Analysis of Covid19 Outbreak

Hotspot detection And Forecasting Cases

## ABSTRACT

An efficient way to control the spread of a disease is the need of the hour. As highlighted by the ongoing COVID-19 pandemic, the virus can spread to all parts of the world in the matter of a few days. Using epidemiologic information and applying suitable clustering techniques, it is possible to visualize the hotspots of the disease and find out areas which require intensive care. This project proposes a model of visualizing the current, district wise reported cases for the pandemic in India, and identifying the areas which are highly likely to contribute to the spread of the virus using clustering, We have used various clustering techniques like KMeans, DBSCAN and HDBSCAN in this project and have compared the performance of each of the algorithms on the dataset. India is among the countries where testing was slow to begin owing to it’s not so good medical infrastructure. Given this fact we need to be able to predict future cases in order to provide adequate medical support on time. In order to fulfill this need, we try to use machine learning models to predict cases in India.

## I.  INTRODUCTION

The newly discovered coronavirus, called the COVID-19, is an infectious disease that originated in China but has spread to all parts of the world since. Majority of the people infected with the coronavirus will experience respiratory issues but will recover without any special medication required. However, the elderly, or the people with some pre-existing health problems are prone to develop a serious illness. Even though there are many ongoing trials for finding a treatment, there are no vaccines as of yet to counter the virus. So, the next best course of action, is to be well informed about the virus and know how it spreads.

To help in doing so, we have visualized the current scenario using a self-modeled district wise dataset with various techniques and have also applied machine learning to find out what are essentially the ‘hotspots’ for the disease in India. We have achieved this by performing clustering, which is an unsupervised characterization of observations from a dataset.

This will enable us to find out which places in the country require extensive testing and healthcare stations. More number of stations will enable us to increase testing, which will help in providing suitable medication for the infected. For the scope of this project, our focus will solely be on our country, India. Unfortunately, India has a limited capacity for testing and therefore, the densely affected areas should be prioritized to help in flattening the curve.

This unprecedented event has caused us to value informed and analyzed decisions to be able to come up with efficient and optimized solutions to the problems. In order to achieve this, we turn to machine learning for pattern recognitions and predictions of cases to prepare early in order to cater for the needs of the infected and thus save lives.

The LSTM models have proved to be very accurate in identifying patterns with long dependencies which is very true for a communicable disease, where the infected cases today decide the infected cases tomorrow under normal circumstances.

Given enough data these deep learning models can accurately model time series data thus any deviations from the predicted results is attributed to the community’s effort in containing the virus via different methods including lockdown, social distancing, sanitary measures etc. in order to stop its spread.

## II. LITERATURE REVIEW

As our chosen topic involves a recent ongoing pandemic, the research done and datasets available have not really been verified yet. Also, since our project involves multiple machine learning concepts, we have divided the literature review based on those concepts as well.

Various machine learning algorithms are being used in order to battle COVID-19. Two graduates of Columbia University, “Andrew Satz” and “Brett Averso”, have started a startup known as EVQLV that can create, screen and optimize a huge amount of therapeutic bodies. Using this technique, they are trying to discover treatments that can help the virus affected people. The algorithm developed by the pair is capable of screening the antibodies rapidly with a very high success rate. [1]

IBM Research is also helping researchers in various fields using its innovative technologies in order to accelerate the research of finding a way to defeat the virus. The company is not only providing its supercomputers to researchers in order to understand the virus in a better way, also the Big Blue is trying to make various artificially intelligent and cloud-based technologies for free so that doctors, researchers can use it to find the solution to COVID-19 as soon as possible. [2]

In the current ongoing global pandemic due to the COVID19 outbreak, the implementation of unsupervised learning algorithms is now more necessary than ever. Clustering can prove to be an effective tool for understanding and analyzing the prevalence of a disease and how it spreads.

In KMeans clustering, we are essentially dividing the dataset into a number of smaller datasets with each one belonging to a different central point. This method has been used in a research conducted by “Parasian D.P Silitonga” in 2017, in which he clustered the patient’s disease data at “Haji Adam Malik” Hospital in Medan. The results provided illustrations of the patterns of disease tendencies of the patients who went to that hospital and ultimately helped the hospital anticipate priority of service based on the trends in the pattern. [3]

DBSCAN is a density-based clustering method. “G.M Nandana”, “Shuchi Mala” and “Ashok Rawat” used DBSCAN to detect the dengue fever hotspots. They have used a clustering approach which analyzes the spatial dynamics of the dengue epidemic over an area. This way, clusters of dengue fever were found in Delhi which helped them identify the hotspots. [4]

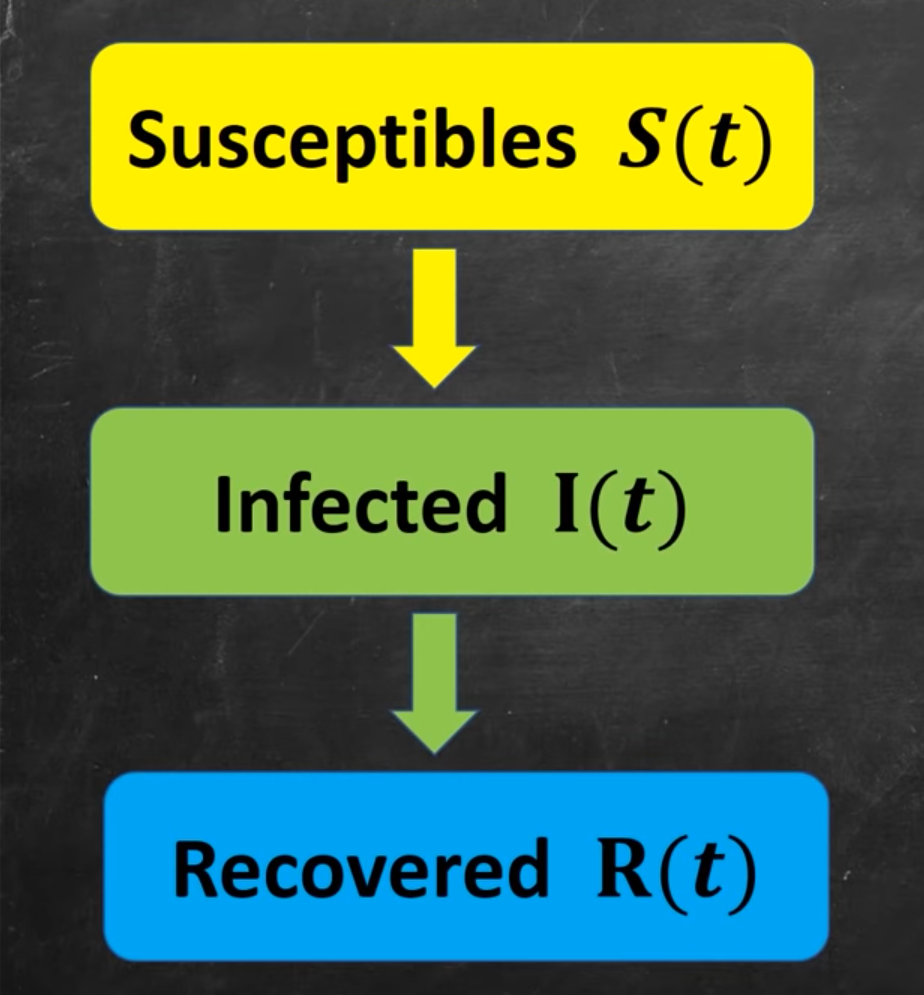
HDBSCAN is a clustering algorithm which was initially developed by “Campello”, “Moulavi” and “Sander”. It is an extended version of DBSCAN which is done by first converting it into a hierarchical clustering algorithm. “Weimo Liu” with his team worked to find out the advantages of density-based clustering over Location based services for clustering geological datasets. They experimented with HDBSCAN on “Yahoo Flickr, Zillow, redfin and Capital Bikeshare”. There has been extensive work on spatial data mining of which clustering is a major technique. [5]

Aside from clustering and classifying existing cases, modelling and forecasting the outbreak of a disease can forewarn us about the extent of damage that might come our way and can help us prepare adequately to alleviate the situation. For the ongoing coronavirus, “Cleo Anastassopoulou”, “Lucia Russo”, “Athanasios Tsakris” and “Constantinos Siettos” have modelled and forecasted the COVID-19 outbreak. On the basis of epidemiological data publicly available for Hubei, China, they gave estimates for the fatality and recovery ratios. By using a Susceptible-Infectious-Recovered-Dead model, they gave estimates for the basic reproduction number (Ro) along with day wise infection mortality and recovery rates. They also attempted to forecast the outbreak of the disease at the epicenter weeks before it actually happened. [6]

## III. MATHEMATICAL MODELLING

The researchers around the world are trying to study the mathematics of spread of epidemic, the predicted deaths, or total time required for this virus to end completely. One of the simplest models to study it is the SIR model. The people in the world can be divided into three categories. First ones are Susceptible, the people who are likely to get infected from the virus. Hence all the people can be susceptible at the initial stage. Next are the infected people, ones who get sick after getting infected from the virus. Next comes the Recovered people, that is when a person’s immune system kicks in. We are also considering that if a person recovers, he cannot get infected again and hence will not infect other susceptible too. Considering the death rate to be low, everyone in the world can be grouped under these three categories. So, this situation can be represented as an equation.

S(t) + I(t) + R(t) = N



(caption: Cascade of classes)

Now, at time t=0 or the initial stage

S(t) = S0

I(t) = I0

R(t) = 0

At the initial stage, recovered people’s count will obviously be zero, because it is the initial stage. The count of infected people would probably be just 1, but due to lack of checkups we will consider it as I0. Now, susceptible will have almost all the people in the world, hence we will term that number as S0. In order to govern the behaviour of S(t), I(t), R(t), we will try to analyze them using a system of differential equations.

= -aSI

= aSI - bI

= bI

Talking about transition from susceptibles to infected, it depends on the interaction of infected people and susceptibles. More are the interactions; more are the chances of more people getting infected. Also, ‘a’ is a positive constant denoting probability of a person getting infected. Also, bI is the number of infected people getting recovered. We will not solve these nonlinear differential equations, SI part bringing the nonlinearity, but we will try to analyze the behaviour using a graph.

A close up of a map

Description automatically generated

Here, we have plotted the rate of change of S, I and R with respect to time t. We have considered a town with a population of 500 people and course of the epidemic to be 60 days. We are also assuming that this epidemic will hit everyone in this town. So, they’ll have to go through the same process of getting infected and then recovering from that disease. The dS/dt curve will start from 500 and will reach 0 eventually. Whereas, dR/dt curve will start from zero initially and will reach 500 at the end of 60 days. dI/dt curve is an interesting curve which initially goes up but goes down eventually. In the initial stage, when the number of susceptibles is greater than number of recovered, the positive part of dI/dt will be dominant and the graph will go upwards. But as soon as the number of recovered people cross the susceptibles, the negative part will dominate and the graph will start going down.

Zooming a biton dI/dt equation

|t=0 = aS0I0 - bI0

In order to check whether there is an epidemic or there is no epidemic, we need to check whether dI/dt is less than zero or not.

(aS0 - b)I0 < 0,

aS0/b < 1

We can give this ratio a name

R0 = aS0/b

If R0 is less than 1 then there is no epidemic, but if it exceeds 1 there is an epidemic. We can change R0 by manipulating it’s constants. ‘a’ can be changed by lowering the transmission rate which can be done by various healthcare techniques, such as quarantine, washing hands regularly. ‘S0’ can be manipulated by lowering the initial susceptibles, say there is a vaccine for the virus and it is already given to half of the people before the virus attacks. We can directly half the value of S0 and hence the probability of an epidemic is halved. ‘b’ is a constant stating the number of recovered people from the infected stage. This can not be governed by us because it fairly depends on the person’s immunity to kick in.

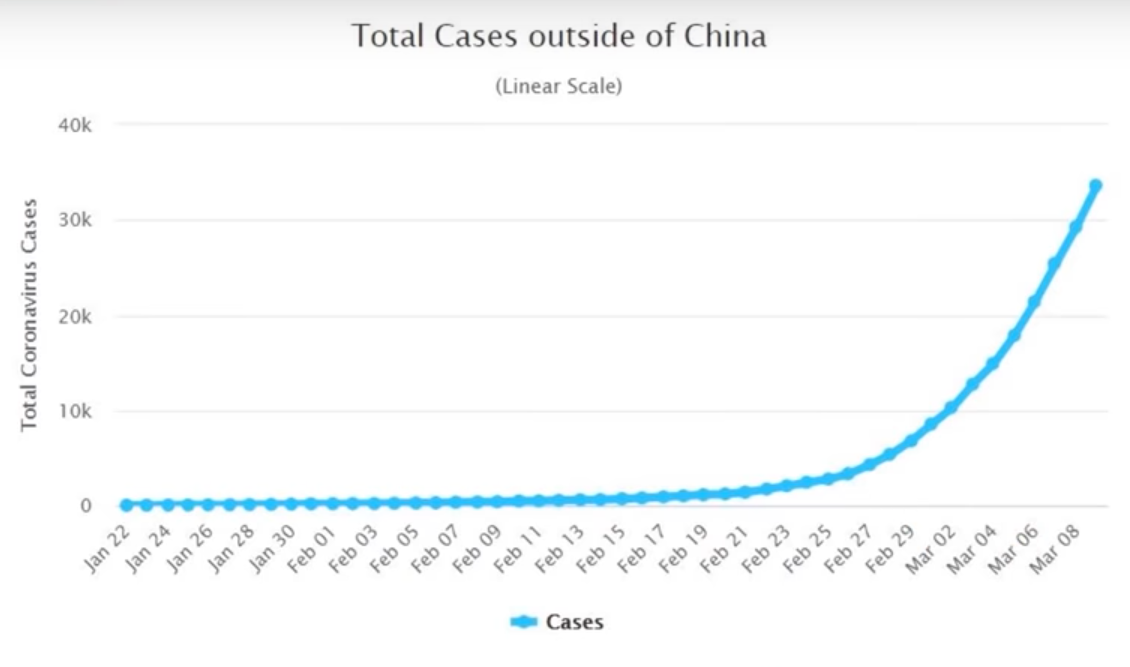
Now, looking at the situation around time t=0,

= I(aS0 - b)

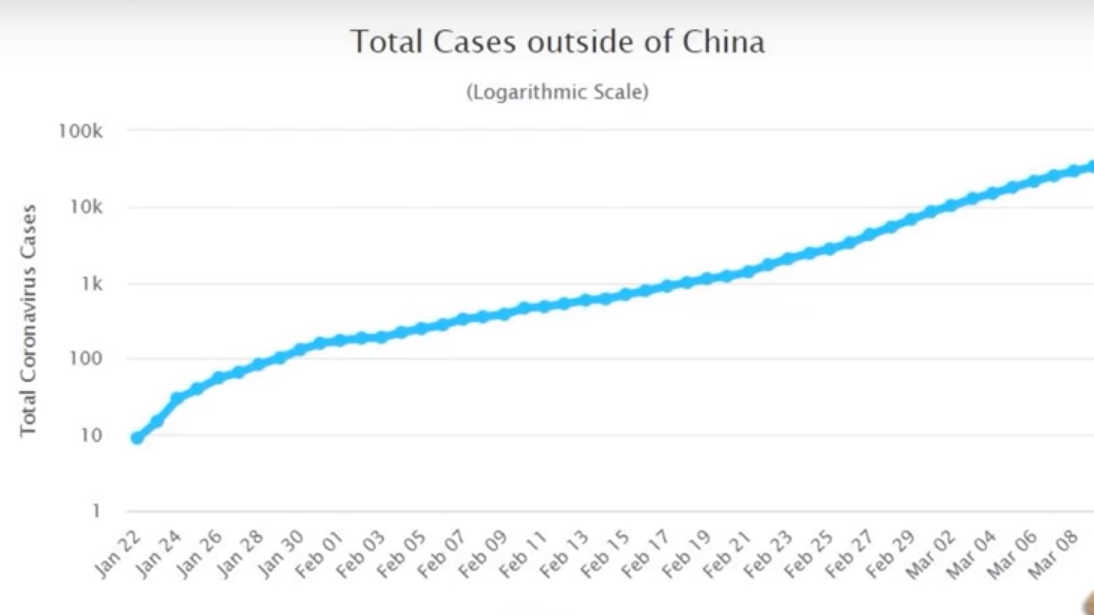
We can consider S0 to be a constant which is only true around time t=0. So, this is a derivative of I where I is just multiplied by a constant. So, a candidate solution for this differential equation can be

I(t) =

So, we can say that the cases will rise exponentially at the initial stage when S0 is constant.



(caption: linear representation of cases)



(caption: logarithmic representation of cases)

These are graphs from a site worldometer.com. We can clearly see that cases are rising exponentially at the initial stage. Also the logarithmic representation is somewhat a straight line. So, we can say that the SIR model can somewhat accurately depict the spread of COVID-19.

## III. METHODOLOGY

### HOTSPOT DETECTION

### Preparing the dataset

We have collected and modeled the data available on (<https://www.mohfw.gov.in/pdf/DistrictWiseList324.pdf>) suitable to our requirement. The dataset comprises of 4 columns, i.e. Name of District, longitude, latitude, Number of people affected. Preparing the data in geographical format helped us in visualizing and clustering it effectively.

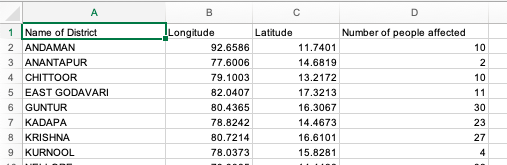


Figure 1: Snapshot of Dataset created

The dataset did not contain any duplicate or erroneous values and hence no external pre-processing was required.

### Visualizing the dataset

Data visualization is a technique to represent data in a visual context for the sake of better understanding. Certain trends and patterns can be easily noticed using visual representation that can go unnoticed in textual data.

We started by plotting the dataset created on google maps to verify the correctness of the dataset. This also helped us visualize the huge number of districts affected by the virus.

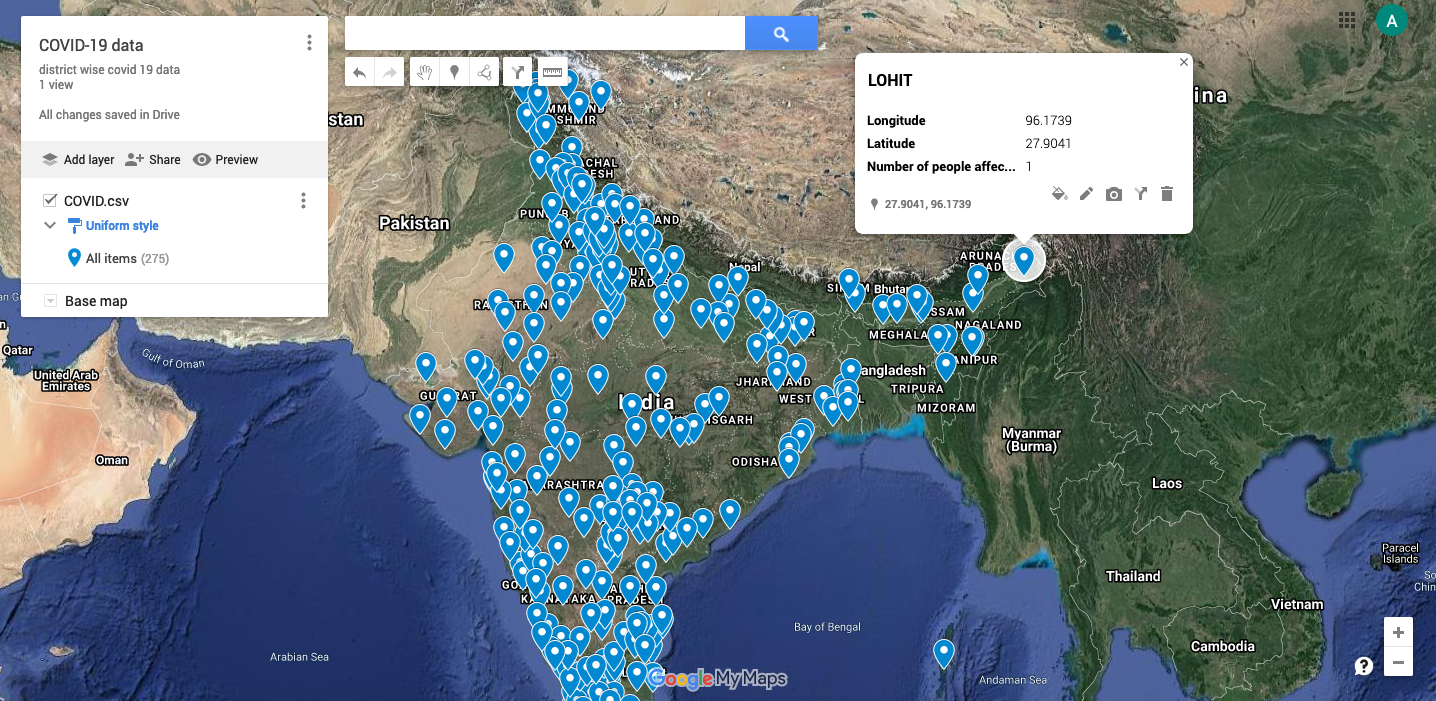


Figure 2: Visualizing the dataset on Google Maps

Using python, we first scaled the ‘Number of people affected’ column in the dataset to a range between 1 to 5 for making it more convenient while plotting the data. We used a scatter plot to visualize our dataframe and then used the folium library to depict the dataset on a map to make our visuals interactive.

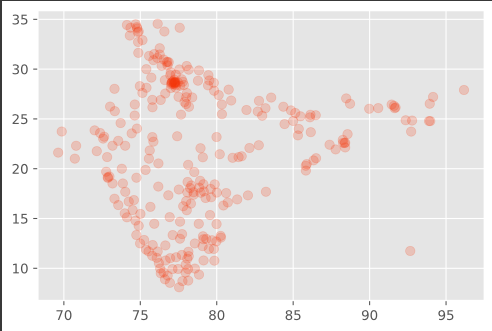


Figure 3: Scatter Plot for Data

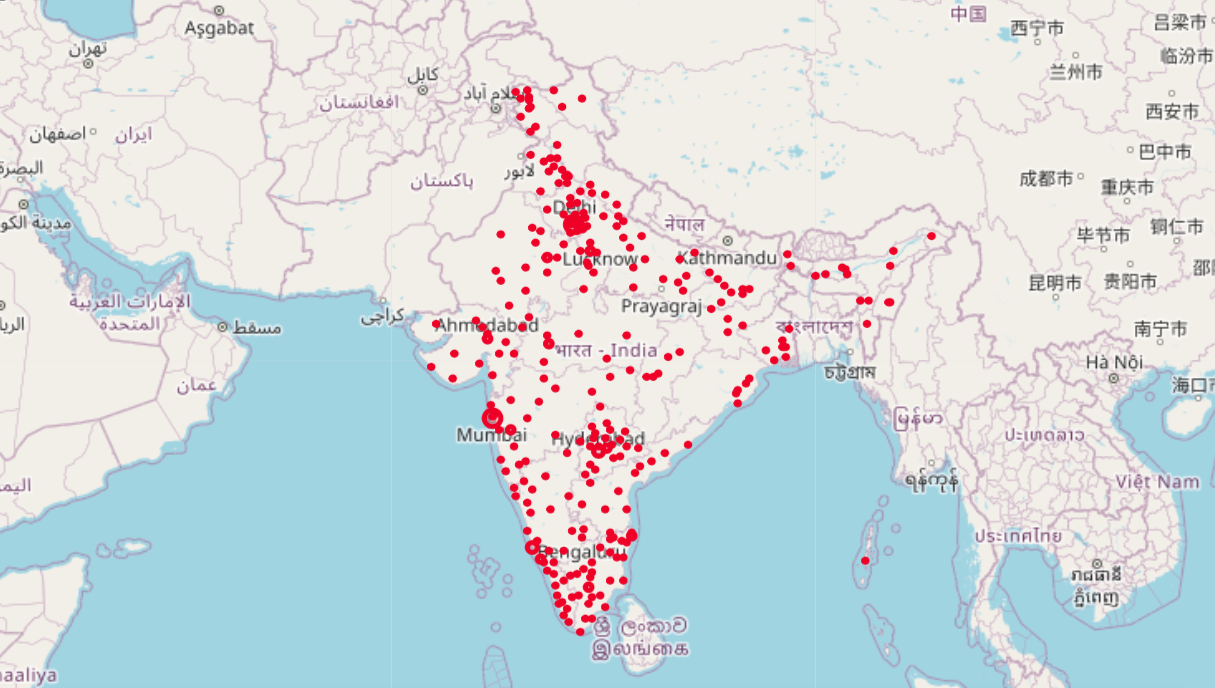


Figure 4: Visualization using Folium map

### 

### KMeans Clustering

In this project we implement K-means clustering as an integral part to identifying potential hotspots that may influence the proliferation of this epidemic. What K-means provides us with is an iterative algorithm that creates a distinction in a given dataset for a preordained non overlapping number of clusters where each point belongs to one specific cluster.

We calculate the optimal number of clusters that would produce the best silhouette score. To find the optimal ‘K’ value we used firstly The Elbow Method and then The Silhouette Method

1. Elbow Method: Find the Within-Cluster Sum of Squared Errors for multiple inputs of ‘K’, and select the ‘K’ for which WSS starts to linearly decrease. We then plot a WSS vs K plot and identify the elbow. WSS can be calculated first by finding the sum of squared errors for the distance of each data point and its cluster centroid and summing them up. The distance metric is similar to Euclidean distance
2. Silhouette Method: Silhouette score measures how similar a data point is to its assigned cluster in comparison to other clusters. For this method, we calculate the silhouette score for each value of ‘K’ or in other words for each iteration of the value K. We then plot the Silhouette score for each K against K. The silhouette score reaches global maximum at optimal value of K.

This Analysis tells us what value of K to use

K-means Clustering:

* First, we set the value of K or number of clusters
* We then import the class KMeans to perform the algorithm
* The algorithm will iterate over the datasets until each data point is assigned to a cluster and then new centroid is calculated
* The results of KMeans is then plotted as a scatter plot to visualize the clusters

### Density Based Clustering

Unlike KMeans, it is difficult to include sample weights of the points for the density-based clustering. So, to get an equivalent effect, we have repeated the data points in excel based on their value on the column ‘Number of people affected’. This was accomplished by writing a visual basic script on excel quickly duplicates the values based on a particular column value.



Figure 5: Script to duplicate values

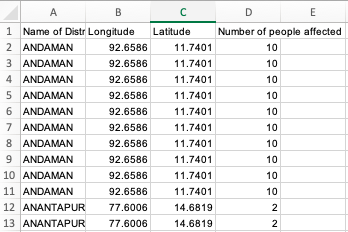


Figure 6: Modified dataset for Density Based Clustering

This new modified dataset helps us account for the number of cases too for each district while performing clustering.

1. **DBSCAN**

In this Algorithm, clusters are formed by combining nearby points and no such centroids exist for the clusters. For this project we have implemented DBSCAN from scratch and compared our results to the inbuilt implementation of DBSCAN through sklearn library.

Firstly, we begin with a random seed point which has a minimum number of points (Minpts) within a radius (eps). We then perform a Breadth First Search on all of the nearby points. For each nearby point, we’ll only grow the cluster forward if it has at least Minpts within the distance eps as its neighbors which will then be added to our FIFO queue, else we’ll call them leaf nodes and won’t expand our clusters from there.

Once we have finished the BFS, we won’t revisit those points as we successfully formed a cluster. We’ll then pick up another random seed point, which has not been included in the previous clusters and repeat the same process until all the points have been assigned a cluster.

Points which have fewer neighboring points than the minimum required and which aren’t leaf points for other clusters are called Noise points and are considered when selecting a new seed.

We finally return a list of cluster labels, where -1 indicates the noise points and the cluster numbering is started from 1.

2. **HDBSCAN**

HDBSCAN, which is a hierarchical density-based clustering approach was used to create clusters of the data of the affected people from COVID-19 all over India. It is an extended version of DBSCAN which uses certain parameters such as min\_samples, min\_cluster\_size, cluster\_epsilon. The algorithm follows the steps as mentioned below, but the HDBSCAN library in python helps us to implement HDBSCAN by just calling the library function and specifying the values according to the need of the model:

1. “Transform the space according to the density/sparsity.”
2. “Build the minimum spanning tree of the distance weighted graph.”
3. “Construct a cluster hierarchy of connected components.”
4. “Condense the cluster hierarchy based on minimum cluster size.”
5. “Extract the stable clusters from the condensed tree.”

Due to the radius specified in the density-based clustering algorithm there are many outliers present in the model which means those points are not a part of any cluster.

### Addressing outliers

Now as discussed above there are some left out points, i.e. they are not a part of any cluster. For data such as COVID-19 affected people, we cannot leave any district not clubbed with any other cluster because we need big clusters in order to analyze the spread of disease. So, we create two data frames, i.e. training and prediction. So, the unassigned nodes will be considered in the nearest cluster formed, with the help of this we can suspect the most possible reason of the disease reaching that district. The new model which is named hybrid in our project will have a comparatively higher Silhouette score because distant points are also taken into consideration while making clusters in this model. This means that there are more numbers of bigger clusters formed when there are no outliers left in the model.

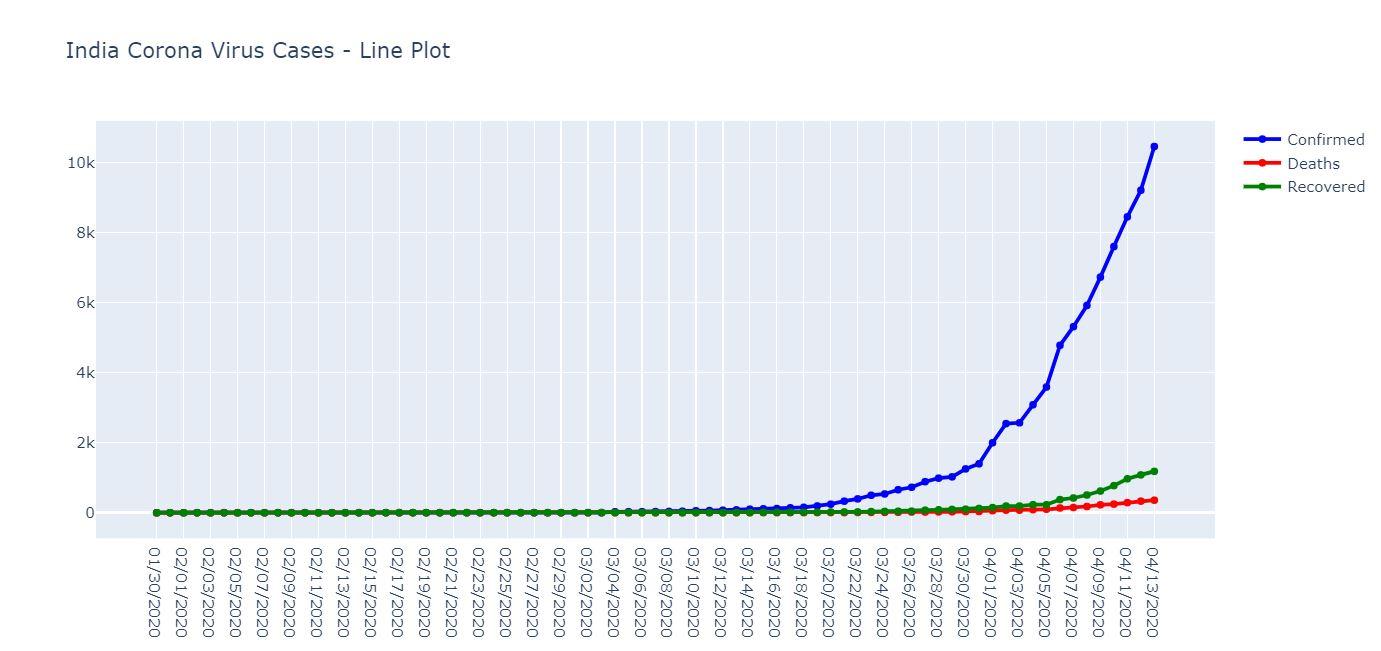
### COVID 19 FORECASTING

**Data Description**

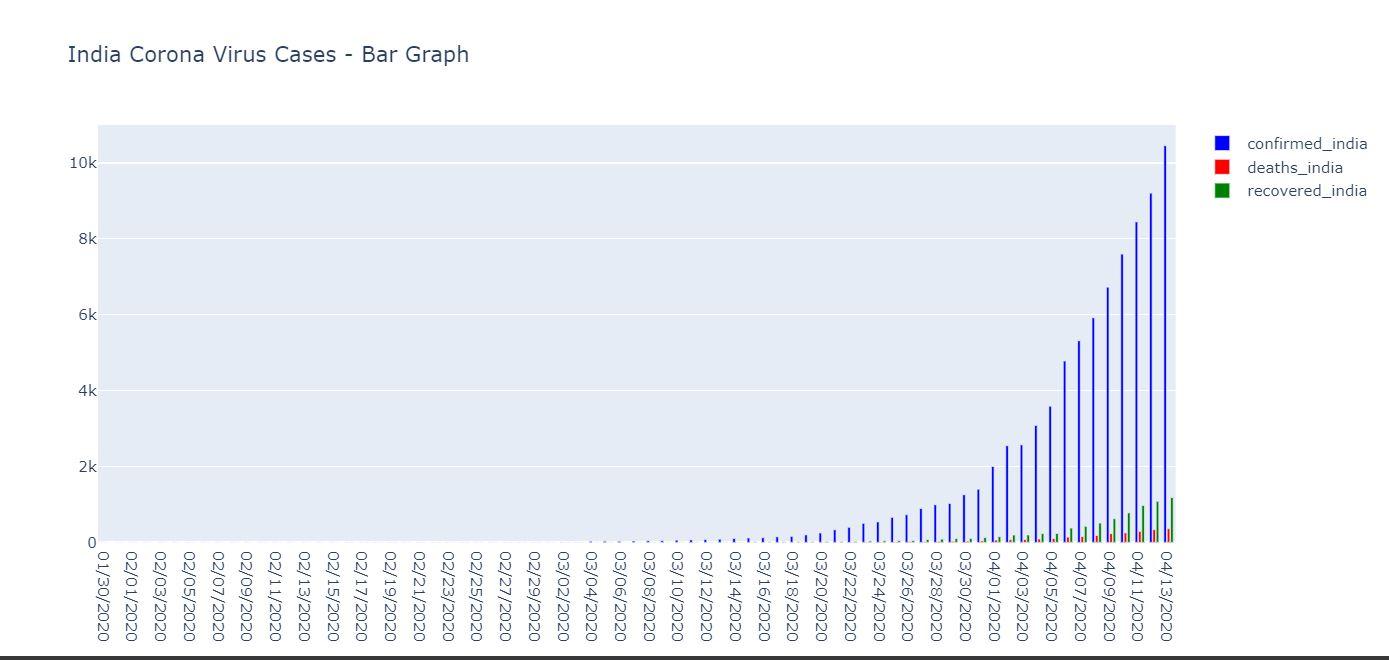
Main file in this dataset is covid\_19\_data.csv and the detailed descriptions are below.

* Sno - “Serial number”
* ObservationDate - “Date of the observation in MM/DD/YYYY”
* Province/State - “Province or state of the observation (Could be empty when missing)”
* Country/Region - “Country of observation”
* Last Update - “Time in UTC at which the row is updated for the given province or country. (Not standardized and so please clean before using it)”
* Confirmed - “Cumulative number of confirmed cases till that date”
* Deaths - “Cumulative number of deaths till that date”
* Recovered - “Cumulative number of recovered cases till that date”

**Exploratory Data Analysis**

We use standard visual data analysis tools to find visible patterns in the data and to modify the data further as per need.

*Figure 7*

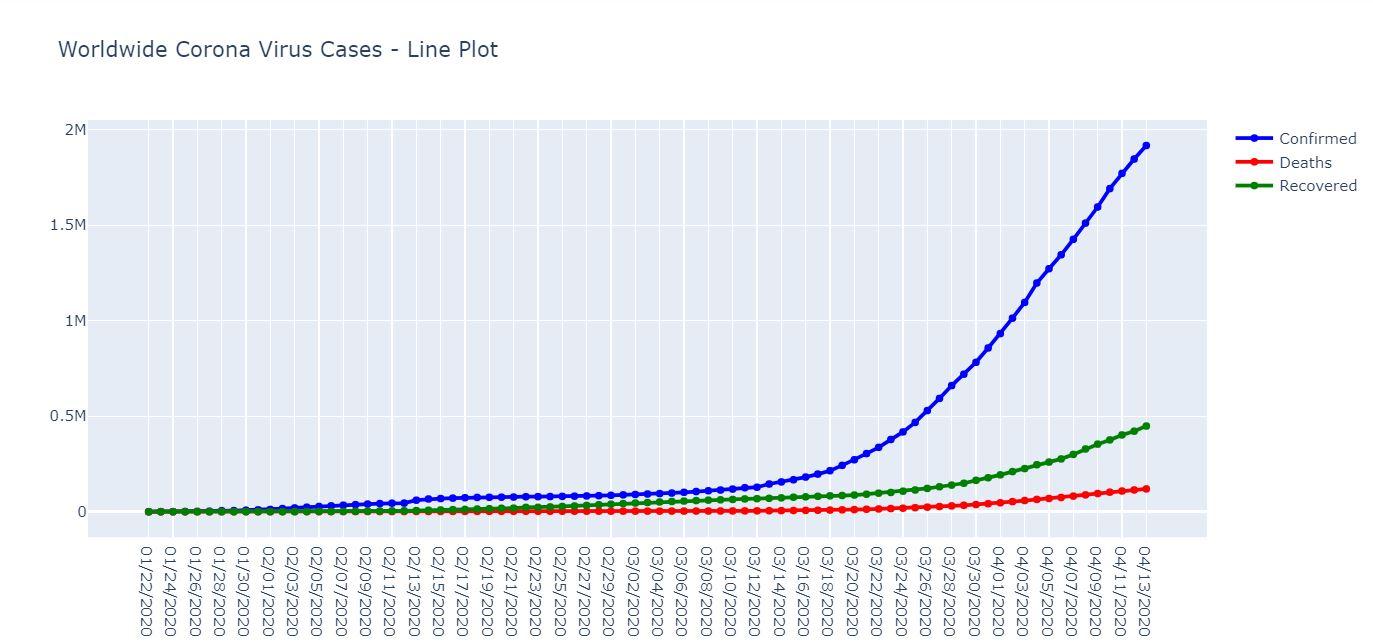


*Figure 8*

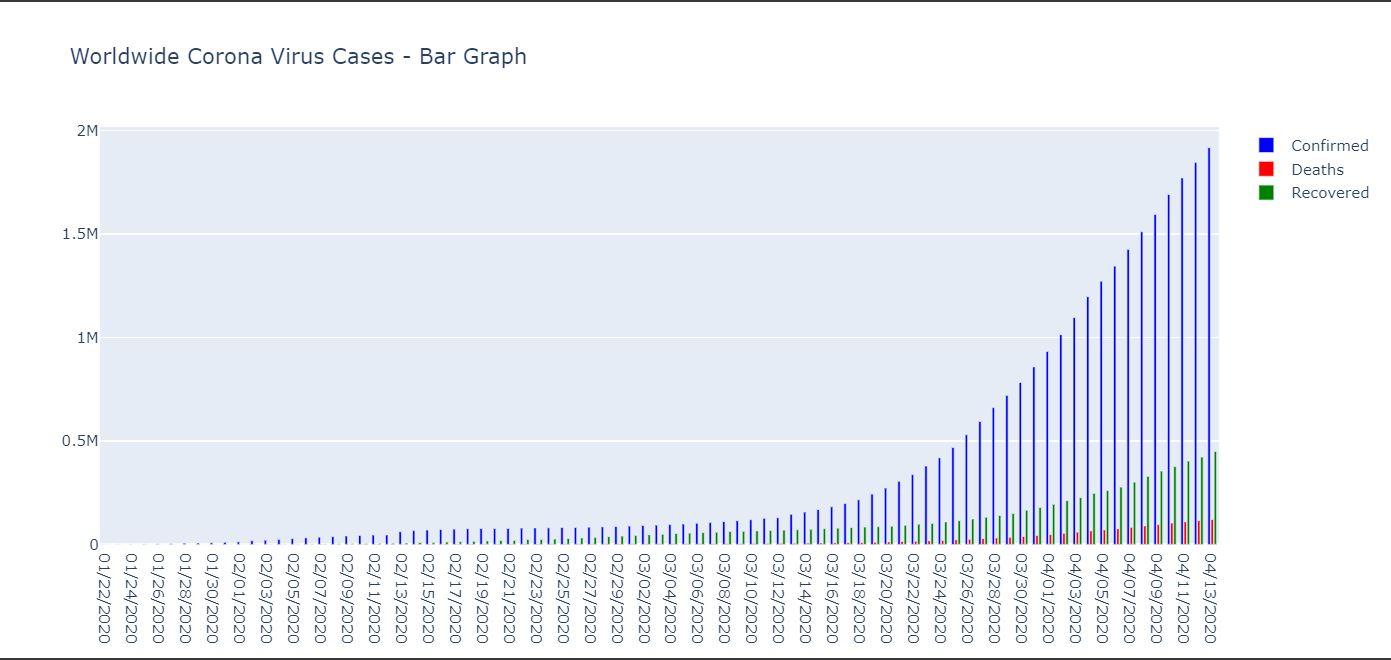
Change in Number of Confirmed, Death and Recovered Coronavirus Cases [INDIA]

Fig 7 - Scatter Plot, Fig 8 - Bar Graph

The graphs already say a lot of things. The cases visibly seem to be rising as is expected by the SIR (Susceptible, Infected, Removed) model. The SIR model however works mostly forwards where certain parameters must be known apriori and is based on pure markov models.

****

*Figure 9*



*Figure 10*

Change in Number of Confirmed, Death and Recovered Coronavirus Cases [WORLD]

Fig 9 - Scatter Plot, Fig 10 - Bar Graph

It is evident from the graphs that India is not testing enough. The change in cases is evidently stochastic which is a characteristic of selective testing.

The bar graph for confirmed, death and recovered cases seems smooth and is expected due to averaging of testing results around the globe.



*Figure 11: Heat map showing number of cases around the globe.*

### Preprocessing the data

#### Slicing and Merging

We merge the data sets to get coordinates and date of reports in a time series format.

We slice the confirmed, death and recovered cases in columns.

We correct redundancy in region names across various datasets.

#### Normalization

Min Max Normalization - We have defined a min max scalar that scales the data between 0 and 1. This scaling is very essential for deep learning models due to two famous problems - 1. Exploding gradient and

2. Vanishing gradient

**Data Generation**

We use a rolling window to generate input sequences of desired length and next day cases as outputs of the model.

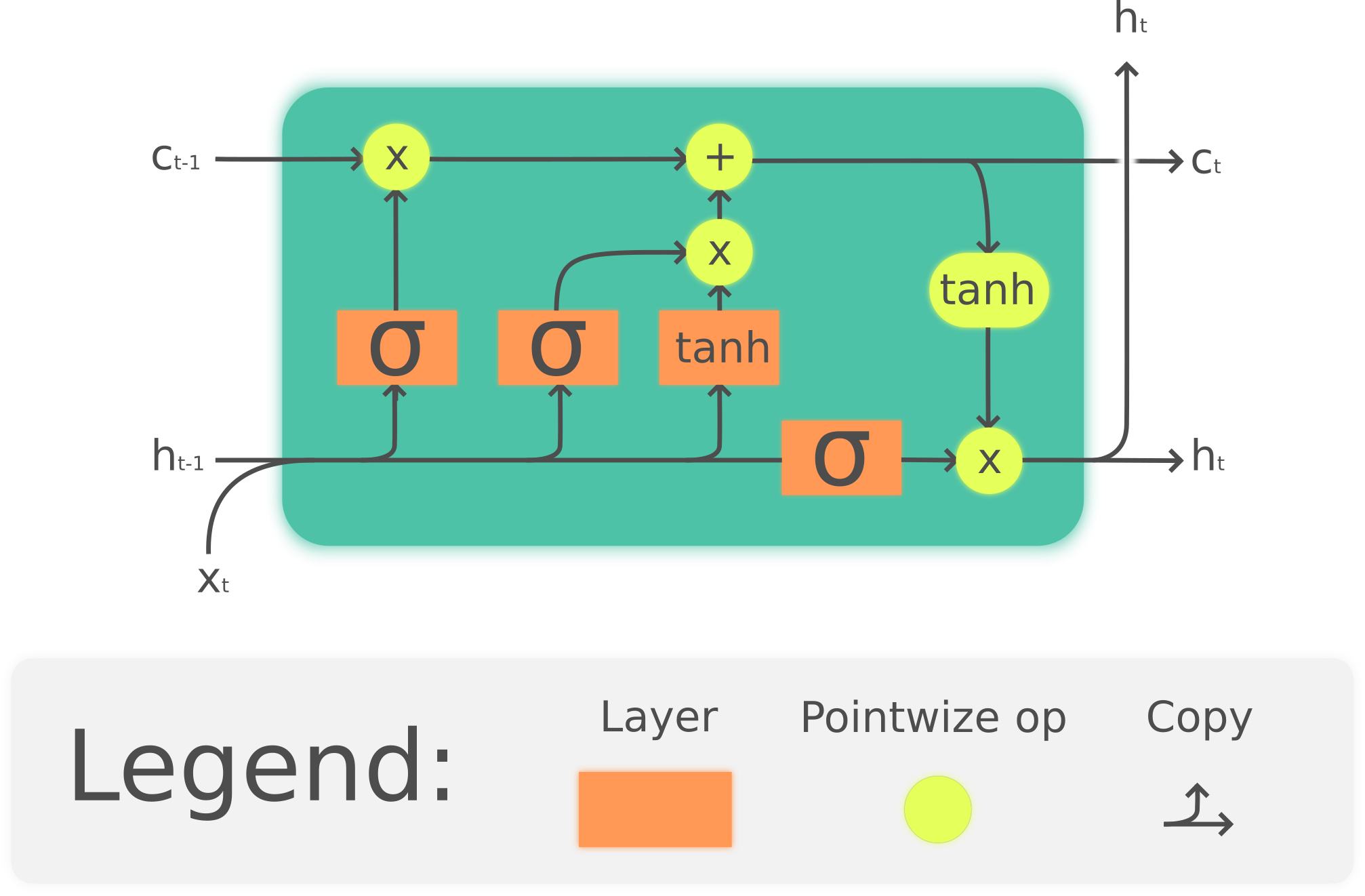
**The Model**

Time series modelling looks into the history of sequence of data and predicts what the future elements of the sequence are going to be.

Long Short Term Memory models are very powerful in modelling time series data.

An LSTM is composed of following components

1. Cell State - Stores the state of output across the entire chain. Cell state is controlled and can be modified by gates.
2. Hidden State - The output state of the cell, depends on all the gates within the cell.
3. Input Gate - Governs how much info from input should flow ahead.
4. Forget Gate - Governs if the cell state will continue to have that state or forget it.
5. Output Gate - Governs the amount of information to be passed to the next hidden state.



*Figure 12: LSTM Cell*

**Loss Metric**

MSE - Mean Squared Error

"In statistics, the mean squared error (MSE) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors — that is, the average squared difference between the estimated values and what is estimated. " [7]

**Adam Optimizer**

Adaptive Moment Estimation Optimization Algorithm

A very general optimization algorithm formed by combination of Momentum and Stochastic Gradient Descent with RMSprop optimization.

**Predictions**

We will have to inverse transform our predicted values in order to get actual predictions.

We further plot our predictions and calculate Root Mean Square Error to gauge the performance of our model.

## 

*Figure 13: Summary of our LSTM Model*

## IV. PERFORMANCE

### CLUSTERING

Find the Within-Cluster Sum of Squared Errors for multiple inputs of ‘K ‘ and plotting the distortions(WSS) against K

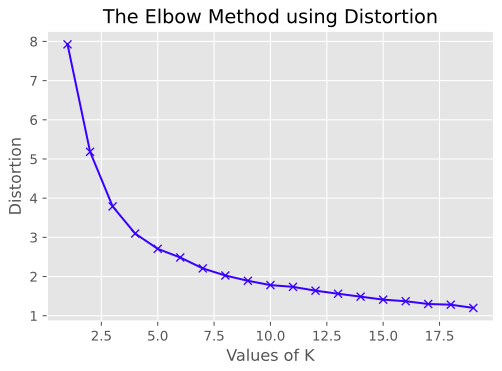


Figure 14: At value k=3 we get an elbow after which we see a linear decrease

We calculate the silhouette score for each value of ‘K’ or in other words for each iteration of the value K. We then plot the Silhouette score for each K against K.

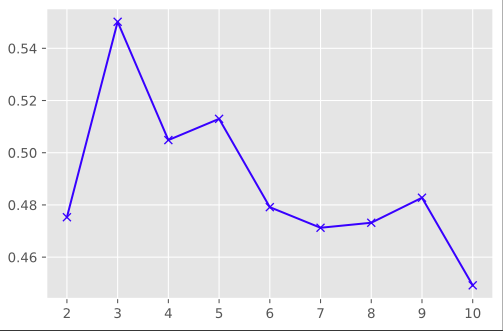
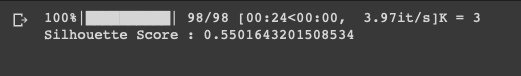


Figure 15: Global maximum gives us the optimal value of K. (K=3)

We have used Silhouette Analysis to measure the performance of our clustering Algorithms. The Silhouette score value represents the similarity of an object to its own cluster (known as cohesion), compared to other clusters (Known as separation). The value of the Silhouette score ranges from -1 to 1, wherein a higher positive value shows how well the objects are matched to their own cluster.

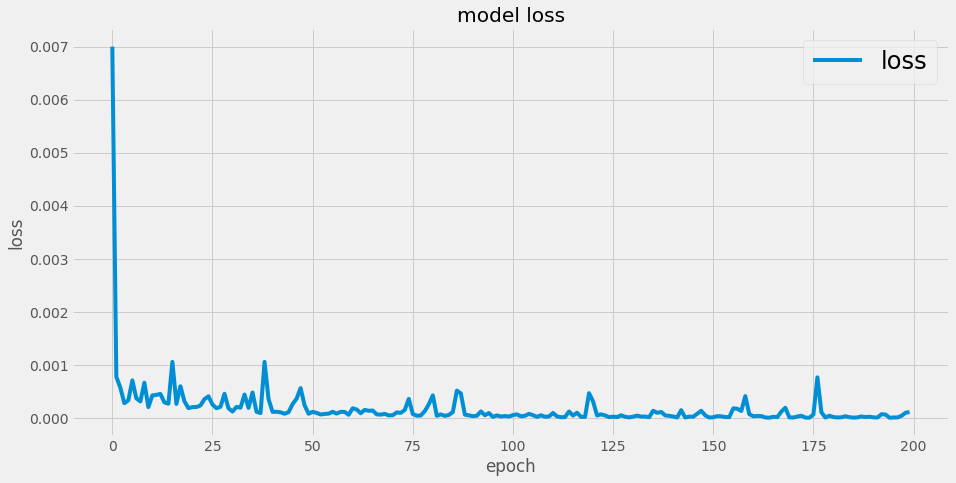
For KMeans clustering, we calculated the best possible value of K in the range of 2 to 100 for obtaining the best silhouette score. We found the best Silhouette score for K=3.



*Figure 16: Optimal value for KMeans Clustering*

For DBSCAN and HDBSCAN, the silhouette score ultimately depends on the parameters provided. We chose parameters in such a way that the clusters would be visually distinct when plotted rather than focusing on just getting a high silhouette score.

### FORECASTING



*Figure 17: Plot of loss during training*



*Figure 18: Root Mean Square Error*

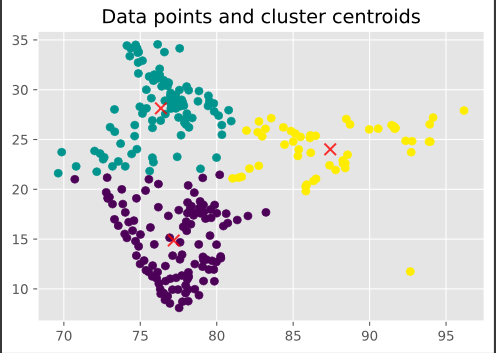
We obtain a decent RMSE value compared to the number of cases involved. This suggests that the LSTM model performed quite well and was able to capture the relationship between future and past confirmed cases.

## 

## V. RESULTS

### CLUSTERING

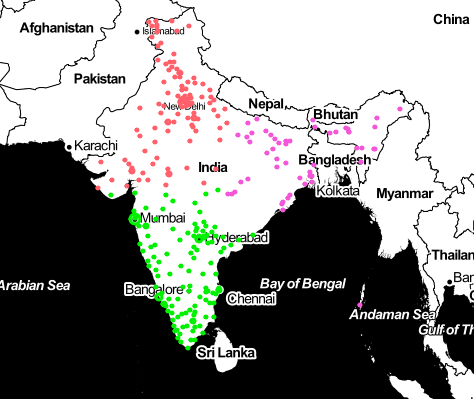
For **KMeans** clustering, we observed the best performance at K=3. Firstly, we plotted the results using a simple scatter plot and identified the respective cluster centers.



*Figure 19: Scatterplot for KMeans clustering*

We observed that even though KMeans has successfully classified the dataset into 3 distinct clusters, the centers obtained in each cluster aren’t really indicating the hotspots for the disease.

We have visualized the clusters formed on an interactive map using folium as well to view the clusters formed geographically



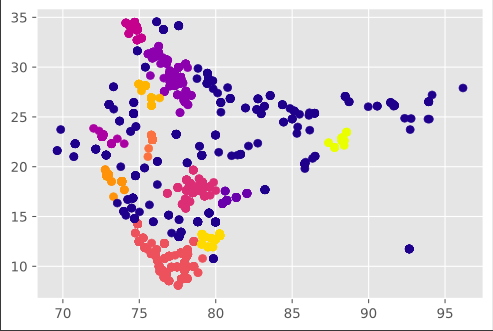
*Figure 20: KMeans visualization using folium*

**DBSCAN**

Due to the shortcomings observed in KMeans, we moved onto a density-based approach. In order to visualize distinct clusters, we chose our Minpts = 50 and eps = 1. These values were taken to not only get distinct clusters, but to also ensure that not too many clusters were formed such that some points themselves constitute a cluster. We also had to make sure that we didn’t get very less clusters to avoid the problem encountered with KMeans.

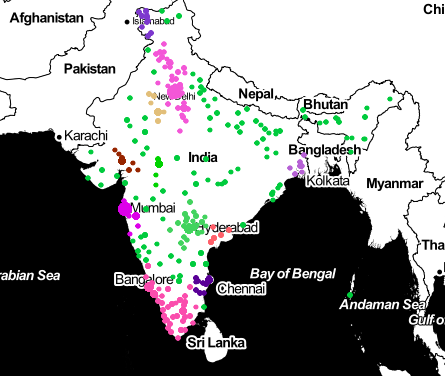


*Figure 21: Results of DBSCAN*



*Figure 22: Scatterplot for DBSCAN*

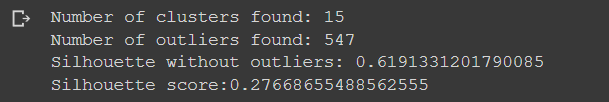
Here we observe the red, orange and purple colors are consistent to clusters at only a single place and hence can be considered as the hotspots for the disease.



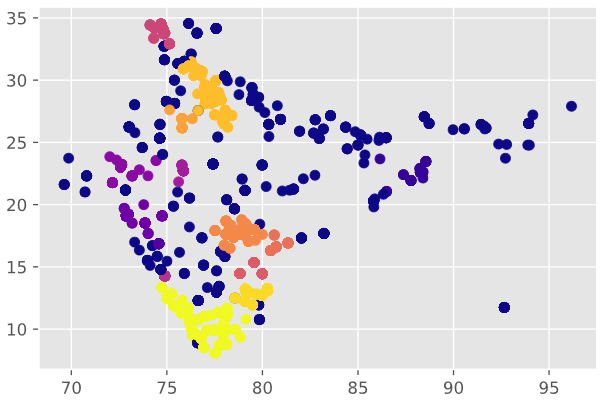
*Figure 23: DBSCAN visualization using folium*

**HDBSCAN**

In Hierarchical DBSCAN, in order to make distinct clusters and to keep the total number of clusters as low as possible to obtain an appropriate result, so that points with most number of cases don’t form a separate clusters of their own, we’ve chosen the parameters min\_sample = 50, min\_cluster\_size = 20, cluster\_selection\_epsilon = 1.

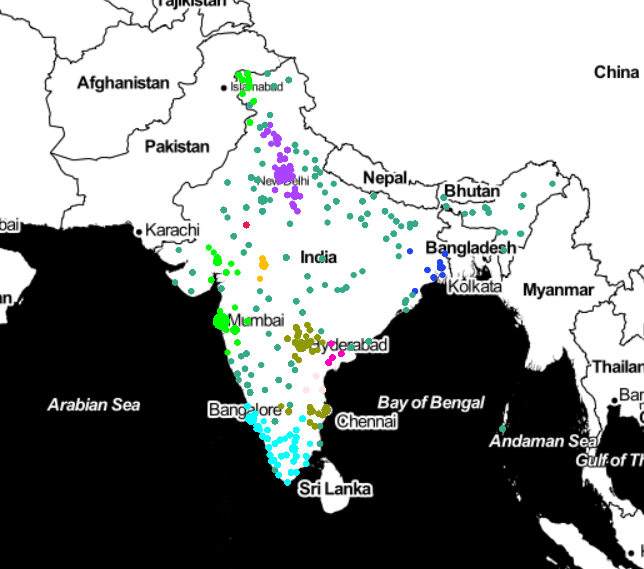


*Figure 24: Results of HDBSCAN*



*Figure 25: Scatterplot for HDBSCAN*

We can say that dark yellow, lemon yellow and orange clusters are consistent at a single place and hence can be considered as COVID-19 hotspots.



*Figure 26: HDBSCAN visualization using folium*

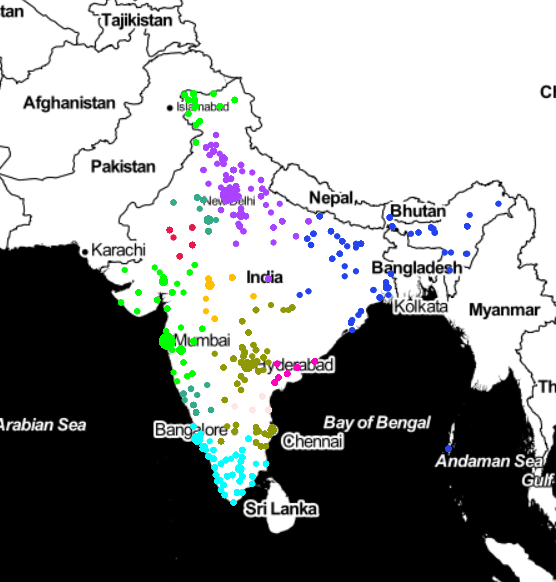
### Addressing Outliers

The main drawback of density-based clustering is its inability to cluster points that do not lie within the epsilon radius of the already formed clusters. Those points are known as outliers. In order to include those outliers in the clusters formed by clustering we have trained the model from already clustered points and hence found the value of the expected cluster for each of the un-clustered points. After combining them with the clusters, these are the final results.



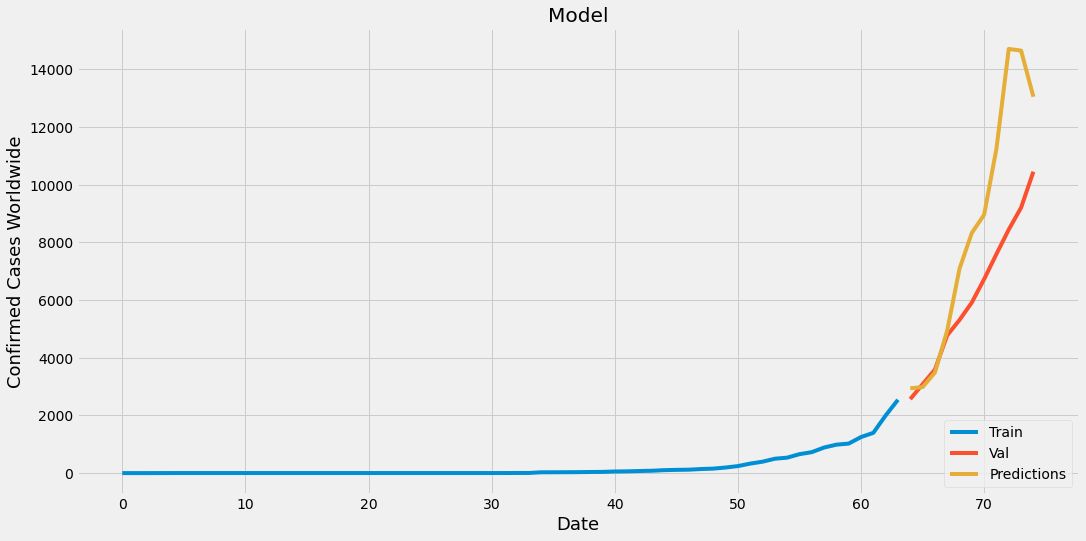
*Figure 27: Results after removing outliers*

As we can see that the number of clusters still remains the same but we can obtain a better Silhouette score after addressing the outliers.

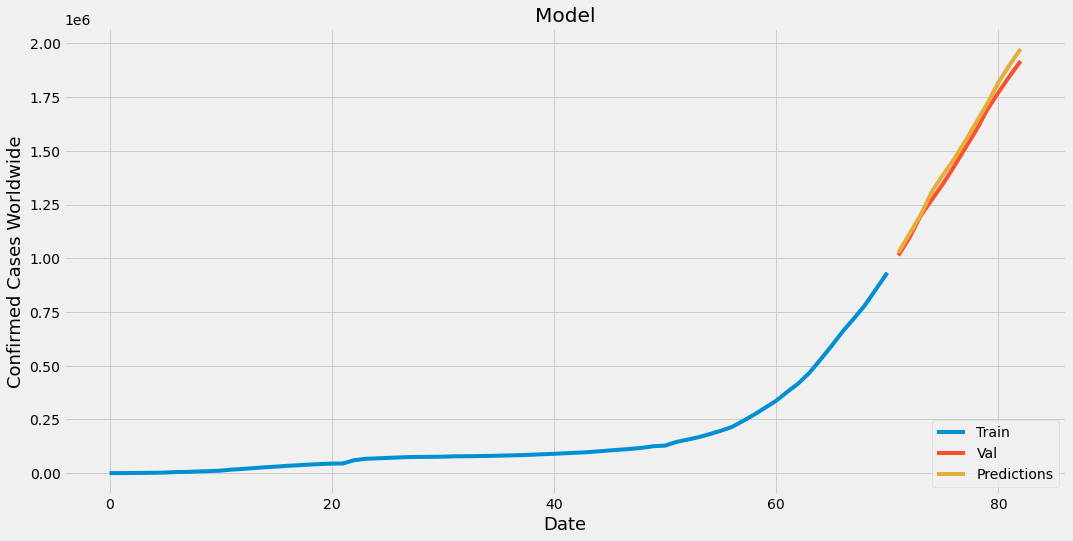


*Figure 28: Folium representation after removing outliers*

**FORECASTING**

****

*Figure 29: Predictions of Cases within India*

****

*Figure 30: Predictions of Cases Worldwide*

We see that LSTM has been able to very well predict the number of confirmed cases worldwide.

This shows effectiveness of LSTM models in ideal scenarios.

In the India curve however we see that the validation cases are a little less than the predicted cases which can be accounted to the lockdown imposed by the government. This behavior is absent in worldwide cases due to averaging of effective and ineffective containment of the virus.

The curve however hasn't saturated yet, which again points directly towards either slow testing or ineffective lockdown implementation by the community.

## VI. CONCLUSION AND FUTURE SCOPE

This was a very rewarding experience for us as we learnt various machine learning models and data manipulation techniques. Hotspots within the country were created with decent accuracy and cases were forecasted with a respectable amount of precision. However, the need of the hour is to be pragmatic and prepare for the worst. These models form the basis of our understanding of the outbreak.

We aim to keep exploring the data related to the virus and perform different analysis in various domains so as to have a comprehensive analysis of the outbreak.

**VII. REFERENCES**

[1] J. Kent, "Data Scientists Use Machine Learning to Discover COVID-19 Treatments," 2020 March 2020. [Online]

[2] S. Das, "IBM Research Releases Cloud & AI-Based Resources To Accelerate COVID-19 Drug Discovery," April 2020. [Online].

[3] P.Silitonga, "Clustering of Patient Disease Data by Using K-Means Clustering," in *International Journal of Computer Science and Information Security (IJCSIS), Vol. 15, No. 7, July 2017*, 2017.

[4] G. Nandana, S. M. and A. R. , "Hotspot Detection of Dengue Fever Outbreaks Using DBSCAN Algorithm," in *9th International Conference on Cloud Computing, Data Science & Engineering (Confluence).*, 2019.

[5] W. L. S. B. S. S. T. N. Z. G. D. Md Farhadur Rahman, "HDBSCAN: Density based Clustering over Location Based Services," 2016.

[6] L. R. A. T. C. S. Cleo Anastassopoulou, "Data-based analysis, modelling and forecasting of the COVID-19 outbreak," 2020.

[7]"Mean squared error," [Online]. Available: https://en.wikipedia.org/wiki/Mean\_squared\_error.