COVID-19 Trend Analysis using Machine Learning Techniques

Abhishek Jaglan
B.Tech Student - CSE
Dr. Akhilesh Das Gupta Institute of Technology and Management
New Delhi, India
jaglanabhishek08@gmail.com

Daksh Trehan
B.Tech Student - CSE
Dr. Akhilesh Das Gupta Institute of Technology and Management
New Delhi, India
daksh.trehan@hotmail.com

Priyansh Singhal
B.Tech Student - CSE
Dr. Akhilesh Das Gupta Institute of Technology and Management
New Delhi, India
singhalpriyansh58@gmail.com

Ms Megha
Assistant Professor - CSE
Dr. Akhilesh Das Gupta Institute of Technology and Management

***Abstract*—With COVID-19 being the highlight of the decade and information related to it on the rise in an unorganized manner, the need for a centralized platform to gather information from on an international front doesn't seem to be far fetched. The data dashboard is formed with help of data taken from reliable sources to portrait it in an interactive and easy to consume format with salient features like a chatbot, cases prediction with help of machine learning and projection, of data in numerous formats updated daily. The paper aims to provide a better understanding to neophytes regarding the current trend of coronavirus in the world along with imparting basic knowledge about the deadly virus.**

Keywords— *COVID-19, Machine Learning, Prediction, Data Dashboard*

1. INTRODUCTION

This project is one of the coronavirus related theme projects. It is a machine learning based website for a data dashboard. A data dashboard is an information management tool that visually tracks, analyses and displays key performance indicators (KPI), metrics and key data points to monitor the specific process. The dashboard consists of two fronts: front and back. The back end consists of data gathering, data preparation, data analysis, chat bot and machine learning, all of which is implemented using Python. The front end consists of making the website, converting the processed information at backend to a consumable form, and deploying all these features online.

At the back-end data for prediction and showcasing data for different purposes was gathered from the official repository of John Hopkins University. For chatbot the data was taken from cdc.gov.in to fetch questions, and answer to faqs.

At the front-end the files were processed into consumable material for website building purposes using python based open-source framework Django. The website was made presentable and interactive using CSS and HTML.
All these features combined formed a live data dashboard as a website updating itself daily, showing total number of cases for each country separately and in form of a world map for better relative understanding of the situation. It also portraits the recovering and infected cases for each countries in a graphical form for detailed view. The dashboard gives you an option to put in your queries and get the answers to them in form of a chatbot along with giving prediction of total number of cases for each country in near future. The website also offers you a feature to download the data in four different form (png, svg vector image and pdf, jpeg).

Basic architecture of the COVID-19 data dashboard is shown in Fig 1. and MVC (Model View Controller) architecture for desired visual effects on the dashboard is shown in Fig 2.

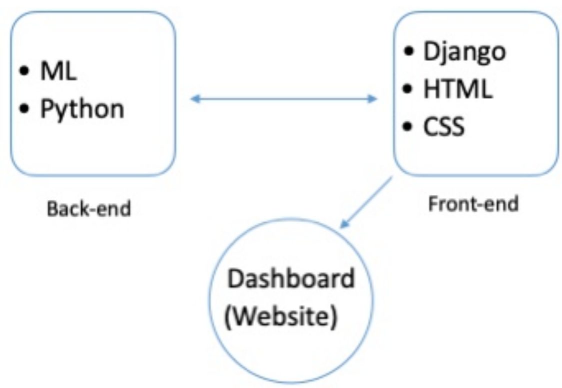

Fig 1. Data Dashboard Architecture

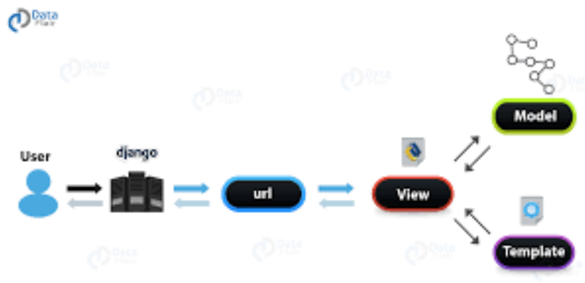

Fig 2. MVC Architecture

2. PREDICTION

2.1 Data Procurement and Preparation

The dataset that has been utilized in prediction is fetched from data archive for "2019 Novel CoronaVirus Visual Dashboard" managed by Center for Systems Science and Engineering (CSSE) at Johns Hopkins University, also, Supported by ESRI Living Atlas Team and the Johns Hopkins University Applied Physics Lab (JHU APL). The data fetched can be regarded as parameterized dataset having pertinent parameters including State, Country, Latitude count, Longitude count and dates. To take in account the data for Confirmed, Death and Recovered cases, separate dataset has been used.

The dataset conscripted is continuous dataset and therefore, is well suited for regression analysis as it needs to predict from continuous dependent variables from various independent ones. The relation between dependent and independent variables can be defined by coefficient of both variables in regression mathematical statement.

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots\ldots + \theta_m x_m$$
$$\text{where } \theta_0 : \text{Bias}$$
$$\theta_1, \theta_2, \ldots \theta_m : \text{Weights with "m" degree of polynomial}$$

2.2 Feature Selection

It involves tailoring our data for best results from our model. Feature Engineering can influence the performance of our model; thus, it is important to choose it precisely. Including impeccable and succinct feature can help us to know better about the framework of our data. The process includes delving and aggregating or decomposing required features to produce new characteristics or alter the provided ones and destigmatizing irrelevant parameters.

To cope up with our prediction model, we have removed squandered parameters such as Combined_key, Latitude count, Longitude count, FIPS. For further, easy processing we have converted the dates to date-time object using strptime().

2.3 Regression Analysis

To predict the future cases, causalities, recovery cases, we have employed Linear Polynomial Regression. Linear Regression is operated on two continuous variables to find relationship between them. Linear Polynomial Regression that can be seen in Fig 3. can be regarded as extended version of Linear Regression, it is implemented on related but non-linear data. It is supervised in nature, and handles non-linear data efficiently.

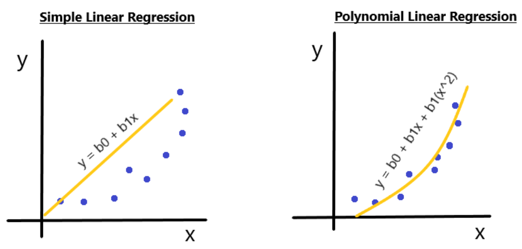

Fig 3. Shows Simple and polynomial Linear Regression

2.4 Implementation

The data has been spitted in 4:1 for training and testing respectively. On implying degree=3, the model can show accuracy up to 71% for worldwide cases, 95.9% for worldwide recovery cases and 93.33% for death cases worldwide. Fig 4. Shows the split of data for training & testing, and the accuracies obtained for different cases using polynomial linear regression.

```
def prediction(x):
    x_train_confirmed, x_test_confirmed, y_train_confirmed, y_test_confirmed = train_test_split(days_since_1_1, x,
                                                                                 test_size=0.2, shuffle=False)
    # transform data for polynomial regression
    model = PolynomialFeatures(degree=2)
    model_x_train_confirmed = model.fit_transform(x_train_confirmed)
    model_x_test_confirmed = model.fit_transform(x_test_confirmed)
    model_future_forecast = model.fit_transform(future_forecast)

    # polynomial regression
    linear_model = LinearRegression(normalize=True, fit_intercept=False)
    linear_model.fit(model_x_train_confirmed, y_train_confirmed)
    test_linear_pred = linear_model.predict(model_x_test_confirmed)
    linear_pred = linear_model.predict(model_future_forecast)
    print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
    print('MSE:',mean_squared_error(test_linear_pred, y_test_confirmed))

    plt.plot(y_test_confirmed)
    plt.plot(test_linear_pred)
    plt.legend(['Test Data', 'Polynomial Regression Predictions'])
    print(linear_model.score(model_x_test_confirmed,y_test_confirmed))

    linear_pred = linear_pred.reshape(1,-1)[0]
    model_df = pd.DataFrame({'Date': future_forecast_dates[-20:], 'Predicted number of Confirmed Cases': np.round(linear_pred[-2
    print(poly_df)
def plot_predictions(x, y, pred, algo_name, color):
    plt.figure(figsize=(16, 9))
    plt.plot(x, y)
    plt.plot(future_forecast, pred, linestyle='dashed', color=color)
    plt.title('Number of Coronavirus Cases Over Time', size=30)
    plt.xlabel('Days Since 1/1/2020', size=30)
    plt.ylabel('Number of Cases', size=30)
    plt.legend(['Confirmed Cases', algo_name], prop={'size': 20})
    plt.xticks(size=20)
    plt.yticks(size=20)
    plt.show()
```

Fig 4. Shows the split of data for training & testing, and the accuracies obtained for different cases using polynomial linear regression in Python

Predicting Cases : The model trained and tested using polynomial linear regression is used to predict the total number of cases on each day. Fig 5. shows the predicted values, and Fig 6. shows the comparison between test data and predicted values.

```
prediction(world_cases)
plot_predictions(adjusted_dates, world_cases, linear_pred, 'Polynomial Regression Predictions', 'yellow')

MAE: 4610620.4577909075
MSE: 34492258688874.74
0.7165876464038675
          Date  Predicted number of Confirmed Cases
0   12/18/2020                           68014520.0
1   12/19/2020                           68461915.0
2   12/20/2020                           68910774.0
3   12/21/2020                           69361097.0
4   12/22/2020                           69812884.0
5   12/23/2020                           70266136.0
6   12/24/2020                           70720852.0
7   12/25/2020                           71177032.0
8   12/26/2020                           71634677.0
9   12/27/2020                           72093786.0
10  12/28/2020                           72554359.0
11  12/29/2020                           73016396.0
12  12/30/2020                           73479898.0
13  12/31/2020                           73944864.0
14  01/01/2021                           74411294.0
15  01/02/2021                           74879188.0
16  01/03/2021                           75348547.0
17  01/04/2021                           75819369.0
18  01/05/2021                           76291657.0
19  01/06/2021                           76765408.0
```

Fig 5. Shows the predicted values for total cases on each day

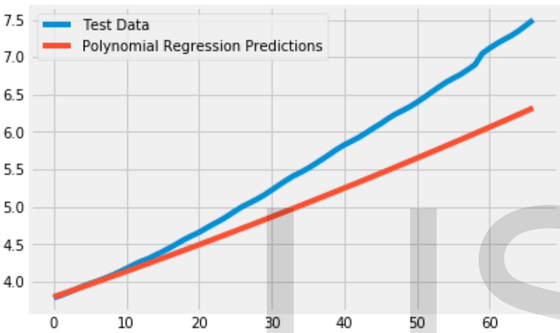

Fig 6. Shows the comparison between test and predicted total values

Predicting Recoveries : The model trained and tested using polynomial linear regression is used to predict the total number of recovered cases on each day. Fig 7. shows the predicted recovery values for each day, and Fig 8. shows the comparison between test data and predicted recovery values.

```
prediction(total_recovered)
plot_predictions(adjusted_dates, recovery_rate, linear_pred, 'Polynomial Regression Predictions', 'yellow')

MAE: 817288.5836635466
MSE: 1427308387724.5173
0.9597446903508242
          Date  Predicted number of Confirmed Cases
0   12/18/2020                           50042390.0
1   12/19/2020                           50398679.0
2   12/20/2020                           50756219.0
3   12/21/2020                           51115008.0
4   12/22/2020                           51475048.0
5   12/23/2020                           51836338.0
6   12/24/2020                           52198879.0
7   12/25/2020                           52562670.0
8   12/26/2020                           52927711.0
9   12/27/2020                           53294002.0
10  12/28/2020                           53661544.0
11  12/29/2020                           54030336.0
12  12/30/2020                           54400378.0
13  12/31/2020                           54771671.0
14  01/01/2021                           55144213.0
15  01/02/2021                           55518006.0
16  01/03/2021                           55893050.0
17  01/04/2021                           56269343.0
18  01/05/2021                           56646887.0
19  01/06/2021                           57025681.0
```

Fig7. Shows the predicted recovery values for recovery cases on each day

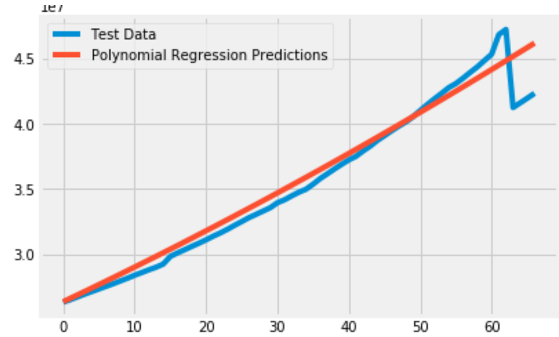

Fig 8. Shows the comparison between test data and predicted recovery values

Predicting Deaths : The model trained and tested using polynomial linear regression is used to predict the total number of death cases on each day. Fig 9. shows the predicted death values, and Fig 10. shows the comparison between test data and predicted death values.

```
prediction(total_deaths)
plot_predictions(adjusted_dates, total_deaths, linear_pred, 'Polynomial Regression Predictions', 'yellow')

MAE: 39924.64938667233
MSE: 1980200426.2692726
0.9337565967812702
          Date  Predicted number of Confirmed Cases
0   12/18/2020                            1715532.0
1   12/19/2020                            1723959.0
2   12/20/2020                            1732406.0
3   12/21/2020                            1740870.0
4   12/22/2020                            1749354.0
5   12/23/2020                            1757856.0
6   12/24/2020                            1766377.0
7   12/25/2020                            1774916.0
8   12/26/2020                            1783474.0
9   12/27/2020                            1792051.0
10  12/28/2020                            1800646.0
11  12/29/2020                            1809260.0
12  12/30/2020                            1817892.0
13  12/31/2020                            1826544.0
14  01/01/2021                            1835213.0
15  01/02/2021                            1843902.0
16  01/03/2021                            1852609.0
17  01/04/2021                            1861335.0
18  01/05/2021                            1870079.0
19  01/06/2021                            1878842.0
```

Fig 9. Shows the predicted values for death cases on each day

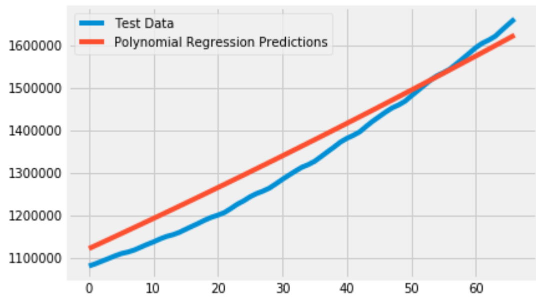

Fig 10. Shows the comparison between test data and predicted death values

3. CHATBOT

3.1 Data Procurement and Preparation

The data is fetched from Frequently Asked Questions section of official website of Center for Disease Control and Prevention using requests and BeautifulSoup library. The dataset includes 70 different questions regarding general awareness of public towards Novel 2019 Coronavirus. The queries and their solutions are collected separately and dumped in json files, which are then aggregated to create a useful data frame.

3.2 TF-IDF Vectorization

To follow-up with our chatbot, we have employed Bag-of-word model using TF-IDF Vectorization, that converts text to feature vectors, thus making estimation a cinch.

Textual data can't be employed on our model directly, instead to make them work, we need to convert them to numerical vectors. The model isn't intricate, all it focuses is on occurrence of words in the document. It can be achieved by assigning a unique number to each word, and the given data can be encoded with the length of vocabulary of known words. The Bag-of-word model is all about the words present in the document and their degree regardless of the order of occurrence.

To calculate word frequencies, TF-IDF Vector is most popular method. It stands for "Term Frequency-Inverse Document Frequency" that stores component of resulting scores assigned to each word. Some words like, "the", "is" might appear a lot often in our document, but that certainly isn't going to help our encoded vector. The goal of TF-IDF vector is to calculate the word-frequency scores for highlighted text that are more interesting. "Term Frequency (TF)" calculates the frequency for each word, whereas, "Inverse Document Frequency (IDF)" downscales the score of much frequently occurring word.

Fig 11. shows TF-IDF Vectorizer automatically running tokenization followed by learning vocabulary and calculating the inverse document frequency weights to calculate score for each word.

There is a very high possibility that the user will not enter same questions as in our corpus, although we can expect the meaning and insight to be same but we can never expect the words to be same too. The next challenge we face is to create similarities between the question corpus that we have fetched and the queries asked by user. To match the question asked by user to the question corpus deployed in our data, we use cosine similarity.

Cosine Similarity is a metric that is used to determine similarity between texts regardless of their size. It tends to determine the cosine angle between two vectors that are projected in multi-dimensional space.

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

```
vectorizer = CountVectorizer()
count_vec = vectorizer.fit_transform(df['Questions']).toarray()
```

```
def COVIDbot(user_response):
  text = vectorizer.transform([user_response]).toarray()
  df['similarity'] = cosine_similarity(count_vec, text)
  return df.sort_values(['similarity'], ascending=False).iloc[0]['Answers']
```

```
COVIDbot('what is coronavirus?')
```

```
'A novel coronavirus is a new coronavirus that has not been previously identified. The virus causing coronavirus disease 2019 (COVID-19),
is not the same as the coronaviruses that commonly circulate among humans\xa0and cause mild illness, like the common cold.\nA diagnosis w
ith coronavirus 229E, NL63, OC43, or HKU1 is not the same as a COVID-19 diagnosis. Patients with COVID-19 will be evaluated and cared for
differently than patients with common coronavirus diagnosis.\n'
```

Fig 11. Shows TF-IDF Vectorizer automatically running tokenization

4. DASHBOARD FORMATION

As we know Django is python based open-source framework which follows MVC(Model View Controller) architectural pattern used in the rapid development of the website with clean design without worrying too much about setting up an environment to start.

For Covid-19 Dashboard data fetched online server is analyzed and used many approaches to serializes the data which is used in views.py files in django to further process the data and represent it to the user.

Data fetched is sent to html(front-end) file for representation in the form of maps, texts as well as charts and to display these chart special Library Chart.js is used . It also involves the part of Chatbot which compare the text entered by user and similar results present to user's questions in file data fetched from online server.

All the data fetched from backend is displayed in front end with the help of styling using CSS and JavaScript . Main goal of the project to make site user attractive and solving queries using Chatbot . Django worked as an intermediate between data analyzed using Machine learning and Representation of data to User.

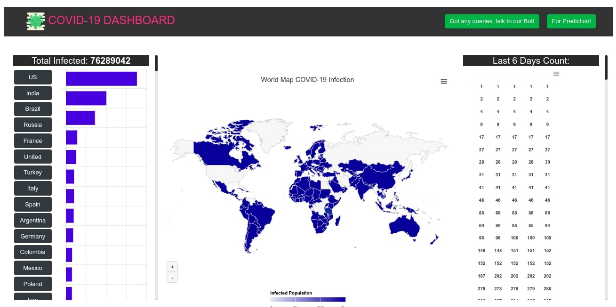

Fig 12. COVID-19 Data Dashboard

5. CONCLUSION AND FUTURE SCOPE

The research paper successfully shows the analysis of the gathered covid-19 data of world. Predicting further transmission of Covid-19 or coronavirus could be useful in stopping its 6further spread. The paper purposefully displays the comprehensive steps taken to implement the data dashboard for better understanding of the data and interactive information exchange which will be helpful in further taking necessary steps to manage the resources for its containment. Furthermore, different features such as Chatbot and Prediction of total cases, recoveries and deaths for the same are executed. These features were executed using TF-IDF vectorization and Machine Learning model based on polynomial regression analysis but in future with further availability of data more complex technology and algorithms can be put to work for better prediction and understanding of the virus' spread. The paper clarifies that polynomial regression analysis gives good accuracy rate and answers questions about its near future impact in terms of human life. This project highly depends on realtime data and its availability is dependent on external factors but as time goes on the models will be able to learn more and more due to heavy flow of data through them giving more accurate and reliable results. For further advancement in the accuracy rate of the model different attributes can be included during the process. Hope this article contributes to the world's response to this epidemic and puts forward some references for further research in future.

6. REFERENCES

[1] Ekta Gambhir, Ritika Jain, Alankrit Gupta, Uma Tomar, "Regression Analysis of COVID-19 using Machine Learning Algorithms" - https://ieeexplore.ieee.org/document/9215356/references#references

[2] Features Evaluation and Treatment Coronavirus (COVID-19), May 2020, [online] Available: https://www.ncbi.nlm.nih.gov/books/NBK554776/

[3] N. S Punn, S. K. Sonbhadra and S. Agarwal, "COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms", medRxiv, June 2020, [online] Available: https://doi.org/10.1101/2020.04.08.20057679

[4] P. Ghosh, R. Ghosh and B. Chakraborty, "COVID-19 in India: State-wise Analysis and Prediction", medRxiv, May 2020, [online] Available: https://doi.org/10.1101/2020.04.24.20077792

[5] S. F. Ardabili, A. Mosavi, P. Ghamisi, F. Ferdinand, A. R. VarkonyiKoczy, U. Reuter, et al., COVID-19 Outbreak Prediction with Machine Learning, April 2020, [online] Available: https://ssrn.com/abstract=3580188

[6] F. Petropoulos and S. Makridakis, Forecasting the novel coronavirus COVID-19, March 2020, [online] Available: https://doi.org/10.1371/journal.pone.0231236

[7] M. K Arti and K. Bhatnagar, Modeling and Predictions for COVID 19 Spread in India, April 2020

[8] H. Shekhar, "Prediction of Spreads of COVID-19 in India from Current Trend", medRxiv, May 2020, [online] Available: 10.1101/2020.05.01.20087460

[9] L. Yan, H. Zhang, J. Goncalves et al., "An interpretable mortality prediction model for COVID-19 patients", Nature Machine Intelligence 2, pp. 283-288, May 2020, [online] Available: https://doi.org/10.1038/s42256-020-0180-7

[10] L. Li, Z. Yang, Z. Dang, C. Meng, J. Huang, H. Meng, et al., "Propagation analysis and prediction of the COVID-19", Infectious Disease Modelling, vol. 5, pp. 282-292, 2020, [online] Available: https://doi.org/10.1016/j.idm.2020.03.002

[11] S. Zhao and H. Chen, "Modeling the epidemic dynamics and cont rol of COVID-19 outbreak in China", Quantitative Biology, vol. 8, no. 1, pp. 11-19, March 2020, [online] Available: https://doi.org/10.1007/s40484-020-0199-0

[12] J. Xie, Z. Tong, X. Guan et al., "Critical care crisis and some recommendat ions during the COVID-19 epidemic in China", Intensive Care Med, vol. 6, no. 6, pp. 837-840, June 2020, [online] Available: https://doi.org/10.1007/s00134-020-05979-7

[13] W. C. Roda, M. B. Varughese, D. Han and M. Y. Lia, "Why is it difficult to accurately predict the COVID-19 epidemic?", Infectious Disease Modelling, vol. 5, pp. 271-281, 2020, [online] Available: https://doi.org/10.1016/j.idm.2020.03.001

[14] W. Naudé, Artificial intelligence vs COVID-19: limitations constraints and pitfalls, pp. 1-5, Apr 2020.

[15] R. Gupta, S.K. Pal and G. Pandey, A Comprehensive Analysis of COVID-19 Outbreak Situation in India, April 2020

[16] V. Bindhu, "Biomedical Image Analysis Using Semantic Segmentation", Journal of Innovative Image Processing (JIIP), vol. 1, no. 02, pp. 91-101, 2019

[17] A. Chandy, "A Review On Iot Based Medical Imaging Technology For Healthcare Applications", Journal of Innovative Image Processing (JIIP), vol. 1, no. 01, pp. 51-60, 2019