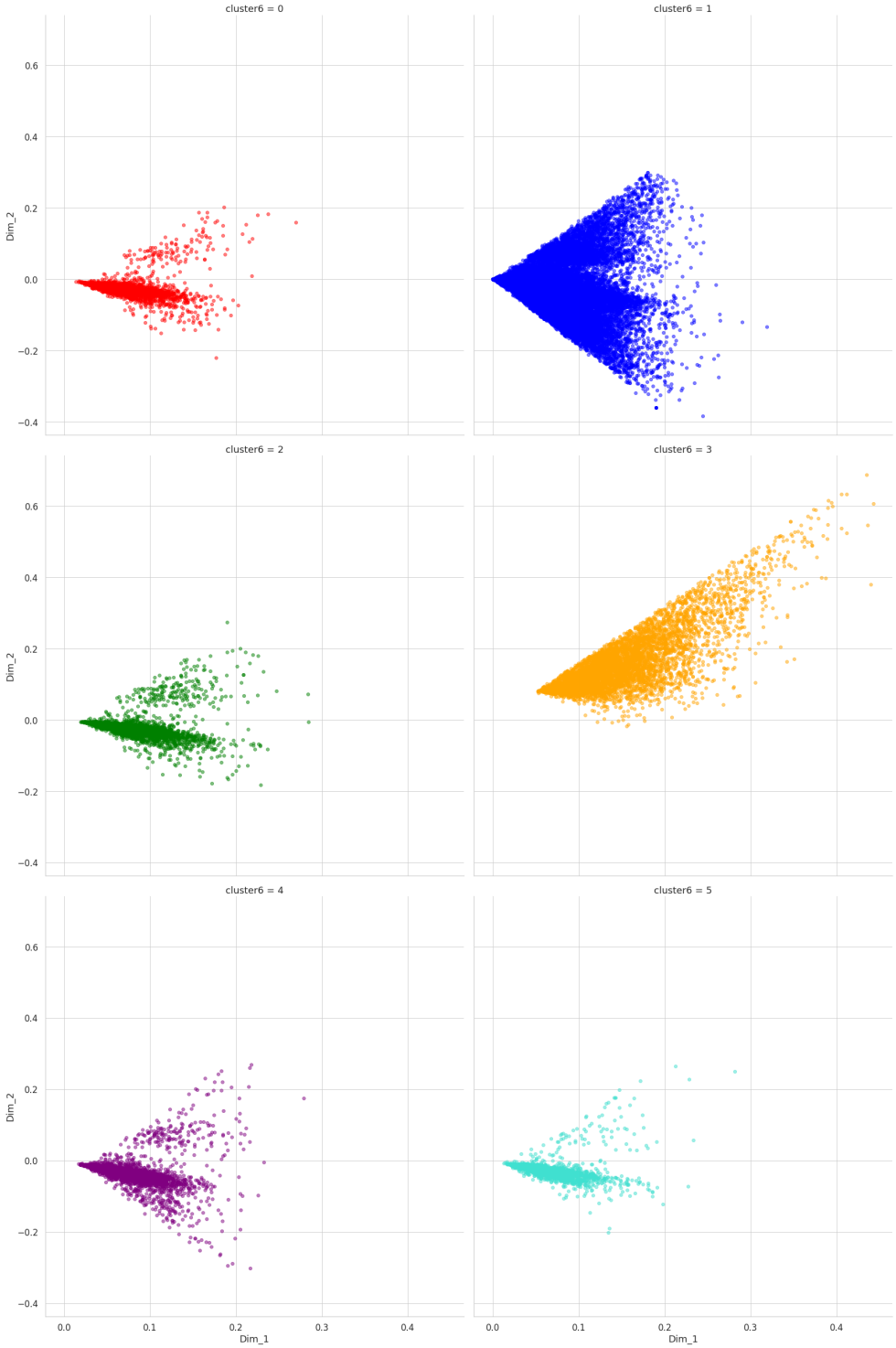


Figure 32: Visualization of each single cluster ($k = 6$).

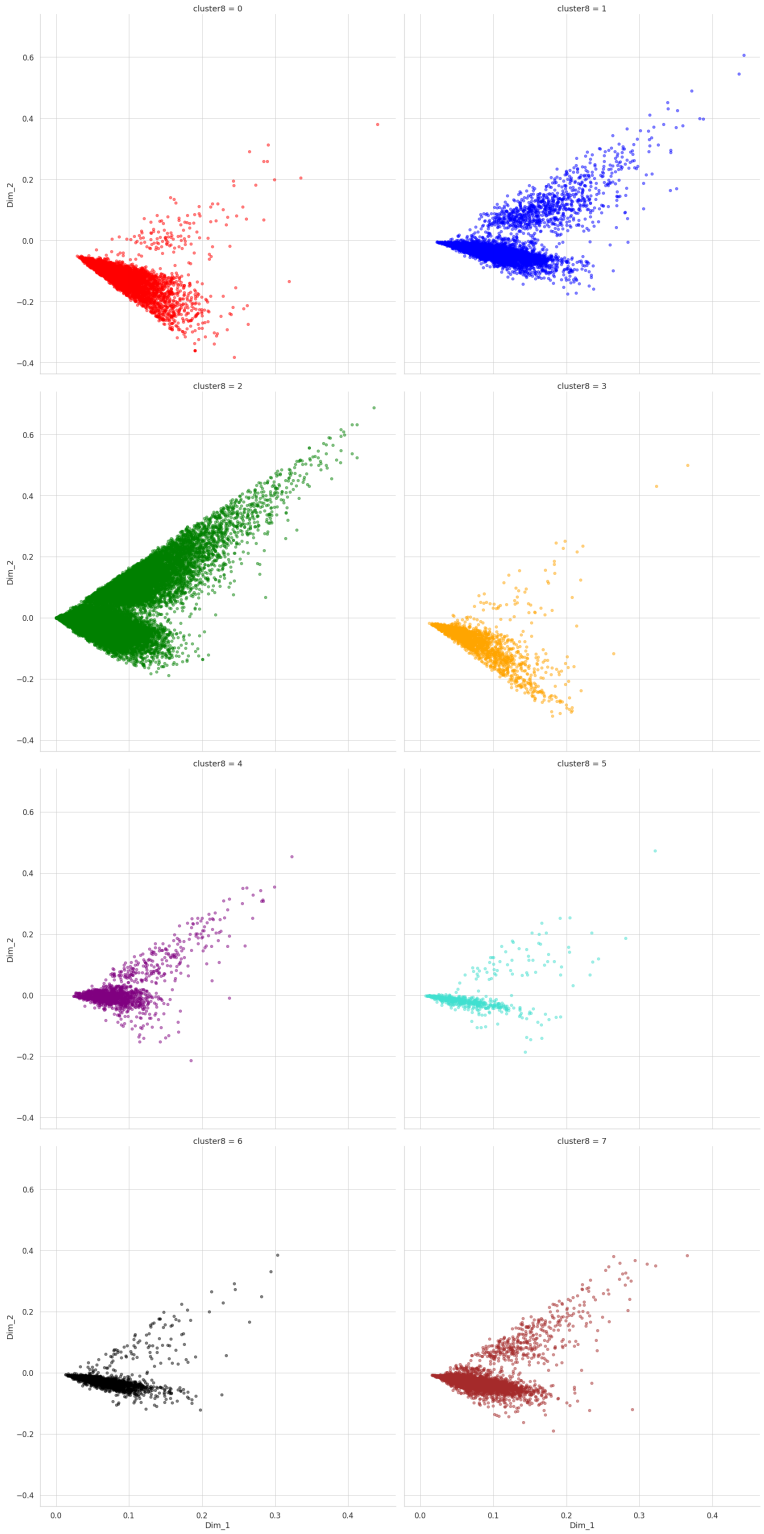


Figure 33: Visualization of each single cluster ($k = 8$).

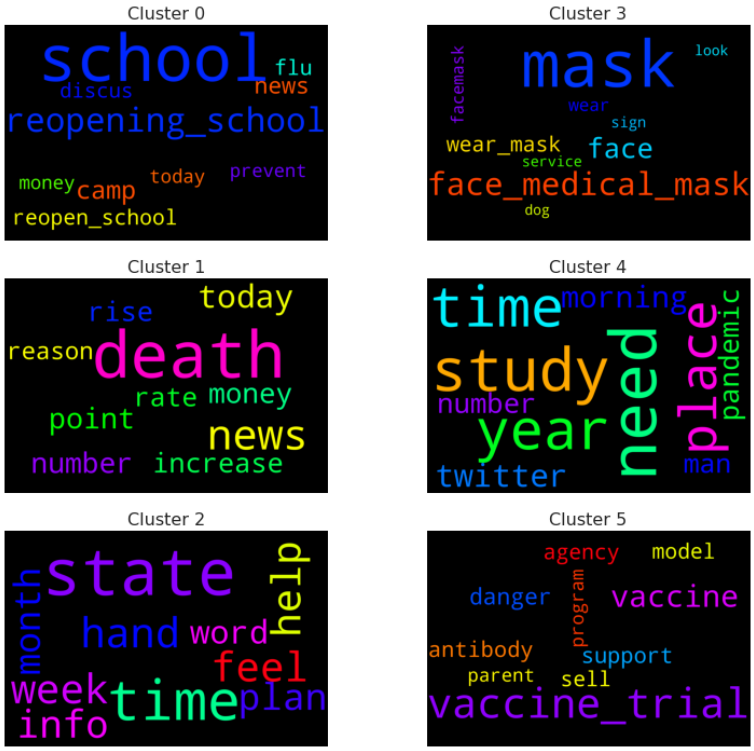


Figure 34: Single cluster word cloud ($k = 6$).

Figure 35: Single cluster word cloud ($k = 8$).

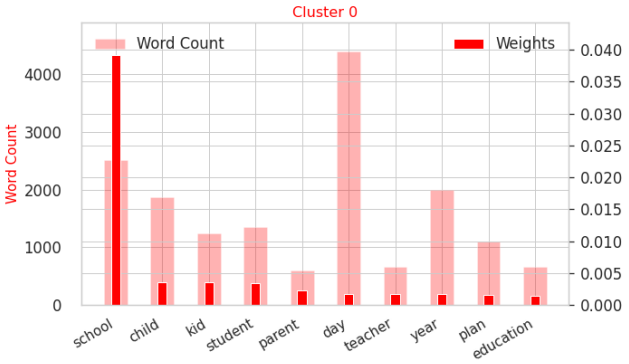


Figure 36

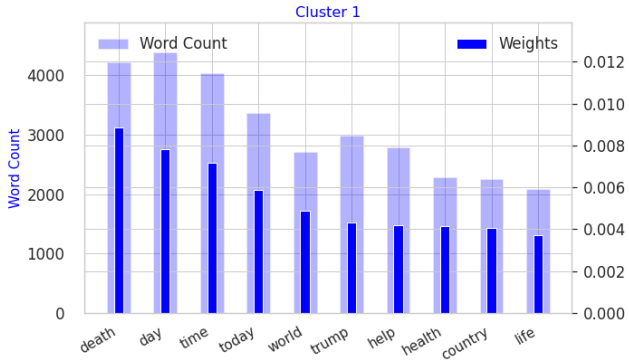


Figure 37

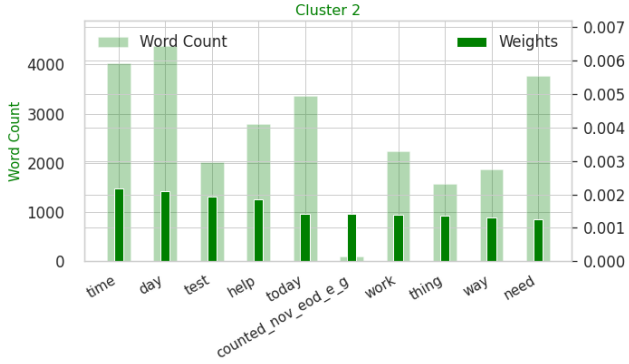


Figure 38

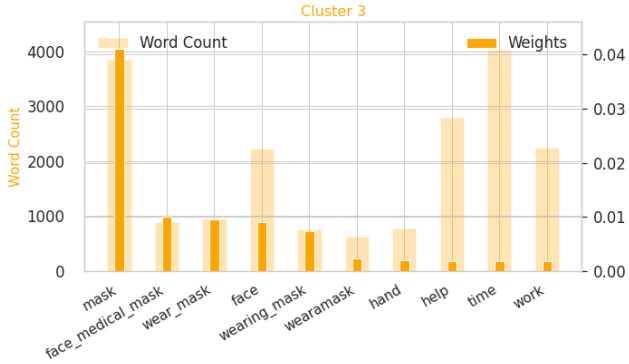


Figure 39

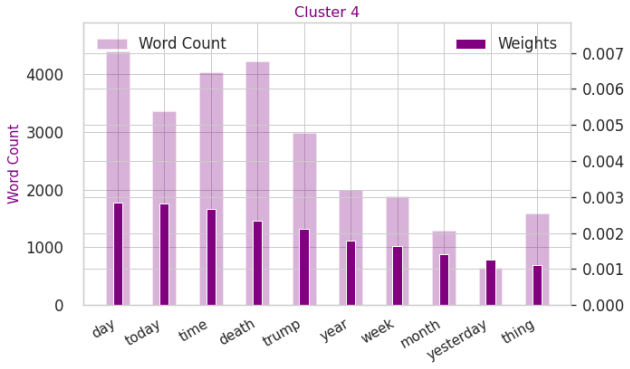


Figure 40

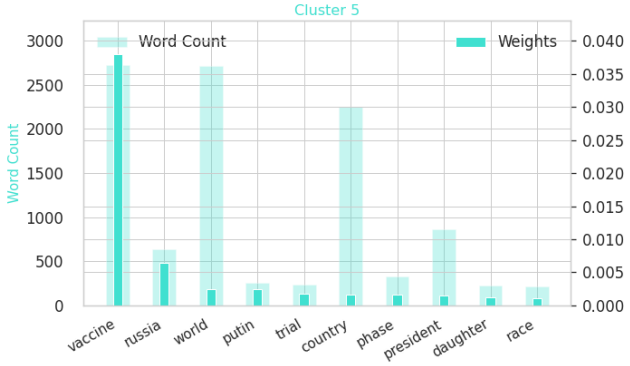


Figure 41

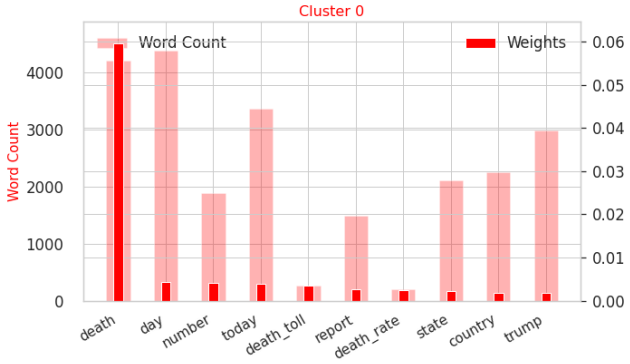


Figure 42

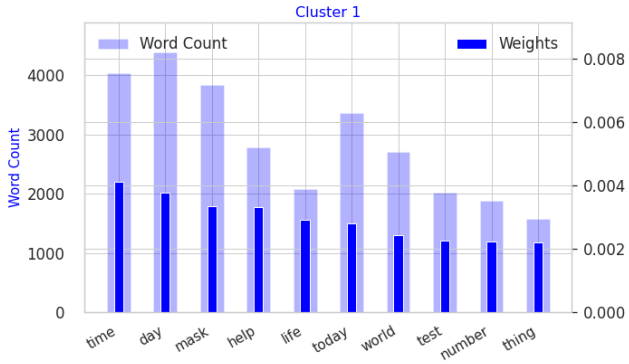


Figure 43

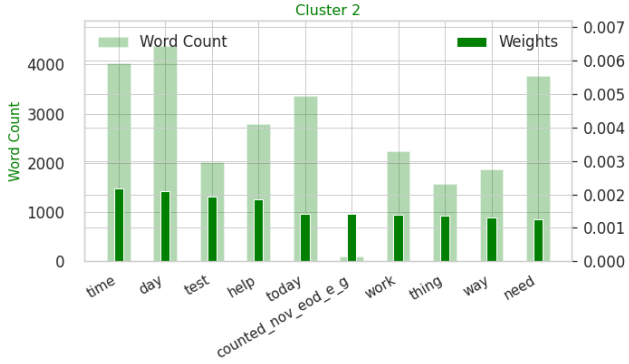


Figure 44

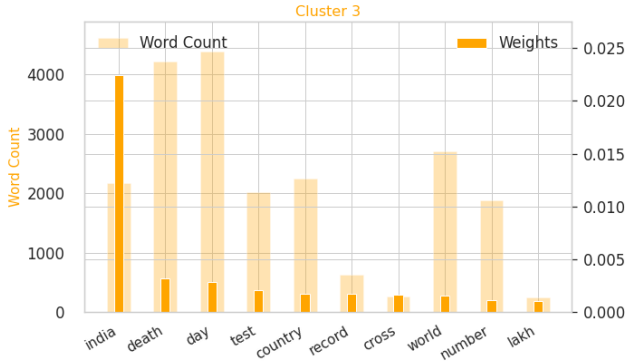


Figure 45

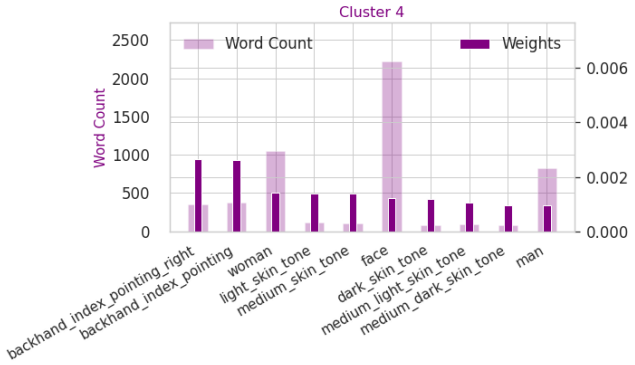


Figure 46

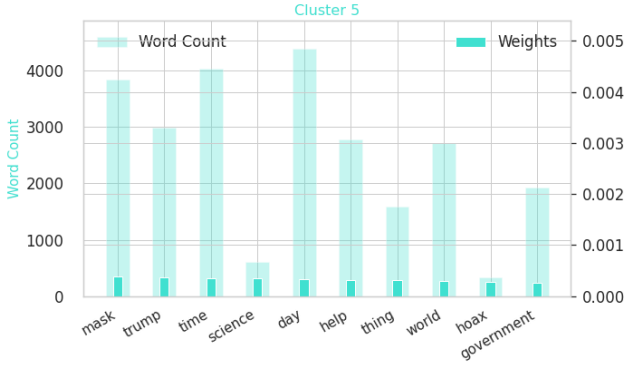


Figure 47

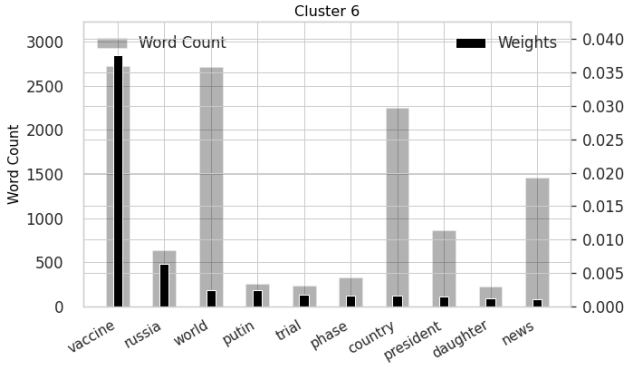


Figure 48

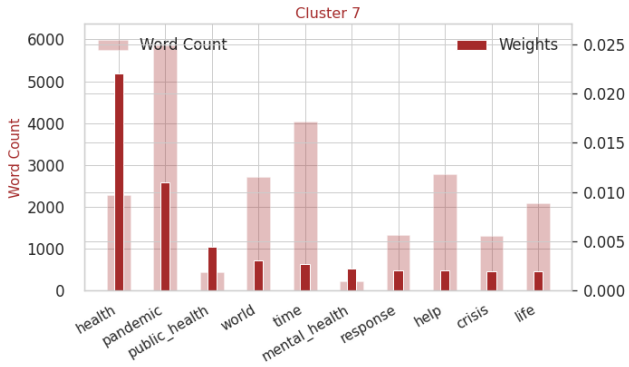

Cluster 7
Word Count
Weights
Word Count
6000
5000
4000
3000
2000
1000
0
0.025
0.020
0.015
0.010
0.005
0.000
health
pandemic
public_health
world
time
mental_health
response
help
crisis
life

Figure 49