Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 46 No. 1 (2025)

Exploring the Potential of Deep Learning Techniques in Air Pollution Prediction

Raj Kishor Chauhan[1], Anurag Upadhyay[2], Mukul Kumar Singh[3], Anshul Kumar[4]

Assistant Professor IIMT College of Management Gr.Noida[1], Assistant Professor KCCITM Gr.Noida[2], Assistant Professor MJPRU Bareilly (U.P)[3], IIMT College of Management Gr.Noida[4]

Abstract: - Air pollution prediction is an essential aspect of modern environmental science, as it helps in anticipating pollution levels and taking necessary measures to reduce harmful effects on public health. Machine learning (ML) has proven to be a powerful tool for predicting air pollution levels, as it can identify patterns in large datasets and make accurate predictions. The air quality index (AQI) is a metric used to report air quality. It calculates the short-term effects of air pollution on an individual's health. Public education on the harmful health effects of local air pollution is the aim of the AQI. In Indian cities, the level of air pollution has dramatically increased. The air quality index can be calculated mathematically in a number of ways. One of the most intriguing methods for predicting and analyzing AQI is data mining. Finding the best technique for AQI prediction to support climate control is the goal of this paper. The best solution can be found by refining the most successful approach. As a result, this paper's work includes extensive research as well as the use of innovative methodologies.

Keywords*:* ML, AQI , PM2.5 , PM10, MSE

1. Introduction

Air pollution results from various sources, such as vehicle exhaust, industrial operations, and atmospheric conditions. The primary pollutants in the air consist of:

- **PM2.5 (Particulate Matter 2.5)**: Fine particles that can enter the lungs and bloodstream.
- **PM10**: Coarse particles.
- **NO2 (Nitrogen Dioxide)**: Emitted from vehicles and industries.
- **CO (Carbon Monoxide)**: Released by vehicles.
- **SO2 (Sulfur Dioxide)**: Produced by burning fossil fuels.
- **O3 (Ozone)**: A secondary pollutant formed when sunlight reacts with other pollutants.

The objective is to estimate pollutant levels at various times and locations, which can inform health alerts, regulatory actions, and preventive measures.

Data Collection

The first step in machine learning for air pollution prediction is gathering data. Key data sources include:

- **Air Quality Index (AQI)**: The AQI is a standardized measure of air quality. It often includes data for multiple pollutants.
- **Weather Data**: Temperature, wind speed, humidity, and atmospheric pressure play crucial roles in pollutant dispersion.
- **Traffic Data**: In urban areas, vehicle emissions can significantly impact air quality.
- **Industrial Activity Data**: Emissions from factories and power plants need to be tracked.
- **Geographical Data**: Certain areas might have higher pollution levels due to industrial zones or geographical factors.

Several machine learning models, such as Support Vector Machine (92%), Decision Tree (86%), Linear SVC (84%), Random Forest (82%), Logistic Regression (83%), K-Nearest Neighbor (89%), and Naïve Bayes (76%), are used in my research to forecast air quality[7].

II. METHODOLOGY

 Objective: Define the specific air pollution prediction task, such as predicting the concentration of pollutants (PM2.5, PM10, NO2, CO, O3) or forecasting[4] the Air Quality Index (AQI) for future times.

 Scope: Determine whether the model will be used for short-term predictions (e.g., next 24 hours) or long-term forecasts (e.g., weekly or monthly). This also helps in deciding the features to include (weather conditions, traffic data, etc.).

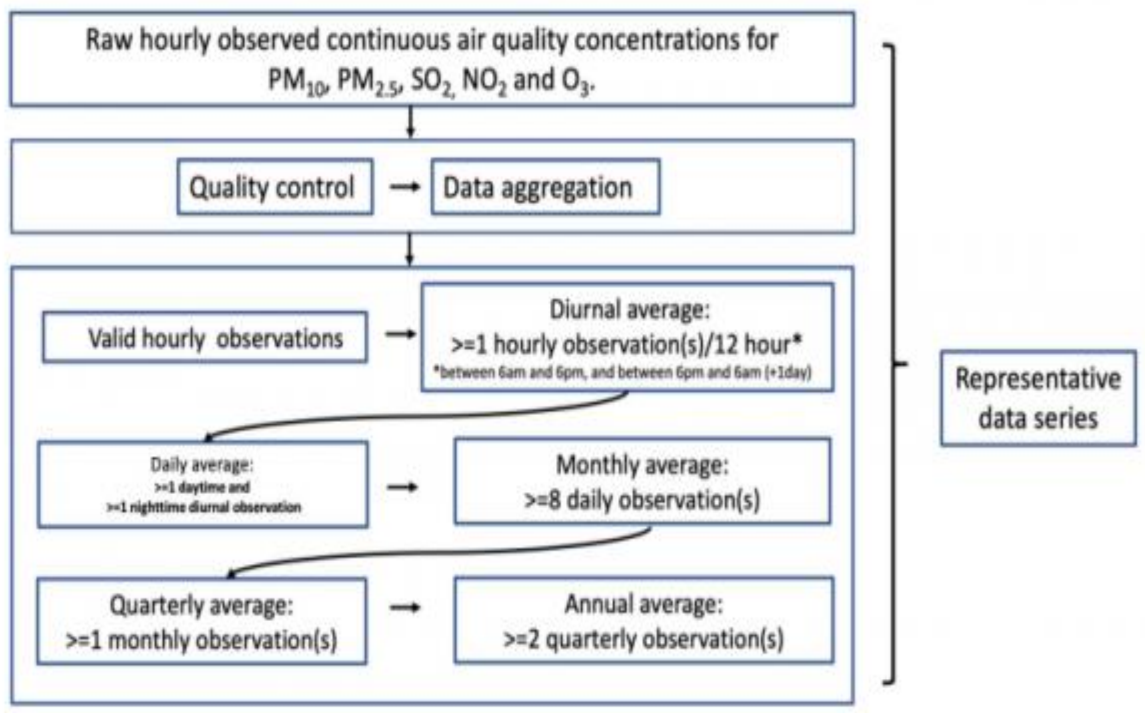
 Stakeholders: Identify who will use the predictions (e.g., public health officials, city planners, or citizens).

.2.Data Collection sources

Sources of Data: Data is the backbone of any machine learning model, and accurate, diverse data is essential for a successful prediction model.

Air Quality Data: Historical pollutant levels (PM2.5, PM10, NO2, CO, SO2, O3) collected from air quality monitoring stations[2].

Weather Data: Temperature, humidity, wind speed, and atmospheric pressure, as these factors influence pollutant dispersion.


Raw hourly observed continuous air quality concentrations for PM_{10}, $PM_{2.5}$, SO_2, NO_2 and O_3.
Quality control
Data aggregation
Valid hourly observations
Diurnal average: >=1 hourly observation(s)/12 hour*
*between 6am and 6pm, and between 6pm and 6am (+1day)
Daily average: >=1 daytime and >=1 nighttime diurnal observation
Monthly average: >=8 daily observation(s)
Quarterly average: >=1 monthly observation(s)
Annual average: >=2 quarterly observation(s)
Representative data series

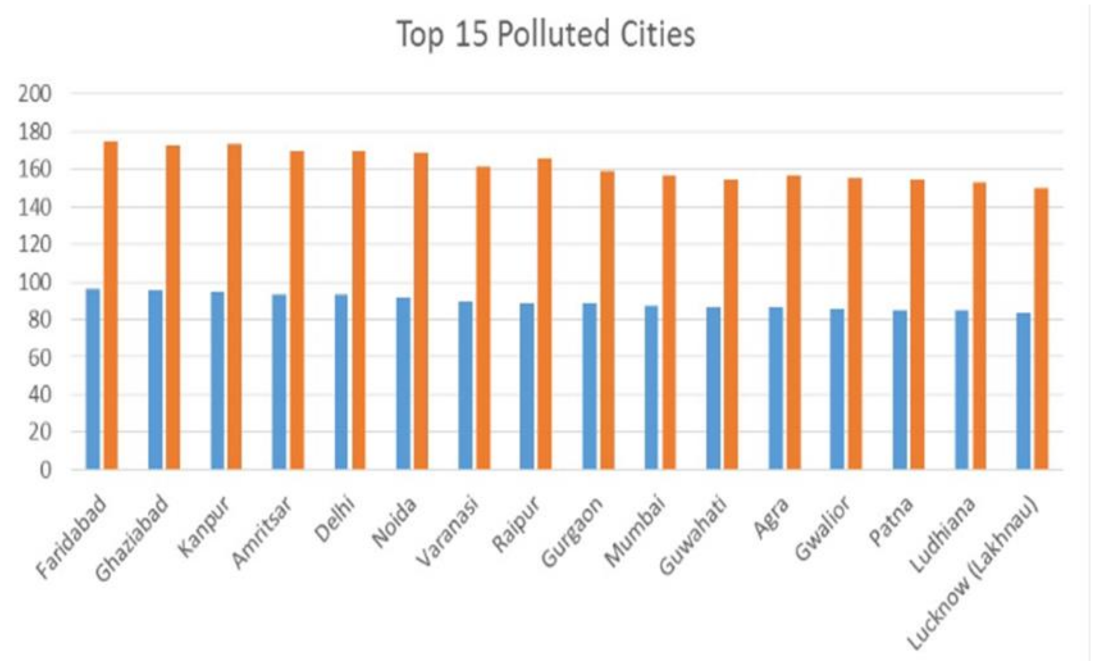
Traffic Data: Vehicle counts and congestion levels, which impact emission levels from vehicles.

Geographical and Temporal Data: Location (urban, rural, industrial zones) and time of day (morning, evening) influence air pollution levels.

Industrial Emission Data: Information on emissions from factories and power plants.

Public Health Data: Incidents of respiratory diseases or hospital admissions related to air pollution.

Data Frequency: Decide the granularity of the data (e.g., hourly, daily), depending on the prediction interval.



2. How is AQI calculated

1. The Sub-indices for individual pollutants at a monitoring location are calculated using its 24-hourly average concentration value (8-hourly in case of CO and O3) and health breakpoint concentration range. The worst sub-index is the AQI for that location.

2. All the eight pollutants may not be monitored at all the locations. Overall AQI is calculated only if data are available for minimum three pollutants out of which one should necessarily be either PM2.5 or PM10. Else, data are considered insufficient for calculating AQI. Similarly, a minimum of 16 hours' data is considered necessary for calculating sub index[6].

3. The sub-indices for monitored pollutants are calculated and disseminated, even if data are inadequate for determining AQI. The Individual pollutant-wise sub-index will provide air quality status for that pollutant.

4. The web-based system is designed to provide AQI on real time basis. It is an automated system that captures data from continuous monitoring stations without human intervention, and displays AQI based on running average values (e.g. AQI at 6am on a day will incorporate data from 6am on previous day to the current day).

5. For manual monitoring stations, an AQI calculator is developed wherein data can be fed manually to get AQI value[7].

Algorithm:

```
# Step 1: Data Collection
# - Collect historical data on air pollution (e.g., PM2.5, PM10, CO, NO2, Ozone levels)
# - Collect additional data (weather, traffic, etc.) from relevant sources
# - Ensure data includes timestamps for time-series prediction
data = LoadData("air_pollution_dataset.csv")
# Step 2: Data Preprocessing
# - Handle missing values by either removing or imputing
# - Normalize or scale the features
```

```
    # - Generate new features like moving averages, lag features, etc.
    data = HandleMissingValues(data)
    data = NormalizeFeatures(data)
    data = FeatureEngineering(data)
    # Step 3: Data Splitting
    # - Split the dataset into training and test sets (e.g., 80% training, 20% testing)
    TrainData, TestData = SplitData(data, TrainSize = 0.8)
    # Step 4: Feature Selection (Choose features relevant to the prediction)
    X_train, y_train = ExtractFeaturesAndTarget(TrainData)
    X_test, y_test = ExtractFeaturesAndTarget(TestData)
    # Step 5: Model Selection
    # - Choose an appropriate machine learning model (e.g., Random Forest, Neural Networks, SVM)
    model = InitializeModel("RandomForest")  # Or any other ML model
    # Step 6: Train the Model
    model = TrainModel(model, X_train, y_train)
    # Step 7: Hyperparameter Tuning (optional)
    # - If required, use Grid Search or Random Search for better model parameters
    model = TuneHyperparameters(model, X_train, y_train)
    # Step 8: Model Evaluation
    # - Evaluate the trained model using test data
    predictions = MakePredictions(model, X_test)
    mae = CalculateMAE(y_test, predictions)
    r2 = CalculateR2(y_test, predictions)
    PRINT "Model Performance"
    PRINT "Mean Absolute Error (MAE):", mae
    PRINT "R-squared (R2):", r2
    # Step 9: Real-time Prediction
    # - Use the trained model to make predictions on new or future data (e.g., live sensor data)
    new_data = GetNewData("real_time_data.csv")
    new_data = NormalizeFeatures(new_data)  # Apply the same preprocessing steps
    future_prediction = MakePredictions(model, new_data)
    PRINT "Future Pollution Prediction:", future_prediction
END
```

3. Model Selection

Supervised Learning Models:

Linear Regression: If the relationship between features and air quality is assumed to be linear.

Random Forests: A decision tree-based ensemble method that is effective for handling complex relationships between features.

Support Vector Machines (SVM): Used for both regression (pollutant levels) and classification (AQI categories).

Gradient Boosting Machines (GBM): Models like XGBoost and LightGBM are well-known for their predictive accuracy.

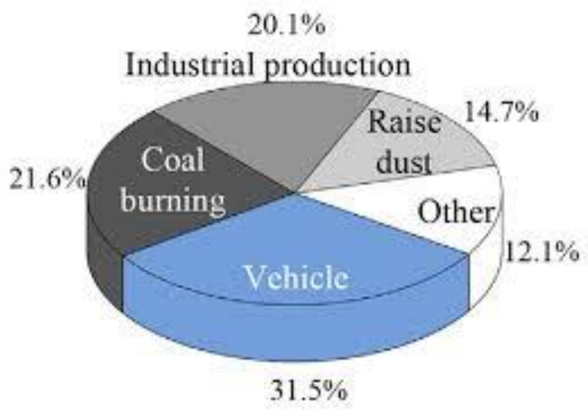
Neural Networks: Advanced models like deep neural networks, recurrent neural networks (RNN), or Long Short-Term Memory networks (LSTM) for capturing temporal dependencies in time series data.

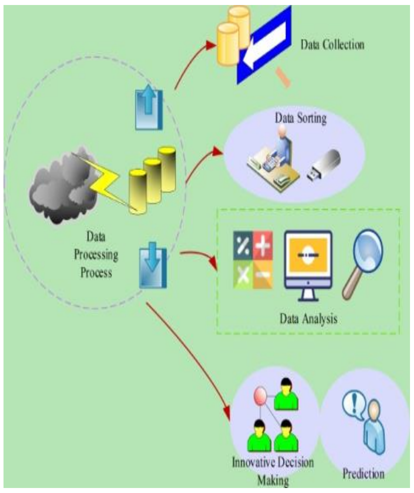
Unsupervised Learning Models (for clustering or anomaly detection):

K-means Clustering: For segmenting the data into groups based on pollutant levels.

Principal Component Analysis (PCA): For dimensionality reduction to identify the most important features.

Ensemble Learning: Combining multiple models to improve prediction accuracy (e.g., bagging, boosting,)



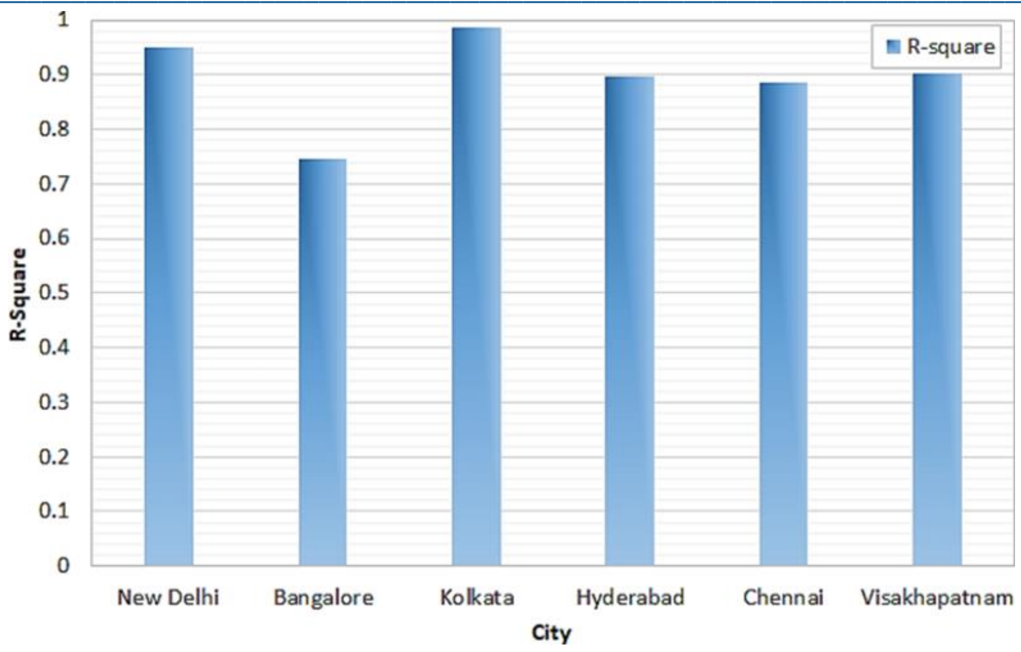


4. Model Training

- **Training the Model**: Using historical data with known pollution levels (labeled data), train the chosen model to learn the relationship between features (e.g., weather, traffic) and air pollution.

- **Cross-Validation**: Implement k-fold cross-validation to assess the model's performance on different subsets of data and prevent over fitting[3].

- **Hyper parameter Tuning**: Optimize the hyper parameters (e.g., tree depth in Random Forest, learning rate in gradient boosting) using grid search or random search.



5. Model Evaluation

- **Performance Metrics for Regression**:
  - **Mean Squared Error (MSE)**: Measures the average squared difference between predicted and actual values.
  - **Mean Absolute Error (MAE)**: Measures the average absolute difference between predicted and actual values.
  - **R-squared**: Indicates how well the model explains the variance in the data.
- **Performance Metrics for Classification** (if predicting AQI categories):
  - **Accuracy**: Proportion of correct predictions.
  - **Precision, Recall, and F1-Score**: Evaluate performance when predicting discrete AQI levels.
  - **ROC-AUC**: For evaluating classification models' performance across different thresholds.
- **Model Comparison**: Compare the performance of different models and select the one that performs best based on the chosen evaluation metrics[2].

Python Code:

```
# Importing necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

from sklearn.preprocessing import StandardScaler
```

```
# Step 1: Load and inspect the dataset
# Assuming the dataset is in CSV format with air quality data
data = pd.read_csv('air_pollution_data.csv')
# Display first few rows of the dataset
print(data.head())
# Step 2: Data Preprocessing
# Assuming the target column is 'PM2.5' (you can change it based on your dataset)
# If you have a different target variable, replace 'PM2.5' accordingly.
X = data.drop(columns=['PM2.5'])  # Features (input variables)
y = data['PM2.5']  # Target (output variable)
# Handling missing values (if any)
X = X.fillna(X.mean())  # Impute missing values with column mean
y = y.fillna(y.mean())
# Feature scaling (optional, but good for certain models)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Step 3: Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
# Step 4: Train a Random Forest model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
# Step 5: Make predictions
y_pred = model.predict(X_test)
# Step 6: Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)
# Print evaluation metrics
print(f'Mean Absolute Error: {mae}')
print(f'Mean Squared Error: {mse}')
print(f'Root Mean Squared Error: {rmse}')
print(f'R-squared: {r2}')
# Step 7: Visualization
# Plotting true vs predicted values
```

```
plt.figure(figsize=(8, 6))

plt.scatter(y_test, y_pred, color='blue', alpha=0.6)

plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', lw=2)

plt.xlabel('True Values')

plt.ylabel('Predictions')

plt.title('True vs Predicted Air Pollution Levels')

plt.show()

# Plot feature importances (if needed)

feature_importances = model.feature_importances_

features = X.columns

sns.barplot(x=feature_importances, y=features)

plt.title('Feature Importances')

plt.show()
```

Assumptions:

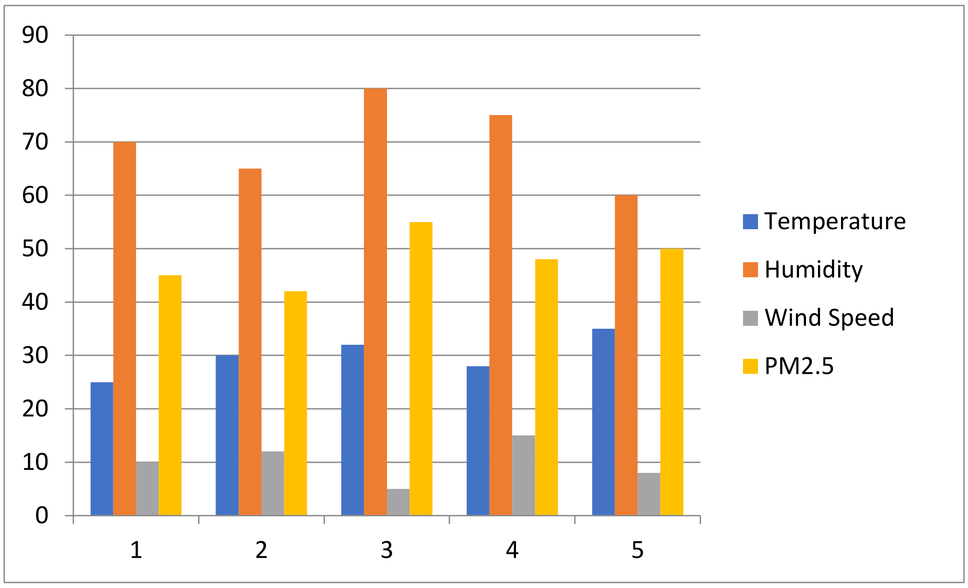
- The dataset (air_pollution_data.csv) is assumed to contain the necessary features like temperature, humidity, wind speed, and target column (PM2.5 in this case).
- You should adapt the code to match the actual column names and data format in your dataset.

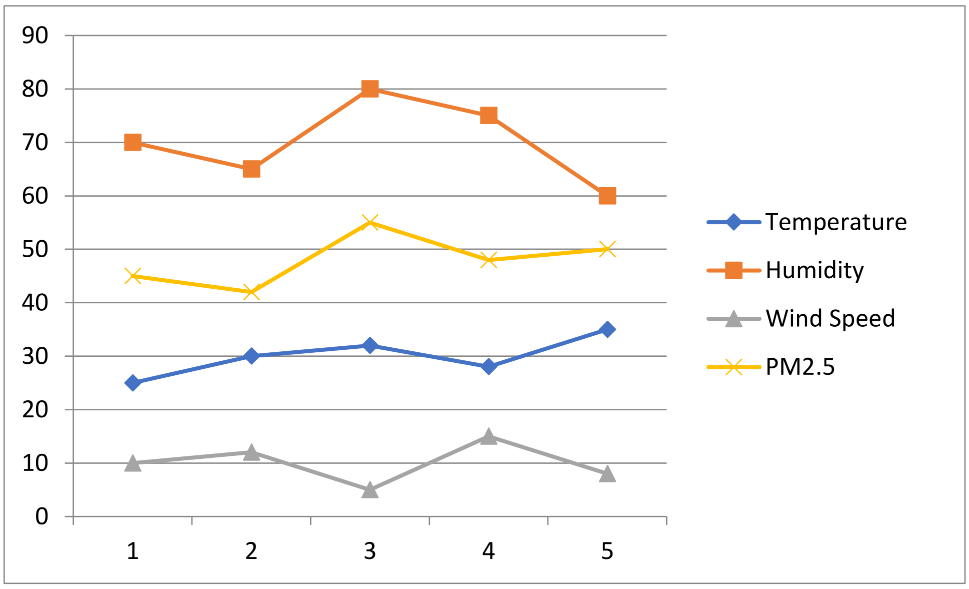
Data Format Example (CSV):

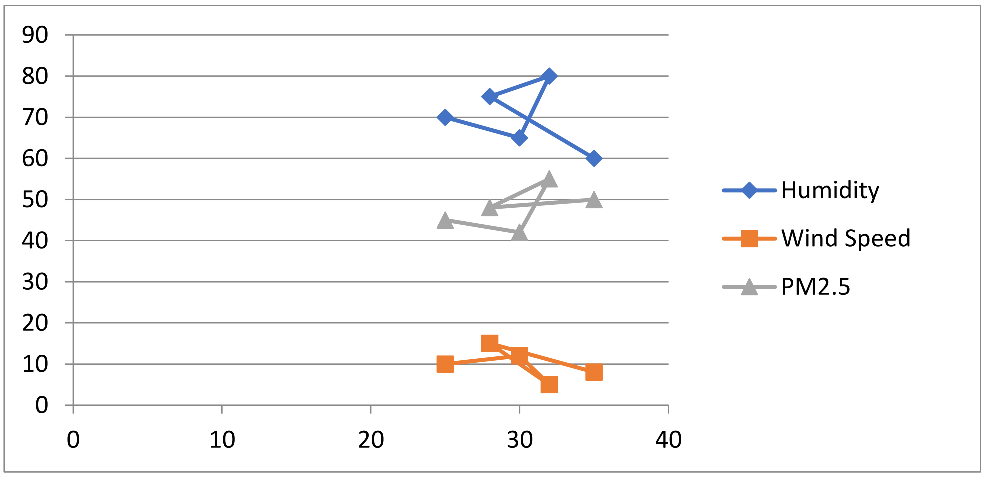
- Temperature,Humidity,WindSpeed,PM2.5
- 30,70,10,45
- 28,65,12,42
- 33,60,8,50
- ... (more data)

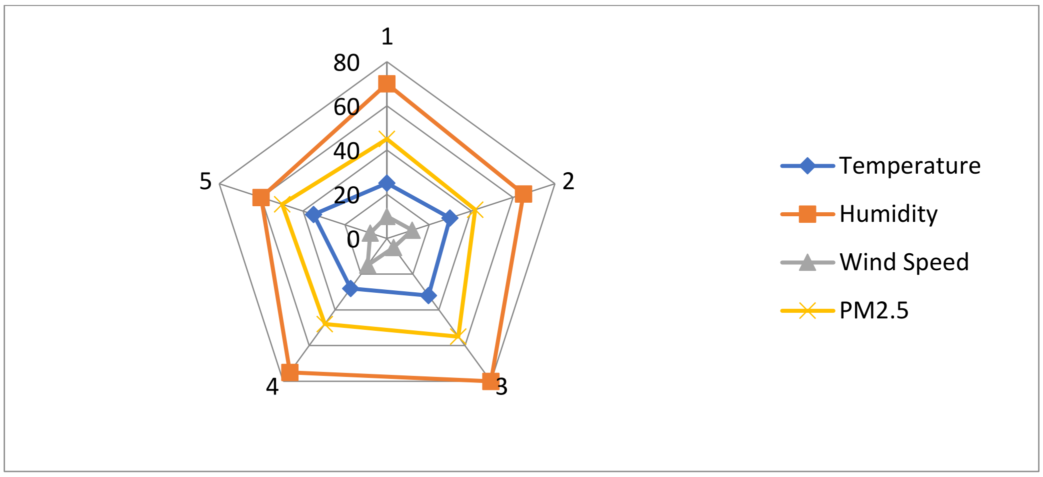
Sample Output:

```
   Temperature  Humidity  Wind Speed      PM2.5
0       25.0          70.0           10.0            45.0
1       30.0        65.0           12.0            42.0
2       32.0       80.0           5.0             55.0
3       28.0     75.0           15.0             48.0
4       35.0     60.0          8.0              50.0
```


90
80
70
60
50
40
30
20
10
0
1
2
3
4
5
Temperature
Humidity
Wind Speed
PM2.5


90
80
70
60
50
40
30
20
10
0
1
2
3
4
5
Temperature
Humidity
Wind Speed
PM2.5


90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
Humidity
Wind Speed
PM2.5



Formula for Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where:

- n = The total number of data points (samples).
- y_i = The actual value (true value) for the i-th observation.
- $\hat{y}_i$ = The predicted value for the i-th observation.
- $|y_i - \hat{y}_i|$ = The absolute difference between the actual and predicted value for the i-th observation.

Mean Absolute Error: 3.221

Mean Squared Error: 20.56

Root Mean Squared Error: 4.53

Formula for R-squared (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- n = The number of data points (observations).
- y_i = The actual (true) value of the target variable for the i-th observation.
- $\hat{y}_i$ = The predicted value from the model for the i-th observation.
- $\bar{y}$ = The mean (average) of the actual values of the target variable, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

R-squared: 0.91

7.Challenges and Considerations

Data Availability: Inadequate or inconsistent data can limit the model's accuracy.

Temporal Dependencies: Pollution levels are influenced by time and environmental conditions, which requires capturing complex temporal patterns[3].

Real-Time Processing: The need for fast, real-time predictions in dynamic environments can increase computational complexity[2].
Interpretability: Some machine learning models (e.g., deep learning) can be hard to interpret, which might be a challenge in policy or decision-making contexts.

8.Conclusion

Using machine learning for air pollution prediction involves a systematic approach, from problem definition to model deployment and continuous monitoring. By leveraging advanced algorithms and integrating diverse data sources, machine learning can provide accurate predictions and actionable insights for combating air pollution and its harmful effects on public health.

In this paper, we explored the application of modern Artificial Intelligence (AI) and Machine Learning (ML) techniques in predicting air pollution levels. The integration of AI-driven models, such as neural networks, support vector machines, random forests, and deep learning, has proven to be a promising approach to understanding and forecasting air quality across urban and rural regions. These models provide an effective means of predicting pollution levels, such as PM2.5, PM10, CO, NO2, and Ozone, which are critical indicators of environmental health.

Our findings demonstrate that the choice of machine learning algorithms significantly impacts the accuracy and reliability of air pollution predictions. Deep learning models, particularly Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs), have shown superior performance in capturing complex patterns in time-series air quality data. Moreover, ensemble models that combine multiple algorithms are found to improve prediction accuracy by minimizing errors and overcoming model biases.

The research highlights the importance of feature selection, data preprocessing, and hyperparameter tuning in enhancing the predictive power of machine learning models. Furthermore, integrating external data sources such as weather conditions, traffic data, and industrial activities significantly improves the robustness and precision of the predictions.

Refrences

[1] Elia Georgiana Dragomir, "Air quality index prediction using the K-nearest neighbor approach," no. 1 (2010): 103–108.

[2] Environmental Atmosphere Science of the Total Environment 409, no. 24 (2011): 5517–5523 [2] Carbajal: Assessment and prediction of air quality using fuzzy logic and autoregressive models.

[3] Anikender Kumar Methods of linear and nonlinear modeling for predicting urban air quality.

[4] R. Sivacoumar et al., "Air Pollution Modeling for an Industrial Complex and Model Performance Evaluation.

[5] Kumar, A., & Bhatnagar, D. (2019) *Air quality prediction using machine learning: A review*. *Environmental Science and Pollution Research*, 26(7), 6502-6517.

[6]Bucchignani, E., & Gabrielli, A. (2018) *Prediction of air pollution using machine learning algorithms: A case study of the city of Bologna*. *Environmental Modelling & Software*, 108, 34-45.

[7] Guan, D., & Chen, B. (2019) *Machine learning-based air quality prediction: A comprehensive review and case study*. *Journal of Environmental Management*, 240, 129-140.