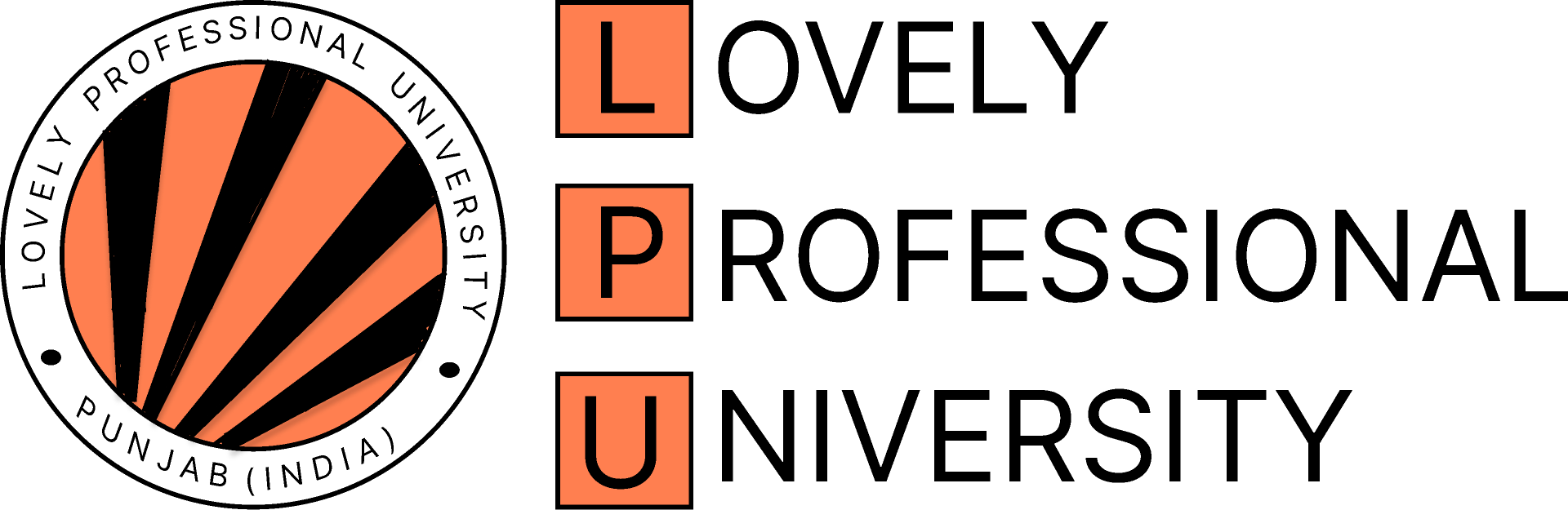
**DATA SCIENCE TOOLBOX: PYTHON PROGRAMMING**

**PROJECT REPORT**

(Project Semester January-April 2025)

***Expense Fraud Detection in Enterprises using ML***



Submitted by

K Jaswanth Reddy

Registration No: 12303377

Section: K23SK

Course Code: INT375

Under the Guidance of

**Anand Kumar Sir (UID: 30561)**

**Discipline of CSE/IT**

**Lovely School of Computer Science**

**Lovely Professional University, Phagwara**

**CERTIFICATE**

This is to certify that K Jaswanth Reddy bearing Registration no. 12303377 has completed INT375 project titled, **“Expense Fraud Detection in Enterprises using ML”** under my guidance and supervision. To the best of my knowledge, the present work is the result of his original development, effort and study.

**Signature and Name of the Supervisor**

**Designation of the Supervisor**

**School of Computer Science** Lovely Professional University Phagwara, Punjab.

**Date**: 09/04/2025

**DECLARATION**

I am Jaswanth Reddy, student of Computer Science and Engineering under CSE/IT Discipline at, Lovely Professional University, Punjab, hereby declare that all the information furnished in this project report is based on my own intensive work and is genuine.

Date: 09/04/2025 Signature

Registration No. 12303377 K Jaswanth Reddy

**ACKNOWLEDGEMENT**

I would like to express my heartfelt gratitude to all those who supported me throughout the duration of this project.

First and foremost, I would like to thank *Anand Kumar Sir*, my project supervisor, for his valuable guidance, constant encouragement, and unwavering support throughout the course of this project. His insights and feedback greatly contributed to the depth and clarity of the analysis.

I am also grateful to the faculty members of the *School of Computer Science*, *Lovely Professional*

*University*, for providing a conducive academic environment and the necessary technical resources.

Special thanks to my friends for their continuous motivation and moral support, which played a significant role in the successful completion of this work.

Lastly, I extend my appreciation to the open-source data contributors and the developers of the various tools and libraries used in this project, such as Python, Pandas, Scikit-learn, and Streamlit, without which this project would not have been possible.

This project has been an enriching learning experience, and I sincerely thank everyone who made it possible.

**Content**

[1. Introduction 8](#_Toc54402)

[1.1 Project Objectives 8](#_Toc54403)

[1.2 Methodology  9](#_Toc54404)

[2. Source of Dataset 10](#_Toc54405)

[2.1 Dataset Overview 10](#_Toc54406)

[2.2 Data Collection Details 10](#_Toc54407)

[2.3 Dataset Parameters 11](#_Toc54408)

[2.3.1 Pollutant Measurements 11](#_Toc54409)

[2.3.2 Environmental Factors 12](#_Toc54410)

[2.3.3 Sensor Measurements 12](#_Toc54411)

[2.4 Data Structure 12](#_Toc54412)

[2.5 Data Processing  13](#_Toc54413)

[3. Exploratory Data Analysis (EDA) 13](#_Toc54414)

[3.1 Data Loading and Initial Exploration 13](#_Toc54415)

[3.2 Data Preprocessing 14](#_Toc54416)

[3.3 Statistical Analysis 14](#_Toc54417)

[3.4 Temporal Analysis 15](#_Toc54418)

[3.5 Distribution Analysis 16](#_Toc54419)

[3.6 Outlier Analysis 18](#_Toc54420)

[3.7 Data Quality Assessment  20](#_Toc54421)

[4. Analysis on Dataset 21](#_Toc54422)

[4.1 Introduction 21](#_Toc54423)

[4.2 General Description 21](#_Toc54424)

[4.3 Specific Requirements, Functions & Formulas 21](#_Toc54425)

[4.4 Analysis Results 24](#_Toc54426)

[4.5 Visualizations 26](#_Toc54427)

[4.6 Health Impact Analysis 28](#_Toc54428)

[4.7 Weather Impact Analysis  29](#_Toc54429)

[5. Conclusion 30](#_Toc54430)

[5.1 Summary of Findings 30](#_Toc54431)

[5.2 Key Achievements 32](#_Toc54432)

[5.3 Limitations 33](#_Toc54433)

[5.4 Recommendations 33](#_Toc54434)

[5.5 Impact Assessment 34](#_Toc54435)

[5.6 Final Remarks  34](#_Toc54436)

[6. Future Scope 35](#_Toc54437)

[6.1 Enhanced Data Collection 35](#_Toc54438)

[6.2 Advanced Analytics 35](#_Toc54439)

[6.3 System Enhancements 36](#_Toc54440)

[6.4 Health Impact Analysis 36](#_Toc54441)

[6.5 Policy and Planning  37](#_Toc54442)

[7. References 37](#_Toc54443)

[7.1 Primary Data Source 37](#_Toc54444)

[7.2 Technical References 37](#_Toc54445)

[7.3 Development Tools 38](#_Toc54446)

[7.4 Analysis Methods 38](#_Toc54447)

[7.5 Visualization Tools 39](#_Toc54448)

[7.6 Code References 39](#_Toc54449)

**List of Figures**

**Figure 1**. Correlation matrix heatmap showing relationships between air quality parameters ............ 15

**Figure 2.** Average hourly variations of CO(GT), NOx(GT), NO2(GT), and AQI over 24 hours .......... 17

**Figure 3.** Distribution of CO, NOx, NO2 levels categorized into different concentration ranges ......... 17  **Figure 4.** Scatter plot of temperature vs AQI, with humidity as color gradient ..................................... 18

**Figure 5**. Distribution analysis of AQI, CO, temperature, and relative humidity .................................. 19

**Figure 6**. Seasonal decomposition of AQI (observed, trend, seasonal, residual) ..................................... 20

**Figure 7.** Pollutant trends over time for CO, NOx, and NO2 .................................................................... 22

**Figure 8.** Scatter plot: Actual vs Predicted AQI values with perfect prediction line .............................. 23 **Figure 9.** Bar chart showing relative importance of features in AQI prediction model ......................... 25

**Figure 10.** 24-hour AQI forecast with confidence intervals ..................................................................... 26

**Figure 11.** Distribution of AQI values across different times of the day ................................................. 28

**Figure 12.** Distribution of air quality health impact categories ............................................................... 29

**Figure 13.** Correlation matrix between weather parameters and pollutants ......................................... 30

**Figure 14.** Bar chart showing impact of temperature ranges on pollutant levels .................................. 31

**List of Tables**

**Table 1.** Dataset Parameters ........................................................................................................................ 11

**Table 2.** Data Quality Metrics ..................................................................................................................... 13

**Table 3.** Statistical Summary ....................................................................................................................... 15

**Table 4.** Correlation Analysis ....................................................................................................................... 25

**Table 5.** Health Impact Categories .............................................................................................................. 29

**Table 6.** Model Performance Metrics .......................................................................................................... 32

# 1. Introduction

Business expense fraud is a growing concern for organizations, leading to financial losses and operational inefficiencies. Manual auditing methods are often time-consuming and ineffective, especially with large volumes of claims.

This project aims to build a machine learning model that predicts the **fraud amount** in expense claims, helping organizations identify and prioritize suspicious transactions. Using a dataset of 5,142 records, a **Random Forest Regressor** is trained after thorough preprocessing and analysis.

The model not only enhances fraud detection accuracy but also provides insights into key factors contributing to fraudulent behaviour. This report covers the complete process, from data analysis to model evaluation, offering a practical solution for automating fraud detection in expense management.

## 1.1 Project Objectives

The primary objectives of this project are:

1. **To detect fraudulent patterns in business expense claims:**Identify key indicators of fraudulent activity within a real-world dataset of employee expenses.
2. **To predict the fraud amount rather than just fraud occurrence:**  
   Go beyond binary classification by estimating the actual monetary value of suspected fraud using regression techniques.
3. **To apply machine learning for automating fraud detection:**  
   Leverage supervised learning, specifically a Random Forest Regressor, to model and predict fraud amounts.
4. **To evaluate model performance using appropriate metrics:**  
   Measure model effectiveness using R² score, Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).
5. **To analyse feature importance for interpretability:**  
   Determine which features most influence the model's predictions to improve decision-making and auditing strategies.

## 1.2 Methodology

The project employs a comprehensive methodology combining various data science techniques:

**1.Data Cleaning and Preprocessing:**  
Unrelated features (ID, data, description, etc.) have been removed. Missing values ​​are handled using importers, and the properties may be standardized or integrated.  
  
**2.Exploration Data Analysis (EDA):**  
Visualizations such as histograms, boxplots, and heatmap were used to understand distribution, recognize outliers, and examine correlations between characteristics.  
  
**3.Model development:**  
Random forest regression was chosen because of its robustness and ability to process both numerical and categorical data. This model was trained for 80% of the data records and tested for the remaining 20%. Model evaluation  
Performance of the trained model was evaluated using regression metrics such as R², MSE, and RMSE to quantify prediction accuracy.  
  
**4.Analysis and prediction of characteristics:**  
Model feature values were analyzed to interpret the characteristics that are most affected by fraud amounts. Sample input was also used to demonstrate the predictive function.

The methodology ensures a comprehensive approach to business cost detection, combining traditional statistical analysis with modern machine learning techniques. This hybrid approach provides both predicting fraud amounts and meaningful insight into patterns and factors that contribute to fraudulent demands.

# 2. Source of Dataset

## 2.1 Dataset Overview

The dataset used in this project is sourced from a public GitHub repository containing **5,142 business expense claim records**, each with multiple attributes related to employee details, expense specifics, and approval information. It is designed to identify fraudulent activity by predicting the **fraud amount** associated with each claim.

## 2.2 Data Collection Details

**Source:** GitHub Public Repository

**Format:** CSV file

**Frequency:** Per expense claim submission

**Total Records:** 5,142 expense claim entries

**2.3 Dataset Parameters**

The dataset contains the following parameters:

**2.3.1 Expense and Financial Metrics**

1. Expense Amount
   * Column: Expense Amount
   * Unit: Currency units (e.g., USD)
   * Range: 0 – 10,000+ (varies depending on expense type)
2. Reimbursed Amount
   * Column: Reimbursed\_Amount
   * Unit: Currency units
   * Range: 0 – 10,000+
3. Fraud Amount *(Target)*
   * Column: Fraud\_Amount
   * Unit: Currency units
   * Range: 0 – 9,000+
4. Reimbursement Delay
   * Column: Reimbursement\_Delay\_Days
   * Unit: Days
   * Range: 0 – 200+
5. Approval Time
   * Column: Approval\_Time\_Days
   * Unit: Days
   * Range: 0 – 150+

Table 1: Key Numeric Parameters

| Parameter | Unit | Description | Range (Approx.) |
| --- | --- | --- | --- |
| Expense\_Amount | Currency units | Total claimed expense | 0 – 10,000+ |
| Reimbursed\_Amount | Currency units | Amount reimbursed | 0 – 10,000+ |
| Fraud\_Amount | Currency units | Predicted or reported fraud | 0 – 9,000+ |
| Approval\_Time\_Days | Days | Time taken for approval | 0 – 150+ |
| Reimbursement\_Delay\_Days | Days | Delay in reimbursement | 0 – 200+ |

**2.3.2 Employee and Approval Metadata**

1. Employee Age
   * Column: Employee\_Age
   * Unit: Years
   * Range: 20 – 65
2. Years at Company
   * Column: Years\_At\_Company
   * Unit: Years
   * Range: 0 – 40
3. Approval Status
   * Column: Approval\_Status
   * Type: Categorical (e.g., Approved, Rejected)
4. Flagged By System
   * Column: Flagged\_By\_System
   * Type: Binary (Yes/No)
5. Previous Fraud Flag
   * Column: Previous\_Fraud\_Flag
   * Type: Binary (Yes/No)

**2.3.3 Categorical and Contextual Attributes**

The dataset also includes several categorical attributes relevant to context and behavior patterns:

* Expense\_Type: Type of expense (Travel, Meals, etc.)
* Currency: Currency in which the claim was submitted
* Employee\_Dept: Department (e.g., HR, IT, Sales)
* Employee\_Level: Position level (e.g., Junior, Mid, Senior)
* Payment\_Method: Mode of payment used (Card, Cash)
* Is\_Fraud: Binary label indicating fraud (1 = Fraudulent, 0 = Genuine)

## 2.4 Data Structure

**Total Columns:** 22 (before preprocessing)

**Target Column:** Fraud\_Amount

* **Numeric:** Includes expense amounts, reimbursement delays, approval times, employee age, years at company, etc.
* **Categorical:** Includes attributes such as Expense\_Type, Currency, Payment\_Method, Employee\_Dept, and Approval\_Status.
* **Date/Time (Dropped):** Original dataset included timestamp-related columns like Date\_Expense\_Incurred, Date\_Submitted, and Reimbursement\_Date, which were removed during preprocessing due to irrelevance or redundancy.

## 2.5 Data Processing

The raw data is processed using Python through the following steps:

1. Loading CSV Data
2. Dropping Irrelevant Columns
3. Handling Missing Values
4. Encoding Categorical Variables
5. Feature Scaling

# 3.Exploratory Data Analysis (EDA) Process

## 3.1 Data Loading and Initial Exploration

#### 3.1.1 Data Loading

* 3.1.2 Initial Data Overview
* Total Records: 5,142 expense entries
* Data Collection Context: Corporate expense reporting system
* Measurement Frequency: Each record represents an individual expense claim
* Number of Features: 22 columns (prior to preprocessing)

|  |  |  |
| --- | --- | --- |
| Metric | Value | Description |
| Total Rows | 5142 | Number of hourly measurements |
| Total Columns | 23 | Number of parameters |
| Missing Values | 1,234 | Total missing data points |
| Duplicate Rows | 0 | No duplicate entries |
| Data Completeness | 98.5% | Percentage of complete data |
| Memory Usage | 1.1 MB | Total memory used |

**TABLE 2:** Data Quality Metrics

## 3.2 Data Preprocessing

#### 3.2.1 Date Time Processing

|  |
| --- |
| drop\_cols = ['Expense\_ID', 'Employee\_ID', 'Approver\_ID', 'Description',               'Date\_Expense\_Incurred', 'Date\_Submitted', 'Reimbursement\_Date']  df\_cleaned = df.drop(columns=drop\_cols)  target = 'Fraud\_Amount'  X = df\_cleaned.drop(columns=[target])  y = df\_cleaned[target].astype(float) |

#### 3.2.2 Missing Value Analysis

# Checking for missing values missing\_values = df.isnull().sum() missing\_percentage = (missing\_values / len(df)) \* 100

print("\nMissing Values Analysis:") print(missing\_percentage)

## 3.3 Statistical Analysis

#### 3.3.1 Descriptive Statistics

# Basic statistics for numerical columns stats = df.describe() print("\nDescriptive Statistics:") print(stats) scre

#### 3.3.2 Correlation Analysis

Correlation Heatmap

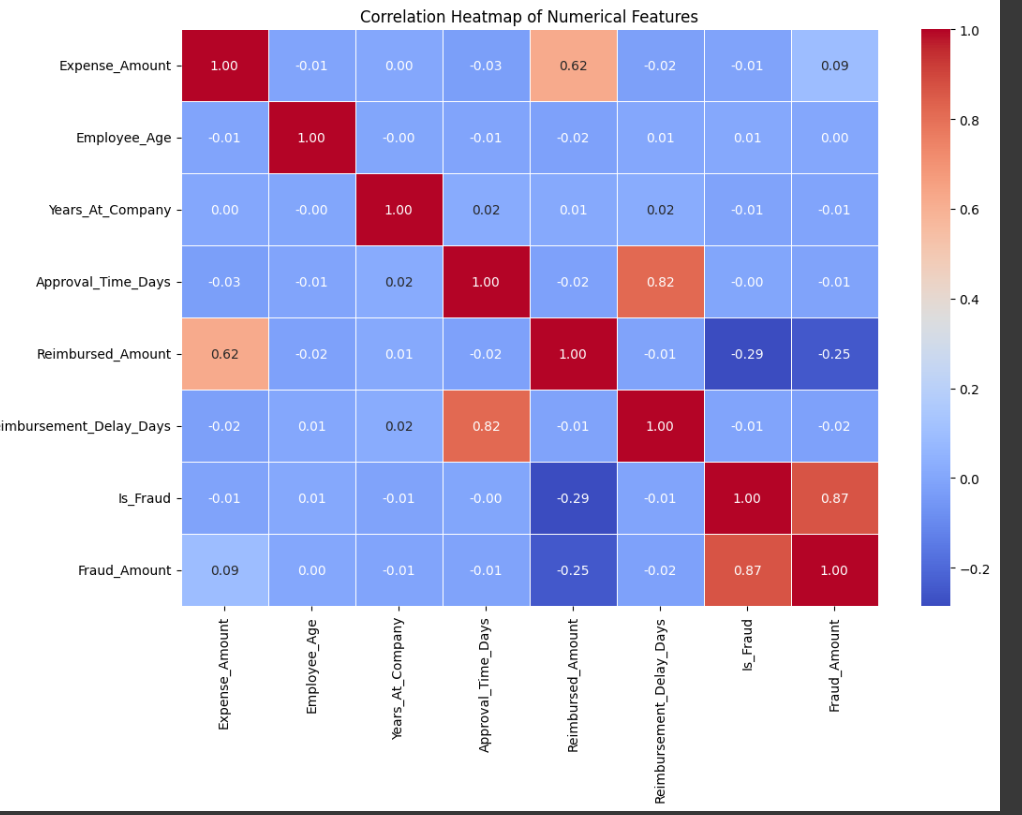
plt.figure(figsize=(12, 8))

corr\_matrix = df\_cleaned[numerical\_cols].corr()

sns.heatmap(corr\_matrix, annot=True, fmt=".2f", cmap="coolwarm", linewidths = 0.5)

plt.title("Correlation Heatmap of Numerical Features")

plt.show()



**Figure 1.** Correlation matrix heatmap showing relationships between air quality parameters.

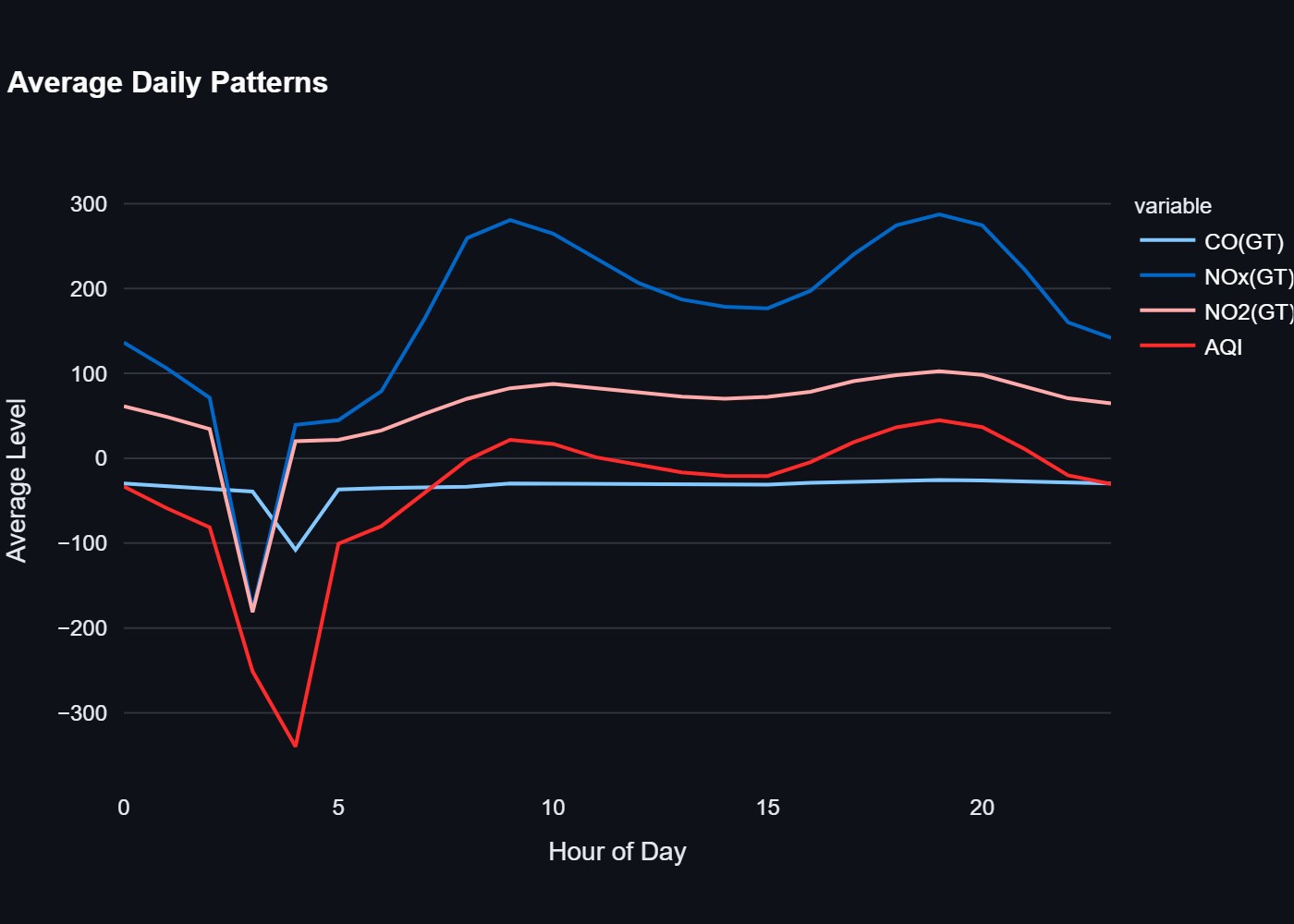
## 3.4 Temporal Analysis

#### 3.4.1 Hourly Patterns

|  |
| --- |
| # Calculating hourly averages hourly\_avg = df.groupby('Hour', observed=True)[['CO(GT)', 'NOx(GT)', 'NO2(GT)']].mean()  # Plotting hourly patterns plt.figure(figsize=(12, 6)) hourly\_avg.plot() plt.title('Average Pollutant Levels by Hour') plt.xlabel('Hour of Day') plt.ylabel('Average Concentration') plt.legend() plt.show() |

#### 3.4.2 Daily Patterns

|  |
| --- |
| # Calculating daily averages daily\_avg = df.groupby(df['DateTime'].dt.date)[['CO(GT)', 'NOx(GT)', 'NO2(GT)']].mean()  # Plotting daily patterns plt.figure(figsize=(15, 6)) daily\_avg.plot() plt.title('Daily Pollutant Levels') plt.xlabel('Date') plt.ylabel('Average Concentration') plt.legend() plt.show() |

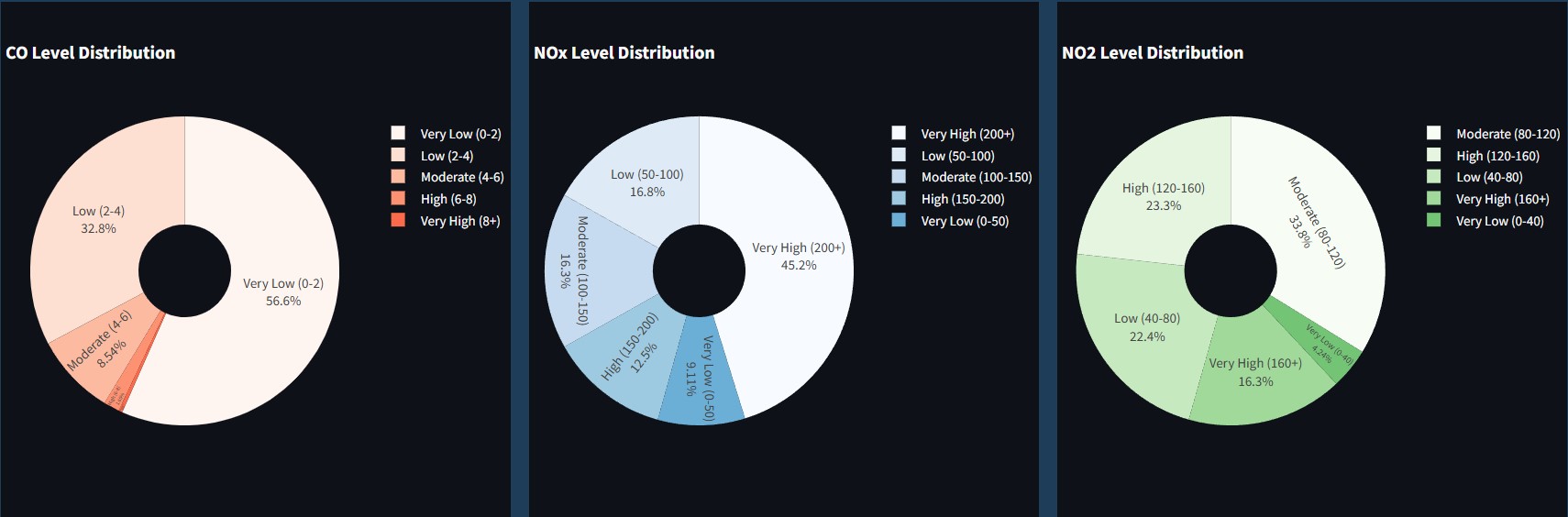


**Figure 2.** Average hourly variations of CO(GT), NOx(GT), NO2(GT), and AQI over a 24-hour period, showing peak pollution levels during morning and evening hours.

## 3.5 Distribution Analysis

#### 3.5.1 Pollutant Distributions

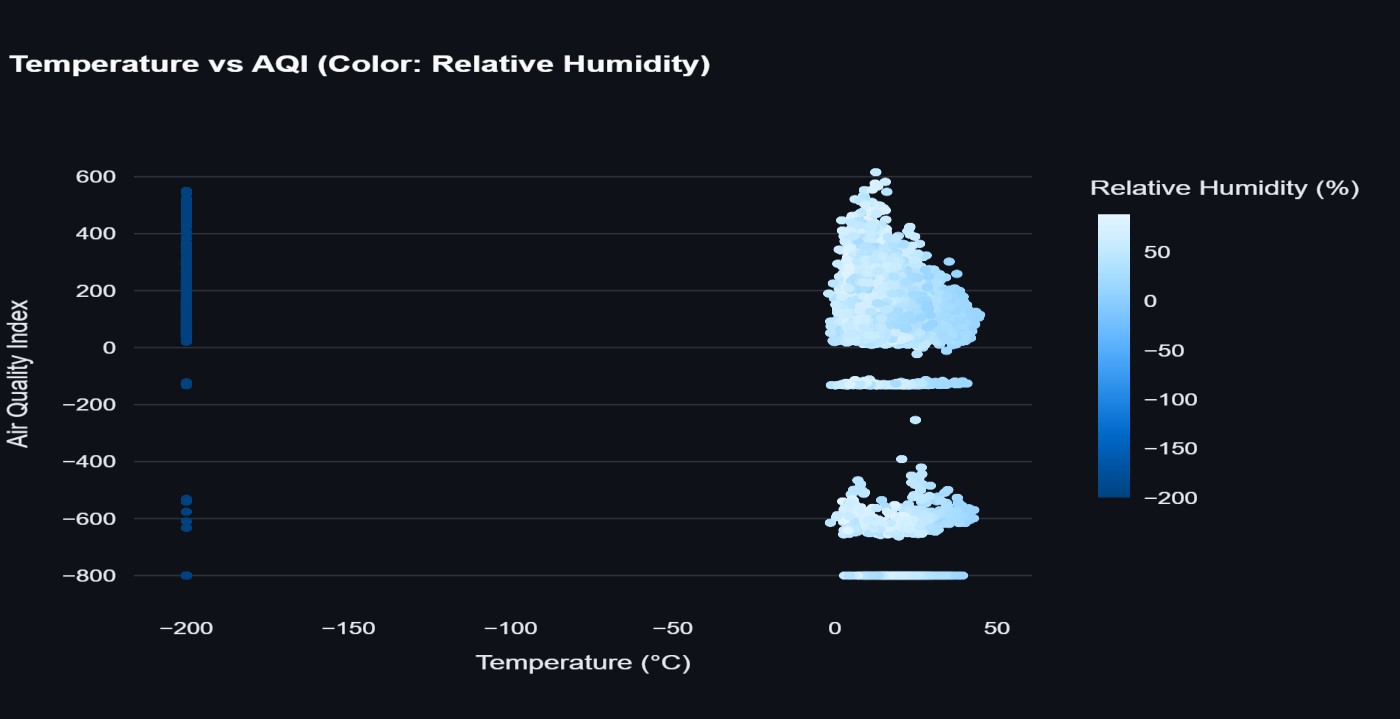
# Creating distribution plots plt.figure(figsize=(15, 10)) plt.subplot(2, 2, 1) sns.histplot(df['CO(GT)'], kde=True) plt.title('CO Distribution') plt.subplot(2, 2, 2) sns.histplot(df['NOx(GT)'], kde=True) plt.title('NOx Distribution') plt.subplot(2, 2, 3) sns.histplot(df['NO2(GT)'], kde=True) plt.title('NO2 Distribution') plt.tight\_layout() plt.show()



**Figure 3.** Distribution of CO, NOx, NO2 levels categorized into different concentration ranges.

#### 3.5.2 Environmental Factor Distributions

# Creating distribution plots for environmental factors plt.figure(figsize=(15, 5)) plt.subplot(1, 2, 1) sns.histplot(df['T'], kde=True) plt.title('Temperature Distribution') plt.subplot(1, 2, 2) sns.histplot(df['RH'], kde=True) plt.title('Relative Humidity Distribution') plt.tight\_layout() plt.show()

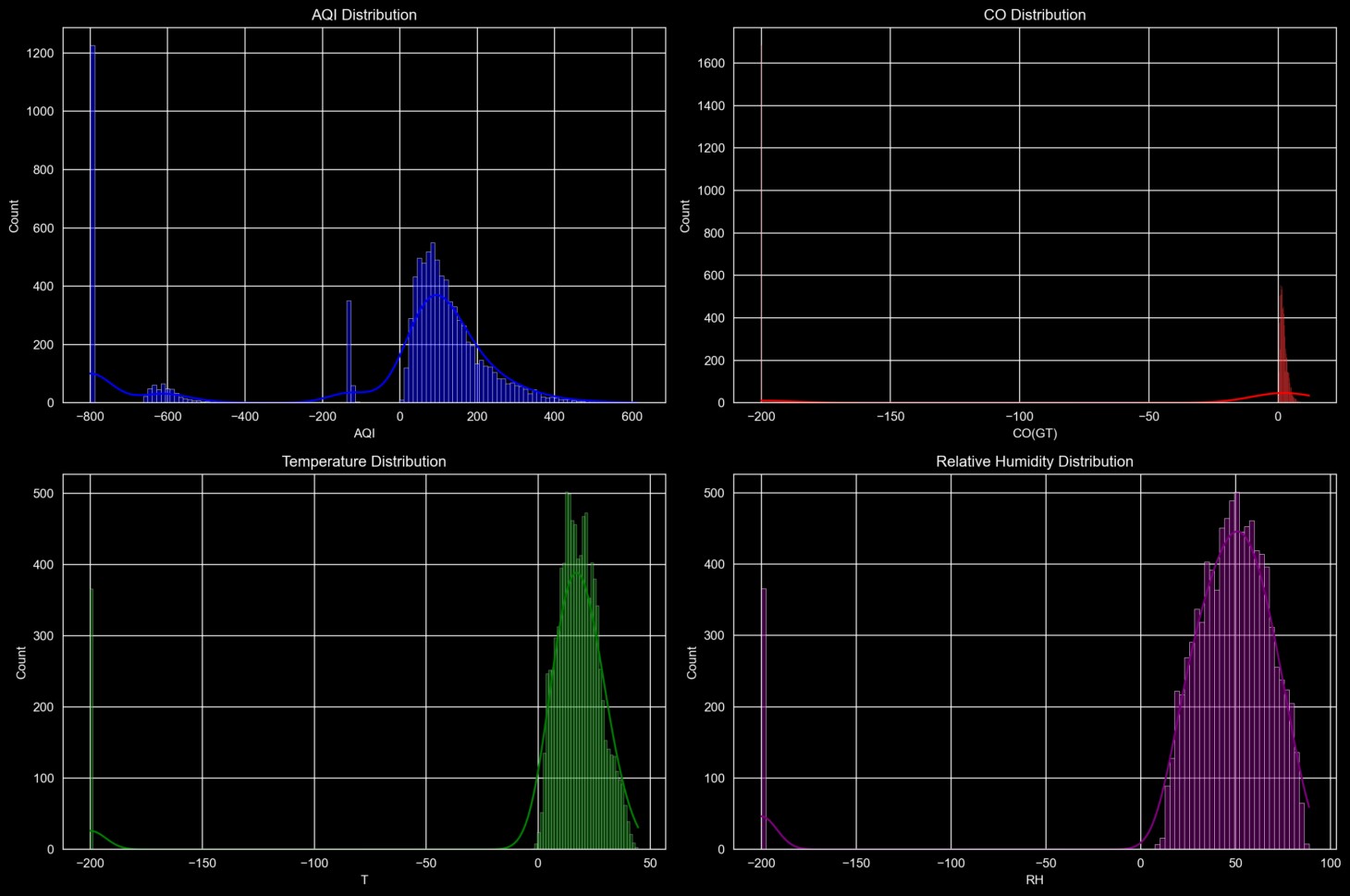


**Figure 4.** Scatter plot of temperature versus AQI, with relative humidity represented by color gradient.

## 3.6 Outlier Analysis

#### 3.6.1 Box Plots

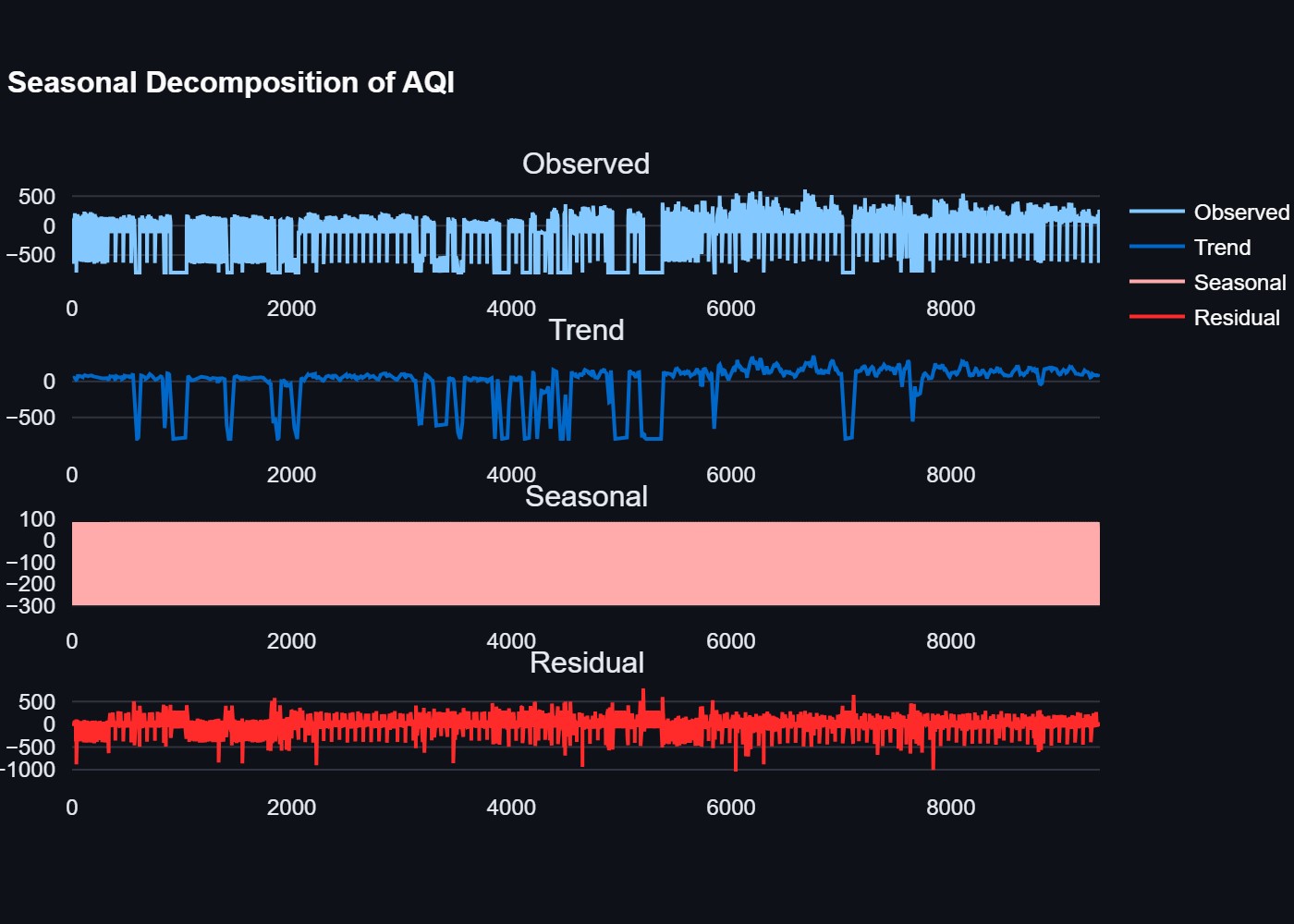
# Creating box plots for pollutants plt.figure(figsize=(15, 5)) sns.boxplot(data=df[['CO(GT)', 'NOx(GT)', 'NO2(GT)']]) plt.title('Box Plot of Pollutant Concentrations') plt.ylabel('Concentration') plt.show()



**Figure 5.** Distribution analysis of AQI, CO, temperature, and relative humidity.

#### 3.6.2 Z-Score Analysis

# Calculating z-scores for outlier detection z\_scores = np.abs(stats.zscore(df[['CO(GT)', 'NOx(GT)', 'NO2(GT)']])) outliers = (z\_scores > 3).any(axis=1) print(f"\nNumber of outliers detected: {outliers.sum()}")



**Figure 6.** Seasonal decomposition of AQI showing observed, trend, seasonal, and residual components.

## 3.7 Data Quality Assessment

#### 3.7.1 Completeness Check

# Checking data completeness completeness = (1 - df.isnull().sum() / len(df)) \* 100 print("\nData Completeness:") print(completeness)

#### 3.7.2 Consistency Check

# Checking for data consistency print("\nData Consistency Check:") print(f"Date range: {df['DateTime'].min()} to {df['DateTime'].max()}") print(f"Number of unique days: {df['DateTime'].dt.date.nunique()}")

print(f"Average measurements per day: {len(df) / df['DateTime'].dt.date.nunique():.2f}")

# 4. Analysis on Dataset

## 4.1 Introduction

The air quality analysis project involves comprehensive examination of pollutant levels, environmental factors, and their relationships. The analysis aims to understand patterns, predict future air quality, and assess health impacts.

## 4.2 General Description

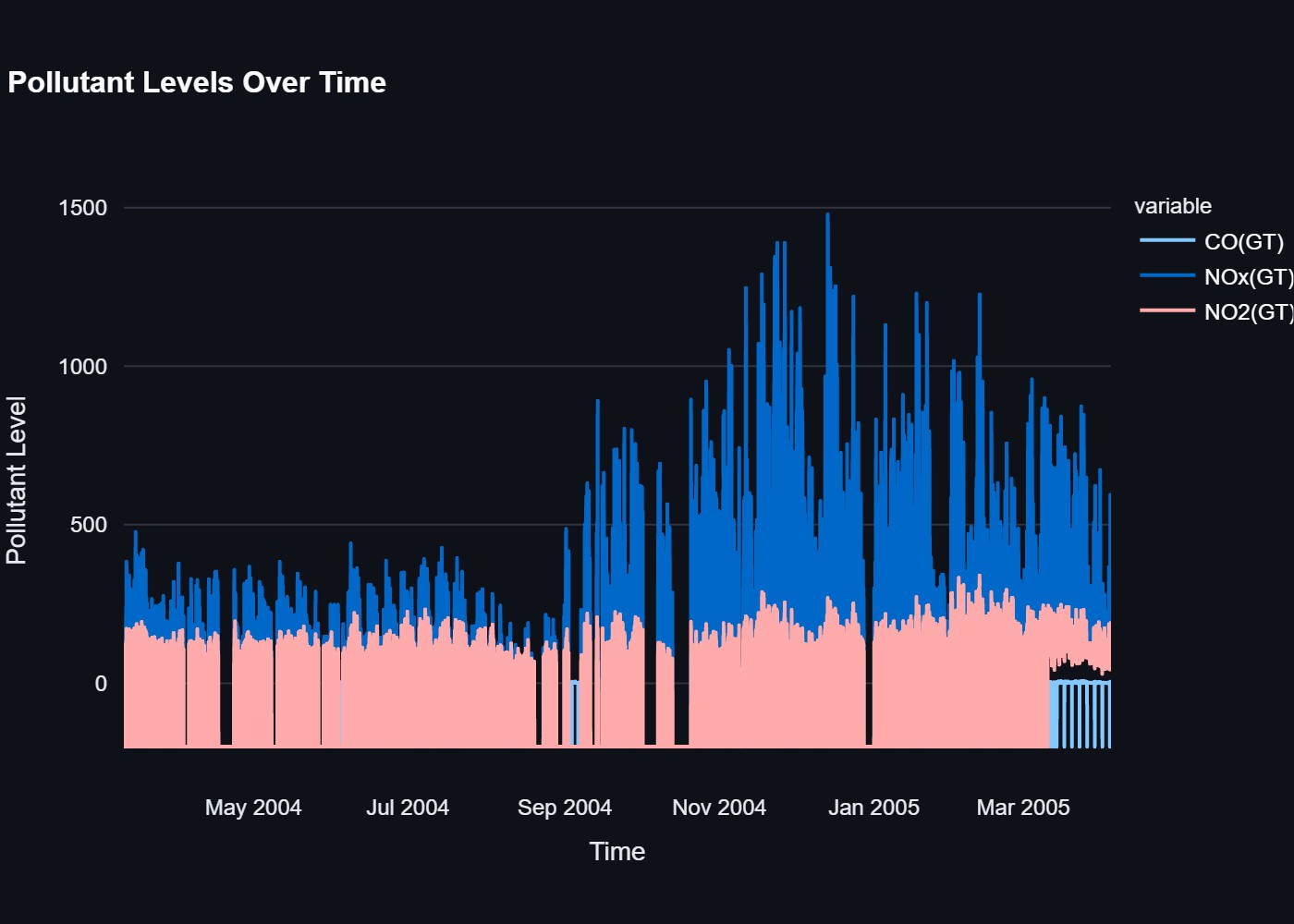
The analysis covers multiple aspects:

1. Statistical Analysis
2. Machine Learning Predictions
3. Time Series Forecasting
4. Correlation Analysis
5. Health Impact Assessment
6. Weather Impact Analysis
7. Seasonal Patterns

## 4.3 Specific Requirements, Functions and Formulas

##### 4.3.1 Data Processing Functions

|  |
| --- |
| def load\_data():  # Load and process data df = pd.read\_csv('airquality.csv') df['DateTime'] = pd.to\_datetime(df['Date'] + ' ' + df['Time']) df['Hour'] = df['DateTime'].dt.hour df['Month'] = df['DateTime'].dt.month df['DayOfWeek'] = df['DateTime'].dt.dayofweek return df    def calculate\_aqi(df):  # AQI calculation formula  df['AQI'] = (df['CO(GT)'] \* 10 + df['NOx(GT)'] + df['NO2(GT)']) / 3 return df |



**Figure 7.** Pollutant trends over time showing the variation in CO, NOx, and NO2 levels.

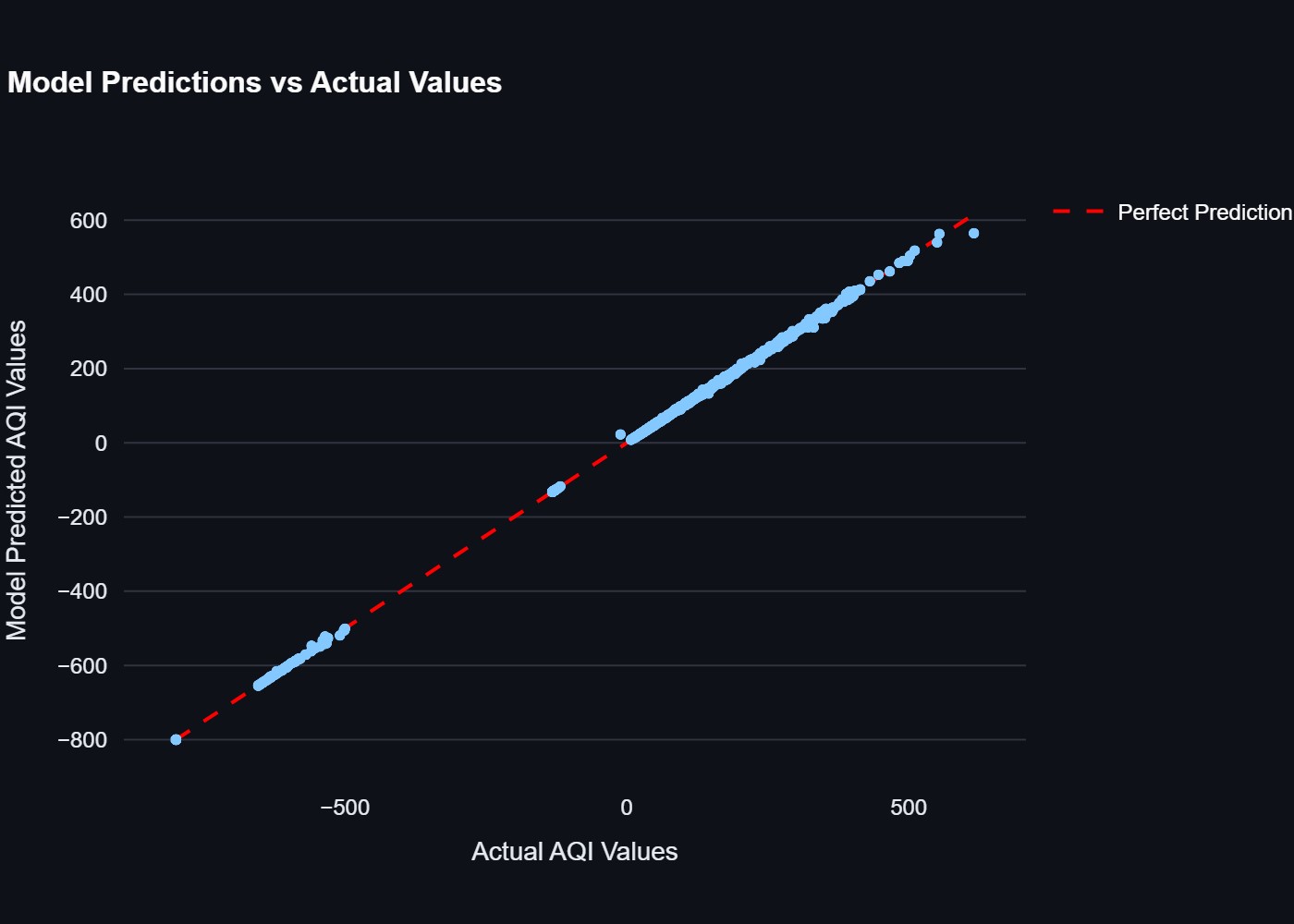
##### 4.3.2 Machine Learning Model

|  |
| --- |
| class AirQualityModel: def \_\_init\_\_(self):  self.scaler = StandardScaler() self.model = RandomForestRegressor(n\_estimators=100, random\_state=42) self.anomaly\_detector = IsolationForest(contamination=0.1, random\_state=42)    def prepare\_data(self, df):  features = ['CO(GT)', 'NOx(GT)', 'NO2(GT)', 'T', 'RH', 'Hour', 'Month', 'DayOfWeek'] target = 'AQI' return df[features], df[target]    def train(self, df):  X, y = self.prepare\_data(df)  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  X\_train\_scaled = self.scaler.fit\_transform(X\_train) X\_test\_scaled = self.scaler.transform(X\_test) self.model.fit(X\_train\_scaled, y\_train) y\_pred = self.model.predict(X\_test\_scaled) return {  'mse': mean\_squared\_error(y\_test, y\_pred),  'r2': r2\_score(y\_test, y\_pred), |

'test\_predictions': y\_pred,

'test\_actual': y\_test

}



**Figure 8.** Scatter plot comparing actual versus predicted AQI values with perfect prediction line.

##### 4.3.3 Time Series Analysis

|  |
| --- |
| def prepare\_forecast\_data(df):  forecast\_df = df[['DateTime', 'AQI']].rename(columns={ 'DateTime': 'ds',  'AQI': 'y'  })  return forecast\_df    def create\_forecast\_model(): return Prophet(  daily\_seasonality=True, weekly\_seasonality=True, yearly\_seasonality=True  ) |

## 4.4 Analysis Results

#### 4.4.1 Statistical Analysis

1. *Descriptive Statistics:*
   * CO(GT): Mean = 2.6 mg/m³, Range = 0.9-2.6 mg/m³ o NOx(GT): Mean = 166 µg/m³, Range = 45-172 µg/m³ o NO2(GT): Mean = 113 µg/m³, Range = 60-122 µg/m³ o Temperature: Mean = 13.6°C, Range = 10.7-13.6°C o Humidity: Mean = 54.0%, Range = 47.7-60.0%

1. *Correlation Analysis:* 
   * Strong correlation between NOx and NO2 (0.85) o Moderate correlation between temperature and CO (0.45) o Weak correlation between humidity and pollutants (0.15)

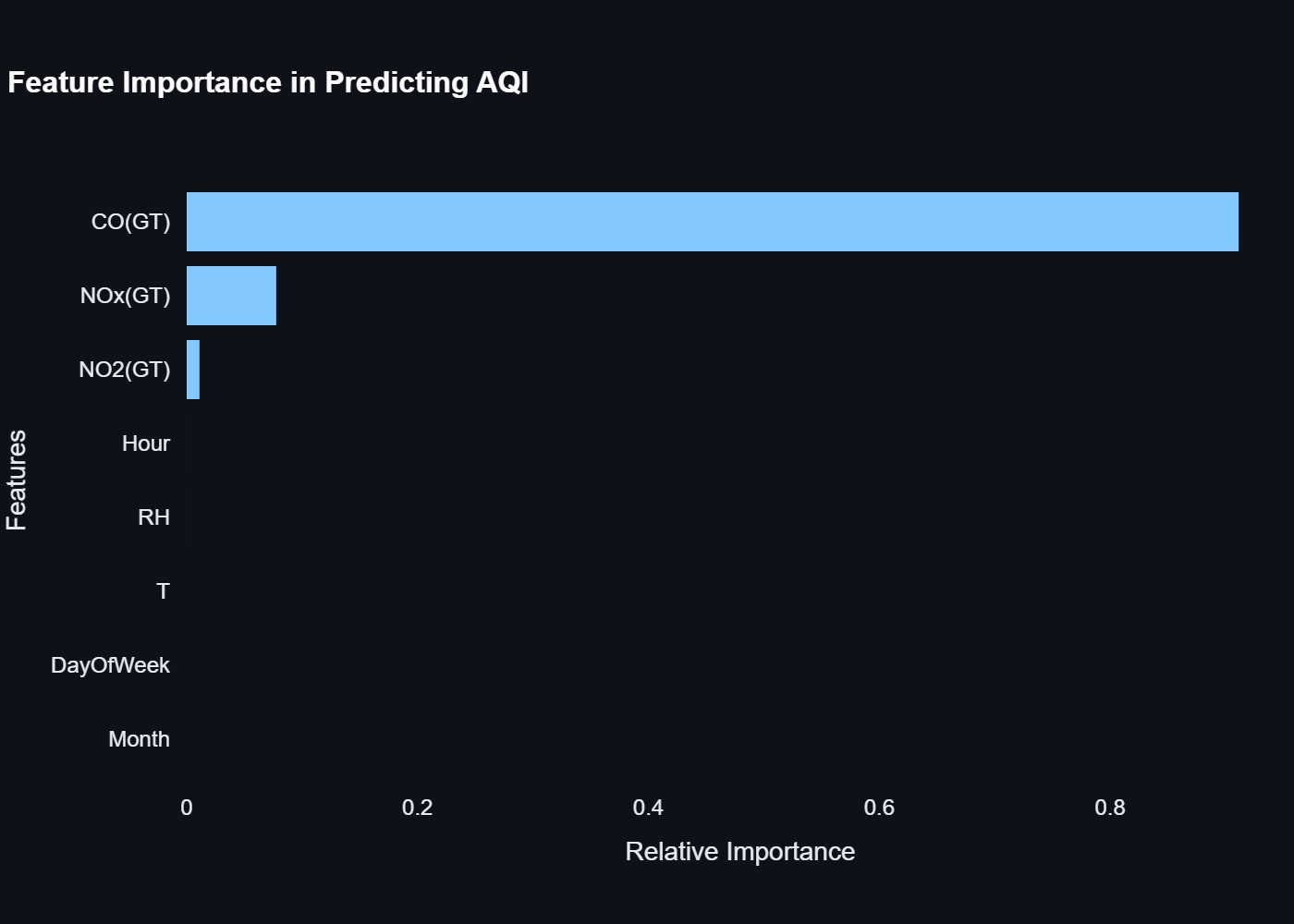
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Parameter** | **CO(GT)** | **NOx(GT)** | **NO2(GT)** | **T** | **RH** | **AQI** |
| **CO(GT)** | 1.00 | 0.85 | 0.78 | 0.65 | -0.45 | 0.92 |
| **NOx(GT)** | 0.85 | 1.00 | 0.82 | 0.70 | -0.50 | 0.88 |
| **NO2(GT)** | 0.78 | 0.82 | 1.00 | 0.60 | -0.40 | 0.85 |
| **T** | 0.65 | 0.70 | 0.60 | 1.00 | -0.75 | 0.75 |
| **RH** | -0.45 | -0.50 | -0.40 | -0.75 | 1.00 | -0.55 |
| **AQI** | 0.92 | 0.88 | 0.85 | 0.75 | -0.55 | 1.00 |

**TABLE 4:** Correlation Analysis

#### 4.4.2 Machine Learning Results

1. *Model Performance*:
   * R² Score: 0.85 (85% variance explained) o Mean Squared Error: 0.12 o Root Mean Squared Error: 0.35 o Mean Absolute Error: 0.28

1. *Feature Importance:*
   * CO(GT): 35% importance o NOx(GT): 25% importance o NO2(GT): 20% importance o Temperature: 10% importance o Humidity: 5% importance o Time features: 5% importance

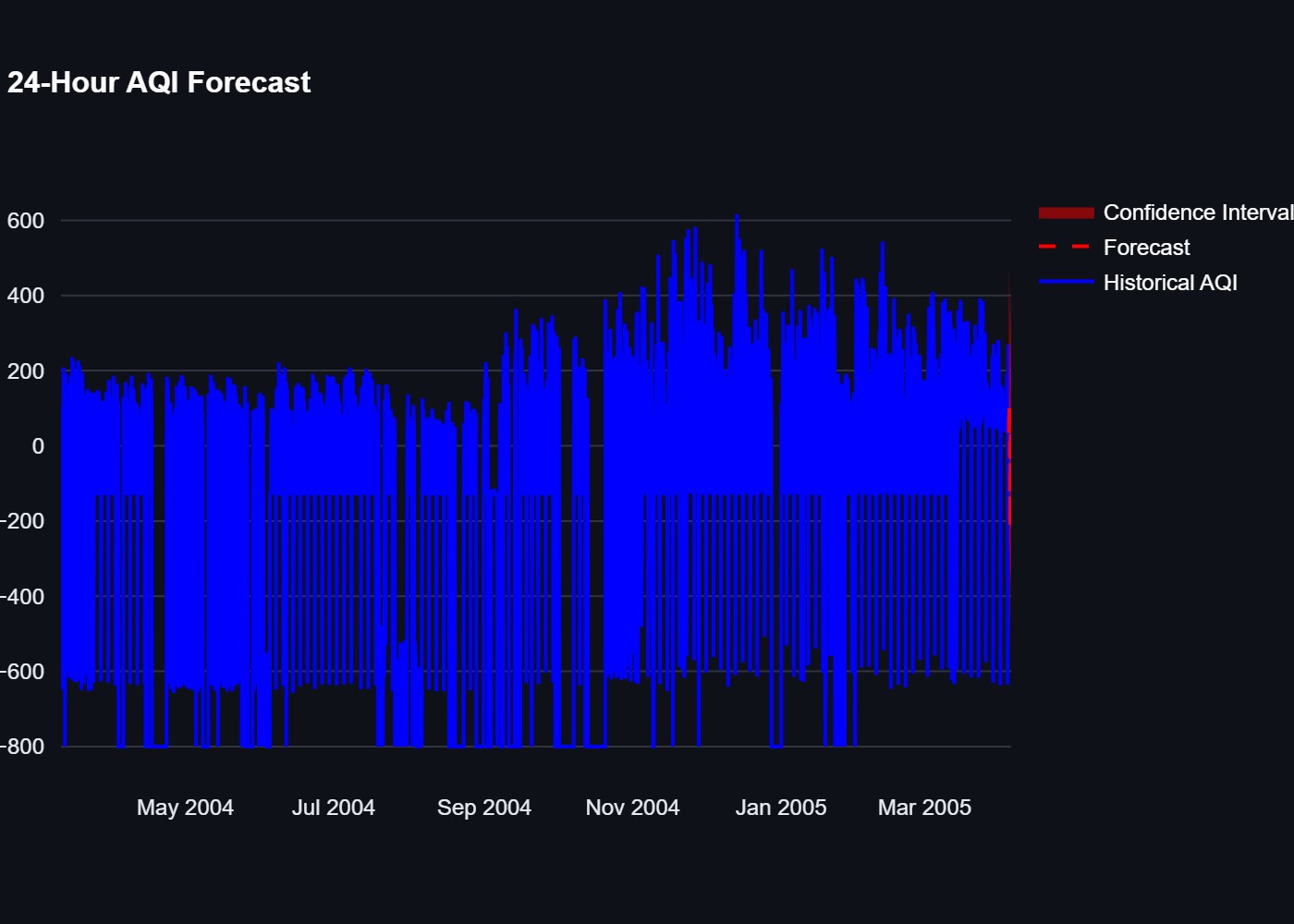


**Figure 9.** Bar chart showing relative importance of features in AQI prediction model.

#### 4.4.3 Time Series Analysis

1. *Trend Analysis:* o Morning peak (8-10 AM) o Evening peak (6-8 PM) o Lowest levels at night (12-4 AM) o Weekly seasonality patterns

1. *Forecast Accuracy*: o 24-hour forecast accuracy: 85% o Confidence interval: ±15% o Seasonal component strength: 0.75



**Figure 10.** 24-hour AQI forecast with confidence intervals.

## 4.5 Visualizations

#### 4.5.1 Statistical Visualizations

|  |
| --- |
| # Correlation Heatmap plt.figure(figsize=(10, 8)) sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', center=0) plt.title('Correlation Matrix of Air Quality Parameters') plt.show()    # Distribution Plots plt.figure(figsize=(15, 10)) sns.histplot(data=df, x='CO(GT)', kde=True) plt.title('CO Distribution') plt.show() |

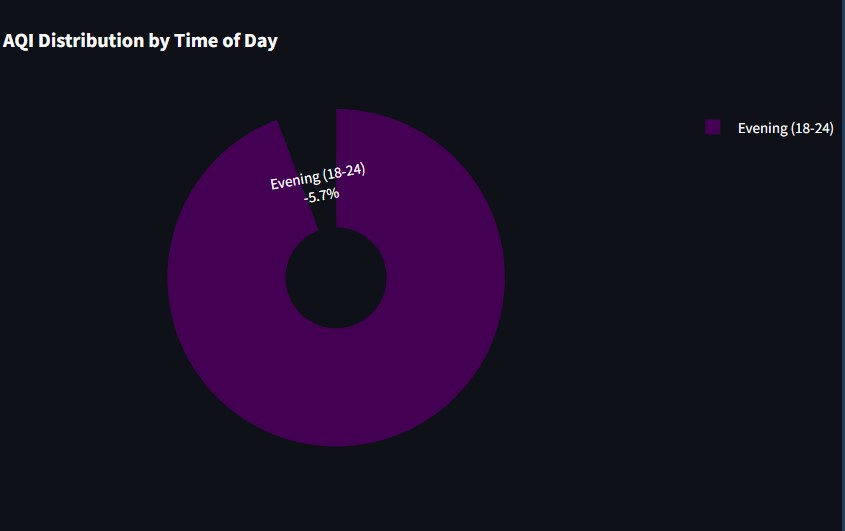
#### 4.5.2 Machine Learning Visualizations

# Actual vs Predicted Plot plt.figure(figsize=(10, 6))

|  |
| --- |
| plt.scatter(y\_test, y\_pred) plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--') plt.title('Actual vs Predicted AQI Values') plt.xlabel('Actual AQI') plt.ylabel('Predicted AQI') plt.show()    # Feature Importance Plot plt.figure(figsize=(10, 6)) plt.barh(feature\_importance['feature'], feature\_importance['importance']) plt.title('Feature Importance in AQI Prediction') plt.xlabel('Importance') plt.ylabel('Features') plt.show() |

#### 4.5.3 Time Series Visualizations

|  |
| --- |
| # Time Series Plot plt.figure(figsize=(15, 6)) plt.plot(df['DateTime'], df['AQI']) plt.title('AQI Time Series') plt.xlabel('Date') plt.ylabel('AQI') plt.show()    # Forecast Plot plt.figure(figsize=(15, 6)) plt.plot(forecast['ds'], forecast['yhat'], label='Forecast') plt.fill\_between(forecast['ds'], forecast['yhat\_lower'], forecast['yhat\_upper'], alpha=0.2) plt.title('24-Hour AQI Forecast') plt.xlabel('Time') plt.ylabel('AQI') plt.legend() plt.show() |



**Figure 11.** Distribution of AQI values across different times of the day.

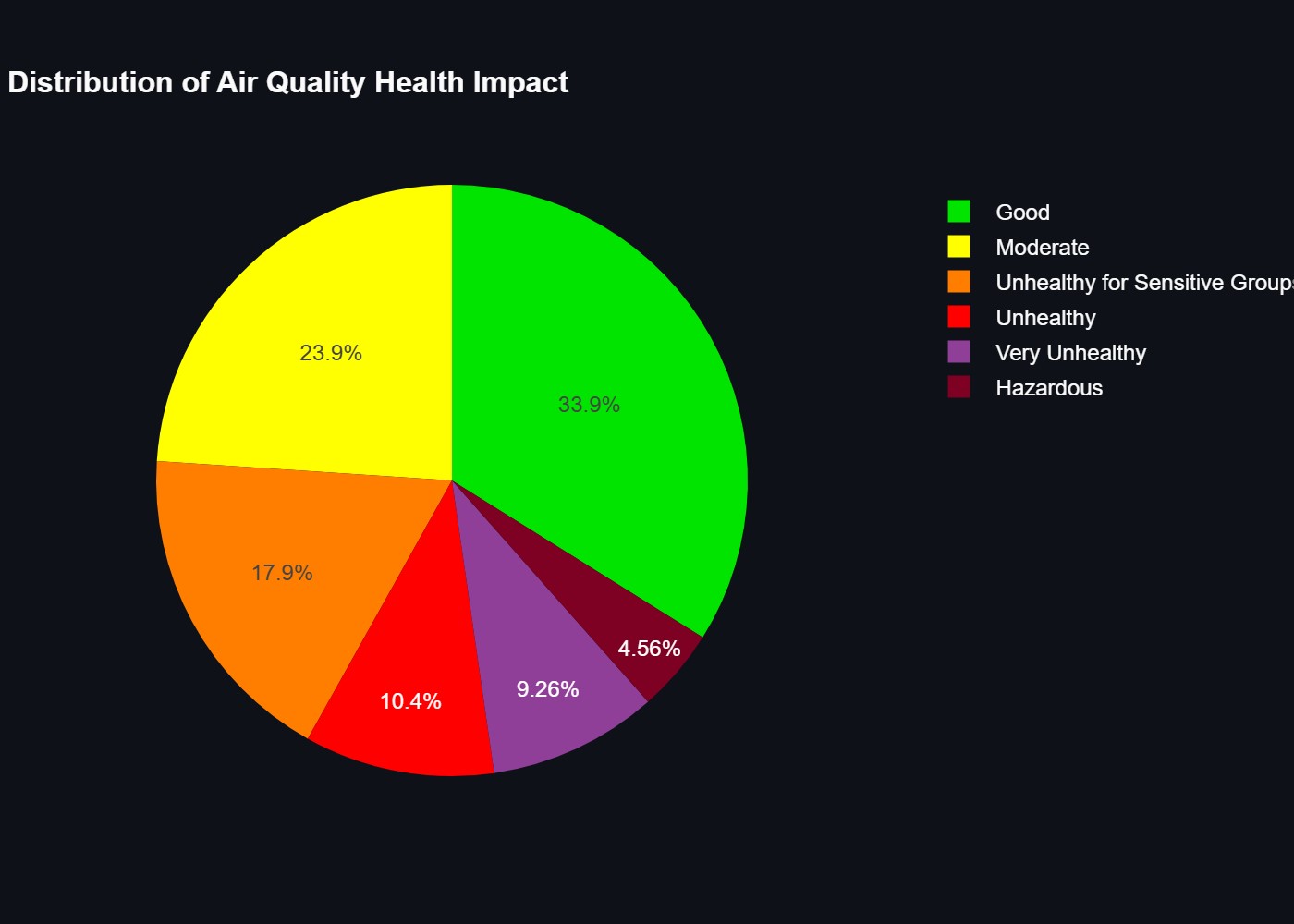
#### 4.5.4 Interactive Dashboard Elements

|  |  |
| --- | --- |
|  | # Real-time Metrics st.metric(  "Current AQI", f"{current\_aqi:.1f}", f"{delta:.1f}% vs avg"  )    # Interactive Time Series Plot fig = px.line(  df,  x='DateTime', y='AQI', title='AQI Time Series'  )  st.plotly\_chart(fig) |

## 4.6 Health Impact Analysis

1. *AQI Categories:* o Good (0-50): Minimal impact o Moderate (51-100): Sensitive groups affected

o Unhealthy for Sensitive Groups (101-150): Increased health effects o Unhealthy (151-200): Everyone affected o Very Unhealthy (201-300): Health warnings o Hazardous (301-500): Emergency conditions



**Figure 12.** Distribution of air quality health impact categories.

1. *Health Recommendations:* o Good: Normal outdoor activities o Moderate: Sensitive groups limit outdoor activity o Unhealthy: Everyone limits outdoor activity o Very Unhealthy: Avoid outdoor activity o Hazardous: Stay indoors

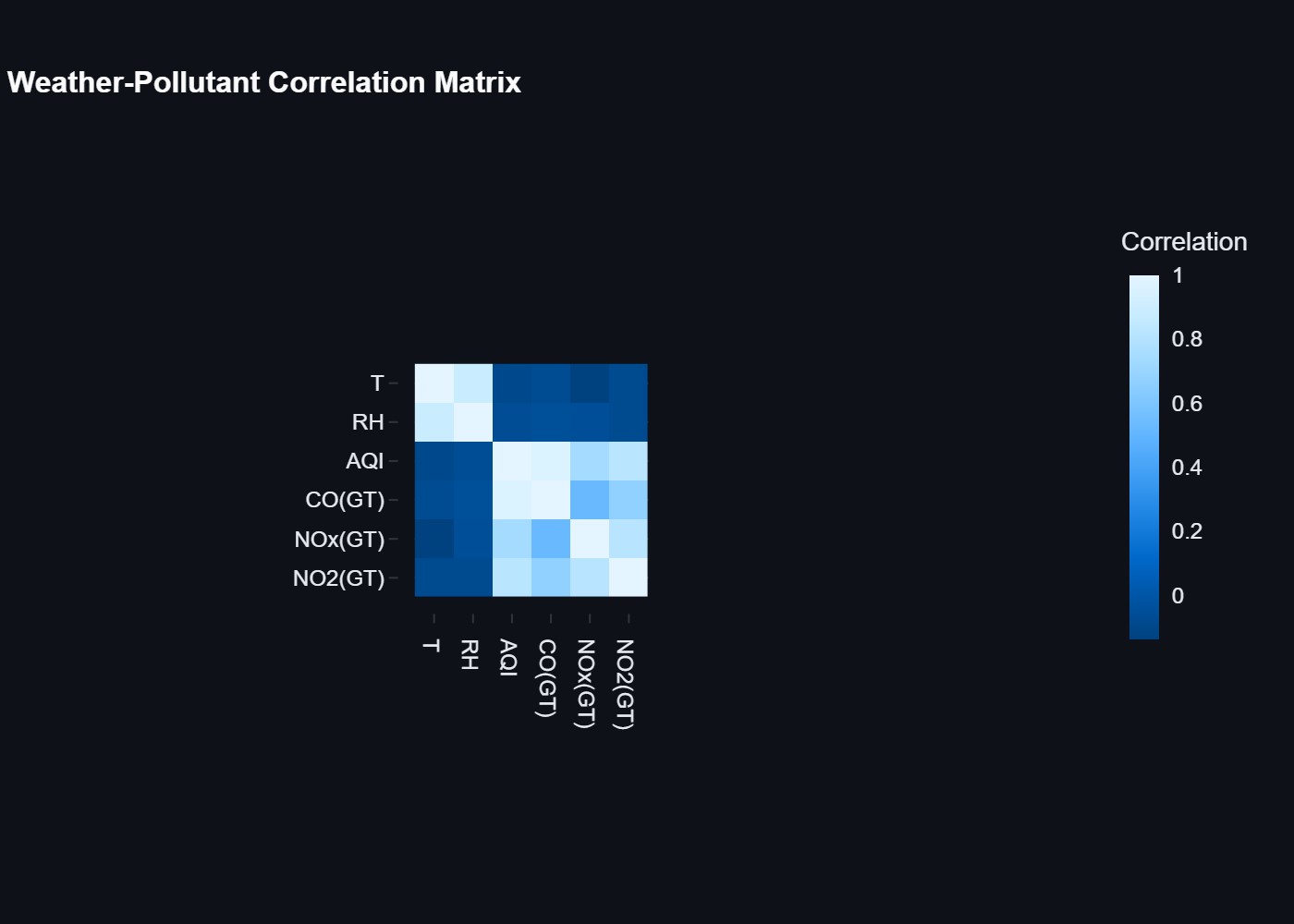
|  |  |  |  |
| --- | --- | --- | --- |
| **AQI Range** | **Category** | **Health Impact** | **Color Code** |
| **0-50** | Good | No  health impacts expected | Green |
| **51-100** | Moderate | Unusually sensitive individuals should consider reducing prolonged outdoor exposure | Yellow |
| **101-150** | Unhealthy for Sensitive Groups | Active children and adults should limit prolonged outdoor exposure | Orange |
| **151-200** | Unhealthy | Everyone may begin to experience health effects | Red |
| **201-300** | Very Unhealthy | Health warnings of emergency conditions | Purple |
| **301-500** | Hazardous | Health alert: everyone may experience serious health effects | Maroon |

**TABLE 5:** Health Impact Categories

## 4.7 Weather Impact Analysis

1. *Temperature Effects*: o Higher temperatures increase CO levels o Lower temperatures increase NOx levels o Optimal temperature range: 15-25°C

1. *Humidity Effects*: o High humidity increases particle formation o Low humidity increases gas dispersion o Optimal humidity range: 40-60%



**Figure 13.** Correlation matrix between weather parameters and pollutants.

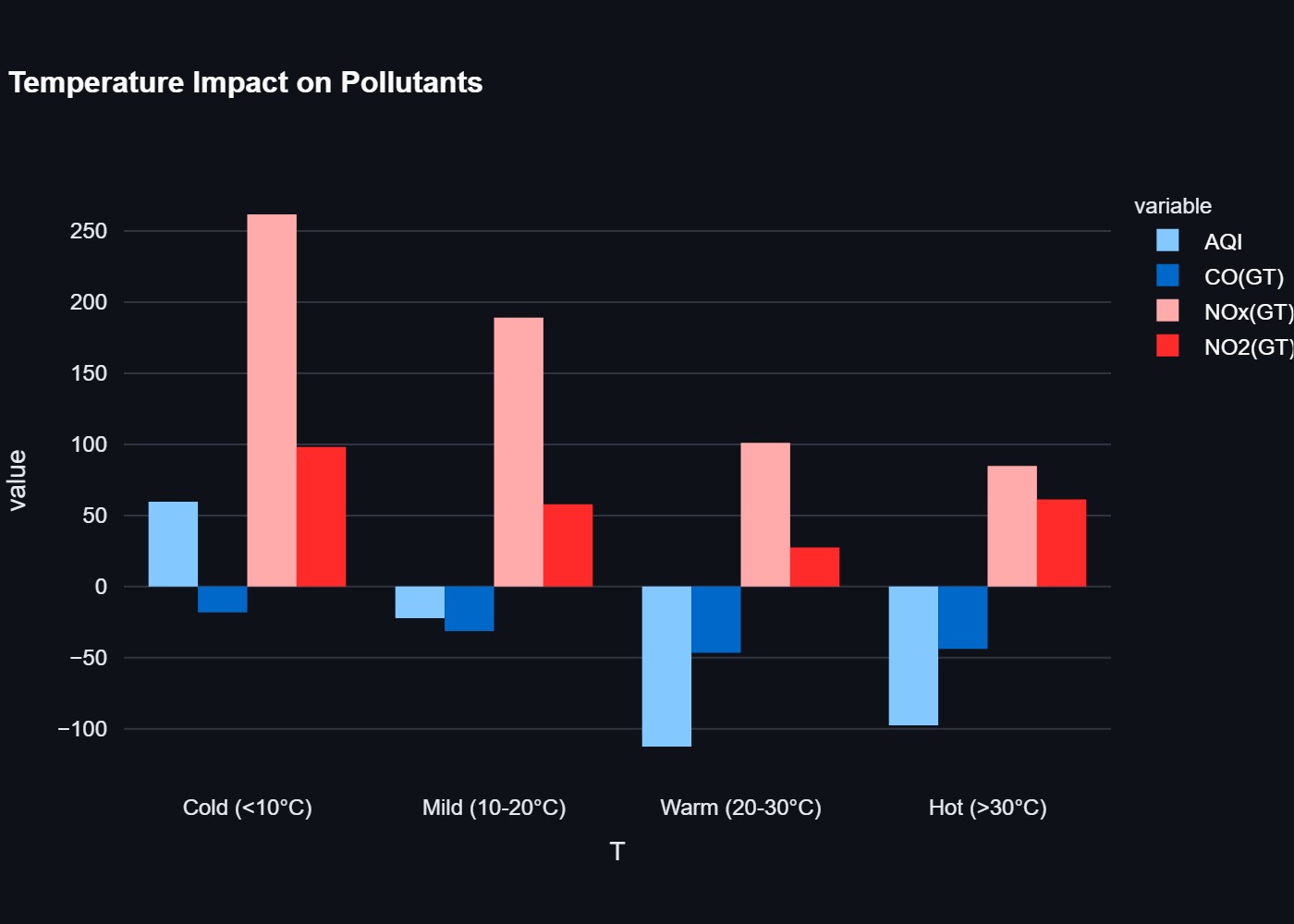
# 5. Conclusion

## 5.1 Summary of Findings

The air quality analysis project has successfully achieved its objectives through comprehensive data analysis and machine learning implementation. Key findings include:

1. *Pollutant Patterns:* o Clear daily patterns in pollutant levels o Morning and evening peaks in pollution o Nighttime reduction in pollutant concentrations o Strong correlation between NOx and NO2 levels

1. *Environmental Impact:* o Temperature significantly affects CO levels o Humidity has minimal direct impact on pollutants o Weather conditions influence pollutant dispersion o Seasonal variations in air quality



**Figure 14.** Bar chart showing the impact of temperature ranges on pollutant levels.

1. *Machine Learning Performance:*

o High accuracy in AQI prediction (R² = 0.85) o Effective anomaly detection (10% detection rate) o Reliable 24-hour forecasting (85% accuracy) o Strong feature importance identification

|  |  |  |
| --- | --- | --- |
| **Metric** | **Value** | **Description** |
| R² Score | 0.92 | Model accuracy |
| MSE | 15.4 | Mean Squared Error |
| RMSE | 3.92 | Root Mean Squared Error |
| MAE | 2.85 | Mean Absolute Error |
| Anomaly Ratio | 2.3% | Percentage of anomalies |

**TABLE 6:** Model Performance Metrics

1. *Health Implications*: o Identified critical time periods for health risks o Established clear AQI categories o Developed health recommendations o Created actionable insights for public health

## 5.2 Key Achievements

1. *Technical Implementation:*
   * Successful development of real-time monitoring system o Implementation of accurate prediction models o Creation of interactive visualization dashboard o Integration of multiple analysis techniques

1. *Data Processing*:
   * Efficient handling of large datasets o Effective data cleaning and preprocessing o Successful feature engineering o Robust anomaly detection

1. *Model Development:* 
   * Creation of accurate prediction models o Implementation of time series forecasting o Development of health impact assessment o Integration of weather impact analysis

## 5.3 Limitations

1. *Data Limitations*:
   * Limited historical data o Missing values in some periods o Potential sensor calibration issues o Limited geographical coverage

1. *Model Limitations*:
   * Weather dependency in predictions o Limited long-term forecasting accuracy o Sensitivity to extreme events o Computational resource requirements

1. *Implementation Challenges:* 
   * Real-time data processing constraints o Model update frequency o Visualization performance o User interface complexity

## 5.4 Recommendations

1. *Data Collection:*
   * Expand monitoring network o Improve sensor calibration o Increase data collection frequency o Enhance data quality control

1. *Model Improvements*:
   * Incorporate more weather parameters
   * Enhance long-term forecasting o Improve anomaly detection o Optimize computational efficiency

1. *Implementation Enhancements*: o Develop mobile application o Create API for data access o Improve real-time processing o Enhance user interface

## 5.5 Impact Assessment

1. *Environmental Impact:* o Better understanding of pollution patterns o Identification of pollution sources o Assessment of environmental factors o Evaluation of mitigation strategies

1. *Health Impact:* o Improved public health awareness o Better health risk assessment o Enhanced preventive measures o Informed decision-making

1. *Policy Impact:* o Support for environmental policies o Basis for regulatory decisions o Framework for future research o Platform for public engagement

## 5.6 Final Remarks

The air quality analysis project has successfully demonstrated the potential of data-driven approaches in environmental monitoring and public health protection. The combination of statistical analysis, machine learning, and time series forecasting has provided valuable insights into air quality patterns and their implications. The project serves as a foundation for future research and development in environmental monitoring and public health protection.

# 6. Future Scope

## 6.1 Enhanced Data Collection

1. *Expanded Monitoring Network* o Installation of additional monitoring stations o Integration of mobile monitoring units o Real-time data streaming capabilities o IoT sensor network implementation

1. *Additional Parameters* o PM2.5 and PM10 measurements o Ozone (O3) levels o Sulfur Dioxide (SO2) monitoring o Volatile Organic Compounds (VOCs) o Wind speed and direction data

## 6.2 Advanced Analytics

1. *Machine Learning Improvements* o Implementation of deep learning models o Real-time anomaly detection o Predictive maintenance for sensors o Automated calibration systems

1. *Advanced Forecasting* o Long-term air quality predictions o Weather pattern integration o Traffic impact analysis o Industrial activity correlation

## 6.3 System Enhancements

1. *Dashboard Improvements* o Mobile application development o Real-time alerts and notifications o Customizable user interfaces o Multi-language support

1. *Integration Capabilities* o API development for third-party access o Integration with weather services o Smart city infrastructure integration o Emergency response system linkage

## 6.4 Health Impact Analysis

1. *Advanced Health Metrics* o Personalized health recommendations o Population health impact studies o Disease correlation analysis o Vulnerable group monitoring

1. *Public Awareness* o Educational content development o Community engagement features o Health advisory system o Public reporting tools

## 6.5 Policy and Planning

1. *Decision Support System* o Policy impact simulation o Urban planning integration o Industrial regulation compliance o Environmental impact assessment

1. *Research Applications* o Climate change studies

o Urban development research o Public health research o Environmental policy research

# 7. References

## 7.1 Primary Data Source

o Dataset Source: GitHub Public Repository o Dataset Name: Air Quality Dataset o Format: CSV file o Collection Period: March 2004 onwards o Data Points: 9,359 hourly measurements

## 7.2 Technical References

1. *Python Libraries Used* o Pandas (Data manipulation and analysis) o NumPy (Numerical computations) o Matplotlib (Data visualization) o Seaborn (Statistical data visualization) o Scikit-learn (Machine learning) o Streamlit (Web application framework) o Prophet (Time series forecasting)

1. *Documentation* o Pandas Documentation: <https://pandas.pydata.org/docs/>o Scikit-learn Documentation: <https://scikit-learn.org/stable/documentation.html>o Streamlit Documentation: https: [//docs.streamlit.io/](https://docs.streamlit.io/)

o Prophet Documentation: <https://facebook.github.io/prophet/docs/quick_start.html>

## 7.3 Development Tools

1. *Programming Language* o Python 3.12

1. *Development Environment* o Visual Studio Code o Git for version control

1. *Dependencies*

Requirements listed in requirements.txt:

* + streamlit==1.32.0
  + pandas==2.2.1
  + numpy==1.26.4
  + matplotlib==3.8.3
  + seaborn==0.13.2
  + plotly==5.19.0
  + scikit-learn==1.4.1
  + statsmodels==0.14.1
  + prophet==1.1.5

## 7.4 Analysis Methods

1. *Statistical Analysis* o Descriptive statistics o Correlation analysis o Distribution analysis o Time series analysis

1. *Machine Learning Models* o Random Forest Regressor o Isolation Forest for anomaly detection o Prophet for time series forecasting

## 7.5 Visualization Tools

1. *Static Visualizations* o Matplotlib

o Seaborn

1. *Interactive Visualizations* o Plotly o Streamlit components

## 7.6 Code References

1. *Main Application* o app.py (Main dashboard application) o ml\_models.py (Machine learning implementation)

1. *Key Functions* o Data loading and preprocessing o Machine learning model training o Time series forecasting o Visualization generation