

**PREDICTION OF COVID-19 USING SUPERVISED**

**MACHINE LEARNING ALGORITHMS**

**A PROJECT REPORT**

***Submitted by***

**HARSHITHA P (REGISTER NO :211417104083)**

**MADDINA SIREESHA (REGISTER NO :211417104134)**

***in partial fulfillment for the award the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.**

**ANNA UNIVERSITY: CHENNAI-600 025**

**AUGUST 2021**

**BONAFIDE CERTIFICATE**

Certified that this project report **“PREDICTION OF COVID-19 USING SUPERVISED MACHINE LEARNING ALGORITHMS”** is the bonafide

work of **HARSHITHA P (211417104083) and MADDINA SIREESHA (211417104134)** who carried out the project work under my supervision.

## SIGNATURE

## Dr.S.MURUGAVALLI,M.E.,Ph.D.,

## HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE,

NAZARATHPETTAI,

POONAMALLEE,

CHENNAI-600 123.

## SIGNATURE

## Mrs.K.KIRUTHIKA, M.Tech

## SUPERVISOR

## ASSISTANT PROFESSOR

DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE,

NAZARATHPETTAI,

POONAMALLEE,

CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on......05/08/2021.....................

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like express our heartfelt and sincere thanks to our Directors

## Tmt.C.VIJAYARAJESWARI, Dr.C. SAKTHIKUMAR, M.E., Ph.D., and

**Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Guide Mrs.K.KIRUTHIKA, M.Tech** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

**NAME OF THE STUDENTS**

HARSHITHA P

MADDINA SIREESHA

**ABSTRACT**

Machine learning (ML) based forecasting mechanisms have their Proved significance to anticipate in preoperative outcomes to improve the decision making on the future course of actions. The ML models have long been used in many application domains which is needed the identification and prioritization of adverse factors for a threat. Several prediction methods are being popularly used to handle forecasting problems. The ML models to forecast the number upcoming patients affected by COVID-19 which is presently considered as thepotential threat to mankind. In this project proposes the COVID -19 predictions of cases in particular area using machine learning algorithms.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | Iii |
|  | **LIST OF FIGURES** | V |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 1 |
|  | 1.2 Problem Definition | 1 |
| **2.** | **LITERATURE SURVEY** | 2 |
| **3.** | **SYSTEM ANALYSIS** | 5 |
|  | 3.1 Existing System | 5 |
|  | 3.2 Proposed system | 6 |
|  | 3.3 Purpose | 6 |
|  | 3.4 Requirement Analysis and Specification | 6 |
|  | 3.4.1 Safety Requirements | 9 |
|  | 3.4.2 Hardware Requirements | 10 |
|  | 3.4.3 Software Requirements | 10 |
|  | 3.5 Technology Stack | 10 |
| **4.** | **DESIGN AND IMPLEMENTATION CONSTRAINTS** | 10 |
|  | * 1. gradient boosting work | 10 |
|  | 4.2 mean square error | 11 |
|  | 4.3 working | 11 |
| **5.** | **SYSTEM DESIGN** | 11 |
|  | 5.1 UML Diagram | 12 |
|  | 5.1.1. Use case Diagram | 12 |
|  | 5.1.2. Sequence Diagram | 12 |
|  | 5.1.3. Activity Diagram | 13 |
|  | 5.1.4. Collaboration Diagram | 14 |
|  | 5.1.5. Class Diagram | 15 |
|  | 5.1.6. Data Flow Diagram | 16 |
| **6.** | **SYSTEM ARCHITECTURE** | 18 |
|  | 6.1 Architecture Overview | 18 |
|  | 6.2 Module Design Specification | 20 |
|  | 6.2.1. Module Explanation | 20 |
|  | 6.2.1.1. pre - processor | 20 |
|  | 6.2.1.2. training | 20 |
|  | 6.2.1.3. testing | 20 |
|  | 6.2.1.4. prediction using data sets | 20 |
|  | 6.3 Program Design Language | 21 |
|  | 6.3.1. python | 21 |
|  | 6.3.2. introduction of python | 25 |
|  | 6.3.3. machine learning | 26 |
| **7.** | **SYSTEM IMPLEMENTATION** | 29 |
| **8.** | **SYSTEM TESTING** | 56 |
|  | 8.1 Unit Testing | 56 |
|  | 8.2 Integration Testing | 56 |
|  | 8.3 Functional Tests | 57 |
|  | 8.4 Performance Test | 57 |
|  | 8.5 Stress Test | 57 |
|  | 8.6. Structured Test | 57 |
|  | 8.7. Testing Techniques / Testing Strategies | 58 |
| **9.** | **CONCLUSION** | 64 |
|  | 9.1 Conclusion and Future Enhancements | 65 |
|  | **APPENDICES** | 66 |
|  | A.1 Sample Screens | 66 |
|  | A.2 Publications | 67 |
|  | **REFERENCES** | 68 |

**LIST OF FIGURES:**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **TITLE** | **PAGE NO** |
| 5.1 | Use case Diagram | 12 |
| 5.2 | Sequence Diagram | 13 |
| 5.3 | Activity Diagram: | 14 |
| 5.4 | Data Flow Diagram | 15 |
| 9.1 | Cases of Covid -19 | 62 |
| 9.2 | According to the region | 63 |
| 9.3 | Covid - 19 testing, vaccinated status | 64 |
| 9.4 | total number of confirmed cases in India | 65 |
| 9.5 | graph of confirmed cases | 66 |

**1.INTRODUCTION**

* 1. **Overview**

Machine learning (ML) has proved itself as a prominent field of study over the last decade by solving many very complex and sophisticated real-world problems. The application areas included almost all the real-world domains such as healthcare, autonomous vehicle (AV), business applications, natural language processing (NLP), intelligent robots, gaming, climate modelling and error method voice, and image processing. ML algorithms’ learning is typically based on trial it opposite of conventional algorithms, One of the most significant areas of ML is forecasting numerous standard ML algorithms have been used in these area to guide the future course of actions needed in many application areas including, weather forecasting, disease forecasting, stock market of forecasting as well as disease prognosis. Various regression and neural networking model shave a wide applicability in predicting the conditions of patients in the future with a specific disease. [Data mining](https://en.wikipedia.org/wiki/Data_mining) is a related field of study, focusing on [exploratory data analysis](https://en.wikipedia.org/wiki/Exploratory_data_analysis) through [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning). Machine learning is also referred to as [predictive analytics](https://en.wikipedia.org/wiki/Predictive_analytics) Coronaviruses are a large family of viruses which may cause disease in animals or humans. Seven coronaviruses can produce infection in people around the world but commonly people get infected with these four human coronaviruses: 229E, NL63, OC43, and HKU1. They usually cause a respiratory infection ranging from the common cold to more severe diseases such as Middle East Respiratory Syndrome (MERS) and Severe Acute Respiratory. Syndrome (SARS) and the most recently discovered coronavirus (COVID-19)causes infectious disease.

11th Feb 2020, the (WHO) officially renamed the clinical condition COVID-19 (a shortening of Corona Virus Disease-19), which was announced in a tweet. An outbreak of COVID-19 caused by the 2019 novel coronavirus (SARS-CoV-2) began in Wuhan, Hubei Proviniece, China in December 2019, the current outbreak is officially a pandemic. Since, knowledge about this virus is rapidly evolving, readers are urged to update themselves regularly. The novel coronavirus disease 2019 (COVID-19) pandemic caused by the SARS-CoV-2 continues to pose a critical and urgent threat to global health. The outbreak early December 2019 in the Hubei province of the People’s Republic of China has spread worldwide. As of October 2020, the overall Number of patients confirmed to have the disease has exceeded 39,500,000, in >180 countries, though the number of people infected is probably much higher. More than 1,110,000 people have died from COVID. This pandemic continues to challenge medical systems worldwide in many aspects, including sharp increases in demands the hospital beds and critical shortages in medical equipment, while many healthcare workers have themselves been infected. Thus, the capacity for immediate clinical decisions and effective usage of healthcare resources is crucial. The most validated diagnosis test for COVID-19, using reverse transcriptase polymerase chain reaction (RT-PCR), has long been in shortage in developing countries. This contributes to increased infection rates and delays critical preventive measures. Effective screening enables quick and efficient diagnosis of COVID-19 and can mitigate the burden on healthcare systems. Prediction models that combine several features to estimate the risk of infection have been developed, in the hope of assisting medical staff worldwide in triaging patients, especially in the context of limited healthcare resources.

These models use features such as computer tomography (CT) scans, clinical symptoms[7](https://www.nature.com/articles/s41746-020-00372-6#ref-CR7), laboratory tests and an integration of these features. However, most previous models were based on data from hospitalized patients, thus are not effective in screening for SARS-CoV-2 in the general population. pandemic of coronavirus disease 2019 (COVID-19) is spreading all over the world. Medical imaging such as X-ray and computed tomography (CT) plays an essential role in the global fight Against the COVID-19, whereas the recently emerging artificial intelligence (AI) technologies further strengthen the power of the imaging tools and help medical specialists. We hereby review the rapid responses in the community of medical imaging (empowered by AI) toward COVID-19.For example, empowered image acquisition can significantly help automate the scanning procedure and also reshape the workflow with minimal contact to patients, providing the best protection on the imaging technicians. Also, AI can improve work efficiency by accurate delineation of infections in X-ray and CT images, facilitating subsequent quantification. Time from exposure and symptom on set is generally between two and 14 days, with an average of five days. Common symptoms include fever, cough, sneezing and shortness of the breath. Complications may include pneumonia, throat pain and acute respiratory distress syndrome. Currently, there is no specific antiviral treatment or vaccine; efforts consist of symptom abolition supportive therapy. Recommended preventive measures include washing your hands with soap, covering the mouth when coughing, maintaining 1-meter distance from other people and monitoring and self-isolation for fourteen days for people who suspect they are infected. The standard tool of diagnosis is by reverse transcription polymerase chain reaction (RT-PCR) from a throat swab or nasopharyngeal swab.

**1.2 Problem Definition**

Coronaviruses are a large family of viruses which may cause disease in animals or humans. Seven coronaviruses can produce infection in people around the world but commonly people get infected with these four human coronaviruses: 229E, NL63, and OC43HKU1. They usually cause a respiratory infection ranging from the common cold to more severe diseases such as Middle East Respiratory Syndrome (MERS) and Severe Acute Respiratory Syndrome (SARS) and the most recently discovered coronavirus (COVID-19) causes infectious disease. This zoonotic disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The WHO originally called this infectious disease Novel Coronavirus-Infected Pneumonia (NCIP). Much research has already been done using various artificial intelligence for diagnosing and predicting COVID-19 infection and recovery. In the work of data mining predictive model For COVID-19 patients recovery were developed with four data mining algorithms but however among them, model made of the decision tree has the highest accuracy of 99.85%. In the work of convolutional neural networks that predict novel coronavirus with x-ray images were developed. The deep learning technique, which is one of the sub-branches of ML, inspired by the structure of the human brain is used for automatic prediction of 2019-nCoV patients. Dataset with chest x-ray images were used, and pre-trained models including InceptionV3, ResNet50 and Inception Res Net V2 were trained and tested on the dataset. The performance result of the study showed that the RestNet pre-trained model gave the highest accuracy among the three models: 98%.

Therefore, this shows that the model can help health workers to make decisions in clinical practice with high-performance accuracy, which can also detect 2019-nCoV in the early stages of infection. In the work of a modified susceptible-exposed-infectious-removed (SEIR) Model and ML Model for prediction of the trend of the 2019-nCoV pandemic in China were developed under public health interventions. The models were effective in predicting the pandemic peaks and size. Population migration data before and after 23rd January 2020 and updated 2019-nCoV epidemiological data were integrated into the SEIR Model to derive the pandemic curve. The ML approach was trained on 2003 SARS data to predict the pandemic. In the work of [[4](https://link.springer.com/article/10.1007/s42979-020-00394-7#ref-CR4)] data mining and a deep learning pilot study were carried out to predict 2019-nCoV incidence by levering Google trend data in Iran. Long Short-Term Memory and Linear Regression Models were used to estimate the number of 2019-nCoV positive cases. The models were evaluated with root mean square error (RMSE) metric and 10 folds cross-validation techniques, respectively. The RMSE of long short-term memory and linear regression Models were 27.187 and 7.562, respectively. Moreover, the study predicted the trend of the 2019-nCoV outbreak. Such predictions can support healthcare managers and policy makers with planning, allocating and deploying healthcare resources effectively. Reference identified an intrinsic 2019nCoV genomic signature using an ML-based alignment-free approach. This approach incorporated ML-controlled digital signal analysis of genome analysis, augmented decision-making process and Spearman's rank correlation coefficient analysis for validation result. The result of the study corroborates the research hypothesis of a bat as the origin 2019-nCoV pandemic and the study further classifies the pandemic As Sarbeco virus within bet a corona virus. More than 5000 unique genomic sequences from the dataset, totaling 61:8 million bp were analyzed with more than 90% accuracy.

In the work of the machine learning-based approach was developed for a real-time forecast of 2019-nCoV outbreak using news alerts reported by Media Cloud and official heath report from Chinese Center Disease for Control and Prevention, internet search activity from Baidu and daily forecast from GLEAM (an agent-based mechanistic model). The approach uses a clustering that enables exploration of geo-spatial synchronicities of 2019-nCoV activities across Chinese provinces. The approach is able to produce an accurate forecast two days ahead of time. The ML-driven approach was also used to predict the severity of the infection in patients. A clinical dataset from Wuhan was proposed in the study, with 15 patients admitted to a hospital in Wuhan, China between January 10th to February, 18th, 2020 being screened. There 375 patients who were discharged including 201 survivors. The prognostic prediction model based on the ML XG boost algorithm was developed and was tested with 29 patients including 3 patients from other hospitals, who were cleared after 19th February 2020. The model was able to predict the mortality risk of 2019-nCoV patients and clinical route to the recognition of critical cases from severe cases, and more so the model helped the doctors with identification of 2019-nCoV patients and intervention with the model potentially being able to reduce the mortality risk. In the study of ML and Vaxign reverse vaccinology tools were used to predict2019-nCoV vaccine candidates. The Vaxign reverse vaccinology tool predicted protein as a likely adhesion while Vaxign ML predicted S protein had a high protective antigenicity score. The predicted vaccine in the study provides new strategies for effective and safe 2019-nCoV vaccine development. Reference presented a data-driven for ML approach for the analysis of the 2019-nCoV pandemic from its early infection dynamics especially inflation counter over time, using US data starting from the first case on 20th January 2020. The novel coronavirus disease 2019 (COVID-19) pandemic caused by the SARS-CoV-2 continues to pose a critical and urgent threat to global health. The outbreak in early December 2019 in the Hubei province of the People’s Republic of China has spread worldwide. As of October 2020, the overall number of patients confirmed to have the disease has exceeded 39,500,000, in >180 countries, though the number of people infected is probably much higher. More than 1,110,000 people have died from COVID-19.

1. **LITERATURE SURVEY**

**LITERATURE SURVEY :**

Ana Lisa V. Gomes Lawrence J. K. Wee, Asif M. Khan, LauraH. V. G. Gil, Ernesto T. A .Marques, Jr., Carlos E. Calzavara - Silvaand Tin Wee Tan2 [2010] presented a novel application of the support vector machines (SVM) algorithm to analyze the expression pattern of 12 genes inblood mononuclear cells (PBMCs) of 28 dengue patients (13) (DH F and 15 DF)during acute viral infection. That model can potentially be used to accurately classify the prognosis of patients with the benign form (DF) from those with the life-threatening(DHF) DENV disease. Benjamin M. Althouse, Yih Yng Ng and Derek A. T. Cummings used Logistic on

regression and Support Vector Machine (SVM) models to predict a binary outcome

defined by whether dengue incidence exceeded a choosen threshold. Incidence prediction models were assessed using r2 and Pearson correlation between predicted and observed dengue incidence. The search terms included in the models include nomenclature terms, terms describing signs and symptoms as well as treatment seeking which made their prediction model more efficient. Vassiliki, Christopher M. Jones, Rodolphe Poupardin, John Vontas and Hilary Ran son used quantitative reverse transcription PCR (RT-PCR) for validating the

selected microarray data. In this paper they detected two previously described target sitemutations,1016I and 1534C in pyrethroid resistant populations from Grand Cayman. Previous research studies on technology adoption have focused on the various forms of fear. For example, anxiety is an important factor that helps manage technology approval and apprehension. Within the education sector, the adoption of technology by students is influenced by anxiety.

Furthermore, apart from anxiety, a lack of experience and skills may also influence technology use. The fear of using technology, combined with poor technological literacy and anxiety, negatively affects the adoption of technology. Hence, it is essential for teachers and educators to focus on psychological development and help students accept the use of technology. Other factors of the fear of using technology within the educational sector include technical readiness and preparedness; technology adoption is negatively influenced by both of these factors. The education sector is not the only sector that has exhibited a fear of technology adoption. Medical sector students usually perceive risks and exhibit negative anxiety when technology is used. In addition, health anxiety is one of the top concerns of the health care sector. Health anxiety includes the apprehension of patients and the fear of receiving results about a severe illness. With regard to the banking sector, various kinds of fear that relate to customers’ perceptions and attitudes toward technology have been recognized. Customers do not want to use their data for mobile payments. Customers fear the use of technology in mobile banking and are negatively influenced by the frauds that have occurred. As a result, they lack both technological experience and trust in technology. With regard to the household sector, the main reasons why technology is not being used include the fear of using technology and the fear that technology will increase the number of family tasks. Various research studies have assessed the issues that relate to technological acceptance and fear. These research studies are based on the TAM and several other models and most of these research studies have assessed how the fear of technology can influence technology acceptance. Various technology users have provided justifications for their fear of technology use. For example, several users have stated that their fear is related to self-confidence. Errors are made when a human is assigned to a job, & excessively worrying about this fact enhances fear.

Moreover, several users have stated that they do not use technology because they believe that technology is time-consuming, and therefore does not allow them to complete their tasks. Various technology acceptance studies have assessed the influence of fear on the breach data privacy, and this is why privacy and security awareness are emphasized in technology research studies Previous studies have not provided sufficient empirical research on the use of mobile learning in United Arab Emirates (UAE) institutions, nor have they considered the factors that influence students’ actual technology use.

When it comes to methodology, technology acceptance researchers have typically analyzed theoretical models by using structural equation modeling and machine learning algorithms. After considering various theoretical models, we conducted this study with the following 2 objectives: (1) examine how students use mobile learning by integrating the TAM and TPB model into 1 theoretical model, and validated. Created theoretical model with the help of machine learning and partial least squares-structural equation modeling (PLS-SEM) algorithms.

The unique pathogenesis mechanisms of SARS-Cov-2, and the related spectrum of symptoms are the subject of many ongoing studies. The model we built provides initial COVID-19 test screening based on simple clinical signs and symptoms. Improving clinical priorities may lower the burden currently faced by health systems, by facilitating optimized management of healthcare resources during future waves of the SARS-Cov-2 pandemic. This is especially important in developing countries with limited resources. This research is not without shortcomings.

We relied on the data reported by the Israeli Ministry of Health, which has limitations, biases and missing information regarding some of the features. For example, for patients labelled as having had contact with a person confirmed to have COVID-19, additional information such as the duration and location (indoors/outdoors) of the contact was not available.

Some symptoms (such as lack of smell and taste) were identified as being very predictive of a COVID-19 infection by previous studies, but were not recorded by the Israeli Ministry of Health. We showed that training and testing a model while filtering out symptoms of high bias in advance still achieved very high accuracy. We also note that all the symptoms were self-reported, and a negative value for a symptom might mean that the symptom was not reported.

It is therefore important to assess the model’s performance in the circumstance that more values are unreported or missing rather than with negative values. To simulate a less biased condition, in our prospective test set, we randomly selected negative reports of all five symptoms at a time, and removed the negative values. When applied to these simulated test sets, the model still showed promising results thus reinforcing our confidence in the model.The framework combines many available data sources (number of positive cases, number of patients in hospitals .

**3.SYSTEM ANALYSIS**

**3.1** **EXISTING SYSTEM:**

In earlier days the disease prediction is done by identifying the symptoms of covid in patients. The prediction begins from this identification. The datasets are collected and prediction is done with the help of the available Data using random forest algorithm in machine learning.

**DRAWBACKS OF EXISTING SYSTEM:**

The prediction can be done only after the patient is affectedby the disease.

We can only predict at the later stage and hence cannot be predicted before the patient is affected.

Cannot be fully avoided only we can control the effects caused by it Disease prediction accuracy is very low.

* 1. **PROPOSED SYSTEM:**

Our proposed system uses The main objective is to predict the possibility of covid

case Region wise. The prediction is based on geographical and climatic factors

with help of Machine learning techniques:

Corona virus spread has conducted the society under the edge Of loss in the social lives. Additionally, it is crucial to investigate the transmission growth ahead and predict the future occurrences of the transmission.

In concurrent, state-of-the-art mathematical models are chosen based on machine learning for a computational process to predict the spread of the virus, for instance: Machine learning and deep learning strategies are performed using the python library to predict the total number of confirmed, recovered, and death cases extensively. This prediction will allow undertaking specific determinations based on transmission growth, such as expanding the lockdown phase, performing the sanitation plan, and providing daily support.

**BENEFITS OF PROPOSED SYSTEM:**

Daily support and supplies. In this segment, we’re going to generate a week ahead forecast of confirmed cases of COVID-19 using Prophet, with specific prediction intervals by creating a base model both with and without tweaking of seasonality-related parameters and additional regressors. Prophet is open source software released by Facebook’s Core Data. It is available for download on CRAN and PyPI.

We use Prophet, a procedure for forecasting time series data based on an additive model where non - linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

Predicting the dengue fever at the earlier stage, we can reduce the number of deaths Caused by it. The accuracyis very high when compared to the existing system. Timely Prediction of Covid is the only way to outbreak the disease. Region wise Prediction is done. Linear regression algorithm makes the estimation procedure simple. Less effort is need for pre-processing the dataset. Scaling and normalization of the data is not required.

**PURPOSE**

The main aim of this project is to predict the covid patients.

**3.4 REQUIREMENT ANALYSIS AND SPECIFICATION**

**3.3.1 FEASIBILITY STUDY**

A feasibility study is carried out to select the best system that meets performance requirements. The main aim of the feasibility study activity is to determine that it would be financially and technically feasible to develop the product.

* + 1. **TECHNICAL FEASIBILITY**

This is concerned with specifying the software will successfully satisfy the user requirement. Open source and business-friendly and it is truly cross platform, easily deployed and highly extensible.

**3.3.2 ECONOMIC FEASIBILITY**

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. The enhancement of the existing system doesn’t incur any kind of drastic increase in the expenses. Python is open source and readily available for all users. Since the project is running in python and Anaconda notebook hence is cost efficient. The enhancement of the existing system doesn’t incur any kind of drastic increase in the expenses. Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system

**3.3.3 SAFETY REQUIREMENTS**

* Avoid the 3Cs: spaces that are closed, crowded or involve close contact.
* Outbreaks have been reported in restaurants, choir practices, fitness classes, nightclubs, offices and places of worship where people have gathered, often in crowded indoor settings where they talk loudly, shout, breathe heavily or sing.
* The risks of getting COVID-19 are higher in crowded and inadequately ventilated spaces where infected people spend long periods of time together in close proximity. These environments are where the virus appears to spread by respiratory droplets or aerosols more efficiently, so taking precautions is even more important.
* Meet people outside. Outdoor gatherings are safer than indoor ones, particularly if indoor spaces are small and without outdoor air coming in.
* For more information on how to hold events like family gatherings, children’s football games and family occasions, read our [Q&A on small public gatherings](https://www.who.int/emergencies/diseases/novel-coronavirus-2019/question-and-answers-hub/q-a-detail/q-a-small-public-gatherings-and-covid-19).
* Avoid crowded or indoor settings but if you can’t, then take precautions:
* Open a window. Increase the amount of ‘natural ventilation’ when indoors.
* WHO has published Q & Ans on ventilation and air conditioning for both the [general public](https://www.who.int/news-room/q-a-detail/q-a-ventilation-and-air-conditioning-and-covid-19) and [people who manage public spaces and buildings](https://www.who.int/emergencies/diseases/novel-coronavirus-2019/question-and-answers-hub/q-a-detail/q-a-ventilation-and-air-conditioning-in-public-spaces-and-buildings-and-covid-19).
* Wear a mask and Don’t forget the basics of good hygiene

Regularly and thoroughly clean your hands with an alcohol-based hand rub or wash them with soap and water. This eliminates germs including viruses that may be on your hands.

* Avoid touching your eyes, nose and mouth. Hands touch many surfaces and can pick up viruses. Once contaminated, hands can transfer the virus to your eyes, nose or mouth. From there, the virus can enter your body and infect you.
* Cover your mouth and nose with your bent elbow or tissue when you cough or sneeze. Then dispose of the used tissue immediately into a closed bin and wash your hands. By following good ‘respiratory hygiene’, you protect the people around you from viruses, which cause colds, flu and COVID-19.
* Clean and disinfect surfaces frequently especially those which are regularly touched, such as door handles, faucets and phone screens.

What to do if you feel unwell?

* Know the full range of symptoms of COVID-19. The most common symptoms of COVID-19 are fever, dry cough, and tiredness. Other symptoms that are less common and may affect some patients include loss of taste or smell, aches and pains, headache, sore throat, nasal congestion, red eyes, diarrhoea, or a skin rash.
* Stay home and self-isolate even if you have minor symptoms such as cough, headache, mild fever, until you recover. Call your health care provider or hotline for advice. Have someone bring you supplies. If you need to leave your house or have someone near you, wear a medical mask to avoid infecting others.
* If you have a fever, cough and difficulty breathing, seek medical attention immediately. Call by telephone first, if you can and follow the directions of your local health authority.
* Keep up to date on the latest information from trusted sources, such as WHO or your local and national health authorities.
  + 1. **HARDWARE REQUIREMENTS**
* Processor       :    Intel Pentium-IV and above
* Hard disk       :    80GB Min.
* RAM             :    512 MB Min. & Above
* Others           :    If any Applicable
  + 1. **SOFTWARE REQUIREMENTS**
* Python 2.7.9 Version

**3.3.7 TECHNOLOGY STACK**

**PYTHON:**

Python is an [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language)   for [general-purpose programming](https://en.wikipedia.org/wiki/General-purpose_programming_language). Python has a design philosophy that emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability), notably using [significant whitespace](https://en.wikipedia.org/wiki/Significant_whitespace). It provides constructs that enable clear programming on both small and large scales. Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). Supports multiple [programming. paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), Including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming), and has a large, and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library). Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [C Python](https://en.wikipedia.org/wiki/CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python,

is [open source](https://en.wikipedia.org/wiki/Open_source) software and has a community-based development model, as do nearly all of its variant implementations.

Python was conceived in the late 1980s by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) at [Centrum Wiskunde & Informatica](https://en.wikipedia.org/wiki/Centrum_Wiskunde_%26_Informatica) (CWI) in the [Netherlands](https://en.wikipedia.org/wiki/Netherlands) as a successor to [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)), which was inspired by [SETL](https://en.wikipedia.org/wiki/SETL), capable of [exception handling](https://en.wikipedia.org/wiki/Exception_handling) and interfacing with the [Amoeba](https://en.wikipedia.org/wiki/Amoeba_(operating_system)) operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's [BenevolentDictatorForLife](https://en.wikipedia.org/wiki/Benevolent_Dictator_For_Life), a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a 5-member "Steering Council" to lead the project.[[43]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-43) As of 2021, the current members of this council are Barry Warsaw, Brett Cannon, Carol Willing, Thomas Wouters, and Pablo Galindo Salgado. Python 2.0 was released on 16 October 2000, with many major new features, including a [cycle-detecting](https://en.wikipedia.org/wiki/Cycle_detection) [garbage collector](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) and support for [Unicode](https://en.wikipedia.org/wiki/Unicode). Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility). Many of its major features were [backported](https://en.wikipedia.org/wiki/Backporting) to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2 to 3 utility, which automates (at least partially) the translation of Python 2 code to Python 3. Python 2.7's [end-of-life](https://en.wikipedia.org/wiki/End-of-life_(product)) date was initially set at 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No more security patches or other improvements will be released for it. With Python 2's [end-of-life](https://en.wikipedia.org/wiki/End-of-life_(product)), only Python 3.6.xand later are supported. Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues, leading to possible [remote code execution](https://en.wikipedia.org/wiki/Remote_code_execution).

**3.3.8 PROGRAM DESIGN LANGUAGE**

In our Dengue Possibility Forecasting Model we are mainly using two

algorithms and they are;

* Gradient Boosting Regression (BGR)
* Mean Square Error (MSE)

**3.3.9 GRADIENT BOOSTING REGRESSION(GBR):**

Various techniques. (e.g,weighted average, majority vote or normal average)

Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. The Gradient Boosting Regression (GBR) algorithm is mainly used for predicting the data, and in GBR we are using Ensemble Technique. When we try to predict the target variable using any machine learning technique, the main causes of difference in actual and predicted values are noise, variance, and bias. Ensemble helps to reduce these factors (except noise, which is irreducible error). An ensemble is just a collection of predictors which come together (e.g. mean of all predictions) to give a final prediction. Essembling techniques are further classified into: Bagging and Boosting. Bagging is a simple essembling technique. We build many independent predictors/models and combine them using some model a One of the most common descriptions of boosted learning is that a group of “weak learners” can be combined to form a “strong learner” Features are the inputs that are given to the machine learning algorithm, the inputs that will be used to calculate an output value. In a mathematical sense, the features of the dataset are the variables used to solve the equation. The other part of the equation is the label or target, which are the classes the instances will be categorized into. Because the labels contain the target values for the machine learning classifier, when training a classifier you should split up the data into training and testing sets. The training set will have targets/labels, while the testing set won't contain these values.

**3.4.0 HOW ARE WEAK LEARNERS COMBINED?**

The take away is that weak learners are best combined in a way that allows each one to solve a limited section of the problem.  Any machine learning routine can be used as a weak learner.  Neural nets, support vector machines or any other would work, but the most commonly used weak learner is the decision tree.

**3.4.1 PROCESS:**

A Regressor attempts to fit a numeric value to something.  The something might be

fitting the population of a country to GPS coordinates stock price of a company to

monthly sales. The end result of a fitted regression analysis is that you pass in

the known features and can predict the unknown output value. Here, is the process

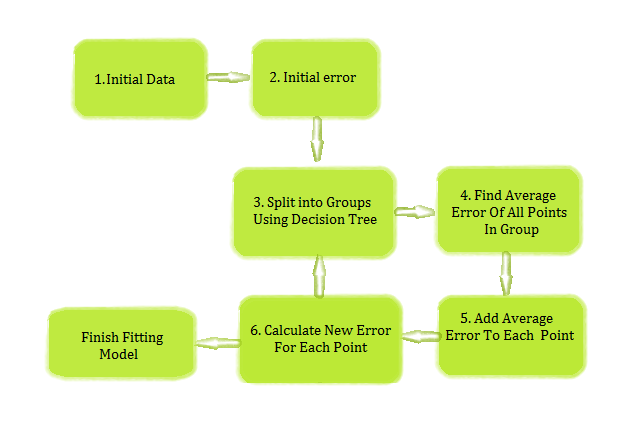
that boosting regression. goal of putting data with similar error into the same

group. For each group, find the average error .

An ensemble is just a collection of predictors which come together (e.g. mean of all predictions) to give a final prediction. Essembling techniques are further classified into: Bagging and Boosting. Bagging is a simple essembling technique. We build many independent predictors/models and combine them using some model a One of the most common descriptions of boosted learning is that a group of “weak learners” . The end result of a fitted regression analysis is that you pass in

the known features and can predict the unknown output value. Here, is the process

that boosting regression. goal of putting data with similar error.

 Fig: represents the GBR flow

**3.7.4 HOW GRADIENT BOOSTING WORK?**

Gradient boosting involves three elements: A loss function to be optimized. A weak learner to make predictions.An additive model to add weak learners to minimize the loss function.

**3.7.5 IMPROVEMENT TO BASIC GRADIENT BOOSTING:**

In this this section we will look at 4 enhancements to basic gradient boosting:

Tree Constraints

Shrinkage

Random sampling

Penalized Learning

**3.7.6 MEAN SQUARE ERROR:**

MSE can represent the difference between the actual observations and the observation values predicted by the model. In this context, it is used to determine the extent to which the model fits the data as well as whether removing some explanatory variables is possible without significantly harming the model's predictive ability. The smaller the mean Squared Error, the closer you are to finding the [line of best fit](http://www.statisticshowto.com/line-of-best-fit/). The reason minimizing squared error is preferred is because it prevents large errors better. Mean Squared Error is just Sum Squared Error divided by the number of data points.

1. **SYSTEM DESIGN:**

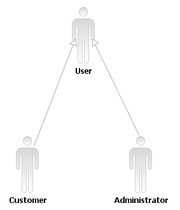
**4.3.1 USECASE DIAGRAM:**

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

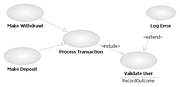
Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse. While a [use case](https://en.wikipedia.org/wiki/Use_case) itself might drill into a lot of detail about every possibility, a use-case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system".[[1]](https://en.wikipedia.org/wiki/Use_case_diagram#cite_note-1)Due to their simplistic nature, use case diagrams can be a good communication tool for [stakeholders](https://en.wikipedia.org/wiki/Project_stakeholder). The drawings attempt to mimic the real world and provide a view for the [stakeholder](https://en.wikipedia.org/wiki/Project_stakeholder) to understand how the system is going to be designed. Siau and Lee conducted research to determine if there was a valid situation for use case diagrams at all or if they were unnecessary. What was found was that the use case diagrams conveyed the intent of the system in a more simplified manner to [stakeholders](https://en.wikipedia.org/wiki/Project_stakeholder) and that they were "interpreted more completely than class diagrams",The purpose of a use case diagram is to capture the dynamic aspect of a system. They provide a simplified graphical representation of what the system should do in a use case. Further diagrams and documentation are needed for a complete functional and technical outlook on the system.

Diagram building blocks



Actor inheritance



Use case relationships

Interaction among actors is not shown on the use case diagram. If this interaction is essential to a coherent description of the desired behavior, perhaps the system or use case boundaries should be re-examined. Alternatively, interaction among actors can be part of the assumptions used in the use case.

**Actor Generalization**

One popular relationship between Actors is Generalization/Specialization. This is useful in defining overlapping roles between actors. The notation is a solid line ending in a hollow triangle drawn from the specialized to the more general actor. 

**Use Case Relationships**

Three relationships among use cases are used often in practice.

**Include**

In one form of interaction, a given use case may include another. "Include is a Directed Relationship between two use cases, implying that the behavior of the included use case is inserted into the behavior of the including use case".

The first use case often depends on the outcome of the included use case. This is useful for extracting truly common behaviors from multiple use cases into a single description. The notation is a dashed arrow from the including to the included use case, with the label "«include»". This usage resembles a macro expansion where the included use case behavior is placed inline in the base use case behavior. There are no parameters or return values

**Extend**

In another form of interaction, a given use case (the extension) may *extend* another. This relationship indicates that the behavior of the extension use case may be inserted in the extended use case under some conditions. The notation is a dashed arrow from the extension to the extended use case, with the label "«extend»". Notes or constraints may be associated with this relationship to illustrate the conditions under which this behaviour will be executed. Modelers use the «extend» relationship to indicate use cases that are "optional" to the base use case. Depending on the modeler's approach "optional" may mean "potentially not executed with the base use case" or it may mean "not required to achieve the base use case goal."

**Generalization**

In the third form of relationship among use:

case a generalization/specialization relationship exists. A given use case may be a specialized form of an existing use case. The notation is a solid line ending in a hollow triangle drawn from the specialized to the more general use case. This resembles the object-oriented concept of sub-classing, in practice it can be both useful and effective to factor out common behaviors, constraints and assumptions to the general use case, describe them once, and deal with it in the same way, except for the details in the specialised.

has the attribute DegreeofHazardousness which is needed only for cargo, but not for passenger luggage. Additional only passenger luggage has a connection to a coupon. Obviously, here two similar but different domain concepts are combined into one class. Through specialization the two special cases of freights are formed: Piece of Cargo  and Piece of Luggage. The attribute Degree of Hazardousness  is placed where it belongs—in Piece of Cargo.A generalization is a form of [abstraction](https://en.wikipedia.org/wiki/Abstraction) whereby common properties of specific instances are formulated as general concepts or claims. Generalizations posit the existence of a domain or [set](https://en.wikipedia.org/wiki/Set_theory) of elements, as well as one or more common characteristics shared by those elements (thus creating a [conceptual model](https://en.wikipedia.org/wiki/Conceptual_model)). As such, they are the essential basis of all valid [deductive](https://en.wikipedia.org/wiki/Deductive) inferences (particularly in [logic](https://en.wikipedia.org/wiki/Logic), mathematics and science), where the process of [verification](https://en.wikipedia.org/wiki/Falsifiability) is necessary to determine whether a generalization holds true for any given situation.

Generalization can also be used to refer to the process of identifying the parts of a whole, as belonging to the whole.

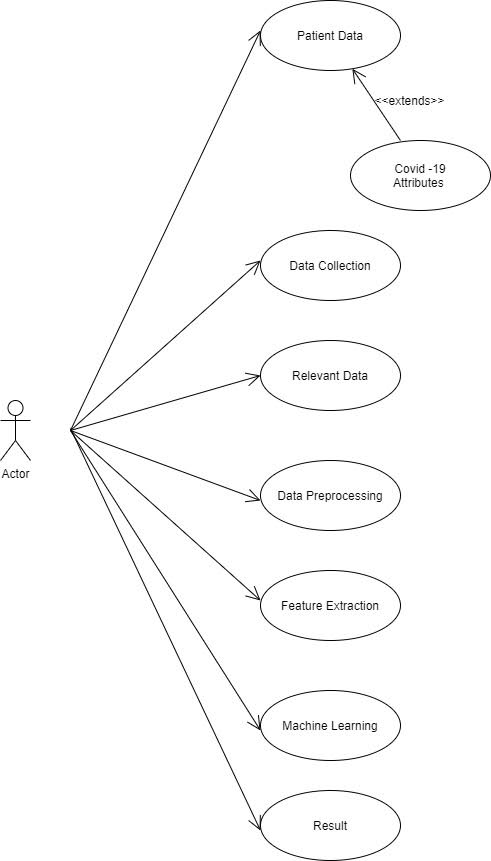


fig 5.1 use case diagram of prediction covid cases

* + 1. **SEQUENCE DIAGRAM:**

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram. A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.  
Note that there are two types of sequence diagrams: UML diagrams and code-based diagrams. The latter is sourced from programming code and will not be covered in this guide. Lucidchart’s [UML diagramming software](https://www.lucidchart.com/pages/examples/uml_diagram_tool) is equipped with all the shapes and features you will need to model both.

**BENEFITS:**

Sequence diagrams can be useful references for businesses and other organizations.

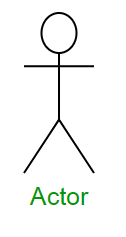
Try drawing a sequence diagram to:

* Represent the details of a UML use case.
* Model the logic of a sophisticated function, or operation.
* See how objects and components interact with each other to complete a process.
* Plan and understand the detailed functionality of an existing or future scenario.

**USES OF SEQUENCE DIAGRAM:**

The following scenarios are ideal for using a sequence diagram:

Usagescenario**:** A usage scenario is a diagram of how your system could potentially be used. It's a great way to make sure that you have worked through the logic of every usage scenario for the system.

Methodlogic**:** Just as you might use a UML sequence diagram to explore the logic of a use case, you can use it to explore the logic of any function, procedure, or complex process. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.  


**Figure –** notation symbol for actor

We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.  
For example – Here the user in seat reservation system is shown as an actor where it exists outside the system and is not a part of the system.

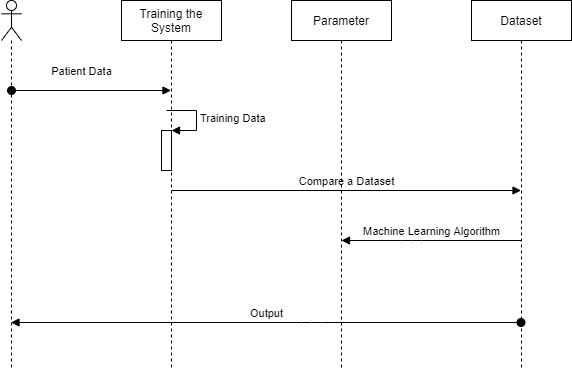


Fig 5.1 A sequence diagram shows a parallel vertical lines, different processes or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in order in which they occur.

**4.3.3 ACTIVITY DIAGRAM:**

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

* Rounded rectangles represent activities.
* Diamonds represent decisions.
* Bars represent the start or end of concurrent activities.
* A black circle represents the start of the workflow.
* An encircled circle represents the end of the workflow.

Acitivity diagram is another important diagram in UML to describe the dynamic

aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow controlled by using different elements such as fork, join, etc..

**PURPOSE OF ACTIVITY DIAGRAM:**

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram

is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single. The purpose of an activity diagram can be described as −

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

* ellipses represent actions;
* diamonds represent decisions;
* bars represent the start (split) or end (join) of concurrent activities;
* a black circle represents the start (initial node) of the workflow;
* an encircled black circle represents the end (final node).

Arrows run from the start towards the end and represent the order in which activities happen. Activity diagrams can be regarded as a form of a structured [flowchart](https://en.wikipedia.org/wiki/Flowchart) combined with a traditional data flow diagram. Typical flowchart techniques lack constructs for expressing concurrency.[[5]](https://en.wikipedia.org/wiki/Activity_diagram#cite_note-5) However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops.

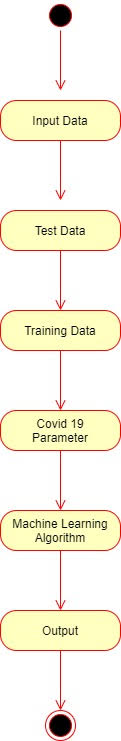


Fig 5.3

The above figure shows Activity diagrams for covid 19 prediction.

* + 1. **DATA FLOW DIAGRAM**

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

Data Flow Diagram (DFD) is a graphical representation of the flow of data through

various process in a system. It depicts the information flow and the transformations

that are applied as the data moves from input to output. DFD effectively conveys the aim of the system without going into details about its implementation. A DFD

usually comprises of four components. These four components can be represented by four simple symbols. They can be explained in details as follow:

* External entities (source/destination of data) are represented by squares

Process (input-processing-output) are represented by rectangles with rounded corners Flow Data Diagram (DFD**)**

DFDs are easier to understand by technical and non-technical audiences DFDs can

provides a high level system overview, complete with boundaries and connections

to other systems and DFDs can provide a detailed representation of system components.

The DFD can be uses to represent any level of data abstraction. Each level has

more information and data functional details compared to its previous level.

Here we are having three levels of DFDs namely,

* Level - 0DFDs
* Level - 1DFDs
* Level - 2DFDs

Flow chart or flow diagram... is a diagram that visually displays interrelated information such as events, steps in a process, functions, etc., in an organized fashion, such as sequentially or chronologically.Flow diagram [is] a graphic representation of the physical route or flow of people, materials, paperworks, vehicles, or communication associated with a process, procedure plan, or investigation.In the second definition the meaning is limited to the representation of the physical route or flow. An example of such a diagram is the illustration of the flows in a nuclear [submarine](https://en.wikipedia.org/wiki/Submarine) [propulsion system](https://en.wikipedia.org/wiki/Propulsion_system), which shows different streams back and forth in the system. The representation of such a system can be supplemented by one or more flowcharts, which show the sequence of one of the flows in one direction, or any of the control flows to manage the system.The physical movement of objects from one location to another can also be visualized in a mix of maps and flowchart or [sankey diagram](https://en.wikipedia.org/wiki/Sankey_diagram), which are called [flow maps](https://en.wikipedia.org/wiki/Flow_map). Flow maps can indicate on a [map](https://en.wikipedia.org/wiki/Map), what flows, moves or migrates, in which direction, and in which quantities etc. A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. DFD effectively conveys the aim of the system without going into details about its implementation. A DFD usually comprises of four components. These four components can be represented by four simple symbols

They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard

.

**5. SYSTEM ARCHITECTURE:**

**5.1. ARCHITECTURE OVERVIEW**

System architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal ascription and

representation of a system, organized in a way that supports reasoning about the

structures and behaviours of the system. One can think of system architecture as a set of representations of an existing (or future) system. These representations initially describe a general, high-level functional organization, and are progressively refined to more detailed and concrete descriptions.

System architecture conveys the informational content of the [elements](https://en.wiktionary.org/wiki/element) consisting of a system, the relationships among those elements, and the [rules](https://en.wiktionary.org/wiki/rule) governing those relationships. The architectural components and set of relationships between these components that an architecture description may consist of hardware, [software](https://en.wikipedia.org/wiki/Software), documentation, facilities, manual procedures, or roles played by organizations.

A system architecture primarily concentrates on the internal [interfaces](https://en.wikipedia.org/wiki/Interface_(computer_science)) among the system's [components](https://en.wiktionary.org/wiki/component) or [subsystems](https://en.wikipedia.org/wiki/System), and on the interface(s) between the system and its external environment, especially the [user](https://en.wikipedia.org/wiki/User_(computing)). One can contrast a system architecture with [system architecture engineering](https://en.wikipedia.org/w/index.php?title=System_architecture_engineering&action=edit&redlink=1) (SAE) - the method and discipline for effectively implementing the architecture of a system.

* SAE is a *method* because a sequence of steps is prescribedto produce or to change the architecture of a system within a [set](https://en.wikipedia.org/wiki/Set_(mathematics)) of [constraints](https://en.wikipedia.org/wiki/Constraint_(mathematics)).
* SAE is discipline because a body of [knowledge](https://en.wikipedia.org/wiki/Knowledge) is used to inform [practitioners](https://en.wiktionary.org/wiki/practitioner).
* As to the most effective way to design the system within a set of constraint.

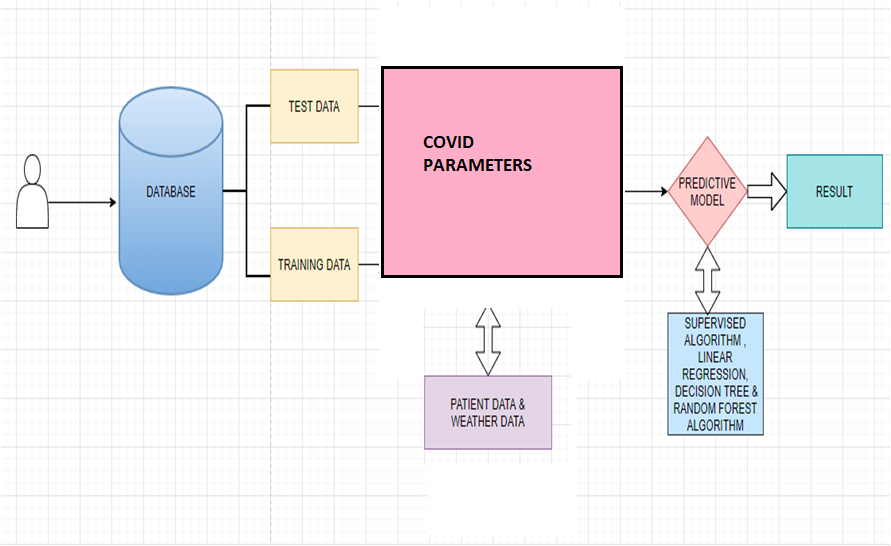


Fig: 6.1

Architectural diagram of covid prediction in people

**6.2. MODULE DESIGN SPECIFICATION**:

The following Modules are described in detail.

* Data pre-processing
* Training the System
* Testing

Prediction using Dataset

**6.2.1. MODULE EXPLANATION:**

**6.2.1.1. DATA PRE-PROCESSING**

The dataset contains all the information which the learning model is

supposed to learn for making right predictions. The raw data might have a lot of

variations in the values of each feature which might lead to incorrect results.

Hence the learning process will re-process the dataset.

The pre-processing techniques include the following;

* Mean removal.
* Scaling.

**6.2.1.2. TRAINING THE SYSTEM**

Real time data is collected from kaggle. Training data is used to make sure the machine recognizes patterns in the data,The cross-validation data is used to ensure better accuracy and efficiency of the algorithm used to train the machine. Supervised learning is used to train the system

**6.2.1.3. SYSTEM TESTING**

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified [requirements](https://en.wikipedia.org/wiki/Requirements). System testing takes, as its input, all of the integrated components that have passed [integration testing](https://en.wikipedia.org/wiki/Integration_testing). The purpose of integration testing is to detect any inconsistencies between the units that are integrated together (called assemblages). System testing seeks to detect defects both within the "inter-assemblages" and also within the system as a whole The actual result is the behavior produced or observed when a component or system is tested. System testing is performed on the entire system in the context of either [functional requirement](https://en.wikipedia.org/wiki/Functional_requirements) specifications (FRS)or [system.](https://en.wikipedia.org/wiki/Requirements_analysis)

[Requirement](https://en.wikipedia.org/wiki/Requirements_analysis) specification (SRS), or both. System testing tests not only the design, but also the behaviour and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software or hardware requirements specification(s).

**6.2.1.4. PREDICTION OF DATA**

Predictive analytics is the practice of extracting information from existing datasets in order to determine patterns and predict future outcomes. The test data is used to see how well the machine can predict new answers based on its training. It is important that no observations from the training set are included in the test set. Predicts the dengue using the training provided to the system .The prediction is done region wise . Timely Prediction of Dengue is the only way to outbreak the disease.

Existing The current data is fed into the predictive model .The predictive model methodology of dengue prediction is not efficient.

The prediction result is finally given to the Ministry of Health and Family Welfare. Though predictive analytics has been around for decades, it's a technology whose time has come. More and more organizations are turning to predictive analytics to increase their bottom line and competitive advantage.

Why now?

* Growing volumes and types of data, and more interest in using data to produce valuable insights.
* Faster, cheaper computers.
* Easier-to-use software.
* Tougher economic conditions and a need for competitive differentiation.

With interactive and easy-to-use software becoming more prevalent, predictive analytics is no longer just the domain of mathematicians and statisticians. Business analysts and line-of-business experts are using these technologies as well.

It is important that no observations from the training set are included in the test set. Predicts the dengue using the training provided to the system .The prediction is done region wise. Timely Prediction of Dengue is the only way to outbreak the disease. Prediction for the number of cases in a pandemic and implications for health care needs and resources have received a lot of attention in the scientific world ,government agencies.and in media lately. With the plethora of models, there is also growing scrutiny about the accuracy of different models, and an appreciation that model parameters need to be refined based on evolving knowledge about the disease trajectory and factors impacting infection and transmission rates.

* 1. **PROGRAM DESIGN LANGUAGE**
     1. **PYTHON**

Python is an [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) [high-level](https://en.wikipedia.org/wiki/High-level_programming_language) [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Python's design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with its notable use of [significant indentation](https://en.wikipedia.org/wiki/Off-side_rule). Its [language constructs](https://en.wikipedia.org/wiki/Language_construct) as well as its [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) approach aim to help [programmers](https://en.wikipedia.org/wiki/Programmers) write clear, logical code for small and large-scale projects. Python is [dynamically-typed](https://en.wikipedia.org/wiki/Dynamic_programming_language) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple  including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly, [procedural](https://en.wikipedia.org/wiki/Procedural_programming)), object-oriented and [functional programming](https://en.wikipedia.org/wiki/Functional_programming). Python is often described as a "batteries included" language due to its comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library). [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) began working on Python in the late 1980s, as a successor to the [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)), and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as [list comprehensions](https://en.wikipedia.org/wiki/List_comprehension) and a garbage collection system using [reference counting](https://en.wikipedia.org/wiki/Reference_counting). Python 3.0 was released in 2008 and was a major revision of the language that is not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility) and much Python 2 code does not run unmodified on Python 3. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

* + - 1. **INTRODUCTION TO PYTHON**

Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Python Programming Language is very well suited for Beginners, also for experienced programmers with other programming languages like C++ and Java.

Below are some facts about Python Programming Language:

1. Python is currently the most widely used multi-purpose, high-level programming language.
2. Python allows programming in Object-Oriented and Procedural paradigms.
3. Python programs generally are smaller than other programming languages like Java.
4. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.
5. The biggest strength of Python is huge collection of standard library which can be used for the following:
   * [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)
   * GUI Applications (like [Kivy](https://www.geeksforgeeks.org/kivy-tutorial/), Tkinter, PyQt etc.)
   * Web frameworks like [Django](https://www.geeksforgeeks.org/django-tutorial/) (used by YouTube, Instagram, Dropbox)
   * Image processing (like [Open CV](https://www.geeksforgeeks.org/opencv-python-tutorial/), Pillow)
   * Web scraping (like Scrapy, Beautiful Soup, Selenium)
   * Test frameworks
   * Multimedia
   * Scientific computing
   * Text processing and many more.

* + - 1. **WORKING OF PYTHON**

Python source files use the ".py" extension and are called "modules." With a Python module hello.py, the easiest way to run it is with the shell command "python hello.py Alice" which calls the Python interpreter to execute the code in hello.py, passing it the command line argument "Alice". See the [official docs page](http://docs.python.org/using/cmdline) on all the different options you have when running Python from the command-line. The outermost statements in a Python file, or "module", do its one-time setup — those statements run from top to bottom the first time the module is imported somewhere, setting up its variables and functions. A Python module can be run directly — as above "python hello.py Bob" — or it can be imported and used by some other module. When a Python file is run directly, the special variable "\_\_name\_\_" is set to "\_\_main\_\_". Therefore, it's common to have the boilerplate if \_\_name\_\_ ==... shown above to call a main () function when the module is run directly, but not when the module is imported by some other module.

In a standard Python program, the list sys.argv contains the command-line arguments in the standard way with sys.argv being the program itself, sys.argv the first argument, and so on. If you know about argc, or the number of arguments, you can simply request this value from Python with len (sys.argv), just like we did in the interactive interpreter code above when requesting the length of a string. In general, len() can tell you how long a string is, the number of elements in lists and tuples (another array-like data structure), and the number of key-value pairs in a dictionary.

**USER DEFINED FUNCTIONS**

The def  keyword defines the function with its parameters within parentheses and its code indented. The first line of a function can be a documentation string ("docstring") that describes what the function does. The doc string can be a single line, or a multi-line description as in the example above. (Yes, those are "triple quotes," a feature unique to Python!) Variables defined in the function are local to that function, so the "result" in the above function is separate from a "result" variable in another function. The return statement can take an argument, in which case that is the value returned to the caller. Here is code that calls the above repeat() function, printing what it returns: One unusual Python feature is that the whitespace indentation of a piece of code affects its meaning. A logical block of statements such as the ones that make up a function should all have the same indentation, set in from the indentation of their parent function or "if" or whatever. If one of the lines in a group has a different indentation, it is flagged as a syntax error.

Python's use of whitespace feels a little strange at first, but it's logical and I found I got used to it very quickly. Avoid using TABs as they greatly complicate the indentation scheme (not to mention TABs may mean different things on different platforms). Set your editor to insert spaces instead of TABs for Python code.

A common question beginners ask is, "How many spaces should I indent?" According to [the official Python style guide (PEP 8)](http://python.org/dev/peps/pep-0008/#indentation), you should indent with 4 spaces.

So use "name" if it's a single name, and "names" if it's a list of names, and "tuples" if it's a list of tuples. Many basic Python errors result from forgetting what type of value is in each variable, so use your variable names (all you have really) to help keep things straight. As far as actual naming goes, some languages prefer under scored\_parts for variable names made up of "more than one word," but other languages prefer camel Casing.

In general, Python [prefers](http://python.org/dev/peps/pep-0008/#function-names) the underscore method but guides developers to defer to camel Casing if integrating into existing Python code that already uses that style. Readability counts. Read more in [the section on naming conventions in PEP 8](https://www.python.org/dev/peps/pep-0008/#naming-conventions).As you can guess, keywords like 'print' and 'while' cannot be used as variable names — you'll get a syntax error if you do.

However, be careful not to use built-ins as variable names. For example, while 'str' and 'list' may seem like good names, you'd be overriding those system variables. Built-ins are not keywords and thus, are susceptible to inadvertent use by new Python developers.

**6.3.1.3 HISTORY:**

Python was conceived in the late 1980s[[39]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-venners-interview-pt-1-39) by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) at [Centrum Wiskunde & Informatica](https://en.wikipedia.org/wiki/Centrum_Wiskunde_%26_Informatica) (CWI) in the [Netherlands](https://en.wikipedia.org/wiki/Netherlands) as a successor to [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)), which was inspired by [SETL](https://en.wikipedia.org/wiki/SETL),[[40]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-AutoNT-12-40) capable of [exception handling](https://en.wikipedia.org/wiki/Exception_handling) and interfacing with the [Amoeba](https://en.wikipedia.org/wiki/Amoeba_(operating_system)) operating system.[[10]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-faq-created-10) Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's [Benevolent Dictator For Life](https://en.wikipedia.org/wiki/Benevolent_Dictator_For_Life), a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a 5-member "Steering Council" to lead the project. As of 2021, the current members of this council are Barry Warsaw, Brett Cannon, Carol Willing, Thomas Wouters, and Pablo Galindo Salgado. Python 2.0 was released on 16 October 2000, with many major new features, including a [cycle-detecting](https://en.wikipedia.org/wiki/Cycle_detection) [garbage collector](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) and support for [Unicode](https://en.wikipedia.org/wiki/Unicode). Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility).[[46]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-3.0-release-46) Many of its major features were [backported](https://en.wikipedia.org/wiki/Backporting) to Python 2.6.xand 2.7.x version series. Releases of Python 3 include the 2to3 utility, which automates (at least partially) the translation of Python 2 code to Python 3. Python 2.7's [end-of-life](https://en.wikipedia.org/wiki/End-of-life_(product)) date was initially set at 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3.

### MACHINE LEARNING

Machine learning is a branch of [artificial intelligence (AI)](https://www.ibm.com/in-en/cloud/learn/what-is-artificial-intelligence) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. In data science, an algorithm is a sequence of statistical processing steps. In machine learning, algorithms are 'trained' to find patterns and features in massive amounts of data in order to make decisions and predictions based on new data. The better the algorithm, the more accurate the decisions and predictions will become as it processes more data.Today, examples of machine learning are all around us. Digital assistants search the web and play music in response to our voice commands. Websites recommend products and movies and songs based on what we bought, watched, or listened to before. Robots vacuum our floors while we do.

Something better with our time. Spam detectors stop unwanted emails from reaching our inboxes. Medical image analysis systems help doctors spot tumors they might have missed. And the first self-driving cars are hitting the road. We can expect more. As big data keeps getting bigger, as computing becomes more powerful and affordable, and as data scientists keep developing more capable algorithms, machine learning will drive greater and greater efficiency in our personal and work lives.

* + - 1. **INTRODUCTION TO MACHINE LEARNING:**

**Regression algorithms:**

Linear and logistic regression are examples of regression algorithms used to understand relationships in data. Linear regression is used to predict the value of a dependent variable based on the value of an independent variable. Logistic regression can be used when the dependent variable is binary in nature: A or B. For example, a linear regression algorithm could be trained to predict a salesperson’s annual sales (the dependent variable) based on its relationship to the salesperson’s education or years of experience (the independent variables.) Another type of regression algorithm called a support vector machine is useful when dependent variables are more difficult to classify.

**Decision trees:**

[Decision trees](https://www.ibm.com/cloud/learn/random-forest) use classified data to make recommendations based on a set of decision rules. For example, a decision tree that recommends betting on a particular horse to win, place, or show could use data about the horse (e.g., age, winning percentage, pedigree) and apply rules to those factors to recommend an action or decision.

**Instance-based algorithms:**

A good example of an instance-based algorithm is K-Nearest Neighbor .It uses classification to estimate how likely a data point is to be a member of one group or another based on its proximity other data point.

**Supervised machine learning:**

It trains itself on a labeled data set. That is, the data labled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify data. For example, a computer vision model designed to identify purebred German Shepherd dogs might be trained on a data set of various labeled dog images.

Supervised machine learning requires less training data than other machine learning methods and makes training easier because the results of the model can be compared to actual labeled results. the data labled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify data.

1. **SYSTEM IMPLEMENTATION:**

# importing the required libraries

import pandas as pd

# Visualisation libraries

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

# Manipulating the default plot size

plt.rcParams['figure.figsize'] = 10, 12

# Disable warnings

import warnings

warnings.filterwarnings('ignore')

# Reading the datasets

# Coordinates of India States and Union Territories

India\_coord = pd.read\_excel('/content/Indian Coordinates.xlsx')

#Day by day data of India, Korea, Italy and Wuhan

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)

#Total Active  is the Total cases - (Number of death + Cured)

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='Reds')

df\_full = pd.merge(India\_coord,df,on='Name of State / UT')

map = folium.Map(location=[20, 70], zoom\_start=4,tiles='Stamenterrain')

for lat, lon, value, name inzip(df\_full['Latitude'], df\_full['Longitude'], df\_full['Total cases'], df\_full['Name of State / UT']):

 folium.CircleMarker([lat, lon], radius=value\*0.8, popup = ('<strong>State</strong>: ' + str(name).capitalize() + '

''<strong>Total Cases</strong>: ' + str(value) + '

'),color='red',fill\_color='red',fill\_opacity=0.3 ).add\_to(map)

Map

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="r")

sns.set\_color\_codes("muted")

sns.barplot(x="Cured", y="Name of State / UT", data=data, label="Cured", color="g")

# Add a legend and informative axis label

ax.legend(ncol=2, loc="lower right", frameon=True)

ax.set(xlim=(0, 35), ylabel="",xlabel="Cases")

sns.despine(left=True, bottom=True)

# Rise of COVID-19 cases in India

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)',plot\_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='rgb(230, 230, 230)')

fig.show()

# import plotly.express as px

fig = px.bar(dbd\_India, x="Date", y="Total Cases", color='Total Cases', orientation='v', height=600,

             title='Confirmed Cases in India', color\_discrete\_sequence = px.colors.cyclical.IceFire)

'''Colour Scale for plotly

<a href="https://plot.ly/python/builtin-colorscales/">https://plot.ly/python/builtin-colorscales/</a>

'''

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,

fig.update\_layout(plot\_bgcolor='rgb(230, 230, 230)')

fig.show()

# import plotly.graph\_objects as go

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.add\_trace(go.Bar(x=dbd\_Wuhan['Date'], y=dbd\_Wuhan['Total Cases'],

#                     marker=dict(color=dbd\_Wuhan['Total Cases'], coloraxis="coloraxis")),2, 2)

fig.update\_layout(coloraxis=dict(colorscale='Bluered\_r'), showlegend=False,title\_text="Total Confirmed cases(Cumulative)")

fig.update\_layout(plot\_bgcolor='rgb(230, 230, 230)')

fig.show()

# import plotly.graph\_objects as go

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()

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()

fig = go.Figure()

#Plotting datewise confirmed cases

fig.add\_trace(go.Scatter(x=confirmed['Date'], y=confirmed['Confirmed'], mode='lines+markers', name='Confirmed',line=dict(color='blue', width=2)))

fig.add\_trace(go.Scatter(x=deaths['Date'], y=deaths['Deaths'], mode='lines+markers', name='Deaths', line=dict(color='Red', width=2)))

fig.add\_trace(go.Scatter(x=recovered['Date'], y=recovered['Recovered'], mode='lines+markers', name='Recovered', line=dict(color='Green', width=2)))

fig.update\_layout(title='Worldwide NCOVID-19 Cases', xaxis\_tickfont\_size=14,yaxis=dict(title='Number of Cases'))

fig.show()

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)

1. **PERFORMANCE ANALYSIS**

The evaluation framework developed here for assessing how well models predicted the total number of cumulative deaths is shown in Fig. [1](https://www.nature.com/articles/s41467-021-22457-w#Fig1) for an example country—the United States, chosen as it has the highest number of reported COVID-19 deaths—and similar figures for all locations included in the study can be found in the Supplementary. Fig. [1](https://www.nature.com/articles/s41467-021-22457-w#Fig1), and similar figures in the Supplementary, also highlight the direction of error for each model in each location. When looking across iterations of forecasts, a wide range of variation can be observed for nearly all of the models. Nevertheless, in many locations, models largely reached consensus regarding trajectories in June–August 2020. Models diverged again when predicting trajectories for November 2020–February 20201, as some models predicted upticks related to seasonality, while others projected continued slow declines in mortality.

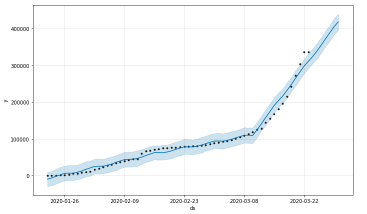
****

Fig represents the total number of death according to the years

The most recent version of each model is shown on the top left, as well as 95% prediction intervals when available. The middle row shows all iterations of each model as separate lines. The vertical dashed lines indicate the first and last model release date for each model. The bottom row shows all errors calculated at weekly intervals (circles). The top right panel summarises all observed errors, using median error (top) and median absolute error (bottom), by weeks of forecasting and month of model estimation. Errors incorporate an intercept shift to account for differences in each model’s input data. This figure represents an example for the United States of country-specific plots made for all locations examined in this study. Graphs for all geographies can be found in the [Supplementary Information](https://www.nature.com/articles/s41467-021-22457-w#MOESM1). Note that while certain models use different input data source than the other modelling groups causing apparently discordant past trends in the top-left panel. We plot raw estimates on the top-left panel; however, we implement an intercept shift to account for this issue in the calculation of errors. Delphi DELPHI-MIT (red), Los Alamos Nat Lab Los Alamos National Laboratory (blue), Youyang Gu (orange), Imperial   Imperial College London (peach), SIKjalpha USC SIKJ-alpha (pink), IHME Institute for Health Metrics and Evaluation (green), UCLA-ML UCLA Statistical Machine Learning Lab (purple).Comparison of cumulative mortality forecasts.

As the research community looks towards the “what now?” surrounding CPMs for Covid-19, it is important that we carefully utilise the available scientifically-sound evidence where possible. Specifically, it is entirely conceivable that emerging models (such as the ISARIC 4C prediction model will show adequate predictive performance results in data similar to that in which the model was developed, but external validation shows poor transferability of the models to new demographics (e.g. new countries) and statistical populations. In this situation, the community should build-upon such models, instead of developing de-novo models in distinct populations. For example, such existing Covid-19 models could be updated and refined using data in other populations, and, thereby, model transferability facilitated. A careful assessment and in-depth study of population characteristics can help find out the best approach to adapt a model to a new context, characterised by its own demographic, clinical, and epidemiological covariates. Repeatedly developing new models from scratch in distinct populations wastes prior information and risks overfitting. In contrast, model updating uses the existing models as a foundation, and builds upon this with the new data so that they are tailored to populations of interest. Finally, an important consideration for all Covid-19 models developed to-date is that they are derived in the context of current care. However, current care continues to change rapidly as the pandemic unfolds. This means that outputs from Covid-19 CPMs need to be interpreted carefully: when, how, and where are question words to be posed to better interpret the potential inference and application of a model. Specifically, the predictions reveal the risk under the applied practices observed within the development dataset. They cannot be used to inform an individual's risk under various competing intervention. A potential solution could be the exploration of counterfactual prediction in which the risk is estimated under fixed care regimes. This separates the baseline risk and the actions taken to mitigate the risk, thereby enabling users to answer “what-if” questions surrounding the impact of different interventions on Covid-19 risk in a particular setting [[**18**](https://erj.ersjournals.com/content/early/2020/10/08/13993003.03728-2020#ref-18)]. Incorporating counterfactual prediction into the modelling might also increase the chances of model transferability across populations.

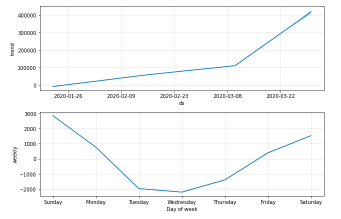
****

Fig show the cases of weekely analysis

1. **CONCLUSION:**

**9.1.** **CONCLUSION AND** **FUTURE ENHANCEMENTS**

Modeling will show us which features, and which combination of features, will be

good predictors of the number of cases. However, it is important to remember that it is not the current weather that determines the number of covid-19 cases. The next post will look into determining the monthly trend . After that, I will describe methods to use historical weather to predict the current amount of Covid-19 fever cases.

**APPENDICES**

**A**.**1** **SAMPLE** **SCREENS**

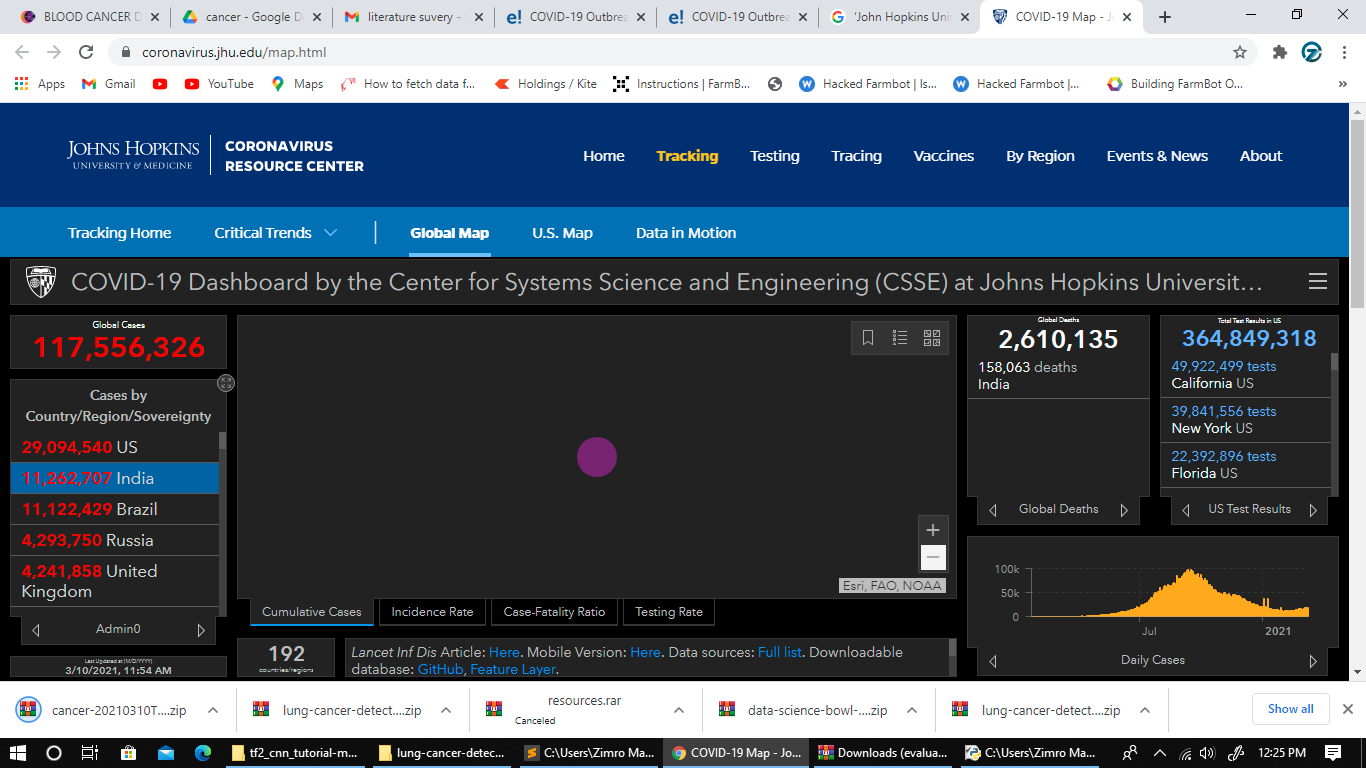
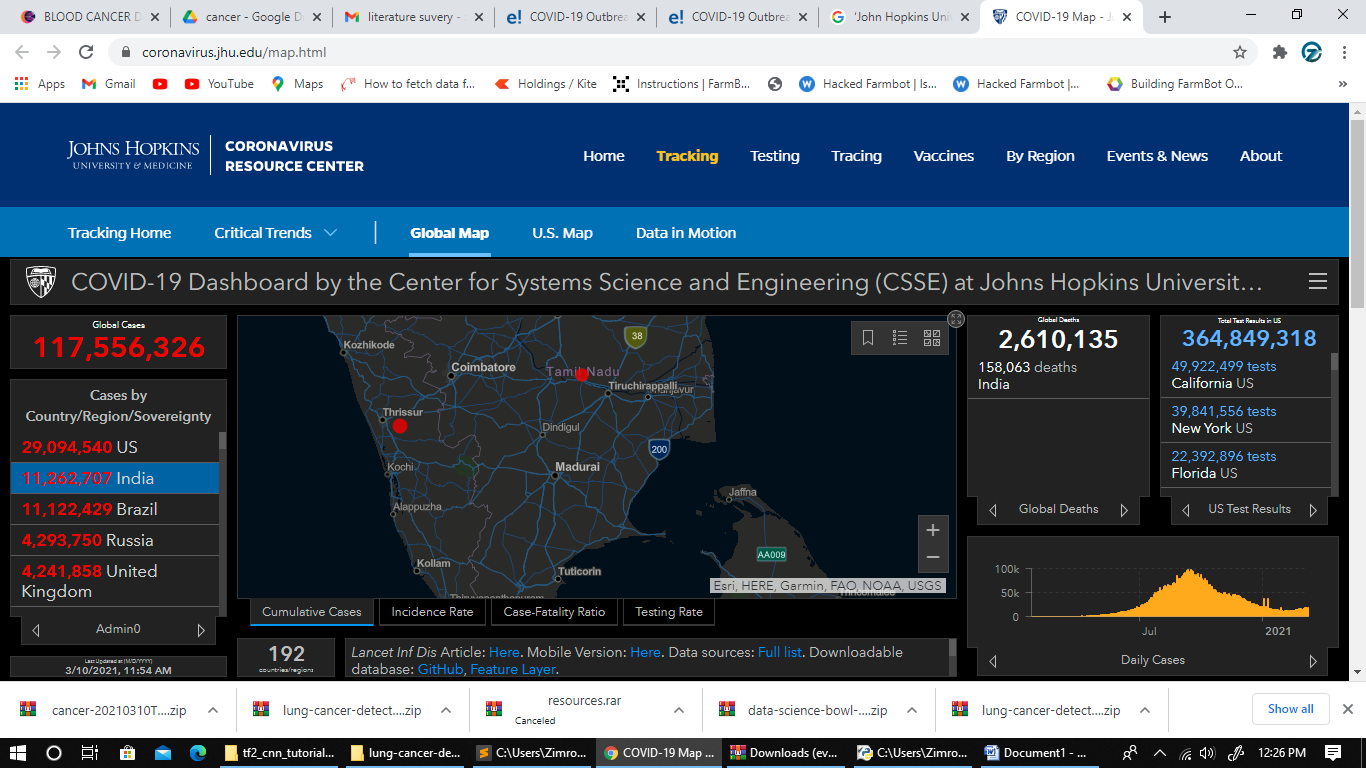


FIG:9.1 represents the cumulative cases



Fig**:** 9.2 cases in the states of india.

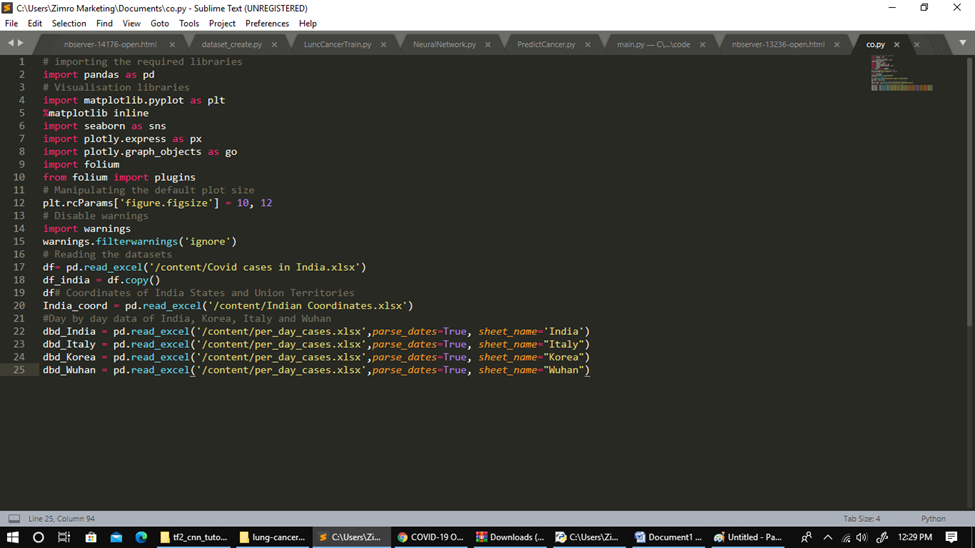


Fig:9.3 sample process

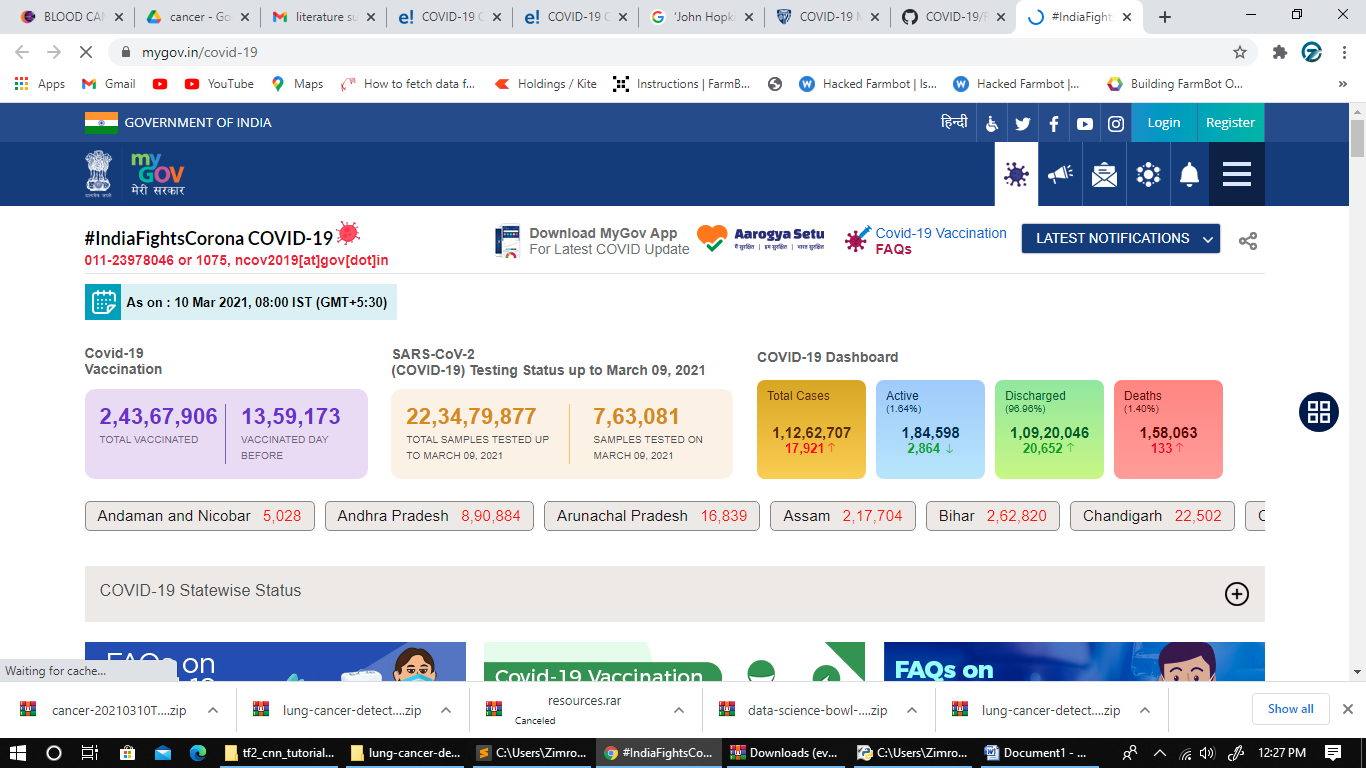
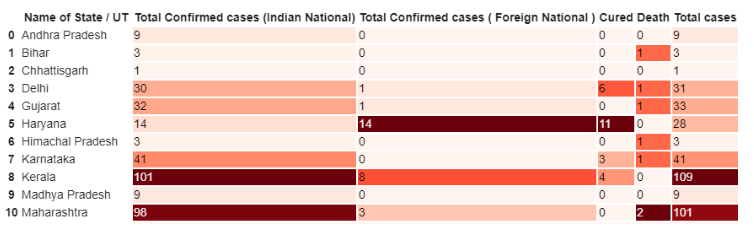


Fig:9.4 Total number of vaccinated people and covid dashboard



**Fig: 9.5 represents the confirmed cases in national**

**A**.**19 PUBLICATIONS**

PREDICTION OF COVID-19 USING SUPERVISED MACHINE LEARNING ALGORITHMS- K Kiruthika, Harshitha P, Maddina sireesha - **wwwijcrt.org (IJCRT-2105007) © 2021 IJCRT MAY 2021, Volume 9, Issue 5.**

**REFERENCES**

1. https://data.gov.in/resources/stateut-wise-details-dengue-cases-country

2015-and-2017-ministry-health-and-family-welfare.

1. https://data.gov.in/catalog/dengue-serotyping-pcr-direct-rna-sample

month-june-2013-rajiv-gandhi-centre-biotechnology

1. https://data.world/chihsuan/tw-dengue-fever
2. <https://www.tycho.pitt.edu/dataset/IN.38362002/>
3. <http://biogps.org/dataset/tag/classic%20dengue/>
4. <https://www.viprbrc.org/brc/epiDataSets.spg?decorator=flavi>
5. Global strategy for Dengue prevention and control 2012-2020. World

Health Organization; 2012.

1. Organization, “Coronavirus,” 2020, <https://www.who.int/health-topics/coronavirus#tab=tab_1>.View at: [Google Scholar](https://scholar.google.com/scholar_lookup?title=Coronavirus&author=W.%20H.%20Organization&publication_year=2020)
2. Barkur and G. B. K. Vibha, “Sentiment analysis of nationwide lockdown due to COVID 19 outbreak: evidence from India,” *Asian Journal of Psychiatry*, vol. 51, Article ID 102089, 2020.View at: [Publisher Site](https://doi.org/10.1016/j.ajp.2020.102089) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Sentiment%20analysis%20of%20nationwide%20lockdown%20due%20to%20COVID%2019%20outbreak:%20evidence%20from%20India&author=G.%20Barkur%20&author=G.%20B.%20K.%20Vibha&publication_year=2020)
3. N. F. F. da Silva, L. F. S. Coletta, E. R. Hruschka, and E. R. Hruschka Jr., “Using unsupervised information to improve semi-supervised tweet sentiment classification,” *Information Sciences*, vol. 355-356, pp. 348–365, 2016.View at: [Publisher Site](https://doi.org/10.1016/j.ins.2016.02.002) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Using%20unsupervised%20information%20to%20improve%20semi-supervised%20tweet%20sentiment%20classification&author=N.%20F.%20F.%20da%20Silva&author=L.%20F.%20S.%20Coletta&author=E.%20R.%20Hruschka&author=&author=E.%20R.%20Hruschka%20Jr.&publication_year=2016)
4. N. Coletta, “An ensemble classification system for twitter sentiment analysis,” *Procedia Computer Science*, vol. 132, pp. 937–946, 2018.View at: [Google Scholar](https://scholar.google.com/scholar_lookup?title=An%20ensemble%20classification%20system%20for%20twitter%20sentiment%20analysis&author=N.%20Coletta&publication_year=2018)
5. W. Medhat, A. Hassan, and H. Korashy, “Sentiment analysis algorithms and applications: a survey,” *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014.View at: [Publisher Site](https://doi.org/10.1016/j.asej.2014.04.011) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Sentiment%20analysis%20algorithms%20and%20applications:%20a%20survey&author=W.%20Medhat&author=A.%20Hassan&author=&author=H.%20Korashy&publication_year=2014)
6. J. Samuel, G. G. M. N. Nawaz Ali, M. M. Rahman, E. Esawi, and Y. Samuel,
7. Apache Kafka, “Apache Kafka,” 2020, [https://kafka.apache.org](https://kafka.apache.org/).View at:Google Scholar
8. S. Das, R. K. Behera, M. Kumar, and S. K. Rath, “Real-time sentiment analysis of witter streaming data for stock prediction,” *Procedia Computer Science*, vol. 132, pp. 956–964, 2018.View at: [Publisher Site](https://doi.org/10.1016/j.procs.2018.05.111) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Real-time%20sentiment%20analysis%20of%20twitter%20streaming%20data%20for%20stock%20prediction&author=S.%20Das&author=R.%20K.%20Behera&author=M.%20Kumar&author=&author=S.%20K.%20Rath&publication_year=2018)
9. L. R. Rath, S. D. Shetty, and S. Deepak Shetty, “Streaming big data analysis for real-time sentiment based targeted advertising,” *International Journal of Electrical and*
10. *Computer Engineering (IJECE)*, vol. 7, no. 1, p. 402, 2017.View at: [Publisher Site](https://doi.org/10.11591/ijece.v7i1.pp402-407) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Streaming%20big%20data%20analysis%20for%20real-time%20sentiment%20based%20targeted%20advertising&author=L.%20R.%20Rath&author=S.%20D.%20Shetty&author=&author=S.%20Deepak%20Shetty&publication_year=2017)
11. Hager, E. M. G. Younis, A. Hendawi, and A. A. Ali, “Heart disease identificatin from patients’ social posts, machine learning solution on Spark,” *Future Generation Computer Systems*, vol. 111, pp. 714–722, 2020.View at: [Publisher Site](https://doi.org/10.1016/j.future.2019.09.056) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=Heart%20disease%20identification%20from%20patients%E2%80%99%20social%20posts,%20machine%20learning%20solution%20on%20Spark&author=A.%20Hager&author=E.%20M.%20G.%20Younis&author=A.%20Hendawi&author=&author=A.%20A.%20Ali&publication_year=2020)
12. N. D. Younis, N. Y. Hashim, Y. M. Mohialden, M. A. Mohammed, T. Sutikno, and A. H. Ali, “A real-time big data sentiment analysis for iraqi tweets using spark streaming,
13. ” *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 4, pp. 1411–1419, 2020.View at: [Publisher Site](https://doi.org/10.11591/eei.v9i4.1897) | [Google Scholar](https://scholar.google.com/scholar_lookup?title=A%20real-time%20big%20data%20sentiment%20analysis%20for%20iraqi%20tweets%20using%20spark%20streaming&author=N.%20D.%20Younis&author=N.%20Y.%20Hashim&author=Y.%20M.%20Mohialden&author=M.%20A.%20Mohammed&author=T.%20Sutikno&author=&author=A.%20H.%20Ali&publication_year=2020)
14. D. Hashim, “A spark‐based big data analysis framework for real‐time sentimen prediction on streaming data,” *Software: Practice and Experience*, vol. 49, no. 9, pp. 1352–1364, 2019.View at:  Google ScholarN. K. Rajput, B. A. Grover,