COVID-19 Insights and Projections

(Aditya Kandya - 11740050, Aniket Raj - 11740120, Deepak Singh - 11740290, Pramit Bhattacharyya - 11740660, Shiv Kumar - 11740950)

Upcoming Clusters

Data Collection

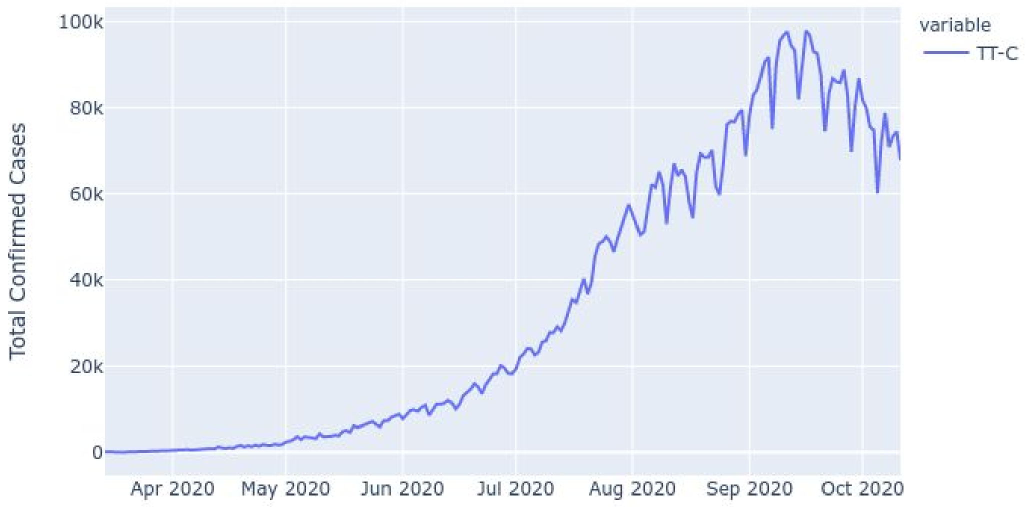
For data collection, we used the state_wise_daily CSV file for the number of confirmed, recovered and deceased cases for each state from https://api.covid19india.org/documentation/csv/. For the testing data, we used the State Level: Testing data json file from https://api.covid19india.org/. Since the files contained more data fields than required, we extracted the relevant fields from the files.

Data Cleaning

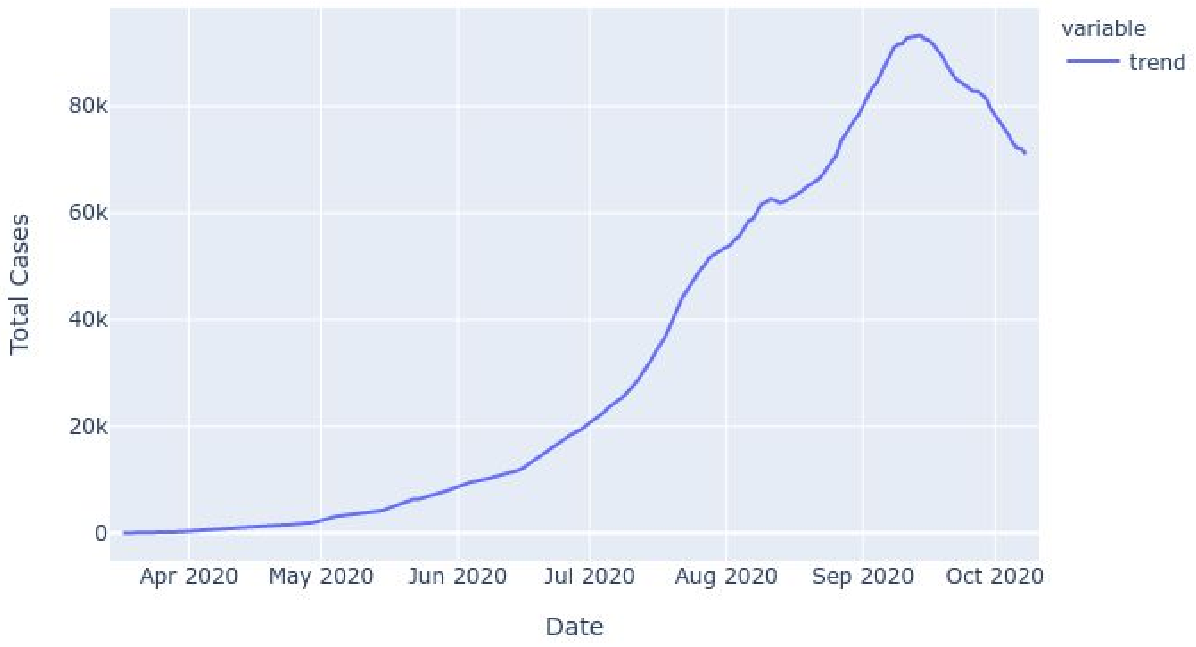
The state_wise_daily file didn't require much cleaning as the data was consistent i.e. contained data for each date starting from 14-Mar. In the State Level: Testing Data file, data for some earlier dates for some states were missing. Since the data was the number of tests performed daily, therefore we replaced the missing entries with 0, as the missing data signified no tests were done on that day. In the dataframe, we also converted the date from string to datetime objects. Dadra & Nagar Haweli and Daman & Diu didn't have separate testing data, so the testing data for both were put in Dadra & Nagar Haweli column.

Data Modelling

The Total Confirmed Cases on each day was analysed.



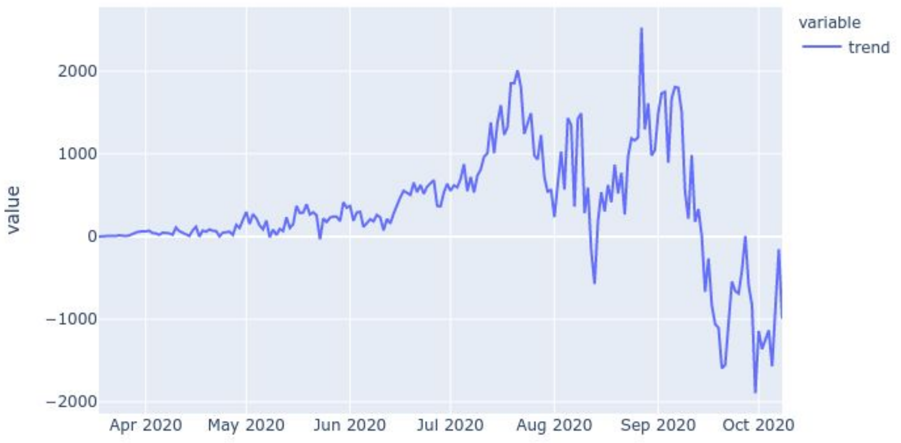
We decomposed the time series to get the *trend*, *seasonality* and *residuals* (using additive method).



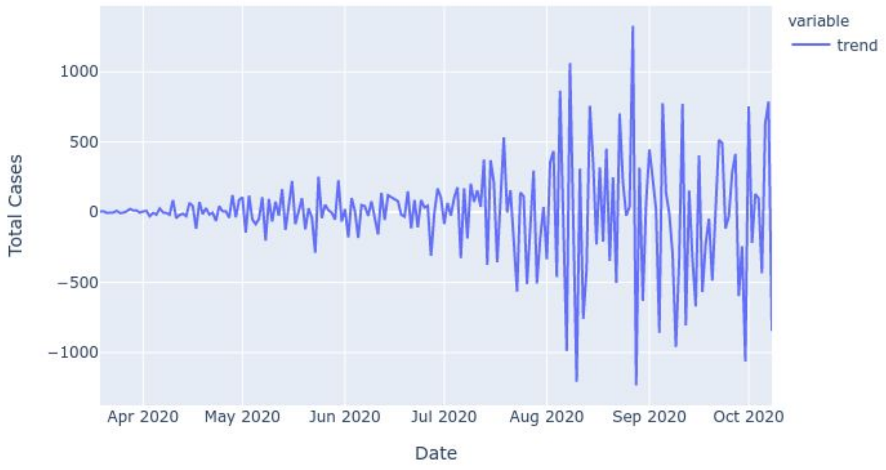
We analysed the trend. The trend was upwards till the starting days of October and downwards from there. Further, it was found out that the time series represented by the same was not stationary. Then, we took the difference of the consecutive series data and the time series so obtained is tested on by the ADFuller test for confirmation of stationarity. We continued this process until the time series obtained passes the ADFuller test and we get a stationary time series. In this case, we had to take the differences twice i.e., I=2.

The following are the graph plots of the series:

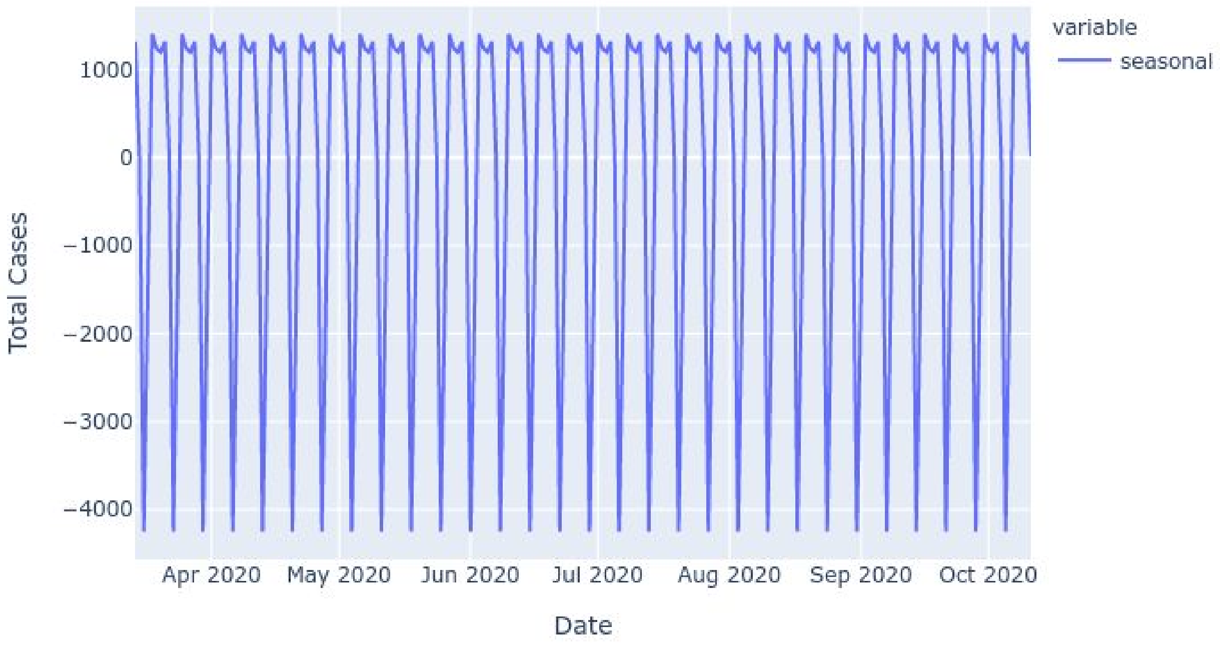
After first differentiation, I = 1.



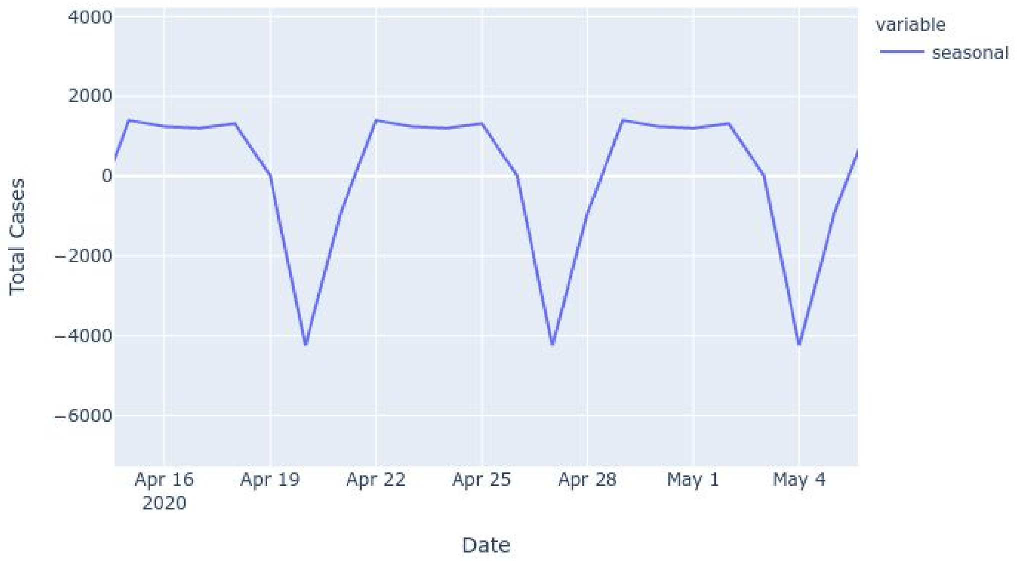
After the second differentiation, I = 2.



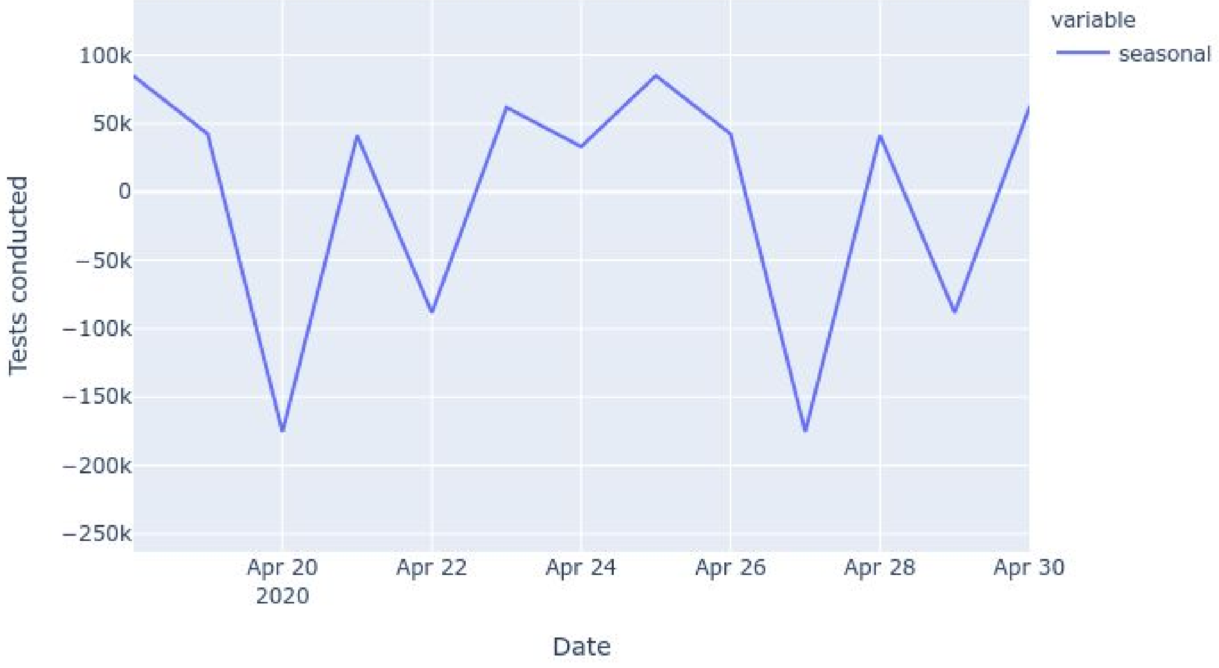
There is seasonality in the time series:



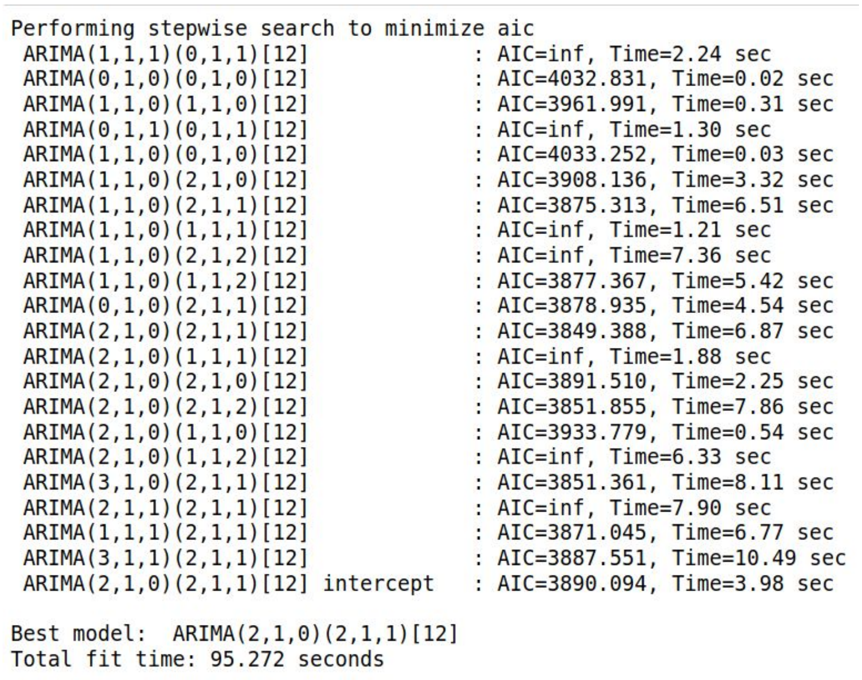
A closer look at the same reveals that the seasonality has a period of around 7 days or a week.



There is an increase for the first two days, remains the same and then decreases again in the last two days. Infact, this can be attributed to the number of tests conducted throughout the week (Number of tests conducted dipped on Sundays).



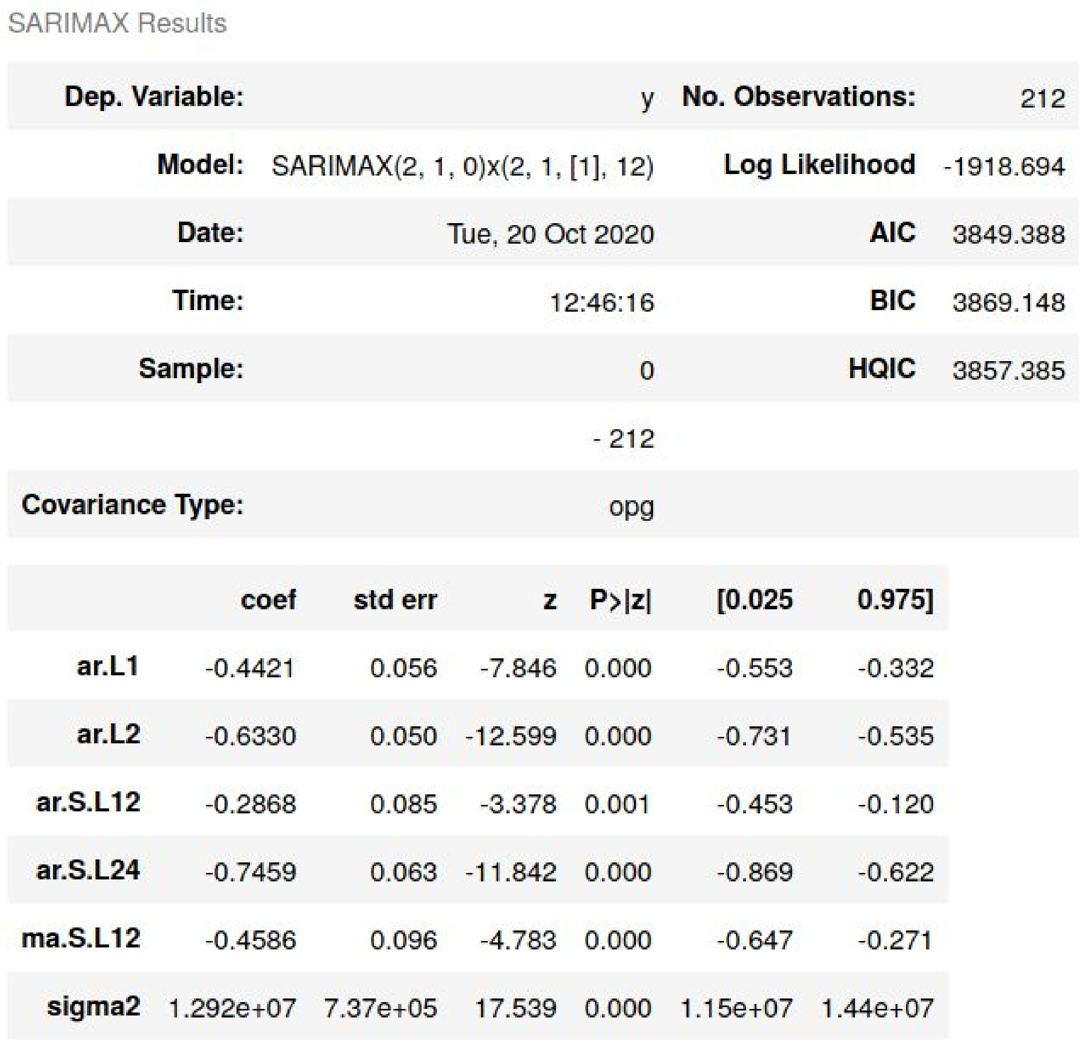
We also used SARIMAX to build a predictor model. We used an auto_arima stepwise search module to get order for ARIMA and seasonal order for SARIMAX.



```
Performing stepwise search to minimize aic
 ARIMA(1,1,1)(0,1,1)[12]             : AIC=inf, Time=2.24 sec
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=4032.831, Time=0.02 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=3961.991, Time=0.31 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=inf, Time=1.30 sec
 ARIMA(1,1,0)(0,1,0)[12]             : AIC=4033.252, Time=0.03 sec
 ARIMA(1,1,0)(2,1,0)[12]             : AIC=3908.136, Time=3.32 sec
 ARIMA(1,1,0)(2,1,1)[12]             : AIC=3875.313, Time=6.51 sec
 ARIMA(1,1,0)(1,1,1)[12]             : AIC=inf, Time=1.21 sec
 ARIMA(1,1,0)(2,1,2)[12]             : AIC=inf, Time=7.36 sec
 ARIMA(1,1,0)(1,1,2)[12]             : AIC=3877.367, Time=5.42 sec
 ARIMA(0,1,0)(2,1,1)[12]             : AIC=3878.935, Time=4.54 sec
 ARIMA(2,1,0)(2,1,1)[12]             : AIC=3849.388, Time=6.87 sec
 ARIMA(2,1,0)(1,1,1)[12]             : AIC=inf, Time=1.88 sec
 ARIMA(2,1,0)(2,1,0)[12]             : AIC=3891.510, Time=2.25 sec
 ARIMA(2,1,0)(2,1,2)[12]             : AIC=3851.855, Time=7.86 sec
 ARIMA(2,1,0)(1,1,0)[12]             : AIC=3933.779, Time=0.54 sec
 ARIMA(2,1,0)(1,1,2)[12]             : AIC=inf, Time=6.33 sec
 ARIMA(3,1,0)(2,1,1)[12]             : AIC=3851.361, Time=8.11 sec
 ARIMA(2,1,1)(2,1,1)[12]             : AIC=inf, Time=7.90 sec
 ARIMA(1,1,1)(2,1,1)[12]             : AIC=3871.045, Time=6.77 sec
 ARIMA(3,1,1)(2,1,1)[12]             : AIC=3887.551, Time=10.49 sec
 ARIMA(2,1,0)(2,1,1)[12] intercept   : AIC=3890.094, Time=3.98 sec

Best model:  ARIMA(2,1,0)(2,1,1)[12]
Total fit time: 95.272 seconds
```

The results for the SARIMAX model’s fit call:


SARIMAX Results

Dep. Variable:	y	**No. Observations:**	212
Model:	SARIMAX(2, 1, 0)x(2, 1, [1], 12)	**Log Likelihood**	-1918.694
Date:	Tue, 20 Oct 2020	**AIC**	3849.388
Time:	12:46:16	**BIC**	3869.148
Sample:	0	**HQIC**	3857.385
	- 212		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
ar.L1	-0.4421	0.056	-7.846	0.000	-0.553	-0.332
ar.L2	-0.6330	0.050	-12.599	0.000	-0.731	-0.535
ar.S.L12	-0.2868	0.085	-3.378	0.001	-0.453	-0.120
ar.S.L24	-0.7459	0.063	-11.842	0.000	-0.869	-0.622
ma.S.L12	-0.4586	0.096	-4.783	0.000	-0.647	-0.271
sigma2	1.292e+07	7.37e+05	17.539	0.000	1.15e+07	1.44e+07

Creating Hotspots:

We used two scoring functions to assess the severity of transmission of COVID-19 in a district.

S1(district,date) = Forecast(district,date)/{(Population(district) x Tests(district,date))}

S2(district,date) = Forecast(district,date)/sqrt{(Population(district) x Tests(district,date))}

where, Tests is the forecast of the number of tests conducted.

We took the mean of the tests conducted for all previous days as the forecasted value for the test to save time. For some of the districts, the mean of the tests was 0 (all previous days data was missing). To tackle this, we computed test positivity ratio (forecast/tests or forecast/sqrt(tests)) on all the districts whose test data was available and filled the mean of the test positivity ratio for all missing ones.

For a particular day, the districts are clustered on the basis of their prediction score for that day and the districts belonging to the cluster with the highest prediction scores are taken into consideration. If the number of such districts is more than threshold (already set), then, it is assumed that the prediction of new cases arising in those areas are almost evenly distributed in and around the locality. So, in such cases, we conclude that there are no hotspots. On the contrary, if the number is lower than the threshold, then we consider the districts as hotspots and list their names along with visualising them on a map.

The clustering is done using the DBSCAN function from the python inbuilt package, sklearn.cluster.`DBSCAN(eps=0.5, min_samples=5).fit(X)`.

Density-Based Spatial Clustering of Applications with Noise(DBSCAN) finds core samples of high density and expands clusters from them. This clustering works good for data which contains clusters of similar density.
Input Parameters:

A. eps: It is the maximum distance between two samples for one to be considered as in the neighborhood of the other. This is not a maximum bound on the distances of points within a cluster. This is the most important DBSCAN parameter to choose appropriately for the data set and distance function.

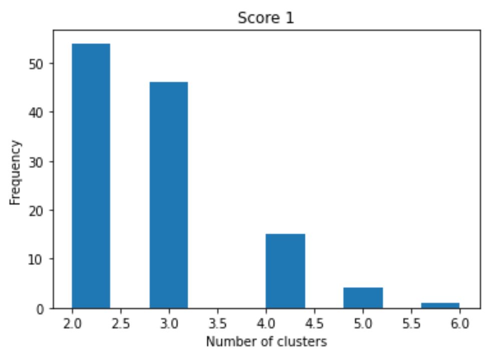
B. min_samples: The number of samples (or total weight), including the point itself, in a neighborhood for a point to be considered as a core point.

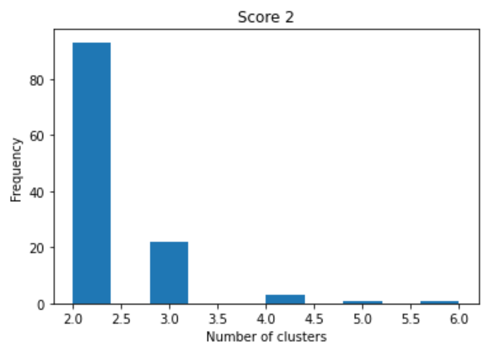
C. Here, X represents a matrix of shape (n_samples, m_features) which is used to calculate the distance between two sample points based on a given algorithm(mostly euclidean). X might also represent a pre-computed matrix of shape (n_samples, n_samples) which contains the distance between each pair of points.

Output:

An array of length equal to the number of samples is given as output. For every valid index i, array[i] either contains a non-negative number which represents the cluster the ith element belongs to, or array[i] can be -1, means that the sample element is an outlier and doesn't fit into any cluster.

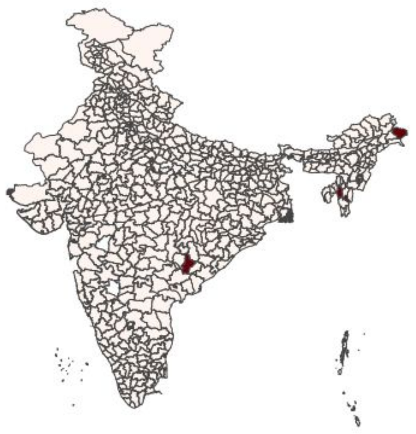
Using DBScan clustering, we obtained upcoming clusters. The histogram on count of clusters predicted is given below for both of the scoring functions.

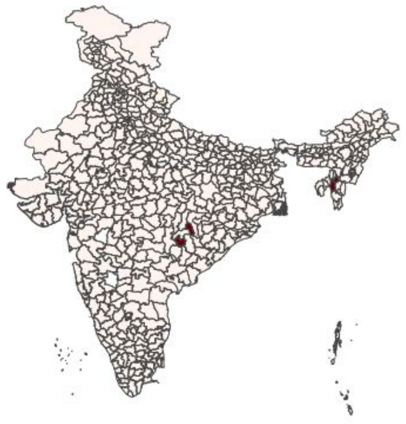




We used both the scoring function and predicted upcoming clusters for the next 120 days and took the union of outputs by both the scoring functions. The idea behind union from both the scoring functions was to not miss a legitimate cluster because that may have serious implications as the authorities may miss to impose restrictions.

We used GeoJSON from https://github.com/datameet/maps to visualise the upcoming clusters on a map.

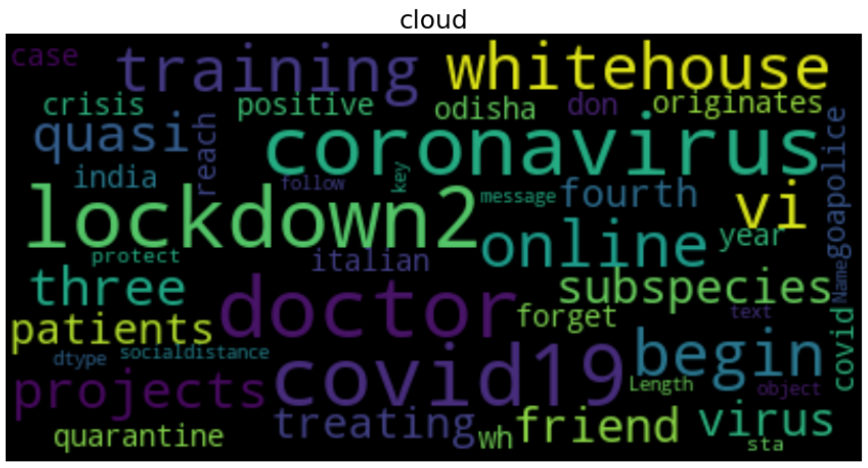




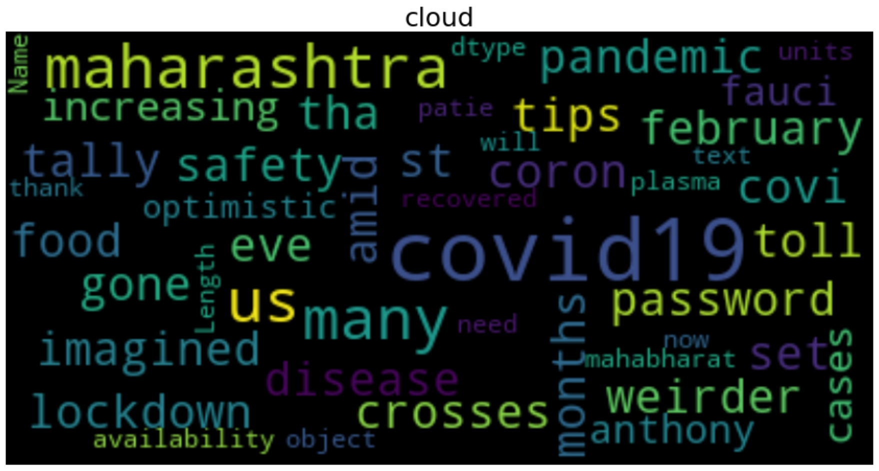
Twitter Sentiment Analysis

The dataset for the sentiment analysis is taken from here where tweets are collected using the Twitter API. In the above dataset we've extracted the tweets from India using the filter of city_name in the tweet object.

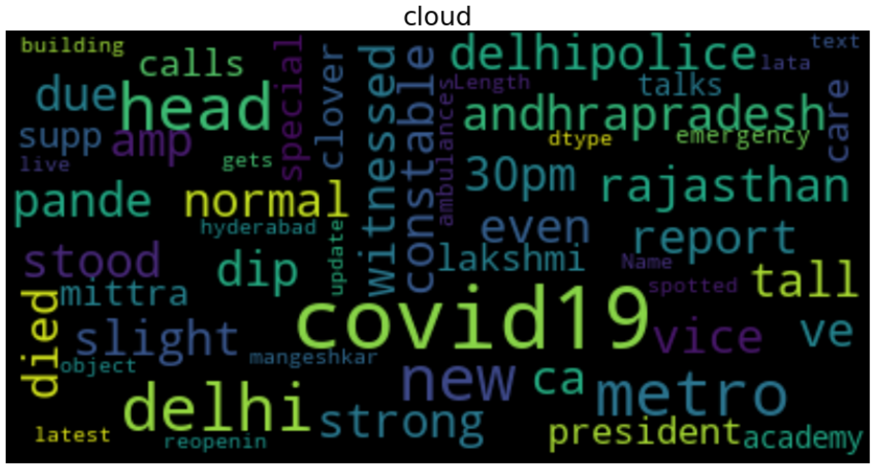
WordCloud of tweets during the pandemic months in India



April 2020

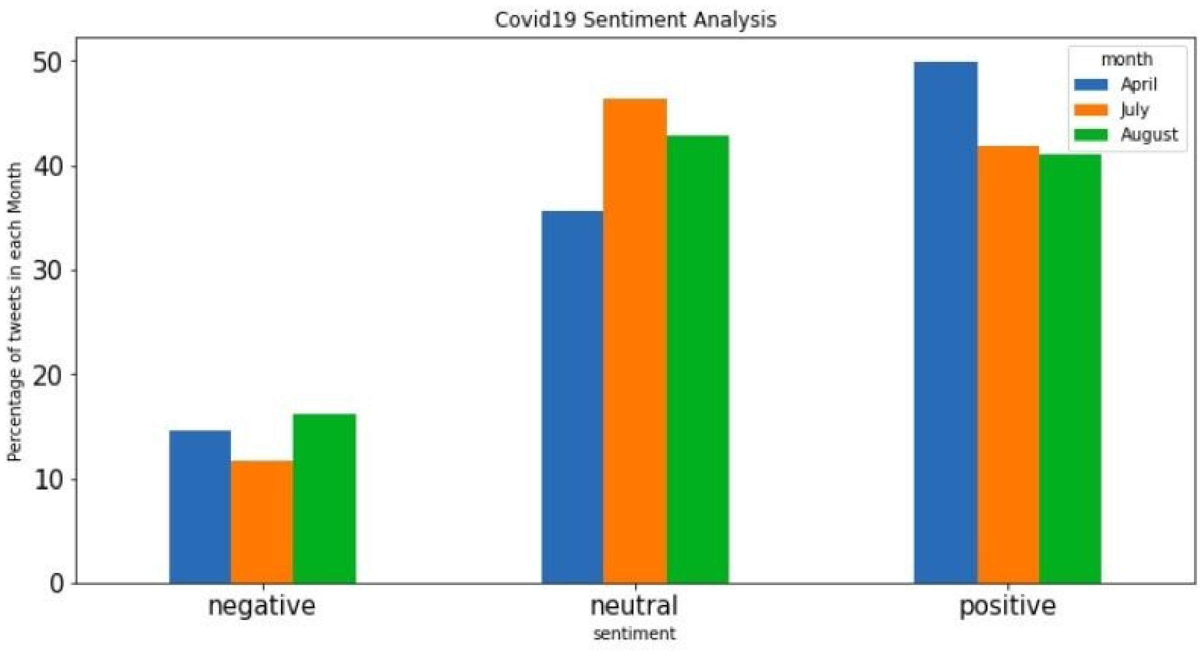


July 2020



August, 2020

Sentiment Analysis during the pandemic months :



Final Results :

- From the wordcloud visualization during the pandemic months, it is interesting to note that during the earlier months (March - May) of the pandemic, people wrote more about the technical details of the diseases concerning the virus, treatment and medical infrastructure.
- Eventually during the months of June-October, there was an apparent shift in the worldcloud where people wrote about COVID-19 in relation with other national and global activities like US Elections and Economy.

- The sentiment analysis presents mixed results in terms of binaries of positive and negative emotions during the pandemic months.

- While there's no apparent conclusion in the neutral or negative sentiments over time, there has been a steady decline in the positive standpoint of the tweets.

- It's still difficult to say if India has reached a state of pandemic fatigue because people are still engrossed in huge volumes pertaining to tweets related to COVID-19.

Member	Data Collection/Cleaning	Data Analysis & Visualisations	Data Modelling
Aditya Kandya	Upcoming clusters		Twitter Sentiment Analysis
Aniket Raj	None	Twitter Sentiment Analysis	Upcoming Clusters
Deepak Singh	None	Upcoming Clusters	Upcoming Clusters
Pramit Bhattacharyya	None	Upcoming Clusters	Upcoming Clusters
Shiv Kumar	Twitter Sentiment Analysis	Twitter Sentiment Analysis	Twitter Sentiment Analysis