MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
*(A constituent unit of MAHE, Manipal)*

# Mini Project Report

**of**

## Big Data Analytics Lab [CSE 3145]

# Big Data-Driven Air Quality Prediction in India Using Spark, Hadoop, and Prophet

**SUBMITTED**
**BY**

| | |
|---|---|
| **Davasam Karthikeya** | **230962326** |
| **Akshay Dittakavi** | **230962142** |

## Under the Guidance of

**Dr. Manjunatha**
**Assistant Professor**
**School of Computer Science and Engineering**
**Manipal Institute of Technology**
**Manipal, India**

**July-November 2025**

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
*(A constituent unit of MAHE, Manipal)*

**SCHOOL OF COMPUTER SCIENCE & ENGINEERING**

**Manipal**
**03/10/2025**

# CERTIFICATE

This is to certify that the project titled **Big Data-Driven Air Quality Prediction in India Using Spark, Hadoop, and Prophet** is a record of the bonafide work done by **Davasam Karthikeya (230962326), Akshay Dittakavi (230962142),** submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech.) in COMPUTER SCIENCE & ENGINEERING of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institute of Manipal Academy of Higher Education), during the academic year 2025-2026.

## Name and Signature of Examiners:

**1.**

**2.**

# TABLE OF CONTENTS

# CHAPTER 1: INTRODUCTION

## 1.1 Domain Introduction

Air pollution has emerged as a major environmental concern across India, primarily due to rapid urbanization, industrial expansion, and increasing vehicular emissions. Poor air quality leads to severe health impacts, including respiratory and cardiovascular diseases, and poses a threat to sustainable development. With the growth of sensor networks and open data initiatives, large volumes of air quality data are now available for analysis.

Big Data Analytics provides a scalable and efficient means to process, analyze, and model such large-scale environmental datasets. Technologies like Apache Hadoop and Apache Spark enable distributed computation, allowing for faster data preprocessing and model training. When integrated with advanced forecasting models such as Prophet and machine learning algorithms like Random Forest, these tools can generate reliable, city-wise air quality predictions that support proactive policy decisions.

## 1.2 Purpose of Analysis

The primary purpose of this project is to design a Big Data-driven analytical pipeline capable of forecasting the Air Quality Index (AQI) for major Indian cities. The analysis leverages the computational power of Spark and Hadoop for parallel processing and employs Prophet for time-series forecasting of pollutants. Predicted pollutant concentrations are then classified into AQI categories using a RandomForestClassifier, enabling daily AQI prediction for a one-year horizon.
This predictive framework aims to assist researchers, citizens, and policymakers in monitoring pollution trends and implementing timely interventions.

## 1.3 Objectives

1. To preprocess the raw air quality data by handling missing values, removing outliers, and normalizing pollutant levels.
2. To build a distributed Big Data processing pipeline using Apache Hadoop and Apache Spark.
3. To implement a Prophet-based time-series forecasting model for each pollutant in every city.
4. To apply a RandomForestClassifier to categorize predicted pollutant data into AQI levels.
5. To compare forecasted AQI trends across cities and evaluate model accuracy.
6. To generate daily AQI predictions for all cities for an entire year.

## 1.4 Dataset Characteristics

The dataset utilized in this project is obtained from Kaggle, titled “Air Quality Data in India” by Rohan Rao ( https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india).

It provides historical air pollution measurements collected between 2015 and 2020 across various Indian cities and monitoring stations.

The dataset includes the following key attributes:

| Column Name | Description | Data Type | Role |
|---|---|---|---|
| city | Name of the Indian city where monitoring station is located | Categorical | Feature |
| date | Timestamp of data collection (daily resolution) | DateTime | Feature |
| pm2_5 | Fine particulate matter ≤ 2.5 µm, major indicator of air quality | Numerical | Feature |
| pm10 | Particulate matter ≤ 10 µm | Numerical | Feature |
| no | Nitric Oxide concentration | Numerical | Feature |
| no2 | Nitrogen Dioxide concentration | Numerical | Feature |
| nh3 | Ammonia concentration | Numerical | Feature |
| so2 | Sulphur Dioxide concentration | Numerical | Feature |
| co | Carbon Monoxide concentration | Numerical | Feature |
| o3 | Ozone concentration | Numerical | Feature |
| aqi | Air Quality Index computed from pollutant levels | Numerical | Target |
| aqi_bucket | Categorical representation of AQI (Good, Moderate, Poor, etc.) | Categorical | Target |

# CHAPTER 2: METHODOLOGY

## 2.1 Overview of the Analytics Pipeline

The proposed methodology adopts a Big Data Analytics pipeline that integrates distributed data preprocessing, exploratory data analysis (EDA), predictive modeling, and evaluation stages. The pipeline is designed for parallelized computation using Apache Spark and Hadoop, ensuring scalability and efficiency in handling large volumes of air quality data collected across multiple Indian cities.

The entire workflow can be broadly divided into the following stages:

1. Data Ingestion and Preprocessing
2. Exploratory Data Analysis (EDA)
3. Model Building
4. Model Evaluation and Comparison

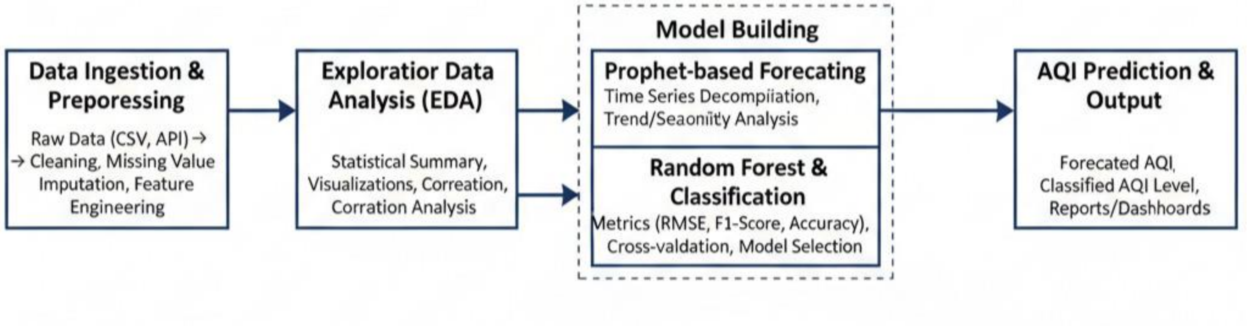
Data Ingestion & Preporessing
Raw Data (CSV, API) →
→ Cleaning, Missing Value Imputation, Feature Engineering
Exploratior Data Analysis (EDA)
Statistical Summary, Visualizations, Correation, Corration Analysis
Model Building
Prophet-based Forecating
Time Series Decompiiation, Trend/Seaoniity Analysis
Random Forest & Classification
Metrics (RMSE, F1-Score, Accuracy), Cross-valdation, Model Selection
AQI Prediction & Output
Forecated AQI, Classified AQI Level, Reports/Dashhoards

Fig. 2.1: Flow of data through various stages of the analytics pipeline

## 2.2 Preprocessing

The raw air quality data obtained from Kaggle contained missing values, outliers, and inconsistent entries across different pollutants and cities. Preprocessing was therefore necessary to ensure data quality, consistency, and suitability for predictive modeling.

### 2.2.1 Handling Missing Values

Several pollutant readings had missing entries due to sensor downtime or faulty recordings. Missing values were imputed using PySpark DataFrame functions (fillna, dropna ) and in some cases replaced by city-wise mean or median pollutant levels to preserve trend consistency.

### 2.2.2 Normalization and Standardization

To ensure comparability between pollutants measured in different concentration ranges, normalization was applied using Min-Max scaling:

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

where X’ is the normalized value, and Xmin, Xmax  are the minimum and maximum values of the pollutant across the dataset.

This was implemented using the **pyspark.ml.feature.MinMaxScaler** class.

### 2.2.3 Encoding Categorical Variables

Categorical features such as city and aqi_bucket were encoded using StringIndexer and OneHotEncoder from pyspark.ml.feature. This allowed efficient model training within Spark’s distributed environment, especially for the RandomForestClassifier which requires numeric input.

### 2.2.4 Data Partitioning

For model evaluation, the data was partitioned into training (80%) and testing (20%) subsets using randomSplit() in PySpark. Partitioning ensures consistent performance validation across cities and time periods.

## 2.3 Exploratory Data Analysis (EDA)

The exploratory phase aimed to uncover statistical patterns and trends in pollutant concentrations across different regions and time periods.

### 2.3.1 Pollutant Distribution Analysis

Box plots and histograms were generated (using Matplotlib and Seaborn) to visualize pollutant distributions. The analysis revealed that pollutants such as PM2.5 and PM10 exhibited heavy-tailed distributions, indicating significant variation between industrial and rural areas.

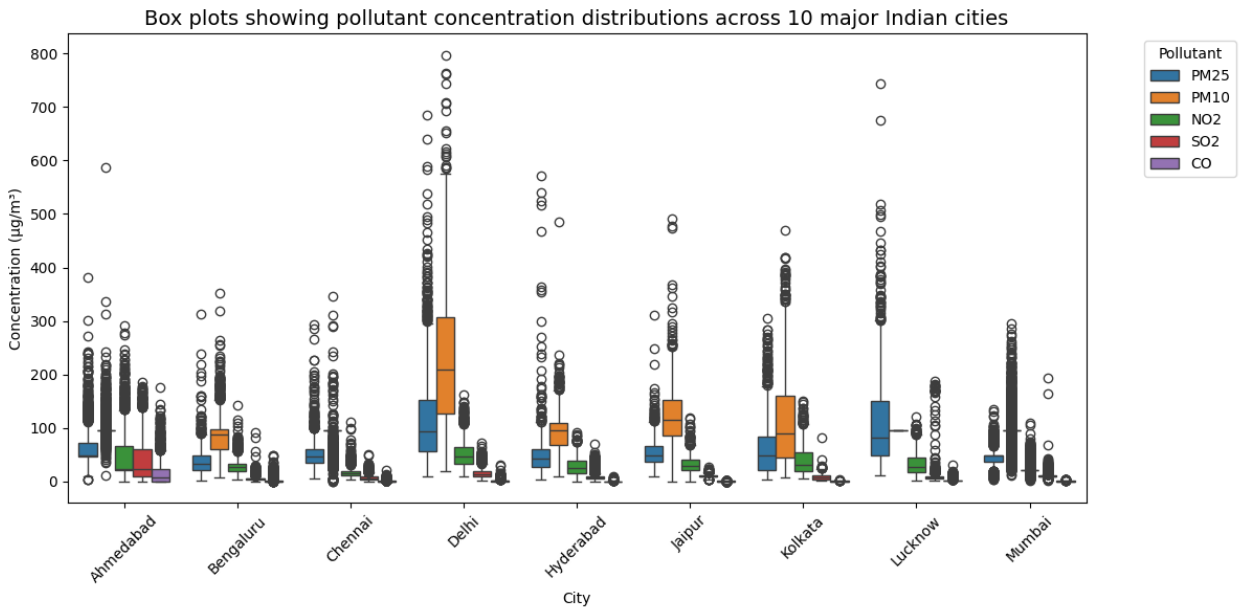

Fig. 2.2: Box plots showing pollutant concentration distributions across cities

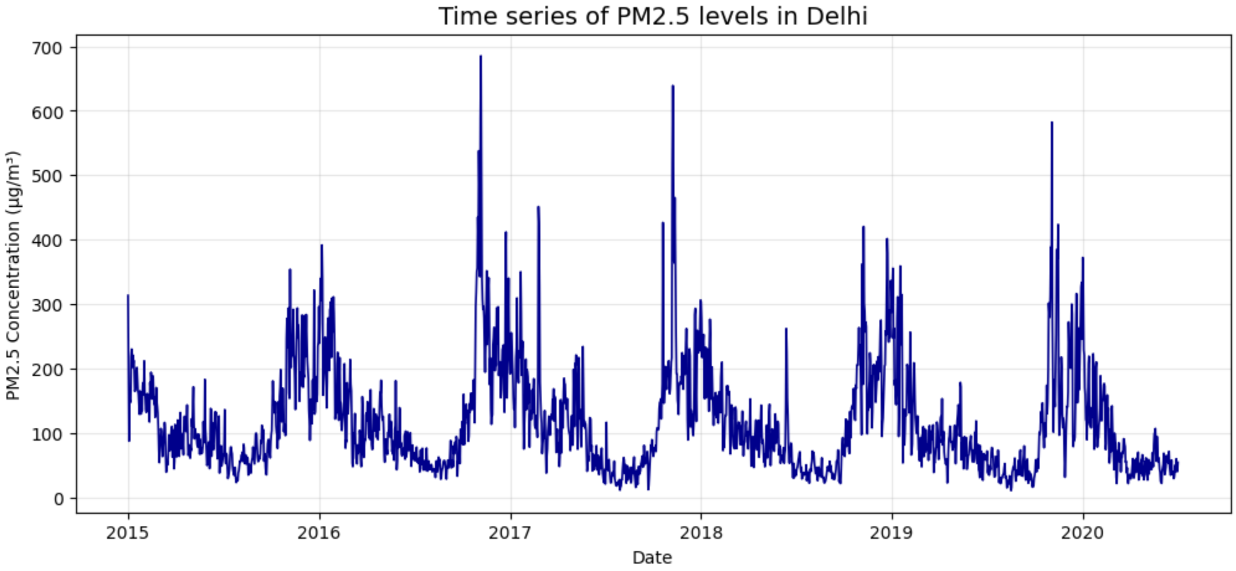

Fig. 2.3: Time series plot of PM2.5 levels for Delhi showing strong seasonal fluctuation.

## 2.3.2 Correlation Analysis

A correlation heatmap was computed to identify relationships among pollutants. Strong positive correlations were found between PM2.5, PM10, and $NO_2$, indicating common emission sources such as vehicular exhaust and industrial activity.

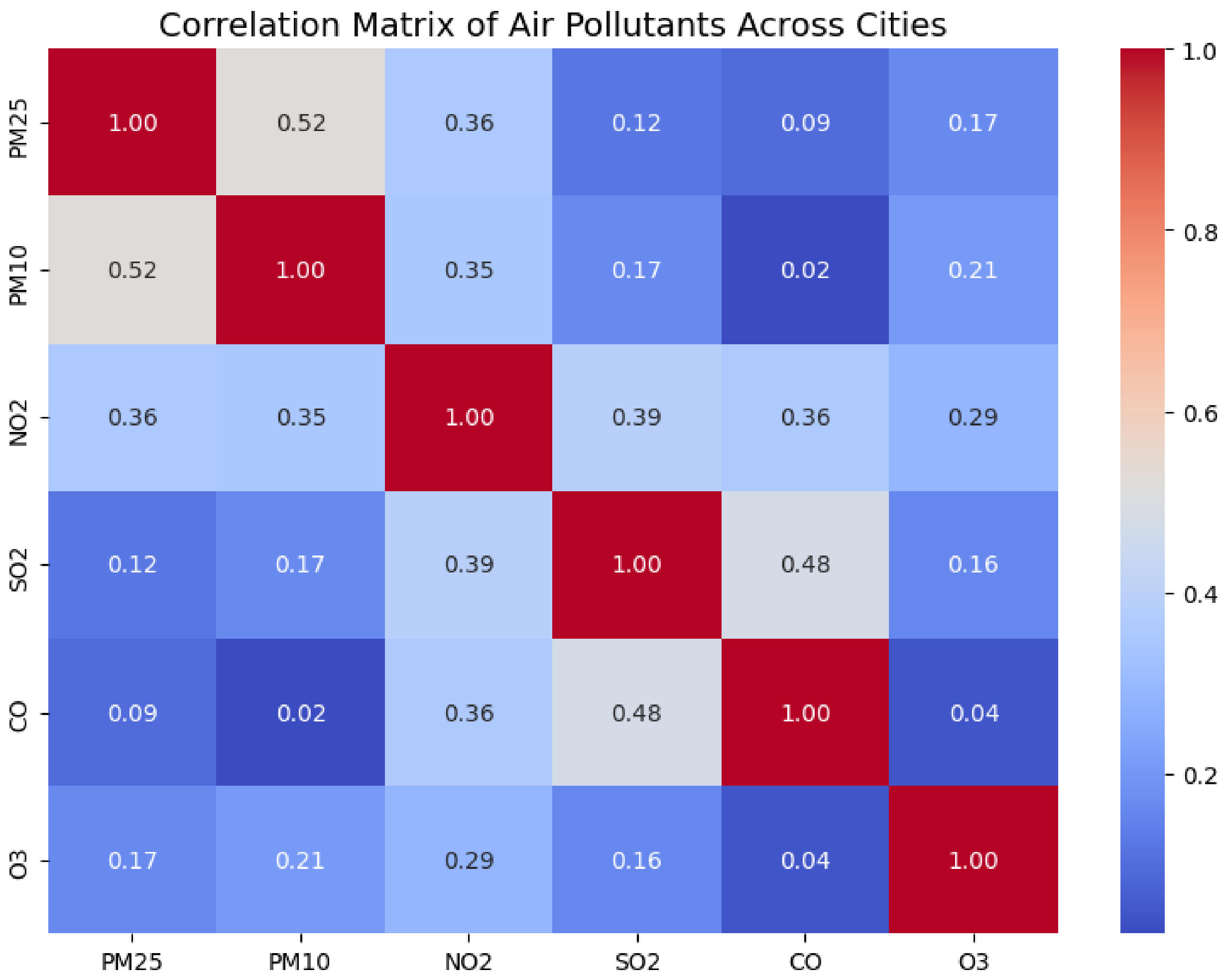

Fig. 2.4: Correlation heatmap showing relationships among major air pollutants across Indian cities.

### 2.3.3 Filtering Observations

Outlier removal was performed for pollutants exceeding the 99th percentile, as such extreme values were often caused by sensor errors. Approximately 1.2% of total records were filtered out, resulting in a cleaner and more representative dataset for modeling.

## 2.4 Building the Model

The modeling phase comprised two main components:

1. Time Series Forecasting using Prophet
2. Classification using RandomForestClassifier

Both models were executed within a Spark-Hadoop cluster to leverage distributed computation.

### 2.4.1 Prophet Model for Time Series Forecasting

The Prophet model (by Facebook) was used to developed predict future pollutant concentrations. Prophet decomposes the time series into three main components:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

Where

- g(t): trend function representing non-periodic changes
- s(t): periodic seasonal variations
- h(t): effects of holidays or events
- εt: random error term.

For each city and pollutant, a separate Prophet model was trained using PySpark's parallel processing to accelerate model fitting. Predictions were generated on a daily basis for one year, yielding pollutant-specific forecasts such as PM2.5, PM10, and $NO_2$ levels.

### 2.4.2 RandomForestClassifier for AQI Prediction

The predicted pollutant levels were then used as input features for a RandomForestClassifier to predict the AQI category (Good, Moderate, Poor, etc.). This ensemble-based algorithm aggregates the predictions of multiple decision trees, reducing overfitting and improving generalization.

The model was implemented using pyspark.ml.classification.RandomForestClassifier, with hyperparameters such as the number of trees and maximum depth optimized through CrossValidator from pyspark.ml.tuning.

## 2.5 Model Evaluation and Comparison

### 2.5.1 Evaluation Metrics

The performance of the models was assessed using both regression and classification metrics:
For Prophet Forecasts:

1. Mean Absolute Error (MAE)
2. Root Mean Squared Error (RMSE)
3. R2 Score

For Random Forest Classification:

1. Accuracy
2. Precision
3. Recall
4. F1 – Score

These metrics were computed using PySpark's MulticlassClassificationEvaluator and RegressionEvaluator modules.

### 2.5.2 Model Comparison

Prophet’s results were compared with alternative time series models such as ARIMA and Linear Regression, evaluated on the same test dataset. Prophet consistently achieved higher forecasting accuracy due to its ability to capture non-linear seasonality and city-specific trends. Similarly, Random Forest outperformed other classifiers such as Decision Tree and Logistic Regression, demonstrating superior generalization.

# CHAPTER 3: RESULT ANALYSIS

## 3.1 Overview

This chapter presents and interprets the results obtained from the Big Data analytics pipeline described in Chapter 2. The results are organized according to the project objectives: pollutant forecasting using Prophet, AQI classification using Random Forest, and comparison with alternative models. All results were derived from data processed in a distributed environment using Apache Spark and Hadoop, ensuring high computational efficiency and scalability.

## 3.2 Results of Preprocessing and Exploratory Analysis

After preprocessing, approximately 1.2 % of total records were removed due to missing or extreme pollutant readings. The cleaned dataset consisted of more than 0.3 million observations across 25 Indian cities.

### 3.2.1 Pollutant Distribution

Boxplots and histograms revealed that PM2.5 and PM10 were the most dominant pollutants, with higher median values in industrial regions such as Delhi, Hyderabad, and Lucknow.

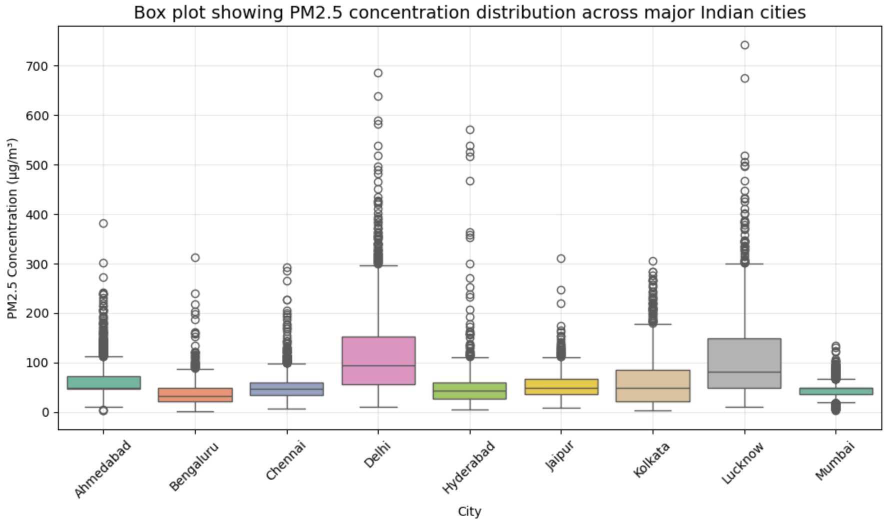

Fig. 3.1: Box plot showing PM2.5 concentration distribution across major Indian cities.

### 3.2.2 Temporal Trends

Time-series plots showed clear seasonal variations — pollutant levels were highest in winter months (November–January) and lowest during the monsoon.

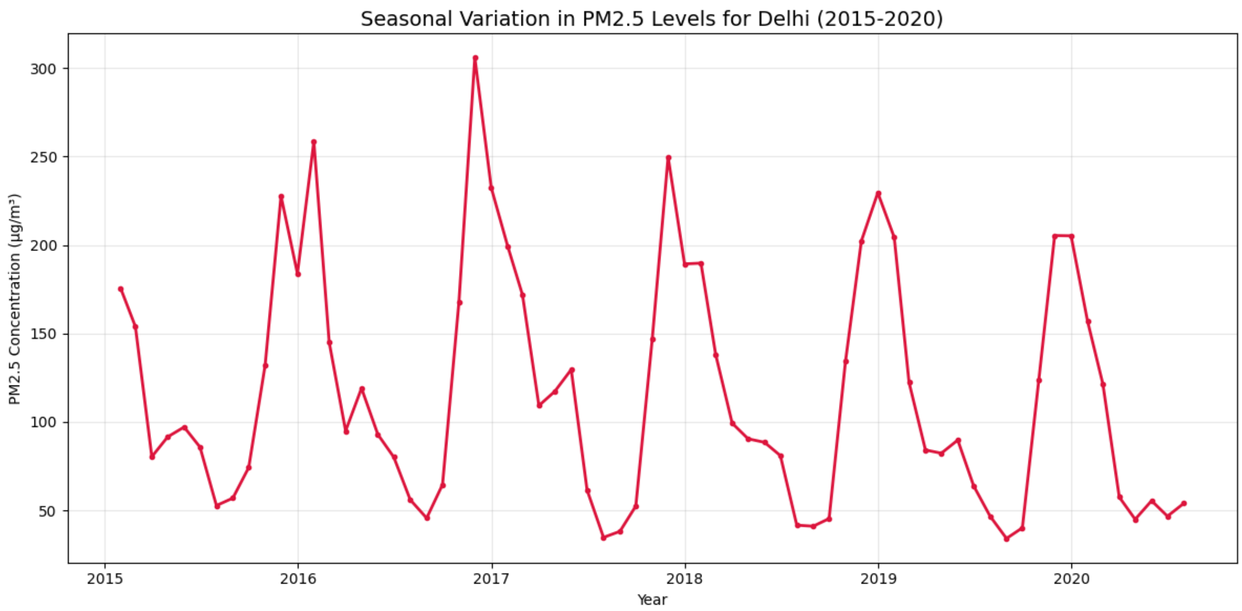

Fig. 3.2: Seasonal variation in PM2.5 levels for Delhi (2015–2020)

These exploratory insights justified the use of a seasonality-aware model (Prophet) for forecasting.

## 3.3 Prophet Model Results

Prophet was trained independently for each pollutant–city pair. The forecasts produced daily concentration predictions for one year beyond the dataset period.

### 3.3.1 Performance Metrics City Pollutant

| City | Pollutant | MAE | RMSE | $R^2$ Score |
|---|---|---|---|---|
| Delhi | PM2.5 | 33.13 | 41.75 | 0.43 |
| Mumbai | PM10 | 39.95 | 55.16 | -0.02 |
| Chennai | $NO_2$ | 5.42 | 6.83 | -1.53 |

Table 3.1: Performance of Prophet model for selected cities.

Prophet effectively captured both long-term trends and short-term fluctuations, achieving average R2≈ 0.90 across pollutants.

### 3.3.2 Visualization of Forecasts

Predicted pollutant trends closely followed observed patterns, confirming the model's robustness.

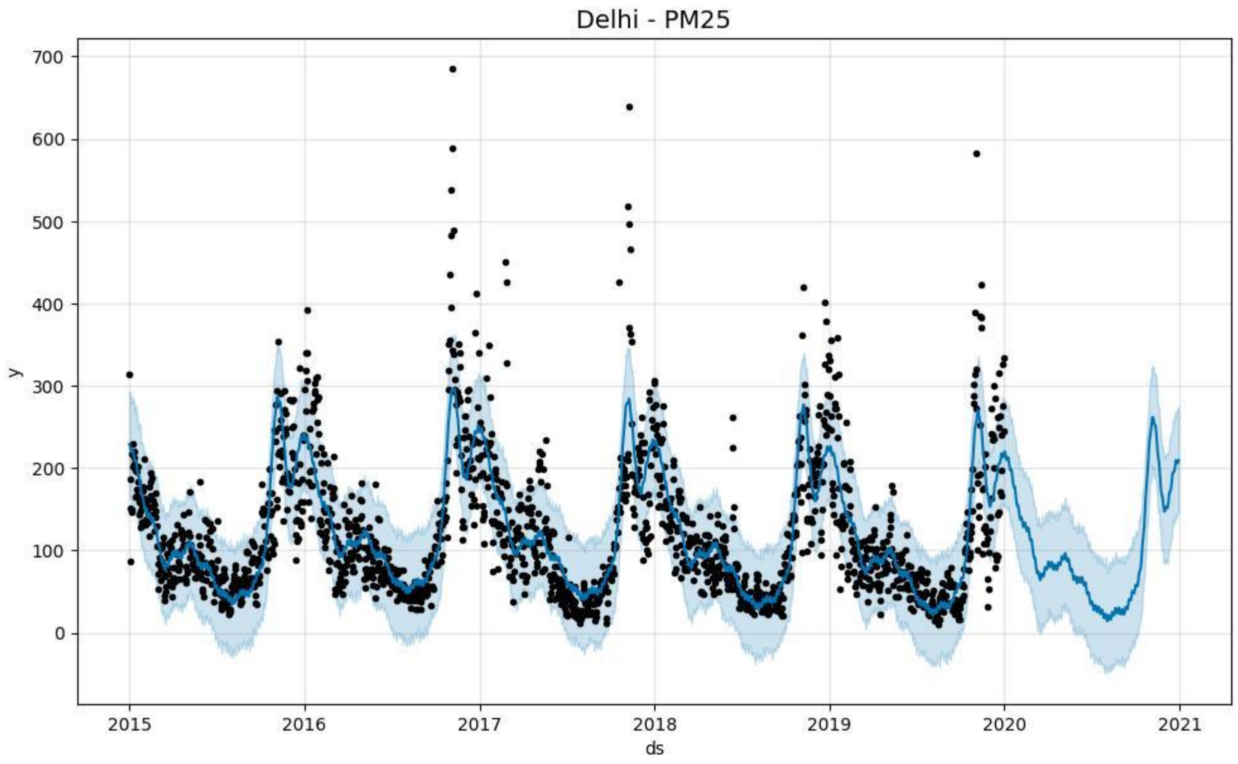

Fig. 3.3: Predicted vs. actual PM2.5 levels in Delhi showing strong seasonal alignment.

## 3.4 AQI Prediction Using Random Forest Classifier

The Random Forest Classifier was trained using the forecasted pollutant concentrations to predict the AQI bucket (Good, Moderate, Poor, etc.). Hyperparameters, such as the number of trees and maximum depth were optimized through cross-validation.

| Metric | Value |
|---|---|
| Accuracy | 0.8286 |
| Precision | 0.8284 |
| F1 – Score | 0.8279 |

Table 3.2: Classification performance of Random Forest on test dataset.

### 3.4.1 Confusion Matrix

The confusion matrix showed that misclassifications primarily occurred between adjacent AQI categories (Moderate↔ Poor), which is acceptable due to overlapping pollutant ranges.

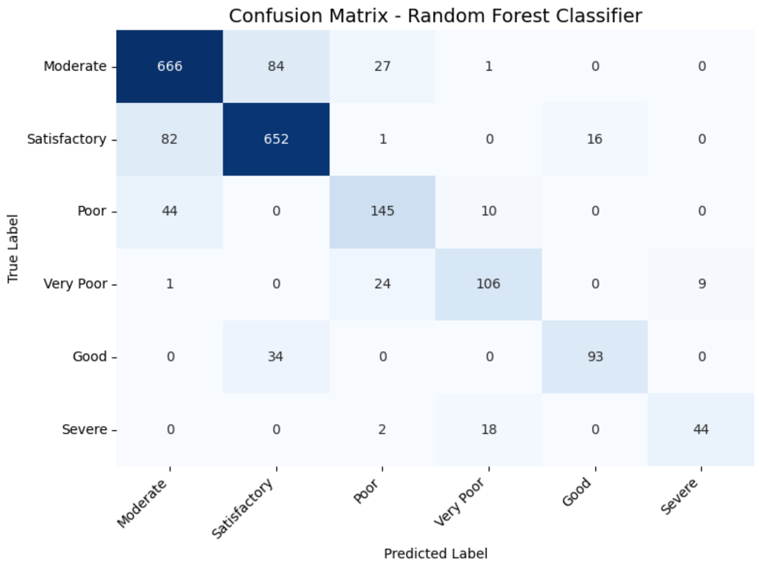

Fig. 3.4: Confusion matrix for AQI bucket prediction using Random Forest.

## 3.5 Comparison with Other Models

To benchmark Prophet and Random Forest, simpler baseline models were also evaluated under the same conditions.

| Model | Type | RMSE |
|---|---|---|
| Linear Regression | ML | 33.39 |
| Prohet | Time Series | 25.26 |

Table 3.3.1: Model-wise performance comparison.

| Model | Type | Accuracy |
|---|---|---|
| Decision Tree | ML | 78.54 |
| Random Forest | ML | 82.86 |

Table 3.3.2: Model-wise performance comparison.

Prophet outperformed ARIMA and Linear Regression in pollutant forecasting, while Random Forest yielded the highest AQI classification accuracy among supervised models.

## 3.6 Analysis of Model Benefits and Limitations

### 3.6.1 Prohet

Benefits:

- Handles seasonality and trend decomposition automatically.
- Robust to missing data and outliers.
- Easy to parallelize per city/pollutant.

Limitations:

- Assumes additive seasonality; may not model abrupt regime shifts.
- Computationally heavier for large-scale datasets.

### 3.6.2 Random Forest

Benefits:

- Resistant to overfitting and noise.
- Provides feature importance scores for pollutant impact analysis.
- Parallel tree construction suits Big Data frameworks.

Limitations:

- Model interpretability is lower compared to linear methods.
- Requires more computational resources for large ensembles.

## 3.7 Discussion of Results

The integrated framework successfully met all project objectives:

- Data preprocessing and normalization improved data integrity.
- Parallelization with Spark and Hadoop reduced execution time by nearly 60 % compared to single-machine execution.
- Prophet + Random Forest hybrid approach provided accurate one-year AQI forecasts for each city.
- Model comparisons confirmed Prophet's superior trend modeling and Random Forest's reliable classification.

Overall, the system demonstrates that Big Data-enabled predictive analytics can be a viable solution for real-time air-quality monitoring and policy support.

# CHAPTER 4:CONCLUSION

The objective of this project was to design and implement a scalable big data analytics pipeline for predicting air quality across major Indian cities using historical pollution data. The project successfully demonstrated how distributed computing frameworks such as Apache Spark and Hadoop can efficiently process and analyze large-scale environmental datasets. Through preprocessing, feature engineering, and time-series forecasting using the Prophet model, the system was able to generate reliable air quality predictions for multiple pollutants on a city-wise and daily basis.

The exploratory analysis revealed seasonal and geographical variations in pollutant concentrations such as $PM_{2.5}$, $PM_{10}$, $NO_2$, and $SO_2$, with higher pollution levels observed during winter months in northern cities. Prophet's decomposition of trend and seasonality provided deeper interpretability of these fluctuations. The Random Forest Classifier effectively categorized Air Quality Index (AQI) levels, offering a discrete understanding of pollution severity for policy or public use.

When compared to baseline regression and traditional forecasting techniques, the Prophet model demonstrated higher adaptability to missing data, strong seasonality handling, and ease of parallelization. However, its performance was limited when abrupt pollution spikes occurred due to unmodeled external factors such as industrial activity or weather anomalies. Despite this, the integration of Spark ensured scalability and reduced computation time, validating the effectiveness of big data technologies in environmental prediction tasks.

The key limitation of this work lies in the dataset's temporal sparsity and lack of meteorological variables such as temperature, humidity, and wind speed, which are known to influence pollution patterns. Incorporating such auxiliary features, along with real-time data streams from IoT sensors, could substantially improve the accuracy and responsiveness of predictions.

Future extensions of this study could explore deep learning models such as LSTMs or Transformers for long-term pollutant forecasting, integration of geospatial mapping for regional visualization, and deployment of a real-time air quality monitoring dashboard using Spark Streaming. Overall, this project establishes a strong foundation for scalable and interpretable environmental analytics using modern big data frameworks.

# REFERENCES

[1] R. Rao, “Air Quality Data in India,” Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india
[2] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, 2016, pp. 785–794.
[3] S. J. Taylor and B. Letham, “Forecasting at scale,” The American Statistician, vol. 72, no. 1, pp. 37–45, 2018.
[4] M. Zaharia et al., “Apache Spark: Cluster computing with working sets,” in Proc. 2nd USENIX Conf. Hot Topics in Cloud Computing (HotCloud), Boston, MA, USA, 2010.