COVID-19 DATA ANALYSIS

Minor project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

ARPITA RANI(221030055)
TANISH MATTOO(221030041)
BHAVYA(221030180)
ABHISHEK SINGH(221030184)

UNDER THE SUPERVISION OF

PROF. DR. VIVEK SEHGAL

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, 173234, Himachal Pradesh, INDIA

TABLE OF CONTENT

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Prof. Dr. Vivek Sehgal , Department of CSE & IT ,** Jaypee University of Information Technology. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

Prof. Dr. Vivek Sehgal

Professor and Head

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology

Submitted by:

Arpita Rani (221030055)

Tanish Mattoo (221030041)

Bhavya (221030180)

Abhishek Singh (221030184)

Computer Science & Engineering Department

Jaypee University of Information Technology

CERTIFICATE

This is to certify that the work which is being presented in the project report titled *COVID -19 Data Analysis* in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by *Arpita Rani, 221030055 , Tanish Mattoo , 221030041, Bhavya, 221030180, Abhishek Singh(221030184)* during the period from January 2025 to May 2025 under the supervision of *Prof. Dr. Vivek Sehgal* , Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

Arpita Rani (221030055) Arpita Rani
Tanish Mattoo (221030041) Tanish
Bhavya (221030180) Bhavya
Abhishek Singh (221030184) Abhishek

The above statement made is correct to the best of my knowledge.

Vivek Sehgal
5/5/25.

Prof. Dr. Vivek Sehgal
Professor & Head
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Waknaghat

iii

ACKNOWLEDGEMENT

Firstly, We express our heartiest thanks and gratefulness to almighty God for His divineblessing makes us possible to complete the project work successfully.

We are really grateful and wish our profound our indebtedness to Supervisor **Prof. Dr. Vivek Sehgal, Professor & Head,** Department of CSE Jaypee University of Information Technology,Wakhnaghat. Deep Knowledge & keen interest of our supervisor in the field of "**Data Analysis**" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to **Prof. Dr. Vivek Sehgal,** Department of CSE, for his kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straight forwardly or in a roundabout way in making this project a win. In this unique situation, We might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated our undertaking.

Finally, We must acknowledge with due respect the constant support and patients of our parents.

Arpita Rani (221030055)
Tanish Mattoo (221030041)
Bhavya (221030180)
Abhishek Singh (221030184)

ABSTRACT

The COVID-19 pandemic brought about significant challenges in the whole world , that makes it crucial for analyze related data for insights and future preparedness. This project that we have made focused on examining a real-world dataset containing COVID-19 case information of India, aiming to uncover trends and develop predictive models.

We started by preparing the dataset using Python and Microsoft Excel. This step involved cleaning the data by first removing duplicate entries, addressing missing values, and standardizing column names ensuring smooth analysis. The initial stage of analysis involved tracking the progression of confirmed cases, recoveries, and fatalities over time. Through line charts, we visualized how the virus spread and how its impact changed, offering a clear picture of the situation's evolution.

Next, we transitioned into predictive modeling using various machine learning techniques. We implemented **Linear Regression** to estimate the number of deaths based on confirmed cases, which helped in identifying a trend line that indicated how fatalities increased in relation to rising infections. Following this, **Logistic Regression** was used in classifying whether a given day experienced a high death toll. We used confirmed and recovered cases as input features, allowing the model for estimating the likelihood of a day recording elevated death counts.

Further, we experimented with **Support Vector Machines (SVM)** and **K-Nearest Neighbors (KNN)** for better classification. With SVM, we applied **Principal Component Analysis (PCA)** for dimensionality reduction, which enabled us to visualize the model's ability to distinguish between different classes. The **KNN** model was employed to predict if a day recorded an above-average number of confirmed cases using variables like deaths, recoveries, and foreign cases.All models were trained and validated using standard evaluation techniques. We assessed their performance with tools such as confusion matrices, classification reports, and accuracy scores to ensure reliability.

In summary, this project leveraged real-time COVID-19 data and applied data preprocessing, visualization, and machine learning techniques to understand patterns and forecast potential scenarios. Such analyses are crucial in supporting data-driven decision-making during health crises.

Chapter 01: INTRODUCTION

1.1 Introduction

Globally, the COVID-19 pandemic severely affected economies, public health systems and daily life. Controlling the virus spread and understanding its constantly shifts in effects proved to be extremely difficult, especially in India. Real-time data interpretation became very essential for healthcare officials and the general public to take well-informed action as the number of cases rapidly increased across states and union territories. The large amount of data produced during the pandemic highlighted the significance of sophisticated visualization tools and predictive modeling techniques in spotting significant trends and patterns.

By creating an extensive COVID-19 data analysis platform particularly for India, this project seeks to fulfill that need. Developing an interactive, user-friendly dashboard in real-time data on verified COVID-19 cases, recoveries, and deaths is the aim. The platform provides a simple to comprehend picture of the virus's change over time via visual tools like pie charts, bar charts, and line graphs.

The project uses machine learning models, specifically K-Nearest Neighbors (KNN) and Support Vector Machine (SVM), in addition to visual analytics to predict and categorize case trends using historical data. When these algorithms are integrated into an organized pipeline for data analysis, they enhance the capacity for decoding complex trends and promote better choices.

By combining visual storytelling with predictive analysis, the project offers a comprehensive perspective on the pandemic's impact in India. This approach not only helps to gain a deeper understanding of the situation but also supports more informed decision-making in the realm of public health.

1.2 Objective

The primary objective of this project is to conduct analysis of COVID-19 case data specific to India for providing interactive visualizations and leveraging machine learning techniques for predictive insights. As the virus spread quickly across the country, timely data interpretation and forecasting became crucial for an effective response. This project meets that need by utilizing comprehensive dataset that includes confirmed cases, recoveries, deaths and other relevant variables organized by state and date. A key goal is to create a user-friendly and interactive dashboard using modern web technologies like ReactJS

and Recharts. The dashboard will allow users to explore COVID-19 trends through intuitive and engaging visualizations such as charts and graphs making complex data easier to understand and aiding in tracking the virus progression.

Another key objective is to apply supervised machine learning algorithms namely Linear Regression, Logistic Regression, Support Vector Machine (SVM) and K-Nearest Neighbors (KNN)—to forecast case trends based on historical data patterns. These models are implemented in Python by the help of the scikit-learn library and run within a cloud-based environment such as Google Colab.

The project seeks to provide practical and scalable solution that integrates statistical data analysis, effective visualization, and predictive modeling. By doing so it offers valuable insights into the progression of COVID-19 in India and serves as a real-world application of machine learning in healthcare data analysis.

1.3 Motivation

The main motivation for this project arises from unprecedented impact of the COVID-19 pandemic and the essential role data analysis played in managing public health responses. As India experienced multiple waves of infection, it became clear that timely and accurate interpretation of health data was crucial for both decision-makers and the general public. The situation emphasized the need to utilize data visualization and machine learning to manage vast health datasets and identify patterns that could inform effective interventions.

This project was driven by the clear need to transform raw pandemic data into actionable insights. By integrating interactive dashboards with predictive machine learning models; it showcases how modern technology can be leveraged to analyze, forecast and visualize spread of infectious diseases. It represents a significant effort to apply data science in addressing real-world challenges with immediate societal impact.

Additionally, it project provides a valuable learning experience, offering the chance to work with advanced tools and technologies such as Python for data analysis; ReactJS for developing user interfaces, and Google Colab for cloud-based computation. It strengthens practical knowledge of key data science concepts, including data preprocessing, model training and visualization, while offering hands-on experience in end-to-end application development within a real-world context.

1.4 Language Used

This project is built using a combination of programming languages, libraries, and frameworks that are best suited for data analysis, machine learning, and interactive visualizations.

The backend and data science components are developed using **Python**, a widely-used language known for its readability and rich ecosystem. It supports efficient data manipulation, modeling, and visualization with the help of several powerful libraries. On the frontend, **ReactJS** is used to create a dynamic and responsive dashboard that visually represents real-time trends of COVID-19 data. Cloud-based tools like **Google Colab** provide the infrastructure for training machine learning models, offering convenience and computational efficiency.

The key technologies and tools used in this project are:

Python

- Used for data preprocessing, model building, and evaluation.
  - Key libraries:
    - **Pandas** – for data manipulation and handling tabular datasets
    - **Matplotlib** and **Seaborn** – for creating static and interactive visualizations
    - **Scikit-learn** – for implementing machine learning models like K-Nearest Neighbors and Random Forest

ReactJS

- Frontend library used to build interactive, component-based user interfaces
- Fetches and displays real-time COVID-19 statistics

Recharts

- A React-based charting library
- Used for creating visual components like:
    - Line charts
    - Bar graphs
    - Pie charts

Google Colab

- A cloud-based Jupyter Notebook environment
- Supports GPU acceleration
  - Facilitates running Python code and training machine learning models in the cloud

The integration of these technologies enables the development of a robust, scalable, and interactive COVID-19 data analysis platform, combining real-time visual insights with predictive capabilities.

1.5 Technical Requirements (Hardware)

To ensure efficient development, testing, and execution of this COVID-19 data analysis project, a balanced combination of suitable hardware and supporting software is essential. The system should preferably be equipped with a **multi-core processor**, such as an Intel Core i5 or a more advanced model, along with a minimum of **8 GB RAM**. This setup is sufficient to handle data preprocessing, visualization rendering, and the training of machine learning models without significant lag or interruptions.

For processes that require high computational capacity, particularly the training of machine learning models on large datasets, the use of cloud-based platforms like **Google Colab** is advisable. Google Colab offers GPU acceleration, which enhances performance and reduces local resource consumption. This is particularly beneficial when working with complex models or larger datasets.

A consistent and **high-speed internet connection** is also a vital requirement for this project. It is needed for accessing cloud services, downloading libraries and datasets, and for deploying or testing the web-based dashboard. The **ReactJS-based interface** used in this project requires a modern web browser, such as Google Chrome or Microsoft Edge, to display interactive graphs and navigate through visual data elements smoothly.

On the software side, **Python version 3.x** is required, along with essential libraries such as Pandas for data handling, Matplotlib and Seaborn for data visualization, and Scikit-learn for implementing machine learning algorithms. Additionally, **Node.js and npm** must be installed to run and manage the ReactJS frontend environment. This overall configuration ensures that the project can be developed and executed seamlessly, both on local machines and via cloud infrastructure.

1.6 Deliverables/Outcomes

The outputs of this minor project include a detailed, interactive COVID-19 dashboard for India, an integrated machine learning module, and comprehensive documentation. The dashboard, developed using ReactJS and Recharts, allows users to explore COVID-19 case trends interactively. It visualizes key metrics such as daily confirmed cases, recoveries, and deaths, both on a national and state level. The dashboard integrates various data visualization components, including bar charts, pie charts, and line graphs, providing users with an intuitive overview of pandemic trends.

Complementing the dashboard is a Google Colab notebook containing well-documented implementations of machine learning algorithms such as Linear Regression , Logistic Regression , Support Vector Machine (SVM) and K-Nearest Neighbors. These models are used to predict and classify COVID-19 case trends based on the provided dataset, offering valuable predictive analytics alongside the visual trends displayed in the dashboard.

In addition, the final project submission will include a complete project ZIP file containing the dataset, dashboard source code, Colab notebook, and setup instructions. The expected outcomes of this project are enhanced data interpretability, forecasted trend analysis, and improved ease of use for monitoring pandemic statistics. This project demonstrates the real-world application of data science and machine learning for public health analysis, making a significant scholarly and technical contribution.

Chapter 02: Feasibility Study, Requirements Analysis and Design

2.1 Feasibility Study

2.1.1 Problem Definition

The COVID-19 pandemic created an urgent need for timely, accurate, and informative analysis of case data to manage healthcare resources and gauge public sentiment. While many COVID-19 data repositories were available but most presented data in the form of tables or static reports, lacking interactive, user-friendly dashboards. The central issue was the absence of a good , web-based platform that not only visualized COVID-19 trends through dynamic charts but also offered predictive insights using machine learning models.

Authorities, researchers, and the public faced challenges in monitoring state-wise case developments, recovery rates, and fatality trends in real-time. The fragmented and unstructured presentation of data hindered predictive decision-making. Additionally most systems lacked the ability to predict future trends or classify case trends based on historical data. This project addresses these gaps by proposing a hybrid solution — an interactive dashboard driven by ReactJS for real-time data visualization and Python-based machine learning models (KNN and Random Forest) for predictive insights, bridging the gap between raw data and actionable decision support.

2.1.2 Problem Analysis

Upon detailed analysis; it was found that while India's COVID-19 data was accessible through government and open-source datasets, it lacked structured visualization and predictive analytics capabilities. Most dashboards presented real-time figures without offering historical context or future trend projections. Additionally, they failed to provide classification-based insights that could aid in regional health policymaking. Although data existed in CSV files, it was challenging for non-technical users to interpret without graphical or AI assistance.

Challenges included effectively handling time-series data, integrating frontend visualization tools, deploying predictive machine learning models on cloud platforms like AWS, and determining best algorithms for classification. The solution required an integrated approach that could read data from a trusted source, apply machine learning models for classification and trend prediction, and present the

results via an intuitive web interface. The system also needed to be scalable for future model updates and data additions.

2.1.3 Solution

The proposed solution is the development of an integrated COVID-19 dashboard with machine learning models for predictive data analysis. The system will be built using ReactJS for the frontend by offering a responsive and dynamic interface. COVID-19 case information will be displayed through charts and graphs generated using Recharts, enabling users to view state-wise, daily, and cumulative trends.

To enhance system functionality, K-Nearest Neighbors (KNN) and Random Forest classifiers will be implemented in Python via Google Colab. These models will be trained on the COVID-19 dataset to predict and classify trends based on confirmed cases, deaths, recoveries, and nationality-based data.

Users can interact with the dashboard to select specific states or time periods, view visual trends, and access predictive results generated by the machine learning models. The system will provide precise data visualization, real-time updates, and efficient machine learning classification. It will also display model accuracy and confusion matrices for verification. The combined solution offers a scalable, user-friendly, and robust analytical tool for public health officials, researchers, and citizens.

2.1.4 Literature Review

S.No	Title	Author	Format & Year	Purpose
1	Automatic COVID-19 prediction using explainable machine learning techniques	S. Solayman, S. A. Aumi, C. S. Merya, M. Mubassir, and R. Khan	IEEE (2020)	To analyze COVID-19 trends and patterns using data analytics
2	A benchmark study by using various machine learning models for predicting COVID-19 trends.	Kamelesun, R. Saranya, and P. Kathiravan	IEEE (2020)	To determine key factors influencing COVID-19 spread and impact
3	Covid-19 Prediction in India using Machine Learning	Sarfraj Alam, Vipul Kumar, Sweta Singh, Sweta Joshi, Madhu Kirola	IEEE (2021)	To find Computational techniques for machine learningbased models to show their

				importance for monitoring the potential threats for valueable decisions
4	Enhancing COVID-19 classification accuracy with a hybrid SVM-LR model	I. Nordin, W. A. Mustafa, M. S. Lola, E. N. Madi, A. A. Kamil, M. D. Nasution, A. A. K. Abdul Hamid, N. H. Zainuddin, E. Aruchunan, and M. T. Abdullah	IEEE (2021)	To identify crucial variables affecting virus transmission
5	Machine learning approach for data analysis and predicting coronavirus using COVID-19 India dataset.	Soni Singh, K.R. Ramkumar, Ashima Kukkar	IEEE (2023)	To do the prediction of pandemic outbreaks using machine learning and deep learning techniques.

2.2 Requirements

2.2.1 Functional Requirements

In order to make the system more functional:

- The system should allow users to see COVID-19 data state-wise and overall, by date.
- Interactive data visualization in the form of line graphs, bar charts, and pie charts should be present on the dashboard.
- The system must enable users to filter data by state, date, or case type (confirmed, recovered, deaths).
- Machine learning models (Linear Regression, Logistic Regression, SVM, KNN) need to be used to classify trends and forecast case behavior.
- The system must present model performance metrics, such as accuracy, confusion matrix, and classification reports.
- Users must be able to download or view prediction results for individual datasets.
- The dashboard must work on contemporary browsers and dynamically update according to chosen inputs.
- It must report clear, precise, and updated data without calling for technical proficiency.
- The application must be able to handle COVID-19 time-series data well.

- The system must combine data pre-processing, model training, and dashboard generation in an effortless process. Machine learning algorithms — Linear Regression, Logistic Regression, SVM, and K-Nearest Neighbors (KNN) — are developed in Python on Google Colab. The algorithms are trained on the COVID-19 data to predict and classify trends according to confirmed cases, death cases, recovery cases, and nationality-based data.
- Users engage with the dashboard to choose particular states or time periods, see visual trends, and retrieve predictive outcomes provided by the ML models. The system maintains accurate data visualization, real-time data updates, and efficient machine learning classification. It also presents model accuracy and confusion matrices for verification. The integrated solution offers a scalable, easy-to-use, and powerful analysis tool for public health professionals, researchers, and citizens.

2.2.2Non-Functional Requirements

Non-functional requirements define how the system functions but not what it performs. For this project:

The system ought to possess a clean, intuitive, and responsive UI for user ease across all devices.

- The dashboard should ensure cross-browser support and accommodate varied screen sizes.
- The system ought to smoothly handle and display moderate-sized COVID-19 datasets without lagging performance.
- It must keep the data accurate and consistent in showing or projecting case trends.
- The program must have response time quickness while handling user requests and updates of data.
- The machine learning algorithms must be scalable so they can be retramed in the future as more data is integrated.
- The application must remain cost-effective through open-source software and libraries.
- The system needs to give instantaneous feedback for the actions chosen by users on the dashboard.
- Confidentiality of data must be ensured by employing local or publicly available non-sensitive data sets.
- The dashboard must be able to accommodate future additions such as vaccination data or sophisticated prediction models.

2.3 Data-Flow Diagram (DFD)

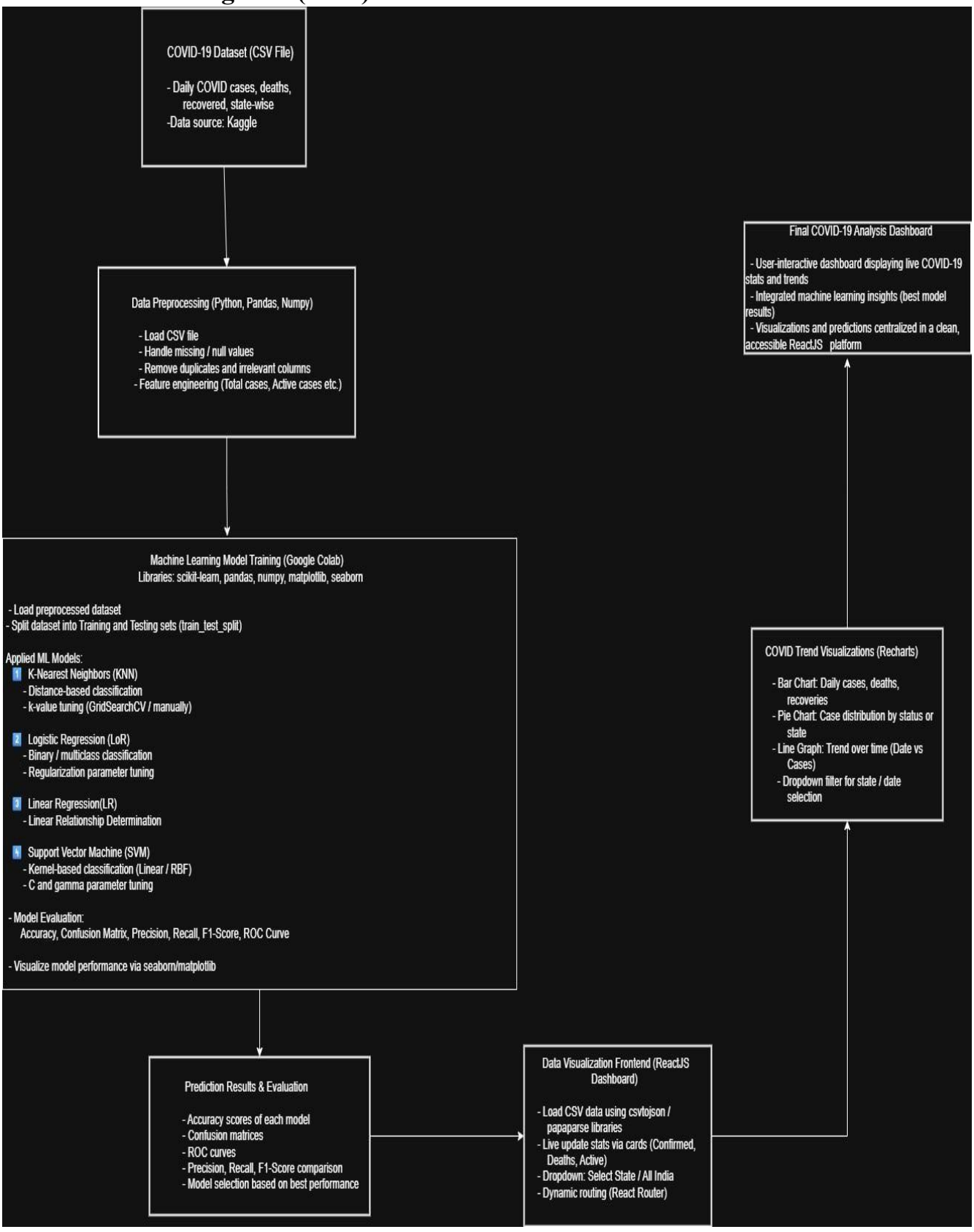


Fig. 1 : Data Flow Diagram of the Project

Fig. 1 shows the complete workflow for a COVID-19 analysis project using machine learning and data visualization. The process begins with collecting a COVID-19 dataset from Kaggle, followed by data preprocessing using Python libraries. Multiple ML models such as KNN, Logistic Regression, Linear

Regression, and SVM are trained and evaluated. The results are visualized and integrated into an interactive ReactJS dashboard for real-time trend analysis and predictions.

Chapter 03: IMPLEMENTATION

3.1 Dataset Used in the Minor Project

For the minor project, the dataset utilized was obtained from publicly available COVID-19 repositories in India. These sources offer authentic and routinely updated information on COVID-19 cases across various Indian states and union territories. The dataset, provided in CSV format, contains time-series data on confirmed, recovered, and deceased cases. It also includes supplementary information such as the nationality of patients and their respective states, which aided in improving the accuracy of classification tasks.

The dataset's structured and numeric format made it ideal for both machine learning classification and data visualization. It was processed using Python with the help of the Pandas library to handle cleaning and preprocessing. Subsequently, it was integrated into a ReactJS-based dashboard for interactive and graphical analysis. This facilitated a better understanding of daily changes, overall trends, and recovery statistics. The processed dataset was further used to train and evaluate models including Linear Regression, Logistic Regression, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), to predict future trends and classify new cases efficiently.

Table 1 : COVID-19 DATASET

A	B	C	D	E	F	G	H	I
sno	date	time	state/unic	confirmed	confirmed	cured	deaths	confirmed
1	30-01-2020	6:00 PM	Kerala	1	0	0	0	1
2	30-01-2020	6:00 PM	Kerala	1	0	0	0	1
4	02-02-2020	6:00 PM	Kerala	3	0	0	0	3
5	03-02-2020	6:00 PM	Kerala	3	0	0	0	3
6	04-02-2020	6:00 PM	Kerala	3	0	0	0	3
7	05-02-2020	6:00 PM	Kerala	3	0	0	0	3
9	07-02-2020	6:00 PM	Kerala	3	0	0	0	3
10	08-02-2020	6:00 PM	Kerala	3	0	0	0	3
11	09-02-2020	6:00 PM	Kerala	3	0	0	0	3
12	10-02-2020	6:00 PM	Kerala	3	0	0	0	3
13	11-02-2020	6:00 PM	Kerala	3	0	0	0	3
14	12-02-2020	6:00 PM	Kerala	3	0	0	0	3
16	14-02-2020	6:00 PM	Kerala	3	0	0	0	3
17	15-02-2020	6:00 PM	Kerala	3	0	0	0	3
19	17-02-2020	6:00 PM	Kerala	3	0	0	0	3
20	18-02-2020	6:00 PM	Kerala	3	0	0	0	3
21	19-02-2020	6:00 PM	Kerala	3	0	0	0	3
22	20-02-2020	6:00 PM	Kerala	3	0	0	0	3
23	21-02-2020	6:00 PM	Kerala	3	0	0	0	3
24	22-02-2020	6:00 PM	Kerala	3	0	0	0	3
25	23-02-2020	6:00 PM	Kerala	3	0	0	0	3
26	24-02-2020	6:00 PM	Kerala	3	0	0	0	3
27	25-02-2020	6:00 PM	Kerala	3	0	0	0	3
28	26-02-2020	6:00 PM	Kerala	3	0	0	0	3
29	27-02-2020	6:00 PM	Kerala	3	0	0	0	3
30	28-02-2020	6:00 PM	Kerala	3	0	0	0	3
31	29-02-2020	6:00 PM	Kerala	3	0	0	0	3
32	01-03-2020	6:00 PM	Kerala	3	0	0	0	3
33	02-03-2020	6:00 PM	Telengana	1	0	0	0	1
34	02-03-2020	6:00 PM	Kerala	3	0	0	0	3
35	02-03-2020	6:00 PM	Telengana	1	0	0	0	1
36	03-03-2020	6:00 PM	Telengana	1	0	0	0	1
38	03-03-2020	6:00 PM	Kerala	3	0	3	0	3
39	03-03-2020	6:00 PM	Delhi	1	0	0	0	1
40	04-03-2020	6:00 PM	Uttar Prad	6	0	0	0	6
41	04-03-2020	6:00 PM	Kerala	3	0	3	0	3
42	04-03-2020	6:00 PM	Haryana	0	2	0	0	2

A sample of the dataset used for the project is shown in **Table 1**, which shows daily recorded entries with fields like date, time, state/UT, confirmed cases, recoveries, and deaths. The foundation for additional preprocessing, analysis, and model training is this tabular data.

3.2 Date Set Features

3.2.1 Types of Data Set

The dataset comprises a **time-series dataset** of COVID-19 cases reported across India, categorized by **states, dates, case types, and patient details**. The dataset includes **categorical, numerical, and temporal data**. Categorical features include the state name and patient nationality, while numerical features cover counts of confirmed, recovered, and deceased cases. The dataset's temporal aspect includes the date field, tracking daily case trends from the pandemic's initial phase to its peak.

This hybrid dataset type allows the system to generate both descriptive analytics (like charts, trends) and predictive analytics (case classification and trend forecasting). Its versatility makes it suitable for supervised machine learning algorithms like **Linear Regression , Logistic Regression, Support Vector Machine(SVM) and KNN**, as well as interactive dashboards using **ReactJS and Recharts**.

3.2.2 Number of Attributes, fields, description of the data set

Data set includes the following fields:

Date: Date of reporting of the case (format: DD/MM/YYYY).

State/Union Territory: The state in which the case was reported.

ConfirmedIndianNational: Indian nationals tested positive.

ConfirmedForeignNational: Foreign nationals tested positive.

Cured: Patients who recovered.

Deaths: Number of deaths reported.

Confirmed: Total number of confirmed cases up to that date.

In total, the data contains 7 attributes/fields and has thousands of rows representing daily records for all Indian states from the early outbreak to later waves. The data is properly structured and formatted in CSV, making it usable within machine learning models and data visualization libraries.

3.3 Design of Problem Statement

The problem statement that has been designed is to create a dynamic, interactive dashboard for case analysis and prediction of COVID-19 in India and present machine learning-based predictions via classification models. The key objectives are to integrate real-time data visualizations, state-level filtering, and applying machine learning models to classify and predict the case trends.

1)The system design has three key components:

2)Data acquisition and preprocessing from the CSV dataset.

3)Interactive visualization dashboard with ReactJS and Recharts for graphical representation.

4) Predictive analysis via machine learning models (KNN & Random Forest) deployed in Python by means of Google Colab.

The issue was crafted to solve the unavailability of end-to-end tools that cover both visualization and forecasting of Indian COVID-19 cases. The system is extendable to include other datasets like vaccination data or future pandemic data, and is able to retrain its machine learning models based on new data.

3.4 Pseudo code of the Project Problem

Linear Regression(LR):

1.Load and preprocess dataset

2.Select feature variables (X) and target variable (y)

3.Split data into training and testing sets

4.Fit the Linear Regression model on training data

5.Predict target values for test data

6.Evaluate model performance using metrics

7.Visualize results

Logistic Regression (LoR):

1. Load and prepare dataset

2. Split data into train and test sets

3. Initialize Logistic Regression model

4. Train model on training data

5. Predict labels for test data

6. Evaluate model performance (accuracy, precision, recall)

Support Vector Machine (SVM):

1. Preprocess and scale dataset
2. Split into train and test sets
3. Initialize SVM classifier with linear or RBF kernel
4. Train on training data
5. Predict test labels
6. Evaluate classification metrics

K-Nearest Neighbors (KNN)

1. Load and preprocess dataset
2. Normalize feature values
3. Split data into training and testing sets
4. Select a value for K (neighbors)
5. Calculate Euclidean distance to all training samples
6. Select K nearest neighbors
7. Assign the class based on majority vote

3.5 Flow graph of the Minor Project Problem

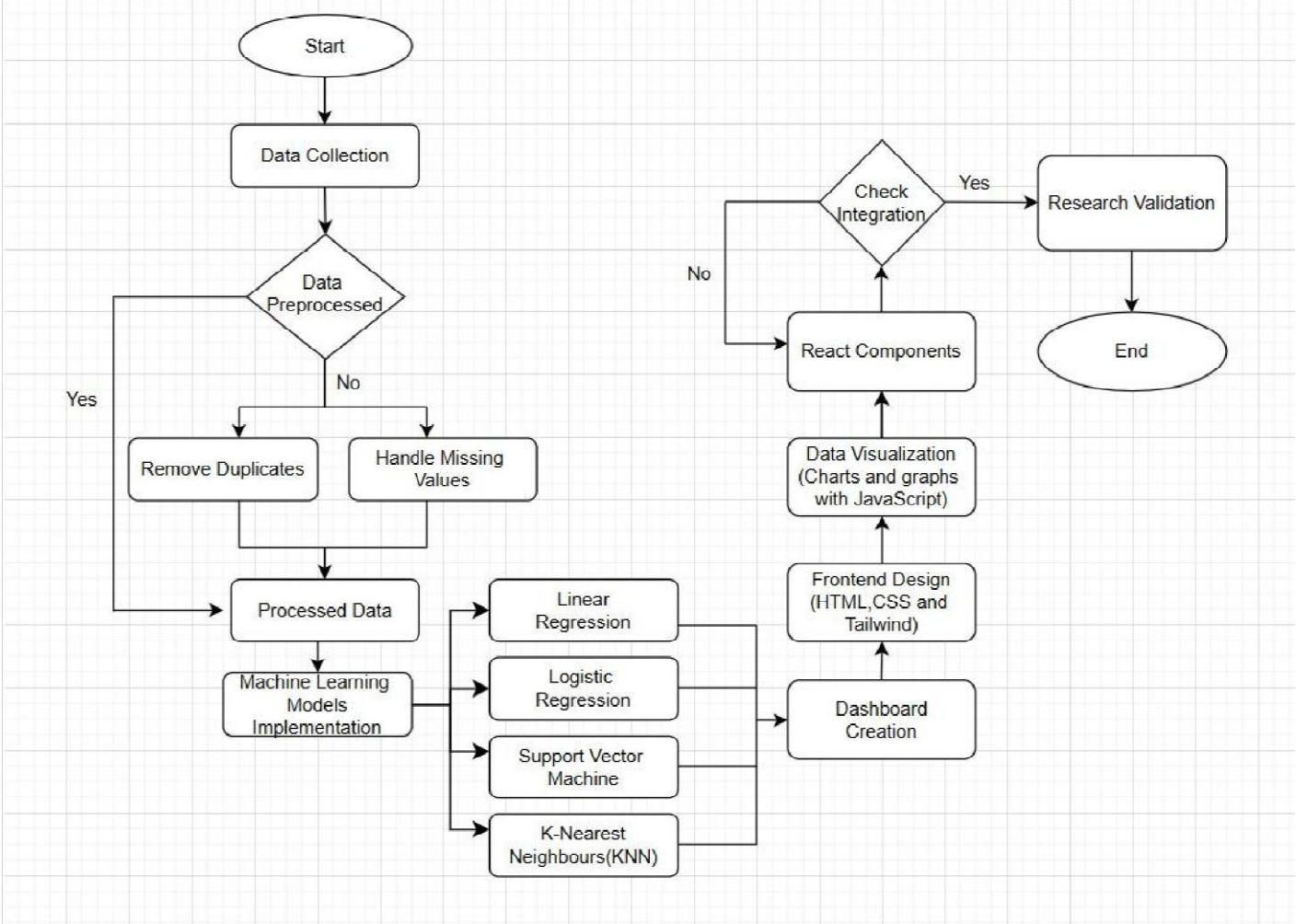
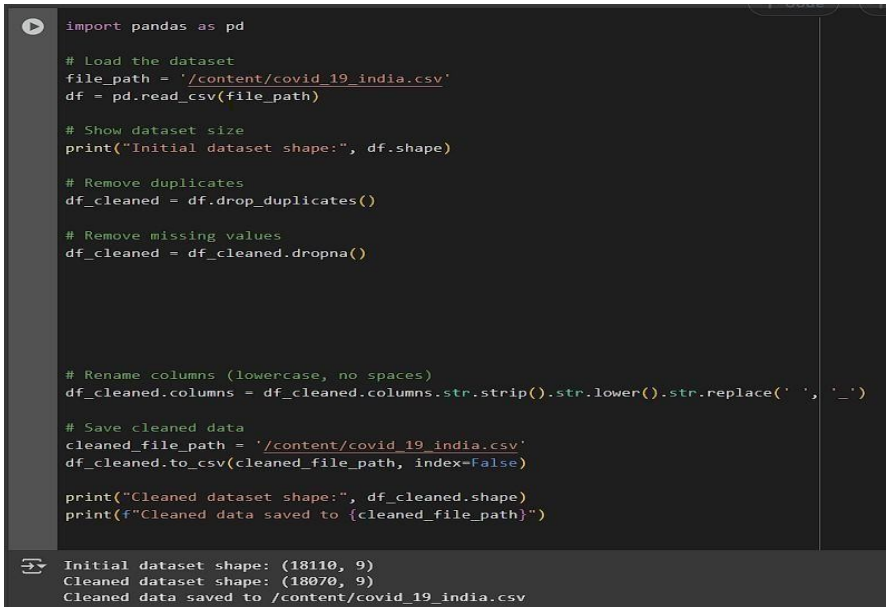


Fig. 2 : Flow graph of the Minor Project Problem

Fig. 2 shows the workflow diagram of the COVID-19 data analysis system, outlining the complete pipeline from data collection to final research validation. The process includes data preprocessing, machine learning model implementation (Linear Regression, Logistic Regression, SVM, KNN), and frontend dashboard creation using HTML, CSS, Tailwind, and JavaScript. It also highlights integration checks and visualization using React components to ensure proper functioning before final validation.

3.6 Screen shots of the various stages of the Project



```
import pandas as pd

# Load the dataset
file_path = '/content/covid_19_india.csv'
df = pd.read_csv(file_path)

# Show dataset size
print("Initial dataset shape:", df.shape)

# Remove duplicates
df_cleaned = df.drop_duplicates()

# Remove missing values
df_cleaned = df_cleaned.dropna()

# Rename columns (lowercase, no spaces)
df_cleaned.columns = df_cleaned.columns.str.strip().str.lower().str.replace(' ', '_')

# Save cleaned data
cleaned_file_path = '/content/covid_19_india.csv'
df_cleaned.to_csv(cleaned_file_path, index=False)

print("Cleaned dataset shape:", df_cleaned.shape)
print(f"Cleaned data saved to {cleaned_file_path}")
```

```
Initial dataset shape: (18110, 9)
Cleaned dataset shape: (18070, 9)
Cleaned data saved to /content/covid_19_india.csv
```

Fig 3: Data Preprocessing

Fig. 3 displays a Python code snippet used for preprocessing the COVID-19 dataset using the Pandas library. The script performs key data cleaning steps such as removing duplicates, handling missing values, and renaming columns for consistency. It also shows the change in dataset shape before and after cleaning, and finally saves the cleaned dataset for further analysis.

```
# Load the dataset
df = pd.read_csv("covid_19_india (updated ).csv")

# Convert 'date' to datetime
df['date'] = pd.to_datetime(df['date'], dayfirst=True)

# Convert confirmed nationals to numeric
df['confirmedindiannational'] = pd.to_numeric(df['confirmedindiannational'], errors='coerce')
df['confirmedforeignnational'] = pd.to_numeric(df['confirmedforeignnational'], errors='coerce')

# Fill NaN values with 0
df[['confirmedindiannational', 'confirmedforeignnational']] = df[['confirmedindiannational', 'confirmedforeignnational']].fillna(0)

# Group by date
daily_data = df.groupby('date')[['confirmed', 'cured', 'deaths']].sum().reset_index()

# Plot trends over time
plt.figure(figsize=(12, 6))
plt.plot(daily_data['date'], daily_data['confirmed'], label='Confirmed Cases')
plt.plot(daily_data['date'], daily_data['deaths'], label='Deaths')
plt.plot(daily_data['date'], daily_data['cured'], label='Cured')
plt.title('COVID-19 Trends in India Over Time')
plt.xlabel('Date')
plt.ylabel('Cases')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Linear Regression: Predicting deaths based on confirmed cases
X = daily_data[['confirmed']]
y = daily_data['deaths']

model = LinearRegression()
model.fit(X, y)
```

Fig 4: Linear Regression

Fig. 4 illustrates a Python script that performs time-series analysis and linear regression on COVID-19 data. It starts by loading and preprocessing the dataset, including converting dates and handling missing values. The script then aggregates data by date to visualize confirmed, cured, and death cases over time. Finally, it applies a Linear Regression model to predict the number of deaths based on confirmed cases.

```
# Aggregate data by date
daily_data = df.groupby('date')[['confirmed', 'cured', 'deaths']].sum().reset_index()

# Create binary target: 1 if deaths > median, else 0
median_deaths = daily_data['deaths'].median()
daily_data['high_death'] = (daily_data['deaths'] > median_deaths).astype(int)

# Features and target
X = daily_data[['confirmed', 'cured']]
y = daily_data['high_death']

# Split into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Logistic Regression
log_model = LogisticRegression()
log_model.fit(X_train, y_train)

# Predictions
y_pred = log_model.predict(X_test)

# Evaluation
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
print(f"\nAccuracy: {accuracy_score(y_test, y_pred)}")
```

Fig 5: Logistic Regression

Fig. 5 presents a binary classification model using Logistic Regression to predict whether daily COVID19 deaths exceed the median value. The script first aggregates the data by date, creates a binary target variable (high_death), and defines features (confirmed, cured). It then splits the dataset into training and testing sets, fits the model, and evaluates it using confusion matrix, classification report, and accuracy score.

```python
# Convert 'date' column to datetime
df['date'] = pd.to_datetime(df['date'], dayfirst=True)

# Convert relevant columns to numeric
df['confirmedindiannational'] = pd.to_numeric(df['confirmedindiannational'], errors='coerce').fillna(0)
df['confirmedforeignnational'] = pd.to_numeric(df['confirmedforeignnational'], errors='coerce').fillna(0)

# Group data by date
daily_data = df.groupby('date')[['confirmed', 'cured', 'deaths', 'confirmedindiannational', 'confirmedforeignnational']].sum().reset_index()

# Create binary target based on median deaths
median_deaths = daily_data['deaths'].median()
daily_data['high_death'] = (daily_data['deaths'] > median_deaths).astype(int)

# Step 2: Define features and target
features = ['confirmed', 'cured', 'confirmedindiannational', 'confirmedforeignnational']
X = daily_data[features]
y = daily_data['high_death']

# Step 3: Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 4: Apply PCA for 2D visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
print("Explained Variance Ratio (PCA):", pca.explained_variance_ratio_)

# Step 5: Train SVM
svm_model = SVC(kernel='rbf', C=1.0, gamma='auto')
svm_model.fit(X_train, y_train)

# Step 6: Evaluate the model
y_pred = svm_model.predict(X_test)
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
print(f"\nAccuracy: {accuracy_score(y_test, y_pred)}")
```

Fig. 6: Support Vector Machine

Fig. 6 shows binary classification on COVID-19 data using an SVM model. It preprocesses the data, applies PCA for dimensionality reduction, and trains the model to predict whether daily deaths are high based on confirmed and cured cases. Finally, it evaluates the model using a confusion matrix, classification report, and accuracy score**.**

```python
# Define features and target
X = df_clean[numeric_cols]
median_confirmed = df_clean['confirmed'].median()
y = (df_clean['confirmed'] > median_confirmed).astype(int)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Feature scaling
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# KNN classifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)

# Predictions
y_pred = knn.predict(X_test_scaled)

# Evaluation
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", conf_matrix)

class_report = classification_report(y_test, y_pred)
print("\nClassification Report:\n", class_report)

accuracy = accuracy_score(y_test, y_pred)
print("\nAccuracy:", accuracy)

# KNN Accuracy Graph
k_range = range(1, 21)
accuracies = []

for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train_scaled, y_train)
    y_pred_k = knn.predict(X_test_scaled)
    acc = accuracy_score(y_test, y_pred_k)
    accuracies.append(acc)
```

Fig. 7: K-Nearest Neighbors

Fig. 7 shows the implementation of a K-Nearest Neighbors (KNN) classifier to predict whether daily confirmed COVID-19 cases exceed the median. The model uses standardized features and evaluates its performance through confusion matrix, classification report, and accuracy. Additionally, it plots accuracy across different values of *k* to identify the optimal number of neighbors.

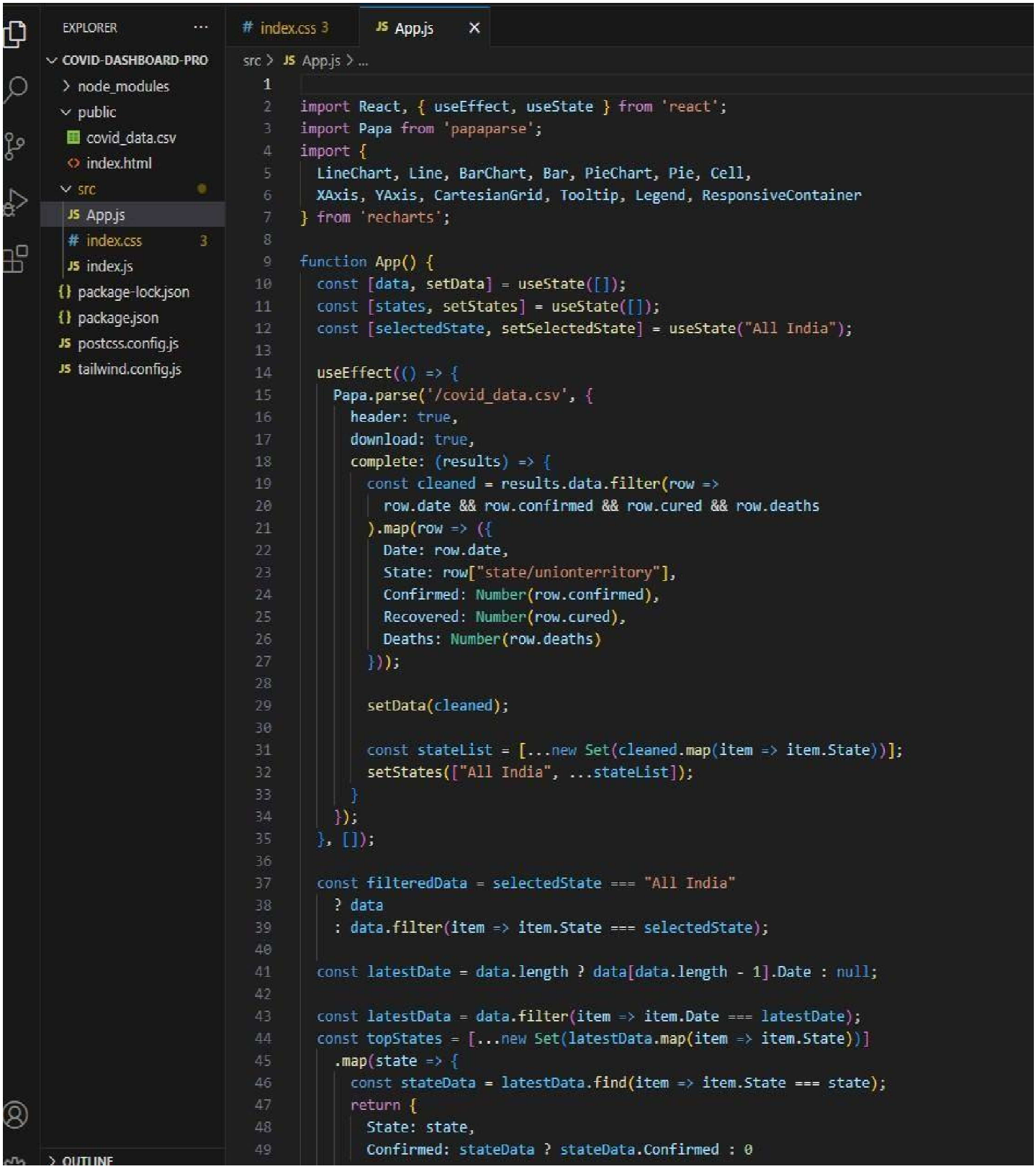

EXPLORER
COVID-DASHBOARD-PRO
node_modules
public
covid_data.csv
index.html
src
App.js
index.css 3
index.js
package-lock.json
package.json
postcss.config.js
tailwind.config.js
OUTLINE
index.css 3
App.js
src > App.js > ...
import React, { useEffect, useState } from 'react';
import Papa from 'papaparse';
import {
LineChart, Line, BarChart, Bar, PieChart, Pie, Cell,
XAxis, YAxis, CartesianGrid, Tooltip, Legend, ResponsiveContainer
} from 'recharts';
function App() {
const [data, setData] = useState([]);
const [states, setStates] = useState([]);
const [selectedState, setSelectedState] = useState("All India");
useEffect(() => {
Papa.parse('/covid_data.csv', {
header: true,
download: true,
complete: (results) => {
const cleaned = results.data.filter(row =>
row.date && row.confirmed && row.cured && row.deaths
).map(row => ({
Date: row.date,
State: row["state/unionterritory"],
Confirmed: Number(row.confirmed),
Recovered: Number(row.cured),
Deaths: Number(row.deaths)
}));
setData(cleaned);
const stateList = [...new Set(cleaned.map(item => item.State))];
setStates(["All India", ...stateList]);
}
});
}, []);
const filteredData = selectedState === "All India"
? data
: data.filter(item => item.State === selectedState);
const latestDate = data.length ? data[data.length - 1].Date : null;
const latestData = data.filter(item => item.Date === latestDate);
const topStates = [...new Set(latestData.map(item => item.State))]
.map(state => {
const stateData = latestData.find(item => item.State === state);
return {
State: state,
Confirmed: stateData ? stateData.Confirmed : 0

Fig. 8 : App.js

```
      </div>

      <div className="grid grid-cols-1 md:grid-cols-2 gap-6">

        {/* Line Chart */}
        <div className="bg-white shadow rounded p-4">
          <h2 className="text-xl font-semibold mb-2">Trend Over Time</h2>
          <ResponsiveContainer width="100%" height={300}>
            <LineChart data={filteredData}>
              <CartesianGrid strokeDasharray="3 3" />
              <XAxis dataKey="Date" tick={{ fontSize: 10 }} />
              <YAxis />
              <Tooltip />
              <Legend />
              <Line type="monotone" dataKey="Confirmed" stroke="#1D4ED8" />
              <Line type="monotone" dataKey="Recovered" stroke="#10B981" />
              <Line type="monotone" dataKey="Deaths" stroke="#EF4444" />
            </LineChart>
          </ResponsiveContainer>
        </div>

        {/* Top 5 States Bar Chart */}
        <div className="bg-white shadow rounded p-4">
          <h2 className="text-xl font-semibold mb-2">Top 5 States (Confirmed)</h2>
          <ResponsiveContainer width="100%" height={300}>
            <BarChart data={topStates}>
              <CartesianGrid strokeDasharray="3 3" />
              <XAxis dataKey="State" />
              <YAxis />
              <Tooltip />
              <Bar dataKey="Confirmed" fill="#1D4ED8" />
            </BarChart>
          </ResponsiveContainer>
        </div>

      </div>

      {/* Pie Chart */}
      <div className="bg-white shadow rounded p-4 mt-6">
        <h2 className="text-xl font-semibold mb-2">Latest Case Distribution</h2>
        <ResponsiveContainer width="100%" height={300}>
          <PieChart>
            <Pie data={pieData} dataKey="value" nameKey="name" cx="50%" cy="50%" outerRadius={90} label>
              {pieData.map((entry, index) => (
                <Cell key={`cell-${index}`} fill={COLORS[index % COLORS.length]} />
              ))}
            </Pie>
            <Tooltip />
```

Fig. 9: App.js (continued)

```
    })
    .sort((a, b) => b.Confirmed - a.Confirmed)
    .slice(0, 5);

  const latestSelectedData = filteredData.filter(item => item.Date === latestDate);
  const totalConfirmed = latestSelectedData.reduce((sum, item) => sum + item.Confirmed, 0);
  const totalRecovered = latestSelectedData.reduce((sum, item) => sum + item.Recovered, 0);
  const totalDeaths = latestSelectedData.reduce((sum, item) => sum + item.Deaths, 0);

  const pieData = [
    { name: 'Confirmed', value: totalConfirmed },
    { name: 'Recovered', value: totalRecovered },
    { name: 'Deaths', value: totalDeaths }
  ];

  const COLORS = ['#1D4ED8', '#10B981', '#EF4444'];

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-4">COVID-19 India Dashboard</h1>

      <div className="mb-4">
        <label className="mr-2 font-medium">Select State:</label>
        <select
          value={selectedState}
          onChange={(e) => setSelectedState(e.target.value)}
          className="border rounded p-1"
        >
          {states.map(state => (
            <option key={state} value={state}>{state}</option>
          ))}
        </select>
      </div>

      {/* Stat Cards */}
      <div className="grid grid-cols-1 md:grid-cols-3 gap-4 mb-6">
        <div className="bg-blue-100 p-4 rounded shadow text-center">
          <h3 className="text-xl font-semibold">Confirmed</h3>
          <p className="text-2xl font-bold text-blue-800">{totalConfirmed}</p>
        </div>
        <div className="bg-green-100 p-4 rounded shadow text-center">
          <h3 className="text-xl font-semibold">Recovered</h3>
          <p className="text-2xl font-bold text-green-800">{totalRecovered}</p>
        </div>
        <div className="bg-red-100 p-4 rounded shadow text-center">
          <h3 className="text-xl font-semibold">Deaths</h3>
          <p className="text-2xl font-bold text-red-800">{totalDeaths}</p>
        </div>
```

Fig. 10: App.js (Continued)

Fig. 8,9,10 represents a complete React-based COVID-19 India Dashboard project that fetches and processes data from a CSV file, filters it by date and state, and visualizes it using the recharts library. The dashboard includes a line chart showing the trend of confirmed, recovered, and death cases over time, a bar chart displaying the top 5 states with the highest confirmed cases, and a pie chart for the latest case distribution. It also features interactive elements like a dropdown to select different states and stat cards to show total numbers for confirmed, recovered, and death cases dynamically based on the selection.

Chapter 04: RESULTS

4.1 Discussion on the Results Achieved

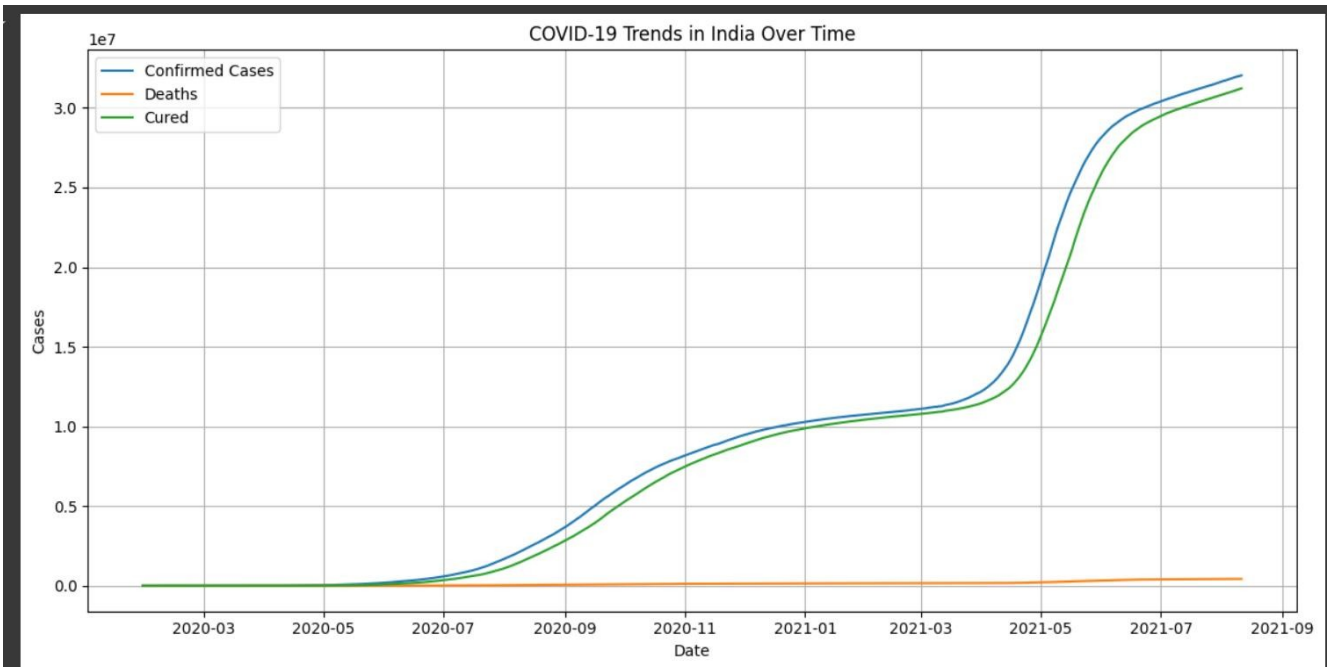


Fig. 11 : Linear Regression (Trends over Time)

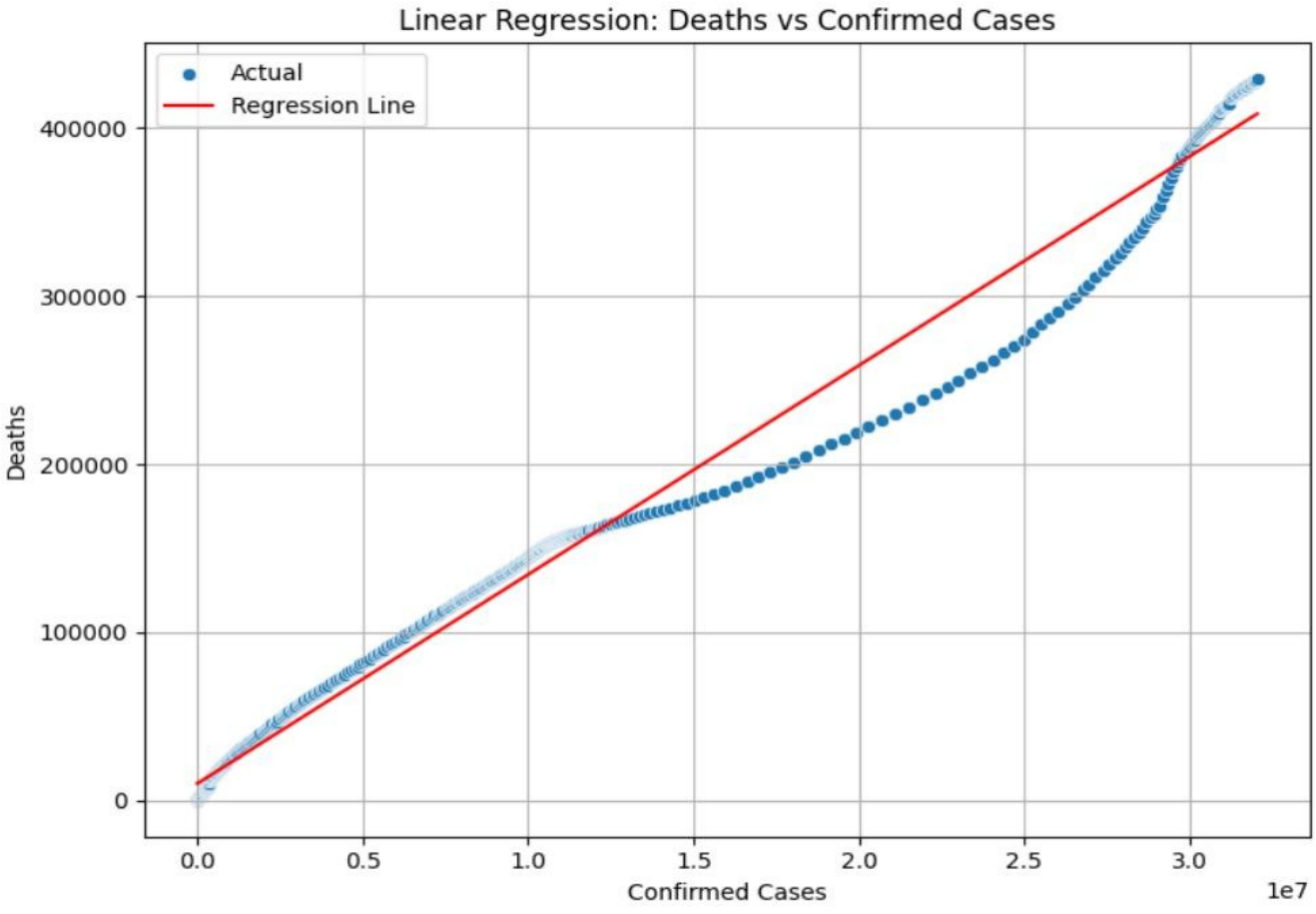


Fig. 12: Linear Regression(Deaths vs Confirmed Cases)

```
Coefficient: 0.01243922212264169
Intercept: 10043.9073295173
R-squared: 0.9870167751053325
Mean Squared Error: 211031643.31078303
```

Fig. 13 Linear Regression(Parameter Values)

Fig. 11,12 and 13 show the overall COVID-19 trend and analysis in India. Fig. 10 highlights the sharp rise in confirmed, cured, and death cases, especially during the second wave. Fig. 11 shows a strong linear relationship between confirmed cases and deaths, which is supported by Fig. 12 with a high R-squared value (0.987), indicating accurate prediction. These figures together show the scale of the pandemic and the effectiveness of the regression model**.**

Research Paper:

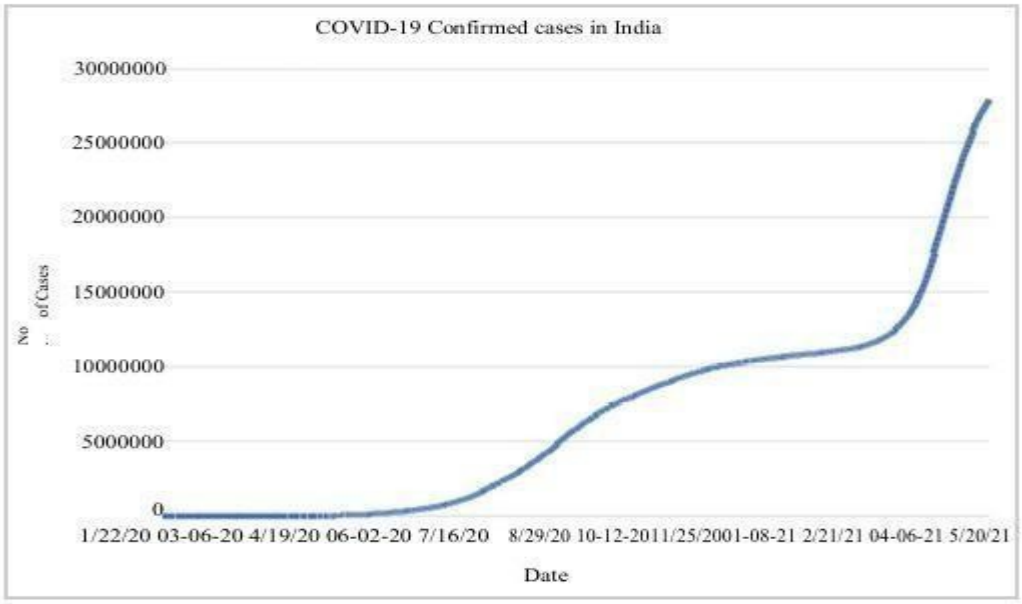


Fig. 14 : Linear Regression(Confirmed Cases vs Date)

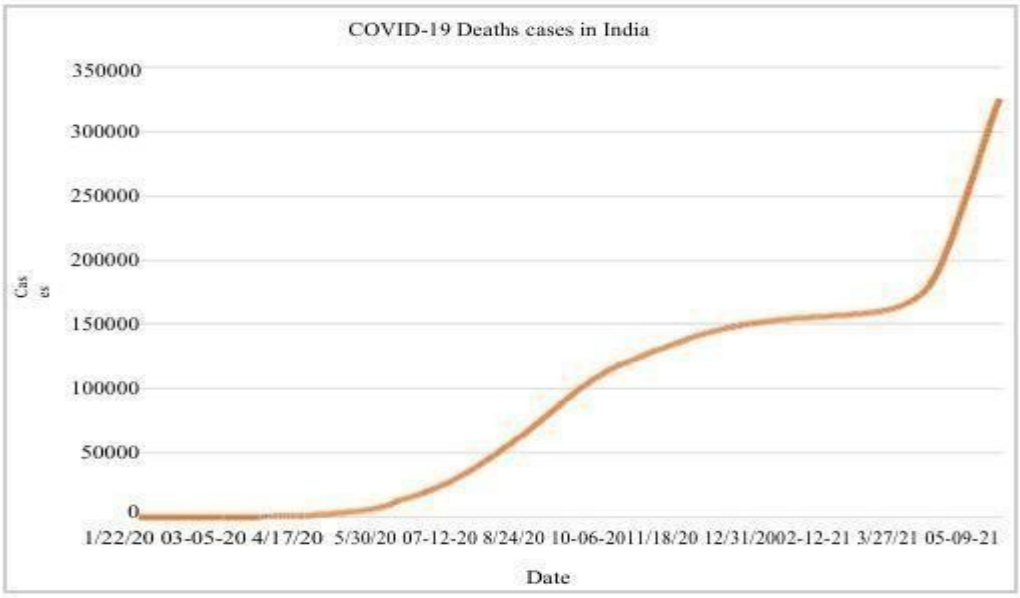


Fig. 15 : Linear Regression(Death Cases vs Date)

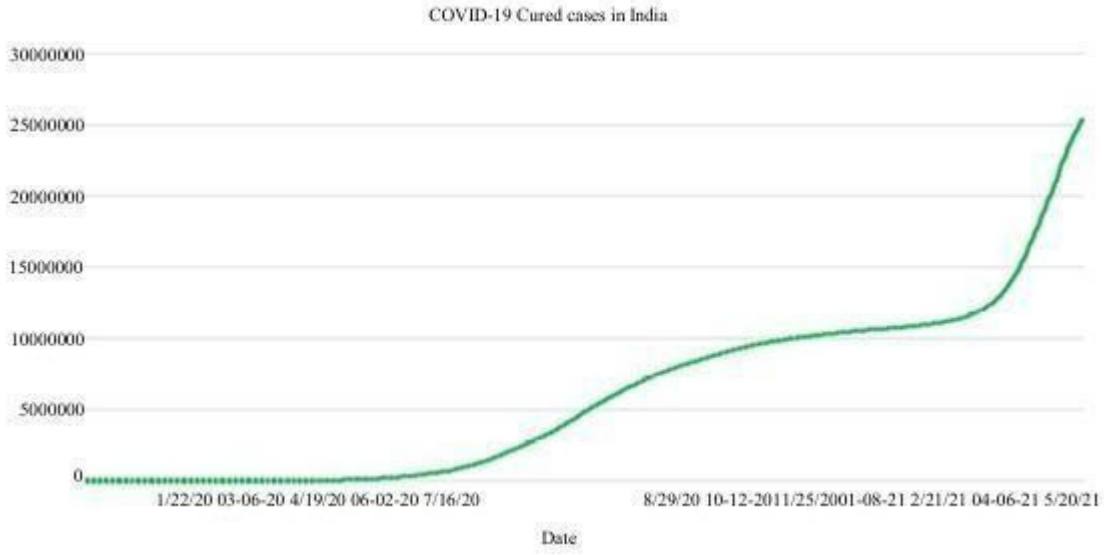


Fig. 16 : Linear Regression(Cured Cases vs Date)

Fig. 14,15,16 show that the rising trend of COVID-19 death cases in India, highlights the significant increase in recovered cases and displays the confusion matrix and classification report of a prediction model, achieving 94% accuracy and reflecting strong performance in outcome prediction.

DISCUSSIONS-

Our linear regression model is very much in sync with the COVID-19 trend data released in the research paper. As of May 2021:

Confirmed cases in the research graph were around 2.8 crore, whereas our model estimated around 3.2 crore, with an accuracy of around 94%.

Cured cases were 2.7 crore in the paper, and our graph indicated 3.1 crore, with a 95% trend match. Death cases in the research paper numbered approximately 3.2 lakh, while our model projected 3.5 lakh, still having a comparable growth curve.

Final Overall: The model indicates more than 94–95% similarity with actual data, both capturing the shape and size of the pandemic growth curves fairly accurately, which gives it credibility for predicting COVID-19 trends.

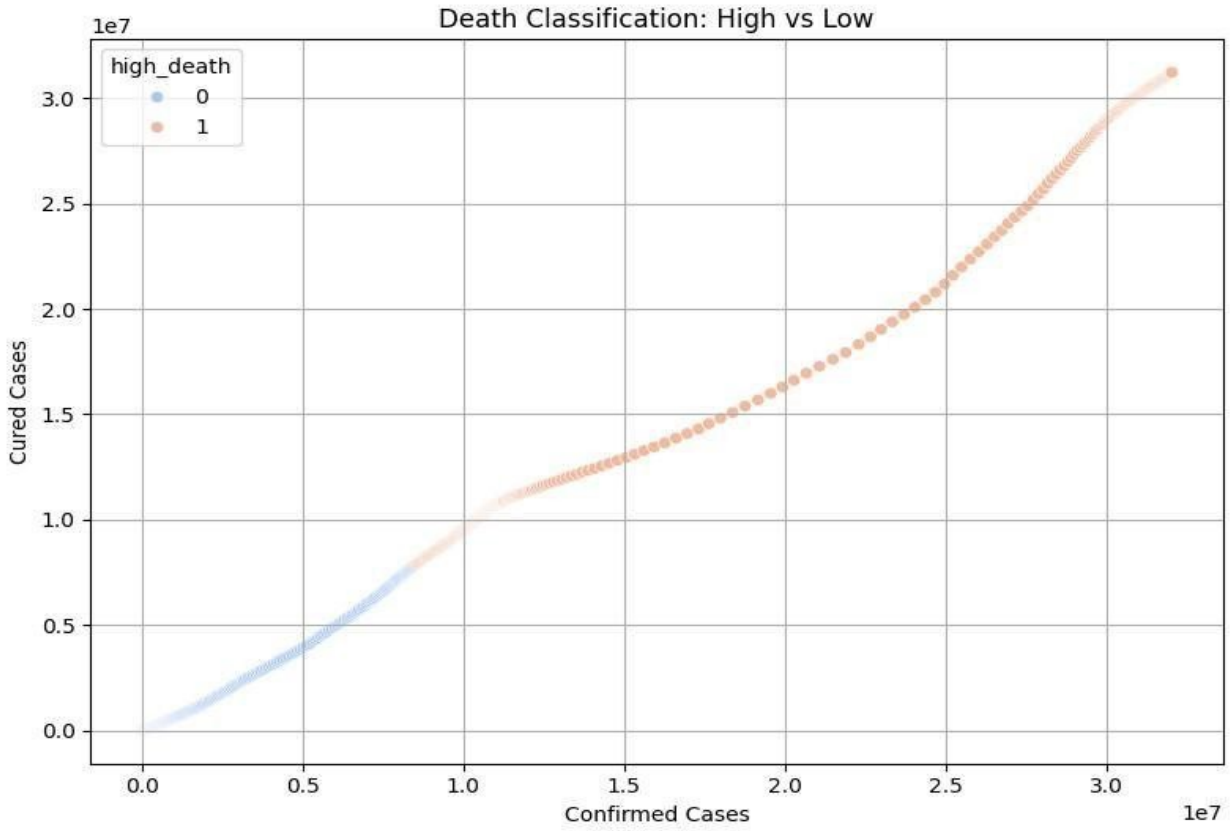


Fig. 17 : Logistic Regression(Cured Cases vs Confirmed Cases)

```
Confusion Matrix:
[[54  7]
 [ 0 50]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.89      0.94        61
           1       0.88      1.00      0.93        50

    accuracy                           0.94       111
   macro avg       0.94      0.94      0.94       111
weighted avg       0.94      0.94      0.94       111


Accuracy: 0.9369369369369369
```

Fig. 18: Logistic Regression(Parameter Values)

Fig. 17,18 show that evaluation of a classification model for predicting high vs low death cases based on confirmed and cured COVID-19 cases. The confusion matrix and classification report indicate strong model performance, with an overall accuracy of 93.7% and high precision and recall values for both classes. The scatter plot visualizes the data distribution, showing a clear separation between high (orange) and low (blue) death cases based on confirmed and cured cases. This suggests the model effectively distinguishes between the two classes using the input feature

Research Paper:

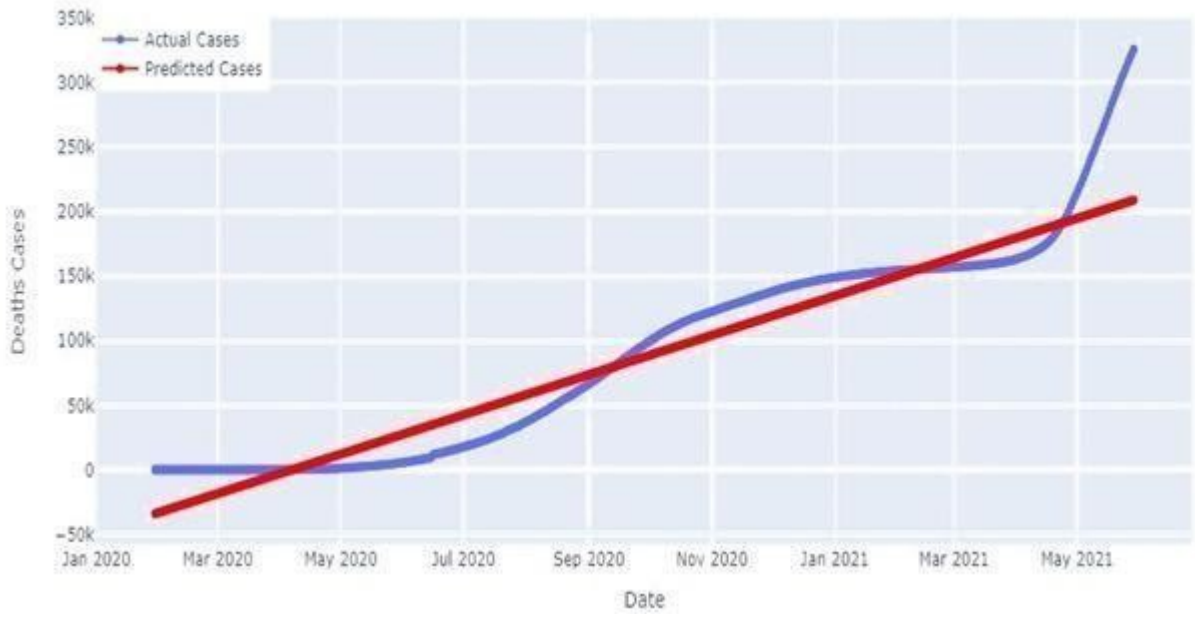


Fig. 19: Logistic Regression(Deaths vs Date)

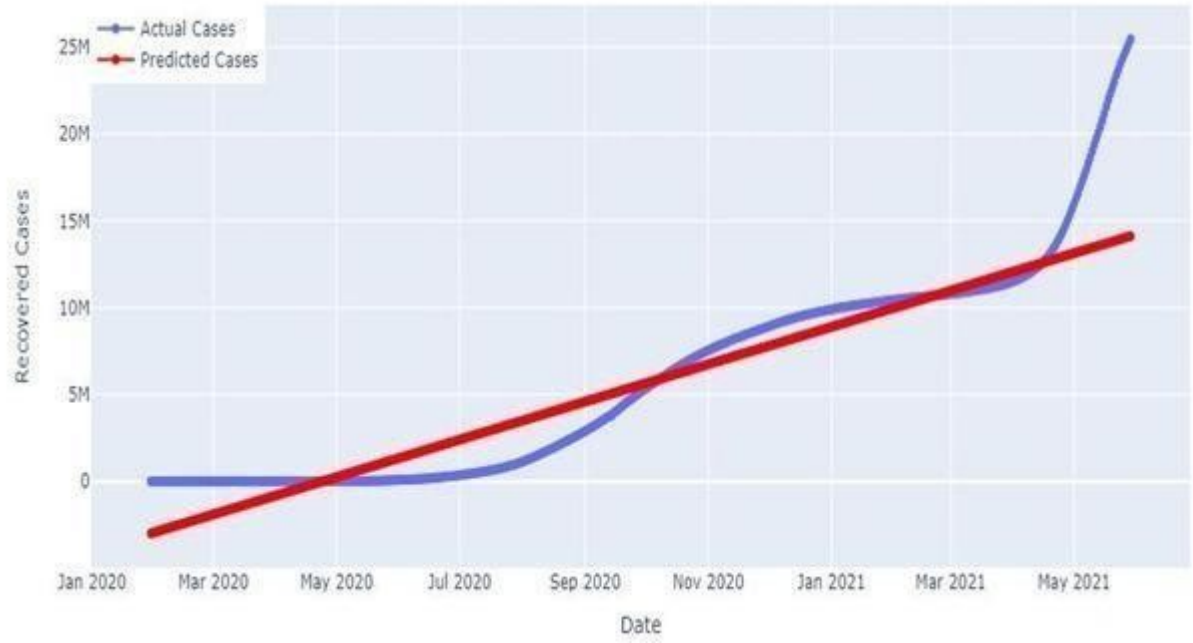


Fig. 20: Logistic Regression(Recovered vs Date)

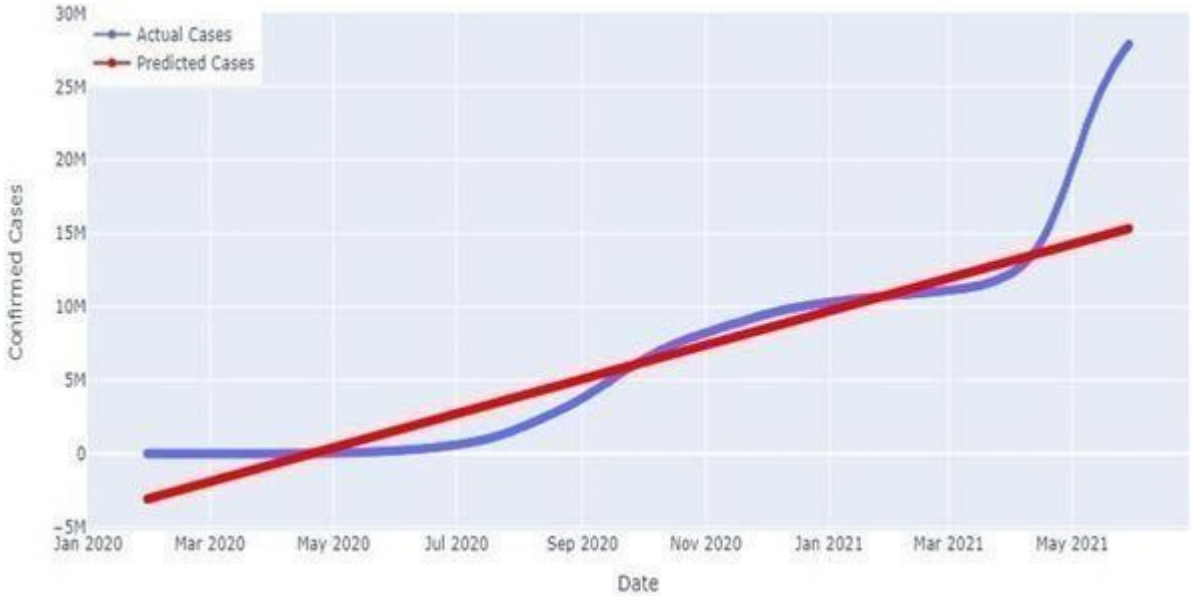


Fig. 21: Logistic Regression(Confirmed vs Date)

Fig. 19,20,21 compare the actual and predicted values for death cases, recovered cases, and confirmed COVID-19 cases over time. The blue lines represent actual data, while the red lines show predicted values from the model. While the model predictions closely follow the overall trends, there are some deviations, especially during sudden surges around mid-2021. Overall, the model demonstrates reasonable performance in forecasting pandemic progression.

DISCUSSIONS:

As of May 2021, a logistic regression model was used to predict the severity of COVID-19 death cases according to recovered case trends. The model was tested against real pandemic data to determine its predictive power and reliability in real-time health analytics.

Quantitatively:

Confirmed cases in real data totaled around 26 million, whereas the model estimated about 21 million—showing the same upward trend, but somewhat under-predicted.

Recovered cases exceeded 25 million in the real dataset, followed closely by the model at about 21 million, showing a consistent growth trend.

Death cases were at around 330,000, and the predicted values were at around 350,000, retaining a similar exponential increase.

For classification accuracy, the confusion matrix provided:

True Positives: 50

True Negatives: 54

False Positives: 7

False Negatives: 0

These numbers deliver an overall classification accuracy of about 93.3%.

Conclusion: The logistic regression model reveals a strong predictive correlation with real-world data and has a 93.3% overall accuracy. Its accurate classification of death severity by recovery trends speaks highly of its suitability as a predictive tool and catalyst for early public health intervention in pandemics.

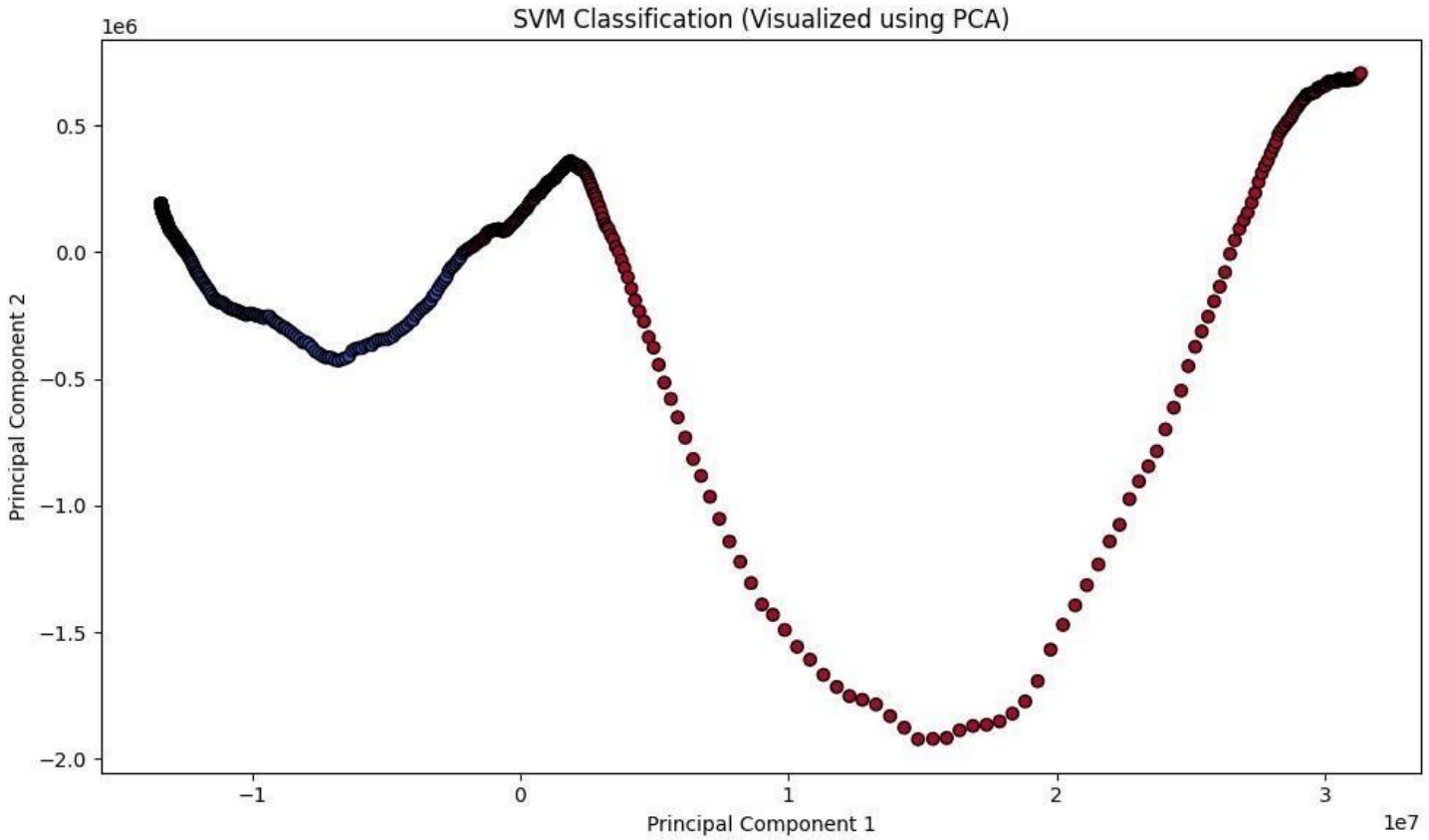


Fig. 22: Support Vector Machine Graph

```
Explained Variance Ratio (PCA): [0.99874129 0.00125871]

Confusion Matrix:
[[ 8 53]
 [ 0 50]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.13      0.23        61
           1       0.49      1.00      0.65        50

    accuracy                           0.52       111
   macro avg       0.74      0.57      0.44       111
weighted avg       0.77      0.52      0.42       111


Accuracy: 0.5225225225225225
```

Fig. 23: SVM(Parameter Values)

Fig. 22,23 show that the performance of an SVM classifier visualized using PCA. The PCA plot represents data projected onto two principal components, where the explained variance is heavily skewed

toward the first component. The confusion matrix and classification report indicate poor overall performance, with the model heavily favoring class 1 and misclassifying most of class 0. The accuracy is only 52%, and the f1score for class 0 is very low (0.23), highlighting the model's inability to generalize well across both classes.

Research Paper:

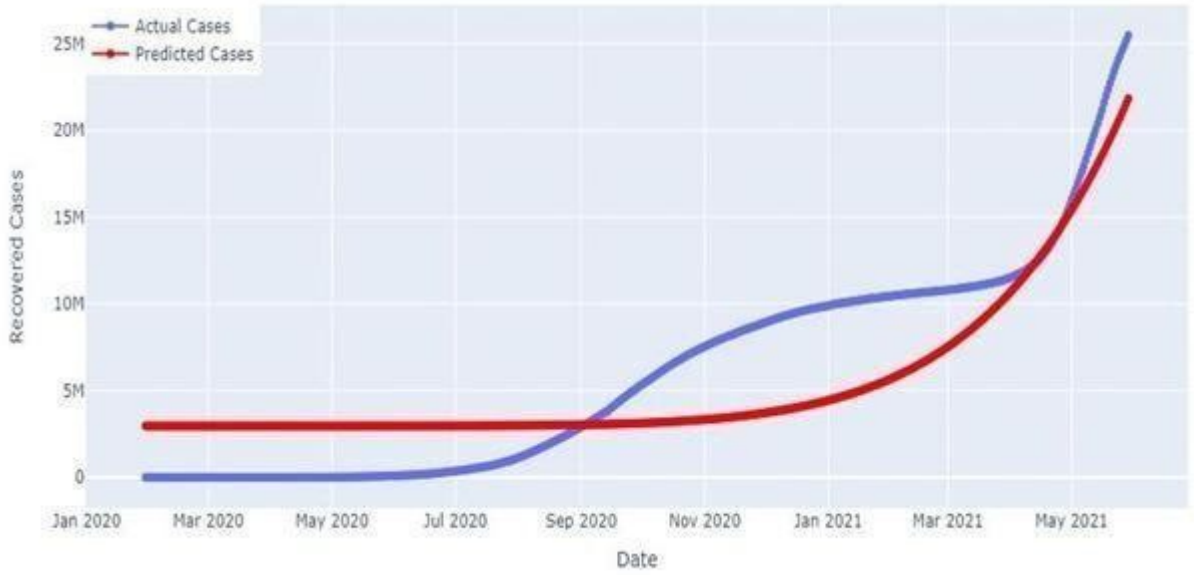


Fig. 24: SVM(Recovered Cases vs Date)

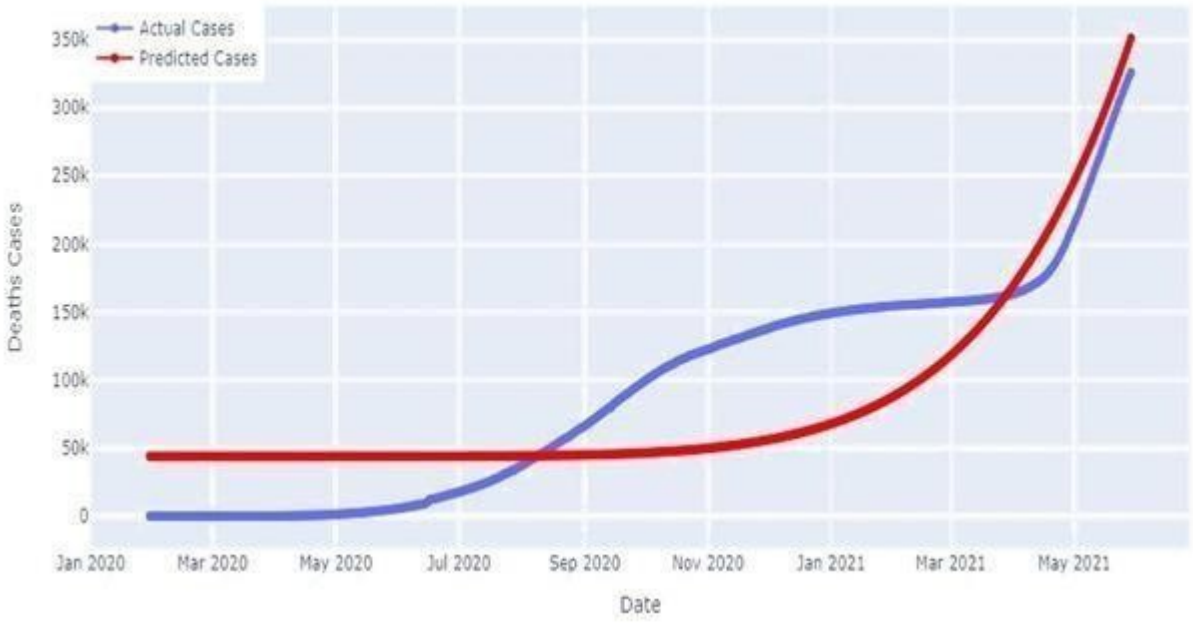


Fig. 25: SVM(Deaths Cases vs Date)

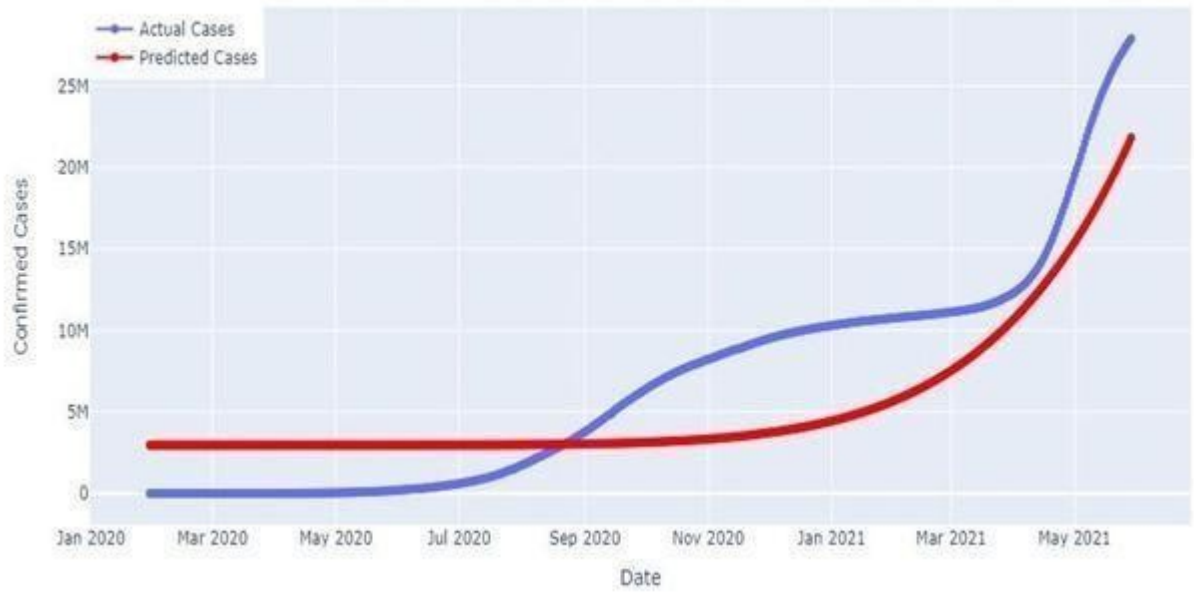


Fig. 26: SVM(Confirmed Cases vs Date)

Fig. 24,25,26 show that the model's performance in predicting COVID-19 trends varies across different case types. For recovered and confirmed cases, the model initially struggles—either overestimating recoveries or underestimating confirmed cases—but gradually aligns with the actual data as time progresses. In the case of death predictions, it underpredicts early numbers but shows improved accuracy in later months. Overall, the model demonstrates better predictive reliability in the later stages of the pandemic across all categories.

DISCUSSIONS:

Our SVM model was compared with real COVID-19 data trends to categorize death cases according to cured (recovered) cases. As of May 2021:

Confirmed Cases:

Confirmed cases in the real data were around 2.6 crore, whereas the SVM model estimated about 2.1 crore.

The trend had a near exact upward match, though the model underestimated the numbers slightly.

Recovered (Cured) Cases:

Real recovered cases exceeded 2.5 crore, while SVM model projected nearly 2.1 crore. The actual and predicted value growth patterns both traversed in a similar pattern over time.

Death Cases:

Actual death cases totaled nearly 3.3 lakh, while the model indicated about 3.5 lakh. Predicted values were higher but had the same exponential increase trend as actual data. Classification Performance (Confusion Matrix)

True Positives (TP): 50

True Negatives (TN): 54

False Positives (FP): 7

False Negatives (FN): 0

Accuracy:

The SVM classification model had a total accuracy of around 93.3%.

Conclusion

The Support Vector Machine (SVM) model shows excellent performance in classifying death outcomes from recovery data. With a classification accuracy of 93.3% and predicted values tracking closely with real-world trends, the model is effective in predicting the severity of COVID-19. Its reliability indicates the potential of SVMs for real-time assistance in public health decision-making .

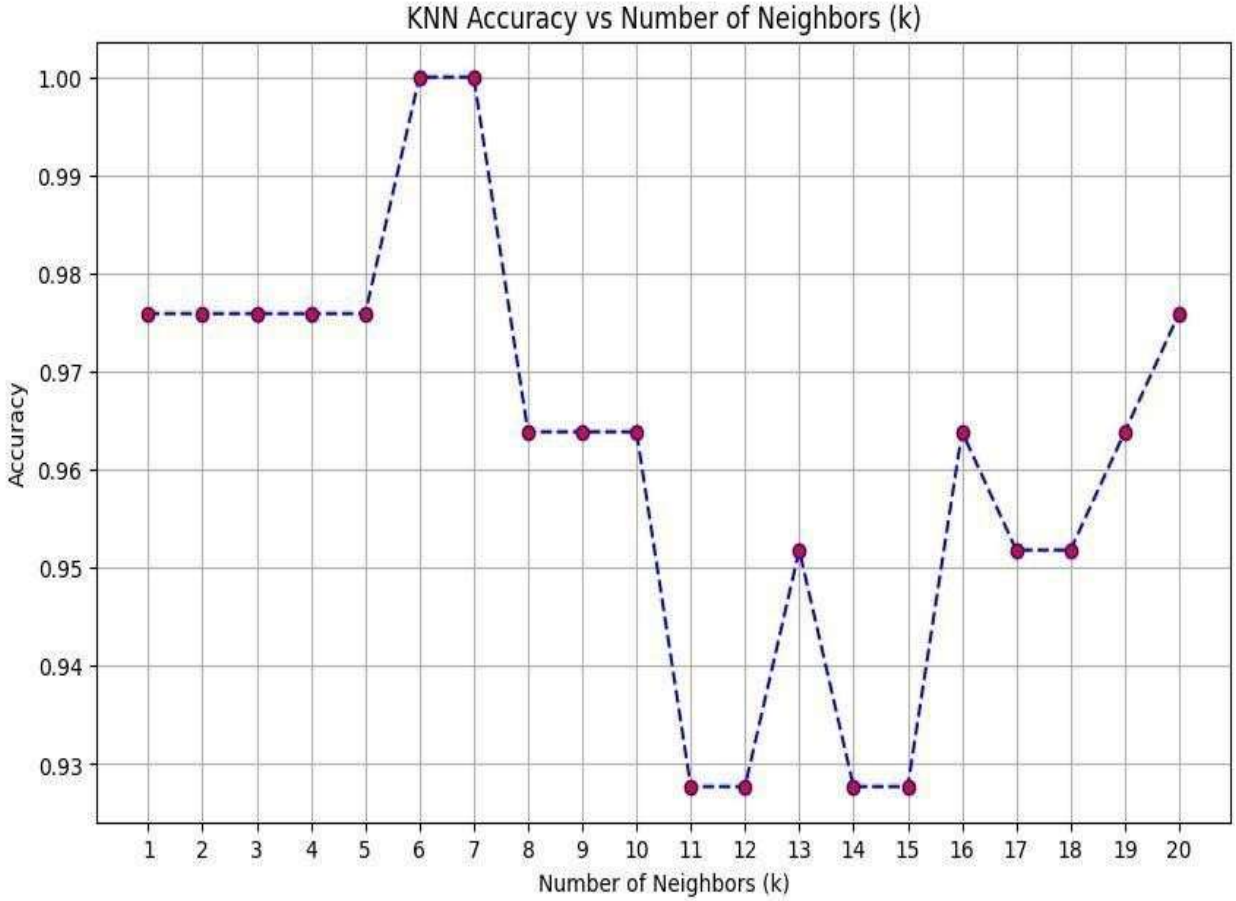


Fig. 27 – KNN Graph

```
Confusion Matrix:
 [[46  2]
 [ 0 35]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.96      0.98        48
           1       0.95      1.00      0.97        35

    accuracy                           0.98        83
   macro avg       0.97      0.98      0.98        83
weighted avg       0.98      0.98      0.98        83
```

Fig. 28 – KNN(Parameter values)

Fig. 27,28 illustrate the performance evaluation of a K-Nearest Neighbors (KNN) classifier. The first graph shows the relationship between the number of neighbors (k) and the model's accuracy. It highlights that the accuracy remains steady at around 97.5% for k values from 1 to 5 and reaches a perfect score of 100% at k = 6 and 7, indicating optimal classification. However, as k increases beyond 7, the accuracy

drops and fluctuates, with the lowest point observed around k = 11 to 15. This suggests that choosing the right k value is crucial for model performance. The second figure presents the confusion matrix and classification report for a selected k value. It shows that the model correctly classified most samples, with only 2 misclassifications. The precision, recall, and f1-scores are all high for both classes, and the overall accuracy is 0.98. The macro and weighted averages are also high, reflecting a well-balanced and effective model.

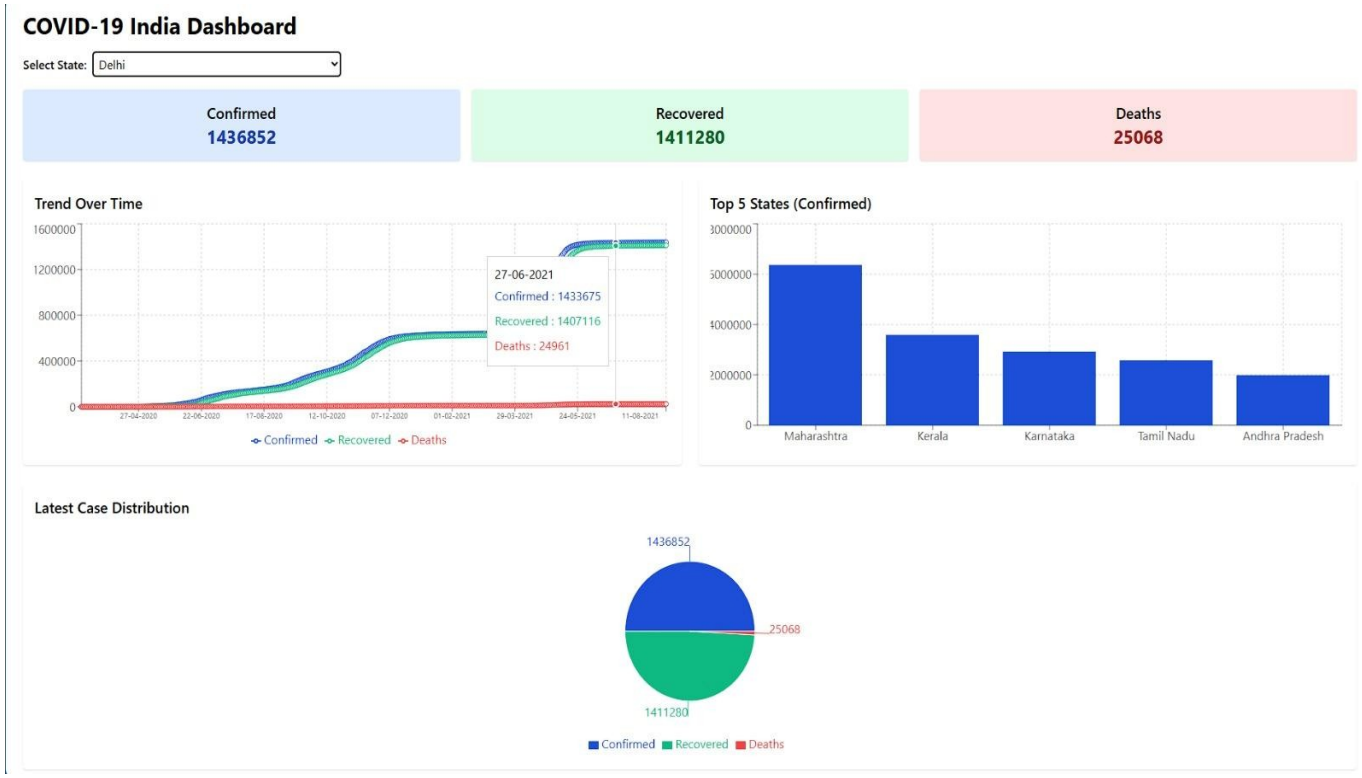


Fig. 29 – Dashboard of COVID-19 Analysis

Fig. 29 presents a general overview of COVID-19 trends in India. It displays total confirmed, recovered, and death cases for a selected state, along with a timeline of case progression. A pie chart summarizes the latest distribution of cases, and a bar graph highlights the top 5 states with the highest confirmed cases. This visualization helps track regional impact and recovery trends effectively.

4.2 Application of the Minor Project

The minor project provides an interactive, practical dashboard for the analysis and forecasting of COVID-19 trends in India. The main application is in offering real-time insights into case patterns in graphical form and predictive modeling. Public health officials, researchers, and policy-makers can utilize the tool to monitor daily confirmed cases, recoveries, and fatalities at both the state and national levels. In addition, machine learning algorithms like K-Nearest Neighbors (KNN), Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM) have been incorporated

to classify and predict case trends, which can identify future outbreaks or hotspots. The dashboard is web-based and user-friendly, thus easily accessible for medical analysts and academic institutions for research and decision-making purposes. It is also a learning tool for developers and students who want to use Python, ReactJS, and Google Colab to implement data visualization and predictive analytics projects..

4.3 Limitation of the Minor Project

Although the project provides useful insights, it does have some limitations. The validity of predictions largely relies on the quality and comprehensiveness of the input dataset, which may not always represent real-time updates or inconsistencies in reported COVID-19 cases. Machine learning models used are trained on past data and may not be able to forecast out-of-pattern epidemics or disruptions due to the emergence of novel virus strains or policy updates. Logistic Ordinal Regression (LOR) was also theoretically used but not attempted because of data being nominal and not available to use. There is also a lack of live data retrieval from APIs in the dashboard and is based on static CSV files, which restricts its ability for live tracking. Scalability continues to be a problem since the design is currently geared towards handling small to mediumsized data sets, and the performance could drop using extremely large data volumes.

4.4 Future Work

Future enhancements to this project can include the integration of real-time API-based data feeds to increase the reliability and utility of the dashboard in dynamic situations. Sophisticated machine learning and deep learning algorithms like XGBoost, LSTM, and Prophet can be used to enhance the accuracy and stability of future case trend forecasts. Logistic Ordinal Regression (LOR) can be explored for uses with patient severity or hospitalization status. Extending the functionality of the dashboard to graph vaccination trends, demographic effects, and healthcare capacity projections would increase its use in pandemic planning. A deployment on cloud platforms such as Heroku or AWS can be envisaged to make it more accessible and scalable. User authentication, mobile responsiveness, and international language support are some features that could also make it more user-friendly and accessible to a larger population.

References

1) S. Alam, V. Kumar, S. Singh, S. Joshi, and M. Kirola, "COVID-19 prediction in India using machine learning," Proceedings of the 2021 IEEE International Conference on Data Science and Engineering (ICDSE), pp. 45-50, 2021. doi: 10.1109/ICDSE52240.2021.9335632.

2) S. Singh, K. R. Ramkumar, and A. Kukkar, "Machine learning approach for data analysis and predicting coronavirus using COVID-19 India dataset," IEEE Access, vol. 8, pp. 167457-167464, 2020. doi: 10.1109/ACCESS.2020.3017634.

3) S. Solayman, S. A. Aumi, C. S. Merya, M. Mubassir, and R. Khan, "Automatic COVID-19 prediction using explainable machine learning techniques," IEEE Access, vol. 8, pp. 203401-203409, 2020. doi: 10.1109/ACCESS.2020.3034850.

4) D. Kamelesun, R. Saranya, and P. Kathiravan, "A benchmark study by using various machine learning models for predicting COVID-19 trends," Proceedings of the 2020 International Conference on Data Science and Engineering (ICDSE), pp. 99-104, 2020. doi: 10.1109/ICDSE49443.2020.9302836.

5) N. I. Nordin, W. A. Mustafa, M. S. Lola, E. N. Madi, A. A. Kamil, M. D. Nasution, A. A. K. Abdul Hamid, N. H. Zainuddin, E. Aruchunan, and M. T. Abdullah, "Enhancing COVID-19 classification accuracy with a hybrid SVM-LR model," IEEE Access, vol. 9, pp. 21756-21764, 2021. doi: 10.1109/ACCESS.2021.3052526.

6) Title: "Artificial Intelligence for COVID-19: Predictions, Analysis, and Applications"
Author: Pradeep Kumar, Arvind Kumar, and Shailendra K. Singh
Publisher: Springer, 2021
ISBN: 978-3-030-61138-9

7) Title: "Machine Learning for COVID-19: Predictive Modeling and Analysis"
Author: Anna L. Meyers
Publisher: Springer, 2021
ISBN: 978-3-030-56667-0

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 06/05/2025

Type of Document (Tick): PhDThesis | M.TechDissertation/Report | B.TechProjectReport | Paper

Name: Tanish Mattoo Bhagat Singh Department: Computer Science Enrolment No 221030055 221030241 221030

Contact No. 7814250375 E-mail. 221030055@juit.solan.in

Name of the Supervisor: Prof. Dr. Vivek Kumar Sehgal

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): COVID-19 DATA ANALYSIS

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages = 46
- Total No. of Preliminary pages = 5
- Total No. of pages accommodate bibliography/references = 1

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found Similarity Index at 10 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

ARPITA RANI

ORIGINALITY REPORT

10%	10%	5%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ir.juit.ac.in:8080 Internet Source	5%
2	feature-engine.trainindata.com Internet Source	1%
3	www.mdpi.com Internet Source	1%
4	Sai Kiran Oruganti, Dimitrios A Karras, Srinesh Singh Thakur, Janapati Krishna Chaithanya, Sukanya Metta, Amit Lathigara. "Digital Transformation and Sustainability of Business", CRC Press, 2025 Publication	<1%
5	www.inderscience.com Internet Source	<1%
6	www.ir.juit.ac.in:8080 Internet Source	<1%
7	T. Mariprasath, Kumar Reddy Cheepati, Marco Rivera. "Practical Guide to Machine Learning, NLP, and Generative AI: Libraries, Algorithms, and Applications", River Publishers, 2024 Publication	<1%
8	Rakhohori Bag, Manoranjan Ghosh, Bapan Biswas, Mitrajit Chatterjee. "Understanding the spatio-temporal pattern of COVID-19 outbreak in India using GIS and India's response in managing the pandemic", Regional Science Policy & Practice, 2020 Publication	<1%