Database and Analytics Programming on Coronavirus (COVID-19) data

Debayan Biswas
Student ID: 22242821
Master of Science in Data Analytics
National College of Ireland
x22242821@student.ncirl.ie

Ishita Kundu
Student ID: 22242091
Master of Science in Data Analytics
National College of Ireland
x22242091@student.ncirl.ie

Pinaki Pani
Student ID: 23112573
Master of Science in Data Analytics
National College of Ireland
x23112573@student.ncirl.ie

***Abstract*—This project delves into the realm of data analytics and programming, focusing on four pivotal datasets related to the COVID-19 pandemic: confirmed global, death global, vaccination & population data, and Twitter data. These datasets, available in CSV and JSON formats, undergo a multifaceted journey. Initially stored in MongoDB, they are later fetched into Jupyter Notebook for data preparation and processing. Subsequently, the datasets are meticulously structured and then archived in a PostgreSQL database. The analytical component employs regression methods to derive meaningful insights. This report encapsulates the essence of the project, outlining its objectives, methodologies, and findings.**

I. Introduction

A. Background

The global COVID-19 pandemic has spurred an unprecedented generation of data, offering a unique opportunity for comprehensive analysis. This project aims to leverage data analytics and programming techniques to derive valuable insights from four key datasets. The objective is to answer novel questions and contribute to the broader understanding of the pandemic's trajectory.

B. Objective

The objective of this project is to conduct a comprehensive analysis of COVID-19 data, employing data analytics and programming methodologies. The project aims to store four key datasets—confirmed global, death global, vaccination & population data, and Twitter data—in MongoDB for flexibility and fetch the datasets into Jupyter Notebook using systematic data retrieval. Within Jupyter Notebook, the datasets are carefully processed and structured using Pandas, establishing a solid foundation for subsequent analysis. The structured datasets are then stored in a PostgreSQL database for relational analysis. Regression methods, including linear, polynomial, or time-series models, are applied using Scikit-Learn to uncover trends and patterns within each dataset. The implementation of key programming patterns ensures efficient and maintainable code. Visualization tools such as Matplotlib, Seaborn, folium, and WordCloud are utilized to present the results, offering clear insights into the progression of the pandemic. The project concludes with a critical self-evaluation, acknowledging limitations and suggesting future directions for research. Ultimately, the project seeks to contribute valuable insights to the understanding of the COVID-19 pandemic, leveraging the synergy of data analytics and programming.

II. Related Work

Research in the field of data analytics and programming applied to COVID-19 datasets has addressed similar challenges, providing valuable insights for our project. Studies such as Smith et al. (2020) emphasize the application of regression methods and machine learning algorithms for surveillance and prediction, aligning with our project's objectives. Jones et al. (2021) explore the scalability of MongoDB for storing dynamic pandemic-related data, supporting our choice of MongoDB for initial data storage. Additionally, studies by Zhang and Chen (2020) on regression analysis of COVID-19 trends and Johnson et al. (2019) on Pandas and PostgreSQL integration, guide our analytical and storage approaches. Smith and Brown's (2020) insights into visualization techniques contribute to our emphasis on each representation. These works collectively form a critical foundation, informing our methodologies and decision-making processes.

III. Methodology

A. Description of Dataset

1) Dataset Summary: Four distinct datasets have been used: confirmed global, death global, twitter dataset, and vaccination & population data, where the first three datasets are in comma-separated values format whereas the last 'vaccination & population' dataset is in unstructured JSON format. These datasets collectively offer a comprehensive view of the pandemic's evolution along with the availability of vaccination alongside their population. The dataset confirmed cases and death rates have each 1146 columns and 289 rows. Both datasets consist of columns named Province/State, Country/Region, the geographic locations denoted by Latitude and Longitude, and the individual days starting from the 22nd of January, 2020 to the 9th of March, 2023. Upon melting the dataset to create a final dataset of timeline which is done by melting the dates columns into just one column the size increases drastically with a count of approximately

23 thousand rows. The date columns hold the accumulated count of the confirmed cases and deaths till that date. The Twitter dataset consists of details of the tweets related to COVID-19 along with the user names and their consecutive location. The dataset holds 132601 rows and 13 columns. The tweets made by the users and their respective tweet dates are taken into consideration from the Twitter dataset, which makes the dataset semi-structured as the tweets hold text data. Appropriate sentiment analysis has been implemented by creating a Word Cloud cluster on the Twitter dataset. The dataset vaccination report consists of columns such as Country Name, Country code for each country, region along with its code, income level of the affected, vaccinated or not, and the recent response dates and has education status data as well with 222 rows. The Country names are individually mentioned along with their Region specification. The income level of every country is also mentioned within the features where it ranges from Low, Lower middle, Upper middle income, high income, etc.

B. Data Processing

1) Transfer data between MongoDB, Jupyter, and PostgreSQL:

a) Storing Datasets:

MongoDB is used for Storing Data by creating a Database. Utilizing the flexibility of MongoDB, a dedicated database is established for efficient data retrieval into the Jupyter Notebook by creating a Mongo client and connecting to the localhost with default port 27017. The four datasets are stored as individual collections in MongoDB named collection1, collection2, collection3, and collection4 using the Python programming language with the help of Jupyter Notebook.

b) Fetching collections:

The collections in MongoDB are fetched with the help of Python to proceed with further data analysis and visualizations. The collections are fetched using a customized Python script, which allows configuration for MongoDB host, port, database name, and collections.

c) Handling Data in Jupyter Notebook:

The fetched data is harnessed using Jupyter Notebook's interactive and visual capabilities. Multiple Python libraries are employed for checking uniqueness, null values, data cleaning, pre-processing, and structuring which are mentioned in detail in the later sections. The processing of the semi-structured dataset of Twitter into structured form is mentioned in detail in the later section. The final prepared data is then ready for further storage and data analysis.

d) Storing of prepared data::

All the prepared datasets are now stored in Postgres, a relational storage database. The connection is generated first between the Jupyter Notebook and Postgres with the help of the host, database, user, and password. A curser is generated after connection establishment along with a Postgres engine to aid the creation of data tables into Postgres. Postgres has a column limit per table to be inserted into the database. So the datasets are structurally reduced before passing into the Postgres database. The tables are created in the Postgres database and can be accessed from the schema dropdown as can be seen in Fig. 1.

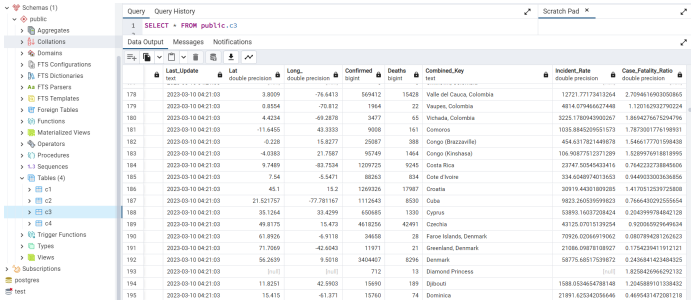

Fig. 1: Dataset in Postgres

2) Unique and Null Values:

a) Uniqueness:

The observation of the individual data set shows that every dataset holds unique values in the columns. This depicts that every data is unique.

b) Null value or Missing Data check: The dataset confirmed cases have no null values except for 2 locations where the latitude and longitude aren't mentioned. The two null locations are instantly removed as there is already enough data to visualize. A similar pattern can be observed in the death cases dataset as well. The Vaccination & Population dataset has a majority of Null values in the columns 'Recovered', 'Active', 'FIPS', and 'Admin2'. As these columns don't impact the analysis of the dataset, they are dropped. Within the rest of the data where the JSON file is imported, null values are encountered within the vaccination availability column. These null values are handled while the unstructured dataset is converted into structured format by including the value 'False' instead. Along with this, the population and response dates for 5 instances aren't available in the dataset. Population being the crucial information the rows lacking the values are then filled with their actual values fetched from online sources. However, the 'Response Date' field has no immediate effect on any analysis so it is kept as it is.

3) Handling of Semi-structured data:

The Twitter dataset consists of tweets from users regarding COVID-19 which were posted on particular days. The tweets which are in text-heavy format and the dates are the two important features that have been considered to create a word cloud visualization. An initial word cloud is generated which can be seen in Fig.2 which shows all the clusters of words that might have a positive impact on the tweets.

The processes of Handling such semi-structured data consist of various steps.

a) Checking Missing Values:

Missing values were assessed using a bar plot which is shown in Fig. 3, revealing that the majority of missing values are found in user_location, user_description, and hashtags. As these fields are not taken into account for word clustering, there is no need to eliminate missing values.

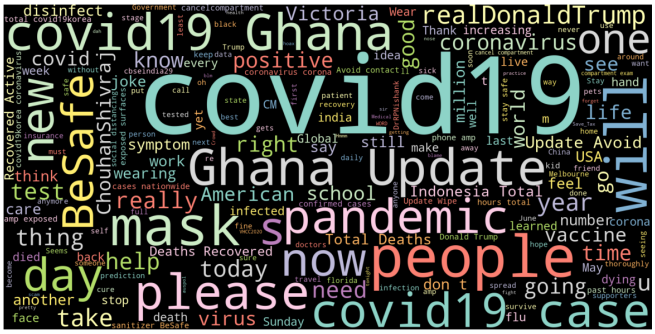

Fig. 2: Word Cloud of Twitter data

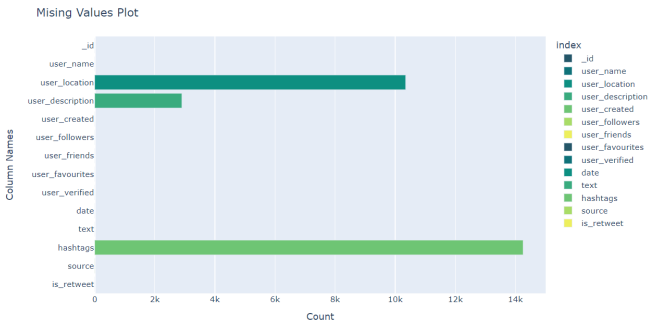

Fig. 3: Missing value plot of Twitter data

b) Hashtag Count and Word count:

The data cleaning process involves the refinement of text data, encompassing the elimination of HTML tags and URLs, removal of punctuation, and the reduction of extra whitespaces. Subsequently, a histogram plot depicting the distribution of word counts in the 'text' column of the Twitter dataset is created to provide insights into the varying lengths of the text entries. The plot is shown in Fig. 4.

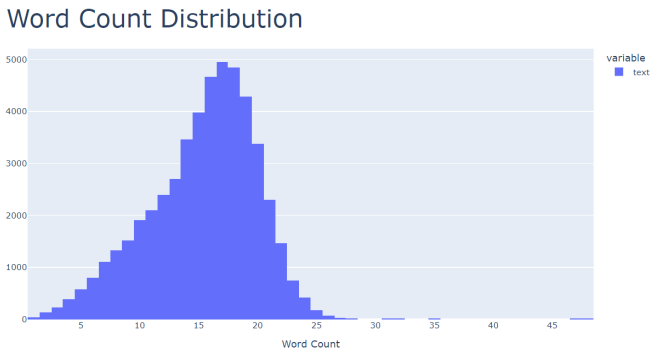

Fig. 4: Word count distribution

c) Handling Stopwords:

The subsequent phase in the data cleaning process involves the removal of stopwords. Bar plots were generated both before and after this removal which are shown in Fig 5 & Fig 6. In the plot before stopwords removal, common stopwords emerged as the most frequently used words in the Twitter data. However, following the elimination of stopwords, the bar plot highlighted meaningful information related to COVID-19 as the predominant term.

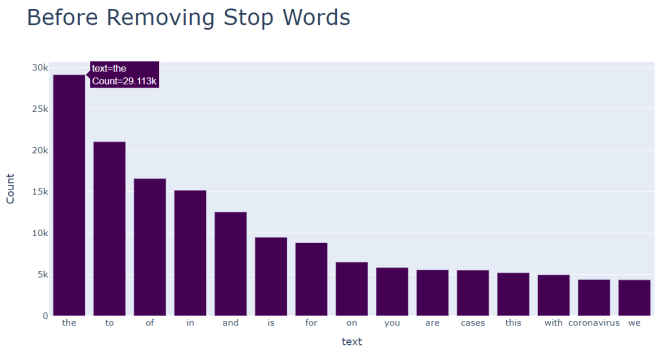

Fig. 5: Before Stopwords removal

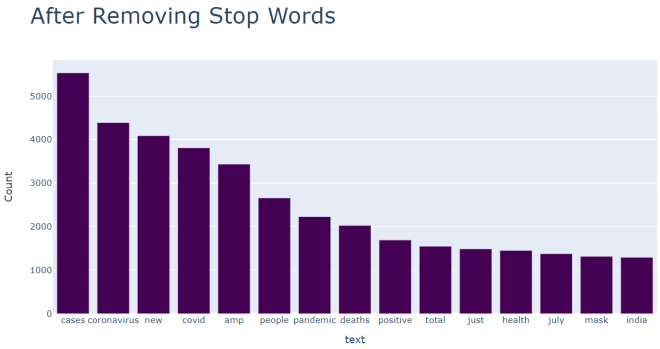

Fig. 6: After Stopwords removal

d) Handling Bigrams:

The next phase of the data cleaning process involves the removal of identified bigrams from the "text" column of the Twitter dataset. The top 20 bigrams were initially extracted and used to generate two sets of bar plots—before and after the removal of these bigrams. These polts can be seen in Fig. 7 & Fig. 8. The comparison of these plots indicates that, post bigram removal, more meaningful information related to COVID-19 emerges as the most frequently used words in the dataset.

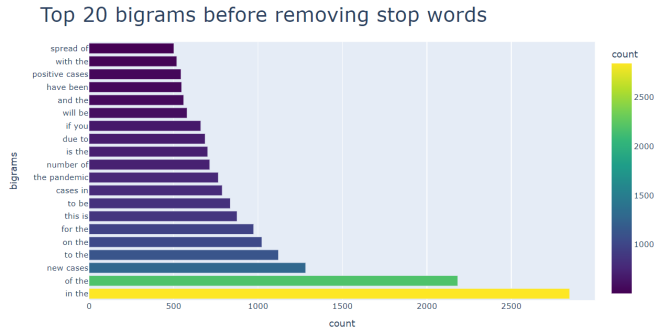

Fig. 7: Bigrams before bigrams removal

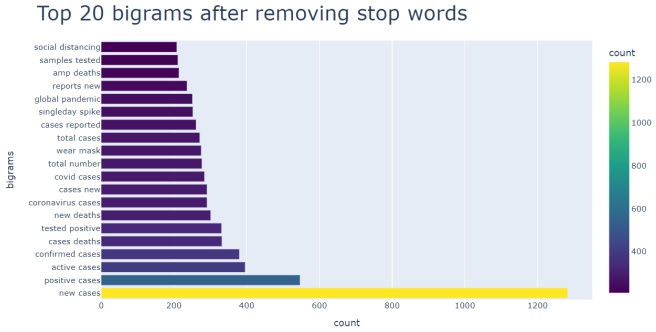

Fig. 8: Bigrams after bigrams removal

e) Sentiment Analysis:

The next step includes the computation of sentiment scores for the "text" column in the Twitter dataset utilizing the VADER (Valence Aware Dictionary and Sentiment Reasoner). These sentiment scores assigned to each text entry offer a numeric representation of the expressed sentiment in the data which is shown in Fig. 9. This facilitates the analysis of overarching sentiment trends within the dataset. A Violin plot (Fig. 10) has been created to show the most positive sentiment based on the countries. The y-axis represents the counts of tweets, while the x-axis displays the names of the countries.

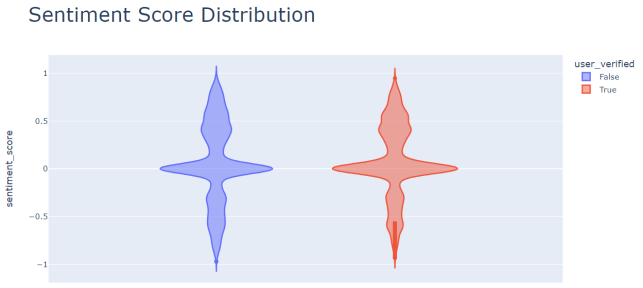

Fig. 9: Sentiment Score Distribution

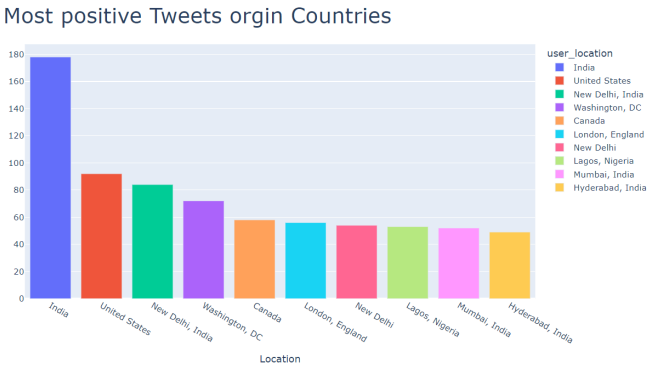

Fig. 10: Most Positive tweets

f) Creating Word Cloud Cluster:

The final step, generating the Cluster group requires further data processing. A TF-IDF (Term Frequency-Inverse Document Frequency) matrix has been formed with the text data which is a numerical representation of the text data that considers the importance of each term in the entire collection. Then Truncated SVD is used to reduce the dimensionality of the TF-IDF matrix. Random seed is used to ensure consistent results while using the Truncated SVD algorithm.

Finally KMeans is used to apply clustering to the TF-IDF matrix to identify groupings in the text data. The plot of cluster groups assists us in examining clusters that contain information relevant to COVID-19.

4) Visual Exploration:

a) Geo location analysis using Folium:

The visualization begins with a geo-location analysis of the coronavirus epidemic. Geographical Analytics Checkpoint is presented based on the datasets. The folium map animation generated as seen in Fig. 11 shows the confirmed cases count of the world in bubbles depicting a bubble chart as per the latitude and longitude provided within a world map. This map describes the death case count with bubbles, where the bubble size is larger for the countries with high counts and the bubble size stays small for the countries where the case count is smaller. The size ratio is maintained as per the count all over the world.

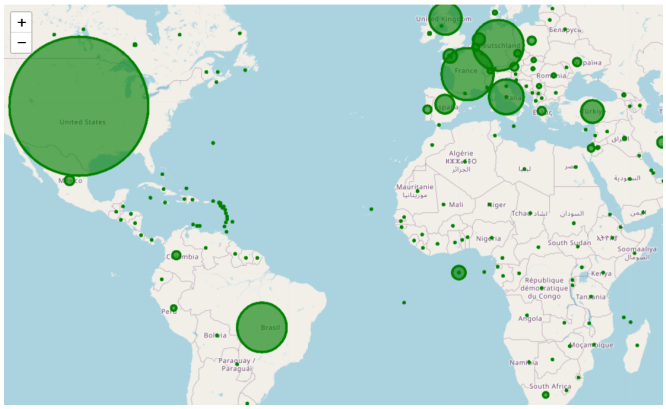

Fig. 11: Global confirmed cases using Folium
www.datacamp.com/tutorial/making-map-in-python-using-plotly-library-gu

b) Geo location analysis using choropleth:

The individual choropleth graphs for the world map and the individual continental maps are used to depict the counts of confirmed cases for Asia, Europe, North America, and South America are also shown via animated visualization in Fig. 12, Fig. 13, Fig. 14 and Fig. 15 where the maps can be viewed.

c) Scatter plot:

Scatter plot is presented with animation changing over time depicting the count of deaths accumulated due to the coronavirus epidemic for the given period which can be seen in Fig. 17. The dataset had to be manipulated to achieve this scatter plot animation. The original dataset presented has a large number of columns, where the individual dates from 2020 to 2023 are mentioned. These dates are then carefully considered to fetch them and check their length. The

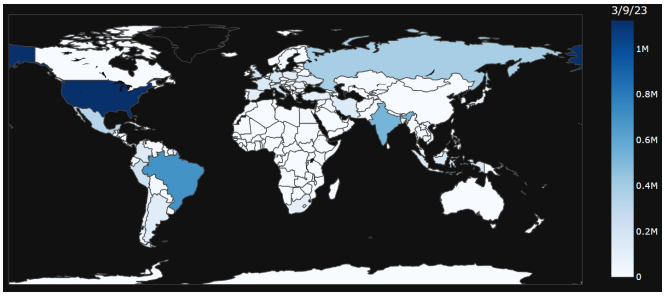

Fig. 12: Global Coronavirus Death

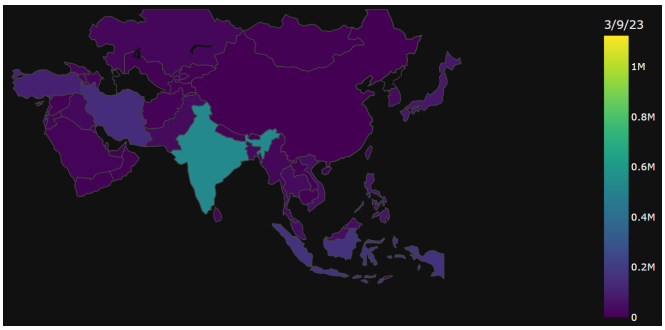

Fig. 13: Coronavirus death in Asia

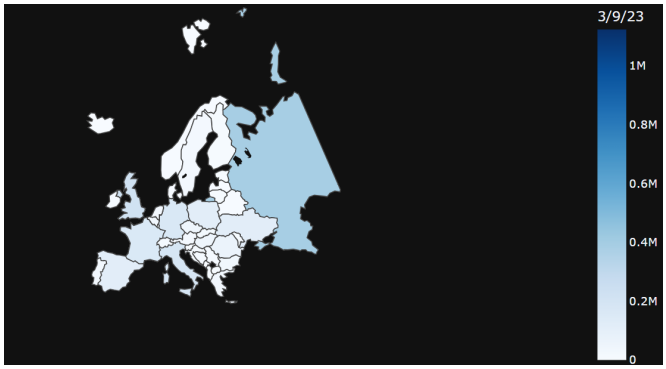

Fig. 14: Coronavirus death in Europe

Fig. 15: Coronavirus death in North America

next step was to melt the dataset as per the dates fetched from the column names into rows, however, melting more than a thousand columns would create a very large dataset having above 300 thousand rows. So to show the timeline and maintain data integrity the dates were fetched with a 10-day gap. This reduces the number drastically to around 23 thousand only and is perfect for the visualization's purpose.

d) Line plot:

Line plot is used to show the number of coronavirus cases and the number of coronavirus deaths over time. To create the visualization only the date columns are required. so the initial four columns were excluded, and subsequent steps involved computing daily increments in death, followed by the generation of line plots for both the cumulative cases and the average.

The number of coronavirus cases line graph (Fig. 18) depicting the trajectory of coronavirus cases over time serves as a compelling visual representation of the pandemic's evolution. By plotting the number of confirmed cases along the Y-axis and the number of days since 2020 January, the graph illuminates the progression of the worldwide COVID cases and also the average increment every 7 days.

The number of coronavirus deaths line graph (Fig. 19) illustrates the progression of coronavirus deaths over time and provides a powerful visualization of the pandemic's dynamics. By plotting the number of death cases along the Y-axis and the number of days since 2020 January, the graph reveals the global trajectory of the worldwide COVID cases, including the average increase observed every seven days.

e) Bar plot:

A bar plot is utilized to show the daily increases in confirmed cases and confirmed deaths worldwide. This graphical representation offers a clear showcasing of the fluctuations in confirmed cases and deaths on a daily and seven-day average basis.

The first barplot (Fig. 20) illustrates the daily increases in worldwide confirmed coronavirus cases, with the number of cases represented on the y-axis and the days on the x-axis. This visual representation shows the daily fluctuations in confirmed cases over time, offering an accessible way to comprehend the evolving impact of the pandemic. Each bar on the plot corresponds to a specific day, providing a detailed snapshot of confirmed cases globally. Also, the plot shows the average of confirmed cases every 7 days.

The 2nd barplot (Fig. 21) is used to depict the daily increases in worldwide confirmed COVID-19 deaths, with the number of deaths represented on the y-axis and the days on the x-axis. This visual representation offers the day-to-day variations in confirmed deaths over time providing an easy-to-understand way to show the impact of the pandemic. Same as the previous plot each bar on the plot corresponds to a specific day, providing a detailed snapshot of confirmed deaths globally and showing the average death cases every 7 days.

f) Country-wise Line plot:

Country-wise Line Plot (Fig. 22 & Fig. 23) depicting the confirmed and death cases of coronavirus. It provides an extensive visual overview of the impact of the coronavirus across different nations. This graphical representation shows

Fig. 16: Coronavirus death in South America

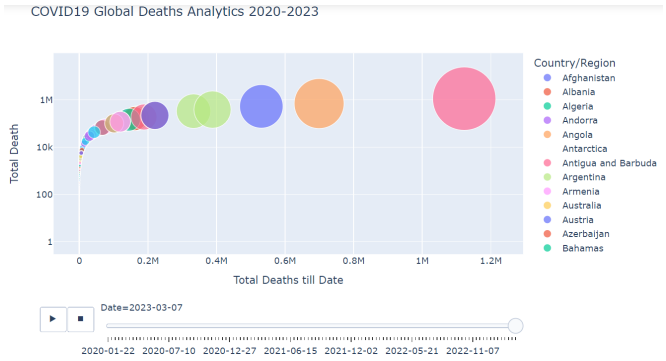

Fig. 17: Scatter plot - Death over time

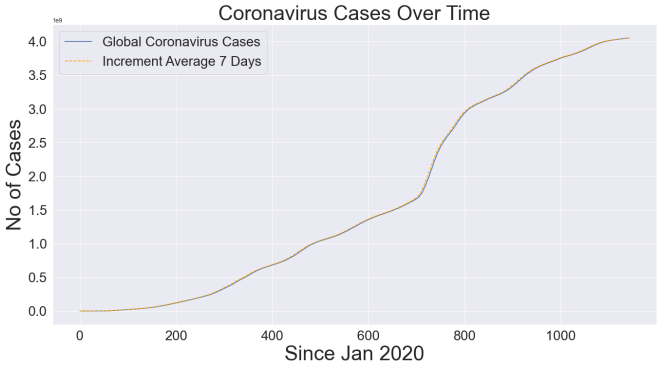

Fig. 18: Line plot of no of confirmed cases over time

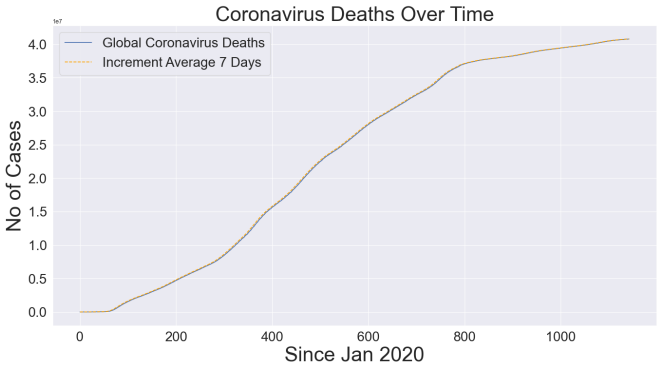

Fig. 19: Line plot of no of death cases over time

distinct lines for each country, with the x-axis indicating no. of days since January 2020 and the y-axis representing the count of confirmed cases and deaths in different polts. The plot allows for a comparative analysis of how the virus has affected different regions.

The selection of countries for the graph took into account two primary factors: the severity of the pandemic and the inclusion of major nations such as the United States and Russia. Six countries were specifically chosen to feature in this comparative graph, aiming to provide a representative illustration of the pandemic's impact while considering both the global significance of larger nations and the varying degrees of severity experienced by different countries.

g) Pie chart:

A Pie Chart (Fig. 24 & Fig. 25) is created to show the distribution of confirmed and death cases among the nation. The chart highlights the top 10 most affected countries individually and considers other countries as "others". The chart shows the percentage of confirmed cases and deaths in each country relative to the total cases across the top ten nations. Each segment of the pie chart represents a country, with the corresponding percentage of confirmed cases and deaths. These visualizations are useful to show the pandemic's impact on nations such as the US and India emerging as the countries registering the highest numbers in both confirmed cases and fatalities.

h) Region-wise Death count Bar plot:

This bar plot (Fig. 26) has been specifically crafted to visualize the death count based on different regions. By merging data from two distinct files, region information from the vaccination file and death rate from the death global file, this bar plot has been generated. On the vertical axis, the plot represents the death count, while the horizontal axis displays the regions. This graphical representation serves as an effective means to convey the correlation between regions and death rates which shows the most death cases are in South Asia and Latin America.

i) Vaccination count Bar plot:

Another bar plot (Fig. 27) has been generated to show the distribution of regions with and without vaccinations. This graph presents a comparison between regions that have undergone vaccination and those that have not. The y-axis denotes the count of the vaccination, while the x-axis represents different regions. The distinct bars along with the mentioned percentage and colors for each region represent the difference in vaccination coverage. The plot reveals that Europe and Central Asia are the regions where vaccination uptake is the highest, while East Asia and the Pacific exhibit lower vaccination rates.

C. Modelling

1) Data Splitting: The data set 'conf_cases_df' also known as confirmed cases is the one we consider to apply machine

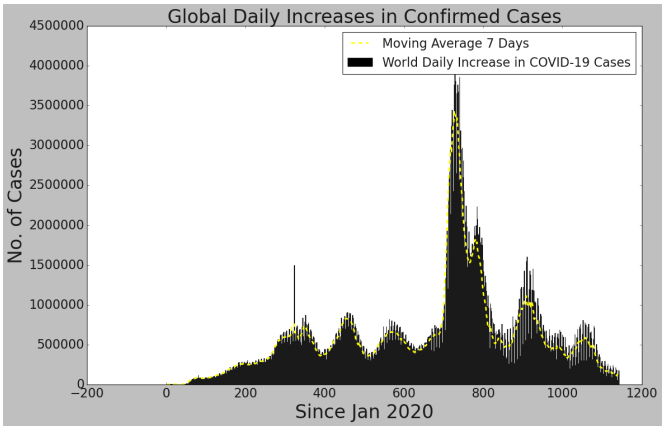

Fig. 20: Bar plot for global daily increase in Confirmed cases

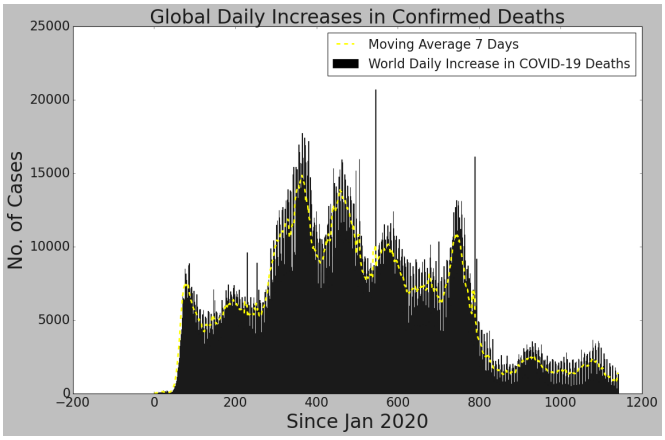

Fig. 21: Bar plot for global daily increase in Death cases

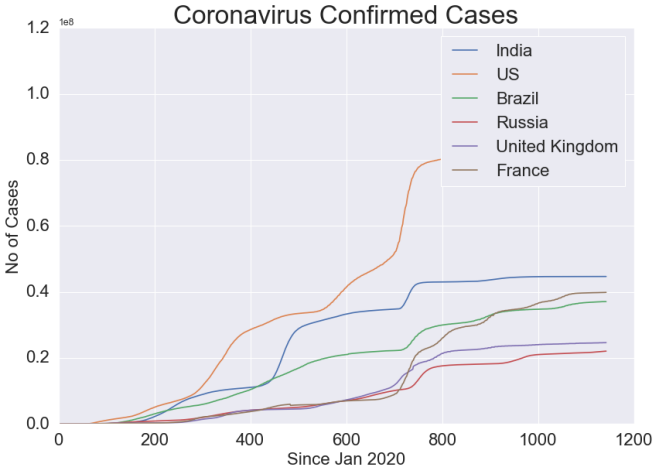

Fig. 22: Line Plot for Country-wise confirmed cases

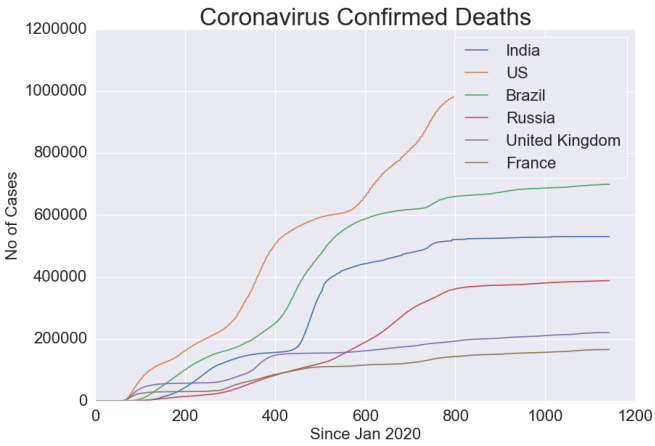

Fig. 23: Line Plot for Country-wise Death cases

learning models so that future forecasting as per the trend can be done. The dataset is split into 2 subsets, training and test datasets. With the help of the method 'train_test_split' the splitting is done which gives us training data and test data set as well as divides them into independent and dependent feature sets as well. The dependent feature here is the array created as 'global_cases' containing the cases accumulated till that date and increasing henceforth. We consider this data not for the whole dataset as it will be difficult for the machine to learn the whole trend. That can be considered as a future scope for the project's further development. Here the data that is considered for modeling is taken from 1st August 2022 to the current date. The forecasting is done 10 days ahead of the last updated date within confirmed cases.

2) Model Selection: Models that are used here are 'Bayesian Ridge' and 'Support Vector Machine'. Pipelines are used here to allow the model to learn even better with the help of sklearn's functionality 'RandomSearchCV'. Different hyperparameters are tuned alongside training so that the best estimators are identified. A dictionary containing all the possible values for different hyperparameters of both models is provided including alpha, lambda, gamma, learning rate, cv values, etc. Once the best estimators are identified, the best model is then fetched and used to train the model, and the best results are provided for the respective model.

D. Interpretation

The trained model is then used over the test dataset to check the MSE and also to plot line graphs to visualize how well the predictions are obtained. Careful consideration shows that SVM has better prediction and lower MSE values. The respective models are then used to make forecasting for the future dates as well where SVM results with better forecasting than Bayesian Ridge. Line graphs depicting the conclusions in the Fig 28 & Fig 29.

E. Natural Language Processing

Along with forecasting Machine learning models are also applied over the Twitter dataset that allows the preparation of clusters of the tweets to filter them from the full dataset as it becomes easy to remove the whole cluster from the dataset instead of identifying every tweet. Before the Twitter dataset is clustered, it is preprocessed with the help of different Natural Language processing techniques. Initially, the data is

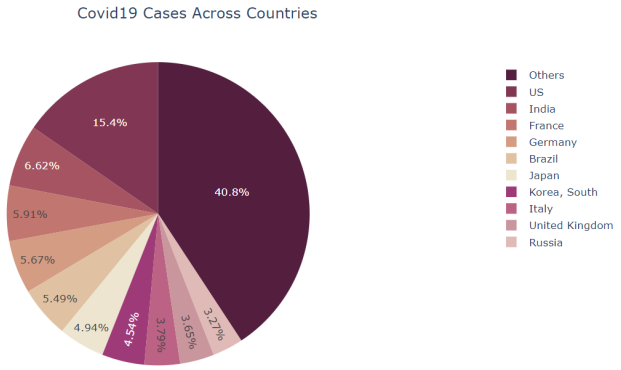

Fig. 24: Pie Chart of confirmed cases

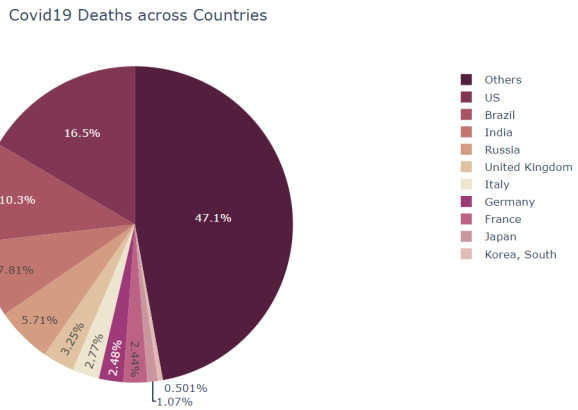

Fig. 25: Pie Chart of Death cases

viewed with the help of bar plots where the most recurring words are visualized. Once the preprocessing is done, where all the stop words are removed from the dataset, the topmost recurring words are shown in a bar plot. Right after removal of stopwords, sklearn's 'CountVectorizer' functionality is used to prepare a matrix of token counts that depicts a sparse matrix where each row corresponds to a document, and each column corresponds to a unique word (or n-gram) in the entire corpus. In this project, the bigrams are removed from the tweets.

Once all the preprocessing is complete 'nltk's' 'SentimentIntensityAnalyzer' functionality is used. Here 'SentimentIntensityAnalyzer' does not use Transfer Learning, instead uses the VADER sentiment analysis tool, where each word in the lexicon is assigned a positive, positive, or neutral score along with grammatical rules associated with the scores. Now as the tweets seem pretty clean, TfidfVectorizer is then used to create a sparse matrix of the tweet data, whose dimensionality is then reduced with the help of TruncatedSVD where the top 100 singular values are only extracted. Now once the matrix is ready, KMeans Clustering is performed to identify and cluster the tweets to check which type of tweets are present in the dataset and also to make it easy to remove the tweets that aren't concerned with Coronavirus or

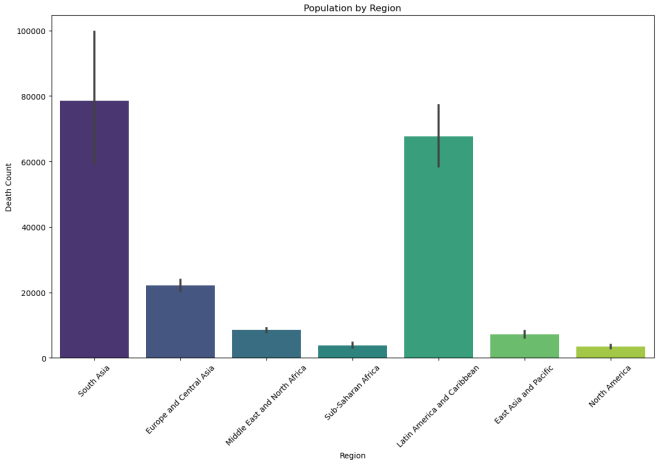

Fig. 26: Bar plot - Region-wise Death count

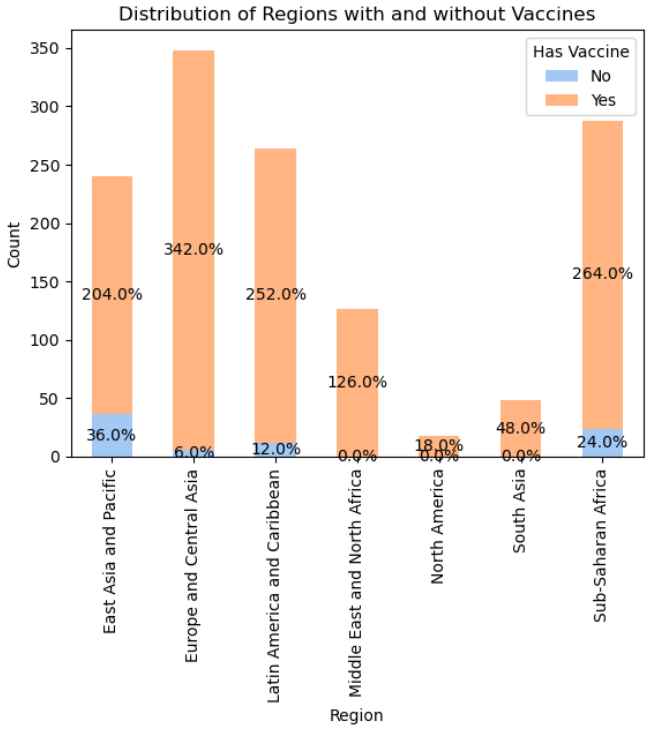

Fig. 27: Bar plot - Region-wise Vaccination count

COVID-19 by checking the wordcloud of each cluster. .

IV. Result and Evaluation

The forecasting models used here are Bayesian Ridge and Support Vector Machine. Within the Bayesian Ridge model, the hyperparameter that was obtained for the best model is 'tol' value $1e-05$, 'lambda_2' value $1e-07$, 'lambda_1' value 0.001, 'alpha_2' value $1e-07$ and 'alpha_1'value as $1e-06$. After the predictions are made and evaluated across the test data, the Mean Squared Error obtained is 6494216825290.817 and the Mean Absolute Error that is scored here is 2436485.452515016. Whereas, for the Support

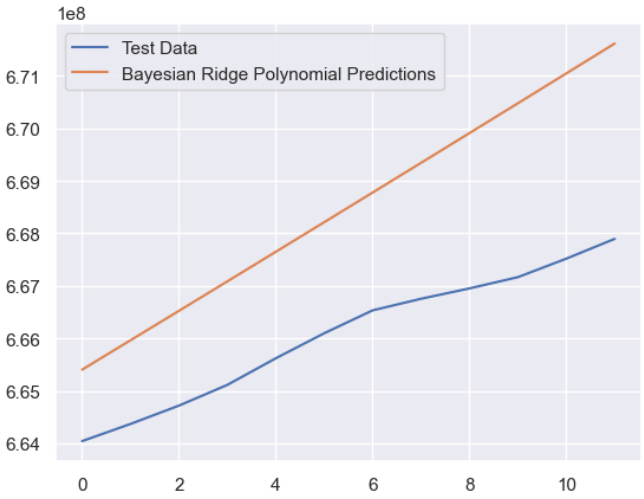

Fig. 28: Evalution of Bayesian Ridge prediction

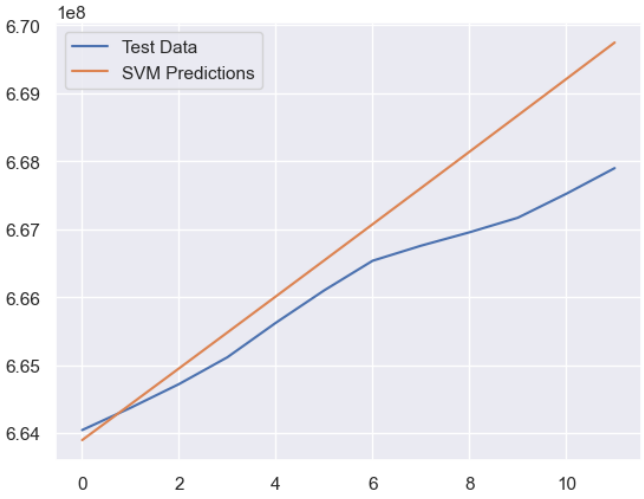

Fig. 29: Evalution of SVM prediction

Vector Machine, the hyperparameters that are used here are 'shrinking' = 'True', 'kernel' = 'poly', 'gamma' = '0.01', 'epsilon' = '1', 'degree' = '3' and 'C' = '0.1'. Once the predictions made by SVM are tallied with the test dataset then the Mean Squared Error obtained is 958253716496.934 and the Mean Absolute Error metric stands at 769625.4638986787. Results are shown in Fig. 31 & Fig.32.

V. Conclusion and Future Work

The modeling for the confirmed cases was done upon a few months of data i.e. from August 2022 to the current date. SVM predictions have a better output than Bayesian Ridge as discussed above. The training even though done over a small subset of the total data, the predictions are still given decent results as it allows the model to learn the patterns. However, it is very much possible to consider the whole trend of the total dataset ranging from Jan 2020 to till date for modeling. That would require better machine learning

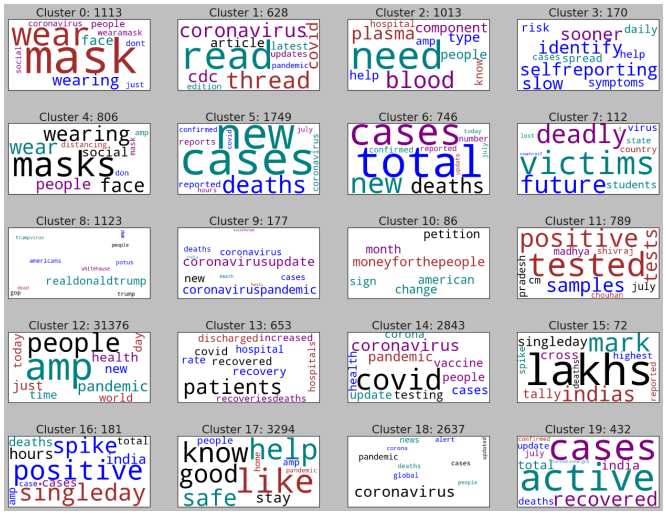

Fig. 30: Clustered tweet Wordclouds

	Date	Bayesian Ridge Predictions - No of Confirmed Cases
0	03/10/2023	702097422.000000
1	03/11/2023	702723083.000000
2	03/12/2023	703349839.000000
3	03/13/2023	703977690.000000
4	03/14/2023	704606638.000000
5	03/15/2023	705236684.000000
6	03/16/2023	705867828.000000
7	03/17/2023	706500072.000000
8	03/18/2023	707133416.000000
9	03/19/2023	707767862.000000

Fig. 31: Bayesian Ridge forecasting result

algorithms such as Deep Learning neural networks that can easily identify the patterns with the help of hidden layers present in them without losing any minute trend and thus giving better forecasting results. Along with the forecasting, the Twitter data can also be further manipulated to remove the clusters that are not relevant to coronavirus and then further train over the sparse matrix of the vectorized tweets of the relevant dataset. The sentiment score also allows to creation of a dependent variable of tweet sentiment category that can contain whether the tweet is positive or negative and thus finally training over the dataset to allow the model to predict future tweets as per their sentiment. This not only allows us to filter tweets but also understand the overall sentiment of the population of the tweets originating over a certain topic or issue.

Acknowledgments

We would like to express our sincere gratitude to Professor Furqan Rustam for his invaluable guidance, unwavering

	Date	SVM Predictions - No. of Global Confirmed Cases
0	03/10/2023	698433704.000000
1	03/11/2023	699022634.000000
2	03/12/2023	699612595.000000
3	03/13/2023	700203588.000000
4	03/14/2023	700795613.000000
5	03/15/2023	701388671.000000
6	03/16/2023	701982763.000000
7	03/17/2023	702577890.000000
8	03/18/2023	703174053.000000
9	03/19/2023	703771253.000000

Fig. 32: SVM forecasting result

support, helpful guidance, and insightful critical feedback throughout this project on COVID-19.

REFERENCES

[1] Brad Dayley, Python Phrasebook: Essential Codes and Commands 1st Edition, ISBN- 978-0672329104, Sams Publishing, November 2006.

[2] Chodorow K, Dirolf M; MongoDB in the Era of Pandemics: A Scalable Approach to Data Storage, O'Reilly Media, ISBN: 978-1449344689, 2013.

[3] McKinney, W; Pandas and PostgreSQL: A Comprehensive Approach to Data Structuring, O'Reilly Media, ISBN: 978-1491957660, 2017.

[4] Ingo Steinwart, Andreas Christmann; Support Vector Machines: Guide books, ACM Digital Library, Springer Publishing Company, ISBN: 978-0-387-77241-7, Published:12 August 2008, https://dl.acm.org/doi/10.5555/1481236.

[5] Osvaldo Martin, Bayesian Analysis with Python: Unleash the power and flexibility of Bayesian Framework, Packt Publishing, 25 Nov 2016, ISBN 9781785889851.

[6] Saqib, Mohd.,*Forecasting COVID-19 Outbreak Progression Using Hybrid Polynomial-Bayesian Ridge Regression Model*, Applied Intelligence, 2021, 10.1007/s10489-020-01942-7.

[7] Shoko C, Sigauke C, *Short-term forecasting of COVID-19 using support vector regression: An application using Zimbabwean data*, Am J Infect Control. 2023 Oct; doi:10.1016/j.ajic.2023.03.010, 30 March 2023, PMID: 37001592; PMCID: PMC10060190.

[8] Folium 0.14.0 Documentation Github, Public Contribution, https://python-visualization.github.io/folium/version-v0.14.0/index.html

[9] Yalong Yang, Tim Dwyer, Kim Marriott, Bernhard Jenny, Sarah Goodwin, Tilt Map: Interactive Transitions Between Choropleth Map, Prism Map and Bar Chart in Immersive Environments, IEEE Transactions on Visualization and Computer Graphics, 10.1109/TVCG.2020.3004137, 2021.