Project Report

on

Covid-19 Analysis by State (2020-2021)

Submitted by

Tushar Negi

24MCC20035

Under the guidance of

Mr. Rishabh Tomar

in partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

CLOUD COMPUTING & DEVOPS

Chandigarh University

April 2025

Certificate

This is to certify that **Tushar Negi**, a student of **Master of Computer Applications (MCA) – Cloud Computing and DevOps**, has successfully completed the **Minor Project** titled **"Covid-19 Analysis by State (2020-2021)"** under the esteemed guidance of **Mr. Rishabh Tomar, Assistant Professor, University Institute of Computing (UIC), Chandigarh University**.

This project was undertaken as a part of the academic curriculum and is submitted in **partial fulfilment of the requirements** for the MCA program. The work presented in this project is a result of **independent research, diligent effort, and dedication**, demonstrating the student's ability to apply theoretical knowledge to practical problem-solving.

The project successfully implements **Big Data processing using Hadoop and MapReduce**, demonstrating an efficient approach to analysing word frequency in large-scale textual data. It reflects the student's understanding of **Big Data frameworks, distributed computing, and data visualization techniques**.

I hereby confirm that this project is an **original work** carried out by the student and has **not been submitted elsewhere** for the award of any other degree, diploma, or certification.

Project Guide:

Mr. Rishabh Tomar

Assistant Professor

University Institute of Computing

Chandigarh University

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+
Accredited University

Acknowledgement

I would like to express my sincere gratitude to **Chandigarh University** and the **University Institute of Computing (UIC)** for providing me with the opportunity to undertake this project, **“Covid-19 Analysis by State (2020-2021)”**

I extend my heartfelt appreciation to my esteemed mentor, **Mr. Rishabh Tomar, Assistant Professor**, for his invaluable guidance, continuous support, and insightful feedback throughout the project. His expertise in **Big Data and Distributed Systems** played a crucial role in the successful completion of this project.

I am also grateful to my friends and peers for their encouragement and discussions, which helped refine my approach. Lastly, I thank my family for their unwavering support and motivation during this research.

This project has been an incredible learning experience, and I hope it serves as a foundation for further exploration in **Big Data analytics and Hadoop-based processing.**

Tushar Negi

MCA – Cloud Computing and DevOps

Chandigarh University

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+
Accredited University

Contents

Abstract

The COVID-19 pandemic has significantly impacted global health systems, economies, and daily life, making it essential to analyse case data efficiently. Given the enormous volume of COVID-19 data generated daily, traditional data processing techniques struggle to handle large-scale datasets effectively. To address this challenge, this project leverages Big Data technologies, specifically Hadoop and MapReduce, for COVID-19 data analysis. By utilizing Hadoop Distributed File System (HDFS) for data storage and MapReduce for parallel processing, this project provides an efficient way to analyse the spread of the virus across different states in India.

The dataset used in this project contains structured COVID-19 case details, including state-wise confirmed cases. The first step involves storing the dataset in HDFS, ensuring fault tolerance and scalability. A MapReduce job is then implemented to process the dataset, where the Mapper extracts relevant information such as state names and confirmed cases, and the Reducer aggregates the total number of cases for each state. The final output is stored in HDFS and later retrieved for analysis.

To enhance the interpretability of the results, Python and Matplotlib are used for data visualization. The processed data is extracted from HDFS, and a bar chart is generated to illustrate the state-wise distribution of COVID-19 cases. This visualization provides an intuitive understanding of how the pandemic has affected different regions, making it easier to identify trends and patterns.

This project highlights the efficiency of Hadoop's distributed computing model in processing large datasets and demonstrates how Big Data analytics can be applied to public health challenges. The insights obtained from this analysis can assist researchers, healthcare professionals, and policymakers in understanding the impact of COVID-19 at a granular level. Furthermore, this project emphasizes the importance of scalable data processing solutions in tackling real-world problems, proving that Hadoop and MapReduce can serve as powerful tools for large-scale data analytics in the healthcare sector.

1. Introduction

The COVID-19 pandemic has significantly impacted public health and economies worldwide, leading to an urgent need for data-driven analysis to understand its spread and trends. With the massive volume of COVID-19 case data being generated daily, traditional data processing techniques struggle to handle such large datasets efficiently. This project leverages **Hadoop and MapReduce**, two powerful Big Data technologies, to analyse COVID-19 cases across different states, enabling efficient data processing and trend visualization.

Hadoop's **distributed computing model** allows for parallel data processing across multiple nodes, making it ideal for handling vast amounts of COVID-19 records. The **MapReduce framework** is used to aggregate confirmed cases for each state, ensuring efficient and scalable analysis. By storing the dataset in **Hadoop Distributed File System (HDFS)**, the project takes advantage of fault tolerance and high availability, which are crucial for handling large-scale data.

This project follows a systematic approach, beginning with setting up the Hadoop environment, processing COVID-19 data using a **MapReduce job**, retrieving the analysed results, and visualizing them using **Python and Matplotlib**. The extracted insights help in understanding the **state-wise distribution of COVID-19 cases**, which can further assist researchers and policymakers in decision-making.

By implementing **Big Data analytics in healthcare**, this project demonstrates the potential of Hadoop and MapReduce in processing and analysing real-world datasets. It highlights the importance of scalable data analysis solutions in tackling global challenges, ensuring that vast amounts of data can be processed efficiently and transformed into meaningful insights.

2. Tools and Technologies Used

This project utilizes various tools and technologies to handle data collection, processing, and visualization effectively. The key components include:

2.1 Big Data Frameworks

- **Apache Hadoop**
  Apache Hadoop is an open-source framework designed for handling large-scale data processing. It provides:
  HDFS (Hadoop Distributed File System): A fault-tolerant and scalable storage solution.
  MapReduce: A distributed computing model used for processing large datasets efficiently.

2.2 Programming Languages

- **Java:** Java is used to implement the MapReduce program that processes AQI data.
  It includes:
  Mapper Function to extract relevant data.
  Reducer Function to compute and aggregate AQI values.
  Driver Class to control execution.
- **Python:** Python is used for data processing and visualization after executing the MapReduce job. It helps in:
  Manipulating the processed AQI data.
  Creating **graphical representations** using Matplotlib.

2.3 Data Sources

- The dataset used in this project consists of COVID-19 case details collected from publicly available sources. It includes the following attributes:
  - **Date**: The reporting date of COVID-19 cases.
  - **State/UT**: The state or union territory where the cases were recorded.
  - **Confirmed Cases**: The number of confirmed COVID-19 cases reported on a given date.
- The dataset is pre-processed and uploaded to HDFS before executing the MapReduce job for analysis.

3. Implementation Steps

The implementation of this project follows a structured approach, leveraging **Hadoop, MapReduce, and Python** to perform COVID-19 data analysis. The process involves setting up the environment, fetching and storing data, running a MapReduce job for analysis, retrieving the results, and visualizing the findings. Below is a detailed breakdown of each step:

3.1 Setting Up the Environment

To begin the project, the necessary software and tools were installed and configured, including:

- Installing **Apache Hadoop** and setting up the **Hadoop Distributed File System (HDFS)** for data storage.
- Configuring the environment to execute **MapReduce jobs** for distributed data processing.
- Setting up a **Linux-based execution environment** to manage Hadoop operations.

Commands to initialize Hadoop:

hadoop-3.3.6/bin/hdfs namenode -format
export PDSH_RCMD_TYPE=ssh
start-all.sh
hdfs dfs -mkdir /input
hdfs dfs -put /home/prerna/Downloads/covid_19_india.csv /input

3.2 Fetching Data

The dataset for this project, **COVID-19 India Dataset**, was sourced from publicly available repositories. The dataset contains:

- **State-wise daily records** of confirmed COVID-19 cases.
- **Time-series data** tracking the spread of the virus across different regions.
- **Metadata fields**, including state name, date, and total cases reported.

Before processing, **data preprocessing** was performed to ensure consistency, including:

- Handling missing values by replacing them with zero or interpolated values.
- Standardizing the data format for seamless analysis.
- Filtering only the relevant columns needed for the MapReduce job.

3.3 Storing Data in HDFS

Once pre-processed, the dataset was uploaded to **Hadoop Distributed File System (HDFS)**, which offers:

- **Scalability**: Capable of storing large datasets efficiently.
- **Fault tolerance**: Ensures data integrity even in the event of node failures.
- **Distributed architecture**: Optimized for parallel processing with MapReduce.

Command to upload data to HDFS:

```
hdfs dfs -put /home/prerna/Downloads/covid_19_india.csv /input
```

3.4 Implementing the MapReduce Job

To analyze COVID-19 trends, a **MapReduce program** was developed in **Java**. The implementation follows a three-phase approach:

1. Mapping Phase

- The **Mapper function** reads COVID-19 data and extracts relevant details such as **state name** and **confirmed cases**.
- Each data record is processed as a **key-value pair**, where the **state name** serves as the key, and the **confirmed cases** act as the associated value.

2. Shuffling & Sorting Phase

- The Hadoop framework automatically **groups case counts by state** and sorts them in preparation for aggregation.

3. Reducing Phase

- The **Reducer function** sums up the total **confirmed cases** for each state across all dates.
- The results are then **written back to HDFS** for further analysis and visualization.

Java code for MapReduce job (CovidAnalysis.java):

```java
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
```

```
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class CovidAnalysis {

    public static class CovidMapper extends Mapper<LongWritable, Text, Text,
IntWritable> {
        @Override
        public void map(LongWritable key, Text value, Context context)
                throws IOException, InterruptedException {
            String line = value.toString();
            String[] parts = line.split(",");
            if (parts.length > 8) {
                try {
                    String state = parts[3];
                    int confirmedCases = Integer.parseInt(parts[8]);
                    context.write(new Text(state), new IntWritable(confirmedCases));
                } catch (NumberFormatException | ArrayIndexOutOfBoundsException e) {
                    // Handle parsing errors
                }
            }
        }
    }

    public static class CovidReducer extends Reducer<Text, IntWritable, Text,
IntWritable> {
        @Override
        public void reduce(Text key, Iterable<IntWritable> values, Context context)
                throws IOException, InterruptedException {
            int sum = 0;
```

```
            for (IntWritable val : values) {
                sum += val.get();
            }
            context.write(key, new IntWritable(sum));
        }
    }

    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: CovidAnalysis <input path> <output path>");
            System.exit(-1);
        }

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Covid Data Analysis");
        job.setJarByClass(CovidAnalysis.class);
        job.setMapperClass(CovidMapper.class);
        job.setReducerClass(CovidReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

3.5 Retrieving Processed Results

After running the **MapReduce job**, the processed results were retrieved from **HDFS**. The output includes:

- **State-wise total confirmed cases**, allowing comparison of COVID-19 impact across different regions.
- **Summarized data**, making it easier to analyze infection trends.

Command to retrieve results:

```
hadoop fs -get /output/part-r-00000 covid_results.txt
```

3.6 Analysing and Visualizing Results

Once the processed data was retrieved, **Python and Matplotlib** were used to generate a **bar chart** for a clear visualization of COVID-19 cases across states. The key steps include:

- **Loading the processed COVID-19 data** into Python using the Pandas library.
- **Creating bar charts** using Matplotlib to display the total confirmed cases per state.
- **Enhancing visualization** with labelled axes and rotated state names for readability.

Python code for visualization (plot_covid.py):

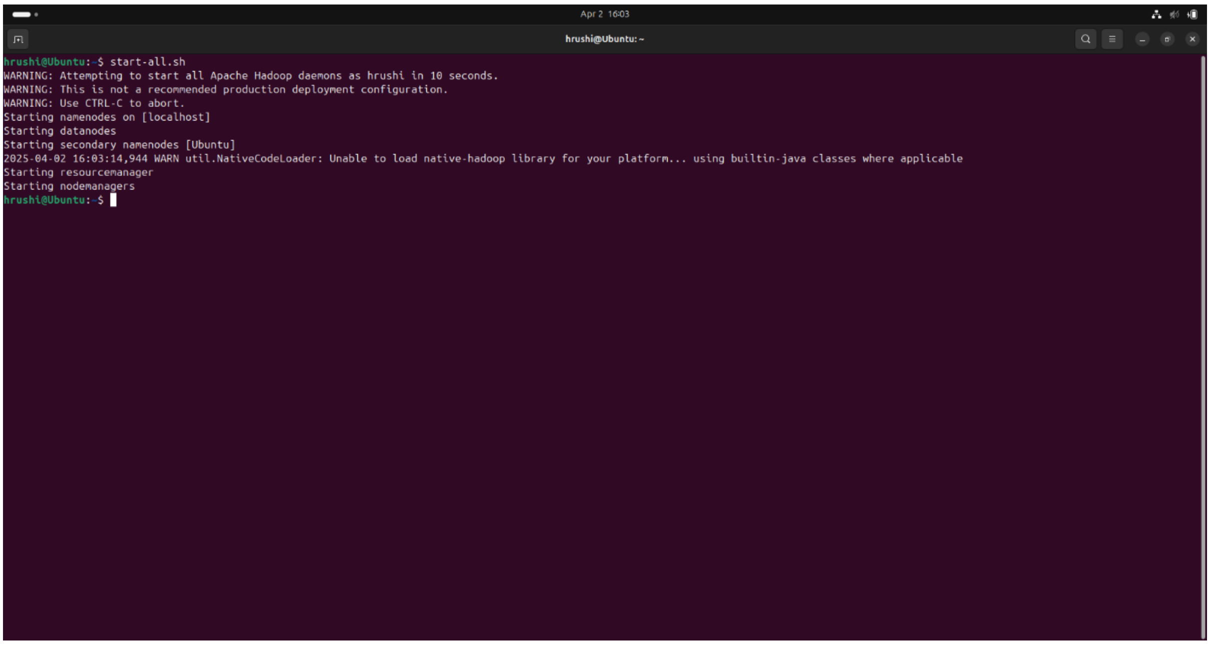
```
import matplotlib.pyplot as plt

states = []
cases = []

with open("covid_results.txt", "r") as f:
    for line in f:
        state, case = line.strip().split("\t")
        states.append(state)
        cases.append(int(case))

plt.figure(figsize=(15, 7))
plt.bar(states, cases, color="skyblue")
plt.xticks(rotation=90)
plt.xlabel("States")
plt.ylabel("Total Confirmed Cases")
plt.title("Total Confirmed COVID-19 Cases by State")
plt.tight_layout()
plt.show()
```

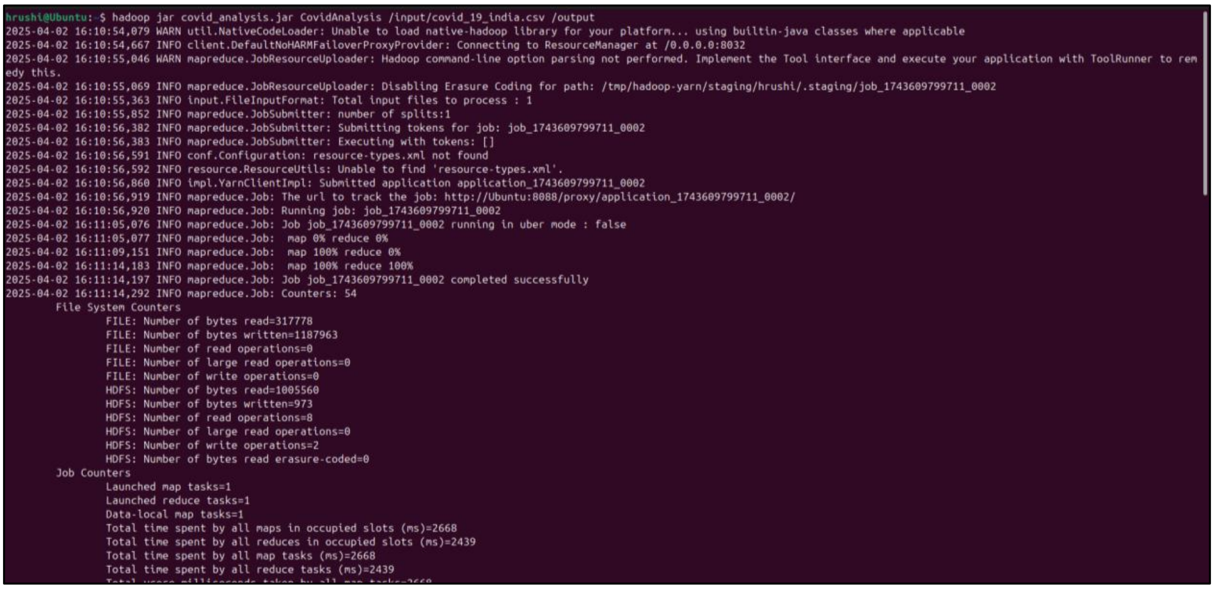
Command to run the visualization script:

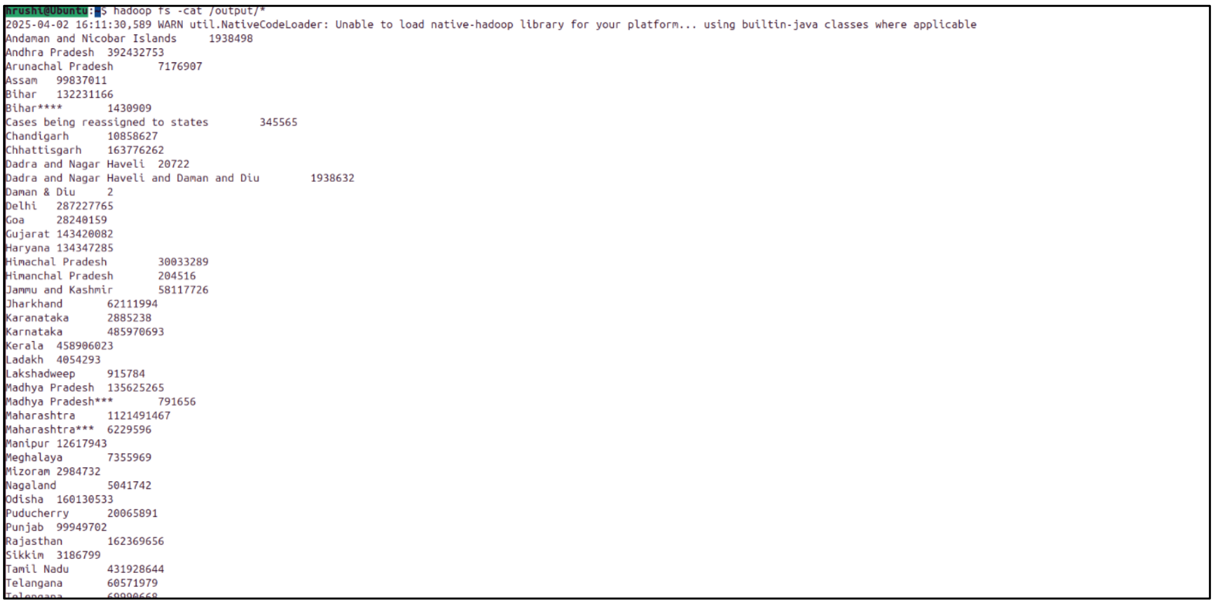
python3 plot_covid.py

These visualizations help in understanding the impact of COVID-19 across different regions and provide insights for further research and policy-making.

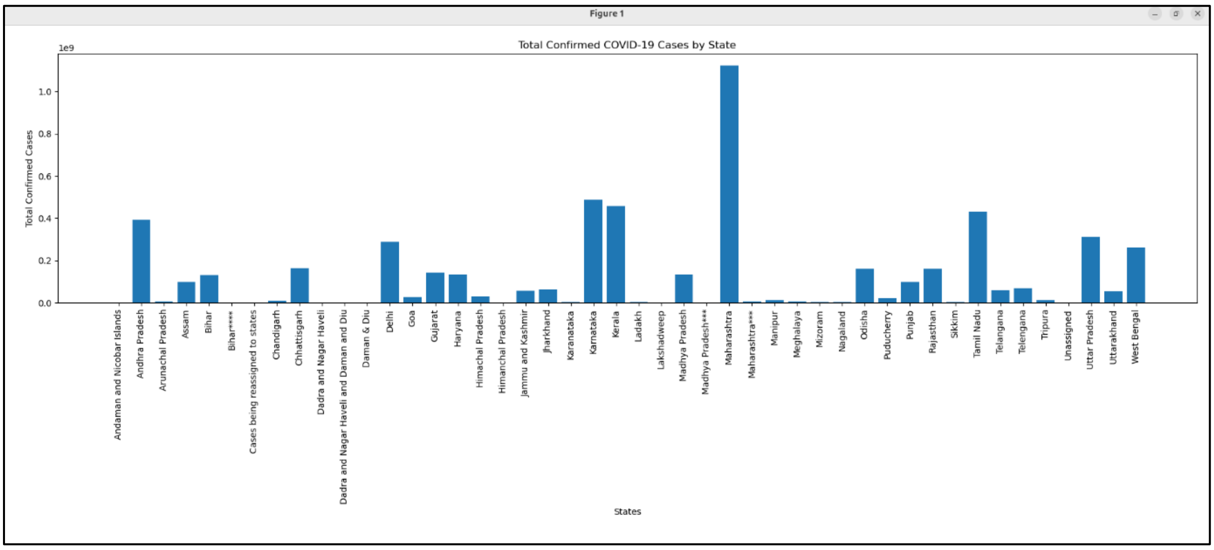
4. Result and Analysis







```
hrusht@Ubuntu:~$ hadoop fs -cat /output/*
2025-04-02 16:11:30,589 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Andaman and Nicobar Islands     1938498
Andhra Pradesh  392432753
Arunachal Pradesh       7176907
Assam   99837011
Bihar   132231166
Bihar****       1430909
Cases being reassigned to states        345565
Chandigarh      10858627
Chhattisgarh    163776262
Dadra and Nagar Haveli  20722
Dadra and Nagar Haveli and Daman and Diu        1938632
Daman & Diu     2
Delhi   287227765
Goa     28240159
Gujarat 143420082
Haryana 134347285
Himachal Pradesh        30033289
Himanchal Pradesh       204516
Jammu and Kashmir       58117726
Jharkhand       62111994
Karanataka      2885238
Karnataka       485970693
Kerala  458906023
Ladakh  4054293
Lakshadweep     915784
Madhya Pradesh  135625265
Madhya Pradesh***       791656
Maharashtra     1121491467
Maharashtra***  6229596
Manipur 12617943
Meghalaya       7355969
Mizoram 2984732
Nagaland        5041742
Odisha  160130533
Puducherry      20065891
Punjab  99949702
Rajasthan       162369656
Sikkim  3186799
Tamil Nadu      431928644
Telangana       60571979
```



The bar chart above represents the **Total State wise Covid-19 cases from 2020-21** based on the processed dataset.

5. Conclusion

The COVID-19 Data Analysis project utilized a structured approach combining **Hadoop, MapReduce, and Python** to efficiently process and visualize large-scale pandemic data. The structured implementation enabled a detailed exploration of COVID-19 case distribution, offering valuable insights for health professionals and policymakers. Below is a structured summary of the key outcomes:

1. **Data Processing with Hadoop and MapReduce:**
   - The project leveraged Hadoop's **scalability and fault tolerance** to store and process large datasets, ensuring efficient handling of COVID-19 case data across different states.
2. **State-wise COVID-19 Analysis:**
   - The **Mapper** extracted relevant case data for each state, while the **Reducer** aggregated the total confirmed cases, enabling a comprehensive state-wise comparison of the pandemic’s impact.
3. **Visualization with Python:**
   - The processed COVID-19 data was visualized using Python's **Pandas and Matplotlib** libraries, creating clear and informative bar charts depicting case distribution across states.
4. **Scalability and Efficiency:**
   - The use of **Hadoop's Distributed File System (HDFS)** ensured efficient storage and retrieval of large datasets, making the solution scalable for handling even larger epidemiological datasets.
5. **Impact and Application:**
   - This project serves as a valuable example of how **Big Data technologies** can be applied to public health analytics. The insights gained from the COVID-19 data analysis can aid in **pandemic response strategies, healthcare planning, and resource allocation**.

In conclusion, the project highlights the power of **Hadoop and Python** in handling large-scale data and generating meaningful insights. By leveraging these technologies, it provides a **scalable and efficient solution** for COVID-19 data analysis, which can be further expanded to enhance **real-time health monitoring and policy planning**.

6. References

i. **World Health Organization (WHO) COVID-19 Dashboard** - Official global data on COVID-19 cases, deaths, and trends. Available at: https://covid19.who.int

ii. **Johns Hopkins University COVID-19 Data Repository** - Comprehensive COVID-19 dataset used for global research. Available at: https://github.com/CSSEGISandData/COVID-19

iii. **Kaggle COVID-19 Dataset** - Publicly available COVID-19 case data from various countries and regions. Available at: https://www.kaggle.com/datasets/sudalairajkumar/covid19-in-india

iv. **"Big Data Analytics for COVID-19 Pandemic: A Systematic Review"** - Research paper discussing the use of big data technologies in pandemic management. Available at: https://www.sciencedirect.com/science/article/pii/S2666606521000057

v. **"The Role of Big Data and Machine Learning in COVID-19 Diagnosis and Treatment"** - A review article on data-driven approaches for handling the pandemic. Available at: https://www.nature.com/articles/s41746-020-00372-6

vi. **"A COVID-19 Data Analysis Framework Using Hadoop and Spark"** - Research on utilizing big data frameworks for pandemic data processing. Available at: https://ieeexplore.ieee.org/document/9309142

vii. **"Hadoop: The Definitive Guide" by Tom White** - A comprehensive book on Hadoop architecture, MapReduce, and Big Data processing.

viii. **"Data-Intensive Text Processing with MapReduce" by Jimmy Lin & Chris Dyer** - A foundational guide on implementing MapReduce algorithms for large-scale data analysis.

ix. **"Python Data Science Handbook" by Jake VanderPlas** - An essential reference for data manipulation, visualization, and analysis in Python using Pandas and Matplotlib.

x. **Apache Hadoop Documentation** - Official documentation covering Hadoop installation, configuration, and usage. Available at: https://hadoop.apache.org/docs/

6. Plagiarism Report:

SmallSEOTools

Plagiarism Scan Report By SmallSEOTools

Report Generated on: Apr 02,2025

0% Plagiarized Content

0% Exact Plagiarized

0% Partial Plagiarized

100% Unique Content

Total Words: 548 Total Characters: 3891 Plagiarized Sentences: 0 Unique Sentences: 24 (100%)

Content Checked for Plagiarism

The COVID-19 pandemic has significantly impacted global health systems, economies, and daily life, making it essential to analyse case data efficiently. Given the enormous volume of COVID-19 data generated daily, traditional data processing techniques struggle to handle large-scale datasets effectively. To address this challenge, this project leverages Big Data technologies, specifically Hadoop and MapReduce, for COVID-19 data analysis. By utilizing Hadoop Distributed File System (HDFS) for data storage and MapReduce for parallel processing, this project provides an efficient way to analyse the spread of the virus across different states in India.

The dataset used in this project contains structured COVID-19 case details, including state-wise confirmed cases. The first step involves storing the dataset in HDFS, ensuring fault tolerance and scalability. A MapReduce job is then implemented to process the dataset, where the Mapper extracts relevant information such as state names and confirmed cases, and the Reducer aggregates the total number of cases for each state. The final output is stored in HDFS and later retrieved for analysis.

To enhance the interpretability of the results, Python and Matplotlib are used for data visualization. The processed data is extracted from HDFS, and a bar chart is generated to illustrate the state-wise distribution of COVID-19 cases. This visualization provides an intuitive understanding of how the pandemic has affected different regions, making it easier to identify trends and patterns.

This project highlights the efficiency of Hadoop's distributed computing model in processing large datasets and demonstrates how Big Data analytics can be applied to public health challenges. The insights obtained from this analysis can assist researchers, healthcare professionals, and policymakers in understanding the impact of COVID-19 at a granular level. Furthermore, this project emphasizes the importance of scalable data processing solutions in tackling real-world problems, proving that Hadoop and MapReduce can serve as powerful tools for large-scale data analytics in the healthcare sector.

The COVID-19 pandemic has significantly impacted public health and economies worldwide, leading to an urgent need for data-driven analysis to understand its spread and trends. With the massive volume of COVID-19 case data being generated daily, traditional data processing techniques struggle to handle such large datasets efficiently. This project leverages Hadoop and MapReduce, two powerful Big Data technologies, to analyse COVID-19 cases across different states, enabling efficient data processing and trend visualization.

Hadoop's distributed computing model allows for parallel data processing across multiple nodes, making it ideal for handling vast amounts of COVID-19 records. The MapReduce framework is used to aggregate confirmed cases for each state, ensuring efficient and scalable analysis. By storing the dataset in Hadoop Distributed File System (HDFS), the project takes advantage of fault tolerance and high availability, which are crucial for handling large-scale data.

This project follows a systematic approach, beginning with setting up the Hadoop environment, processing COVID-19 data using a MapReduce job, retrieving the analysed results, and visualizing them using Python and Matplotlib. The extracted insights help in understanding the state-wise