**Contents**

SERIAL NO. CONTENTS PAGE NO

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: LITERATURE SURVEY 2

CHAPTER 3: PROJECT FORMULATION 4

CHAPTER 4: CHATBOT 5

CHAPTER 5: DATA AGGREGATION 8

CHAPTER 6: DATA ANALYSIS 10

CHAPTER 7: BUILDING THE BOT 14

CHAPTER 8: RESULT ANALYSIS 26

REFERENCES 27

**CHAPTER 1**

INTRODUCTION

We are all together in a fight against the COVID-19 pandemic. Chatbots, if effectively designed and deployed, could help us by sharing up-to-date information quickly, encouraging desired health impacting behaviors, and lessening the psychological damage caused by fear and isolation. Despite this potential, the risk of amplifying misinformation and the lack of prior effectiveness research is cause for concern. Immediate collaborations between healthcare workers, companies, academics and governments are merited and may aid future pandemic preparedness efforts.

During the novel coronavirus (COVID-19) pandemic, institutions like the Centers for Disease Control and Prevention (CDC) and the World Health Organization (WHO) have begun utilizing chatbots to share information, suggest behavior, and offer emotional support. The CDC has named theirs “Clara”. Chatbots are software programs that talk with people through voice or text in their natural language. Some well-known examples include “Alexa” from Amazon, “Siri” from Apple, and “Cortana” from Microsoft. They often come pre-installed on smartphones or home-based smart speakers. In recent years, chatbot use for health-related purposes has increased considerably, from supporting clinicians with clinical interviews and diagnosis to aiding consumers in self-managing chronic conditions. While promising, the use of chatbots may pose safety risks. Chatbots have varied widely in their responses to questions about physical health, suicide, intimate partner violence, substance abuse, and other sensitive conversations. In one study, about a third (29%) of chatbot responses to health questions could have caused harm, and about half of those (16%) could have resulted in death if acted upon. The COVID-19 pandemic puts in stark relief the potential for chatbots to help save lives.

**CHAPTER 2**

LITERATURE SURVEY

# We examined many publications from the last five years, which are related to chatbots. Then we presented different related works to our subject, and the AI concepts needed to build an intelligent conversational agent based on NLP. Finally, we presented a functional architecture that we propose to build an intelligent chatbot for health care assistance.

# Journal 1: Data Analysis by Web Scraping using Python

# To know how the data extraction process has evolved has so much one must understand the techniques involved in this method of web scraping is important scraping has been around nearly as long as the web. Web scraping, in a way also makes the project independent of other non-reliable datasets too as the dataset is created mostly from trusted website by the programmer.

# Extraction and Analysis of information are generally utilized by the Digital distributers and catalogues, Travel, Real home, and E-trade. Then again, examination and figuring come path back with the advances in accumulation components and the innovation of Real Databases: The data had been seen and dealt with as data to be set up for data examination.

# Python has especially designed libraries for the purpose of scrapping like BeautifulSoup and Scrapy .

# Journal 2: A Smart Chatbot Architecture based NLP and Machine Learning for Health Care Assistance

In this paper, they performed a detailed survey on some recent literature. They examined many publications from the last five years, which are related to chatbots. They presented different related works to our subject, and the AI concepts needed to build an intelligent conversational agent based on NLP and DL models Finally, they presented a functional architecture that they propose to build an chatbot for health care assistance.

The mean research topics in natural language processing (NLP) are user intention identification and Information extraction. In past years researchers presented several models. Recently the

development of Artificial intelligence and especially deep learning and deep neural network models have helped a lot in building self-learning chatbots. However, several attempts have

been made to treat the seq2seq model problems with the help of deep learning concepts such as deep neural networks (DNN), recurrent neural networks (RNN), and convolutional neural

networks (CNN).

Journal 3: Programming challenges of Chatbot: Current and Future Prospective

A chatbot is an instant messaging account that able to provide services using instant messaging frameworks with the aim of providing conversational services to users in an efficient manner.

CHATBOT PROGRAMMING CHALLENGES

There are a lot of challenges which are associated with chatbots. Some of them are as follows. A. Natural language processing the first and foremost challenge of the chatbot is to handle NLP issue by mastering their syntax. If we ask them that " what's the weather?". You will get an answer but what if we ask "Could you check the weather?" you might not get the proper answer. Such type of programming issues falls in natural language processing category which is a key focus for the companies like Facebook, Google with Deep Text and Syntax Net respectively. B. Machine learning Getting NLP is one aspect of designing and development of Chatbots while Machine Learning is another aspect of the Chatbot design and development. Our computer systems should able to learn the correct response should be which can be achieved with efficient programming with AI concepts

LIMITATION AND FUTURE OF NLP AND MACHINE LEARNING

As per our discussion, it is quite clear that chatbot needs to provide vast logic and linguistic resources which are input, output and entities phrases. Chatbot with complex queries handling need high attention in using singular and plural forms, need to take care of synonyms, hyponyms, and finally, the sentimental analysis should be done carefully

A simple chatbot is not a challenging task as compared to complex chatbots and developers should understand and consider the stability, scalability and flexibility issues along with high level of intention on human language. In short, Chatbot is ecosystem and moving quite fast and with the passage of time new features are added in the existing platform. Recent advancements in the machine learning techniques may able to handle complex conversation issue such as payments correctly.

**CHAPTER 3**

PROBLEM FORMULATION

In an effort to better inform the world about CODIV-19, the chat bot will be offering instant and accurate statistics and information about COVID-19 easily.

**CHAPTER 4**

CHATBOT

What is a chatbot?

Chatbot is an artificial Intelligence’s application that tries to understand the customer’s needs and then assist them to perform a particular task like answering queries, transactions, form submission, bookings etc.

As the name suggests it is a machine that chats with you. The trick though is to make it as human-like as possible.

What are the types of chatbot?

There are two types of chatbots:

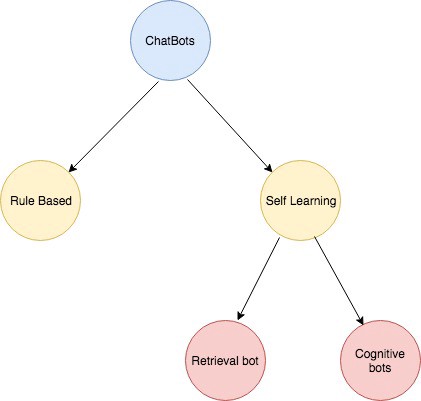
1.) Rule Based: In this approach, bot answers the questions based on some rules on which it trained on but it wouldn’t be able to manage complex queries.

2.) Self Learning: In this approach, we use Machine Learning-based approaches which are more efficient than rule base approach.

**Self-Learning Bots**are classified further as **Retrieval bots** and **Cognitive bots**

**Retrieval bots**are bots which doesn’t have cognitive ability but on the basic of context and messages of the conversation it would selects the best response from a predefined response. Response can be generated from rule based if-else condition to machine learning cluster and classifiers.

**Cognitive bots**are bots which has cognitive ability which can generate the answers and provides different answers. These are more intelligent bots.



**FIG 1 -TYPES OF CHATBOT**

# However, in this report, we would look into simple retrieval chatbot to understand the working of chatbot using python.

# Benefits of a chatbot

Using a chatbot offers many benefits for both the consumer and the producer.

First of all, the benefits for the consumer.

* Proactive**:** the chatbot can start a conversation with a consumer based on, for example, his or her time spent on a website. It is possible that a consumer has been on a website for a long time and cannot find what he/she is looking for, at this moment a chatbot that asks whether you could use some help can come in handy.
* Direct answer**:** the chatbot generates an answer within seconds, while telephone customer service or mail often has a longer waiting time.
* Available 24/7**:** the chatbot does not have to sleep or take a break, this means that there is an available service at any time of the day.
* Relevant information: the chatbot can directly guide a consumer and help find relevant information. For example, you can avoid a long search on an extensive website by asking the chatbot for management.
* Involvement: the chatbot can show attractive pictures, videos, GIFs during a conversation, which stimulates the consumer to get more involved in the conversation.

Now the benefits for the producer:

* Cost reduction: one of the most powerful benefits of a chatbot is that it saves costs (provided the bot works well). It offers cheap and fast customer service and prevents extreme waiting times. This means that a company can free up time for more complex questions and may require less manpower for customer service.
* Satisfied consumers: based on the aforementioned benefits for consumers, you can assume that consumer satisfaction improves (provided the bot works well). This, in turn, offers advantages for organizations that implement the chatbot.
* Feedback: The chatbot can ask for feedback at the end of each conversation. Either in the form of an open question or a multiple-choice question. This provides a platform that can process additional feedback and analyse it directly.
* In the workplace: employees can ask the chatbot questions regarding business aspects within a company. Such as how do I change my password and who do I contact for a certain problem. This ensures efficiency within companies.

# **NLP — What is it?**

NLP is a technological process that allows computers to derive meaning from user text inputs. In doing so, it attempts to understand the intent of the input, rather than just the information about the intent itself. There are a number of different ways in which this function can be built. These vary, and can be chosen based on how you intend to implement and utilize NLP.

**CHAPTER 5**

DATA AGGREGATION (WEB SCRAPPING)

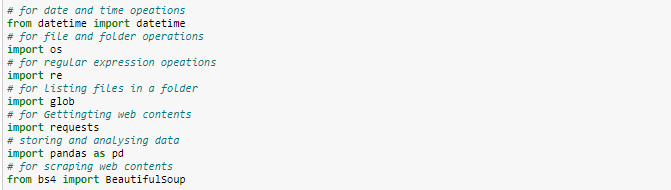
Since the pandemic is a recent issue, there is a lack of information about it on the internet and hence it is the most important part of our project to collect data from various trusted websites.

For grabbing the statistics of cases in India, we scraped the data from the Ministry of Health and Family Welfare (<https://www.mohfw.gov.in/>).

Steps involved:

1)Importing necessary libraries:

For scrapping we are going to use BeautifulSoup library



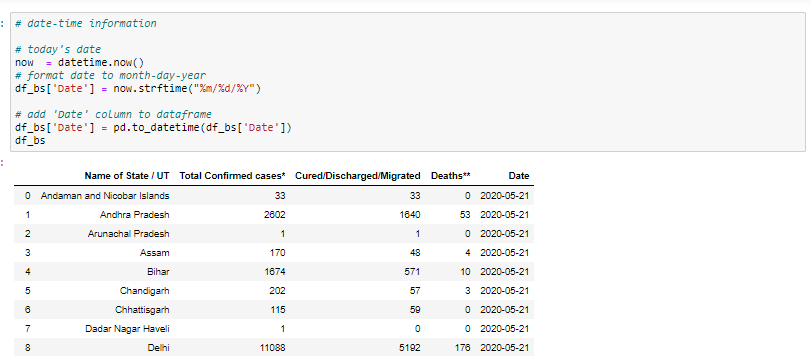
2) Using bs4 to get all tables from the website and then finding the table we need.



3) Setting all received data in a list



4)Adding the data to dataframe and adding a date column



5) Finally we save the data frames as a csv file for each date. All the csv files are then combined to form a single ‘complete’ csv file.

Note: In between we also performed some of the data cleaning methods to remove ‘#,\*’ from the data as they might create a problem in analysis.

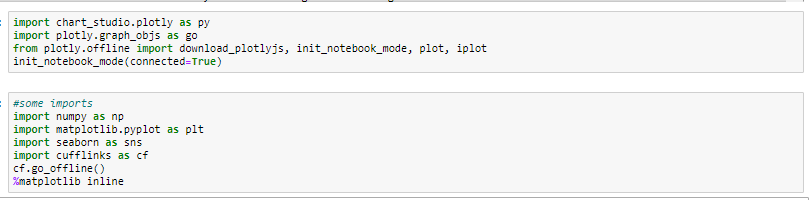
6) Similarly, we also scraped data for our chatbot too. WHO’s FAQ page was one of the trusted pages where we scraped data from. This work can be seen in web\_scraping2.ipynb file.

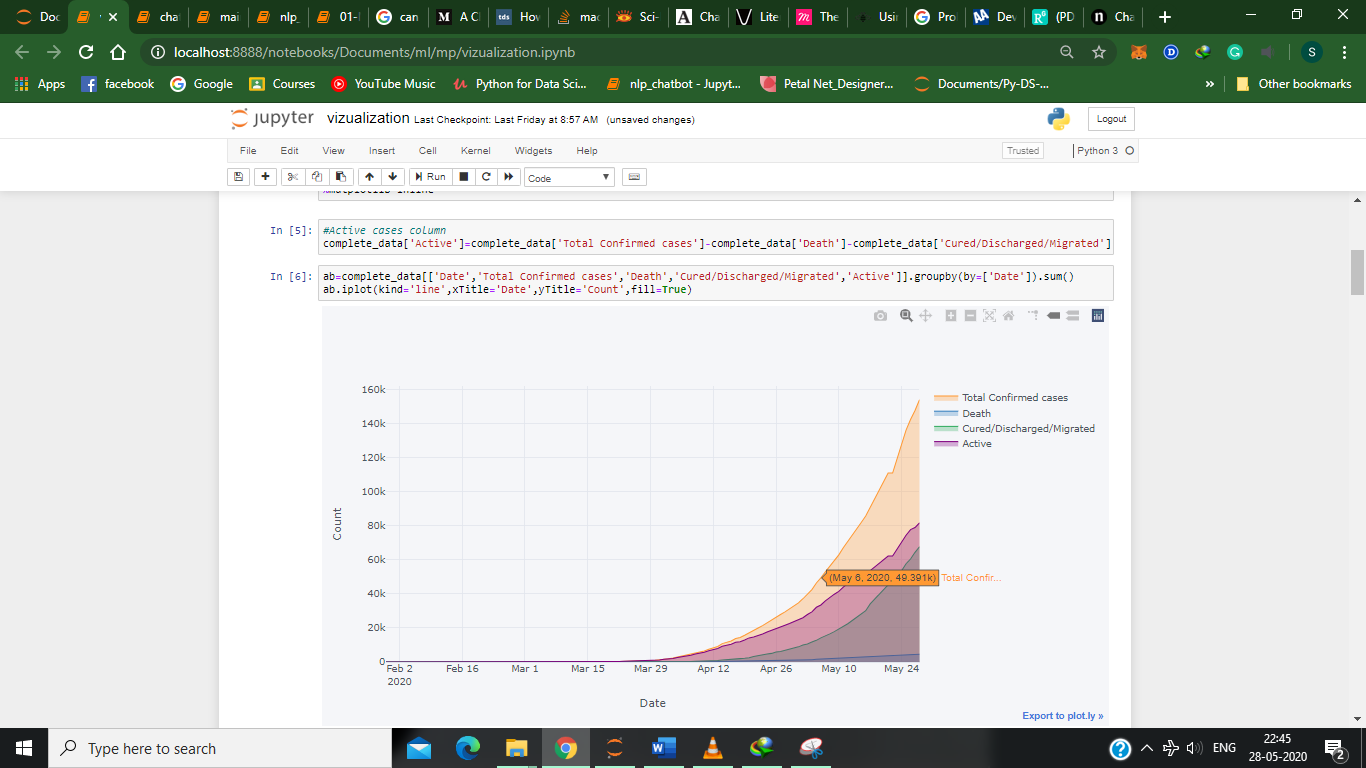
**CHAPTER 6**

**Exploratory Data Analysis of cases in India**

Here is the step by step guide of how we are analyzing data.

* 1. Importing some important visualization libraries. We are going to use cufflinks mostly for our case as it gives interactive plots.

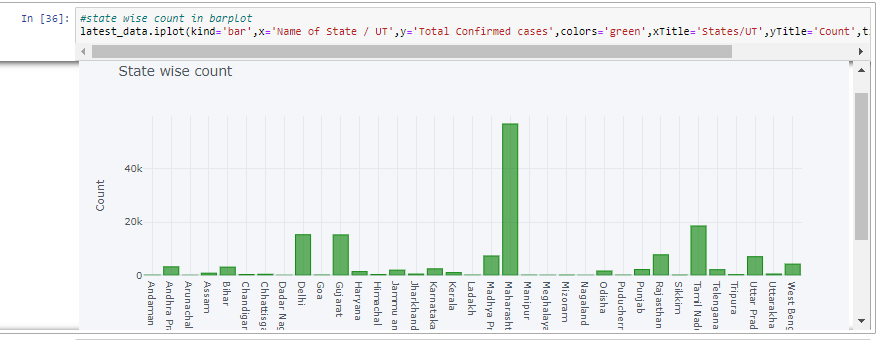


2) Representing each column of dataset in the form of interactive line graph by iplot(kind=’line’) .  


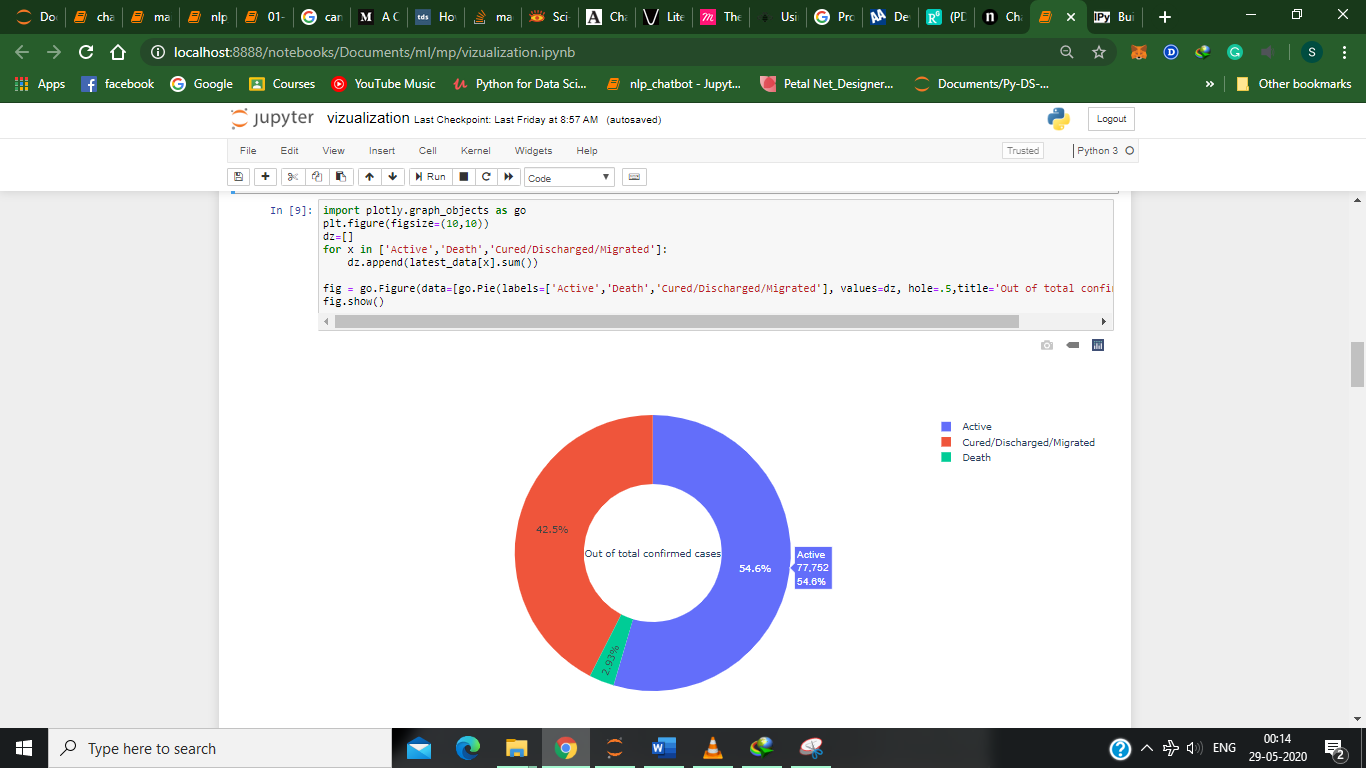
**FIG 2 : The curve which needs to be “flattened”**

Note: This gives us some of the most important insights of the rate of increase of cases with time. It can be analyzed that the curve of active cases is starting to “flatten” however the total cases curve is showing spikes. We can also see that mortality rate is very low and the recovery rate is increasing.

3) Plotting a bargraph for each state representing the count of confirmed cases in that state



4) Plotting a donut chart representing percentage of infected patients that are cured, active and dead. Hence it can be clearly seen that the mortality rate of COVID-19 is 2.93%.

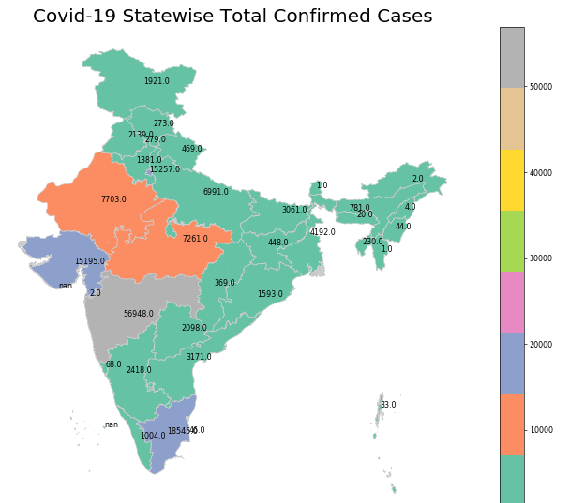


5) We have also plotted the cases per state on the map of India. For this we used geopandas. We also downloaded a shape file having the border polygon shape info of each state.



6) We add this as a geometry column to our original dataset and finally print the map.





**FIG 3:COVID CASES CONFIRMED IN INDIA**

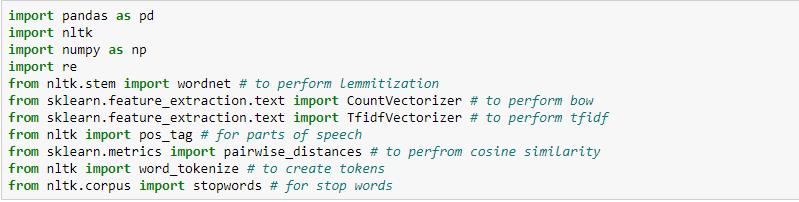
From this map, it can be vizualized that the western and southern part of India is more affected than the north and east part from the pandemic.

**CHAPTER 7**

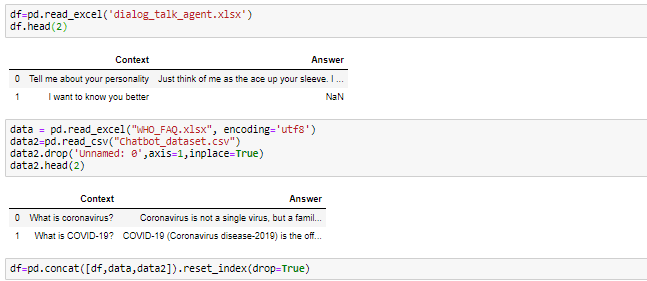
Building the Bot

We are going to build a simple chatbot using **nltk**library( **Natural Language Toolkit).**Itis a leading platform for building Python programs to work with human language data. Before using nltk we did the following steps as prerequisite.

1) Importing necessary libraries.

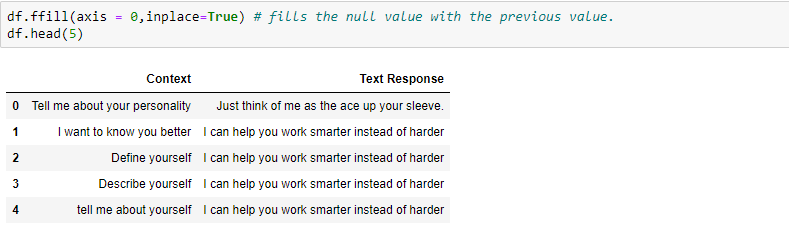


2) Import the datasets into the pandas data frames and merge them into one.

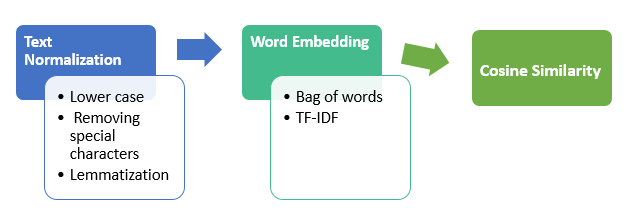


3) One of the above datasets(dialog\_talk\_agent.xlsx) contains 1592 data points and two columns context which can be inferred as the query and text response is the response for that query. we can see that there are null values in the dataset if you open the dataset in excel and observe we can find that in our dataset the data is in different clusters i.e., the same type of questions in one place and then followed by next similar kind of questions.

Null values are present for the same type of questions whose response can be almost similar and in that similar group of questions, the response is given to the first and the rest filled with null.so what we can do is use ffill()which returns the value of previous response in place of null values as below.



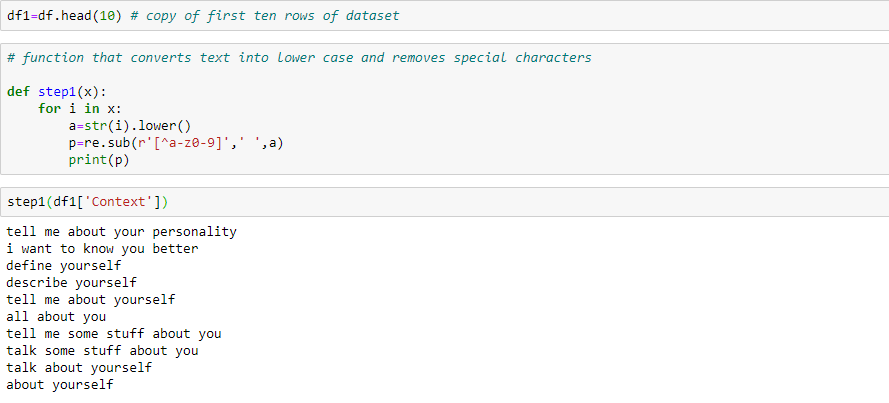
4) Now our final data is ready and we are going to do language processing on it.



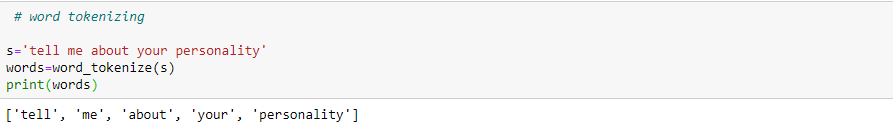
**Fig 4: STEPS INVOLVED IN LANGUAGE PROCESSING**

So let’s get into our first step i.e**,** text normalization where we convert the data into lower case and then remove special characters and then perform lemmatization.

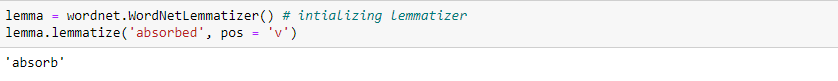
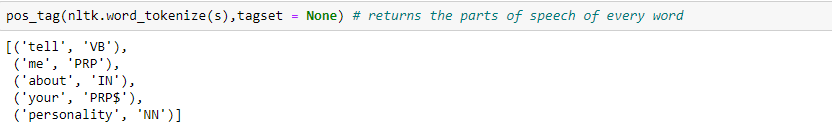
We create a function that converts given text to lower case and removes special characters and numbers.



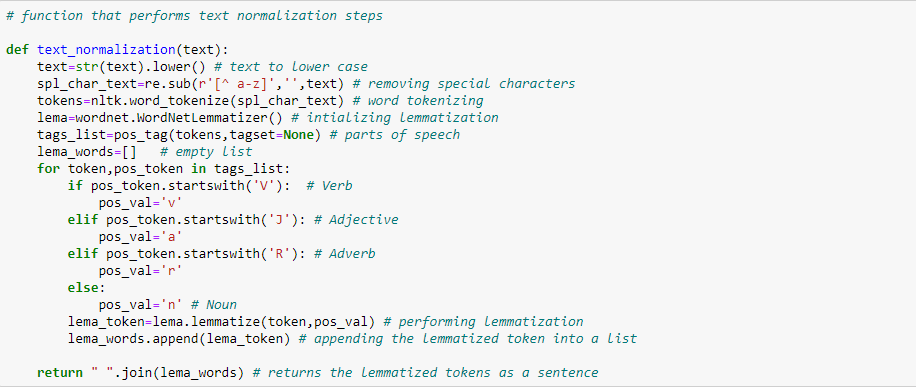
We can see that our text is clean. Word tokenizing is the process of converting the normal text strings into a list of tokens.



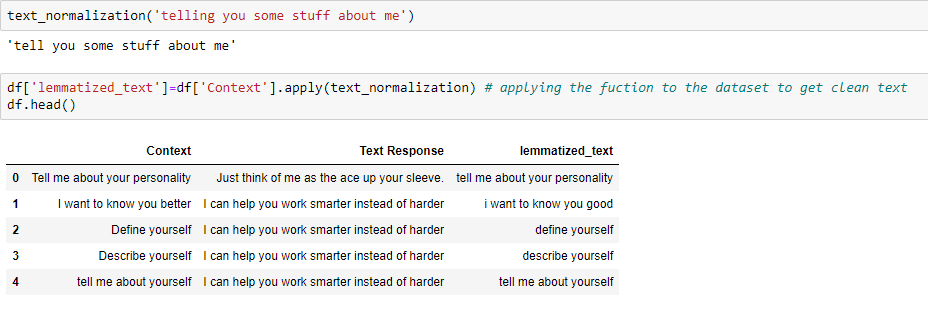
The **[pos\_tag](https://stackoverflow.com/questions/15388831/what-are-all-possible-pos-tags-of-nltk" \t "_blank)**function returns the parts of speech of each token so that the lemmatizer function detects the parts of speech of token and then it converts the token to its root word as below



We shall now create a function that performs all the steps mentioned above



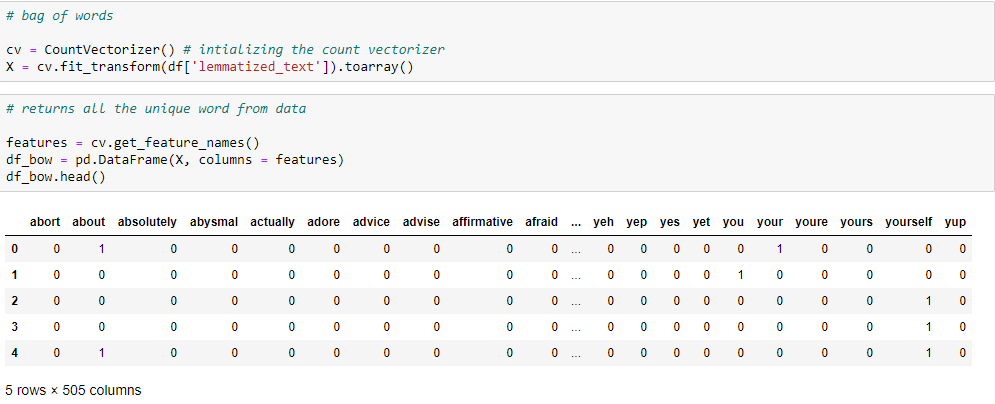
Let’s check our function and apply it to the dataset.



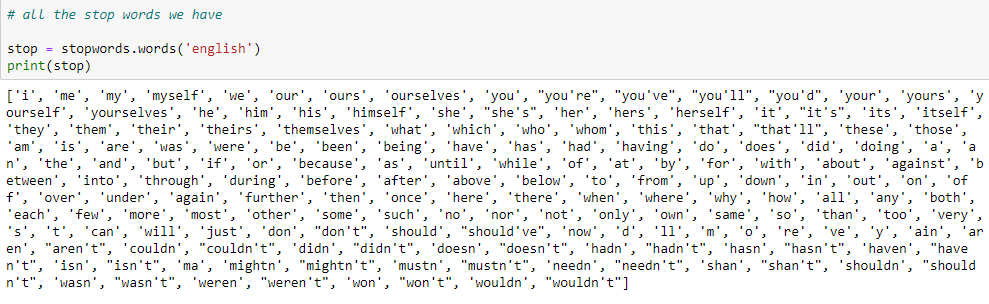
We can see that our function worked well and thus we applied the same to our data. Our next step is word embedding, it is representation for text where words that have the same meaning have a similar representation. We have two models for this process bag of words (bow) and tf-idf ( Term Frequency-Inverse Document Frequency).

## Using bag-of-words:

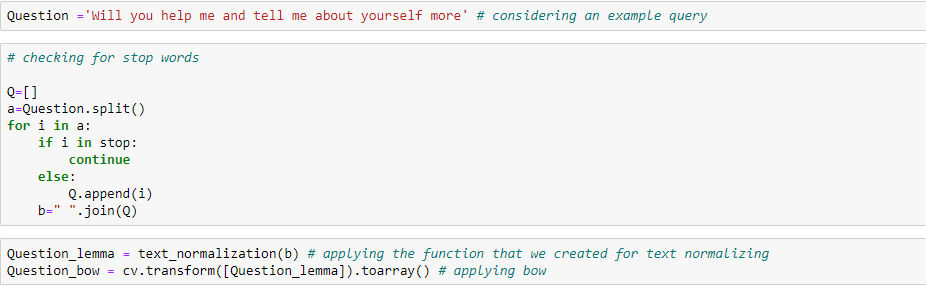
The bag-of-words is a representation of text that describes the occurrence of words within a document. Consider if our dictionary contains the words {Playing, is, love}, and we want to vectorize the text “Playing football is love”, we would have the following vector: (1, 0, 1, 1).



Stop words are extremely common words that would appear to be of little value in matching a user’s need and hence they are excluded from the vocabulary entirely. Below are the predefined stop words.



Let us consider an example and try getting a response to the query.

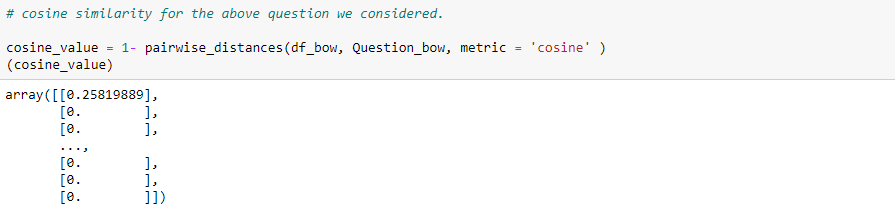


From above We can see that we have taken a question ‘Will you help me and tell me about yourself more’ and then perform text normalization and then applying the bow to the question. Now to get the related response we shall find the cosine similarity between the question and the lemmatized text we have.

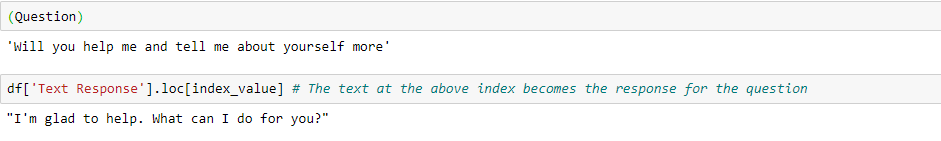
Cosine Similarity:

Cosine similarity is a measure of similarity between two vectors. It returns a value that is computed by taking the dot product and dividing that by the product of their norms between two vectors.

**Cosine Similarity (a, b) = Dot product (a, b) / ||a|| \* ||b||**



We can see that at index 194 we have the highest similarity text for the query we considered. Let us print the text at that position and see whether it is related or not.



### What is TF-IDF approach?

TF-IDF stands for term frequency-inverse document frequency, and the tf-idf weight is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the tf-idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query.

One of the simplest ranking functions is computed by summing the tf-idf for each query term; many more sophisticated ranking functions are variants of this simple model.

Typically, the tf-idf weight is composed by two terms: the first computes the normalized Term Frequency (TF), aka. the number of times a word appears in a document, divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

**TF: Term Frequency**, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization:

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

**IDF: Inverse Document Frequency**, which measures how important a term is. While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus, we need to weigh down the frequent terms while scale up the rare ones, by computing the following:

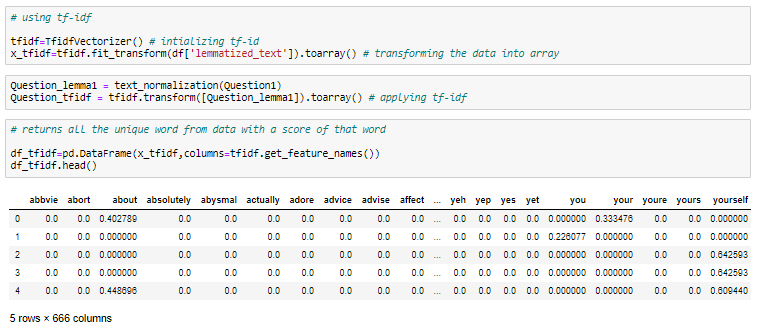
IDF(t) = log\_e(Total number of documents / Number of documents with term t in it).

See below for a simple example.

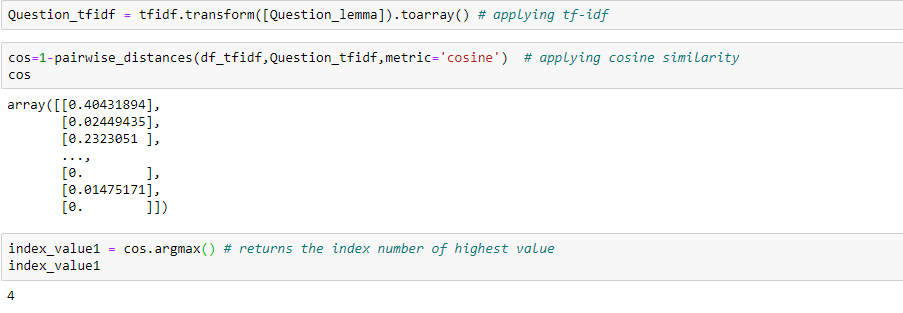
**Example:**

Consider a document containing 100 words wherein the word cat appears 3 times.

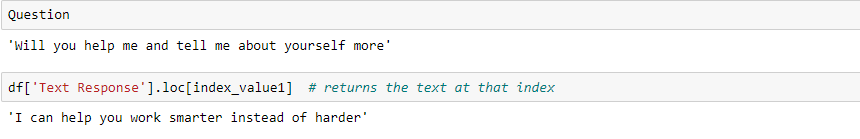
The term frequency (i.e., tf) for cat is then (3 / 100) = 0.03. Now, assume we have 10 million documents and the word cat appears in one thousand of these. Then, the inverse document frequency (i.e., idf) is calculated as log (10,000,000 / 1,000) = 4. Thus, the Tf-idf weight is the product of these quantities: 0.03 \* 4 = 0.12.



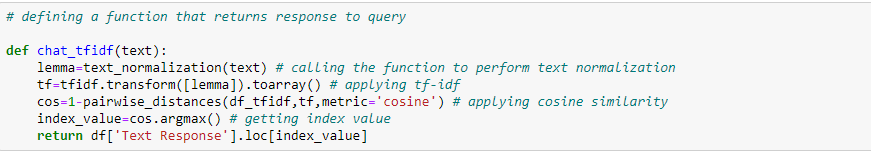
The above are the values obtained using tf-idf. Now using cosine similarity lets us find the response that we get with tf-idf.



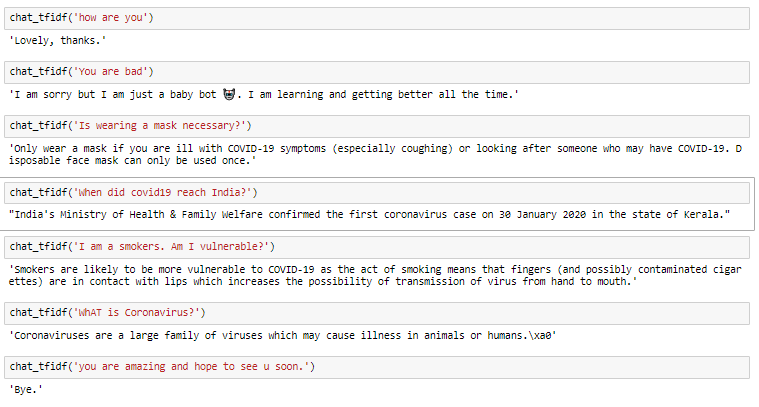
At index 4 we got higher similarity text that relates to our question. Let us see the response to our question.



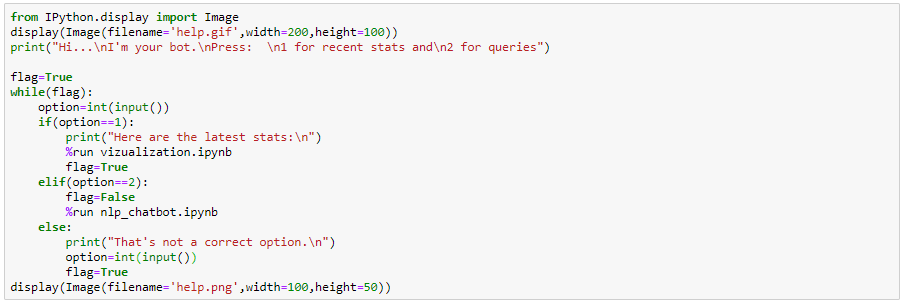
Now let’s build a function that returns the response to the query using tf-idf. It is very simple we just have to combine all the topics we saw earlier in this article.

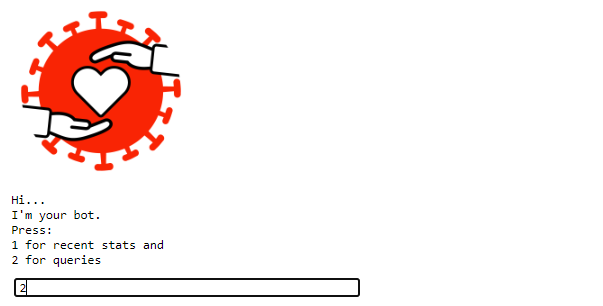


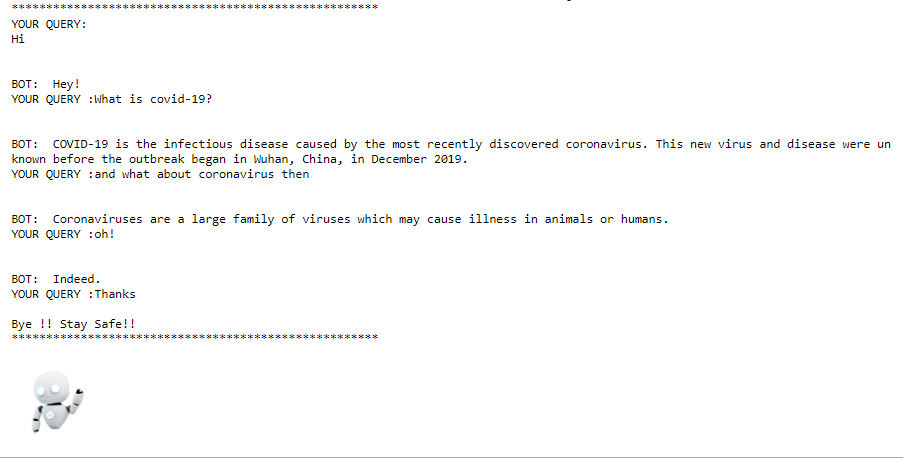
Lets see some output responses for a different queries.



At the end we created a simple main ipynb file where we aggregated both the analysis and the query resolving part into a single program and this the main file that the user is going to access. Here are the screenshots of the final working program.







**CHAPTER 8**

RESULT ANALYSIS

Strengths

1. Web scraping allowed us to acquire non-tabular or poorly structured data from recognized websites and convert it into a usable, structured format, such as a .csv file or spreadsheet. Scraping helped more than just acquiring data: it helped us archive data and track changes to data online.

2. Data analysis on real time basis keeps the user well informed about the present scenario. We have also provided all the possible visualizations which makes the user easier to understand the results. These visualizations can also be used for better planning by the government.

3. The chatbot is a retrieval bot hence user can give any question. Due to lack of data it is possible that some questions might get weird answers but it performs well on some general queries.

4. Among the two models used i.e bag-of-words and Tfidf we found that both of them performed pretty good but tfidf is more advanced form and is hence expected to give better output in a real-world scenario.

Shortcomings

1. The current chatbot created for questions regarding the coronavirus is based on a somewhat small data set. This means that this chatbot is not able to answer every question. To make this chatbot even better, more data can be collected from WHO and the dataset should be kept up to date, for there are often updates regarding the coronavirus.

2. The model also demands the use of neural network and deep learning algorithms. However due to limited knowledge on the topic it was not successfully implemented.

3. The model should either be hosted in a website or a social media platform or implemented as a stand-alone application to have a global reach.

**REFERENCES**

[1] Soufyane Ayanouz, Boudhir Anouar Abdelhakim, Mohammed Benhmed “A Smart Chatbot Architecture based NLP and Machine Learning for Health Care Assistance”, Association for Computing Machinery.

[2] David Mathew Thomas, Sandeep Mathur, “Data Analysis by Web Scraping using Python”, IEEE Conference Record # 45616

[3] AM Rahman , Abdullah Al Mamun , Alma Islam, “Programming challenges of Chatbot: Current and Future Prospective”, IEEE , International Islamic University Chittagong

[4] Rishi Sidhu, “How to Build a Chatbot — A Lesson in NLP”, <https://towardsdatascience.com/how-to-build-a-chatbot-a-lesson-in-nlp-d0df588afa4b>

[5] Pratik Nabriya, “Tracking Coronavirus(COVID-19) Spread in India using Python”, <https://towardsdatascience.com/tracking-corona-covid-19-spread-in-india-using-python-40ef8ffa7e31>