Industrial Internship Report on

”Air Quality Index Prediction Using Machine Learning”

Prepared by

VANAJA YADLA

Executive Summary
This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT). This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks’ time. My project was Air Quality Index Prediction Using Machine Learning This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

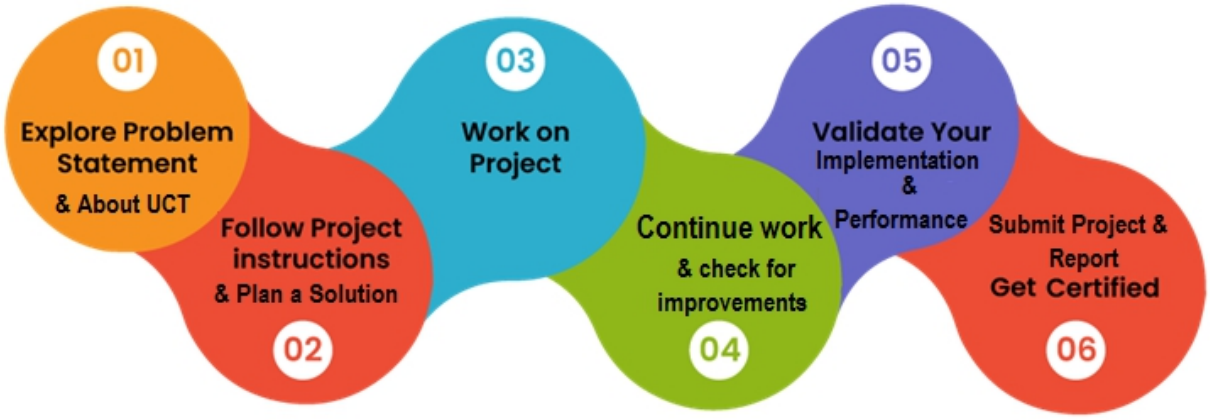
1 Preface

Summary: An Air Quality Index (AQI) prediction system involves gathering historical and real-time data on air quality and pollutants. This data is preprocessed and transformed, with relevant features identified. After splitting the data into training, validation, and test sets, a suitable machine learning model is chosen and trained using the training set. Hyperparameters are fine-tuned using the validation set, and the model's performance is evaluated on the test set. This system aims to accurately forecast AQI levels, enabling informed decision-making to mitigate the effects of poor air quality on public health and the environment.

Problem Statement: Air pollution is a significant environmental concern that affects public health and the quality of life in urban areas. Monitoring and predicting air quality levels is crucial for making informed decisions, implementing effective pollution control measures, and ensuring the well-being of residents.The aim of this project is to develop a machine learning model that accurately predicts the Air Quality Index (AQI) based on various environmental and meteorological factors. The AQI is a standardized measure used to communicate the level of air pollution to the public, indicating the potential health risks associated with the air quality.

Opportunity given by USC/UCT.

How Program was planned



Thank to all who have helped me directly or indirectly.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

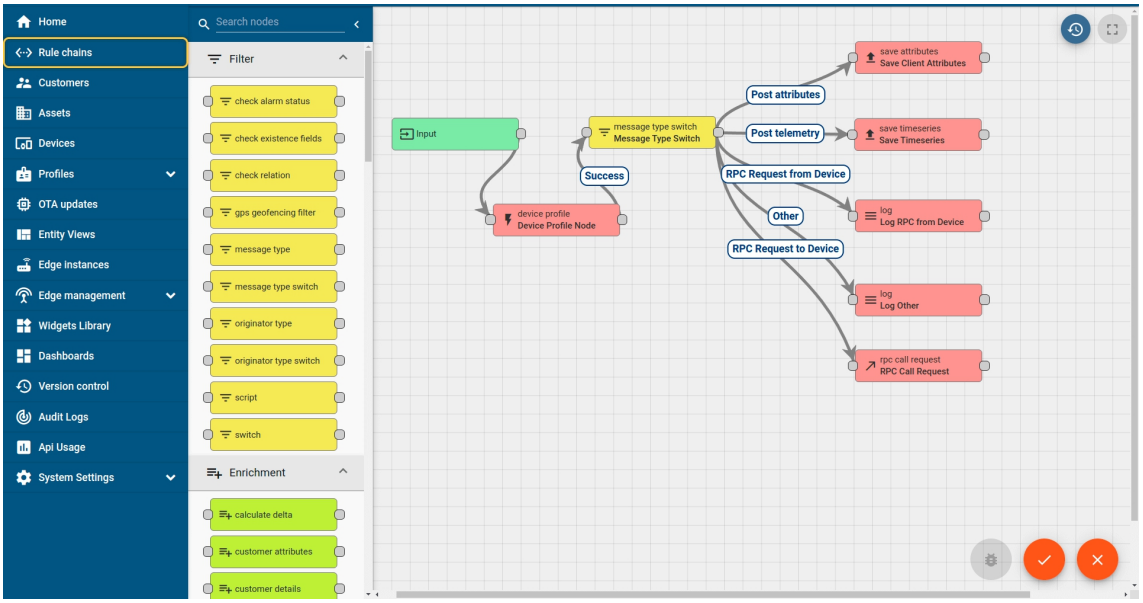
i. UCT IoT Platform (UCT Insight)

UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine



ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

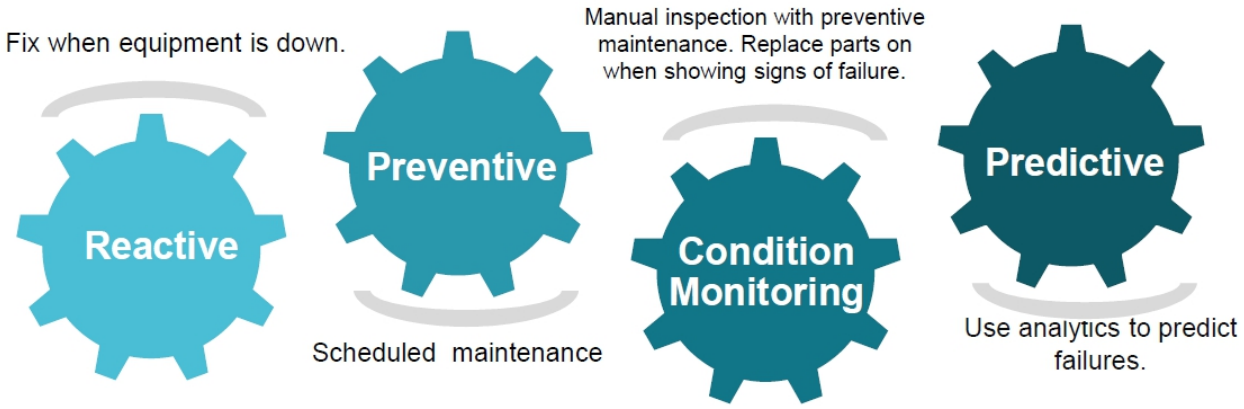
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

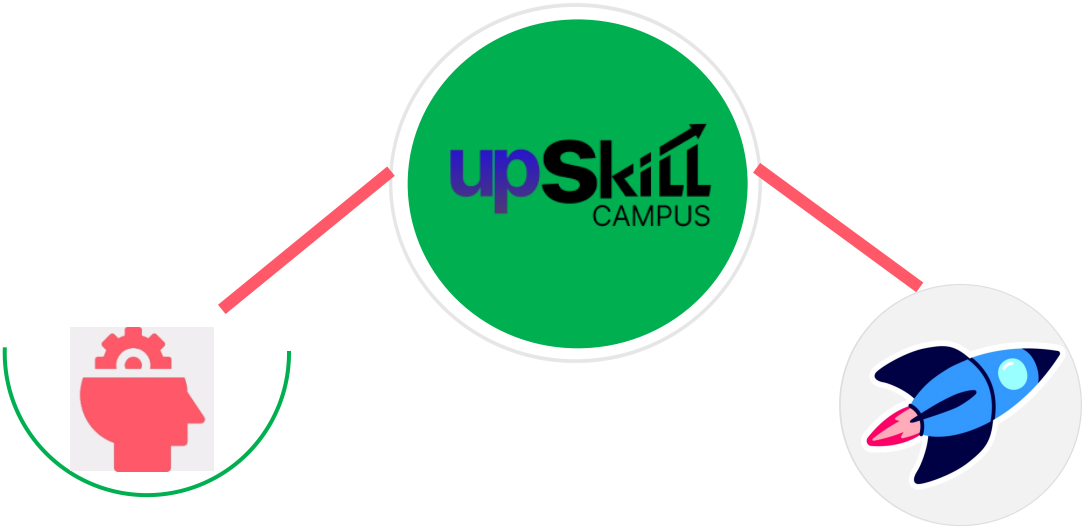
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

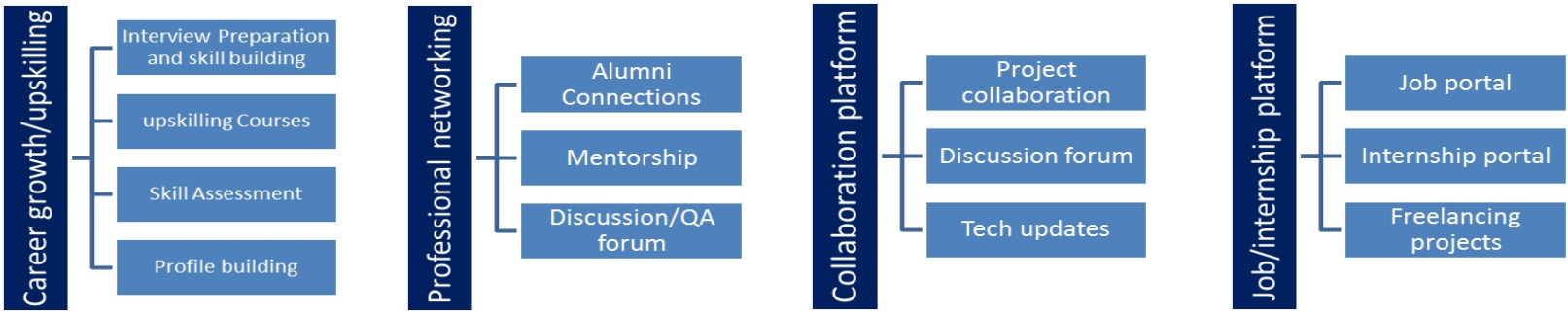
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] https://ieeexplore.ieee.org/document/10150203

[2] https://ieeexplore.ieee.org/abstract/document/8929517

[3] https://ieeexplore.ieee.org/abstract/document/9432078

2.6 Glossary

Terms	Acronym
Machine learning	Machine learning is focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programme
Air Quality Index (AQI)	AQI is a standardized measurement that assesses the quality of the air in a specific area. It condenses complex air pollutant concentrations into a single numerical value, making it easier for the public to understand the potential health risks associated with the air they breathe.

3 Problem Statement

Air pollution is a significant environmental concern that affects public health and the quality of life in urban areas. Monitoring and predicting air quality levels is crucial for making informed decisions, implementing effective pollution control measures, and ensuring the well-being of residents.The aim of this project is to develop a machine learning model that accurately predicts the Air Quality Index (AQI) based on various environmental and meteorological factors. The AQI is a standardized measure used to communicate the level of air pollution to the public, indicating the potential health risks associated with the air quality.

4 Existing and Proposed solution

Existing solution:

The existing system for air quality assessment often relies on conventional monitoring methods, such as fixed-location air quality monitoring stations. These stations collect data on various pollutants at specific points, providing a limited spatial coverage of air quality information. While these systems offer valuable insights, they might not capture the fine-grained variations in air quality within a city or region.

Limitations:

i. **Limited Accuracy:** Inability to effectively capture the complex relationships among various environmental factors and pollutants, leading to less accurate predictions.

ii. **Lack of Adaptability:** Conventional systems might struggle to adapt to changing conditions or new sources of pollution.

iii. **Limited Scalability:** Traditional systems may face challenges when scaling up to large areas.

iv. **Data Overload and Complexity:** Air quality index prediction involve handling large volumes of diverse data, including pollutants data, historical records.

v. **High Time:** Manual methods of air quality index prediciton are time-consuming.

Proposed solution:

The proposed system aims to develop an accurate Air Quality Index (AQI) prediction system using a machine learning model. This system will utilize historical and real-time data on pollutant concentrations and environmental factors. Through careful data preprocessing, feature engineering, and selection, the system will train a machine learning model and predict the output.

AQI_Range: The user can feed the AQI_Range and enter into system. The user can take the different type of ranges as an input for predicting the quality of air.

Feature Extraction: Following are the features of input image which are extracted is - Pollutants

Value addition:

a) **Improved Accuracy:** Machine Learning models can process and analyze large volumes of complex data, capturing intricate relationships and patterns in air quality variables.

b) **Real-time Monitoring:** Machine Learning based AQI prediction systems can provide real-time or near-real-time updates on air quality conditions. This rapid feedback allows for timely interventions and alerts to protect public health.

c) **Adaptability:** Machine Learning models can adapt to changing conditions and learn from new data, making them well-suited for dynamic environments where air quality can change rapidly due to factors like weather, traffic, and industrial activities.

d) **Data Fusion:** These systems can incorporate data from various sources such as air quality monitoring stations and more. By combining multiple data sources, Machine Learning models can generate a more comprehensive and holistic picture of air quality.

4.1 Code submission (Github link)

https://github.com/VanajaYadla23/Air_quality_index_prediction_using_ML/blob/main/code%20pdf.pdf

4.2 Report submission (Github link) :

https://github.com/VanajaYadla23/Air_quality_index_prediction_using_ML

5 Proposed Model

Design flow of proposed solution:

1. **Data Collection:** Collect historical data on air quality, including pollutant concentration levels (PM2.5, PM10, CO, NO2, SO2, O3) . Use a reliable data source such as government environmental agencies or local monitoring stations.

2. **Data Preprocessing:** Handle missing values, outliers, and inconsistencies in the dataset. Normalize or scale the features to ensure uniformity and improve model convergence.

3. **Feature Engineering**: Create additional features or transform existing ones based on domain knowledge and analysis. For example, derive features like the Air Quality Index (AQI) of the previous day, day of the week, and seasonal indicators. Incorporate lagged values of air quality and meteorological variables to capture time dependencies.

4. **Data Splitting:** Split the dataset into training, validation, and testing sets. Time-based splitting is crucial to account for temporal dependencies.

5. **Model Selection:** Consider using a machine learning model suitable for time series forecasting or regression tasks. Some options include: Time Series Models: ARIMA, SARIMA, or state-of-the-art models like Prophet. Machine Learning Models: Random Forest, Gradient Boosting, or neural networks (LSTM, GRU, or Transformer-based models).

6. **Hyperparameter Tuning:** Optimize the hyperparameters of the selected model using techniques like grid search or Bayesian optimization.

7. **Model Training:** Train the model using the training dataset while validating its performance on the validation set. Ensure the model captures both short-term and long-term trends in air quality.

8. **Model Evaluation:** Assess the model's performance using appropriate evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared. Pay attention to metrics that are interpretable in the context of air quality.

5.1 High Level Diagram

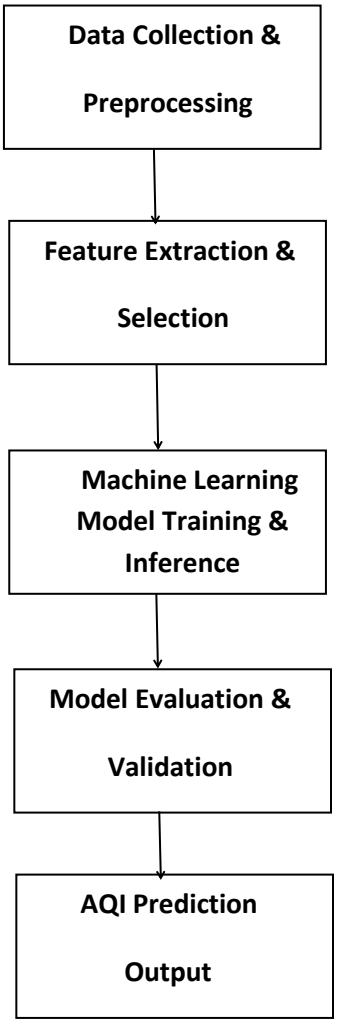


Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low Level Diagram

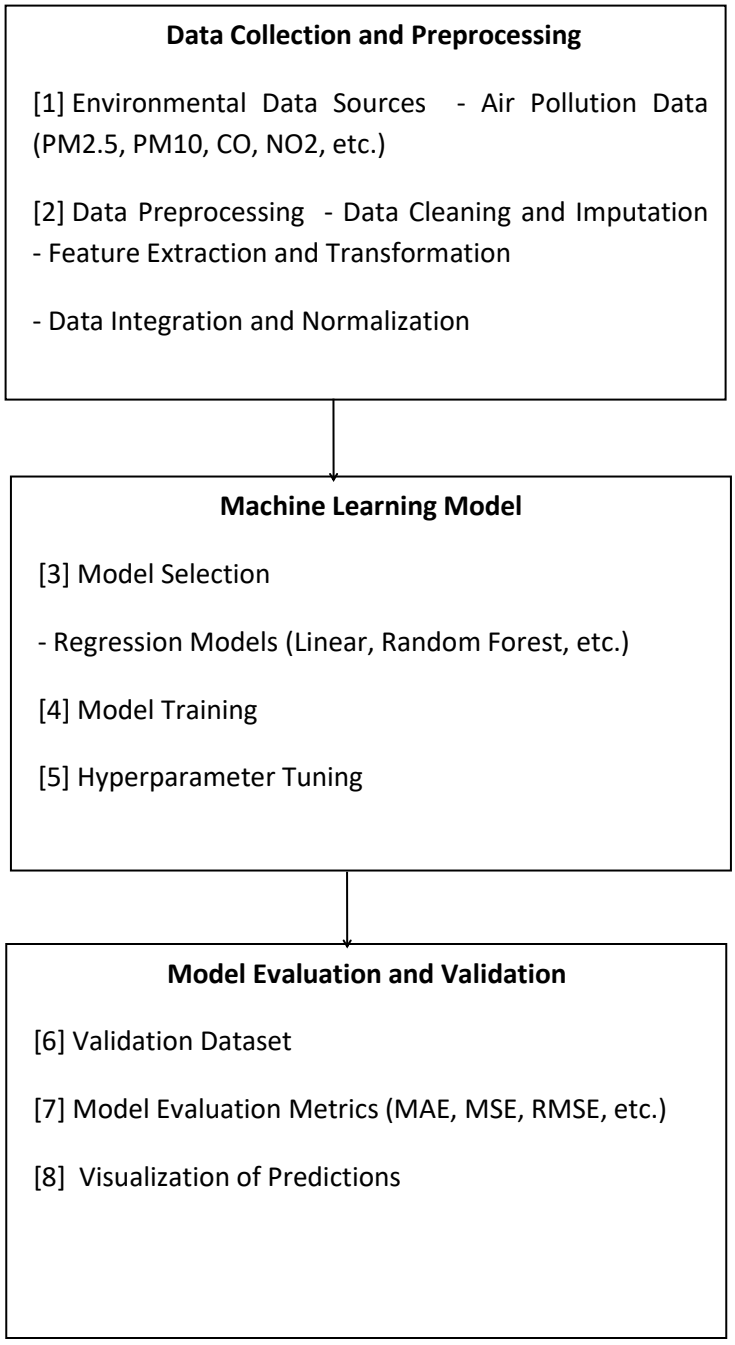


Figure 2: LOW LEVEL DIAGRAM OF THE SYSTEM

5.3 Interfaces

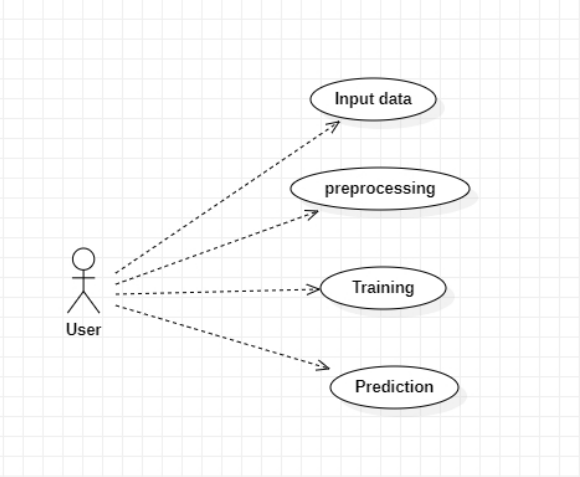


Figure 3: USE CASE DIAGRAM OF THE SYSTEM

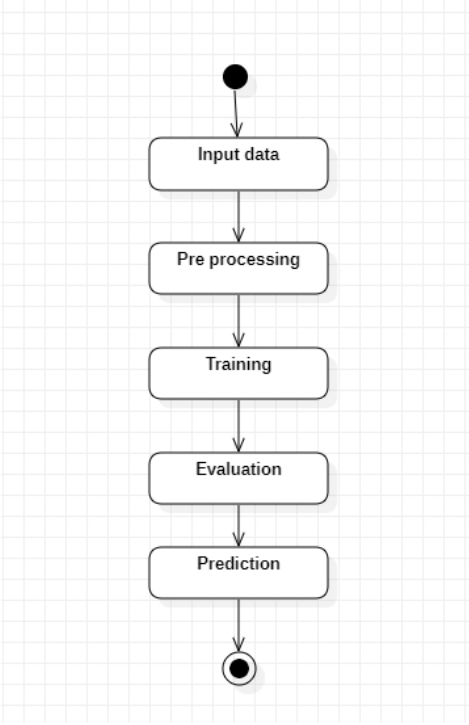


Figure 4: STATE CHART DIAGRAM OF THE SYSTEM

6 Performance Test

i. Memory Constraints:

Design Approach: Optimize your machine learning model and data preprocessing techniques to minimize memory usage. Consider using more memory-efficient algorithms or data structures.

Test Results: Measure memory consumption during training and inference to ensure it remains within acceptable limits. Monitor memory usage in real-time deployments.

Recommendations: If memory constraints are exceeded, consider using cloud-based solutions with scalable memory resources or implementing data streaming techniques to reduce the memory footprint.

ii. MIPS (Speed and Operations per Second) Constraints:

Design Approach: Optimize the model for efficient computation by utilizing hardware acceleration (e.g., GPUs, TPUs) and parallel processing. Implement model quantization or compression for faster inference.

Test Results: Benchmark the system for inference speed and operations per second, ensuring it meets the required throughput.

Recommendations: If the system falls short of speed requirements, consider hardware upgrades or distributed computing solutions. Profile code for performance bottlenecks and optimize them.

iii. Accuracy Constraints:

Design Approach: Use appropriate machine learning algorithms and feature engineering techniques to enhance prediction accuracy. Implement ensemble methods or model stacking for improved results.

Test Results: Evaluate the model's accuracy against real-world AQI measurements. Calculate metrics like MAE, RMSE, or R-squared to assess performance.

Recommendations: If accuracy is below acceptable levels, collect more diverse and high-quality data, refine feature selection, and experiment with different model architectures. Continuously update the model as more data becomes available.

iv. Durability Constraints:

Design Approach: Ensure system components, including hardware and software, are robust and fault-tolerant. Implement redundancy and failover mechanisms.

Test Results: Conduct stress testing and simulate failure scenarios to verify system durability. Monitor system uptime and performance over extended periods.

Recommendations: In case of failures, have backup systems in place. Implement automatic recovery mechanisms and regularly update software to address vulnerabilities and ensure long-term durability.

v. Power Consumption Constraints:

Design Approach: Optimize hardware selection and software configurations for energy efficiency. Use low-power components and consider energy-efficient algorithms.

Test Results: Measure power consumption during different system operations and evaluate it against constraints.

Recommendations: If power consumption exceeds limits, consider energy-efficient hardware upgrades, such as ARM-based processors, and optimize code for energy efficiency. Implement dynamic power management strategies.

vi. Scalability Constraints:

Design Approach: Ensure the system architecture is scalable to handle increasing data volumes and user requests. Employ load balancing and horizontal scaling.

Test Results: Evaluate system performance under load and scalability tests. Measure response times and throughput as the workload increases.

Recommendations: If scalability is an issue, consider cloud-based solutions that can easily scale resources as needed. Implement microservices architecture to enable independent scaling of components.

6.1 Test Plan/ Test Cases

Objectives: Verify the accuracy of AQI predictions. Validate the system's ability to handle various input scenarios. Ensure robustness and reliability under different conditions.

Test Environment: Testing will be performed on a dedicated test server. Real historical AQI and meteorological data will be used for testing.

Test Cases:

1. Data Preprocessing and Feature Engineering Tests:

Test Case 1.1: Verify that missing data in the input dataset is handled appropriately (e.g., imputation).

Test Case 1.2: Check for outliers and confirm that they are handled correctly (e.g., removal or transformation).

Test Case 1.3: Confirm that feature scaling or normalization is performed as required.

2. Model Training and Validation Tests:

Test Case 2.1: Train the model using a subset of the dataset and confirm that it converges without errors.

Test Case 2.2: Validate the model's accuracy on a separate validation dataset using metrics such as MAE, MSE, RMSE, and R-squared.

Test Case 2.3: Ensure that the model does not overfit the training data by comparing training and validation performance.

3. Hyperparameter Tuning Tests:

Test Case 3.1: Perform hyperparameter tuning and validate that the model's performance improves.

Test Case 3.2: Ensure that hyperparameter tuning does not result in overfitting or reduced generalization.

4. Time Series Testing:

Test Case 4.1: Test the model's ability to make accurate predictions for AQI values at different time intervals (e.g., hourly, daily, monthly).

5. Out-of-Sample Testing:

Test Case 5.1: Use a separate test dataset (not seen during model training) to evaluate the model's performance.

6. Real-time Testing:

Test Case 6.1: Simulate real-time data input and verify that the model produces timely and accurate predictions.

7. Prediction Interval Testing:

Test Case 7.1: Confirm that prediction intervals (e.g., 95% confidence intervals) are correctly computed and capture the uncertainty in AQI predictions.

8. Feature Importance Testing:

Test Case 8.1: Assess whether the model's feature importance scores align with domain knowledge and expectations.

9. Bias and Fairness Testing:

Test Case 9.1: Check for bias or fairness issues in the model's predictions, particularly related to demographic or geographic factors.

10. Real-World Scenario Testing:

Test Case 10.1: Introduce real-world challenges, such as sudden data gaps, sensor malfunctions, or extreme weather events, and assess the model's ability to handle them gracefully.

11. Scalability Testing:

Test Case 11.1: Evaluate the system's performance when handling a high volume of requests, ensuring that response times remain acceptable.

12. User Interface Testing:

Test Case 12.1: Test the user interface for usability, accessibility, and accuracy in presenting AQI predictions to end-users.

13. Backtesting:

Test Case 13.1: Conduct backtesting by comparing past model forecasts with actual observations to assess historical accuracy.

14. User Feedback Testing:

Test Case 14.1: Collect feedback from users and assess their satisfaction with the system and its predictions.

Test Reporting: Document and report the results of each test case, including any issues or anomalies discovered. Prioritize and address any critical issues before deployment.

Test Automation: Implement automated testing where applicable to facilitate repeated testing as new data becomes available or the model is updated.

Test Maintenance: Establish a schedule for regular testing and maintenance to ensure the AQI prediction system remains accurate and reliable over time.

This test plan covers a range of scenarios and factors to thoroughly evaluate the AQI prediction system and ensure its effectiveness in providing accurate and reliable air quality information to users.

6.2 Test Procedure

i. **Data Splitting:** Divide your dataset into three subsets: training data, validation data, and test data. A common split is 70% for training, 15% for validation, and 15% for testing. Ensure that the data split is done randomly to avoid any bias.

ii. **Preprocessing:** Apply the same preprocessing steps to the test data as you did to the training data. This includes handling missing values, scaling/normalizing features, and encoding categorical variables.

iii. **Model Loading:** Load the pre-trained machine learning model that you intend to evaluate.

iv. **Prediction:** Use the loaded model to predict AQI values for the test data.

v. **Evaluation Metrics:** Calculate a set of evaluation metrics to assess the model's performance. Common metrics include:

Mean Absolute Error (MAE): Measures the average absolute difference between predicted and actual AQI values.

Mean Squared Error (MSE): Measures the average squared difference between predicted and actual AQI values.

Root Mean Squared Error (RMSE): The square root of MSE. Coefficient of Determination (R-squared, R2): Measures how well the model explains the variance in the AQI data.

Depending on your specific requirements, you may also calculate other metrics like precision, recall, or F1-score if you treat AQI prediction as a classification problem (e.g., categorizing air quality as good, moderate, unhealthy).

vi. **Visualizations:** Create visualizations to compare predicted AQI values with actual AQI values. Time series plots or scatter plots can be useful for this purpose.

6.3 Performance Outcome

Mean Absolute Error (MAE): A lower MAE indicates that, on average, the model's predictions are closer to the actual AQI values. For example, an MAE of 10 means that, on average, the model's predictions are off by 10 units on the AQI scale.

Mean Squared Error (MSE) or Root Mean Squared Error (RMSE): A lower MSE or RMSE implies smaller prediction errors. These metrics emphasize larger errors more than MAE, making them suitable for penalizing outliers.

Coefficient of Determination (R-squared, R2): An R2 value close to 1 suggests that the model explains most of the variance in AQI, indicating a good fit. An R2 value near 0 means the model performs no better than a simple mean prediction.

7 My learnings

In simple terms, when we want to predict the Air Quality Index (AQI), we're essentially teaching a computer to guess how clean or polluted the air will be in a specific place and time. We do this by giving the computer lots of information about things like weather, pollution levels, and other factors.

Then, we test how good the computer is at guessing by comparing its predictions to the actual air quality measurements. We use different tests to see if the computer is doing a good job, like checking how close its guesses are to the real values.

By doing this, we can create a computer program that can tell us how clean or polluted the air will be in the future. This is important for our health and the environment, as it helps us make decisions to stay safe and reduce pollution.

8 Future work scope

The scope of air quality index prediction using Machine Learning provides enhanced accuracy and precision, integration of multi-model data, Real-time monitoring and adaptive recommendations.