

CORONAVIRUS OUTBREAK PREDICTION USING MACHINE LEARNING

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering in
COMPUTER SCIENCE AND ENGINEERING

VELLAKARA PURUSHOTHAM NIKHITHA (37110818)
TANZEEMA A (37110769)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI-
600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **VELLAKARA PURUSHOTHAM NIKHITHA (37110818)** and **TANZEEMA A(37110769)** who carried out the project entitled "**CORONAVIRUS OUTBREAK PREDICTION USING MACHINE LEARNING**" under my supervision from December 2020 to March 2021.

Internal Guide

Mrs.RAMYA G FRANKLIN MCA.,M.E.,

Head of the Department

Dr.S.VIGNESHWARI M.E.,Ph.D., Dr.L.LAKSHMANAN

M.E., Ph.D.,

Submitted for Viva voce Examination held on 10-04-2021

Internal Examiner

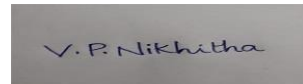
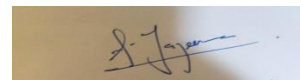
External Examiner

DECLARATION

We **VELLAKARA PURUSHOTHAM NIKHITHA(37110818)** and **TANZEEMA A(37110769)** hereby declare that the Project Report entitled “**CORONAVIRUS OUTBREAK PREDICTION USING MACHINE LEARNING**” done by us under the guidance of **Mrs.RAMYA G FRANKLIN MCA.,M.E.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE : 10-04-2021

PLACE : CHENNAI

A rectangular box containing a handwritten signature in black ink that reads "V.P. Nikhitha".A rectangular box containing a handwritten signature in blue ink, which appears to be "S. Jayaram".

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. T.Sasikala M.E., Ph.D., Dean,** School of Computing, **Dr.S.Vigneshwari M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Mrs.Ramya G Frankin MCA.,M.E.,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

On this research, predicting the coronavirus is divided into many phases, statewide analysis which covers active, total, cured and death cases, also the increase in cases on daily basis which covers India, Korea, Italy and Wuhan, confirmed, death and recovered cases. This also covers thread of novel coronavirus cases worldwide and predicts the outcome of outbreak in the forthcoming days. Google Colab and an open source platform called Prophet are used to implement this model. We use Prophet for forecasting the time series data for predicting the outbreak of virus. Google Colab provides us Jupyter notebook environment which is suitable for machine learning concepts and it is a free online cloud. Support Vector Machine, Regression and Data Visualization are the Machine learning concepts used for better outcome of the model. This model is all the source analysis of spread of virus with total, new, active cases with prediction of future outbreak and weekly study epidemic.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	i
	LIST OF TABLES	ii
	LIST OF FIGURES	iv
	LIST OF ABBREVIATIONS	v
1.	INTRODUCTION	1
	1.1 BACKGRO UND	1
	1.2 SUPPORT VECTOR REGRESSION	3
	1.3 POLYNOMIAL REGRESSION	5
2.	AIM AND SCOPE OF THE PROJECT	7
	2.1 AIM AND SCOPE	7
	2.2 SOURCE FROM DATA	8
	2.3 EPIDEMIC ANALYSIS	8
	2.4 ALGORITHMS	9
	2.5 SURVEY	10
3.	SYSTEM DESIGN AND METHODOLOGY	11
	3.1 HOW COVID ANALYSIS AND PREDICTION WORKS	11
	3.2 STUDY DESIGN AND DATA DESIGN	12
	3.3 STATISTICAL MODEL FOR PREDICTIONS	12
	3.4 PARAMETER CHOICES FOR SHORT TERM FORECASTS	19
	3.5 PARAMETER CHOICES FOR LONG TERM FORECASTS	20
4.	RESULTS AND DISCUSSION	22
	4.1 DECISION BOUNDARIES	22
	4.2 INTERPRET ABILITY	22
5.	CONCLUSION AND FUTURE WORK	24

REFERENCES APPENDIX	25
A.SCREENSHOTS	26
B.SOURCE CODE	26
	52

LIST OF FIGURES

Figure No	Figure Name	Page No.
1.2	Scikit-learn	4
2.5	Algorithms Process	10
3.4	Proposed Model	20
4.2	Architecture of model	23

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
AI	Artificial Intelligence
SVM	Support Vector Machine
SAS	Statistical Analysis System
IOT	Internet of Things

CHAPTER 1

INTRODUCTI

ON

1.1 BACKGROUND:

In December of 2019, a soaring number of cases were reported in Wuhan, China. It is soon determined as novel corona virus, referred to as COVID-19. On 11 March 2020, the (WHO) World Health Organization recognized COVID-19 as a global pandemic with significantly high infection and mortality rates compared with its predecessors, including SARS and MERS. As of March 24, 2020, the virus has spread to more than 170 countries, with more than 422, 613 confirmed cases and 18,891 death tolls. The initial reports indicated that the death rate varies among countries thanks to differences in demography, age distribution, and health infrastructure. An overall 2.3% mortality rate among COVID-19 patients was reported in china. However, a significantly higher mortality rate. This week CDC (Centers for Disease Control and Prevention) received forecasts of national COVID-19 deaths over the next 4 weeks from 34 modeling groups. Those forecasts predict:

The number of new COVID-19 deaths reported each week, which indicates how reported deaths are likely to increase or decrease in the coming weeks. The total number of corona virus deaths reported by the end of each week which helps us understands the likely overall impact of the pandemic in the coming weeks.

The 34 modeling groups, 31 provided forecasts of both new and total deaths, one provided forecasts of latest deaths only, and two provided forecasts of total deaths only. This week's national ensemble forecast predicts that weekly reports of new corona virus deaths may decrease over the next 4 weeks, with 4,500 to 10,600 new deaths reported during the week ending August 29. The ensemble forecast predicts that 175,000 to 190,000 total corona virus deaths are going to be reported by August 29. State-level forecasts ensemble predict that the number of reported new deaths per week may increase over the next four weeks in Hawaii and Puerto Rico and may decrease in Florida, Mississippi, New Mexico , the Northern Mariana Islands , Ohio, Texas, Vermont, and therefore the Virgin Islands .

Corona virus spread has conducted the society under the edge of loss in sociallives, and many troubles due to it.

- Support Vector Regression(SVR)
- Polynomial Regression(PR)

The prediction of outbreak of coronavirus is necessary in order to take safety measures in this pandemic and keep the increasing active cases in control. Not only the person's health is degraded, increasing cases makes it hard to control and treat each individual person as the government, scientists and officials are not still keen on studying the virus and producing the vaccine.

This model effectively forecasts the outbreak in future and by analyzing the cases we can either take safety measures or go under lockdown as the initial cases were very hazardous. Many scientists and universities are researching the traits of virus with their experiments.

The coronavirus outburst took place in Wuhan (China) in December 2019 and how has become an epidemic. When a student arrived in Kerala from Wuhan, India reported its first coronavirus case on January 30, 2020. The virus is a chain reaction. It should be dealt carefully if not it won't be controllable later. So, at the crucial times like this it is totally necessary to predict the future outbreak of the virus so that the next time the pandemic becomes dangerous we could prevent and control the forthcoming damage and save lives. The normal models and statistical models are receiving more attention in the research of coronavirus. The long term prediction models face the uncertainty and the required and essential data is hard to find for long term prediction. Existing models need to be upgraded.

The existing models show the technology use in studying and analyzing and predicting coronavirus. Many technologies like Block chain, Internet of things, big data, Artificial Intelligence and fifth generation are part of the research and study of coronavirus.

The machine learning algorithms play a huge role in prediction or forecasting the future outcomes or analyzing the things very accurately and efficiently. In this model, we use the machine learning algorithms like Support Vector Machine, Regression and Data Visualization. The histogram plays a huge part in pictorial representation of the model.

1.2 SUPPORT VECTOR REGRESSION (SVR):

In Support Vector Regression, Regression method can be used so that it maintains all the main features that characterize the algorithm (maximal margin). The same principles for classification is used by Support Vector Regression(SVR) and SVM, with only a few minor differences. First of all, since output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In approximation to the SVM in that case of regression a margin of tolerance called epsilon is set which would have already requested from the problem. But besides this fact, there's also a more complicated reason that the algorithm is more complicated therefore to be taken into consideration. However, the main idea is always the same: such as to minimize error, individualizing the hyper plane which maximizes the margin, keeping in mind that part of the error is tolerated. The predictive analysis data-classification algorithm that assigns new data elements to at least one of labeled categories may be a SVM(Support Vector Machine). Another version of the SVM algorithm, multiclass SVM, augments SVM to be used as classifier on a dataset that contains more than one class (grouping or category)

A supervised machine learning algorithm which can be used for regression problems is Support Vector Machine. it uses as a technique called the kernel trick to transform your data, based on the transformations it finds an optimal boundary between the possible outputs.

SVM take cares of outliers better .SVM outperforms well when there are large features and less training data.

HYPERPLANE: are often used as a choice plane or space which is split between a group of objects having different classes.

SUPPORT VECTORS: Data points that are closest to the hyper plane are called support vectors. This separates the line that will be defined with the help of these data points.

MARGIN: It may be defined as the gap between the two lines on the closest data points of different classes. It can be calculated as the perpendicular distance between

from the line to the support vectors. This large margin is considered as good margin and small margin is considered as a bad margin.

MAIN GOAL OF SVM: First SVM will generate hyper planes iteratively that segregates the classes in the best way. Then, secondly it will choose the hyper plane that separates the classes correctly.

TYPES OF KERNELS: 1) Linear kernel-Formula for the linear kernel is as below

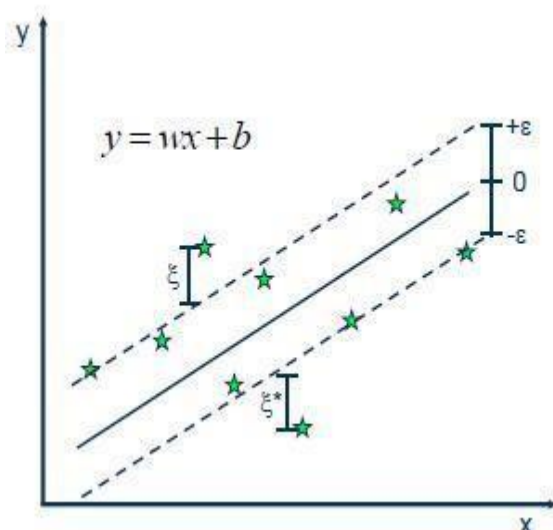
$$K(x_1, x_2) = \text{sum}(x_1 * x_2)$$

2) Polynomial kernel-Formula for this is as below

$$K(x_1, x_2) = 1 + \text{sum}(x_1 * x_2)^d$$

The advantages of support vector machines are:

- It is effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- They use a subset of training points in the decision function (called support vectors), which is also a memory efficient.
- Versatile: These different Kernel functions can be specified for the decision function. Common kernels are provided, but it's also possible to specify custom kernels.



• Minimize:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*)$$

• Constraints:

$$y_i - wx_i - b \leq \varepsilon + \xi_i$$

$$wx_i + b - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

1.3 Polynomial Regression (PR):

The relationship between the independent variable x and the dependent variable y is modelled as an n th degree polynomial in x using polynomial regression which is a form of regression analysis in statistics. The value of x and the corresponding conditional mean of y , denoted $E(y | x)$ is a non linear relationship befitted by Polynomial regression. As a statistical estimation problem polynomial regression is linear, in the sense that the regression function $E(y | x)$ is linear in the unknown parameters that are estimated from the data although it fits a nonlinear model to data. Polynomial regression is considered to be a special case of multiple linear regression for this reason.

The higher-degree terms are the explanatory variables that result from the expansion of baseline variables. In classification settings such variables are used.

To model the expected value of a dependent variable y in terms of the value of an independent variable (or vector of independent variables) x is the goal of regression analysis. In simple linear regression, the model is used, where ϵ is an unobserved random error with mean zero conditioned on a scalar variable x . For each unit increase in the value of x , the conditional expectation of y increases by β_1 units in this model. The model takes the available data which includes latitude, longitude, days, cases, deaths, positive outcomes from the datasets and validates it by training the model, then the hyper parameters are applied with data visualization, regression and support vector machine which by using prophet predicts the outbreak of cases that are confirmed, death and recovered.

To find the correlation between each and every features heat map is used. Heat map uses the correlation matrix. We visualize the results in the form of graphs and histogram.

The system architecture is shown here which takes us through the steps through which we predict the coronavirus outbreak.

The datasets are studied and examined and are used to analyze various modes of virus outburst and predict the future view of outcome.

Such a linear relationship may not hold in many settings. In terms of the temperature, if we are modeling the yield of a chemical synthesis at which the synthesis takes place, we may find that the yield improves by increasing amounts for each unit increase in temperature for example. We might propose a quadratic model of the form in this case. When the temperature is increased from x to $x + 1$ units, the expected yield changes by (This can be seen by replacing x in this equation with $x+1$ and subtracting the equation in x from the equation in $x+1$) in this model. For infinitesimal changes in x , the effect on y is given by the total derivative with respect to x .

This model is comparative analysis of cases in different countries and analysis of active cases and total cases in Indian states. The cured percentage and death percentage in the Indian states is analyzed. The total cases and new cases are shown for India, Italy, South Korea and Wuhan. The worldwide novel coronavirus cases are analyzed for recovered, death and confirmed and we predict the future outcome of outbreak. During this analysis and prediction we found that India is doing better so far in controlling the outbreak compared to other countries.

CHAPTER 2

AIM AND SCOPE OF THE PROJECT

2.1 AIM AND SCOPE:

Since the last decade, digital technologies are playing critical roles in major health sector problems including disease prevention, this worldwide health emergency also seeking technological support to tackle COVID-2019. In a paper, authors highlighted the possible applications of trending digital technologies like internet of things (IoT), big-data analytics, AI (AI), deep learning and blockchain technology to develop strategies for monitoring, detection and prevention of epidemic; and also to spot the impact of the epidemic to the healthcare sector. In a research work proposed by Benvenuto et al., authors proposed an autoregressive integrated moving average (ARIMA) model to predict the spread of COVID-2019. In this paper, the author forecasted the varied parameters for subsequent 2 days supported the study about the prevalence and incidence of the COVID-2019. The correlogram and ARIMA forecast graph for the epidemic incidence and prevalence is also demonstrated in this research. Deb et al. proposed a statistic method to research incidence pattern and therefore the estimated reproduction number of COVID-19 outbreak. They performed statistical analysis to explore the trends of the outbreak to spotlight this epidemiological stage of a neighborhood in order that various policies are often identified to deal with COVID-19 pandemic in different countries cite¹¹. As per this situation, it's essential to know the first spread patterns of the infection to plan and control the effective safety measures. In this direction, Kucharski et al. proposed a scientific model of critical SARS-CoV2 transmission by using different datasets to study the COVID-19 outbreak inside and outside Wuhan. With this, they explored the possible spread of disease outbreak outside Wuhan [Recently there are several studies conducted on the epidemiological outbreak of COVID-19 using exploratory data analysis (EDA) based on various available datasets. These studies mainly focus on the occurrence of confirmed, death and the recovered cases in Wuhan and those of the world to understand

the suspected threats and subsequent planning of containment activities. Laureate, in their research work raised the issue of the criticality of the incubation period for COVID-

19. The research highlighted that the location-specific rate of disease spread follows either steady or explosive power-law growth with different scaling exponents. With this

understanding, the authors analyzed the impact of lockdown in various parts of the earth . Based on the above literature, it's evident that sufficient work is out there on exploratory data analysis to know the prevailing trend of the epidemic but still there's a lot of scopes to develop and test efficient machine learning based prediction models in order that proactive strategies might be identified to cater the immediate need.

2.2 SOURCE FROM DATA:

The growing day to day prevalence data of COVID-2019 from January 31, 2020, to MARCH 23, 2020, were retrieved from the official repository of Johns Hopkins University. The dataset consists of daily case reports and daily statistic summary tables. In the present study, we've taken time-series summary tables in CSV format having three tables for confirmed, death and recovered cases of COVID-2019 with six attributes

i.e. province/state, country/region, last update, confirmed, death and recovered cases, where the update frequency of the dataset is once during each day . If the COVID-19 confirmed, recovered, and death cases distribution across the planet since the time data was recorded. It is easy to watch the exponential growth of the spread which must be controlled everywhere the planet .

2.3 EPIDEMIC ANALYSIS:

The COVID-19 spread has brought the planet under the brink of loss of human lives thanks to which it's of utmost importance to research the transmission growth at the earliest and forecast the forthcoming possibilities of the transmission. With this objective, state-of-the-art mathematical models are adopted supported machine learning like support vector regression (SVR) and polynomial regression (PR), and deep learning regression models sort of a standard deep neural network (DNN) and recurrent neural networks (RNN) using long STM (LSTM) cells. To predict the total number of confirmed, recovered, and the death cases worldwide Machine learning and deep learning approaches are used using python libraries. The prediction will allow undertaking the specified decisions supported transmission growth like increasing the lockdown period, executing the sanitation procedure, providing the everyday resources,

etc.

2.4 ALGORITHMS:

The regression approaches for epidemic analysis are trained and tested on realtime data using the number of confirmed, recovered, and death cases as the label for the corresponding day. With extensive experiments, machine learning approaches are implemented with the polynomial kernel of degree 6 and other coefficient values as $\gamma=0.01$, $\epsilon=1$, and $C=0.1$. The standard DNN consists of a dense input layer with 128 neurons, three hidden dense layers with 256 neurons and output layer is consists of a single neuron whereas the RNN, consists of three stacks of LSTM layers having 64 neurons combined with 10% dropout to avoid the overfitting problem and final output layer with a single neuron. The mean squared error (MSE) is the most widely used objective function and root mean square error (RMSE) as a metric function for evaluating the regression models. The MSE loss can be computed by using equation 1. $L(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^N (y - \hat{y})^2$ (1) where y indicates the original value, $\hat{y}$ indicates the predicted value, and N is the number of samples predicted. Due to the limited availability of the data, the trained models are validated against the training data and then utilized to forecast the number of confirmed, recovered, and death cases for the next 10 days. Results and discussion The discussed machine learning and deep learning approaches output the possible number of cases for the next 10 days across the world. Figure 6 illustrates the predicted trend of the COVID-19 using SVR, PR, DNN, and LSTM with worldwide data. Table 1 indicates the RMSE score of the approaches computed against the available number of COVID-19 cases. It is also observed that training of LSTM model is heavily dependent on the deviation in the values, with the fact that larger the deviation more the time it takes to train. Hence, the number of cases were scaled using min-max scaler to fit the LSTM model and later the predicted cases were re-scaled to the original range using invert.

2.5 SURVEY:

The world is under the grasp of SARS-CoV2 (COVID-19) virus. Early prediction of the transmission can help to take necessary actions. This article proposed to utilize the machine learning and deep learning models for epidemic analysis using data from Johns Hopkins dashboard. The results show that polynomial regression (PR) yielded a minimum root mean square error. It is made available under a CC-BY-NC-ND. It would lead to huge loss of lives as it presents the exponential growth of the transmission worldwide if the spread follows the predicted trend of the PR model. As observed from China, this growth of the COVID-19 can be reduced and quenched by reducing the number of susceptible individuals from the infected individuals. This is achievable by becoming unsocial and following the lockdown initiative with discipline. The study can further be extended to utilize other machine learning and deep learning models.

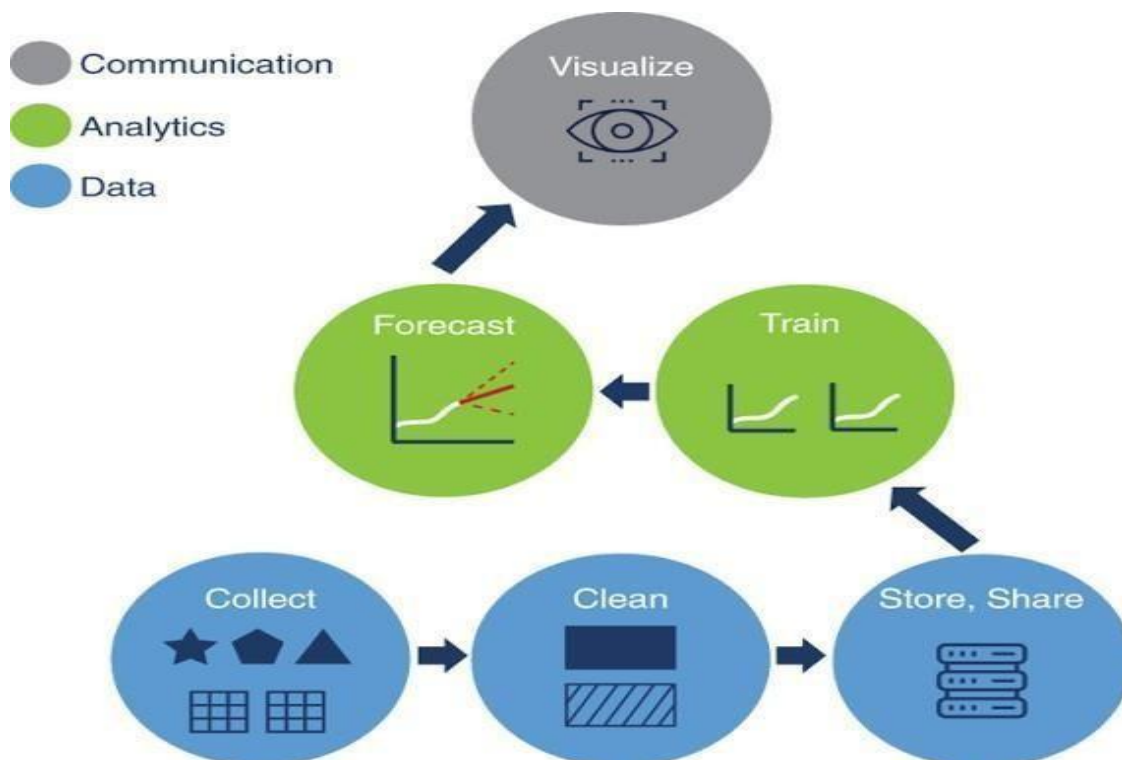


Fig-2.5 Process

CHAPTER 3

SYSTEM DESIGN AND METHODOLOGY

3.1 HOW COVID ANALYSIS AND PREDICTION WORK

Using GA, PSO, and GWO, the parameters of several simple mathematical models (i.e., logistic, linear, logarithmic, quadratic, cubic, compound, power and exponential) were fitted. On training for 30 days, the logistic model outperformed other methods and showed promising results. Extrapolation of the prediction beyond the first observation range of 30-days shouldn't be expected to be realistic considering the new statistics. For the five countries, the fitted models generally showed low accuracy and also weak generalization ability. The model was insufficient for extrapolation, as expected although the prediction for China was promising. The computational cost for GWO was reported as satisfactory and in turn the logistic GWO outperformed the PSO and GA. The logistic model fitted with GWO was used for comparative analysis consequently for further assessment of the ML models. Two scenarios were proposed in the next step, for introducing the machine learning methods for time-series prediction. From previous days Scenario 1 considered four data samples from the progress of the infection, as reported in table 3. The sampling for processing was done weekly for scenario 1. However, scenario 2 was dedicated to daily sampling for all previous consecutive days. The scope of this study was expanded by providing these two scenarios. Training and test results for the 2 machine learning models (MLP and ANFIS) were considered for the 2 scenarios. A detailed investigation was also administered to explore the foremost suitable number of neurons. Throughout the study ,for the MLP, the performances of using 8, 12 and 16 neurons were analyzed. For the ANFIS, the membership function (MF) sorts of Tri, Trap, and Gauss were analyzed throughout the study. The four counties of Italy, Wuhan, South Korea and India were considered. The performance of both ML models for these countries varied amongst the 2 different scenarios. Given the observed results, it is not possible to select the most suitable scenario Therefore, both daily and weekly sampling can be used in machine learning modeling. Comparison between analytical and machine learning models using the deviation from the target value indicated that the MLP in both scenarios delivered the most accurate results.

Up to 150 days using the ML models, extrapolation for long-term prediction of was

tested. For the four countries, the actual prediction of MLP and ANFIS was reported which showed the progression of the outbreak.

3.2 Study Design and Data Source

We used the present daily data on number of COVID-19 infected cases, recoveries, and deaths in India to predict the amount of infected and removed cases at any given time (L. Wang et al., 2020). We obtained the data (up to March 23) from the 2020 Novel Coronavirus Visual Dashboard operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) and from covid19india.org (covid19india, 2020; Dongetal., 2020; Johns Hopkins University Center for Systems Science and Engineering, n.d.).

3.3 Statistical Model for Predictions

The true underlying probabilities of the three compartments follow a latent Markov transition process, and that we observe only the daily proportions of infected cases and removed cases and by assuming this the eSIR model works. First, let us establish some notation. Assume that the observed proportions of infected and removed cases on day t are denoted by Y_t^I and Y_t^R , respectively. Further, denote the true underlying probabilities of the S, I, and R compartments on day t by θ_t^S , θ_t^I , and θ_t^R , respectively, and assume that for any t , $\theta_t^S + \theta_t^I + \theta_t^R = 1$.

Assuming a usual SIR model on the true proportions (Appendix Figure A1-A), we have the following set of differential equations:

$$\begin{aligned} \frac{d\theta_t^S}{dt} &= -\beta\theta_t^S\theta_t^I, \\ \frac{d\theta_t^I}{dt} &= \beta\theta_t^S\theta_t^I - \gamma\theta_t^I, \\ \frac{d\theta_t^R}{dt} &= \gamma\theta_t^I. \end{aligned}$$

$$\frac{d\theta_t R}{dt} = \gamma \theta_t I - \frac{d\theta_t}{dt} R \quad = \quad \gamma \theta_t I - \frac{d\theta_t}{dt} R$$

Here, $\beta > 0$ denotes the disease transmission rate, and $\gamma > 0$ denotes the removal rate. The basic reproduction number $R_0 = \frac{\beta}{\gamma}$ indicates the expected number of cases generated by one infected case in the absence of any intervention and assuming that the whole population is susceptible. We assume a Beta-Dirichlet state-space model, independent conditionally on the underlying process at this stage, for the observed infected and removed proportions.

$$Y_t^I | \theta_t, \tau \sim \text{Beta}(\lambda I \theta_t, \lambda(1 - \theta_t)), Y_t^R | \theta_t, \tau \sim \text{Beta}(\lambda R \theta_t, \lambda(1 - \theta_t))$$

$$Y_t^I | \theta_t, \tau \sim \text{Beta}(\lambda I \theta_t, \lambda(1 - \theta_t)), Y_t^R | \theta_t, \tau \sim \text{Beta}(\lambda R \theta_t, \lambda(1 - \theta_t))$$

$$Y_t^I | \theta_t, \tau \sim \text{Beta}(\lambda I \theta_t, \lambda(1 - \theta_t)), Y_t^R | \theta_t, \tau \sim \text{Beta}(\lambda R \theta_t, \lambda(1 - \theta_t))$$

On the latent proportions, furtherly the Markov process is built as:

$$\theta_t | \theta_{t-1}, \tau \sim \text{Dirichlet}(\kappa f(\theta_{t-1}, \beta, \gamma))$$

where θ_t denotes the vector of the underlying population probabilities of the three compartments, whose mean is modeled as an unknown function of the probability vector from the previous time point, along with the transition parameters; $\tau = (\beta, \gamma, \theta_0, \lambda, \kappa)$ denotes the whole set of parameters where λI , λR and κ are parameters controlling variability of the observation and latent process, respectively. The function $f(\cdot)$ is then

determined by the SIR dynamical system, solved as the mean transition probability using a fourth-order Runge-Kutta approximation.

The eSIR Model has the priors and the MCMC Algorithm Setup. On the initial vector of latent probabilities the prior is set as

$$\theta_0 \sim \text{Dirichlet}(1 - Y_{1I} - Y_{1R}, Y_{1I}, Y_{1R}), \theta_0 S = 1 - \theta_0 I - \theta_0 R. \mathbf{\theta} \sim \text{Dirichlet}(1 - Y_{1I} - Y_{1R}, Y_{1I}, Y_{1R}), \theta_0 S = 1 - \theta_0 I - \theta_0 R.$$

The prior distribution of the basic reproduction number is $R_0 \sim \text{LogNormal}(0.582, 0.223)$ so that $E(R_0) = 2$ and $SD(R_0) = 1$, where E and SD denote the mean and standard deviation, respectively. The prior distribution of the removal rate is $\gamma \sim \text{LogNormal}(-2.955, 0.910)$ so that $E(\gamma) = 0.082$ and $SD(\gamma) = 0.1$. The prior mean of the removal rate γ indicates an average infectious period of 12 days, which is originally set using estimates from the SARS outbreak in Hong Kong (Mkhatshwa & Mummert, 2010) due to the similarity between the two viruses; this value also aligns well with several recent studies on COVID-19 in China (Chen et al., 2020; Li et al., 2020; Ryu & Chun, 2020). The basic reproduction number, 2.0, is approximately the average of estimates from many other COVID-19 studies on the Indian population is the prior mean. To evaluate how robust the model is toward the prior settings using Indian population COVID-19 data we have conducted a sensitivity analysis. With more observed data of a longer exponentially increasing period and stronger intensities by focusing on cities or states that are highly exposed the sensitivity issue can be minimized. Note that the prior mean of the distribution of the transmission rate β equals

γ_{R0} The default choice is to set large variances in both observed and latent processes for the variability parameters, which may be adjusted over the course of the epidemic with more data becoming available:

$\kappa, \lambda_I, \lambda_R \sim \text{iid Gamma}(2, 0.0001)$. $\kappa, \lambda_I, \lambda_R \sim \text{iid Gamma}(2, 0.0001)$.

Denoting t_0 as the last date of data availability, and assuming that the forecast spans over the period $[t_0 + 1, T]$, our algorithm is as follows.

0. Take M draws from the posterior $[\theta_{1:t_0}, \tau \mid Y_{1:t_0}]$.

1. For each solution path $m \in \{1, \dots, M\}$, iterate between the following two steps via MCMC.

i. Draw $\theta_t(m)$ from $[\theta_t \mid \theta_{t-1}(m-1), \tau(m)]$, $t \in \{t_0 + 1, \dots, T\}$.

ii. Draw $Y_t(m)$ from $[Y_t \mid \theta_t(m), \tau(m)]$, $t \in \{t_0 + 1, \dots, T\}$.

Modeling Intervention. By assuming that the intervention will result in a decrease in the transmission from the S compartment to the I compartment we model the effect of interventions. By decreasing the effective rate of transition we do so (or, equivalently, the chance of interaction between members of S and

l), by introducing a time-varying transmission rate modifier

$\pi(t) \in [0, 1]$. The flow between the three compartments (Appendix Figure A1-B) via a set of differential equations is updated as follows:

$$\begin{aligned} \frac{d\theta S}{dt} &= -\beta \pi(t) \theta S \theta I, \frac{d\theta S}{dt} = -\beta \pi(t) \theta S \theta I, \\ \frac{d\theta I}{dt} &= \beta \pi(t) \theta S \theta I - \gamma \theta I, \frac{d\theta I}{dt} = \beta \pi(t) \theta S \theta I - \gamma \theta I, \\ \frac{d\theta R}{dt} &= \gamma \theta I, \frac{d\theta R}{dt} = \gamma \theta I. \end{aligned}$$

The reproductivity is thus modified by the intervention over time as $R_0 \pi(t)$. We follow an example given by L to better understand the introduction of this effect modifier. Wang et al. (2020). Suppose at a time t , $q_S(t) \in [0, 1]$ is the chance of an at-risk person being in home isolation, and $q_I(t) \in [0, 1]$ is the chance of an infected person being in hospital quarantine. Consequently, the chance of disease transmission when an at-risk person meets an infected person is $\beta \{1 - q_S(t)\} \theta S \{1 - q_I(t)\} \theta I := \beta \pi(t) \theta S \theta I$. This $\pi(t)$ modifies the chance of a susceptible person meeting with an infected person, which is termed as a transmission modifier in effect. The functional form of $\pi(t)$ is a continuous function that reflects a combination of steadily increased community-level awareness and responsible quarantine and preventive measures, and the country-wide lockdown measures initiated by

the government in this research. Into the differential equations as well as the MCMC algorithms, this predefined transmission modifier can be smoothly incorporated. In reflecting the changing pattern of human intervention that affects the transmission rate of the epidemic within the population its functional form can be quite flexible.

Implementation of the eSIR Model. The differential equations were solved via a fourth-order Runge–Kutta approximation and we implemented the proposed algorithm in R package rjags (Plummer et al.). To ensure quality of the MCMC, we set the adaptation number to be 10^4 , thinned the chain by keeping one draw from every 10 random draws to reduce autocorrelation, set a burn-in period of 10^5 draws to let the chain stabilize, and started from 4 separate chains. Thus, in total, we have $2 \times 10^5 \times 10^5$ effective draws with about $2 \times 10^6 \times 10^6$ draws discarded. Consequently there might be risk the quality of data but one could reduce the computation time. Not only posterior estimation on parameters and prevalence of all three compartments in the SIR model is provided, but also predicted proportions of the infected and the removed cases at future time points is provided with this implementation. We used 1.34 billion as the population of India, thus treating the country as a homogeneous system for the outbreak to obtain predicted case counts from the predicted prevalence.

Uncertainty Quantification. Uncertainty associated with all parameters and functions of parameters can be calculated from exact posterior draws without relying on large-scale approximation or the delta method, this is one major advantage of a Bayesian implementation. The credible intervals (CrI) for the prevalence are computed using the posterior distribution of proportions given the observed confirmed and removed prevalence, that is, $Y(t_0+1):TI \mid Y_{1:t_0I}, Y_{1:t_0R} Y_{\{(t_0+1):T\}^I} \mid Y_{1:t_0I}, \backslash$
 $Y_{1:t_0R} Y_{\{(t_0+1):T\}^R} \mid Y_{1:t_0I}, Y_{1:t_0R}$ and
 $Y(t_0+1):TR \mid Y_{1:t_0I}, Y_{1:t_0R} Y_{\{(t_0+1):T\}^R} \mid Y_{1:t_0I}, \backslash$
 $Y_{1:t_0R} Y_{\{(t_0+1):T\}^I} \mid Y_{1:t_0I}, Y_{1:t_0R}$, where t_0 denotes the last observed date, and TTT denotes the last forecast date. More

specifically, suppose we want to compute the 95% posterior CrI for the

observed proportion of confirmed cases on the first day of forecast, that is, a CrI for the random variable $Y_{t_0+1}|Y_{t_0} + 1^{\{I\}}|Y_{t_0} + 1$. Then, from the MMM solution paths of the posterior, we have the draws

$\{Y_{t_0+1}(m), 1 \leq m \leq M\} \leftarrow \{Y_{t_0} + 1^{\{I\}}(m), 1 \leq m \leq M\}$. We construct a 95% posterior CrI for $Y_{t_0+1}|Y_{t_0} + 1^{\{I\}}|Y_{t_0} + 1$ by simply computing the 2.5th upper and lower percentiles from this set of M draws. The confidence interval for the sum can be calculated in a similar way to sums of the draws from the I and R compartments at a given time with use of cumulative prevalences. From prevalences by using population size case counts can be obtained. Similar techniques apply to $\theta_{t_0+j}|\theta_{t_0} + j^{\{I\}}|\theta_{t_0} + j$ for any $1 \leq j \leq T - t_0$ and transmission parameters like β and γ . For instance, a 95% posterior CrI for β can be constructed by calculating the 2.5th upper and lower percentiles of $\{\beta(m), 1 \leq m \leq M\}$.

Therefore, we could simply define $R(m) = \beta(m)\gamma(m) \forall 1 \leq m \leq M$ and compute the 95% posterior CrI for the effective reproduction number R from

$\{R(m), 1 \leq m \leq M\} \leftarrow \{R^{\{I\}}(m), 1 \leq m \leq M\}$.

The compute and storage requirements increase rapidly with the number of training vectors because Support Vector Machines are powerful tools. Separating support vectors from the rest of the training data, the core of an SVM is a quadratic programming problem (QP).

Between $O(n_{\text{features}} \times n_{\text{samples}}^2)$ and $O(n_{\text{features}} \times n_{\text{samples}}^3)$ depending on how efficiently the libsvm cache is used in practice (dataset dependent), the QP solver used by the libsvm-based implementation scales. If the data is very sparse n_{features} should be replaced by the average number of non-zero features in a sample vector.

For the linear case, the algorithm used in Linear SVC by

the liblinear implementation is much more efficient than its libsvm-based SVC counterpart and can scale almost linearly to millions of samples and/or features.

CUSTOM KERNELS: You can define your own kernels by either giving the kernel as a python function or by precomputing the Gram matrix.

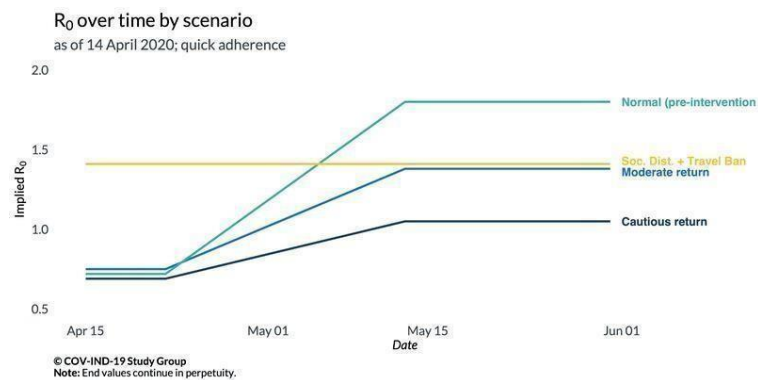
Custom kernels of classifiers behave the same way as any other classifiers, except that:

- Field `support_vectors_` is now empty, only indices of support vectors are stored in `support_`
- A reference (and not a copy) of the first argument in the `fit()` method is stored for future reference. You will have unexpected results, if that array changes between the use of `fit()` and `predict()`.

3.4 Parameter Choices for Short-Term Forecasts:

To assess the short-term impact of lockdown as well as the long-term impact of lockdown and post-lockdown activities, we made projections of the cumulative number of cases over a time horizon. For the short-term forecast on April 30, we assumed lockdown is implemented until March 23 with either a 1- or a 2- week delay in people's adherence/compliance to lockdown restrictions. With two hypothetical scenarios, we compared these projections: (A) no non-pharmaceutical intervention (i.e., a constant disease transmission rate over time since the first case was reported in India), (B) With social distancing and travel bans only, a moderate intervention (i.e., a decreased transmission rate compared to no intervention). The prior mean for R_0 (the expected number of cases generated by one infected person assuming that the whole population is susceptible) was set at 2.0. Based on the early phase data in India and is consistent with other models this was estimated. We chose the transmission rate and the removal rate such that the means for the prior distribution of the basic reproductive number R_0 are 2.0 and 1.5, respectively ($SD = 1$) for the no intervention and the moderate intervention scenarios,. The change in R_0 from 2.0 to 1.5 as an effect of intervention was created based on what we saw regarding the effect of interventions and the relative reduction of

R_{0R_0} in Wuhan (C. Wang et al., 2020). The assumption on similar effect of interventions on the pandemic across the two countries does not seem too restrictive given the similar population size and comparable population densities in China and India. Our chosen mean for the prior of R_{0R_0} starts with 2.0 and moves back up to 1.5 after the lockdown ends as described in Figure 3 (assuming a gradual, moderate resumption of daily activities) for the current scenario of lockdown. The drop in R_{0R_0} from 2.0 to 0.8 during lockdown represents a 60% reduction, which is proportionally slightly less than the ~65% drop estimated in R_{0R_0} from the introduction of *cordon sanitaire*, or restriction of movement of people from the coronavirus outbreak in Wuhan.



3.4 Proposed model

3.5 Parameter Choices for Long-Term Forecasts

We considered three hypothetical post-lockdown scenarios: (i) people return to normal activities due to the urgent desire for reconnecting after lockdown, (ii) people return to moderate activities as they did during the period with social distancing and travel ban intervention, and (iii) people make a cautious return out of fear for the coronavirus and partake in subdued activities, for the long-term forecasts. For these three scenarios, we assume the prior mean on R_{0R_0} moves back up from 0.8 to 2.0, 1.5 and 1.2, respectively, 3 weeks after lockdown ends on April 14. We compared

these post-lockdown scenarios with another hypothetical scenario involving perpetual social distancing and travel ban without any lockdown (we fixed the prior mean on R_0 at 1.5 over the entire forecasting interval). The time-dependent changes to R_0 values across our simulation scenarios are depicted in Figure 3.1.

The model takes the available data which includes latitude, longitude, days, cases, deaths, positive outcomes from the datasets and validates it by training the model, then the hyper parameters are applied with data visualization, regression and support vector machine which by using prophet predicts the outbreak of cases that are confirmed, death and recovered.

To find the correlation between each and every features heat map is used. Heat map uses the correlation matrix. We visualize the results in the form of graphs and histogram.

The system architecture is shown here which takes us through the steps through which we predict the coronavirus outbreak.

CHAPTER 4

RESULTS AND DISCUSSION

Due to the immense number of algorithms available in the literature, the question of which model type to apply to a Machine Learning task can be a daunting one. As one can outperform the other in a certain class of problems while consistently coming in behind for another class, it can be difficult to compare the relative merits of two methods.

4.1 DECISION BOUNDARIES

A supervised machine learning method, the support vector machine (SVM) algorithm, has demonstrated high performance in solving classification problems in many biomedical fields, especially in bioinformatics. SVM discriminates between two classes by generating a hyperplane that optimally separates classes after the input data have been transformed mathematically into a high-dimensional space in contrast to logistic regression, which depends on a pre-determined model to predict the occurrence or not of a binary event by fitting data to a logistic curve. It may have important discriminative power for classification, especially in cases where sample sizes are small and a large number of variables are involved (high-dimensionality space) because the SVM approach is data-driven and model-free. To develop automated classification of diseases and to improve methods for detecting disease in the clinical setting, this technique has been recently used .

4.2 INTERPRETABILITY

We chose covid-19 as an example to test the potential power of SVM as an approach for classifying individuals into groups defined by disease status. In the U.S., covid-19 affects an estimated 23.6 million people, of whom about one third are unaware that they have the disease . Another 57 million people have pre-covid-19, with elevated blood glucose levels that increase their risk of developing virus, heart disease, and stroke. Recent studies indicate that covid-

19 can be prevented by lifestyle changes or pharmacotherapy among persons with pre-covid-19. Early screening and diagnosis is thus central to effective prevention strategies. To identify people at high risk of

developing covid-19 or with pre-covid-19 based on common risk factors such as body mass index (BMI) and family history of symptoms got enrolled, numerous risk scores and prediction models have been developed .

For example, a recently published risk calculator uses logistic regression to identify people with pre-covid-19 and undiagnosed covid-19 by employing combinations of common risk variables. Our objective is to analyse the present condition in India and other countries and also to analyse and examine the recovered, death and confirmed cases. The cured percentage and death percentage in the Indian states is analyzed. The total cases and new cases are shown for India, Italy, South Korea and Wuhan. The worldwide novel coronavirus cases are analyzed for recovered, death and confirmed and we predict the future outcome of outbreak. During this analysis and prediction we found that India is doing better so far in controlling the outbreak compared to other countries.

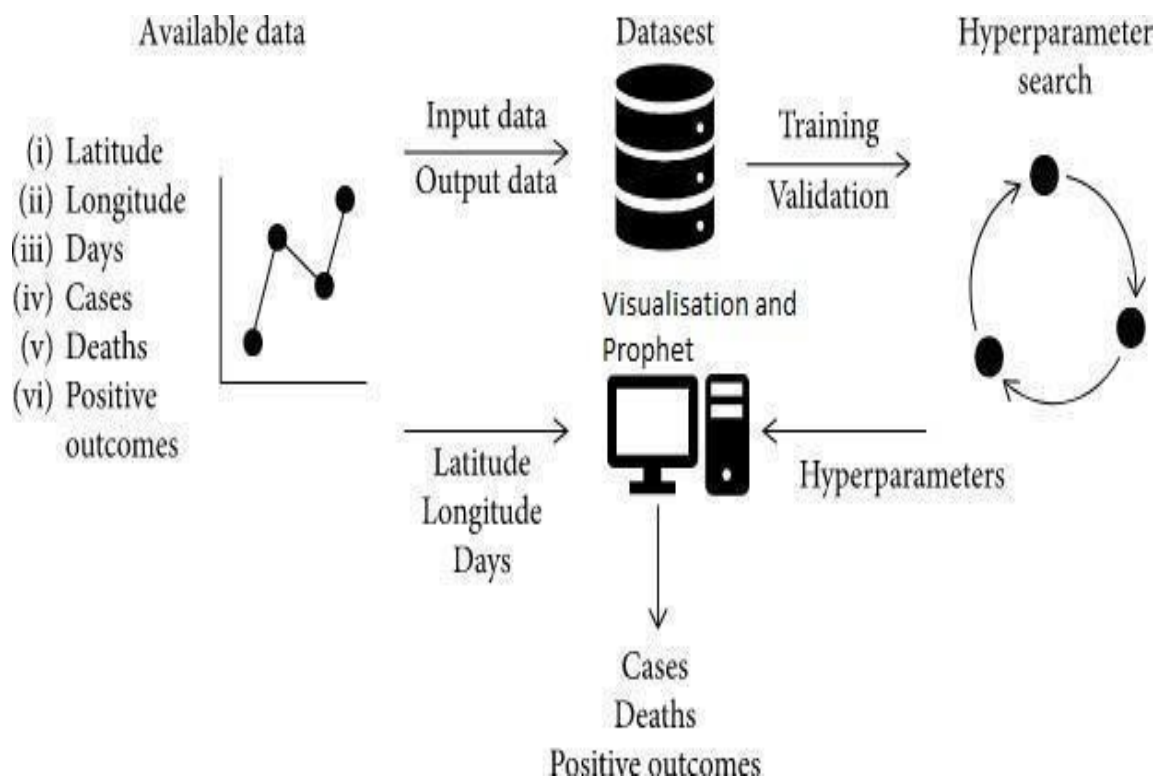


Fig-4.2 Architecture of Model

CHAPTER-5

CONCLUSION AND FUTURE WORK

The work of creating a model which can effectively predict and analyze coronavirus outbreak and analyse the present condition in India and to see if this trend is similar to other countries and to explore the world wide data and to forecast the world wide coronavirus cases using prophet has been successfully implemented. This model is very accurate and efficient in predicting coronavirus.

This model at present gives 95 percent accuracy. In the future there is a possibility of increasing the accuracy of model with efficient datasets. This model framework can be implemented using block chain, IOT, AI, many other upcoming and growing technologies for predicting and analyzing coronavirus. The pandemic is in the verge of increasing number of cases. So it is necessary for us to forecast the future cases and take precautions and be safe.

REFERENCES

1. Platt "Probabilistic outputs for SVMs and comparisons to regularized likelihood methods".
2. Wu, Lin and Weng, "Probability estimates for multi-class classification by pairwise coupling", JMLR 5:975-1005, 2004.
3. Fan, Rong-En, et al., "LIBLINEAR: A library for large linear classification.", Journal of machine learning research 9.Aug (2008): 1871-1874.
4. Chang and Lin, LIBSVM: A Library for Support Vector Machines.
5. Bishop, Pattern recognition and machine learning, chapter 7 Sparse Kernel Machines
6. "A Tutorial on Support Vector Regression", Alex J. Smola, Bernhard Schölkopf - Statistics and Computing archive Volume 14 Issue 3, August 2004, p. 199-222.
7. Schölkopf et. al New Support Vector Algorithms
8. Crammer and Singer On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines, JMLR 2001.

APPENDIX

A.SCREENSHOTS

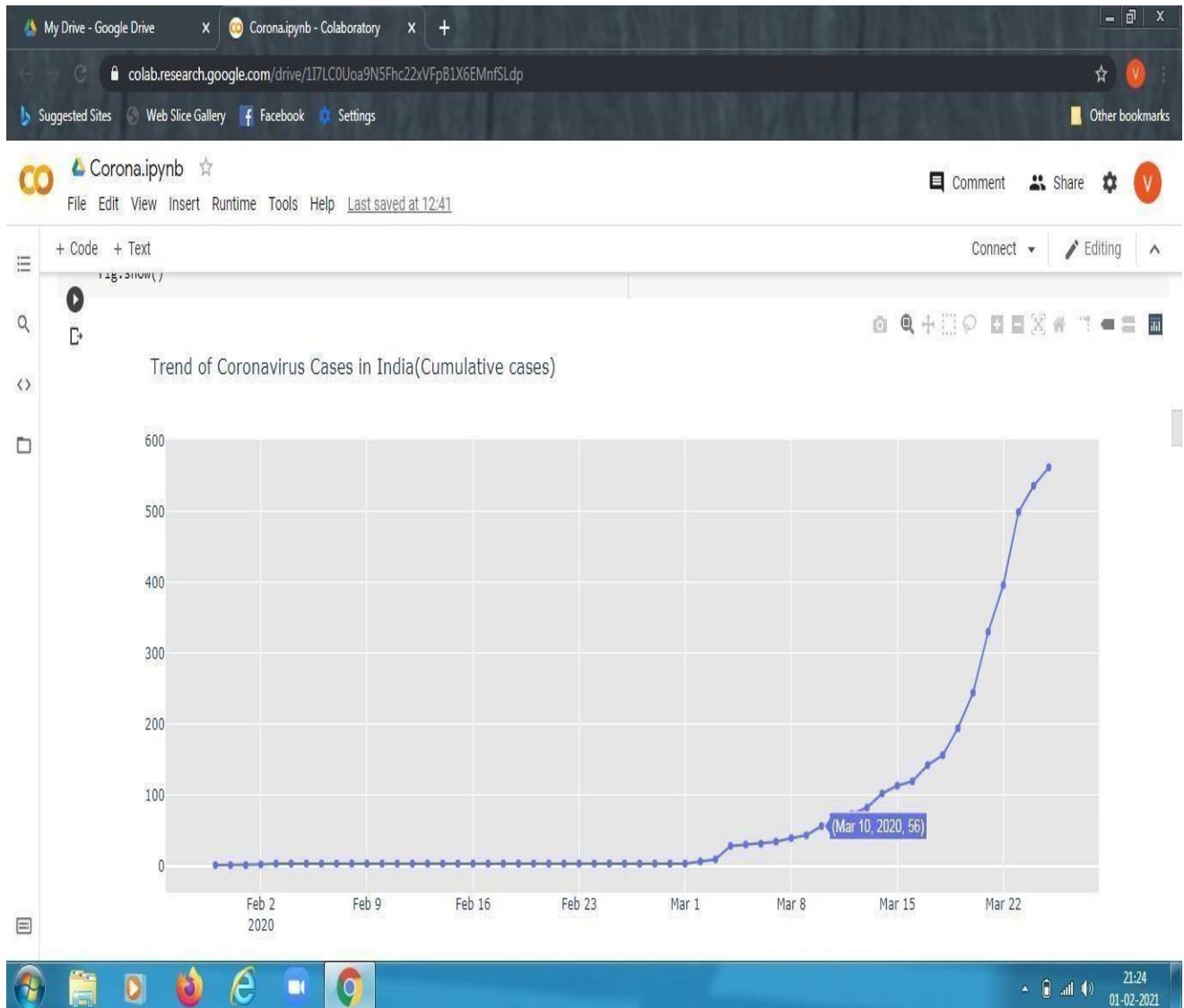


Fig-a:Trend of Coronavirus cases in India

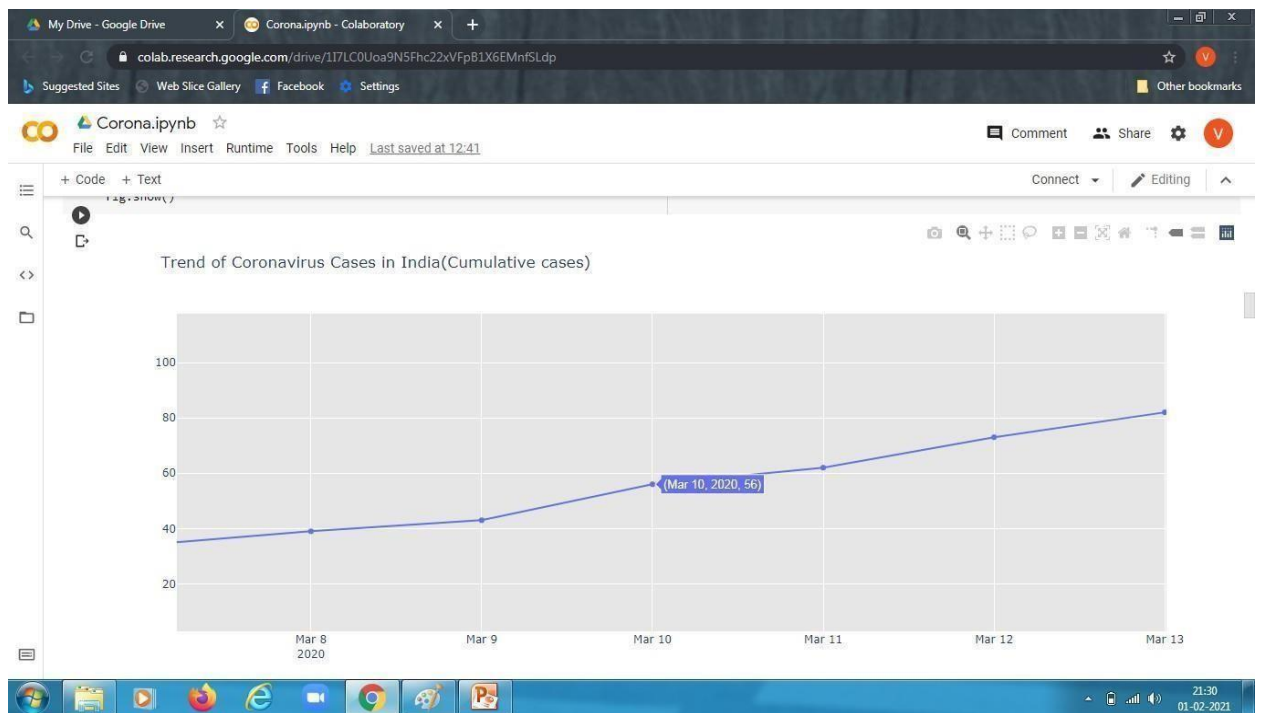


Fig-b: Closer version of fig.a

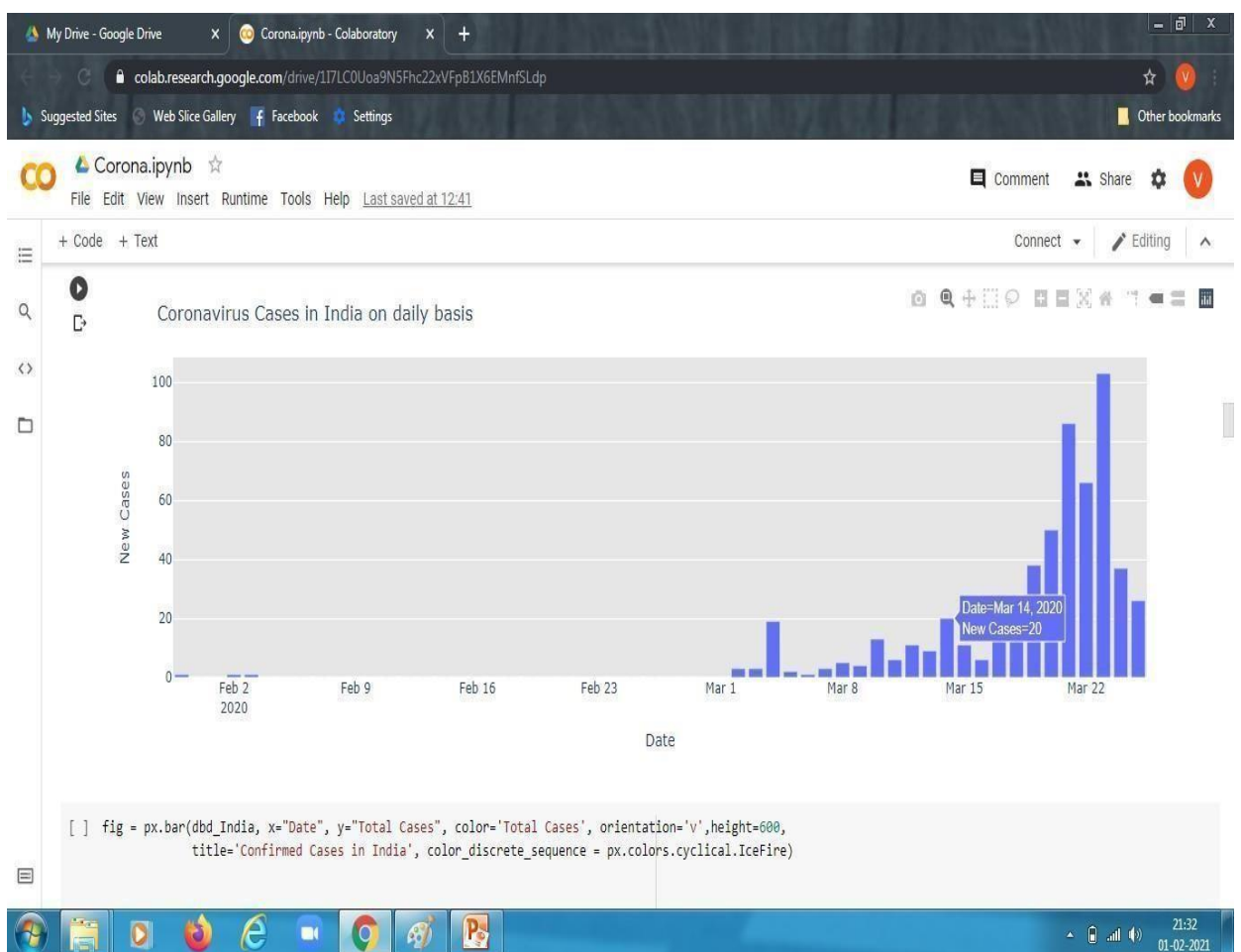


Fig-c: Trend of coronavirus cases in India on daily basis

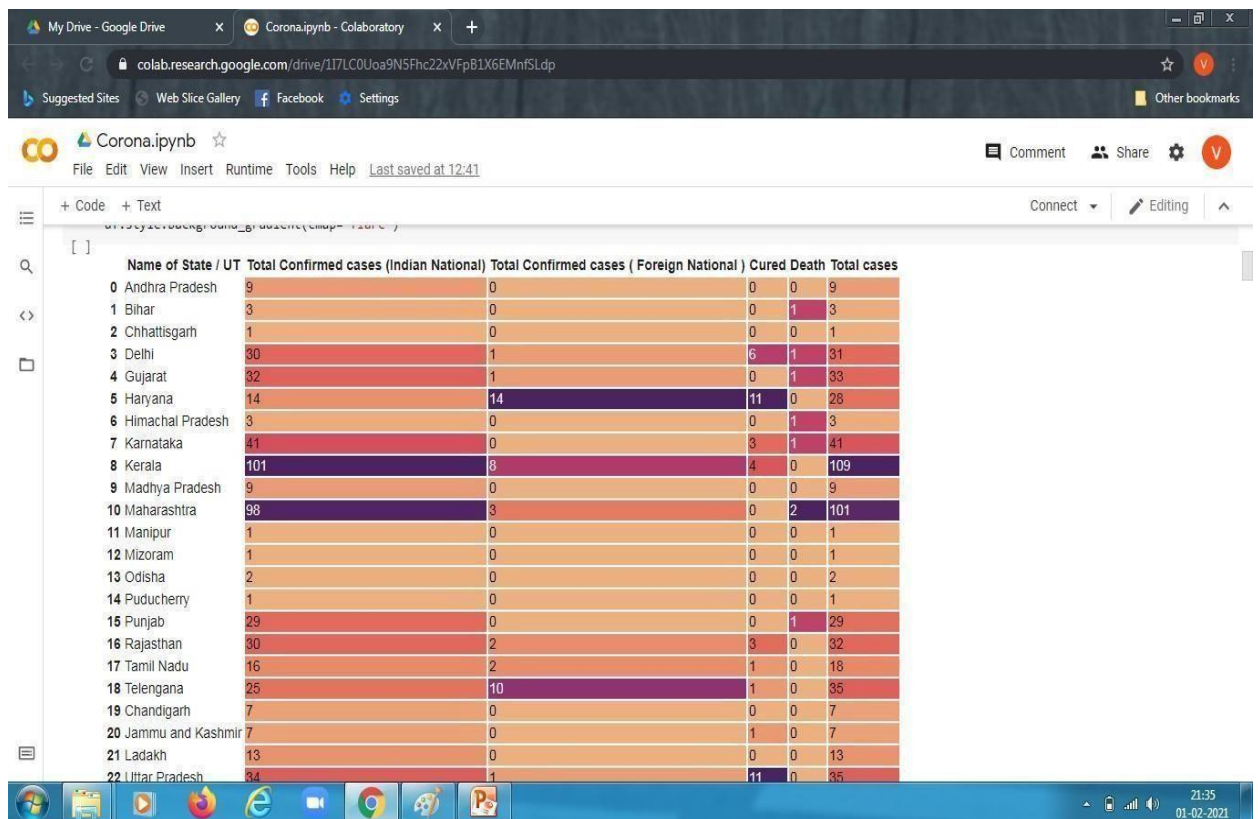


Fig-d:State wise coronavirus cases in India

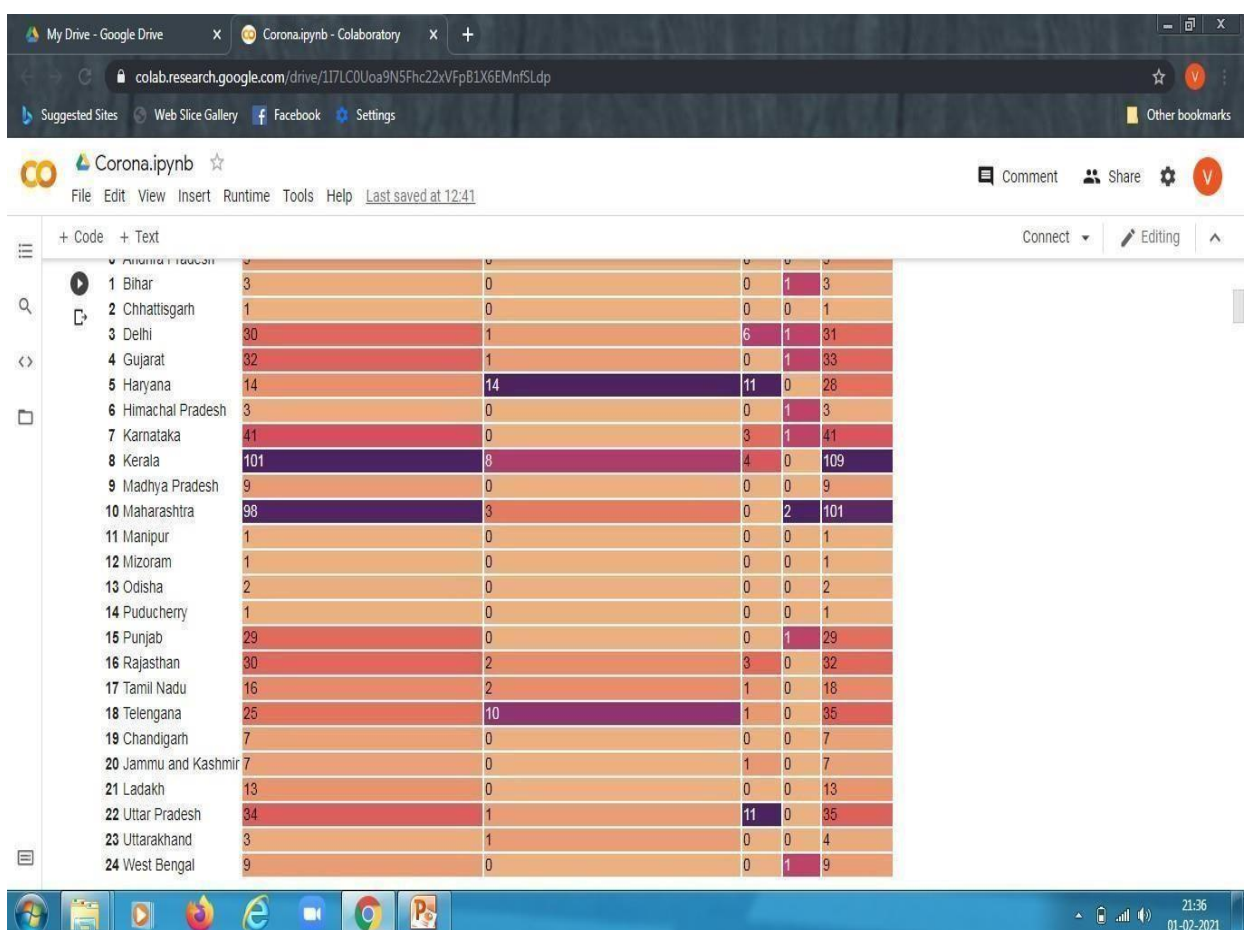


Fig-e:State wise coronavirus cases in India

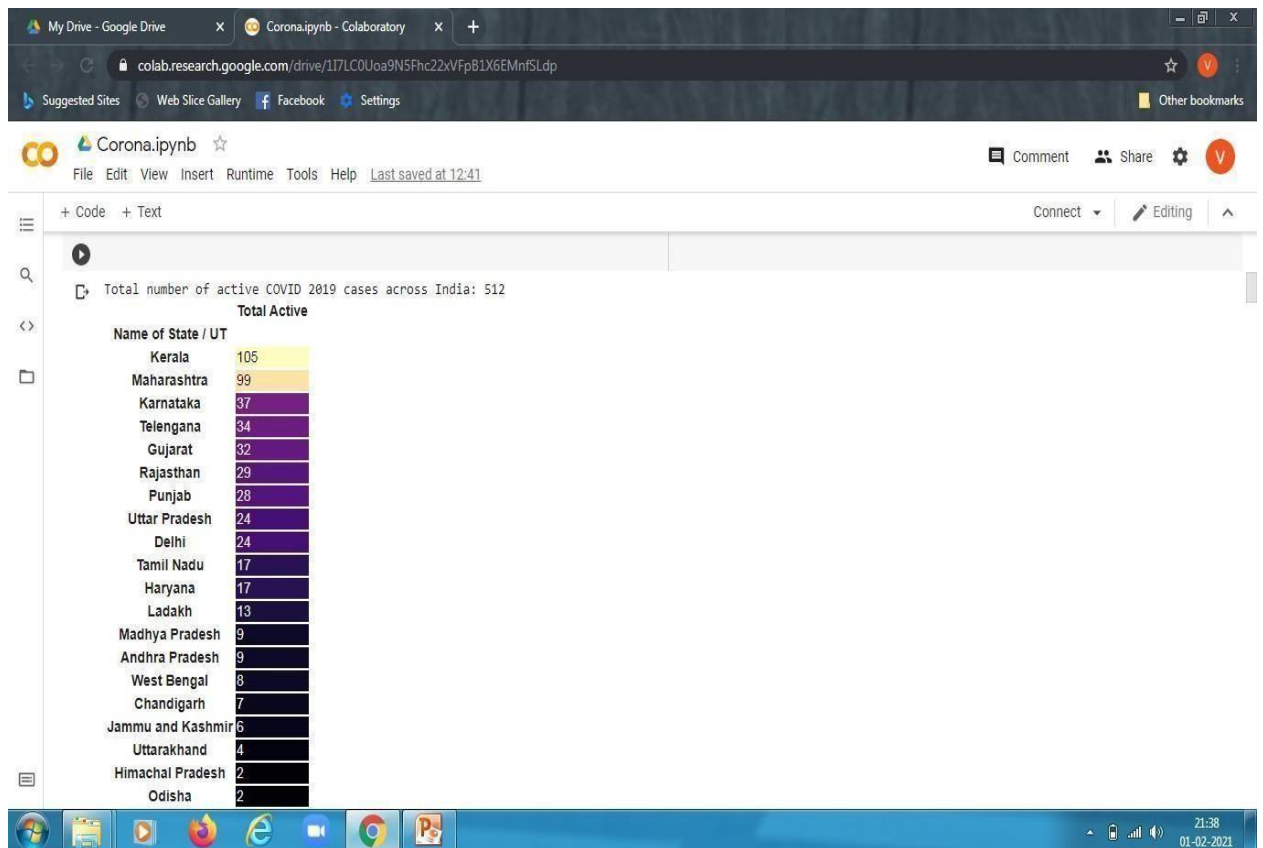


Fig-f:Active coronavirus cases in India

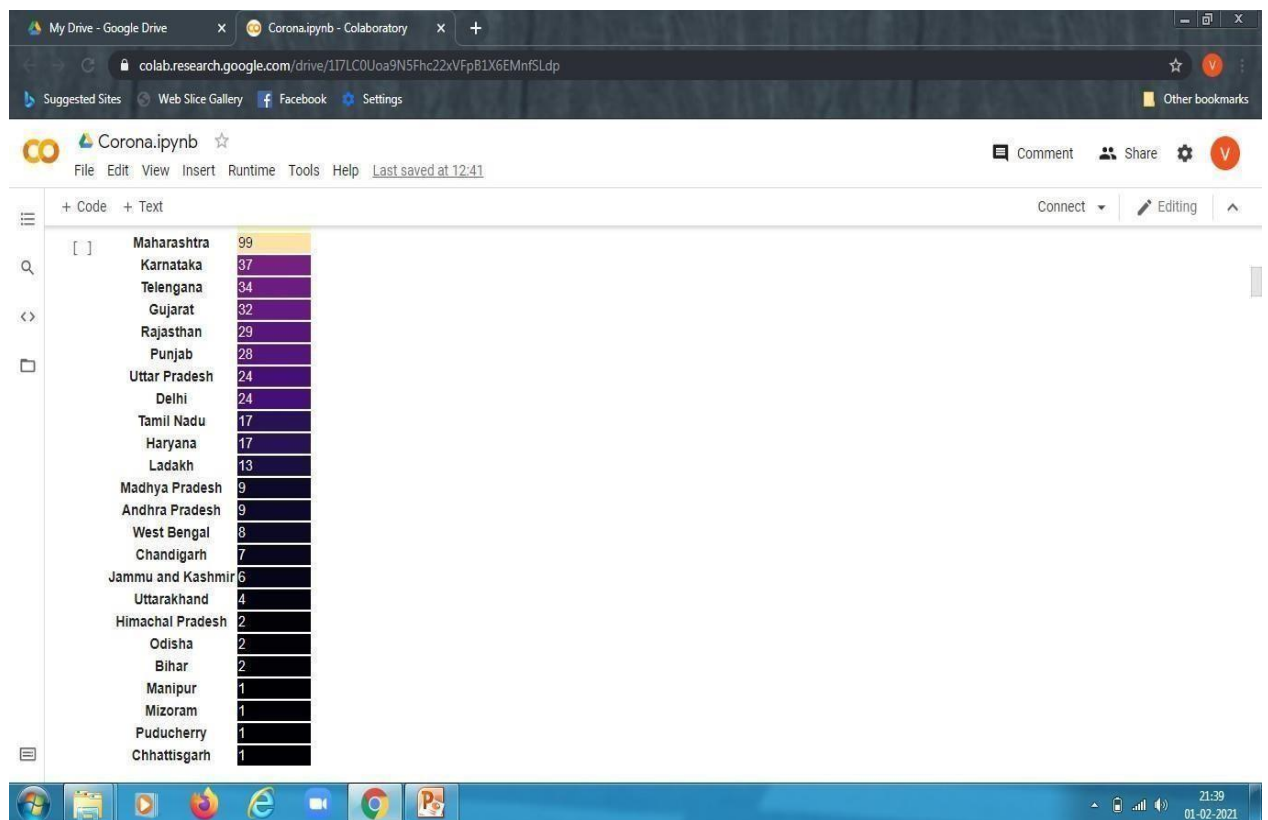


Fig-g:Active coronavirus cases in India

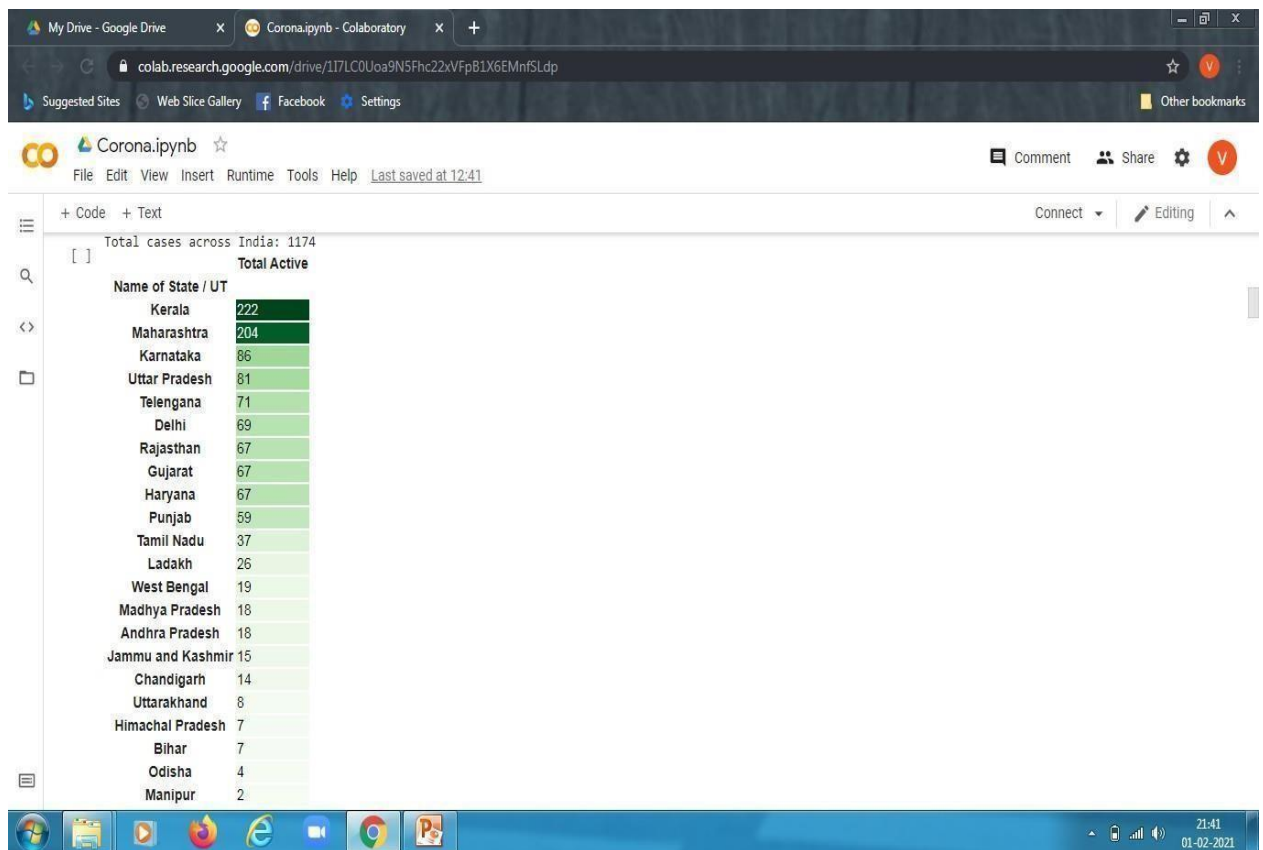


Fig-h: Total cases across India

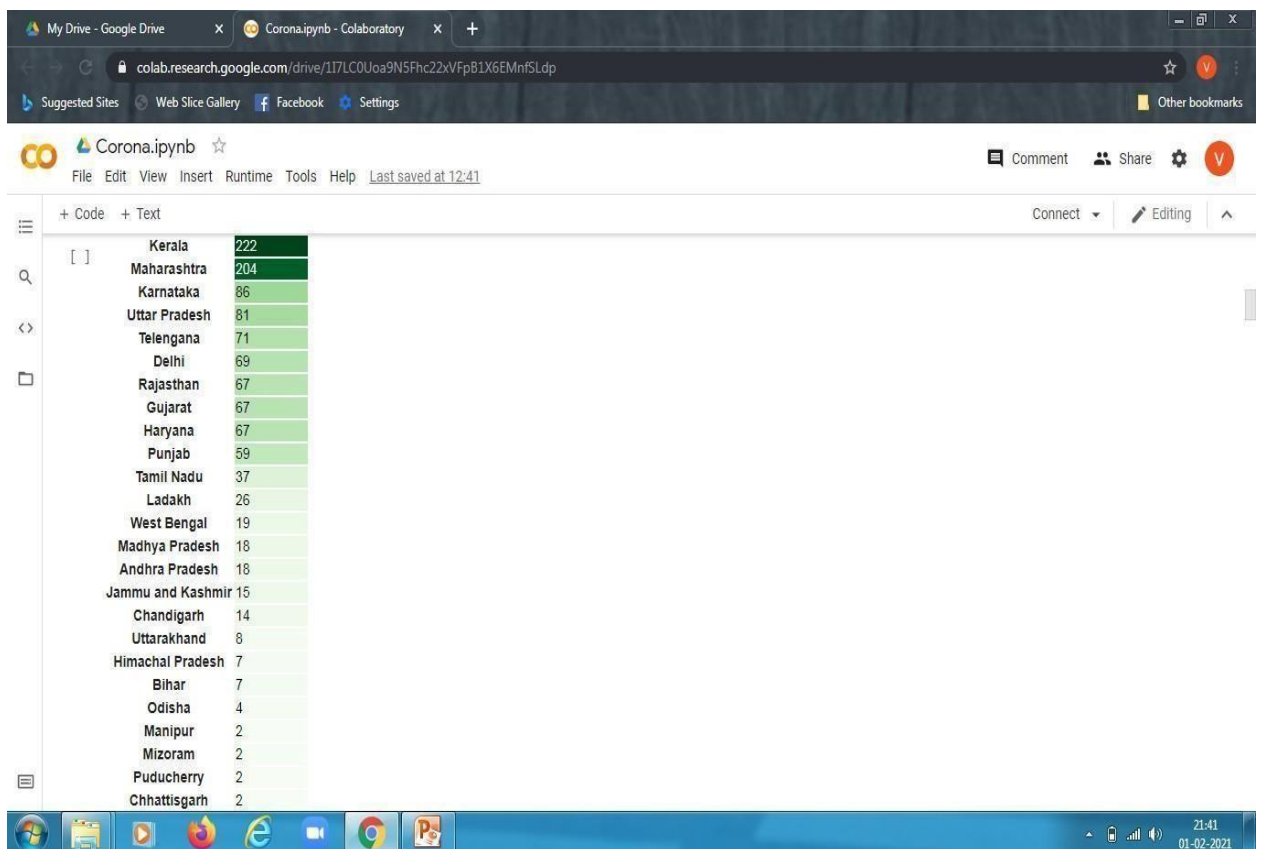


Fig-i: Total cases across India

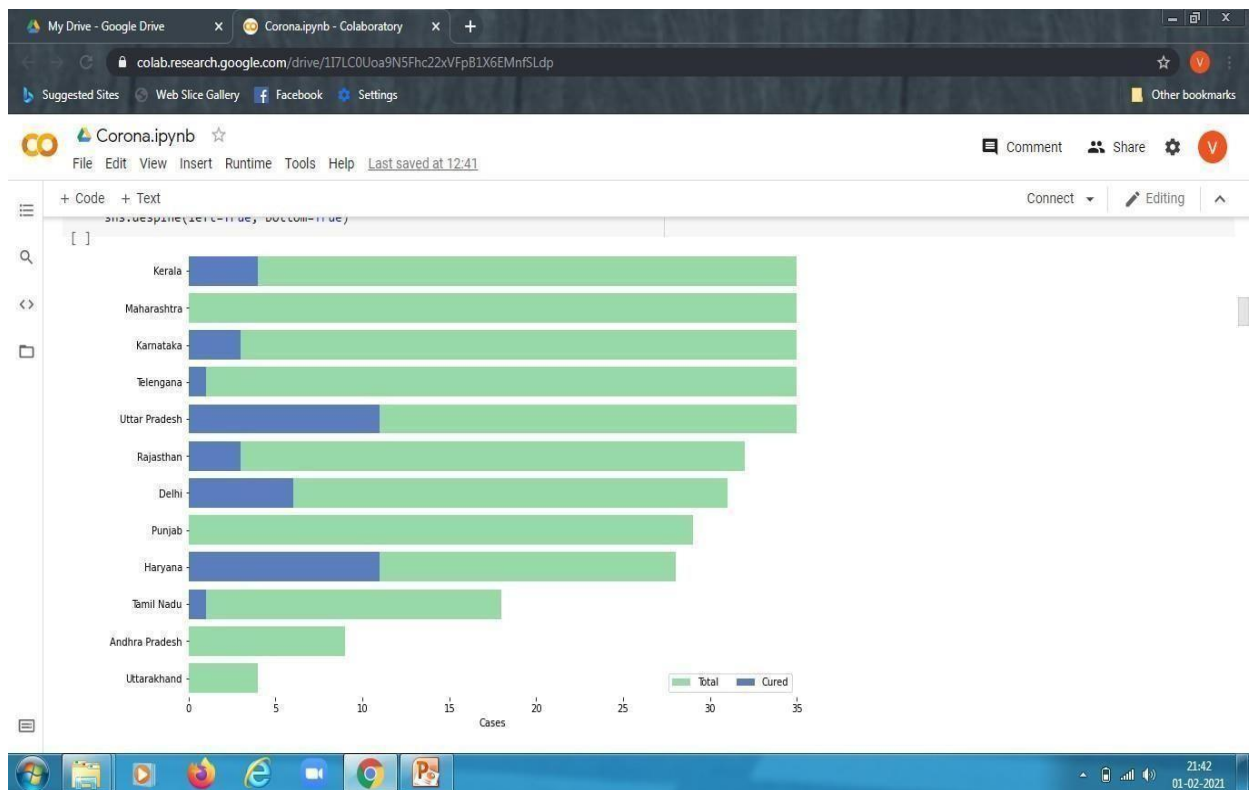


Fig-j: Cured cases in India

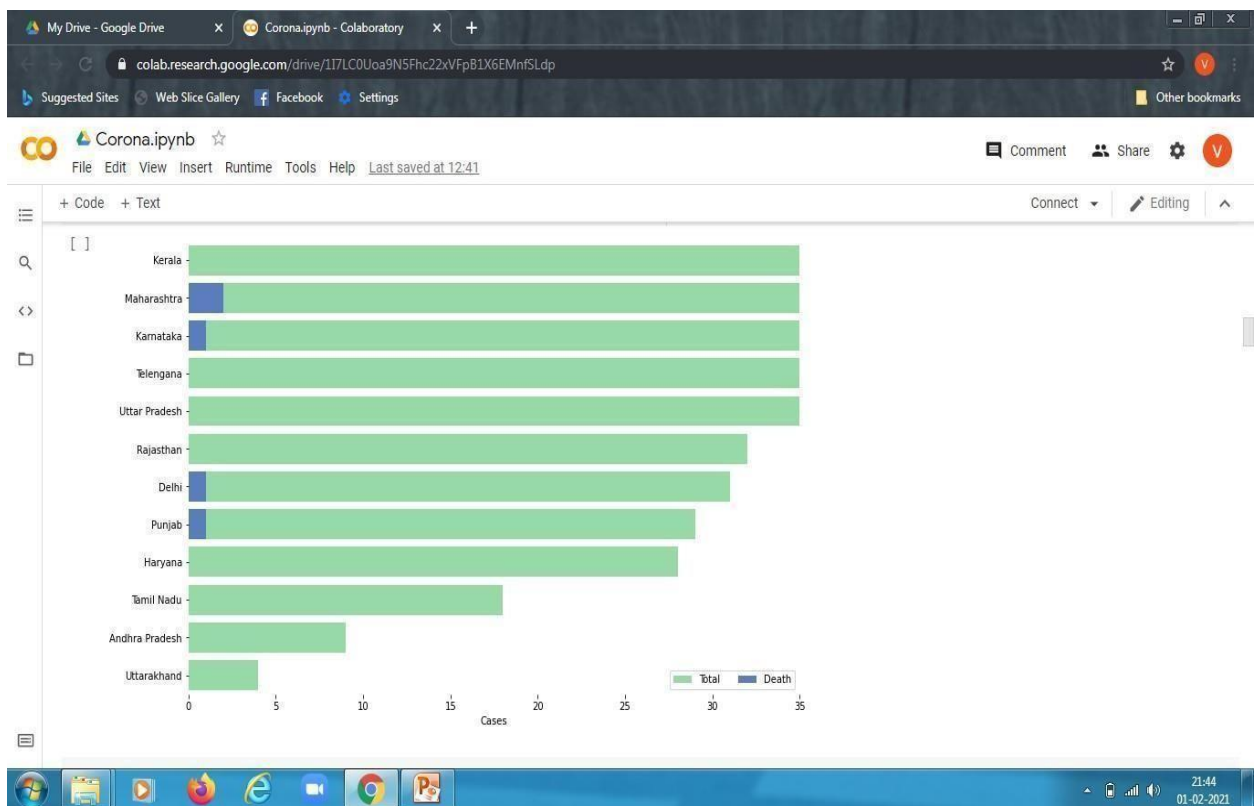


Fig-k: Death cases in India

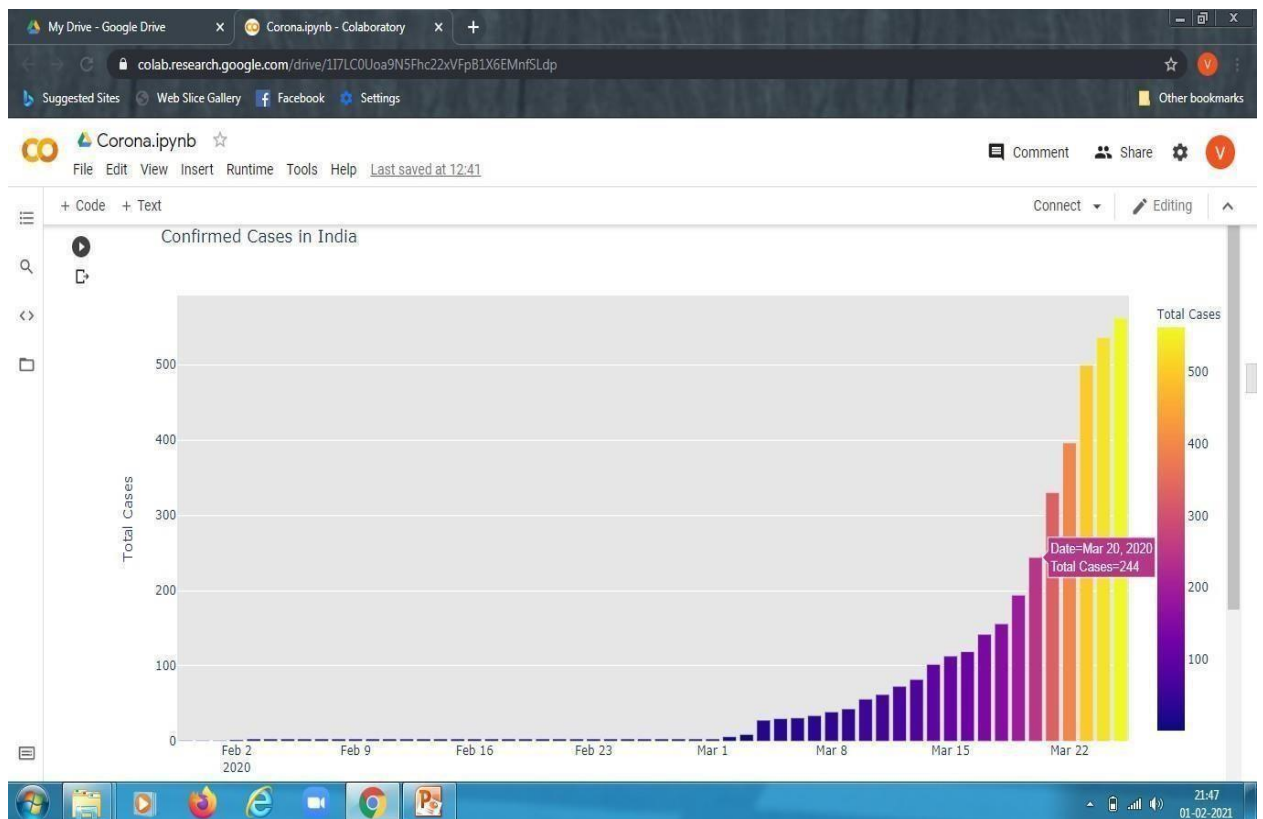


Fig-I:Confirmed cases in India

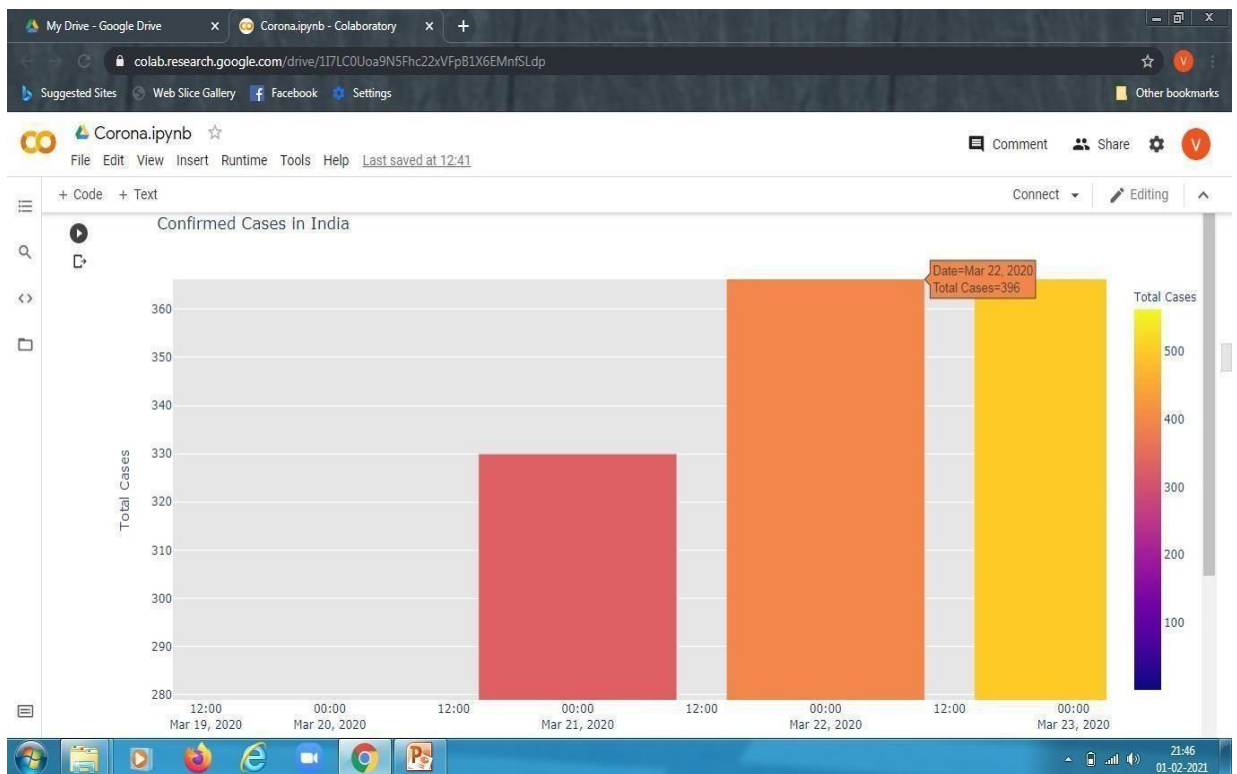


Fig-m:Closer view at confirmed cases in India

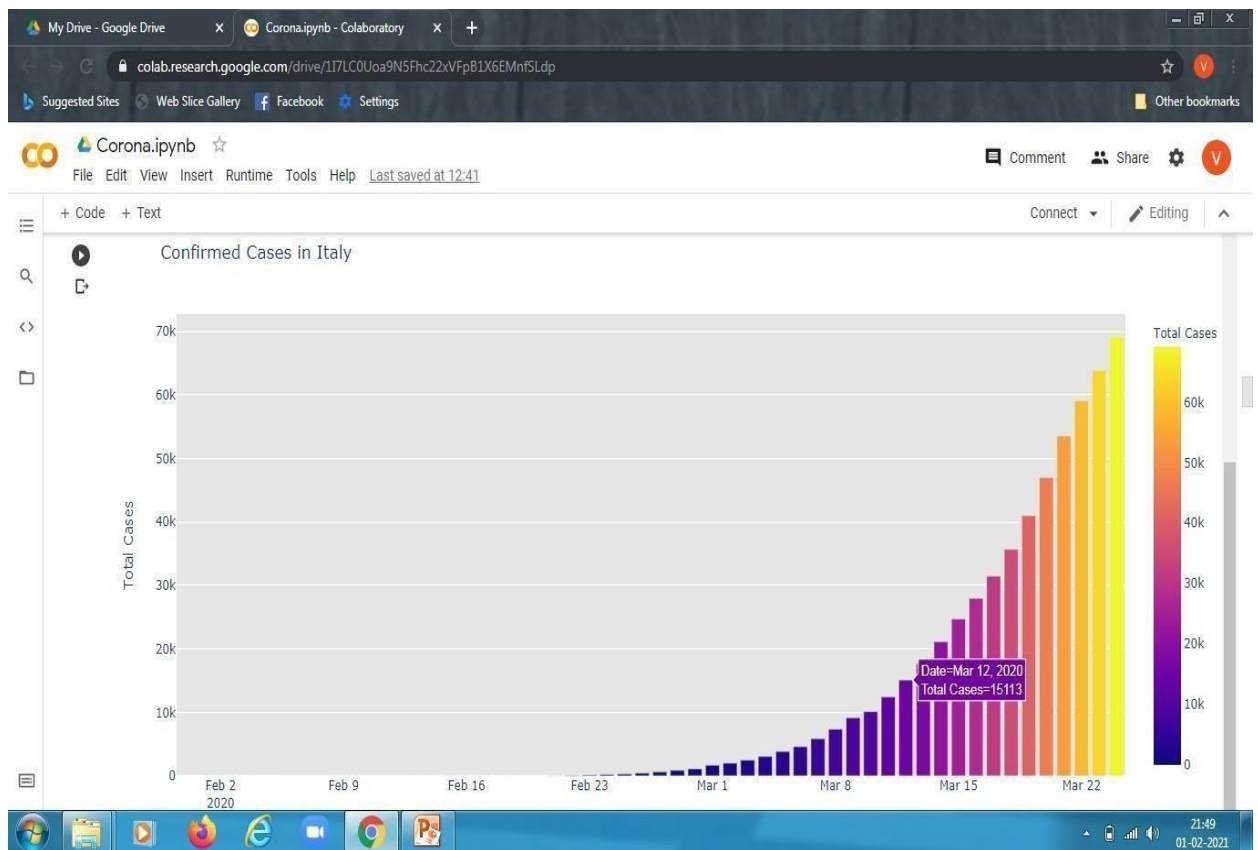


Fig-n:confirmed cases in Italy

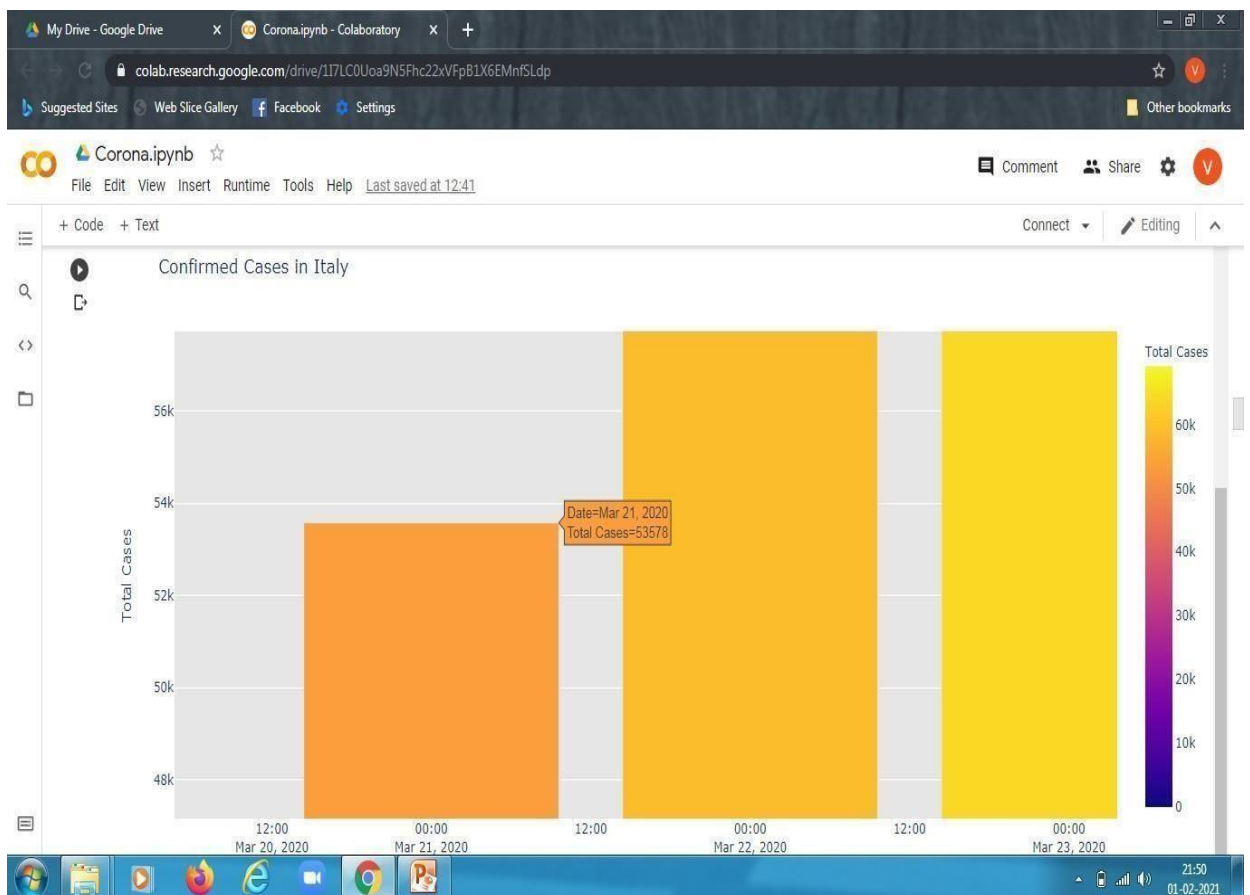


Fig-o:closer view at confirmed cases in Italy

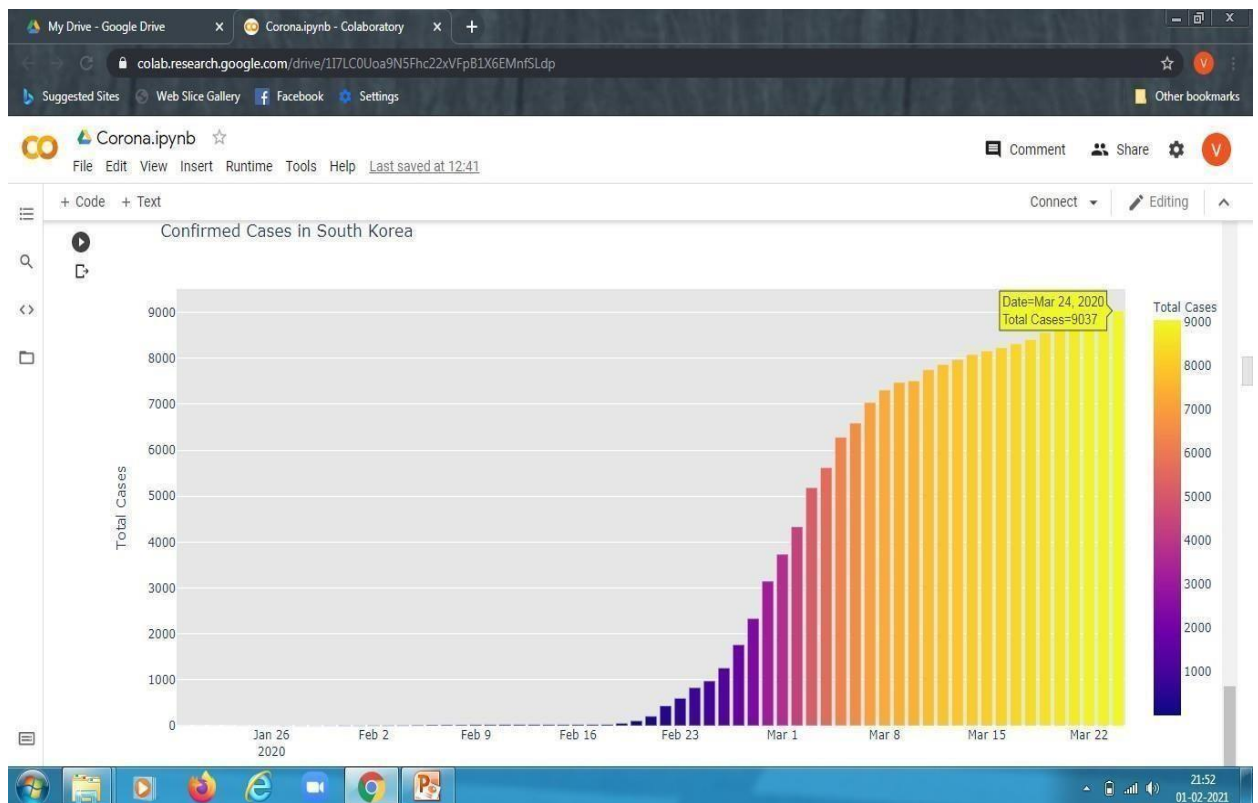


Fig-p:confirmed cases in South Korea

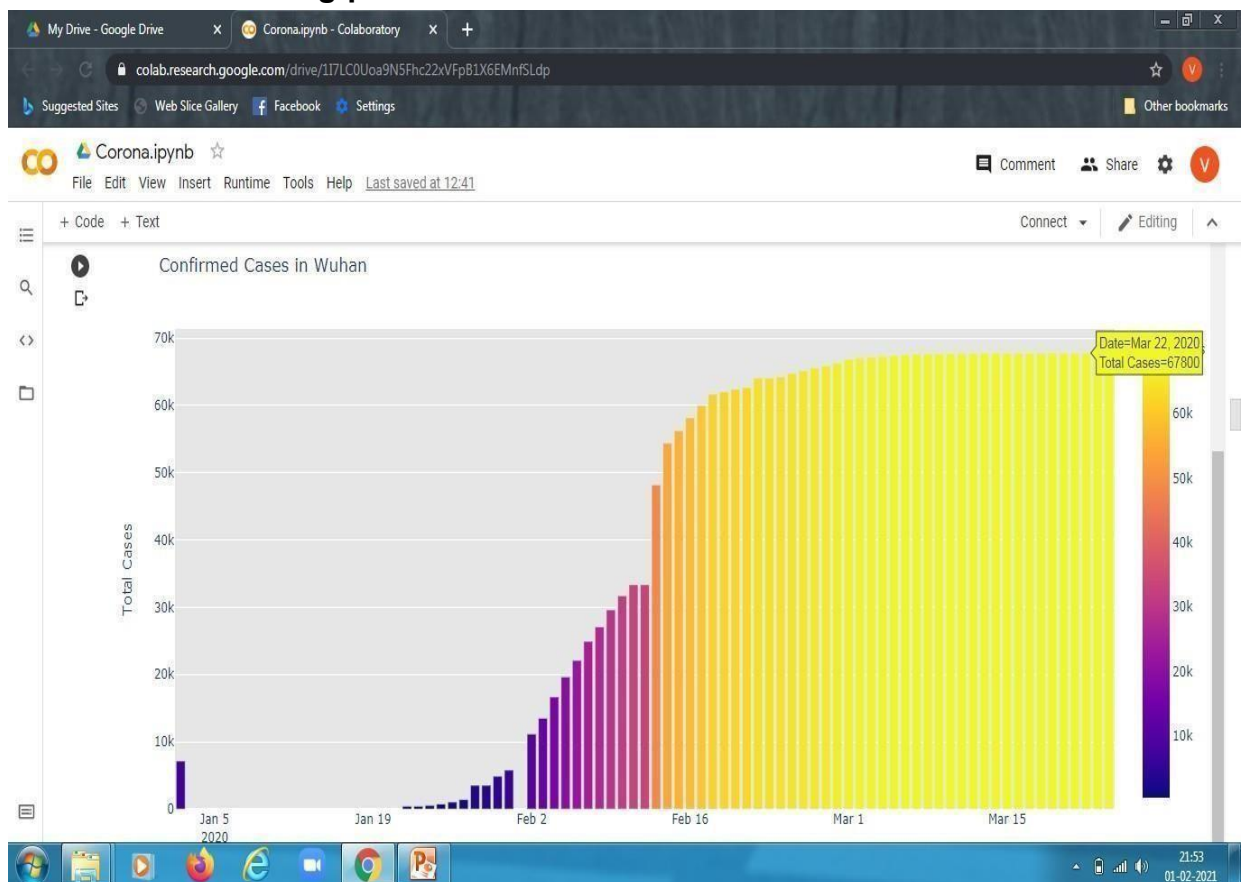


Fig-q:confirmed cases in Wuhan

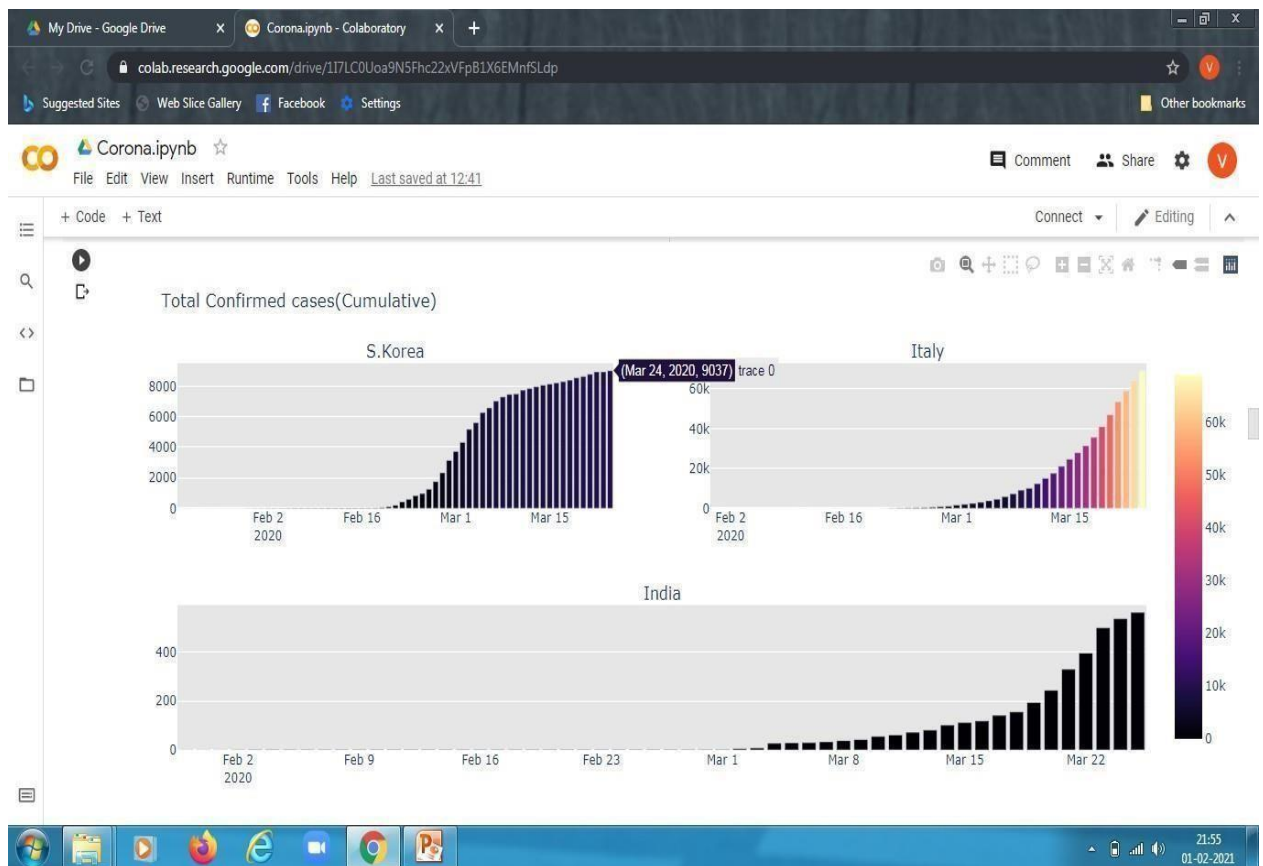


Fig-r: Cumulative cases

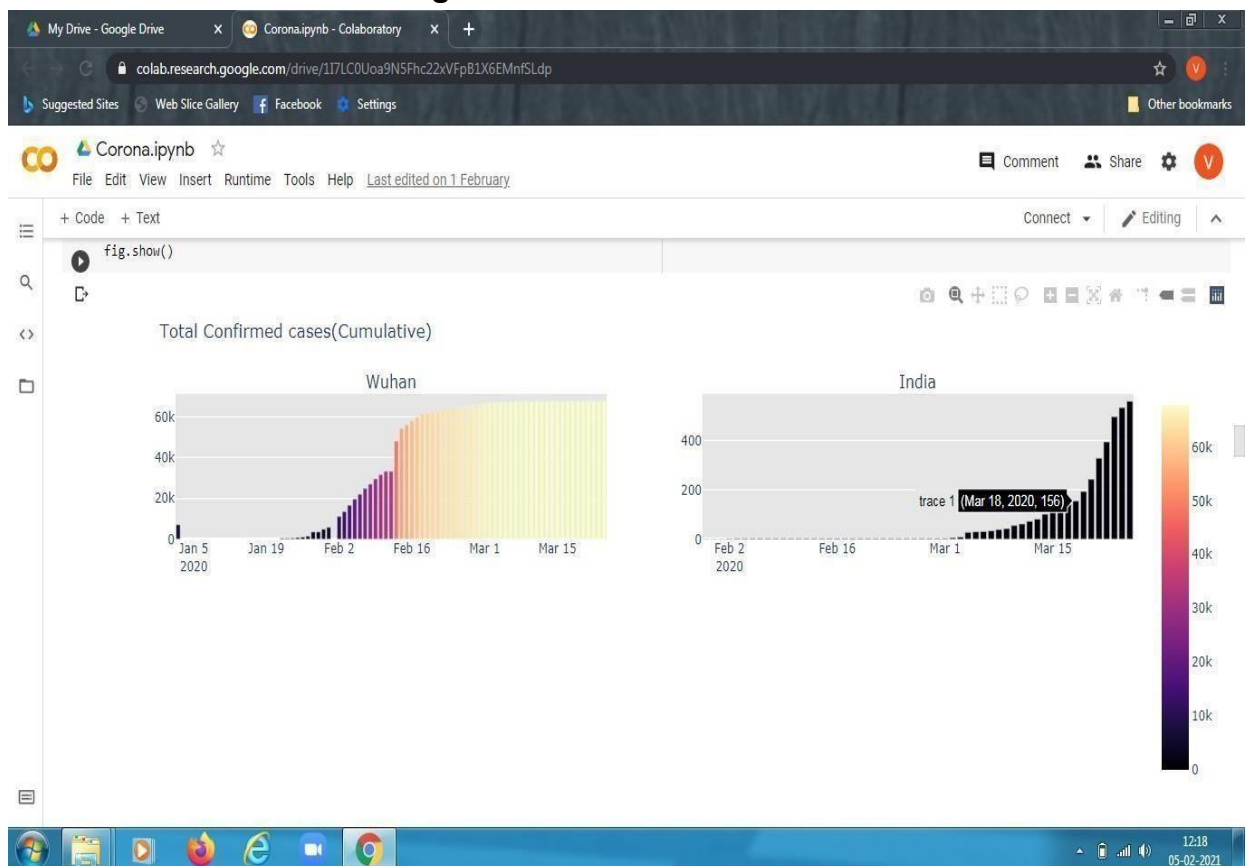


Fig-s: Cumulative cases

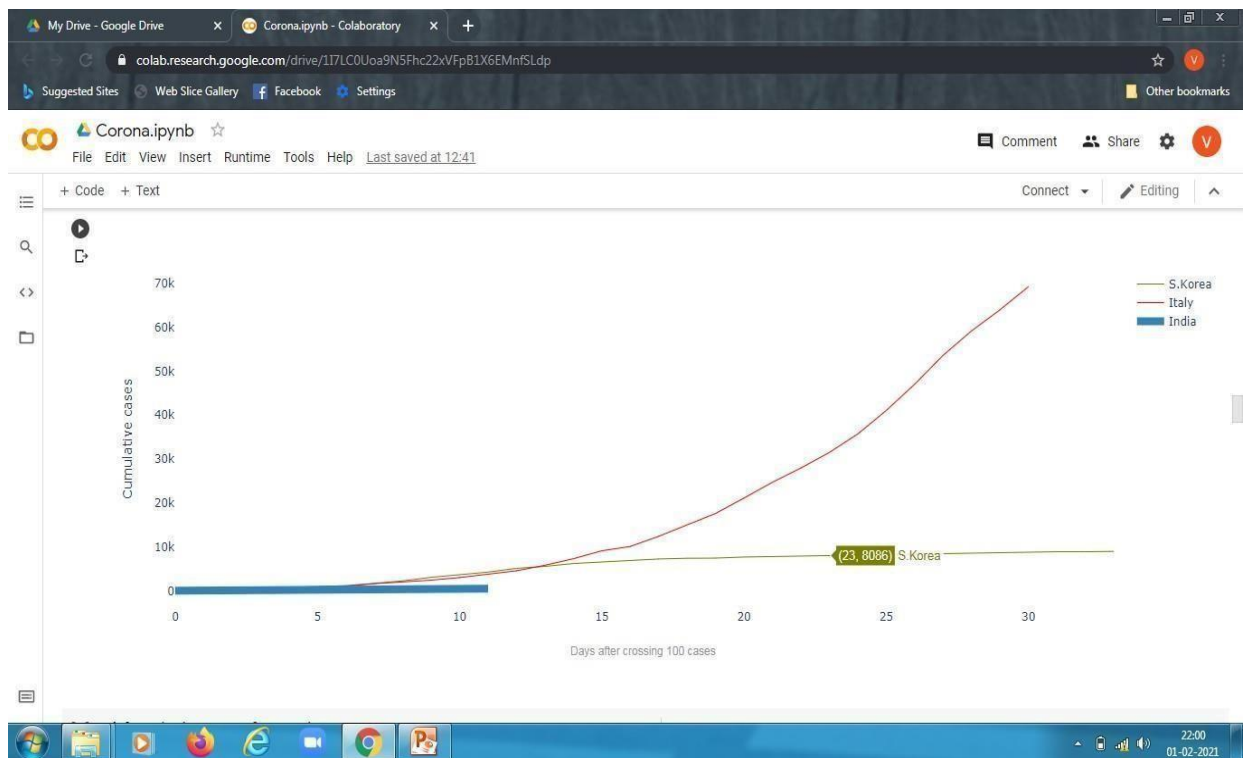


Fig-t: Total cases after crossing 100 days

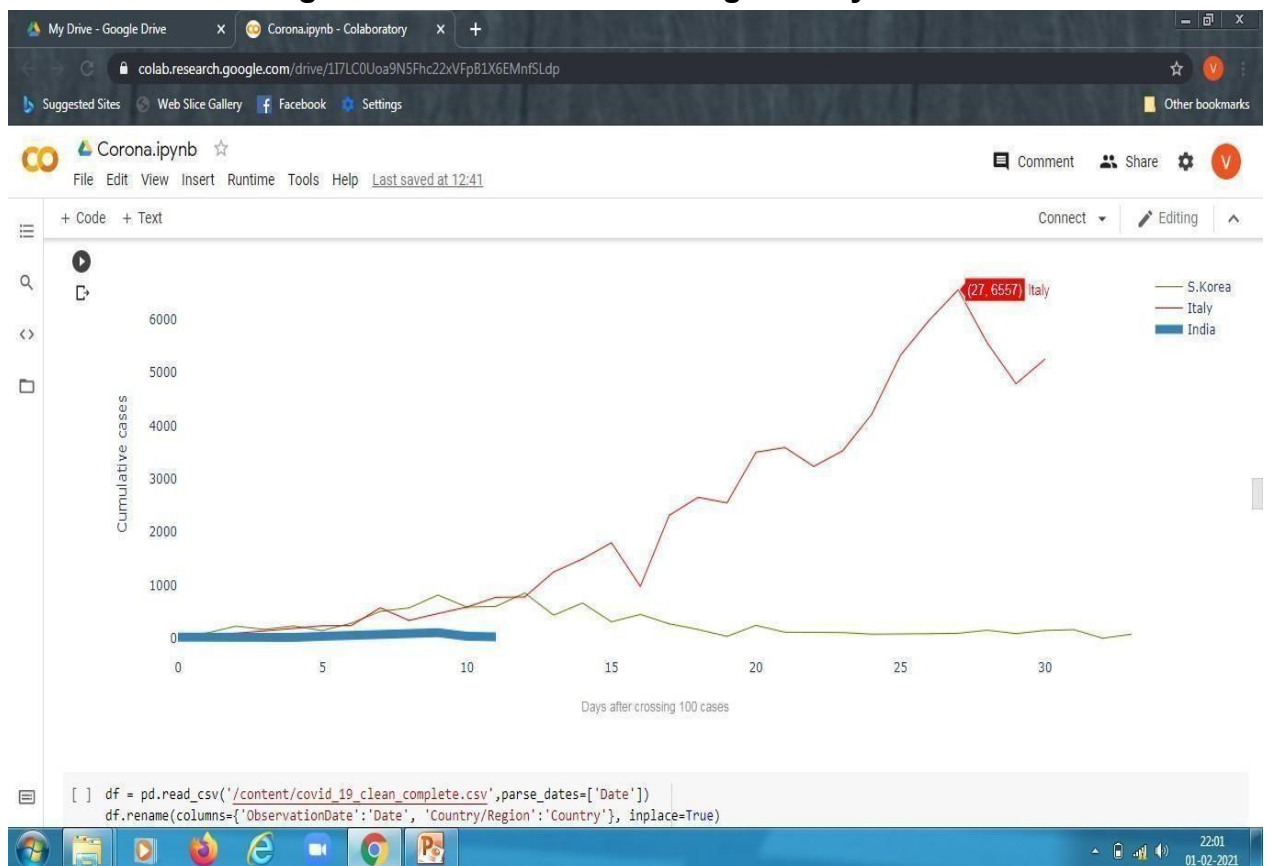


Fig-u: New cases after crossing 100 days

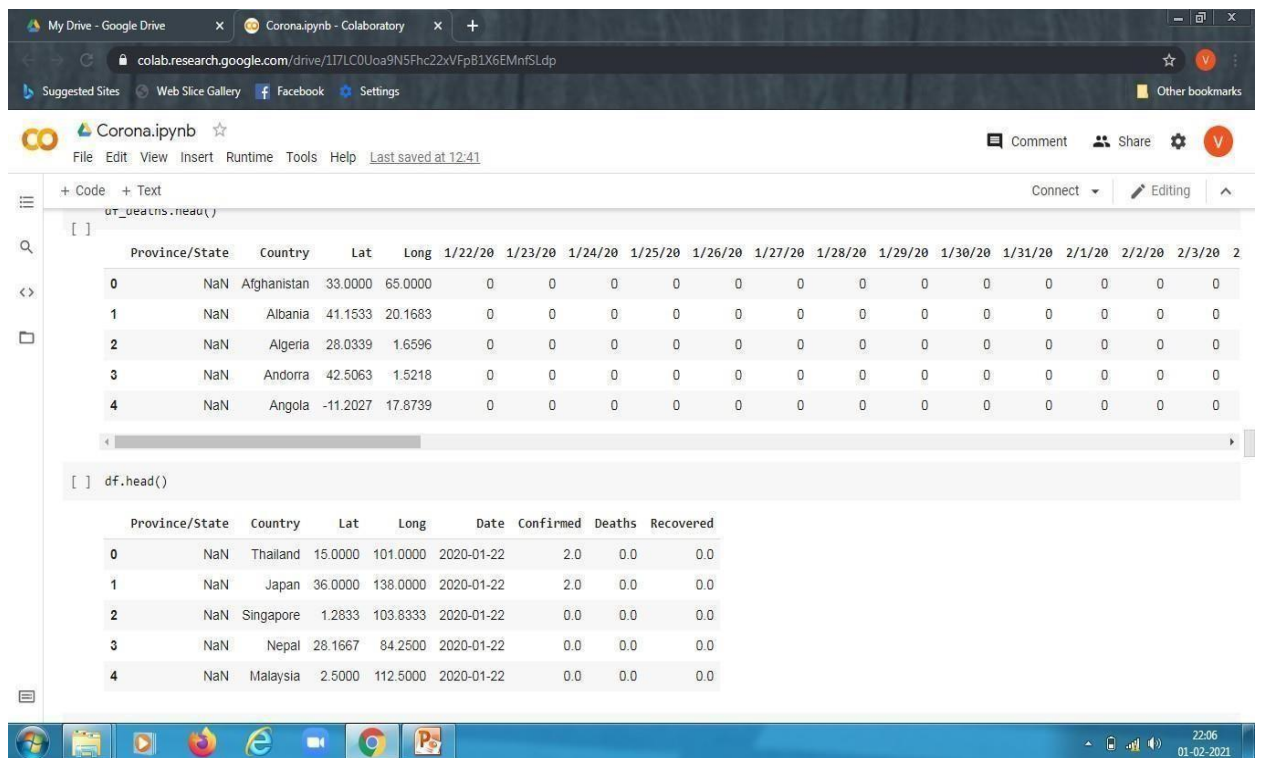


Fig-v:Deaths dataset

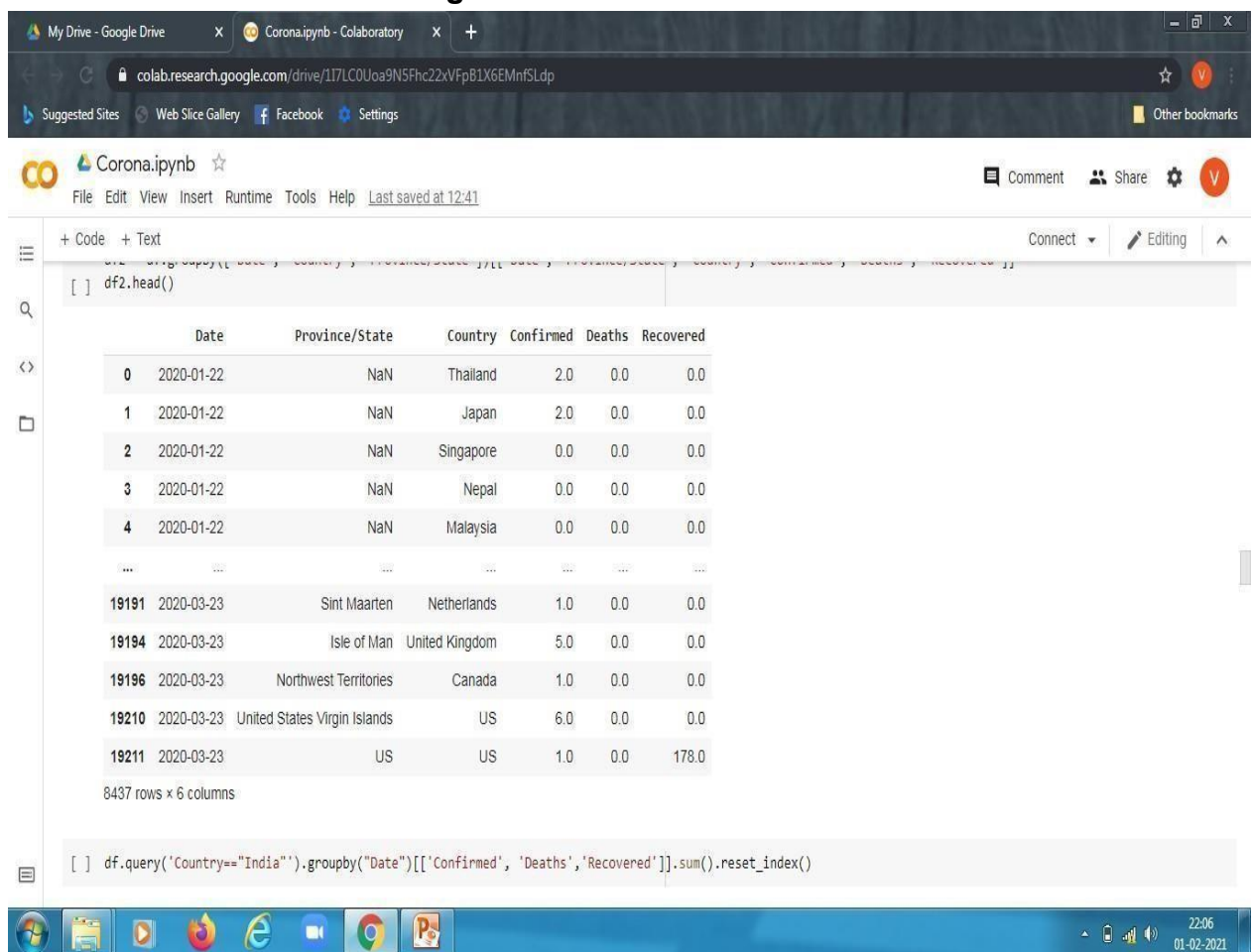


Fig-w:per day cases dataset

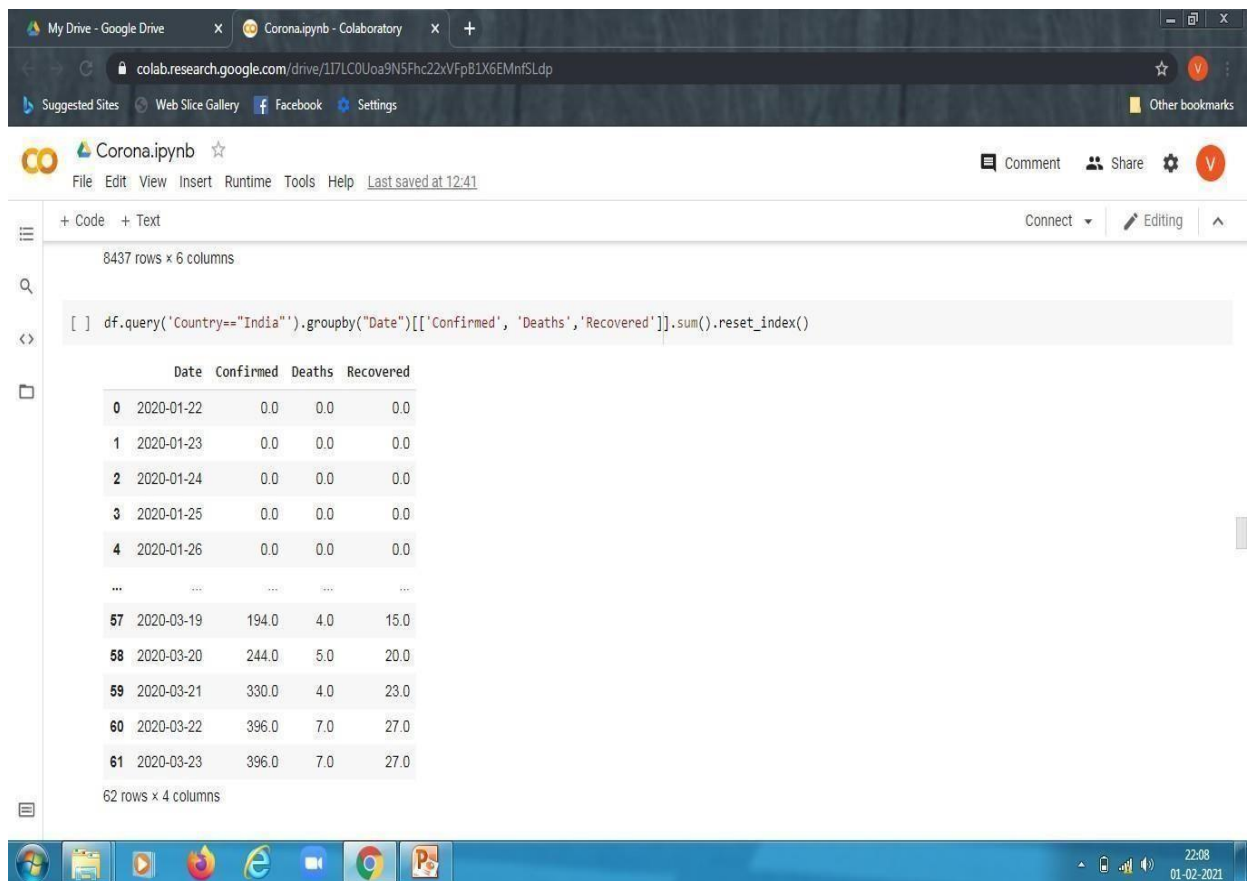


Fig-x:India per day dataset

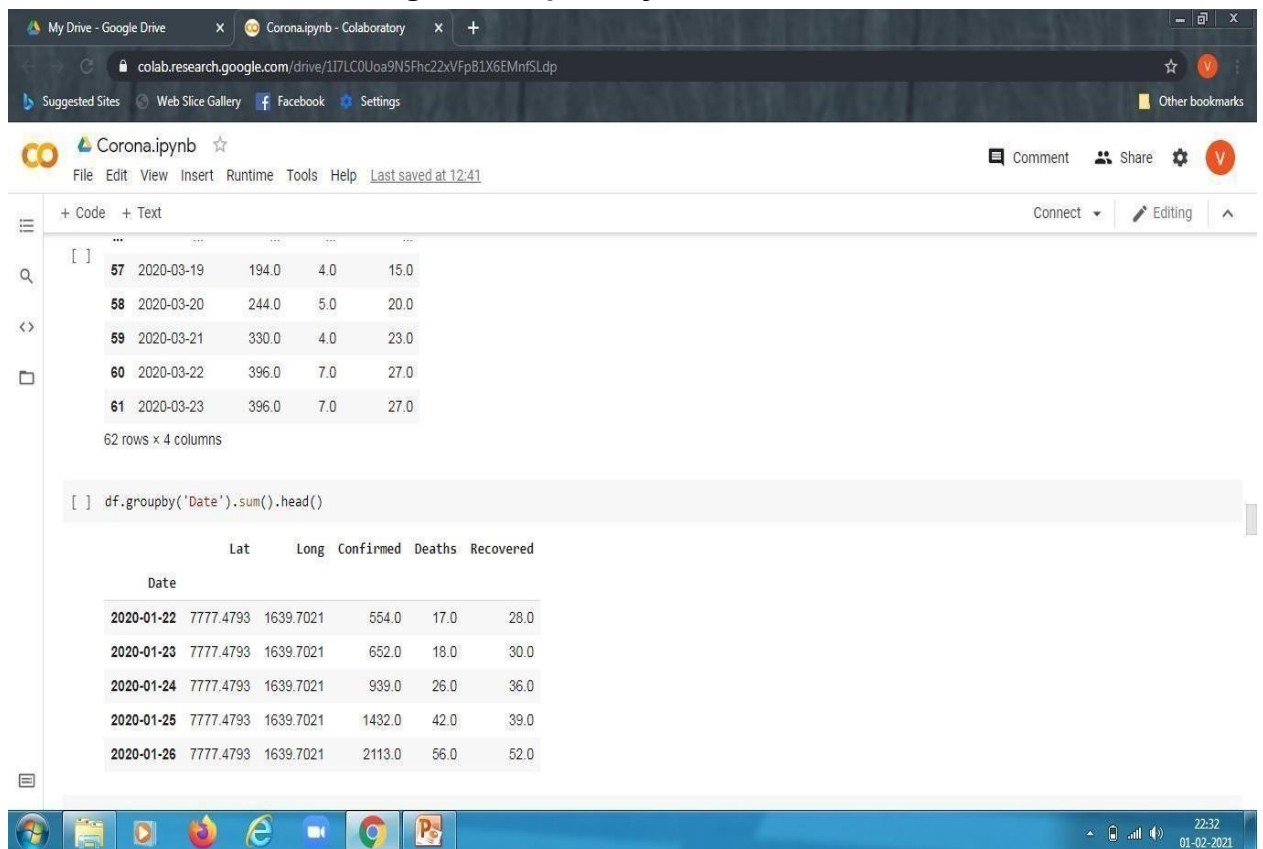


Fig-y:New cases dataset



Fig-z: Worldwide novel coronavirus cases

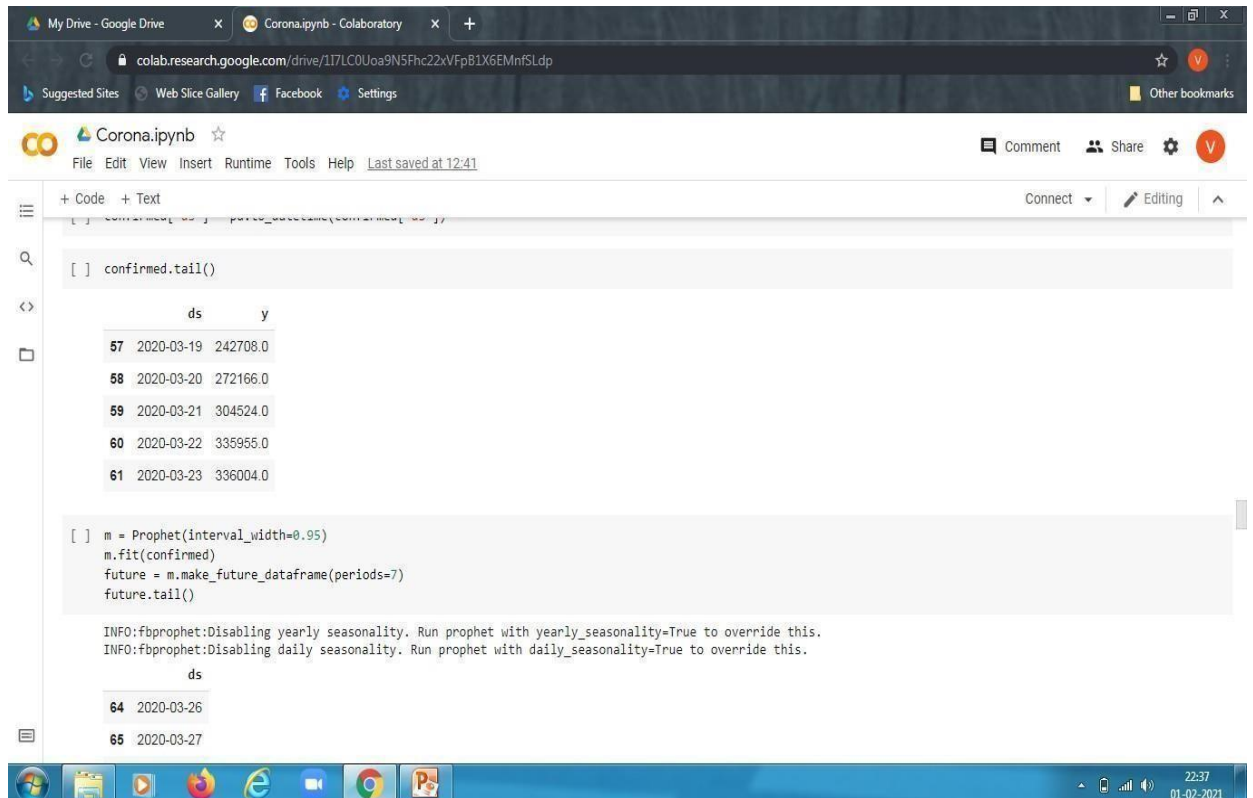


Fig-A: Confirmed cases datasets

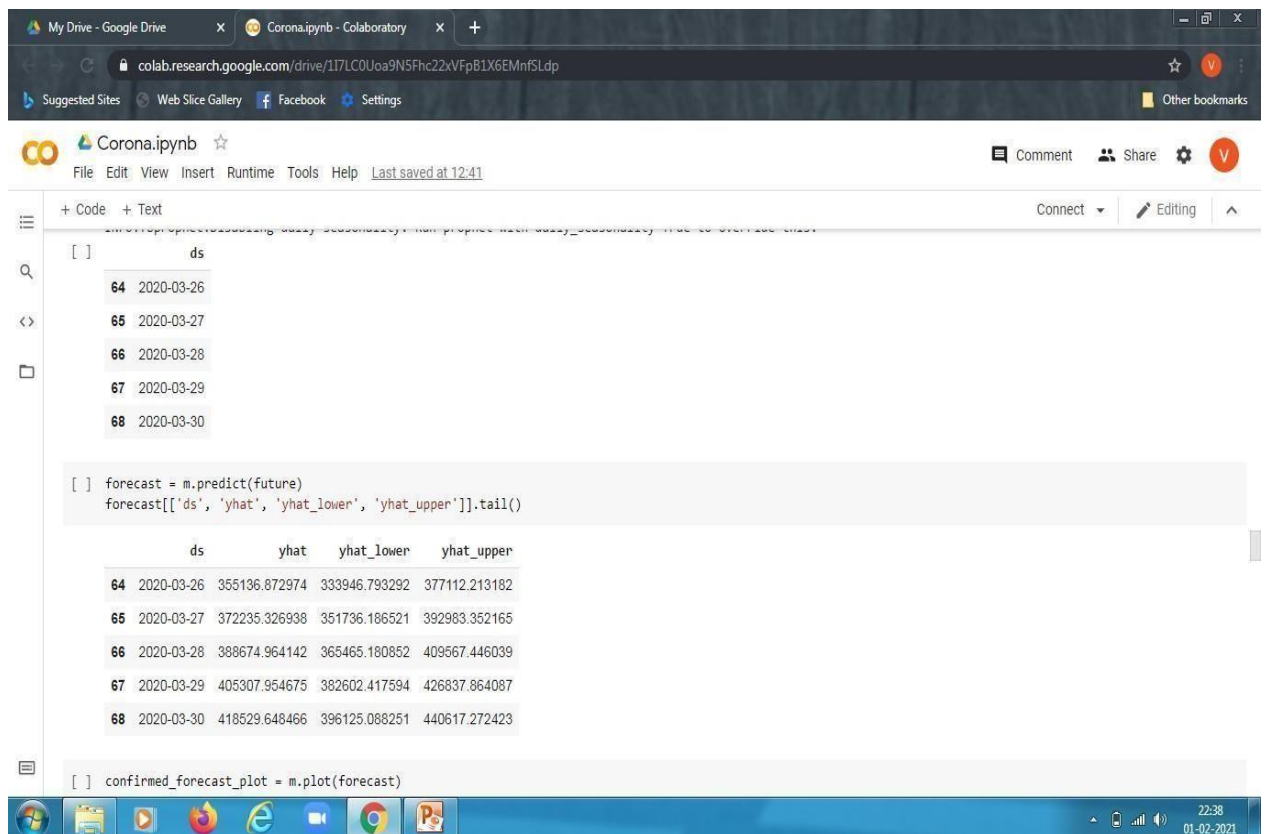


Fig-B:Confirmed cases data analysis

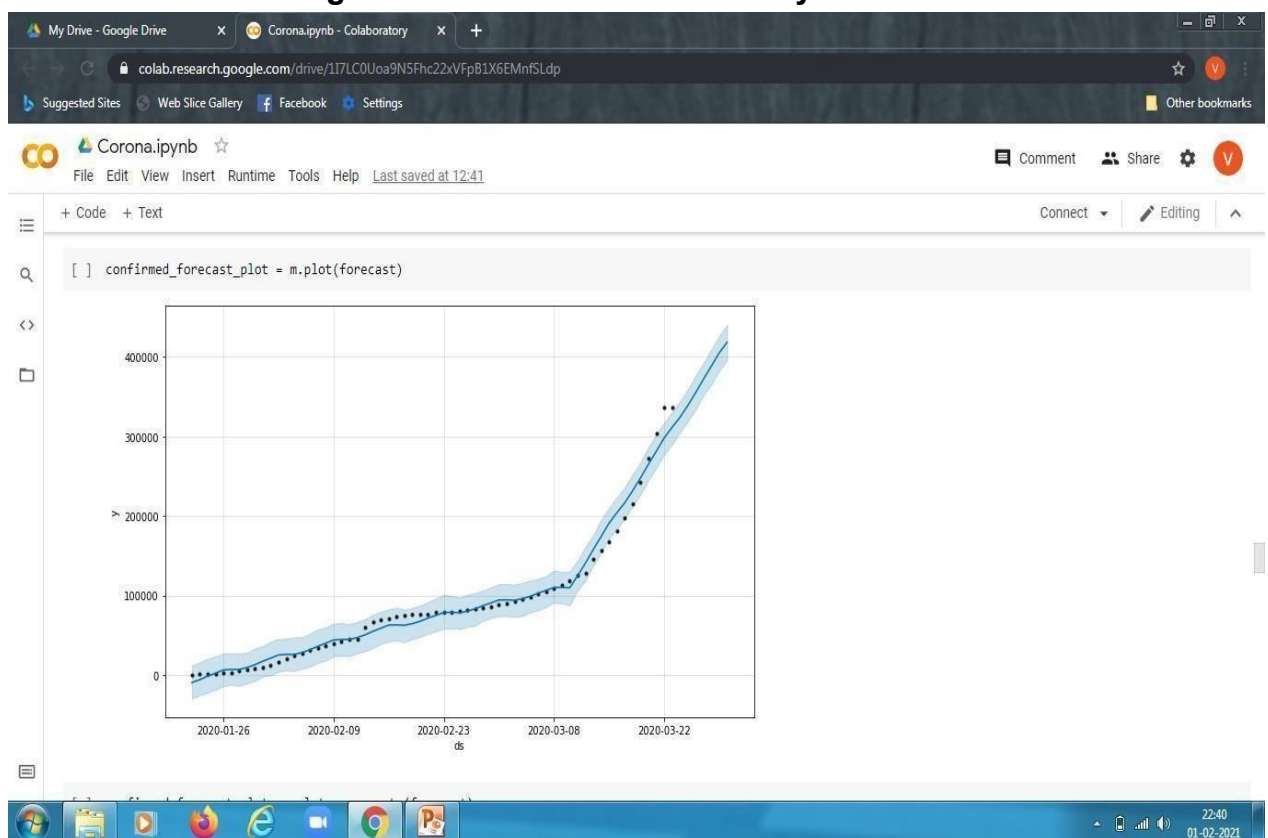


Fig-C:Confirmed cases prediction

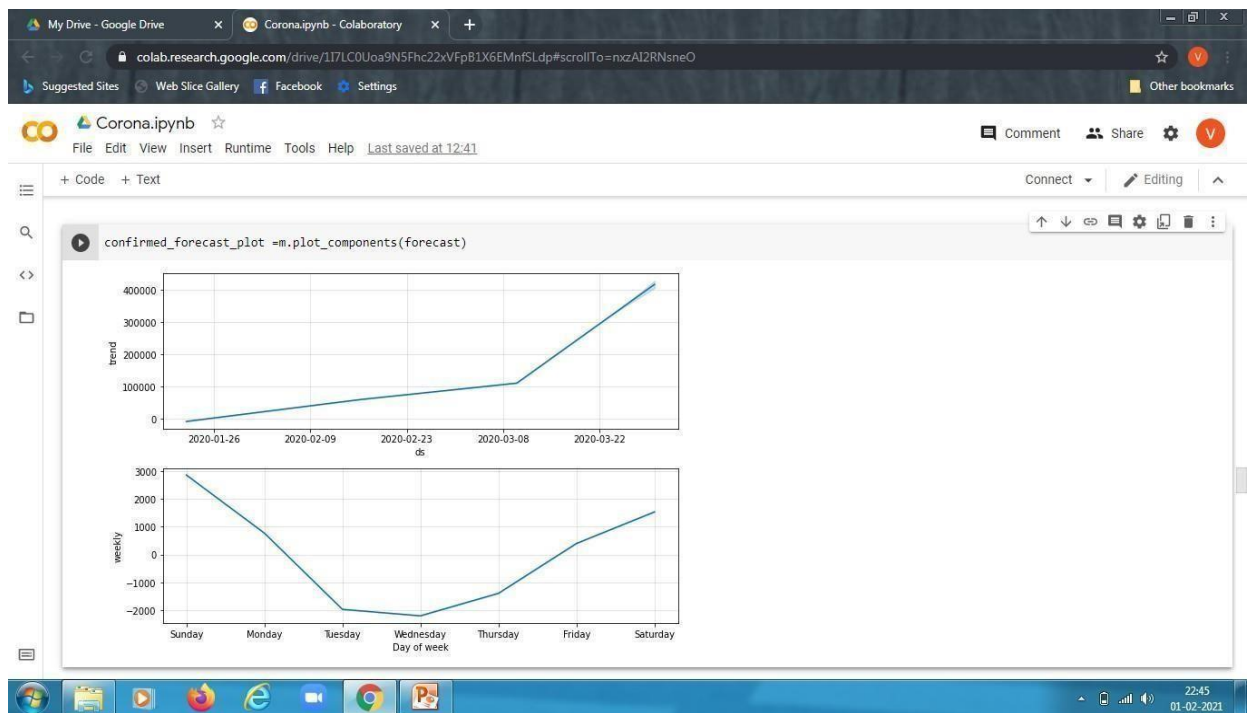


Fig-D: Confirmed cases weekly prediction

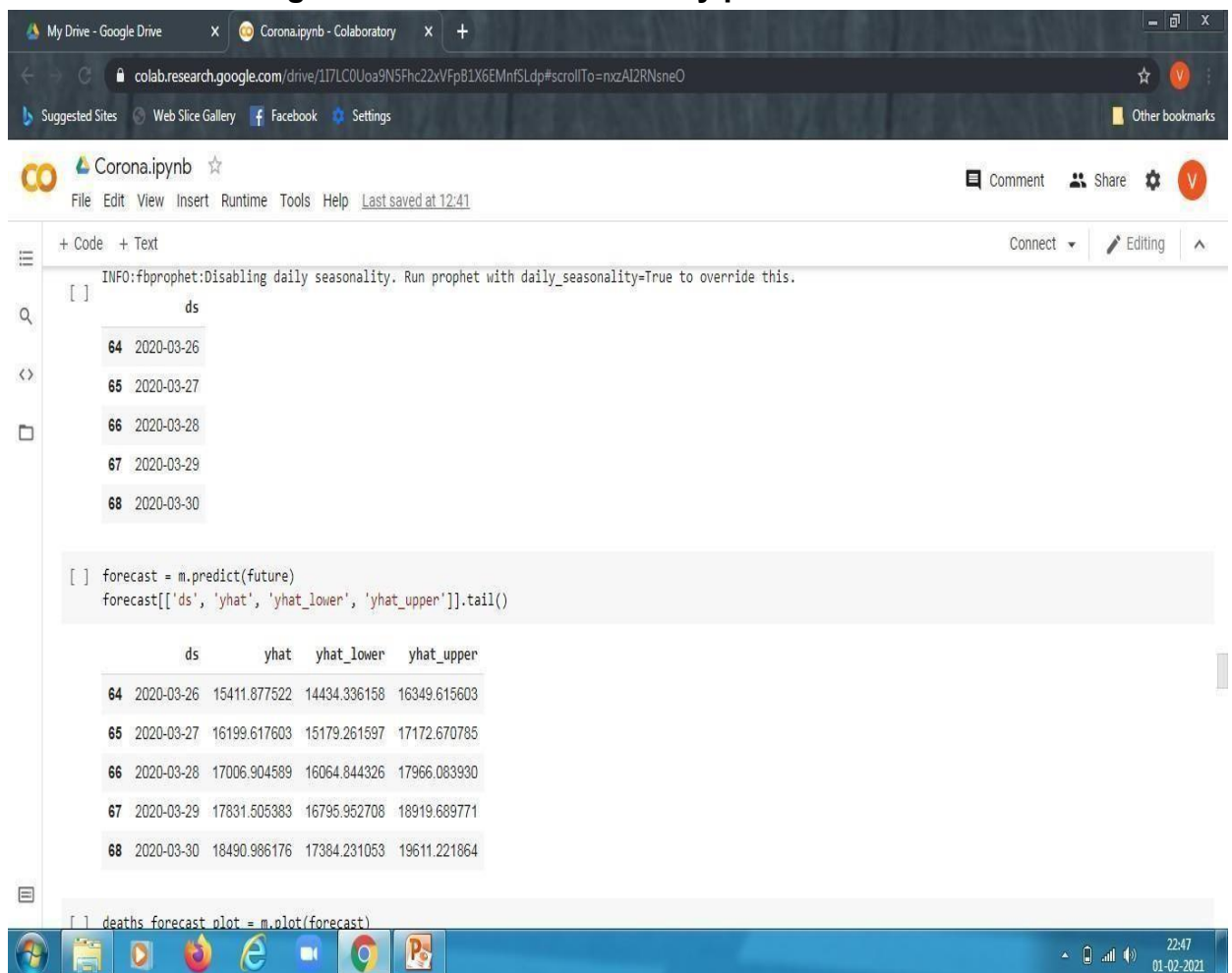


Fig-E: Death cases dataset and analysis

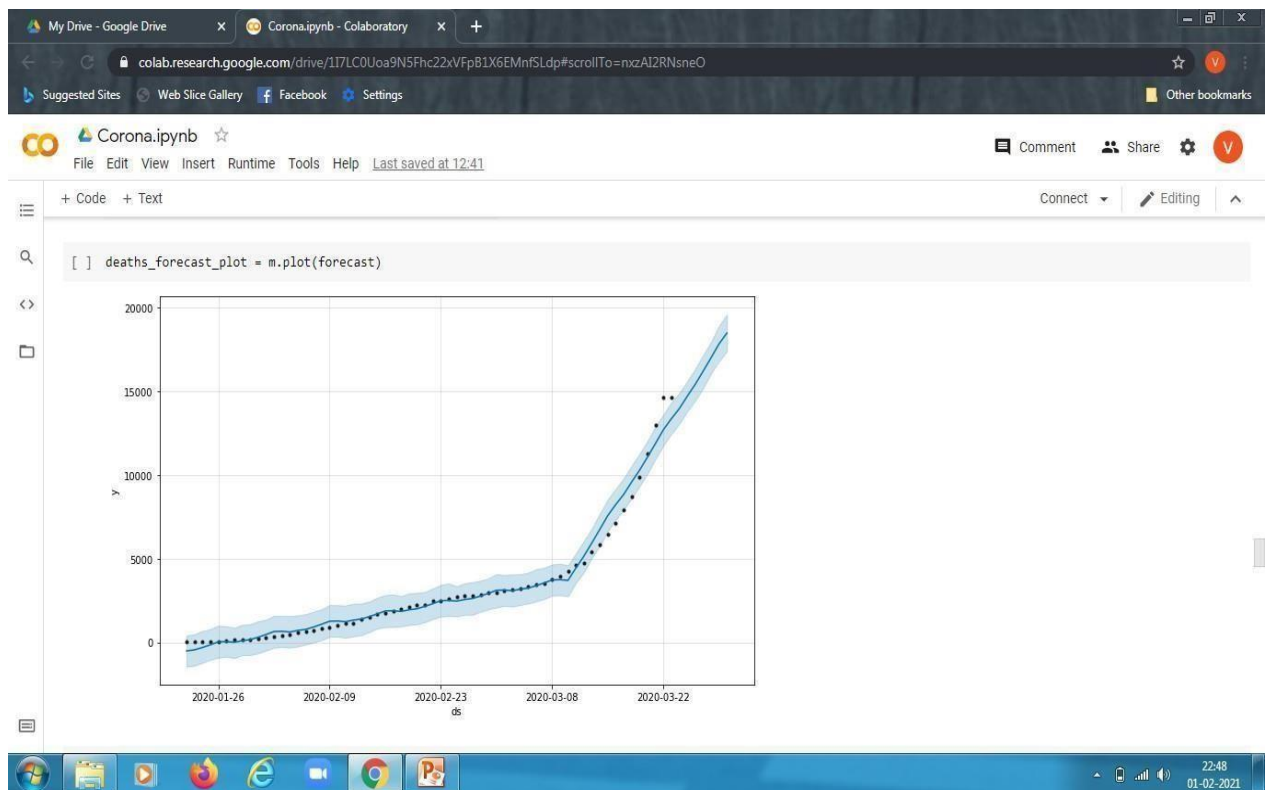


Fig-F:Death cases prediction

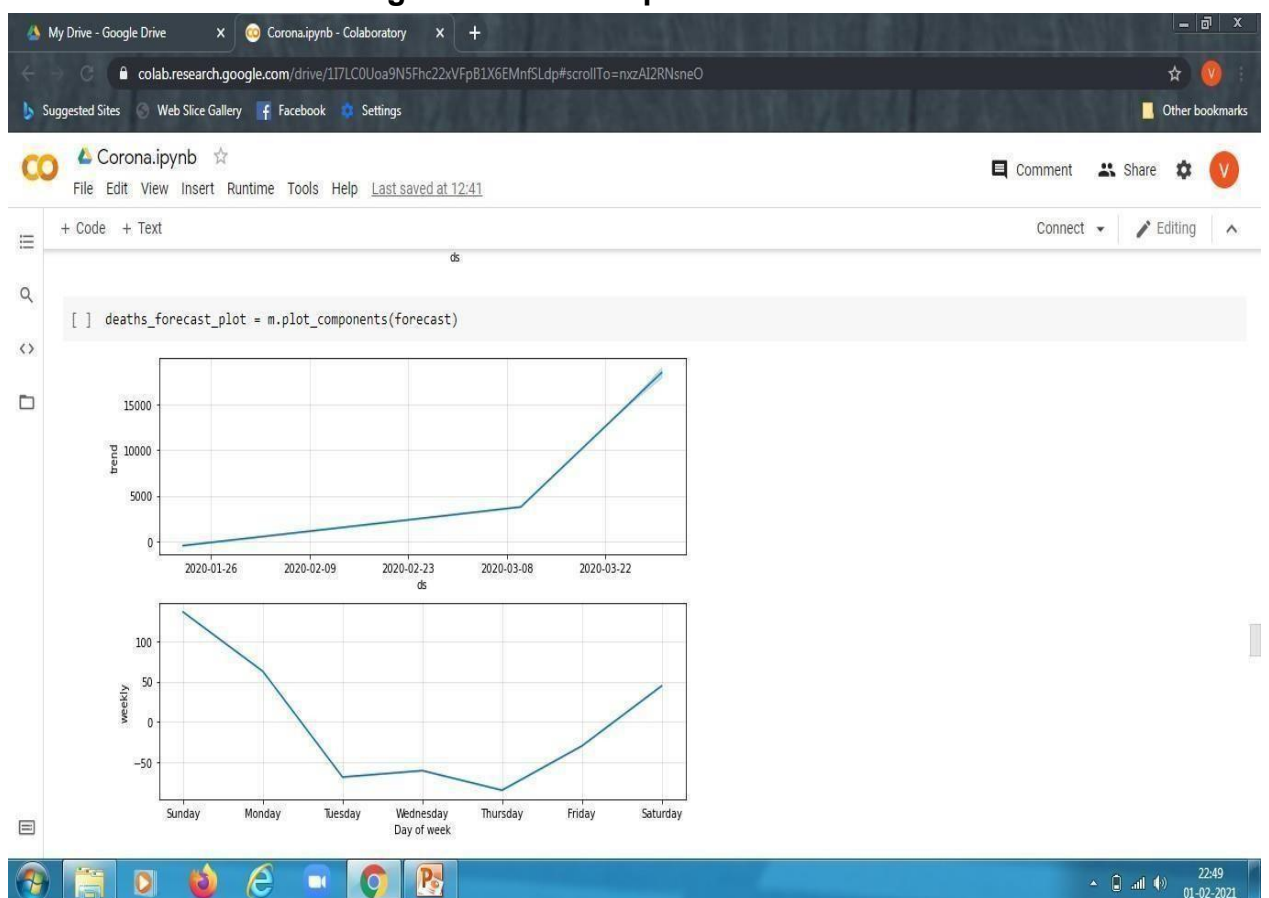


Fig-G:Death cases weekly forecast

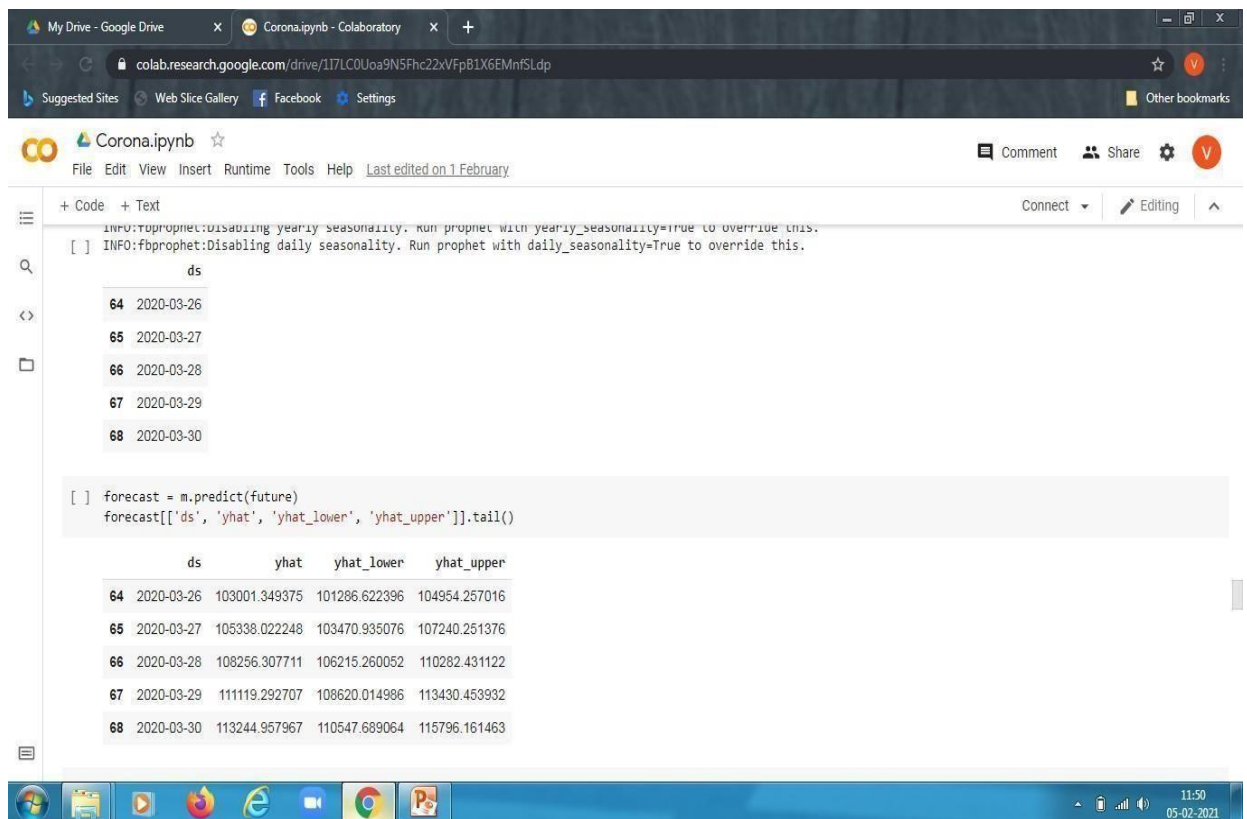


Fig-H:Recovered cases dataset and analysis

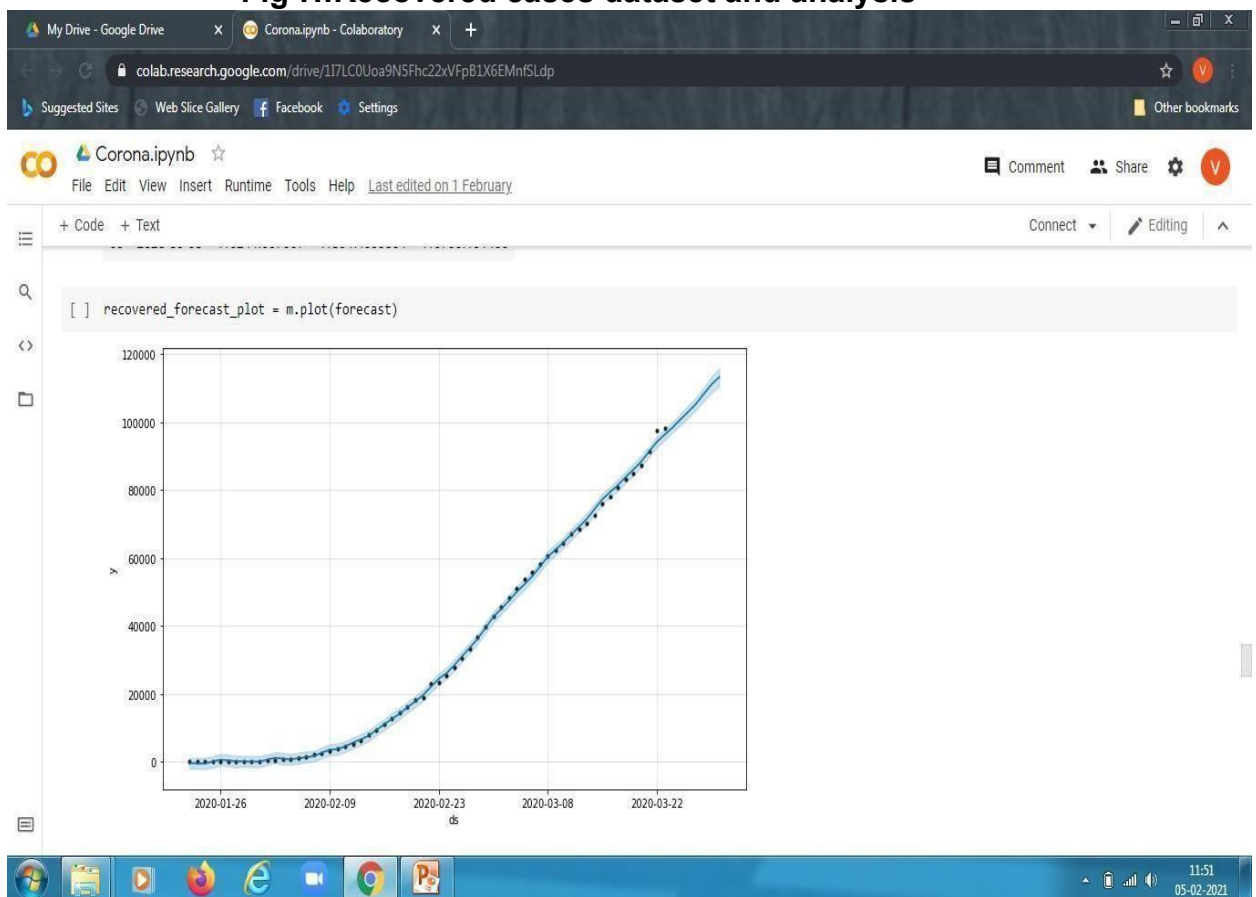


Fig-I:Recovered cases forecast

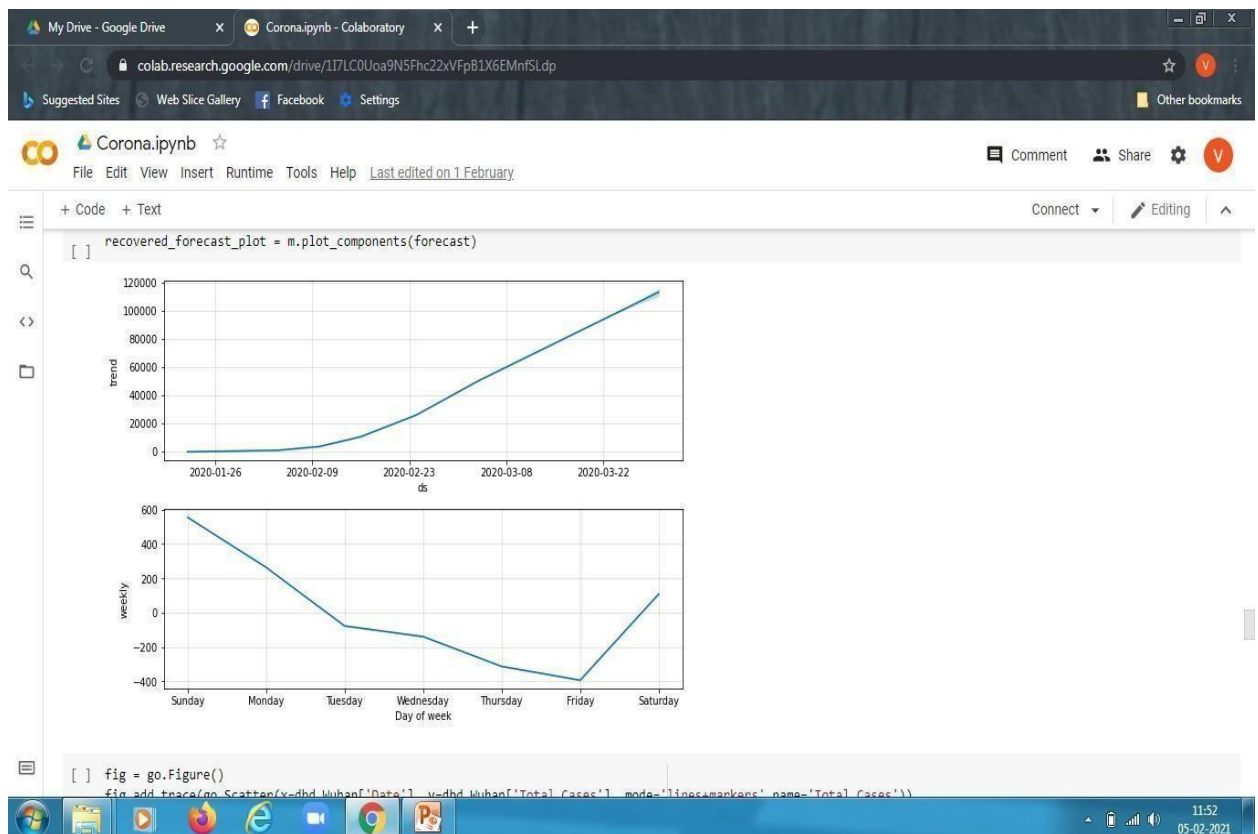


Fig-J:Recovered cases weekly forecast

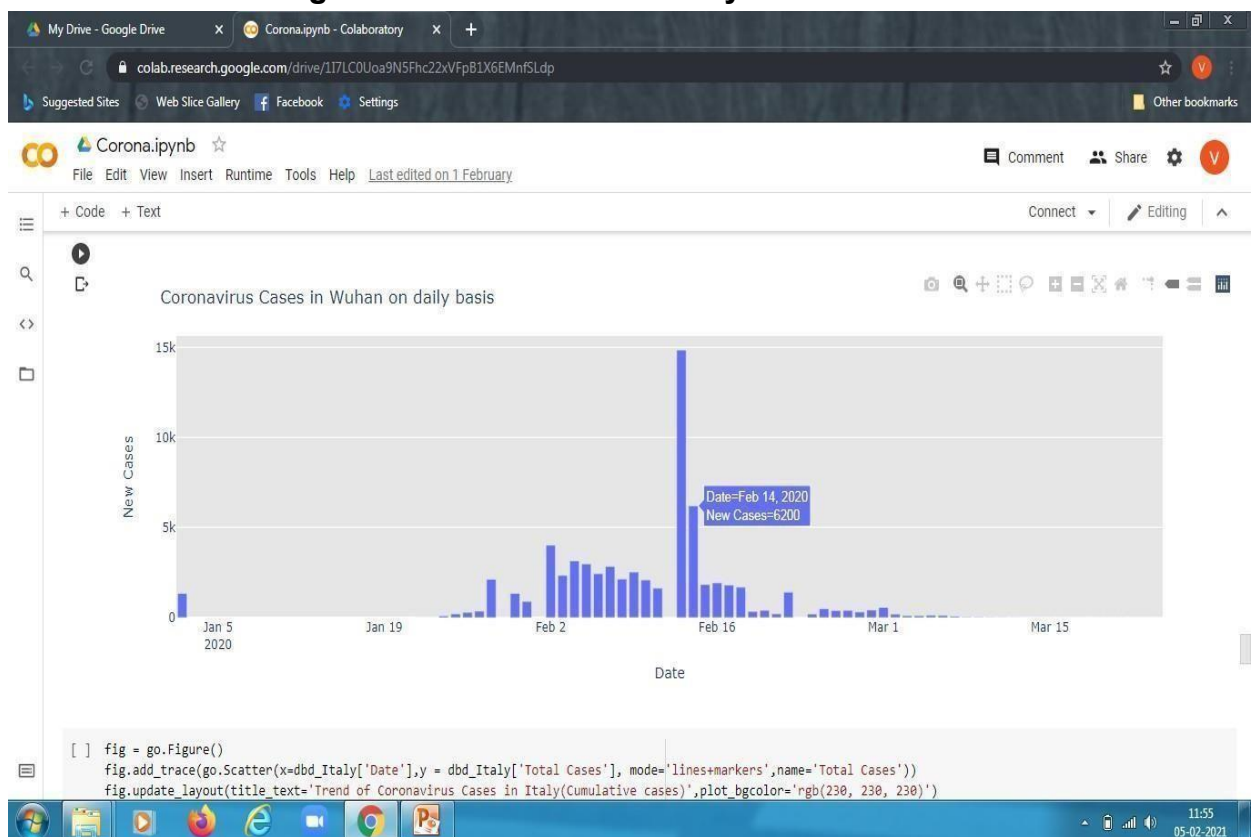


Fig-K:Trend of daily cases in Wuhan



Fig-L:Trend of coronavirus cases in Wuhan



Fig-M:Trend of coronavirus cases in Italy

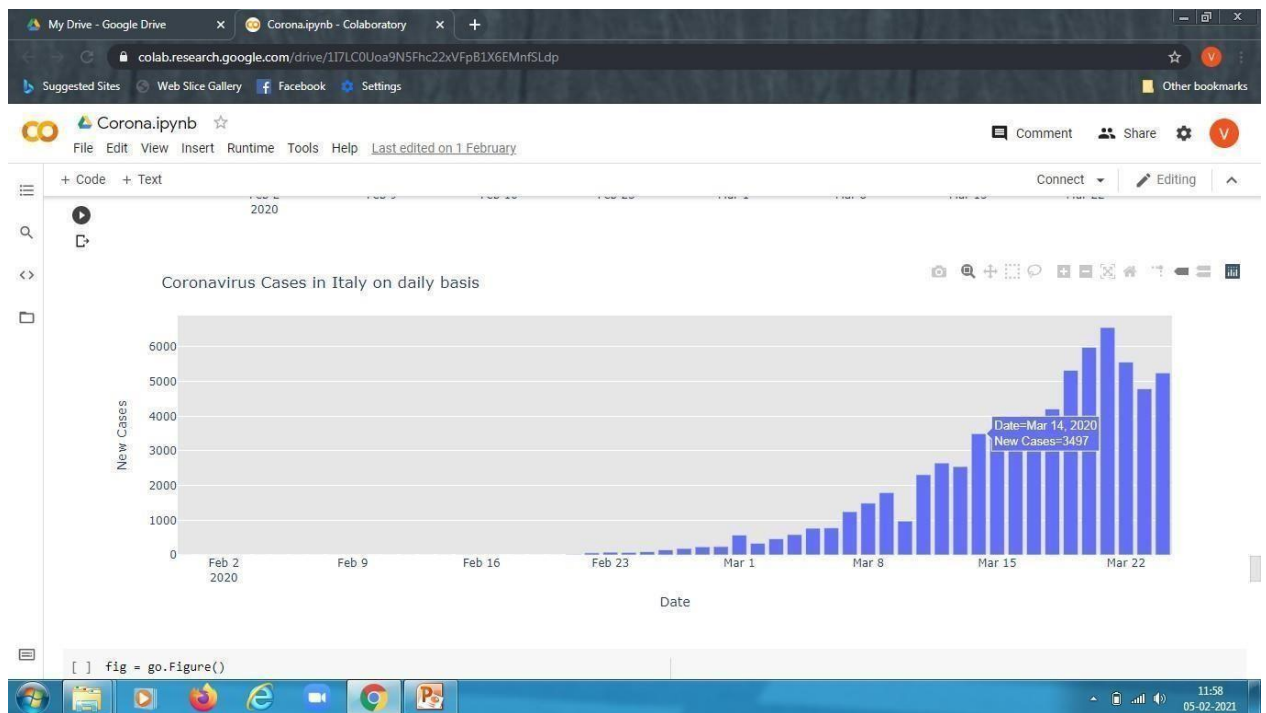


Fig-N:Trend of daily cases in Italy

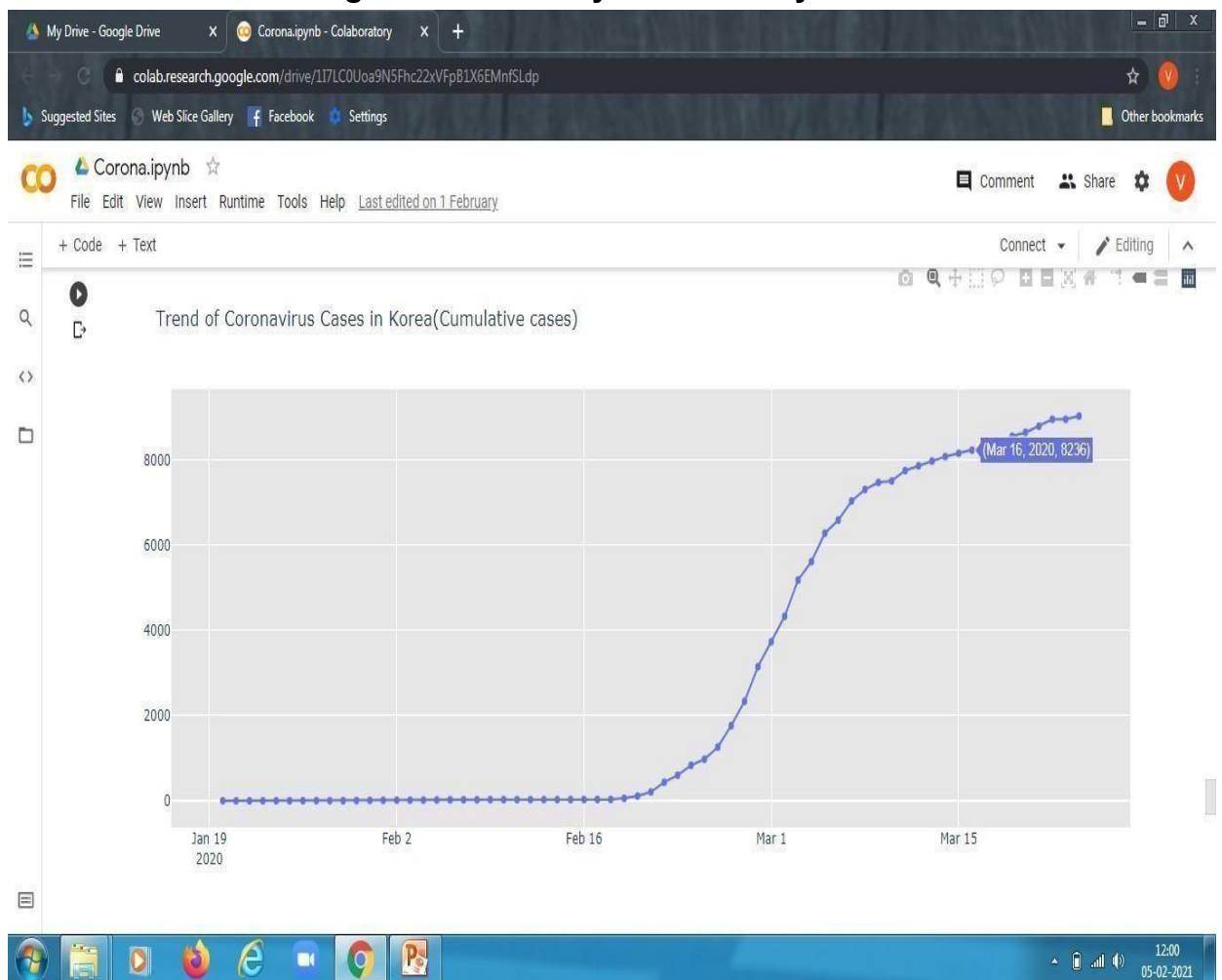


Fig-O:Trend of coronavirus cases in South Korea

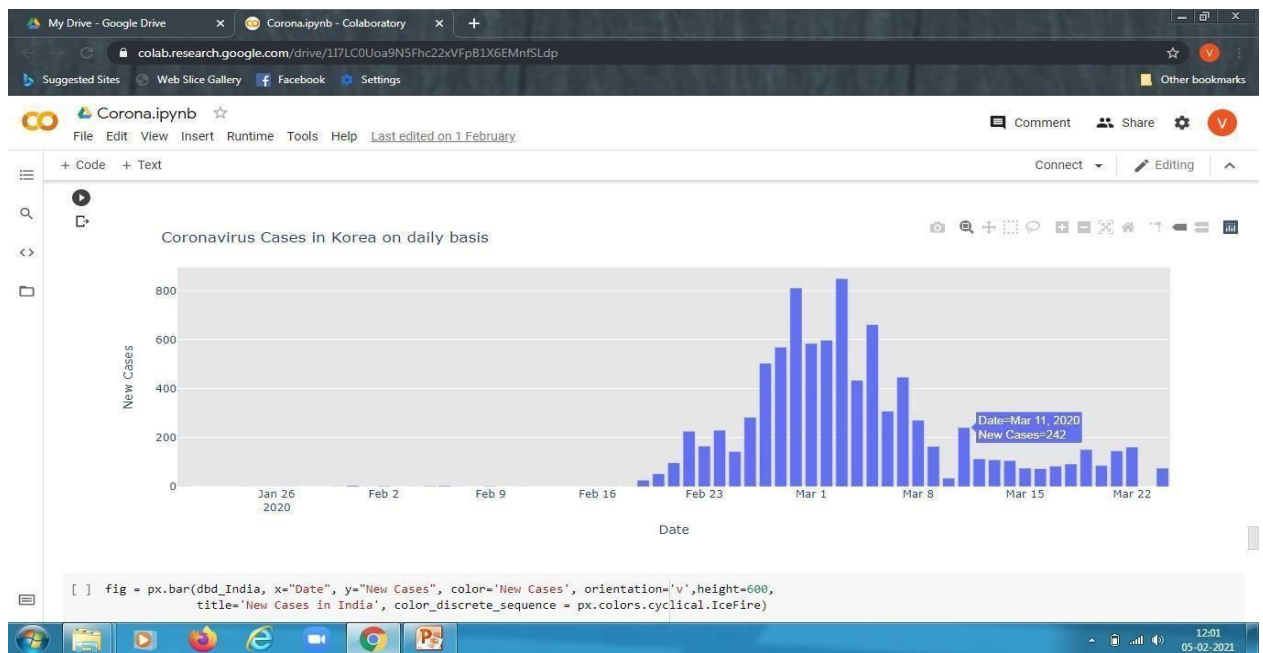


Fig-P:Trend of daily cases in South Korea

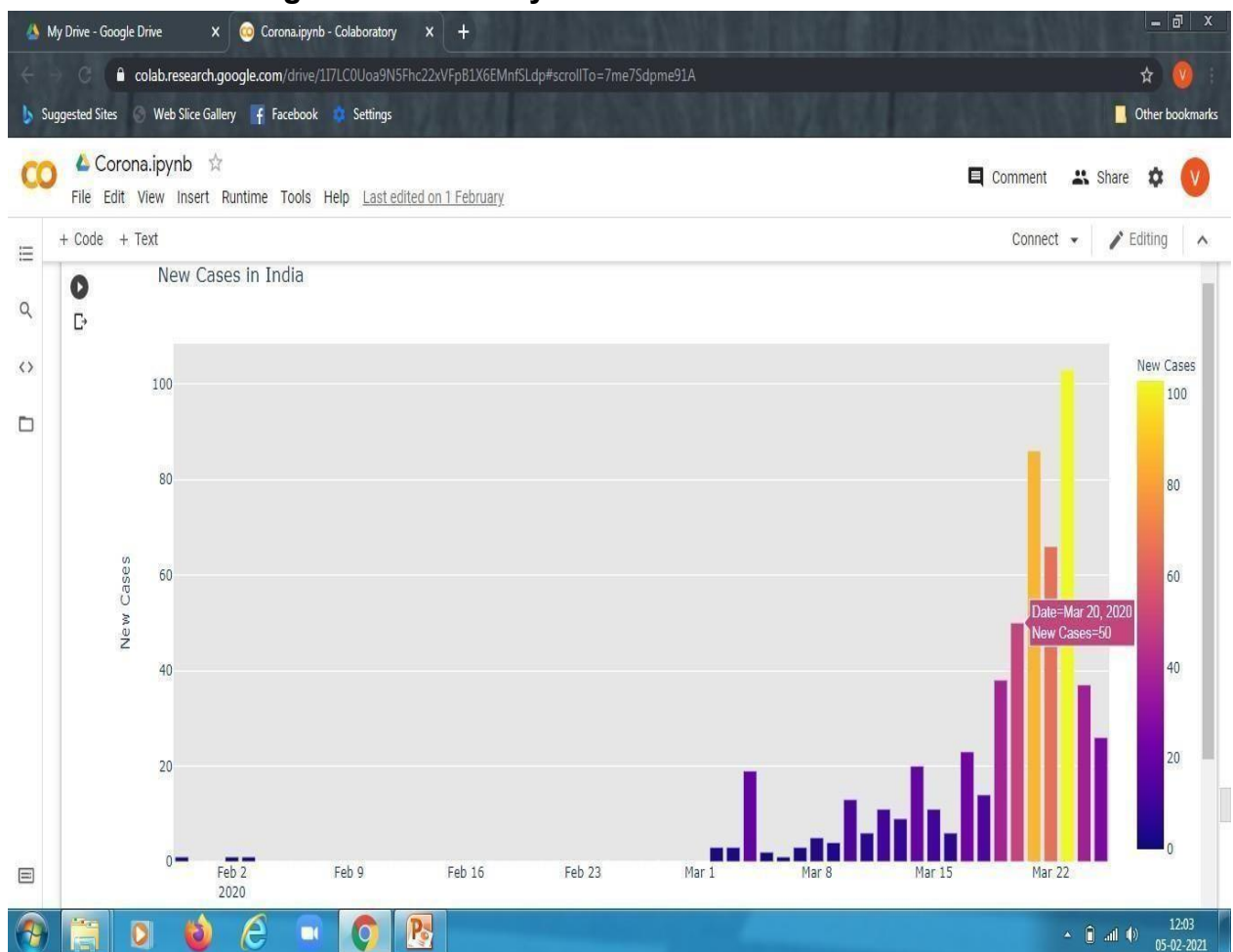


Fig-Q:New cases in India

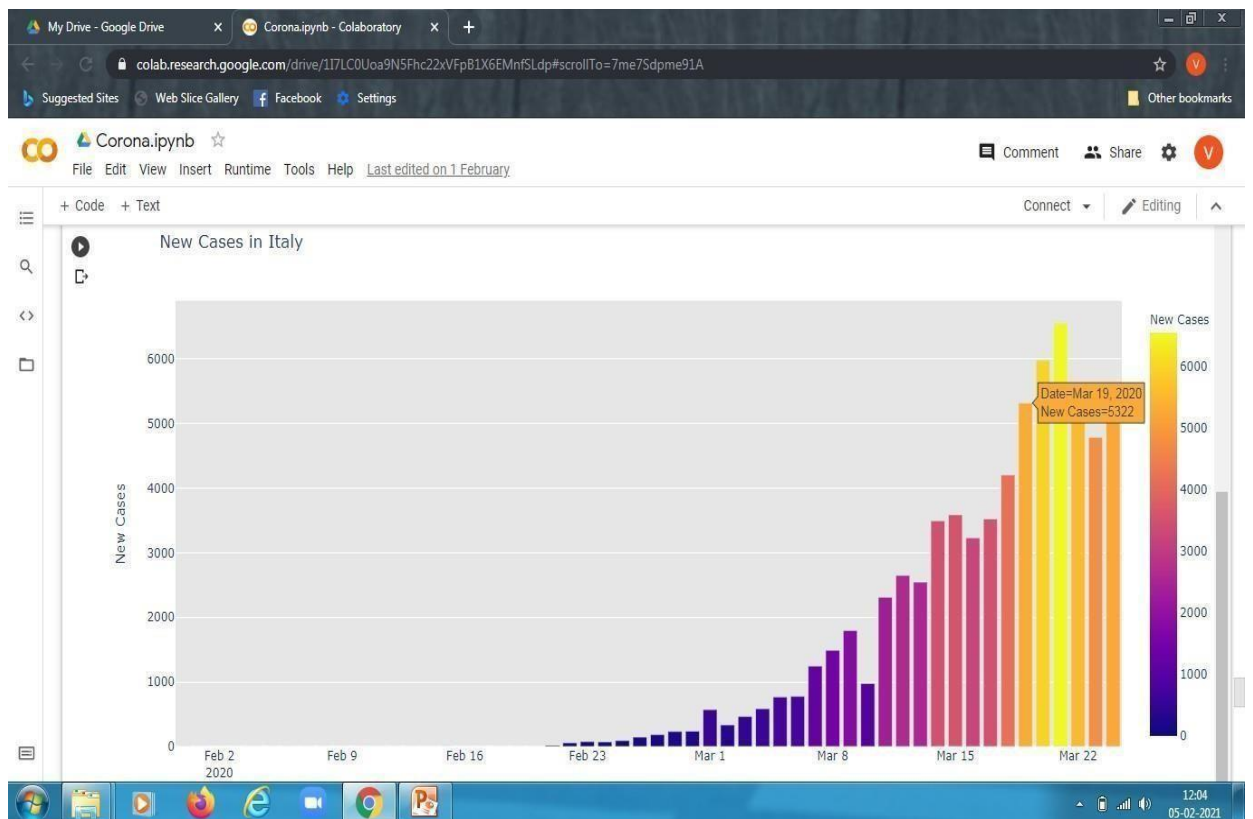


Fig-R:New cases in Italy

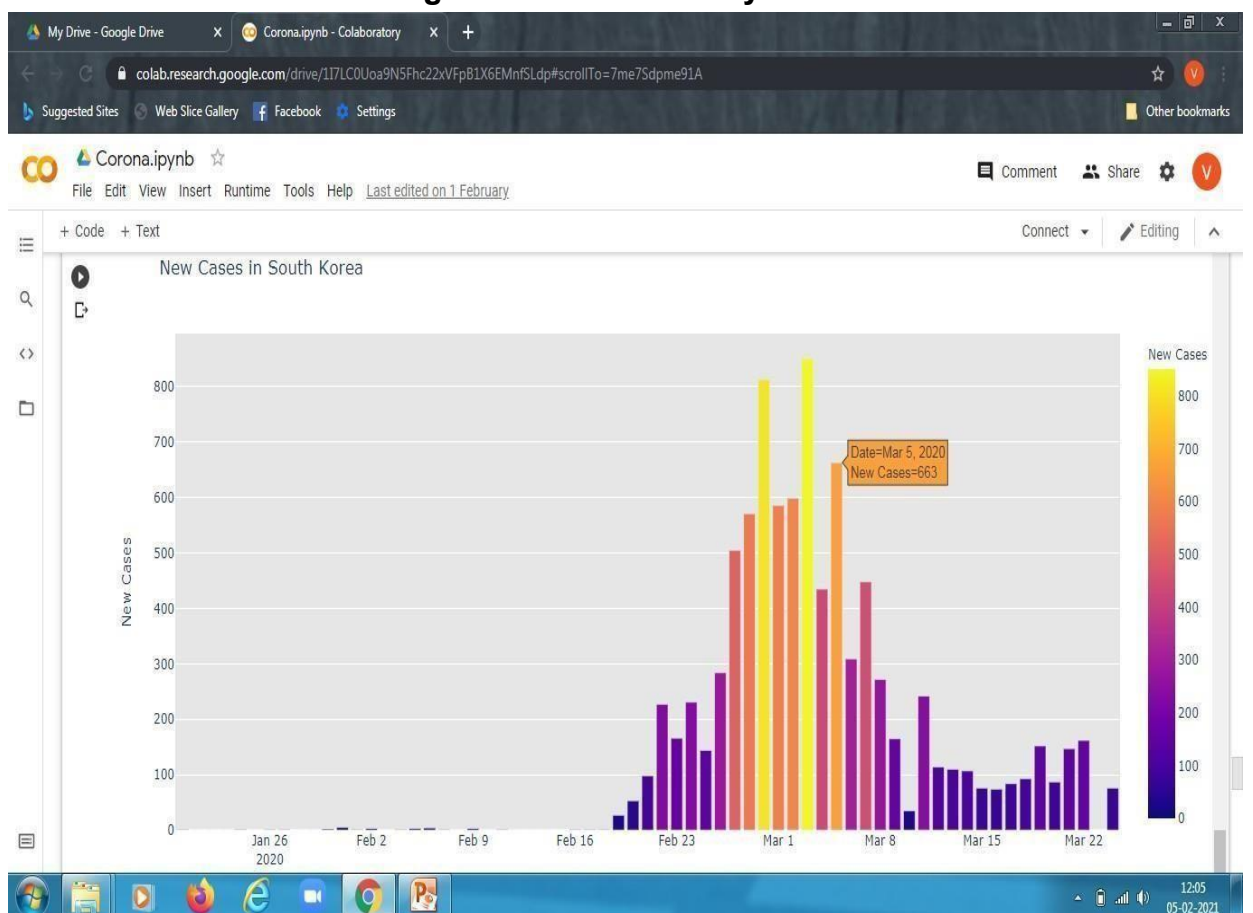


Fig-S:New cases in South Korea

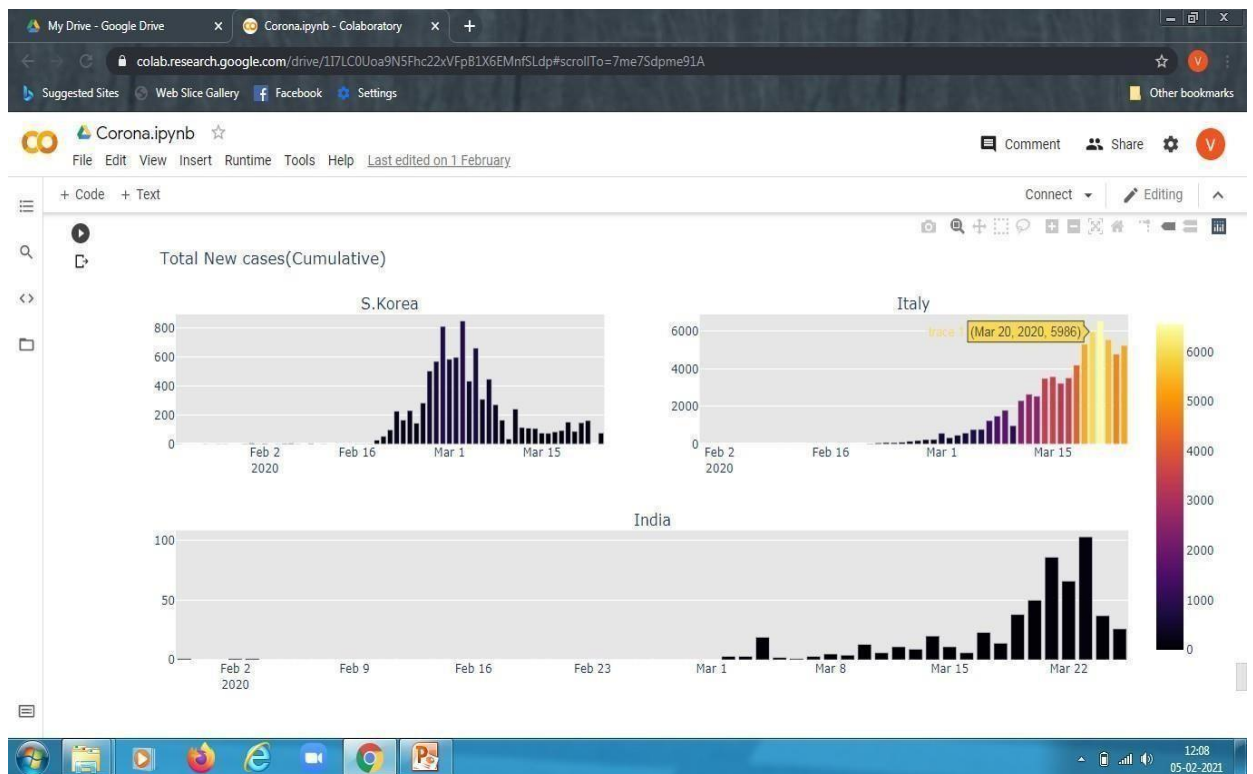


Fig-T:Cumulative new cases

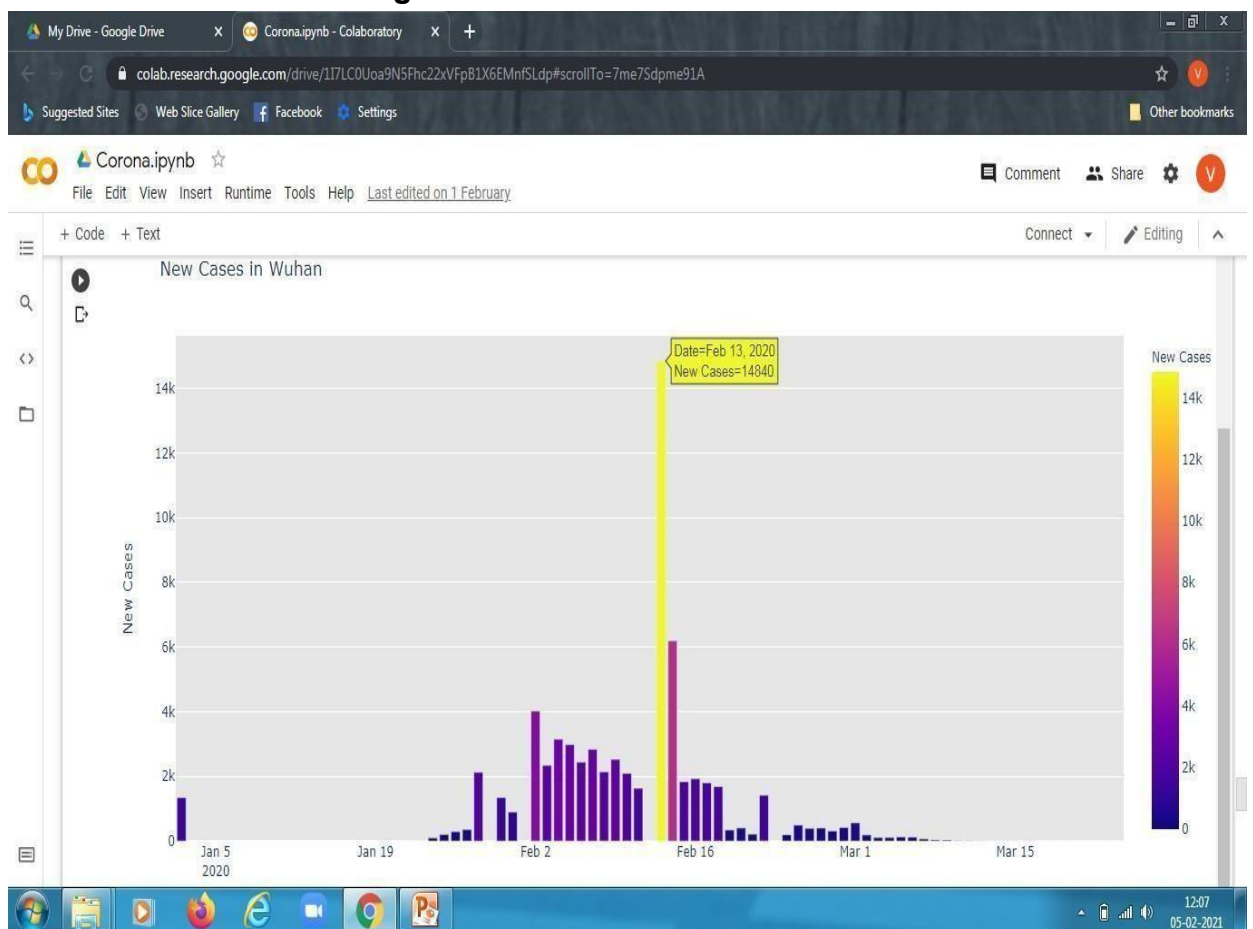


Fig-U:New cases in Wuhan

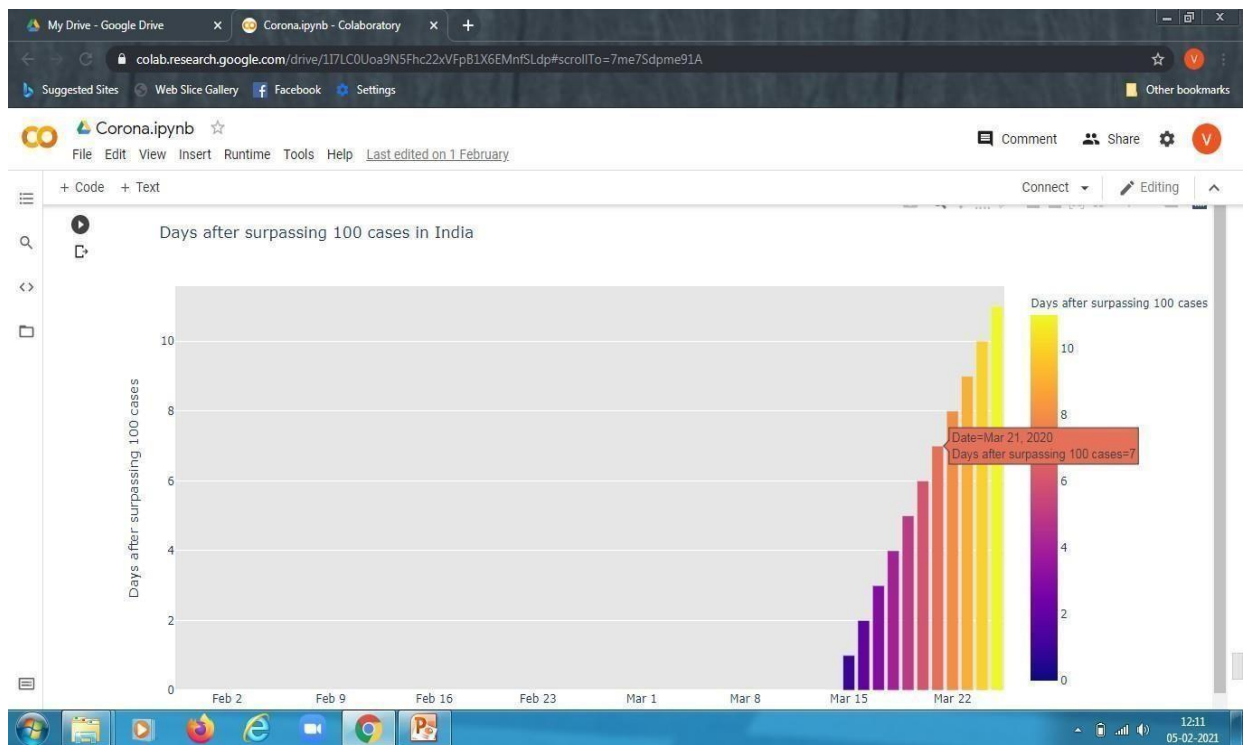


Fig-V:Cases after surpassing 100 in India

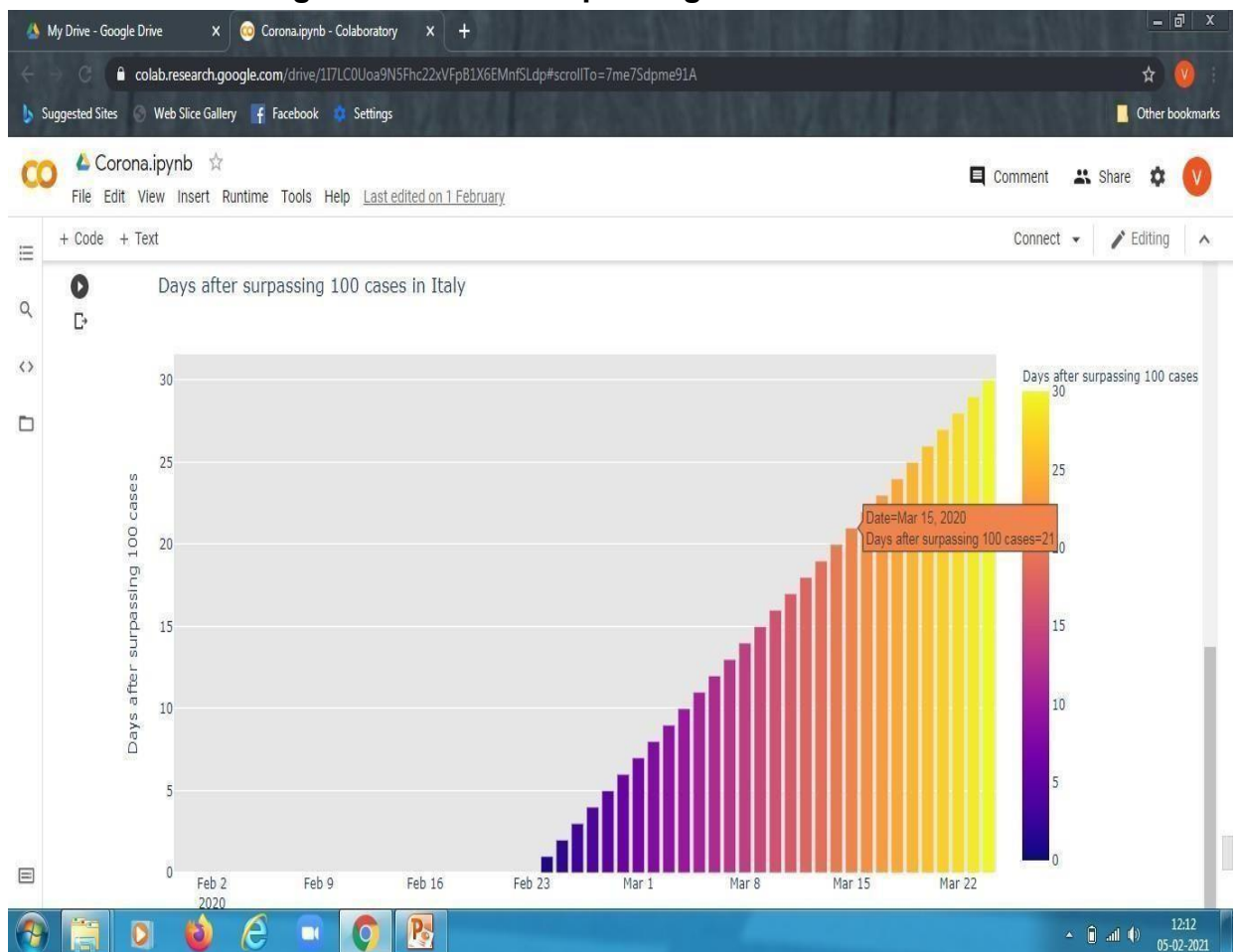


Fig-W:Cases after surpassing 100 in Italy

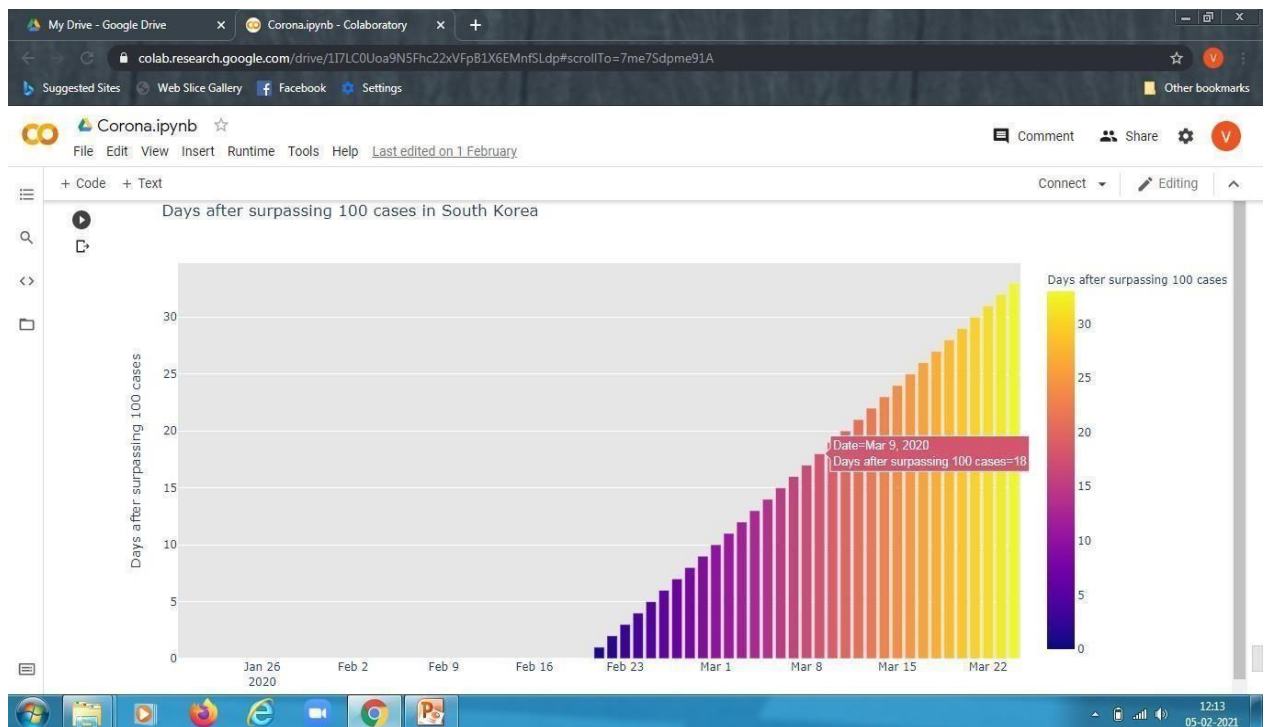


Fig-X: Cases after surpassing 100 in South Korea

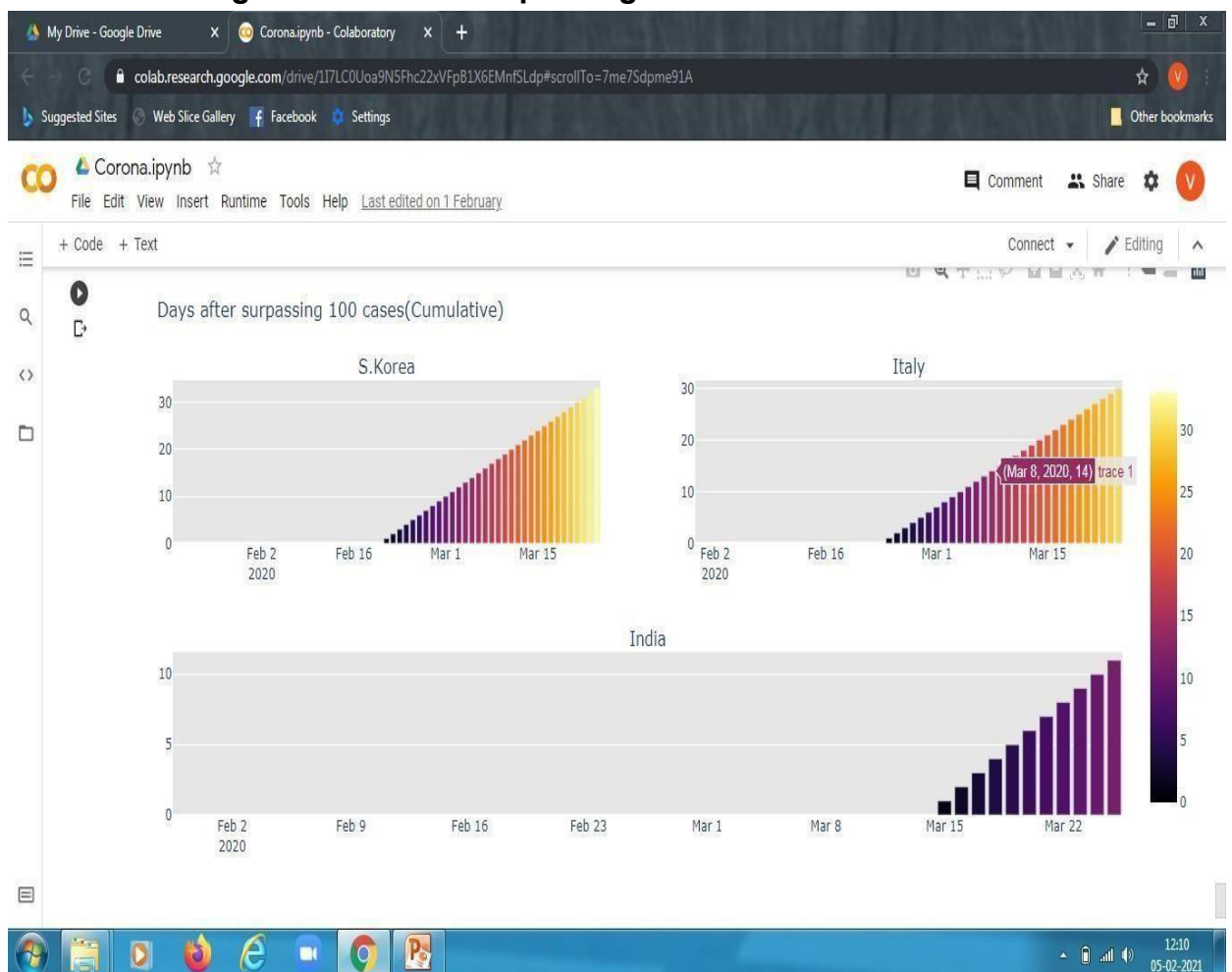


Fig-Y: Cumulative cases

B.SOURCE CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import folium
from folium import plugins
plt.rcParams['figure.figsize'] = 10,12
import warnings
warnings.filterwarnings('ignore')

df=pd.read_excel('/content/Covid cases in India.xlsx')
df_india=df.copy()
df

India_coord = pd.read_excel('/content/Indian Coordinates.xlsx')
dbd_India =
pd.read_excel('/content/per_day_cases.xlsx',parse_dates=True,sheet_name='India')
dbd_Italy =
pd.read_excel('/content/per_day_cases.xlsx',parse_dates=True,sheet_name='Italy')
dbd_Korea = pd.read_excel('/content/per_day_cases.xlsx',parse_dates=True,sheet_name='Korea')
dbd_Wuhan = pd.read_excel('/content/per_day_cases.xlsx',parse_dates=True,sheet_name='Wuhan')

df.drop(['S. No.'],axis=1,inplace=True)
df['Total cases'] = df['Total Confirmed cases (Indian National)'] + df['Total Confirmed cases ( Foreign National )']
total_cases = df['Total cases'].sum()
print('Total number of confirmed COVID 2019 cases across India till date (22nd March, 2020):', total_cases)

df.style.background_gradient(cmap='flare')

df['Total Active'] = df['Total cases'] - (df['Death'] + df['Cured'])
total_active = df['Total Active'].sum()
print('Total number of active COVID 2019 cases across India:',total_active)
Tot_Cases = df.groupby('Name of State / UT')['Total Active'].sum().sort_values(ascending=False).to_frame()
Tot_Cases.style.background_gradient(cmap='magma')

df['Total Active'] = df['Total cases'] + (df['Death'] + df['Cured']) + df['Total Confirmed cases ( Foreign National )'] + df['Total Confirmed cases (Indian National)']
total_active=df['Total Active'].sum()
print('Total cases across India:',total_active)
```

```
Tot_Cases = df.groupby('Name of State / UT')['Total Active'].sum().sort_values(ascending=False).to_frame()
Tot_Cases.style.background_gradient(cmap='Greens')
```

```
f, ax = plt.subplots(figsize=(12, 8))
data = df_full[['Name of State / UT', 'Total cases', 'Cured', 'Death']]
data.sort_values('Total cases', ascending=False, inplace=True)
sns.set_color_codes("pastel")
sns.barplot(x="Total cases", y="Name of State / UT", data=data, label="Total", color="g"
)
```

```
sns.set_color_codes("muted")
sns.barplot(x="Cured", y="Name of State / UT", data=data, label="Cured", color="b")
```

```
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=(0, 35), ylabel="", xlabel="Cases")
sns.despine(left=True, bottom=True)
```

```
f, ax = plt.subplots(figsize=(12, 8))
data = df_full[['Name of State / UT', 'Total cases', 'Cured', 'Death']]
data.sort_values('Total cases', ascending=False, inplace=True)
sns.set_color_codes("pastel")
sns.barplot(x="Total cases", y="Name of State / UT", data=data, label="Total", color="g")
```

```
sns.set_color_codes("muted")
sns.barplot(x="Death", y="Name of State / UT", data=data, label="Death", color="b")
```

```
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=(0, 35), ylabel="", xlabel="Cases")
sns.despine(left=True, bottom=True)
```

```
import plotly
plotly.io.renderers.default = 'colab'
```

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=dbd_India['Date'], y=dbd_India['Total Cases'], mode='lines+
markers', name='Total Cases'))
fig.update_layout(title_text='Trend of Coronavirus Cases in India(Cumulative cases)', plo
t_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
import plotly.express as px
fig = px.bar(dbd_India, x="Date", y="New Cases", barmode='group', height=400)
fig.update_layout(title_text='Coronavirus Cases in India on daily basis', plot_bgcolor='rg
b(230, 230, 230)')
```

```
fig.show()
```

```
fig = px.bar(dbd_India, x="Date", y="Total Cases", color='Total Cases', orientation='v', h
eight=600,
             title='Confirmed Cases in India', color_discrete_sequence = px.colors.cyclical.lc
```

eFire)

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
fig = px.bar(dbd_Italy, x="Date", y="Total Cases", color='Total Cases', orientation='v', height=600,
             title='Confirmed Cases in Italy', color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
fig = px.bar(dbd_Korea, x="Date", y="Total Cases", color='Total Cases', orientation='v', height=600,
             title='Confirmed Cases in South Korea', color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
fig = px.bar(dbd_Wuhan, x="Date", y="Total Cases", color='Total Cases', orientation='v', height=600,
             title='Confirmed Cases in Wuhan', color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
from plotly.subplots import make_subplots
```

```
fig = make_subplots(
    rows=2, cols=2,
    specs=[[{}], {}],
    [{"colspan":2}, None],
    subplot_titles=("S.Korea", "Italy", "India", "Wuhan"))
fig.add_trace(go.Bar(x=dbd_Korea['Date'], y=dbd_Korea['Total Cases'],
                    marker=dict(color=dbd_Korea['Total Cases'], coloraxis="coloraxis")), 1, 1)
fig.add_trace(go.Bar(x=dbd_Italy['Date'], y=dbd_Italy['Total Cases'],
                    marker=dict(color=dbd_Italy['Total Cases'], coloraxis="coloraxis")), 1, 2)
fig.add_trace(go.Bar(x=dbd_India['Date'], y=dbd_India['Total Cases'],
                    marker=dict(color=dbd_India['Total Cases'], coloraxis="coloraxis")), 2, 1)
fig.update_layout(coloraxis=dict(colorscale='magma'),
showlegend=False, title_text="Total Confirmed cases(Cumulative)")
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
from plotly.subplots import make_subplots
```

```

fig = make_subplots(
    rows=2, cols=2,
    specs=[[{}], {}],
    [{"colspan":2}, None]],
    subplot_titles=("Wuhan", "India"))
fig.add_trace(go.Bar(x=dbd_Wuhan['Date'], y=dbd_Wuhan['Total Cases'],
                    marker=dict(color=dbd_Wuhan['Total Cases'], coloraxis="coloraxis")),1, 1)
fig.add_trace(go.Bar(x=dbd_India['Date'], y=dbd_India['Total Cases'],
                    marker=dict(color=dbd_India['Total Cases'],coloraxis="coloraxis")),1, 2)
fig.update_layout(coloraxis=dict(colorscale='magma'),showlegend=False,title_text="Total Confirmed cases(Cumulative)")

```

```

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()

```

```

title = 'Main Source for News'
labels = ['S.Korea', 'Italy', 'India']
colors = ['rgb(122,128,0)', 'rgb(255,0,0)', 'rgb(49,130,189)']

```

```

mode_size = [10, 10, 12]
line_size = [1, 1, 8]

```

```

fig = go.Figure()

```

```

fig.add_trace(go.Scatter(x=dbd_Korea['Days after surpassing 100 cases'],
                        y=dbd_Korea['Total Cases'],mode='lines',
                        name=labels[0],
                        line=dict(color=colors[0], width=line_size[0]),
                        connectgaps=True))
fig.add_trace(go.Scatter(x=dbd_Italy['Days after surpassing 100 cases'],
                        y=dbd_Italy['Total Cases'],mode='lines',
                        name=labels[1],
                        line=dict(color=colors[1], width=line_size[1]),
                        connectgaps=True))
fig.add_trace(go.Scatter(x=dbd_India['Days after surpassing 100 cases'],
                        y=dbd_India['Total Cases'],mode='lines',
                        name=labels[2],
                        line=dict(color=colors[2], width=line_size[2]),
                        connectgaps=True))

```

```

annotations = []
annotations.append(dict(xref='paper',yref='paper', x=0.5,
y=-0.1,
                        xanchor='center', yanchor='top',
                        text='Days after crossing 100 cases ',
                        font=dict(family='Arial',
                                size=12,

```

```
        color='rgb(150,150,150)'),  
        showarrow=False))
```

```
fig.update_layout(annotations=annotations,plot_bgcolor='white',yaxis_title='Cumulative cases')
```

```
fig.show()
```

```
title = 'Main Source for News'
```

```
labels = ['S.Korea', 'Italy', 'India']
```

```
colors = ['rgb(122,128,0)', 'rgb(255,0,0)', 'rgb(49,130,189)']
```

```
mode_size = [10, 10, 12]
```

```
line_size = [1, 1, 8]
```

```
fig = go.Figure()
```

```
fig.add_trace(go.Scatter(x=dbd_Korea['Days after surpassing 100 cases'],  
                        y=dbd_Korea['New Cases'],mode='lines',  
                        name=labels[0],  
                        line=dict(color=colors[0], width=line_size[0]),  
                        connectgaps=True))
```

```
fig.add_trace(go.Scatter(x=dbd_Italy['Days after surpassing 100 cases'],  
                        y=dbd_Italy['New Cases'],mode='lines',  
                        name=labels[1],  
                        line=dict(color=colors[1], width=line_size[1]),  
                        connectgaps=True))
```

```
fig.add_trace(go.Scatter(x=dbd_India['Days after surpassing 100 cases'],  
                        y=dbd_India['New Cases'],mode='lines',  
                        name=labels[2],  
                        line=dict(color=colors[2], width=line_size[2]),  
                        connectgaps=True))
```

```
annotations = []
```

```
annotations.append(dict(xref='paper',yref='paper',x=0.5,y=-0.1,
```

```
                        xanchor='center',yanchor='top',  
                        text='Days after crossing 100 cases',  
                        font=dict(family='Arial',  
                                size=12,  
                                color='rgb(150,150,150)'),  
                        showarrow=False))
```

```
fig.update_layout(annotations=annotations,plot_bgcolor='white',yaxis_title='Cumulative cases')
```

```
fig.show()
```

```
df = pd.read_csv('/content/covid_19_clean_complete.csv',parse_dates=['Date'])
```

```
df.rename(columns={'ObservationDate':'Date', 'Country/Region':'Country'},  
inplace=True)
```

```

df_confirmed = pd.read_csv("/content/time_series_covid19_confirmed_global.csv")
df_recovered = pd.read_csv("/content/time_series_covid19_recovered_global.csv")
df_deaths = pd.read_csv("/content/time_series_covid19_deaths_global.csv")

df_confirmed.rename(columns={'Country/Region':'Country'}, inplace=True)
df_recovered.rename(columns={'Country/Region':'Country'}, inplace=True)
df_deaths.rename(columns={'Country/Region':'Country'}, inplace=True)

df_deaths.head()

df.head()

df2 = df.groupby(["Date", "Country", "Province/State"])[['Date', 'Province/State', 'Country',
'Confirmed', 'Deaths',
'Recovered']] df2.head()

df.query('Country=="India").groupby("Date")[['Confirmed', 'Deaths', 'Recovered']].sum().reset_index()

df.groupby('Date').sum().head()

confirmed = df.groupby('Date').sum()['Confirmed'].reset_index()
deaths = df.groupby('Date').sum()['Deaths'].reset_index()
recovered = df.groupby('Date').sum()['Recovered'].reset_index()

fig = go.Figure()
fig.add_trace(go.Scatter(x=confirmed['Date'], y=confirmed['Confirmed'],
mode='lines+markers', name='Confirmed', line=dict(color='Blue'))))
fig.add_trace(go.Scatter(x=deaths['Date'], y=deaths['Deaths'], mode='lines+markers',
name='Deaths', line=dict(color='Red'))))
fig.add_trace(go.Scatter(x=recovered['Date'], y=recovered['Recovered'],
mode='lines+markers', name='Recovered', line=dict(color='Green'))))
fig.update_layout(title='Worldwide NCOVID-
19 Cases', xaxis_tickfont_size=14, yaxis=dict(title='Number of Cases'))

fig.show()

from fbprophet import Prophet

confirmed = df.groupby('Date').sum()['Confirmed'].reset_index()
deaths = df.groupby('Date').sum()['Deaths'].reset_index()
recovered = df.groupby('Date').sum()['Recovered'].reset_index()

confirmed.columns = ['ds', 'y']
confirmed['ds'] = pd.to_datetime(confirmed['ds'])

```

```
confirmed.tail()
```

```
m = Prophet(interval_width=0.95)
```

```

m.fit(confirmed)
future = m.make_future_dataframe(periods=7)
future.tail()

forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()

confirmed_forecast_plot = m.plot(forecast)

confirmed_forecast_plot = m.plot_components(forecast)

deaths.columns = ['ds', 'y']
deaths['ds'] = pd.to_datetime(deaths['ds'])

m = Prophet(interval_width=0.95)
m.fit(deaths)
future = m.make_future_dataframe(periods=7)
future.tail()

forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()

deaths_forecast_plot = m.plot(forecast)

deaths_forecast_plot = m.plot_components(forecast)

recovered.columns = ['ds', 'y']
recovered['ds'] = pd.to_datetime(recovered['ds'])

m = Prophet(interval_width=0.95)
m.fit(recovered)
future = m.make_future_dataframe(periods=7)
future.tail()
forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()

recovered_forecast_plot = m.plot(forecast)

recovered_forecast_plot = m.plot_components(forecast)

fig = go.Figure()
fig.add_trace(go.Scatter(x=dbd_Wuhan['Date'], y=dbd_Wuhan['Total Cases'], mode='lines+markers', name='Total Cases'))
fig.update_layout(title_text='Trend of Coronavirus Cases in Wuhan(Cumulative cases)',
plot_bgcolor='rgb(230 ,230 ,230)')
fig.show()

import plotly.express as px

```

```
fig = px.bar(dbd_Wuhan, x="Date", y="New Cases",barmode='group',height=400)
```

```
fig.update_layout(title_text='Coronavirus Cases in Wuhan on daily basis',plot_bgcolor='rgb(230,230,230)')
```

```
fig.show()
```

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=dbd_Italy['Date'],y = dbd_Italy['Total Cases'], mode='lines+markers',name='Total Cases'))
fig.update_layout(title_text='Trend of Coronavirus Cases in Italy(Cumulative cases)',plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
import plotly.express as px
fig = px.bar(dbd_Italy, x="Date", y="New Cases", barmode='group',height=400)
fig.update_layout(title_text='Coronavirus Cases in Italy on daily basis',plot_bgcolor='rgb(230,230,230)')
```

```
fig.show()
```

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=dbd_Korea['Date'],y = dbd_Korea['Total Cases'], mode='lines+markers',name='Total Cases'))
fig.update_layout(title_text='Trend of Coronavirus Cases in Korea(Cumulative cases)',plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
import plotly.express as px
fig = px.bar(dbd_Korea, x="Date", y="New Cases", barmode='group', height=400)
fig.update_layout(title_text='Coronavirus Cases in Korea on daily basis',plot_bgcolor='rgb(230,230,230)')
```

```
fig.show()
```

```
fig = px.bar(dbd_India, x="Date", y="New Cases", color='New Cases',
orientation='v',height=600,
title='New Cases in India', color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
fig = px.bar(dbd_Italy, x="Date", y="New Cases",color='New Cases', orientation='v',
height=600,
title='New Cases in Italy', color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```

```
fig = px.bar(dbd_Korea, x="Date", y="New Cases",color='New Cases', orientation='v', height=600,
```

```

        title='New Cases in South Korea',color_discrete_sequence =
px.colors.cyclical.IceFire)

```

```

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()

```

```

fig = px.bar(dbd_Wuhan, x="Date", y="New Cases",color='New Cases', orientation='v',
height=600,
        title='New Cases in Wuhan',color_discrete_sequence = px.colors.cyclical.IceFire)

```

```

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()

```

```

from plotly.subplots import make_subplots

```

```

fig = make_subplots(

```

```

    rows=2, cols=2,
    specs=[[{}], {}],
        [{"colspan":2}, None]],
        subplot_titles=("S.Korea", "Italy", "India", "Wuhan")
    )

```

```

fig.add_trace(go.Bar(x=dbd_Korea['Date'], y=dbd_Korea['New Cases'],
        marker=dict(color=dbd_Korea['New Cases'], coloraxis="coloraxis")),1, 1)

```

```

fig.add_trace(go.Bar(x=dbd_Italy['Date'], y=dbd_Italy['New Cases'],
        marker=dict(color=dbd_Italy['New Cases'], coloraxis="coloraxis")),1, 2)

```

```

fig.add_trace(go.Bar(x=dbd_India['Date'], y=dbd_India['New Cases'],
        marker=dict(color=dbd_India['New Cases'], coloraxis="coloraxis")),2, 1)

```

```

fig.update_layout(coloraxis=dict(colorscale='inferno'),showlegend=False,title_text="Total New cases(Cumulative)")

```

```

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()

```

```

fig = px.bar(dbd_India, x="Date", y="Days after surpassing 100 cases", color='Days after surpassing 100 cases', orientation='v',height=600,
        title='Days after surpassing 100 cases in India', color_discrete_sequence = px.colors.cyclical.IceFire)

```

```

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()

```

```

fig = px.bar(dbd_Italy, x="Date", y="Days after surpassing 100 cases", color='Days after surpassing 100 cases', orientation='v',height=600,
        title='Days after surpassing 100 cases in Italy', color_discrete_sequence = px.colors.cyclical.IceFire)

```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')  
fig.show()
```

```
fig = px.bar(dbd_Korea, x="Date", y="Days after surpassing 100 cases", color='Days
after surpassing 100 cases', orientation='v',height=600,
            title='Days after surpassing 100 cases in South Korea',
            color_discrete_sequence = px.colors.cyclical.IceFire)
```

```
fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
from plotly.subplots import make_subplots
```

```
fig = make_subplots(

    rows=2, cols=2,
    specs=[[{}], {}],
            [{"colspan":2}, None]],
    subplot_titles=("S.Korea", "Italy", "India", "Wuhan"))
fig.add_trace(go.Bar(x=dbd_Korea['Date'], y=dbd_Korea['Days after surpassing 100 cas
es'],
                    marker=dict(color=dbd_Korea['Days after surpassing 100 cases'], coloraxis=
s="coloraxis")),1, 1)
fig.add_trace(go.Bar(x=dbd_Italy['Date'], y=dbd_Italy['Days after surpassing 100 cases']
,
                    marker=dict(color=dbd_Italy['Days after surpassing 100 cases'], coloraxis=
"coloraxis")),1, 2)
fig.add_trace(go.Bar(x=dbd_India['Date'], y=dbd_India['Days after surpassing 100
case s'],
                    marker=dict(color=dbd_India['Days after surpassing 100 cases'], coloraxis
="coloraxis")),2, 1)
fig.update_layout(coloraxis=dict(colorscale='inferno'),
showlegend=False,title_text="Day s after surpassing 100 cases(Cumulative)")

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```