

COVID-19 FUTURE FORECASTING USING SUPERVISED MACHINE LEARNING MODELS

Submitted in partial fulfillment of the requirements for the award of Bachelor
of Engineering degree in Computer Science and Engineering

by

KAPAKAYALA NAGA SAI KRISHNA VINAY SWAMI (Reg. No. 38110234)

RAGAM SAI PAVAN (Reg. No. 38110438)



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

SATHYABAMA

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

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MAY - 2022



SATHYABAMA

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

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KAPAKAYALA NAGA SAI KRISHNA VINAY SWAMI (Reg. No. 38110234) RAGAM SAI PAVAN (Reg. No. 38110438)** who carried out the project entitled "**COVID-19 Future Forecasting Using Supervised Machine Learning Models**" under my supervision from November 2021 to April 2022.

**Internal Guide
Dr. P. Sardar Maran**

External Guide (if Applicable)

**Head of the Department
Dr. L. LAKSHMANAN M.E., Ph.D.,**

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **KAPAKAYALA NAGA SAI KRISHNA VINAY SWAMI, RAGAM SAI PAVAN** hereby declare that the Project Report entitled **COVID-19 Future Forecasting Using Supervised Machine Learning Models** done by me under the guidance of **Dr. P. Sardar Maran (Internal)** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE: 23-03-2022

K. Vinay Swami

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to Dr. T. Sasikala M.E., Ph.D, Dean, School of Computing Dr. L. Lakshmanan M.E., Ph.D. , and Dr. S. Vigneshwari M.E., Ph.D. Head of the Department, Dept. of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Dr. P. Sardar Maran for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

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

ABSTRACT

The outbreak of the Novel Coronavirus or the COVID-19 in various parts of the world has affected the world as a whole and caused millions of deaths. This remains an ominous warning to public health and will be marked as one of the greatest pandemics in world history. This paper aims to provide a better understanding of how various Machine Learning models can be implemented in real-world situations. Apart from the analysis done on the world figures, this paper also analyzes the current trend or pattern of Covid-19 transmission in India. With the help of datasets from the Ministry of Health and Family Welfare of India, this study puts forward various trends and patterns experienced in different parts of the world. The data to be studied has been obtained for 154 days i.e. from January 22, 2020, till June 24, 2020. For future references, the data can be further analyzed, and more results can be obtained.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	i
	LIST OF FIGURES	v
	LIST OF TABLES	vi
1	INTRODUCTION	1
	1.1 Proposed Algorithms	1
	1.1.1 Polynomial Regression	1
2	LITERATURE SURVEY	5
	2.1 LITERATURE SURVEY	5
3	AIM AND SCOPE OF THE PRESENT INVESTIGATION	10
	3.1 Purpose	10
	3.2 Scope	10
	3.3 Existing system	10
	3.4 Proposed system	11
	3.5 Software Environment	11
4	EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED	18
	4.1 Input Design	18
	4.2 Output Design	18
	4.3 Data Flow Diagram	19
	4.4 UML Diagrams	23
	4.5 Goals	24
	4.6 Modules	24
	4.6.1 Data collection	24
	4.6.2 Expected Spread And Forecasting Validation	25
	4.6.3 Model Comparison With The Seird Model	25
	4.6.4 Covid-19 Forecasting	25
		25

	4.6.5 Accurate Forecasting	26
	4.7 System Architecture	26
	4.8 System Requirements	
5	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	28
	5.1 Test Plan	28
	5.2 Verification	28
	5.3 Validation	28
	5.4 Basics of Software Testing	28
	5.5 Black box Testing	28
	5.6 White Box Testing	29
	5.7 Types of Testing	29
	5.7.1 Unit Testing	29
	5.7.2 Integration Testing	29
	5.7.3 Functional Testing	30
	5.7.4 System Testing	30
	5.7.5 Stress Testing	30
	5.7.6 Performance Testing	30
	5.7.7 Usability Testing	30
	5.7.8 Acceptance Testing	31
	5.7.9 Regression Testing	31
	5.8 Requirement Analysis	31
	5.9 Functional Requirements	31
	5.10 Non-Functional Requirements	33
6	SUMMARY AND CONCLUSION	35
	REFERENCES	36
	APPENDIX	37
	A. SOURCE CODE	37-42
	B. SCREENSHOTS	42-44

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1.1	Linear Model, Polynomial Model	3
1.2	Multiple Linear Regression Model	4
4.1	Bubble Chart	21
4.2	Work Flow Diagram	22
4.3	Prediction	23
4.4	System Architecture	26
6.1	No. Of Corona Virus Confirmed Cases	42
6.2	No. Of Corona Virus Confirmed Cases Per Country	43
6.3	Login Page	43
6.4	Welcome Page	44

LIST OF TABLES

SL NO.	TABLE NAME	PAGE NO.
3.1	Methods	16

CHAPTER-1

INTRODUCTION

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 modeling, voice, and image processing. ML algorithms' learning is typically based on trial and error method quite opposite of conventional algorithms, which follows the programming instructions based on decision statements like if-else. One of the most significant areas of ML is forecasting, numerous standard ML algorithms have been used in this area to guide the future course of actions needed in many application areas including weather forecasting, disease forecasting, stock market forecasting as well as disease prognosis. Various regression and neural network models have wide applicability in predicting the conditions of patients in the future with a specific disease. There are lots of studies performed for the prediction of different diseases using machine learning techniques such as coronary artery disease, cardiovascular disease prediction, and breast cancer prediction. In particular, the study is focused on live forecasting of COVID-19 confirmed cases and study is also focused on the forecast of COVID-19 outbreak and early response. These prediction systems can be very helpful in decision making to handle the present scenario to guide early interventions to manage these diseases very effectively. The most challenging aspect of its spread is that a person can possess the virus for many days without showing symptoms. The causes of its spread and considering its danger, almost all the countries have declared either partial or strict lockdowns throughout the affected regions and cities. Medical researchers throughout the globe are currently involved to discover an appropriate vaccine and medications for the disease. Since there is no approved medication till now for killing the virus so the governments of all countries are focusing on the precautions which

can stop the spread. Out of all precautions, "be informed" about all the aspects of COVID-19 is considered extremely important. To contribute to this aspect of information, numerous researchers are studying the different dimensions of the pandemic and produce the results to help humanity.

1.1 PROPOSED ALGORITHMS

1.1.1 Polynomial Regression

- Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below:

$$y = b_0 + b_1x_1 + b_2x_1^2 + b_3x_1^3 + \dots + b_nx_1^n$$

- It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.
- It is a linear model with some modification in order to increase the accuracy.
- The dataset used in Polynomial regression for training is of non-linear nature.
- It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.
- Hence, "In Polynomial regression, the original features are converted into Polynomial features of required degree (2,3,...,n) and then modeled using a linear model."

Need for Polynomial Regression:

The need of Polynomial Regression in ML can be understood in the below points:

- If we apply a linear model on a **linear dataset**, then it provides us a good result as we have seen in Simple Linear Regression, but if we apply the same model without any modification on a **non-linear dataset**, then it will produce a drastic output. Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.
- So for such cases, **where data points are arranged in a non-linear fashion, we need the Polynomial Regression model**. We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.

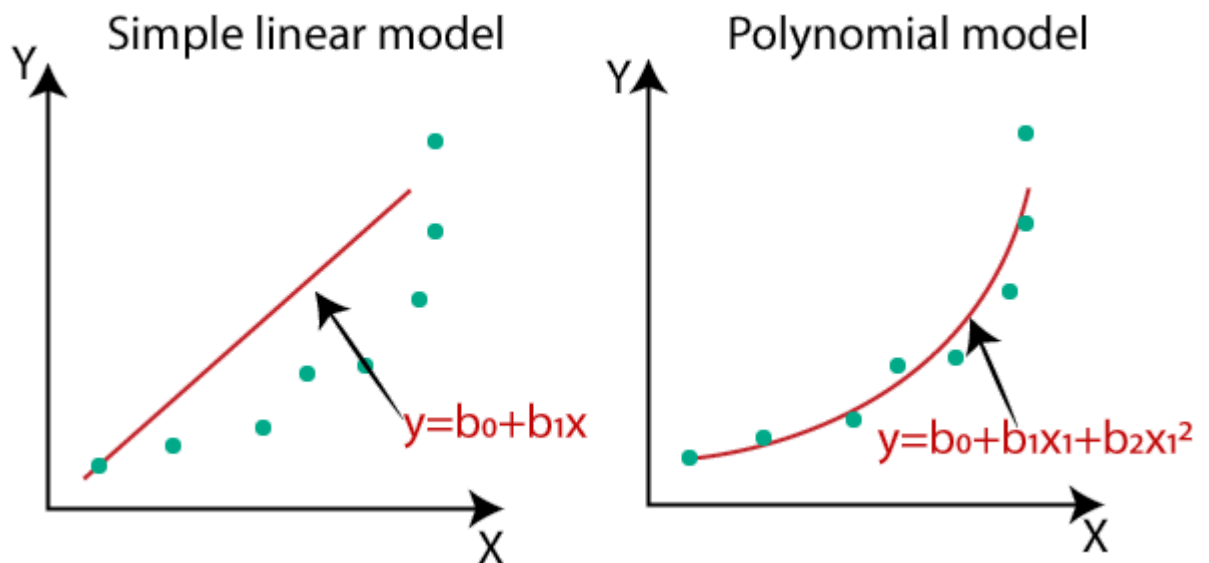


Fig:1.1: Linear Model, Polynomial model

- In the above image, we have taken a dataset which is arranged non-linearly. So if we try to cover it with a linear model, then we can clearly see that it hardly covers any data point. On the other hand, a curve is suitable to cover most of the data points, which is of the Polynomial model.
- Hence, if the datasets are arranged in a non-linear fashion, then we should use the Polynomial Regression model instead of Simple Linear Regression.

Note: A Polynomial Regression algorithm is also called Polynomial Linear Regression because it does not depend on the variables, instead, it depends on the coefficients, which are arranged in a linear fashion.

Equation of the Polynomial Regression Model:

Simple Linear Regression eq: $y = b_0 + b_1x$ (a)

Multiple Linear Regression eq: $y = b_0 + b_1x + b_2x_2 + b_3x_3 + \dots + b_nx_n$ (b)

Polynomial Regression eq: $y = b_0 + b_1x + b_2x^2 + b_3x^3 + \dots + b_nx^n$ (c)

When we compare the above three equations, we can clearly see that all three equations are Polynomial equations but differ by the degree of variables. The Simple and Multiple Linear equations are also Polynomial equations with a single degree, and the Polynomial regression equation is Linear equation with the nth degree. So, if we add a degree to our linear equations, then it will be converted into Polynomial Linear equations.

Multiple Linear Regression

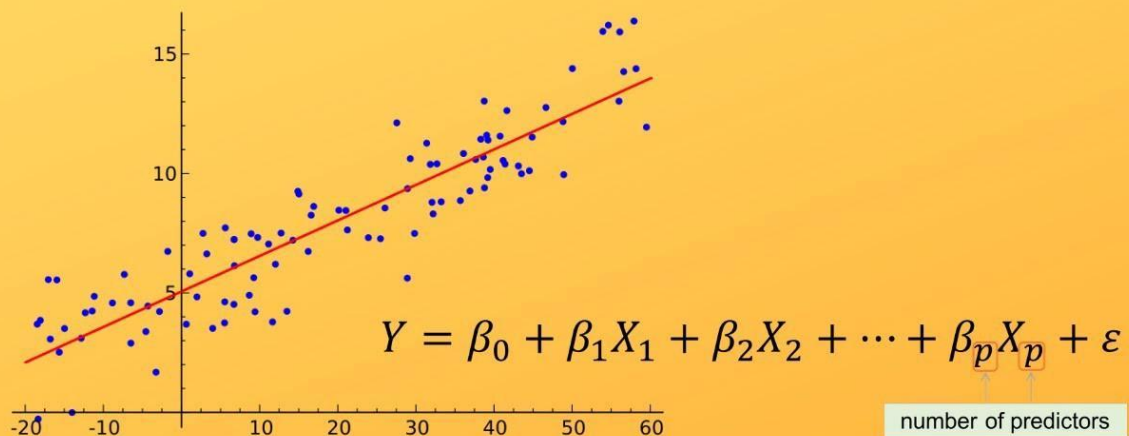


Fig:1.2: Multiple Linear Regression Model

CHAPTER-2

LITERATURE REVIEW

2.1 LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

[1] COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms

The catastrophic outbreak of Severe Acute Respiratory Syndrome - Coronavirus (SARS-CoV-2) also known as COVID-2019 has brought the worldwide threat to the living society. The whole world is putting incredible efforts to fight against the spread of this deadly disease in terms of infrastructure, finance, data sources, protective gears, life-risk treatments and several other resources. The artificial intelligence researchers are focusing their expertise knowledge to develop mathematical models for analyzing this epidemic situation using nationwide shared data. To contribute towards the well-being of living society, this article proposes to utilize the machine learning and deep learning models with the aim for understanding its everyday exponential behaviour along with the prediction of future reachability of the COVID-2019 across the nations by utilizing the real-time information from the Johns Hopkins dashboard.

[2] COVID-19 in India: State-wise Analysis and Prediction

The highly infectious coronavirus disease (COVID-19) was first detected in Wuhan, China in December 2019 and subsequently spread to 212 countries and territories around the world, infecting millions of people. In India, a large country of about 1.3 billion people, the disease was first detected on January 30, 2020, in a student returning from Wuhan. The total number of confirmed infections in India as of May 3, 2020, is more than 37,000 and is currently growing fast.

[3] COVID-19 Outbreak Prediction with Machine Learning

Several outbreak prediction models for COVID-19 are being used by officials around the world to make informed-decisions and enforce relevant control measures. Among

the standard models for COVID-19 global pandemic prediction, simple epidemiological and statistical models have received more attention by authorities, and they are popular in the media. Due to a high level of uncertainty and lack of essential data, standard models have shown low accuracy for long-term prediction. Although the literature includes several attempts to address this issue, the essential generalization and robustness abilities of existing models needs to be improved.

This paper presents a comparative analysis of machine learning and soft computing models to predict the COVID-19 outbreak. Among a wide range of machine learning models investigated, two models showed promising results (i.e., multi-layered perceptron, MLP, and adaptive network-based fuzzy inference system, ANFIS). Based on the results reported here, and due to the highly complex nature of the COVID-19 outbreak and variation in its behavior from nation-to-nation, this study suggests machine learning as an effective tool to model the outbreak.

[4] Forecasting the novel coronavirus COVID-19

What will be the global impact of the novel coronavirus (COVID-19)? Answering this question requires accurate forecasting the spread of confirmed cases as well as analysis of the number of deaths and recoveries. Forecasting, however, requires ample historical data. At the same time, no prediction is certain as the future rarely repeats itself in the same way as the past.

Moreover, forecasts are influenced by the reliability of the data, vested interests, and what variables are being predicted. Also, psychological factors play a significant role in how people perceive and react to the danger from the disease and the fear that it may affect them personally.

This paper introduces an objective approach to predicting the continuation of the COVID-19 using a simple, but powerful method to do so. Assuming that the data used

is reliable and that the future will continue to follow the past pattern of the disease, our forecasts suggest a continuing increase in the confirmed COVID-19 cases with sizable associated uncertainty.

The risks are far from symmetric as underestimating its spread like a pandemic and not doing enough to contain it is much more severe than overspending and being over careful when it will not be needed. This paper describes the timeline of a live forecasting exercise with massive potential implications for planning and decision making and provides objective forecasts for the confirmed cases of COVID-19.

[5] Prediction of Spreads of COVID-19 in India from Current Trend

The article describe modelling efforts for evaluating the current level of COVID-19 infections in India, using exponential model. The Data from 15 march 2020 to 30 April 2020 are used for validating the model, where intrinsic rise rate is kept constant. It is observed that some states of India, like Maharastra, Gujarat and Delhi have a much higher daily infection cases.

This is modelled by assuming an initial higher infections, keeping rise rate same. The sudden outbursts are captured using offset of values for these three states. Data from other states like Madhya Pradesh, Uttar Pradesh and Rajasthan are also analysed and they are found to be following the same constants as India is following. Worldwide, many attempts are made to predict outburst of COVID-19 and in the model, described in this paper, turning point is not predicted, as cases in India are still rising. The developed model is based on daily confirmed infections and not on cumulative infections and rationalization is carried out for the population of various regions, while predicting infections for various states.

Assigning a decay constant at this stage will be a premature exercise and keeping that in mind, exponential model predicts that India will attain 1 lakh case by 15 May 2020. The figure of 2 lakh and 3 lakh will be attained on 22 May 2020 and 26 May 2020, respectively.

[6] An interpretable mortality prediction model for COVID-19 patients

The sudden increase in COVID-19 cases is putting high pressure on healthcare services worldwide. At this stage, fast, accurate and early clinical assessment of the disease severity is vital. To support decision making and logistical planning in healthcare systems, this study leverages a database of blood samples from 485 infected patients in the region of Wuhan, China, to identify crucial predictive biomarkers of disease mortality.

For this purpose, machine learning tools selected three biomarkers that predict the mortality of individual patients more than 10 days in advance with more than 90% accuracy: lactic dehydrogenase (LDH), lymphocyte and high-sensitivity C-reactive protein (hs-CRP). In particular, relatively high levels of LDH alone seem to play a crucial role in distinguishing the vast majority of cases that require immediate medical attention. This finding is consistent with current medical knowledge that high LDH levels are associated with tissue breakdown occurring in various diseases, including pulmonary disorders such as pneumonia. Overall, this Article suggests a simple and operable decision rule to quickly predict patients at the highest risk, allowing them to be prioritized and potentially reducing the mortality rate.

CHAPTER-3

AIM AND SCOPE OF THE PRESENT INVESTIGATION

3.1 PURPOSE

The purpose of this document is forecasting covid-19 future using machine learning algorithms. In detail, this document will provide a general description of our project, including user requirements, product perspective, and overview of requirements, general constraints. In addition, it will also provide the specific requirements and functionality needed for this project - such as interface, functional requirements and performance requirements.

3.2 SCOPE

The scope of this SRSdocument persists for the entire life cycle of the project. This document defines the final state of the software requirements agreed upon by the customers and designers. Finally at the end of the project execution all the functionalities may be traceable from the SRSto the product. The document describes the functionality, performance, constraints, interface and reliability for the entire life cycle of the project.

3.3 EXISTING SYSTEM

Information from news articles played an important role in empowering citizens during an epidemic. However, our analysis found more negative articles than positive articles which is a concern. (2548 vs. 561) The word clouds corresponding to the negative and positive statements displayed similar words. A content analysis of text mined Ebola on news articles and scientific publications showed similar findings

3.4 PROPOSED SYSTEM

The paper also presented a comprehensive study of the spread of the virus outbreak situation in India which will further help in taking necessary steps to manage the huge population of India. For the same, two Machine Learning models were used but in future Deep Learning models or hybrid two or models can be used to forecast the further spread of the virus. Real-time data has been used for this analysis and not the conventional data because the data validity of real time data highly depends on the time and its response time requirements come from the external world. Although the real time data can be relaxed in a few cases unlike the conventional data which

needs to satisfy every case, real time data is approximately correct with performance metrics calculated as a number of transactions missing their deadlines per unit time. Through our analysis can be concluded that the Polynomial Regression Algorithm.

3.5 SOFTWARE ENVIRONMENT

Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.

- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python <https://www.python.org>.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.

- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

Type the following text at the Python prompt and press the Enter –

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result –

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file –

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

This produces the following result –

Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods –

Table 3.1 : Methods

Sl. No	Methods & Description
1	GET Sends data in unencrypted form to the server. Most common method.
2	HEAD Same as GET, but without response body
3	POST Used to send HTML form data to server. Data received by POST method is not cached by server.
4	PUT Replaces all current representations of the target resource with the uploaded content.
5	DELETE Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of 'nm' parameter obtained from the form data is obtained by –

```
user = request.form['nm']
```

It is passed to **'/success'** URL as variable part. The browser displays a **welcome** message in the window.

Change the method parameter to **'GET'** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of 'nm' parameter is now obtained by –

```
User = request.args.get('nm')
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to 'nm' parameter is passed on to **'/success'** URL as before.

CHAPTER-4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

4.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?

- How the data should be arranged or coded?

- The dialog to guide the operating personnel in providing input.

- Methods for preparing input validations and steps to follow when error occur.

4.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is

determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the Future.

- Signal important events, opportunities, problems, or warnings.

- Trigger an action.

- Confirm an action.

4.3 DATA FLOW DIAGRAM

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

5. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
6. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
7. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
8. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

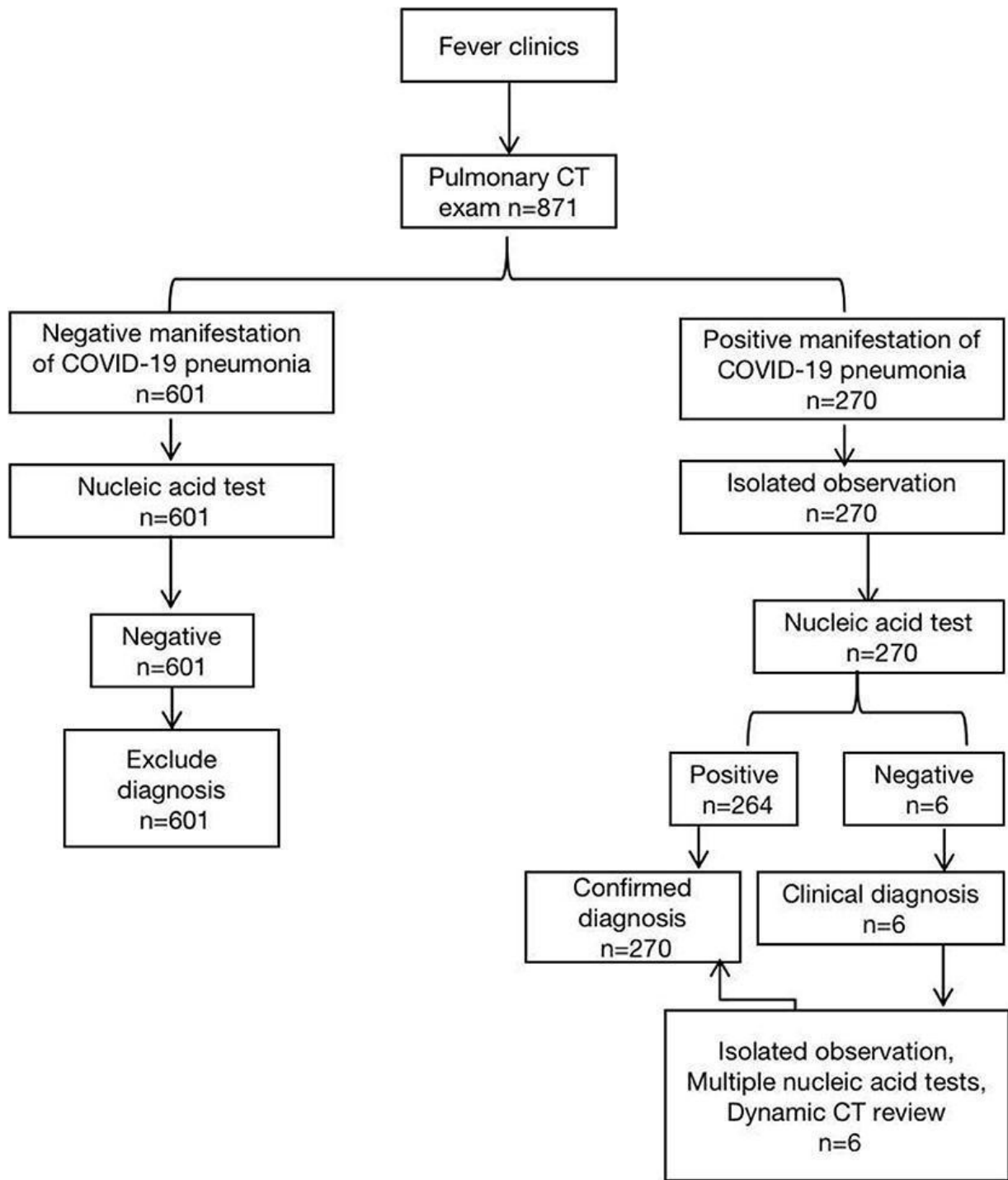


Fig:4.1: Bubble Chart

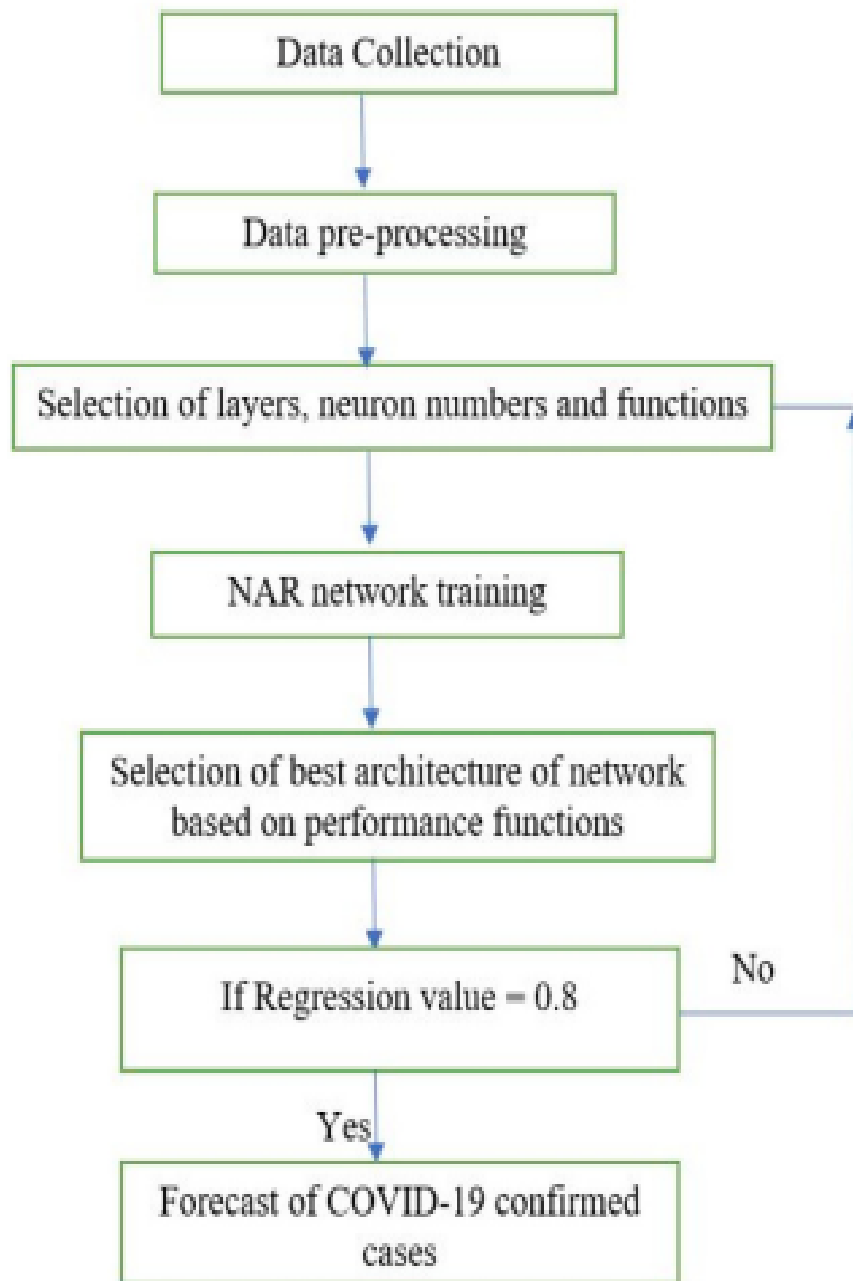


Fig:4.2: Work Flow Diagram

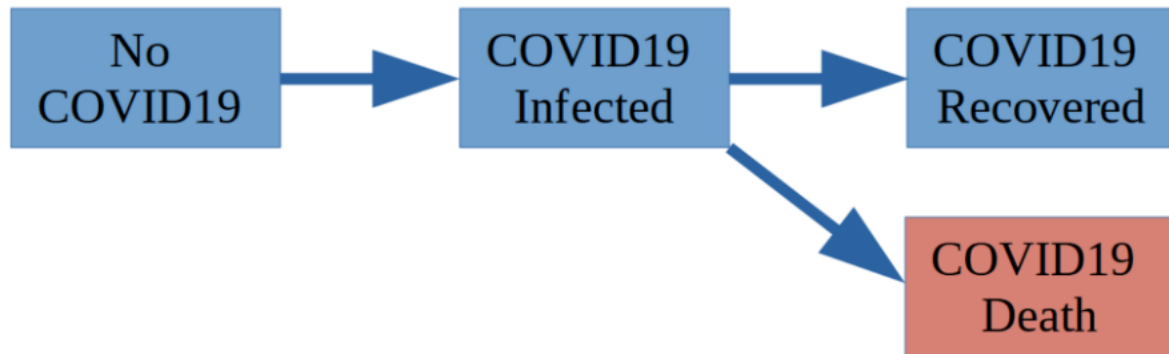


Fig:4.3: Prediction

4.4 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.5 GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

4.6 MODULES

- Data collection
- EXPECTED SPREAD AND FORECASTING VALIDATION
- MODEL COMPARISON WITH THE SEIRD MODEL
- Covid-19 forecasting
- Accurate forecasting

4.6.1 Data collection

we present a generalized model to characterize the COVID-19 pandemic composed with eight different states or phenotypes. These states are supposed to characterize all the life cycle of an infected case before the infection, during the infection, and after being discharged, either deceased or recovered. Each state in our model is supposed to characterize, at a given unit of time t , in our case day, a specific population's behaviour as follows.

4.6.2 Expected Spread And Forecasting Validation

we extrapolate the COVID-19 spread in the studied countries using our fitted models to forecast its evolution during the next period. We also discuss in details the evolution of each state. In addition, we provide the upper and lower bounds of the expected spread representing the worst and best cases scenarios for these countries based on our model.

4.6.3 Model Comparison With The Seird Model

we can deduce the effect of the weights on the forecast performance. In fact, the forecast of the number of infected cases proves that the weighting technique helps in improving the forecast of the observed data but it will reduce the efficiency of the fitting. Indeed, if the model is trained to perfectly follow the training data, then the model will over-fit the training data and it will perform badly during the forecast.

4.6.4 Covid-19 Forecasting

We have proposed a mechanistic model for assessing and forecasting the spread of the COVID-19 pandemic. The pandemic of interest is very novel and even though it has already spread all over the world, scientists still do not have enough knowledge to model it properly, especially due to the lack of data and evidence about its contamination process.

4.6.5 Accurate Forecasting

The fitting technique is represented along with the model that aims to endorse its forecasting results by focusing on the last trends of the data, and hence, giving accurate results that are on the verge of these trends. Multiple countries with different spread trends are investigated with the proposed work. We reveal that our model is able to fit the real world data of different countries showing different spread trends. It is shown that accurate forecasting results are obtained.

4.7 SYSTEM ARCHITECTURE

The system architectural design is the design process for identifying the subsystems making up the system and framework for subsystem control and communication. The goal of the architectural design is to establish the overall structure of software system.

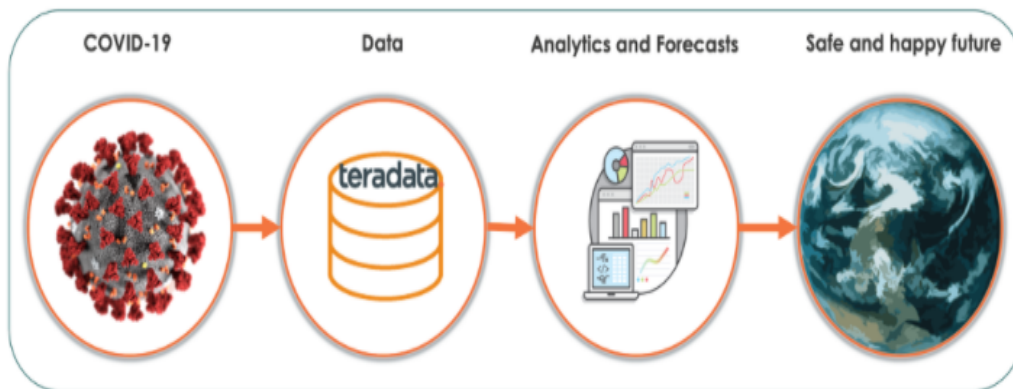


Fig:4.4: System Architecture

4.8 SYSTEM REQUIREMENTS

a) Hardware Requirements

System : Pentium Dual Core.
 Hard Disk : 120 GB.
 Monitor : 15"LED
 Input Devices : Keyboard, Mouse
 Ram : 1GB.

b) Software Requirements

Operating system : Windows 7.
 Coding Language : python
 Toolkit : Jupitor Notebook
 DATABASE : EXCEL

CHAPTER-5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

5.1 TEST PLAN

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a

software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

5.2 VERIFICATION

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

5.3 VALIDATION

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

5.4 BASICS OF SOFTWARE TESTING

There are two basics of software testing: black box testing and white box testing

5.5 BLACK BOX TESTING

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

5.6 WHITE BOX TESTING

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black

box testing is often used for validation and white box testing is often used for verification.

5.7 TYPES OF TESTING

There are many types of testing like

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing

5.7.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

5.7.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

5.7.3 Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

5.7.4 System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

5.7.5 Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

5.7.6 Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

5.7.7 Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

5.7.8 Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

5.7.9 Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing

5.8 REQUIREMENT ANALYSIS

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analyzing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

5.9 FUNCTIONAL REQUIREMENTS

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

Flexibility

The flexibility of the project is provided in such a way that is has the ability to run on different environments being executed by different users.

Safety

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

5.10 NON- FUNCTIONAL REQUIREMENTS

Portability

It is the usability of the same software in different environments. The project can be run in any operating system.

Performance

These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

Accuracy

The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

Maintainability

Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyse, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This

is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the

client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

CHAPTER-6

SUMMARY AND CONCLUSION

In this study, we developed a mechanistic model composed of eight states to characterize the COVID-19 spread in different countries. An ODE system is formulated to mathematically model the interactions between the different states. A curve fitting using real world observed data sets is developed to determine accurate fitting solutions and estimate the parameters of the ODE model. We use the LM algorithm to solve and estimate the ODE system parameters. We have shown that the employed algorithms outperform a reference model, i.e., BFGS. The fitting technique is represented along with the model that aims to endorse its forecasting results by focusing on the last trends of the data, and hence, giving accurate results that are on the verge of these trends. Multiple countries with different spread trends are investigated with the proposed work.

REFERENCES

- [1] Features, Evaluation and Treatment Coronavirus (COVID-19) . Available: <https://www.ncbi.nlm.nih.gov/books/NBK554776/>, May 18 (2020).
- [2] N. S Pun, S. K. Sonbhadra, S. Agarwal, “COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms”, medRxiv, Available: June 1 (2020).
- [3] P. Ghosh, R. Ghosh, B. Chakraborty, “COVID-19 in India: State-wise Analysis and Prediction”, medRxiv, Available: doi: <https://doi.org/10.1101/2020.04.24.20077792>, May 19 (2020).
- [4] S. F. Ardabili, A. Mosavi, P. Ghamisi, F. Ferdinand, A. R. VarkonyiKoczy, U.Reuter, T. Rabczuk, P. M. Atkinson, “COVID-19 Outbreak Prediction with Machine Learning”, Available at SSRN: <https://ssrn.com/abstract=3580188> or <http://dx.doi.org/10.2139/ssrn.3580188>, April 19 (2020).

- [5] F. Petropoulos, S. Makridakis, “Forecasting the novel coronavirus COVID-19”, Available: <https://doi.org/10.1371/journal.pone.0231236>, Published: March 31 (2020)
- [6] Arti. M. K, K. Bhatnagar, “Modeling and Predictions for COVID 19 Spread in India”, Available:, Published: April (2020).
- [7] H. Shekhar, “Prediction of Spreads of COVID-19 in India from Current Trend”, medRxiv, Available: doi:, May 06, (2020).
- [8] Yan, L., Zhang, H., Goncalves, J. et al. “An interpretable mortality prediction model for COVID-19 patients”. Nature Machine Intelligence 2, pp. 283–288, Published: 14 May (2020).
- [9] L. Li, Z. Yang, Z. Dang, C. Meng, J. Huang, H. Meng, D. Wang, G. Chen, J. Zhang, H. Peng, Y. Shao, “Propagation analysis and prediction of the COVID-19”, Infectious Disease Modelling vol. 5, Available:
- [10] S. Zhao, H. Chen, “Modeling the epidemic dynamics and control of COVID-19 outbreak in China”. Quantitative Biology, vol. 8, Issue March (2020).
- [11] J. Xie, Z. Tong, X. Guan. et al. “Critical care crisis and some recommendations during the COVID-19 epidemic in China”. Intensive Care Med, vol.6 Issue 6, Available:, June (2020).

APPENDIX:

A. SOURCE CODE:

```
import pandas as pd
import numpy as np
from sklearn import preprocessing, cross_decomposition, model_selection, metrics
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
%matplotlib inline
import dateutil
from tqdm import tqdm
from sklearn import linear_model
```

```

import datetime
import numpy as np
import pandas as pd

import plotly.graph_objects as go
from ipywidgets import widgets
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels import tsa
import numpy as np
import pandas as pd

import plotly.graph_objects as go
from ipywidgets import widgets

```

```

confirmedGlobal['NewContryCode']=confirmedGlobal['Country/Region']+confirmedGlobal['Province/State'].fillna('')
deathGloabal['NewContryCode']=deathGloabal['Country/Region']+deathGloabal['Province/State'].fillna('')
recoverGlobal['NewContryCode']=recoverGlobal['Country/Region']+recoverGlobal['Province/State'].fillna(' ')
recoverGlobal

```

```

welcome_input = ("hello", "hi", "greetings", "sup", "what's up","hey",)
welcome_response = ["hi", "hey", "*nods*", "hi there", "hello", "I am glad! You are talking to me"]
def welcome(user_response):
    for word in user_response.split():
        if word.lower() in welcome_input:
            return random.choice(welcome_response)

```

```

X_train_confirmed, X_test_confirmed, y_train_confirmed, y_test_confirmed = train_test_split(days_since_1_1, world_cases, test_size=0.20, shuffle=False)
poly = PolynomialFeatures(degree=2)
poly_X_train_confirmed = poly.fit_transform(X_train_confirmed)
poly_X_test_confirmed = poly.fit_transform(X_test_confirmed)
poly_future_forecast = poly.fit_transform(future_forecast)
linear_model = LinearRegression(normalize=True, fit_intercept=False)
linear_model.fit(poly_X_train_confirmed, y_train_confirmed)
test_linear_pred = linear_model.predict(poly_X_test_confirmed)
linear_pred = linear_model.predict(poly_future_forecast)
print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
print('MSE:', mean_squared_error(test_linear_pred, y_test_confirmed))

```

```

plt.plot(y_test_confirmed)
plt.plot(test_linear_pred)

```

```

plt.legend(['Test Data', 'Polynomial Regression Predictions'])
linear_model.score(poly_X_test_confirmed, y_test_confirmed)
svm_confirmed = SVR(shrinking=True, kernel='poly', gamma=0.01,
epsilon=1, degree=5, C=0.1)
svm_confirmed.fit(X_train_confirmed, y_train_confirmed)
svm_pred = svm_confirmed.predict(future_forecast)
svm_test_pred = svm_confirmed.predict(X_test_confirmed)
plt.plot(y_test_confirmed)
plt.plot(svm_test_pred)
plt.legend(['Test Data', 'SVM Predictions'])
print('MAE:', mean_absolute_error(svm_test_pred, y_test_confirmed))
print('MSE:', mean_squared_error(svm_test_pred, y_test_confirmed))

```

```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import random
import math
import time
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error
import datetime
import operator
plt.style.use('fivethirtyeight')
%matplotlib inline

```

```

def prediction(x):
    X_train_confirmed, X_test_confirmed, y_train_confirmed, y_test_confirmed =
train_test_split(days_since_1_1, x, test_size=0.2, shuffle=False)
    # transform data for polynomial regression
    poly = PolynomialFeatures(degree=2)
    poly_X_train_confirmed = poly.fit_transform(X_train_confirmed)
    poly_X_test_confirmed = poly.fit_transform(X_test_confirmed)
    poly_future_forecast = poly.fit_transform(future_forecast)

    # polynomial regression
    linear_model = LinearRegression(normalize=True, fit_intercept=False)
    linear_model.fit(poly_X_train_confirmed, y_train_confirmed)
    test_linear_pred = linear_model.predict(poly_X_test_confirmed)
    global linear_pred
    linear_pred = linear_model.predict(poly_future_forecast)
    print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
    print('MSE:', mean_squared_error(test_linear_pred, y_test_confirmed))

```

```

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

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

```

```

country_vaccine = data_df.groupby(["country", "iso_code",
"vaccines"])[['total_vaccinations',

'total_vaccinations_per_hundred',

'daily_vaccinations',

'daily_vaccinations_per_million',

'people_vaccinated',

'people_vaccinated_per_hundred',

'people_fully_vaccinated', 'people_fully_vaccinated_per_hundred'

]].max().reset_index()
country_vaccine.columns = ["Country", "iso_code", "Vaccines", "Total
vaccinations", "Percent", "Daily vaccinations",
'Daily vaccinations per million', "People
vaccinated", "People vaccinated per hundred",
'People fully vaccinated', 'People fully vaccinated
percent']

```

```

vaccine =
data_df.groupby(["vaccines"])[['total_vaccinations','total_vaccinations_per_hun
dred',

```

```

'daily_vaccinations', 'daily_vaccinations_per_million'].max().reset_index()
vaccine.columns = ["Vaccines", "Total vaccinations", "Percent", "Daily
vaccinations",
                    "Daily vaccinations per million"]
def draw_trace_bar_vaccine(data, feature, title, xlab, ylab, color='Blue'):
    data = data.sort_values(feature, ascending=False)
    trace = go.Bar(
        x = data['Vaccines'],
        y = data[feature],
        marker=dict(color=color),
        text=data['Vaccines']
    )
    data = [trace]

    layout = dict(title = title,
                  xaxis = dict(title = xlab, showticklabels=True, tickangle=45,
                              zeroline=True, zerolinewidth=1,
                              zerolinecolor='grey',
                              showline=True, linewidth=2, linecolor='black',
                              mirror=True,
                              tickfont=dict(
                                  size=10,
                                  color='black')),),
                  yaxis = dict(title = ylab, gridcolor='lightgrey', zeroline=True,
                              zerolinewidth=1, zerolinecolor='grey',
                              showline=True, linewidth=2, linecolor='black',
                              mirror=True),
                  plot_bgcolor = 'rgba(0, 0, 0, 0)', paper_bgcolor = 'rgba(0, 0,
0, 0)',
                  hovermode = 'closest',
                  height = 800
    )
    fig = dict(data = data, layout = layout)
    iplot(fig, filename='draw_trace')

```

Save the following script as login.html

```

<html>
<body>
<form action="http://localhost:5000/login" method="post">
<p>Enter Name:</p>
<p><input type="text" name="nm"/></p>
<p><input type="submit" value="submit"/></p>
</form>
</body>
</html>

```

Now enter the following script in Python shell.

```
from flask import Flask, redirect, url_for, request
app = Flask(__name__)
@app.route('/success/<name>')
def success(name):
    return 'welcome %s' % name
@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        user = request.form['nm']
        return redirect(url_for('success', name=user))
    else:
        user = request.args.get('nm')
        return redirect(url_for('success', name=user))
if __name__ == '__main__':
    app.run(debug=True)
```

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

B. SCREENSHOTS

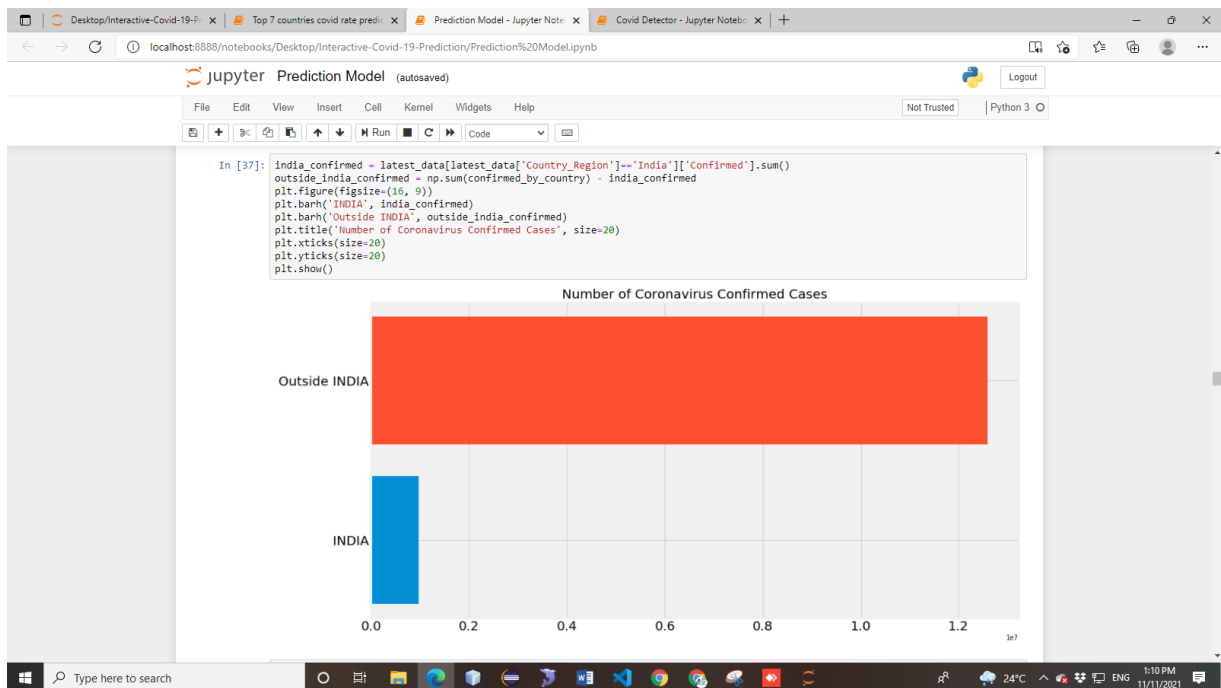


Fig:6.1 No. Of Corona Virus Confirmed Cases

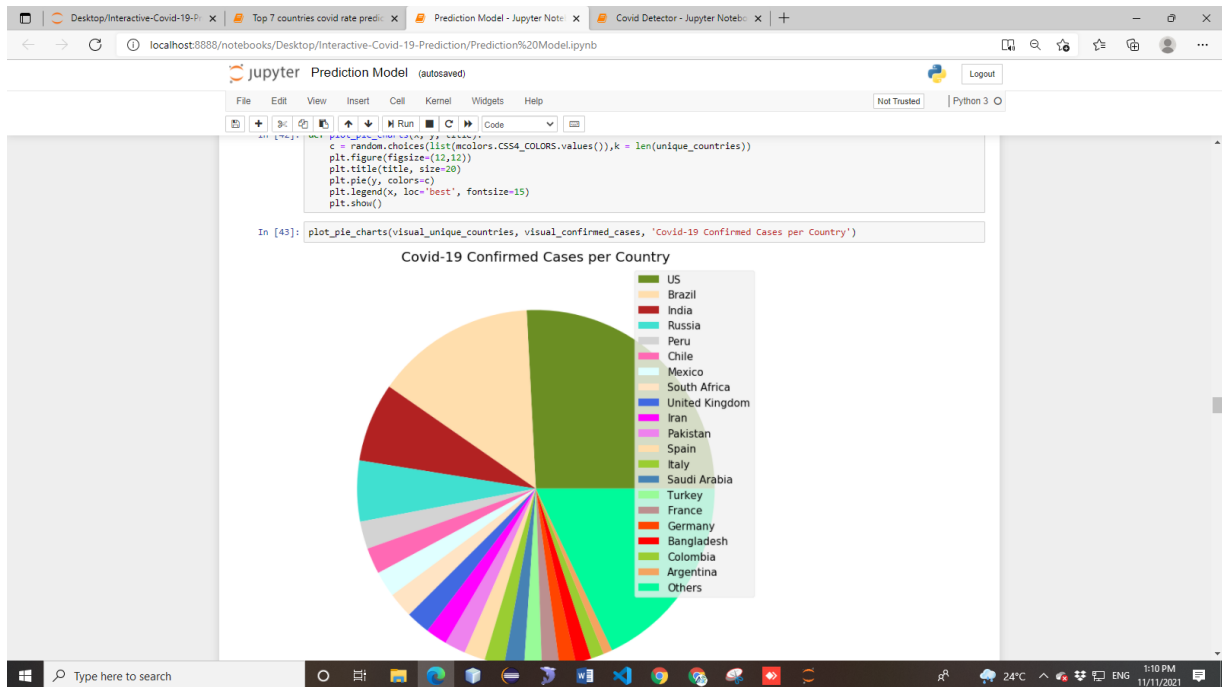


Fig:6.2 No. Of Corona Virus Confirmed Cases Per Country



Fig:6.3 Login Page

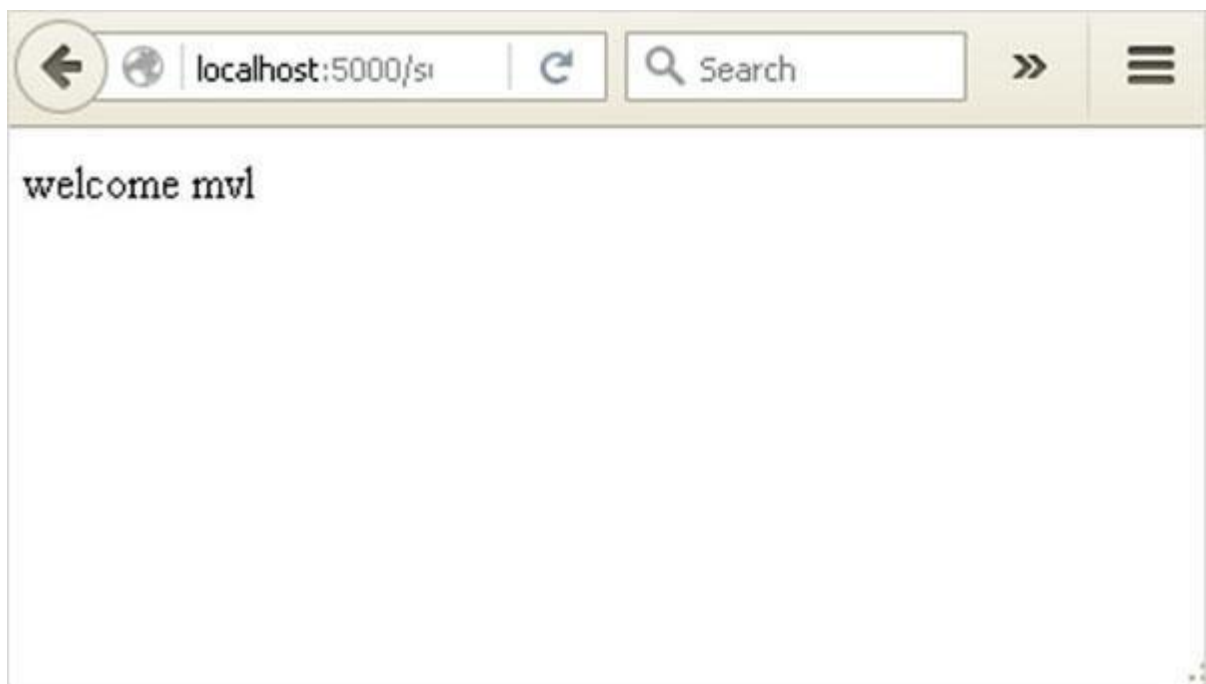


Fig:6.4 Welcome Page

MACHINE LEARNING APPROACH FOR CLUSTERING OF COUNTRIES TO IDENTIFY THE BEST STRATEGIES TO COMBAT COVID-19.docx

ORIGINALITY REPORT

9%

SIMILARITY INDEX

7%

INTERNET SOURCES

8%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

1

www.medrxiv.org

Internet Source

4%

2

Submitted to Asia Pacific University College of Technology and Innovation (UCTI)

Student Paper

1%

3

Ashish Menon, Nithin K Rajendran, Anish Chandrachud, Girish Setlur. "Modelling and simulation of COVID-19 propagation in a large population with specific reference to India", Cold Spring Harbor Laboratory, 2020

Publication

1%

4

its.nure.ua

Internet Source

1%

5

papers.ssrn.com

Internet Source

1%

6

Hamdi Friji, Raby Hamadi, Hakim Ghazzai, Hichem Besbes, Yehia Massoud. "A Generalized Mechanistic Model for Assessing

<1%

and Forecasting the Spread of the COVID-19 Pandemic", IEEE Access, 2021

Publication

7

journals.plos.org

Internet Source

<1 %

8

annalsofrscb.ro

Internet Source

<1 %

9

Piyush Kumar. "What impacts have geographical locations on the cases and deaths from COVID-19/SARS-CoV-2 pandemic in 36 states and union territories of India:- observational analysis in India - V2", Research Square Platform LLC, 2021

Publication

<1 %

Exclude quotes On

Exclude matches Off

Exclude bibliography On