24-25J-224

Intelligent Eco-Urban Monitoring System (IEMS)

Sri Lanka Institute of InformationTechnology

SLIIT UNI
THE KNOWLEDGE UNIVERSITY

OUR TEAM

24-25J-224

Dr. Samantha Rajapaksha

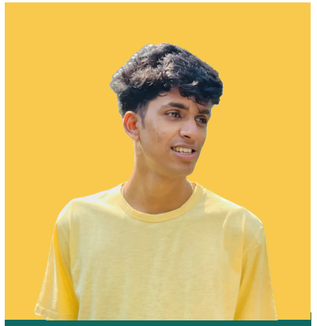
Head | Department of IT

Supervisor

Ms. Kaushika Kavindi

Assistant Lecturer

Co-Supervisor



Thuduwage I.M.H.G

IT21169380

Software Engineering

Arandara S. D.

IT21164330

Software Engineering

Karunarathne R. Y. D.

IT21169144

Software Engineering

Kodithuwakku C.K.

IT21156960

Software Engineering

SLIIT UNI

THE KNOWLEDGE UNIVERSITY

IEMS

INTRODUCTION

IEMS

Urban Environmental Challenges

- Rapid urbanization leads to issues like poor air quality, green space loss, noise pollution, and high vehicle emissions.

Why Address These Issues?

- Sustainable urban living is vital for residents' well-being, requiring effective monitoring and solutions.

Introduction to IEMS

- IEMS addresses these challenges with integrated monitoring and prediction, featuring EcoSensor AI, GreenVision AI, NoiseGuard AI, and Eco Go.

RESEARCH PROBLEM

IEMS

Environmental Challenges in Urban Areas

- **Air Quality Concerns**
  - Urban pollution from transport and industry includes harmful pollutants like PM2.5, NO2, and O3, posing health risks.
- **Green Space Degradation**
  - Urbanization reduces and fragments green spaces, impacting carbon capture and biodiversity.
- **Noise Pollution**
  - Noise from transport and construction causes stress and health issues like sleep disturbances.
- **Vehicle Emissions**
  - Increasing traffic contributes to air pollution and greenhouse gases, requiring emission reduction strategies.

IEMS

PROPOSED SOLUTION

Integrated Environmental Monitoring System (IEMS)

- **EcoSensor AI**
  - Real-time air quality monitoring and predictions.
- **NoiseGuard AI**
  - Smart noise monitoring and source identification.
- **Eco Go**
  - Predicts and reduces vehicle CO2 emissions.
- **GreenVision AI**
  - Manages green spaces using satellite imagery.

Main Objective

Enhancing Urban Sustainability

Goal

To enhance urban sustainability through effective monitoring, prediction, and mitigation of environmental challenges.

Objectives

- Optimize green spaces.
- Improve air quality.
- Reduce noise pollution.
- Lower vehicle emissions by leveraging advanced AI and IoT technologies.

IEMS

Sub Objectives

Specific Goals and Tasks

Arandara.S.D:

- Deploy IoT sensors for CO2 level monitoring.
- Train models for air quality prediction and recommendations.
- Implement adaptive sensor calibration using AI for dynamic accuracy adjustments.

Thuduwage I.M.H.G:

- Analyze satellite imagery for green space management.
- Train deep learning models for vegetation assessment.
- Provide visualization tools for data analysis and decision making.

IEMS

Sub Objectives

Specific Goals and Tasks

Karunarathne R.Y.D:

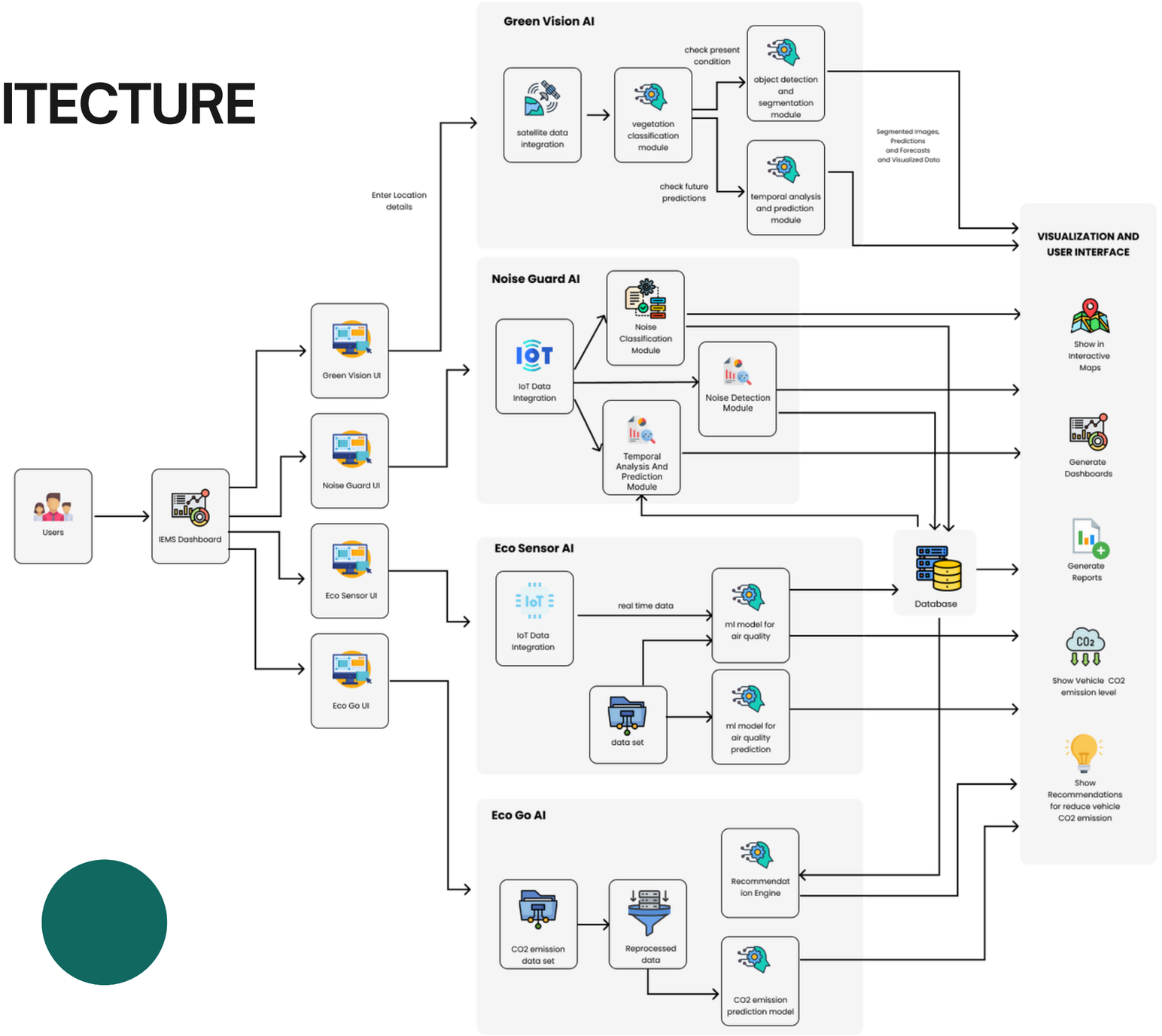
- Implement smart noise monitoring with IoT sensors.
- Classify noise sources and predict future noise levels.
- Develop alert mechanisms for noise thresholds.

Kodithuwakku C.K

- Develop a personalized CO2 emissions prediction and reduction engine.
- Train machine learning models and implement user feedback systems.
- Continuously retrain models to adapt to new data and feedback.

SYSTEM ARCHITECTURE

IEMS



Team Member 1

Green Vision AI

I.M.H.G. Thuduwage
IT21169380

GreenVision AI:
Managing Urban Green Spaces

Background

Urban green spaces are essential for reducing heat, improving air quality, and supporting biodiversity, but urban density and pollution make maintenance challenging.

- **Deep Learning Analysis**
  - AI analyzes satellite images to manage green spaces.
- **Vegetation Indices and Object Detection**
  - Assesses health and distribution of greenery.
- **Informed Decision-Making**
  - Identifies areas needing more green space.
- **Environmental Benefits**
  - Enhances biodiversity, reduces heat, and improves environmental quality.

Research Problem

Urban areas struggle to assess and manage green spaces due to fragmented data and lack of effective analysis tools. An advanced system is needed to analyze and enhance green spaces.

IT21169380 | Thuduvage I.M.H.G. | 24-25J-224

Obejctives

IEMS

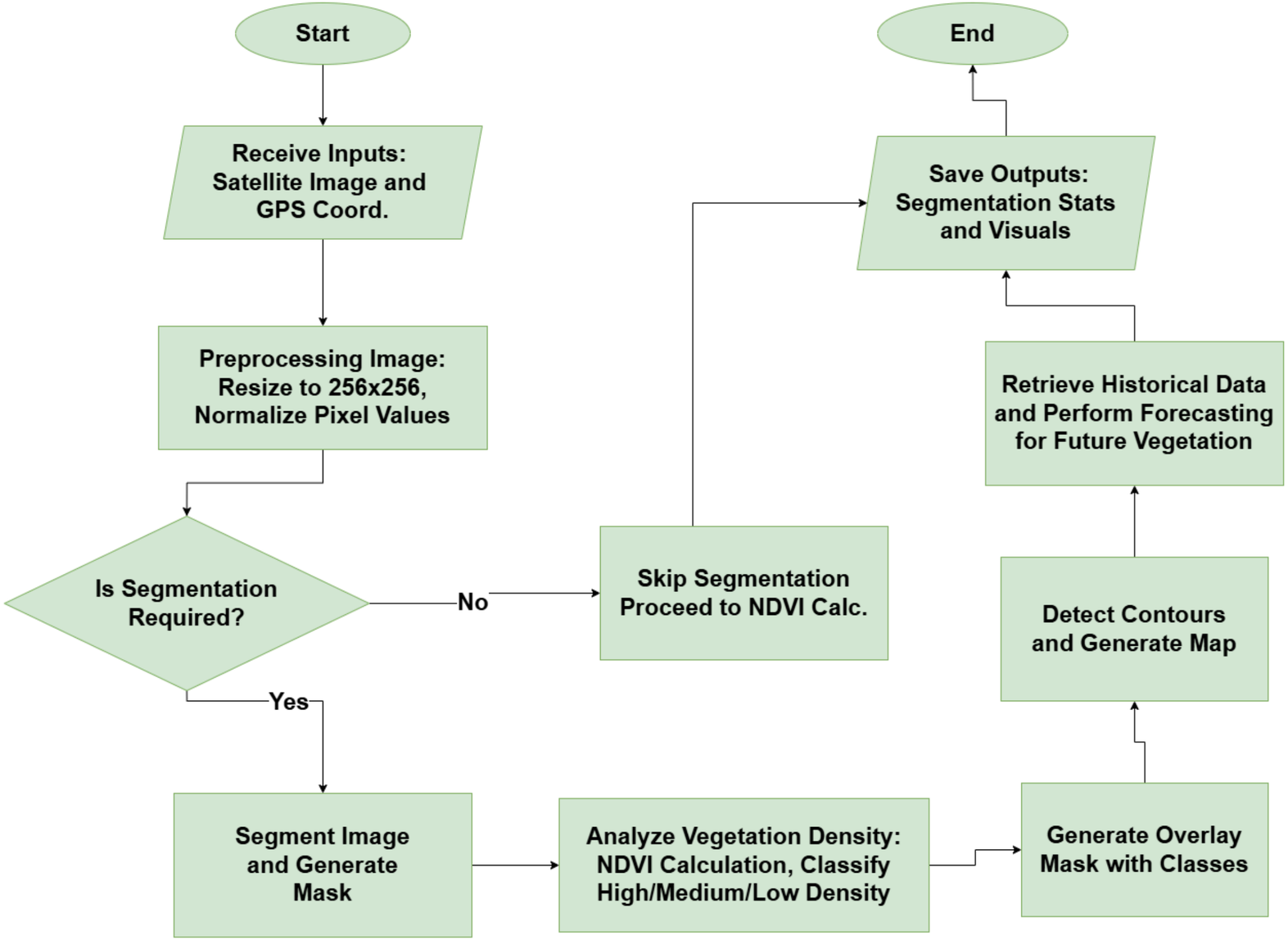
Main Objectives

- Improve urban sustainability by monitoring, predicting, and managing green spaces.
- Provide cities with advanced tools for better urban development and environmental management

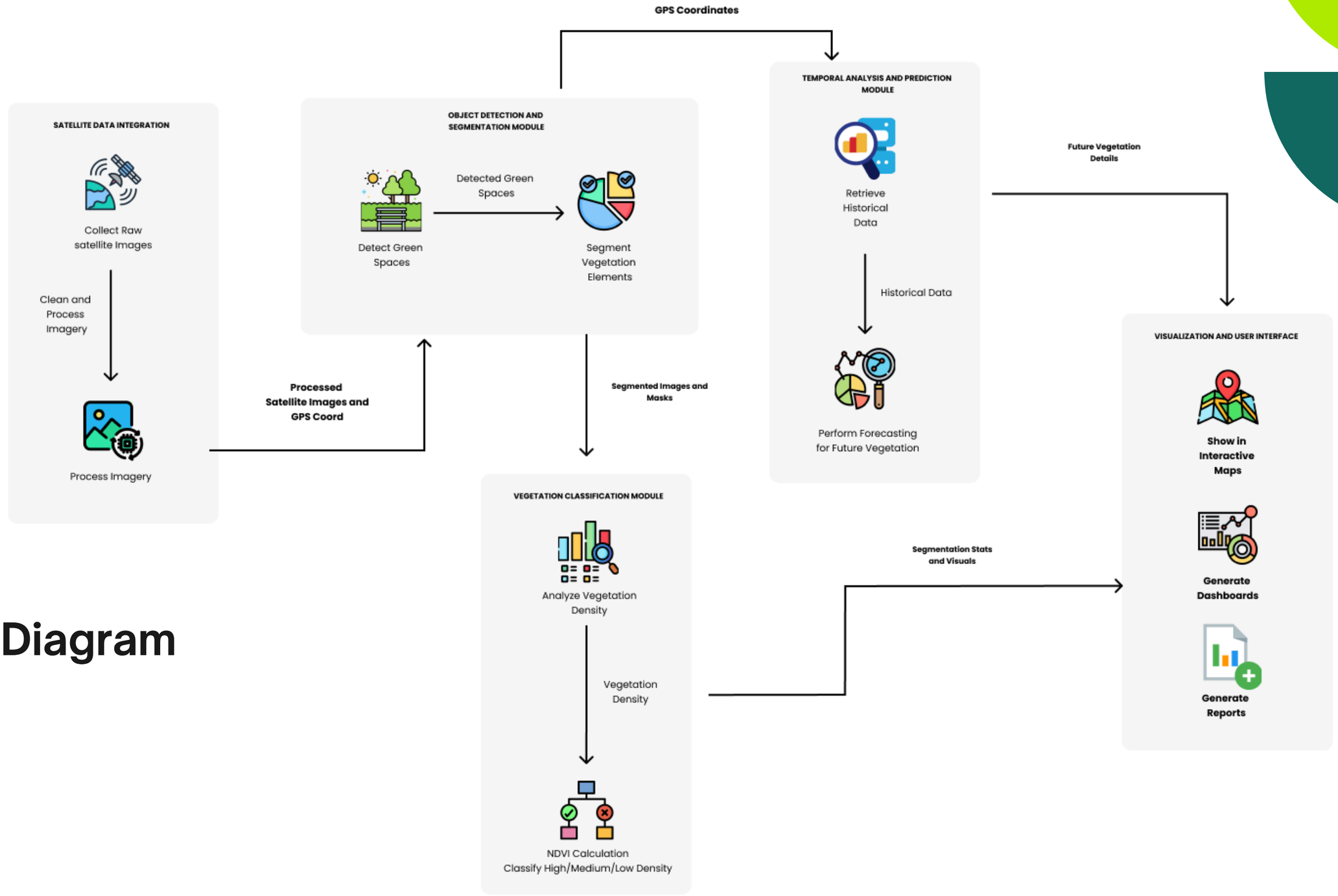
Sub-Objectives

- Use satellite images and AI to manage green spaces.
- Identify areas needing more greenery based on green coverage and Temporal Analysis and Prediction.
- Guide decisions on creating new green spaces to improve biodiversity and reduce heat.

Flowchart


Start
Receive Inputs: Satellite Image and GPS Coord.
Preprocessing Image: Resize to 256x256, Normalize Pixel Values
Is Segmentation Required?
No
Skip Segmentation Proceed to NDVI Calc.
Yes
Segment Image and Generate Mask
Analyze Vegetation Density: NDVI Calculation, Classify High/Medium/Low Density
Generate Overlay Mask with Classes
Detect Contours and Generate Map
Retrieve Historical Data and Perform Forecasting for Future Vegetation
Save Outputs: Segmentation Stats and Visuals
End

System Diagram



Dataset Overview

Tree Segmentation Dataset

Source: ArcGIS Urban Green Space Data.

Key Features:

- High-resolution satellite images.
- Labeled regions for tree and non-tree areas.
- Includes GPS coordinates for spatial analysis.

LSTM Forecasting Dataset

Source: ArcGIS Urban Green Space Data.

Structure:

- Rows: 132 (monthly data from January 2014 onwards).
- Columns:
  - Timestamp: Date values (e.g., 2014-01-01).
  - L0001 to L1000: Satellite measurements or tree coverage data as float values

Purpose:

- To predict vegetation density trends using historical data.

Data Preprocessing

Input Image Preprocessing:

- Resize images to 256×256
- Normalize pixel values using Min-Max scaling.
- Convert to RGB format for model compatibility.

Mask Preprocessing:

- Convert ground truth masks to binary (tree vs. background).
- One-hot encoding for segmentation model training.

NDVI Calculation:

- Categorized vegetation density into high, medium, and low levels

Time Series Forecasting:

- Load and Clean Data
  - Remove unnecessary columns (e.g., Timestamp).
  - Retain 1000 feature columns for processing.
- Sliding Window Generation
  - Input (X): Last 12 time steps (past data).
  - Output (Y): Next 3 time steps (future predictions).
- Reshape Data
  - Input Shape: (samples, 12, 1000)
  - Output Shape: (samples, 3, 1000)
- Split Outputs
  - Separate predictions for each forecast month
- Inference
  - Use the last 12 time steps to forecast the next 3 time steps for specific locations.

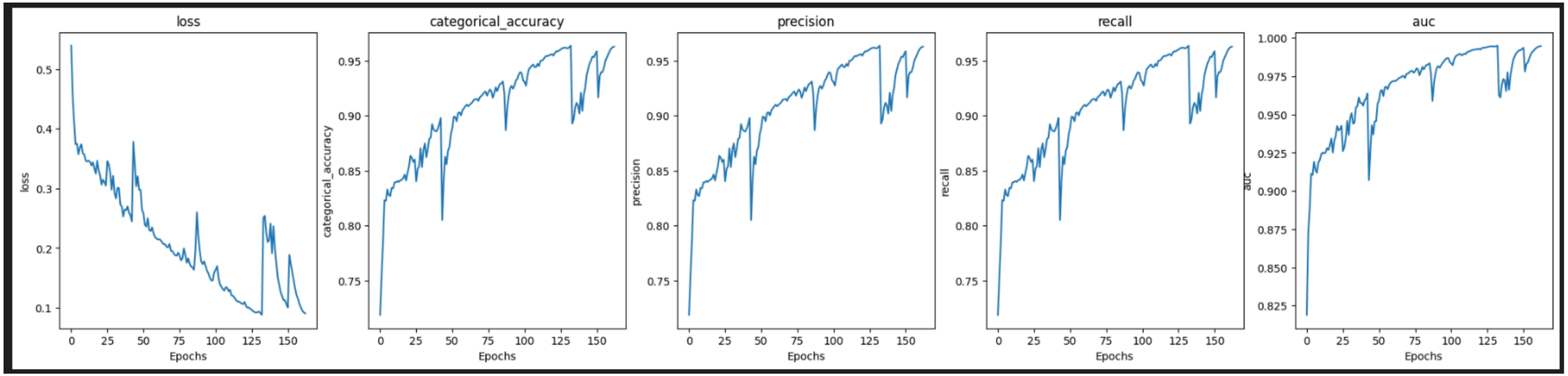
Model Architecture

1. Tree Segmentation Model
    - A Convolutional Neural Network (CNN) trained to identify tree regions in images.
    - Outputs a binary mask indicating areas with trees (foreground) and non-tree regions (background).
2. Vegetation Classification
    - NDVI Calculation for Categorized vegetation density into high, medium, and low levels
3. Tree Coverage Forecasting Model (LSTM):
    - A Long Short-Term Memory (LSTM) model that predicts future tree coverage values based on historical time-series data. It processes sequences of coverage measurements to forecast values for the next few periods.

Accuracy Details

Segmentation Model

- Accuracy - 0.9626
- precision - 0.9626
- auc - 0.9948
- loss - 0.0906



Example Usage

POST http://127.0.0.1:5000/api/tree Send

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

Key		Value	Description	Bulk Edit
file	File	⚠ WhatsApp Image 2024-11-02 at 10.20.38 ...		
latitude	Text	80.133474		
longitude	Text	7.127885		

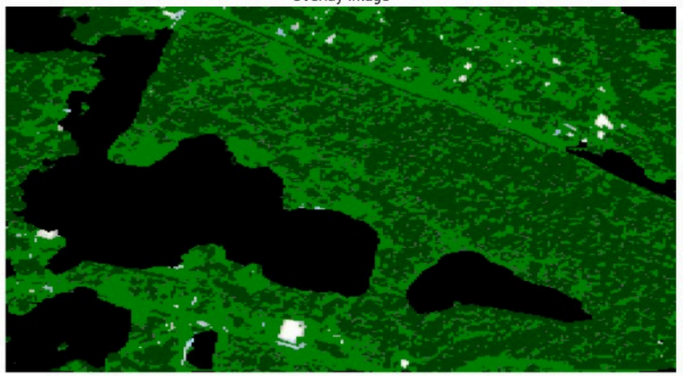
Body Cookies Headers (6) Test Results 200 OK 3.08 s 904 B Save Response

Pretty Raw Preview Visualize JSON


```
"forecast": {
    "month 1": "42.880001068115234 %",
    "month 2": "42.9900016784668 %",
    "month 3": "43.11000061035156 %"
},
"segmentation_results": {
    "contour": "store/Contour/4c8a75f4-e0b2-4a52-9a65-83606161a47a.png",
    "original": "store/Original/79be3643-15fd-4bdb-9a9a-e8358cf5ca3a.png",
    "overlay": "store/Overlay/bbecc58b-f618-4db3-a4ca-4b7ca2a2f005.png",
    "predict": "store/Predict/1b59cf2f-2b7e-45ca-a424-aac86bc948d0.png"
},
"segmentation_stats": {
    "Green Percentage": "74.36 %",
    "High Vegetation Density Coverage": "3.35 %",
    "Low Vegetation Density Coverage": "55.73 %",
    "Medium Vegetation Density Coverage": "40.92 %",
    "NDVI Score": 0.02
```


Original Image

Overlay Image



Contour Image

Predict Image

Technologies to be Used

Deep Learning

- CNNs, RNN, U-Net for classification and segmentation, OpenCV

Temporal Analysis

- LSTM

Remote Sensing

- Satellite imagery analysis.

TOOLS & TECHNOLOGIES

IEMS

Requirements

IEMS

System Requirements

- High-resolution satellite imagery, High-Performance Computing

Software Requirements

- **Functional Requirements**
  - Data Acquisition and Preprocessing
  - Vegetation Analysis and Classification
  - Temporal Analysis and Forecasting
  - Visualization and Reporting
- **Non-Functional Requirements**
  - Performance
  - Scalability
  - Reliability
  - Usability

Team Member 2

EcoSensor AI

Arandara S. D.
IT21164330

EcoSensor AI
Air Quality Monitoring ,
Prediction and Recommondation Management

Background

Urbanization challenges air quality, vital for health and sustainability. EcoSensor AI uses IoT sensors for real-time pollutant monitoring and AI for analysis and predictions, offering recommendations to maintain optimal CO2 levels.

Air Quality Monitoring and IoT Sensor Deployment

- Strategically place IoT sensors in urban areas to collect data on air pollutants like CO2

Data Analysis and Prediction

- Use advanced machine learning models, such as Random Forest, Gradient Boosting, and LSTM networks, to analyze data and predict future air quality trends..

Recommendations for Air Quality Management

- EcoSensor AI includes predictive models that provide recommendations to maintain optimal CO2 levels and overall air quality.

Research Problem

Urbanization challenges air quality and green space management, impacting public health and the environment. Traditional systems lack real-time data, future air quality predictions, and actionable recommendations.

IT21164330 | Arandara S.D. | 24-25J-224

Obejctives

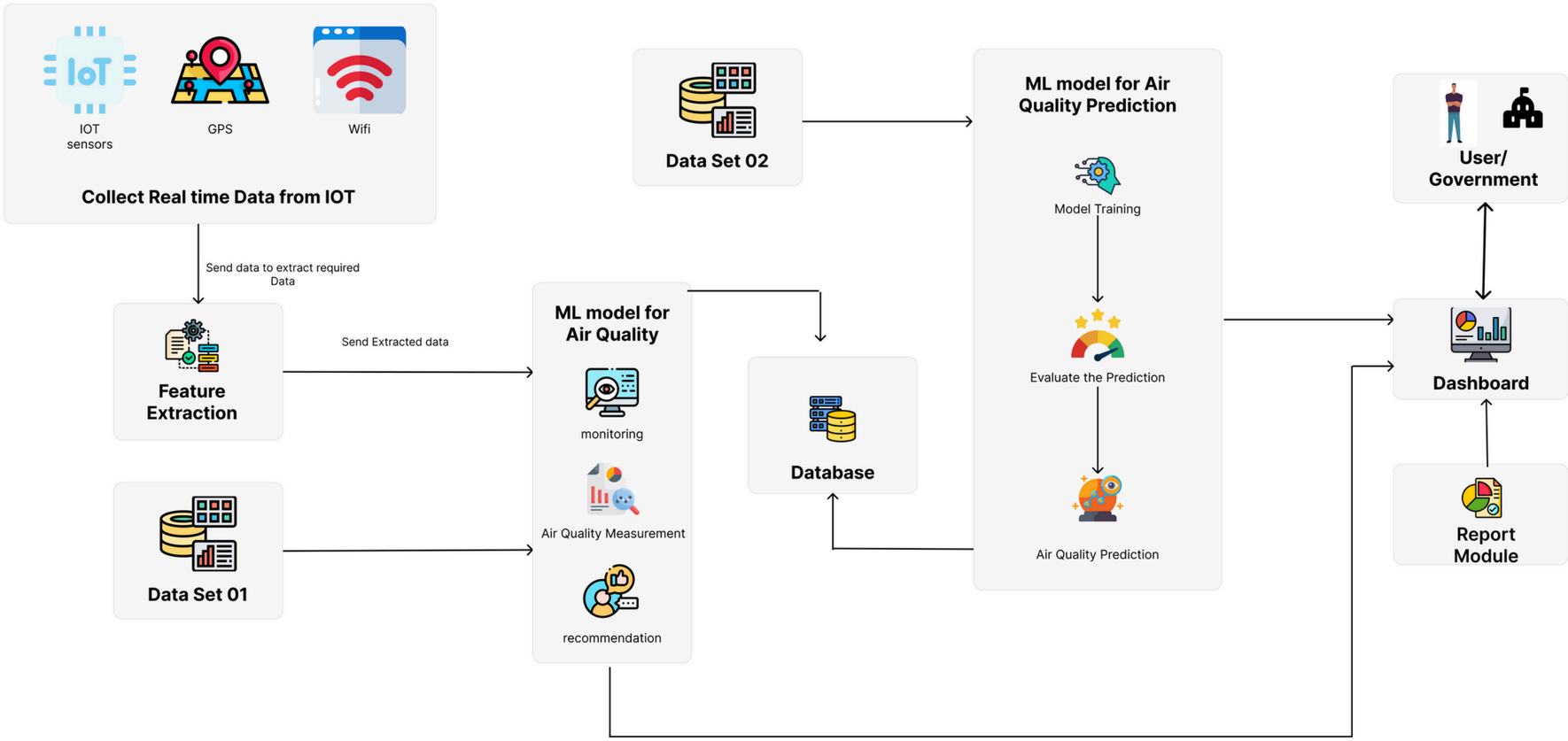
Main Objectives

- Enhance urban sustainability through effective monitoring, prediction, and management of air quality using IoT sensors and AI.
- Generate actionable recommendations for air quality management based on predictive models.

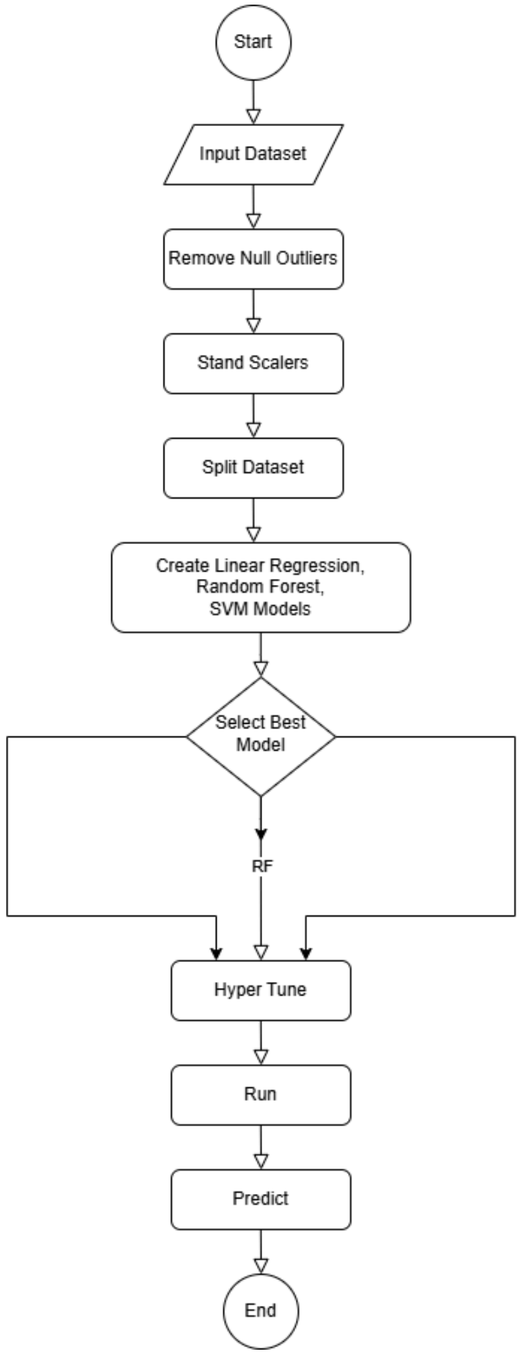
Sub-Objectives

- Deploy IoT Sensors
- Develop Predictive Models
- Provide Recommendations
- Develop User Interface

System Diagram



IT21164330 | Arandara S.D. | 24-25J-224



Flowchart

IT21164330 | Arandara S.D. | 24-25J-224

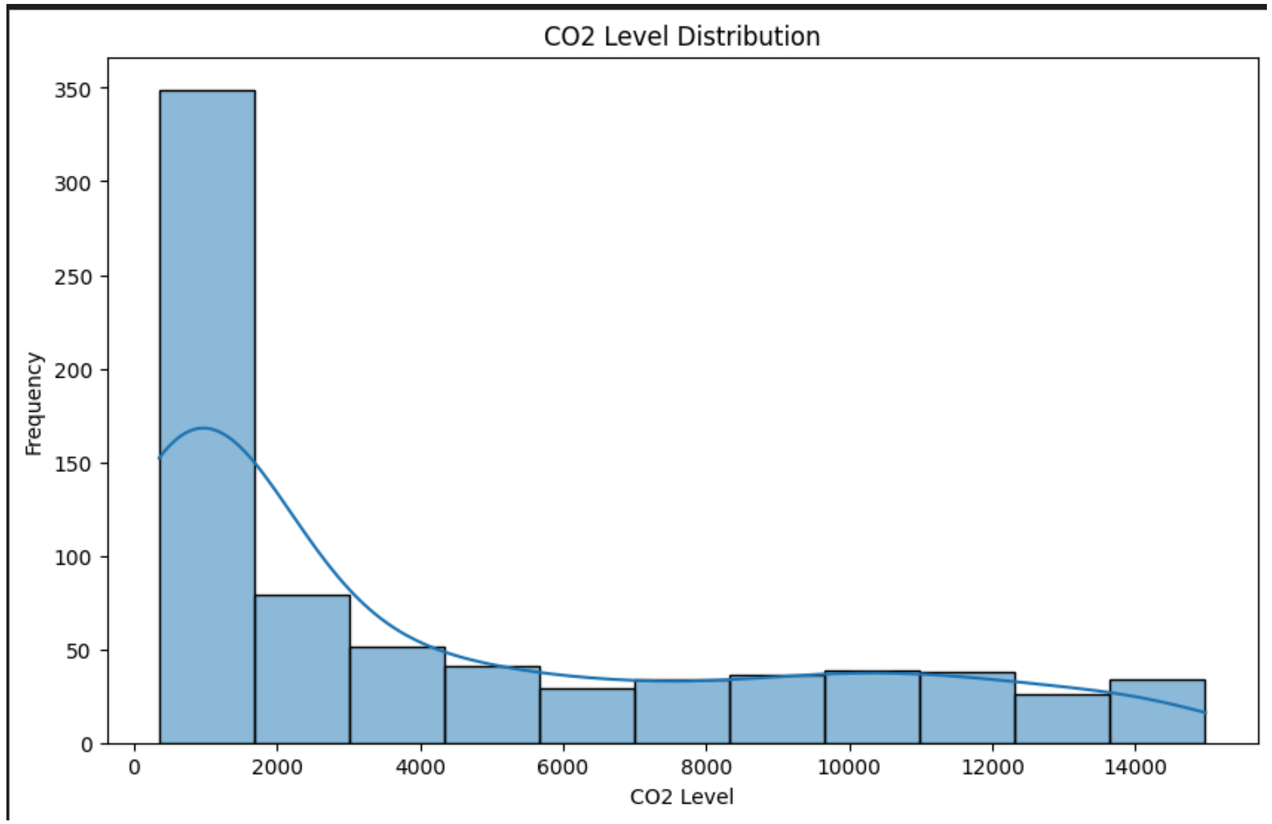
Dataset

```
df.head()
```

	CO_2 Level (ppm)	Effect	CO (ppm)	PM2.5 (µg/m³)	Methane (ppm)	Nitrogen Oxides (ppm)	Temperature (°C)	Humidity (%)	VOC (ppb)
0	428.292361	Natural outdoor levels; healthy	0.442655	9.852667	0.411164	0.252945	18.862547	51.285714	136.734052
1	402.594658	Natural outdoor levels; healthy	0.808799	8.617683	0.309702	0.368291	15.388789	48.325684	193.341313
2	422.353457	Natural outdoor levels; healthy	0.577187	11.477201	0.111720	0.347241	20.190177	56.372490	135.942383
3	358.147390	Natural outdoor levels; healthy	0.193339	14.507447	0.263480	0.286673	11.054196	41.126987	171.718145
4	403.809466	Natural outdoor levels; healthy	0.508706	7.431245	0.397900	0.167295	13.798695	53.821365	128.627986

	CO_2 Level (ppm)	CO (ppm)	PM2.5 (µg/m³)	Methane (ppm)	Nitrogen Oxides (ppm)	Temperature (°C)	Humidity (%)	VOC (ppb)
count	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000
mean	4318.878142	20.218587	115.674460	2.763502	2.609249	29.399536	65.429501	1296.967274
std	4453.732135	27.712495	132.947541	2.563694	2.598757	9.399094	17.369522	1395.675772
min	350.916555	0.009098	5.073578	0.100096	0.115011	10.073126	30.070424	100.586473
25%	696.999875	2.016757	19.972546	0.706212	0.518957	23.088303	51.836831	302.911251
50%	2012.374035	5.008412	49.447253	1.991510	1.503020	27.181613	65.055533	599.249772
75%	7551.899555	30.963709	183.789846	4.014190	4.061178	34.952487	78.502773	2062.829435
max	14974.070333	99.930571	498.343197	9.960192	9.992801	49.987500	99.868261	4998.670246

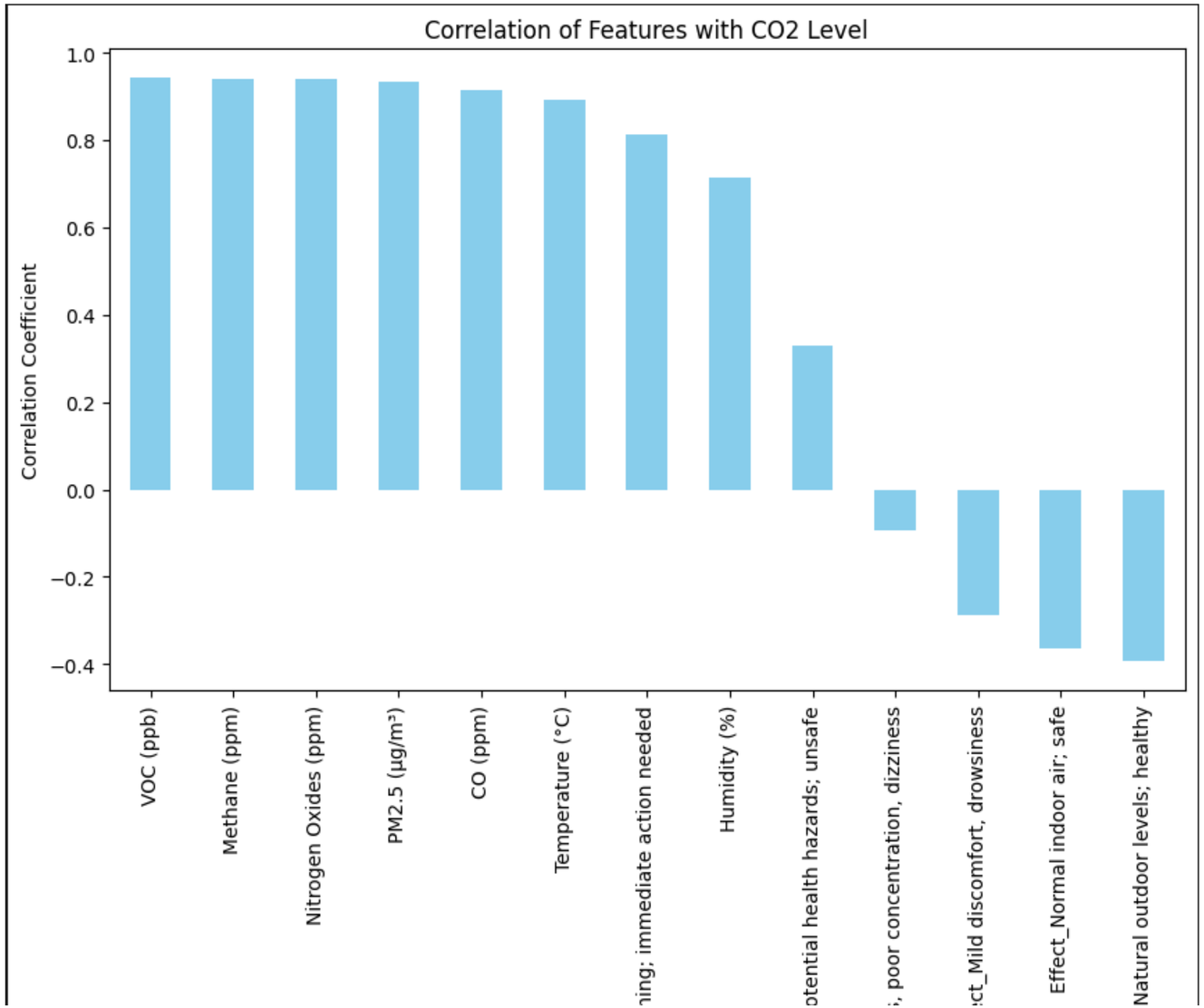
Dataset


CO2 Level Distribution
Frequency
CO2 Level
0
50
100
150
200
250
300
350
0
2000
4000
6000
8000
10000
12000
14000

Data Preprocessing

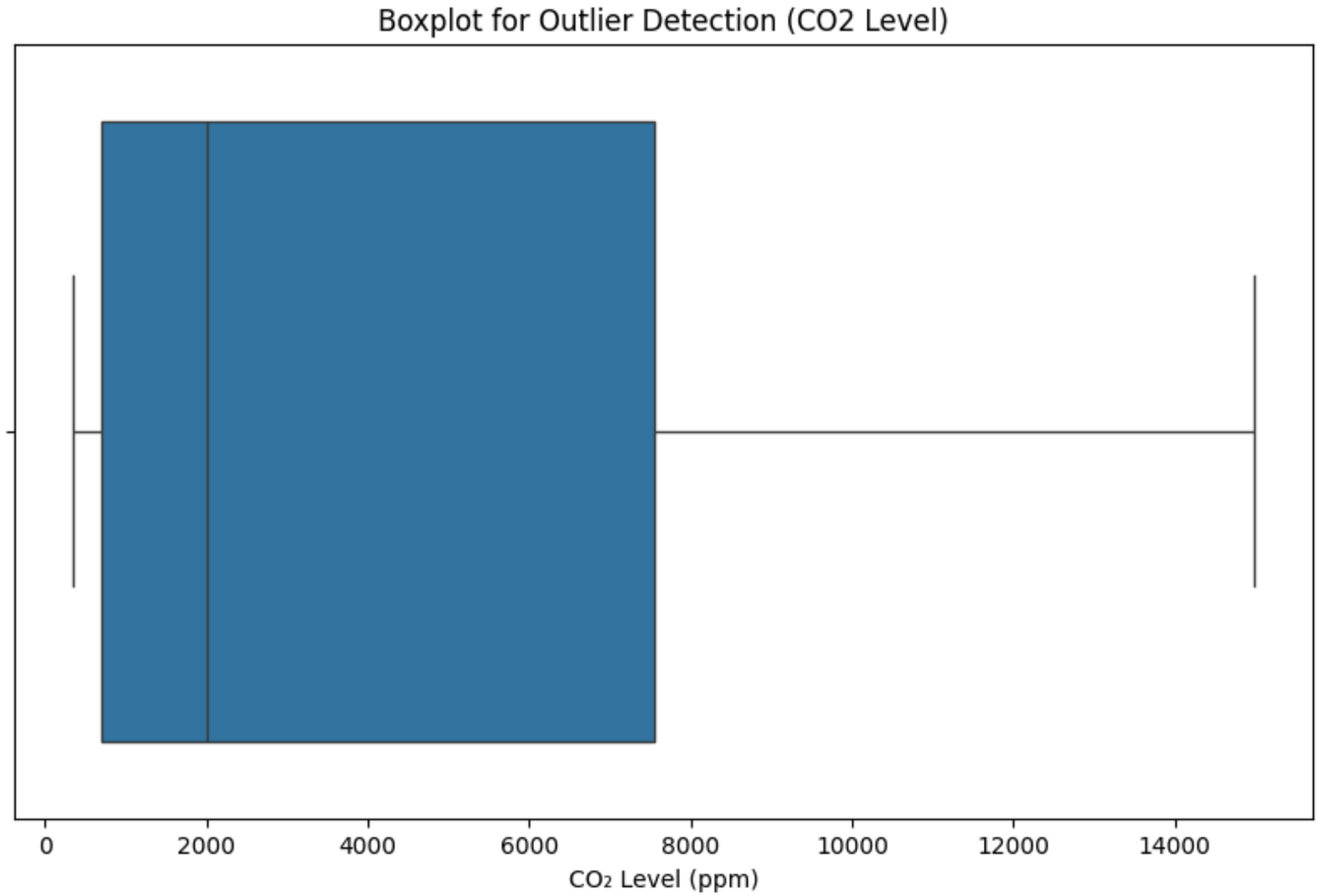
- Check Null values
- Select features using the correaltions
- Remove Outliers
- use feature Scaling (Standard Scaler)
- Split Data Set

Data Preprocessing

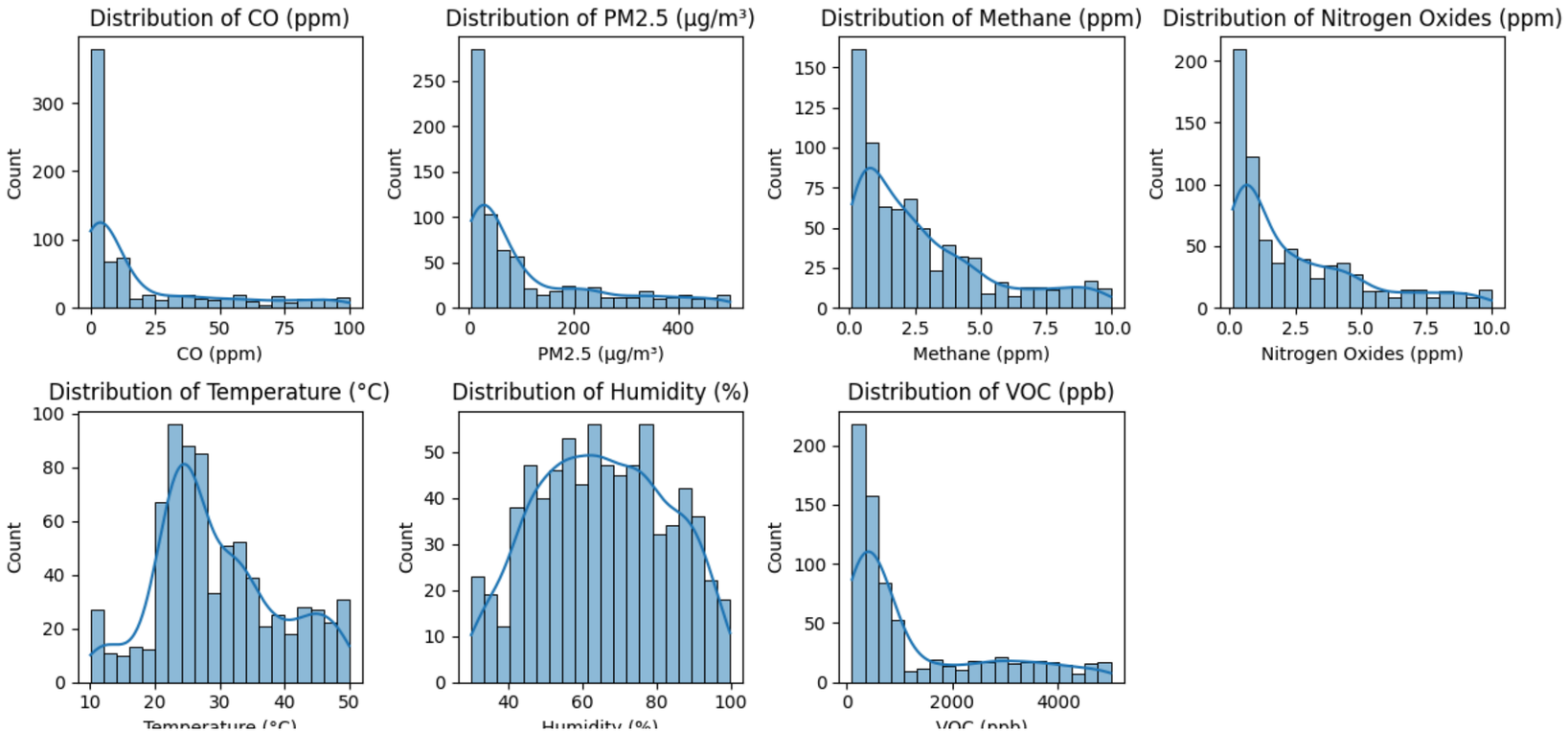


IT21164330 | Arandara S.D. | 24-25J-224

Data Preprocessing

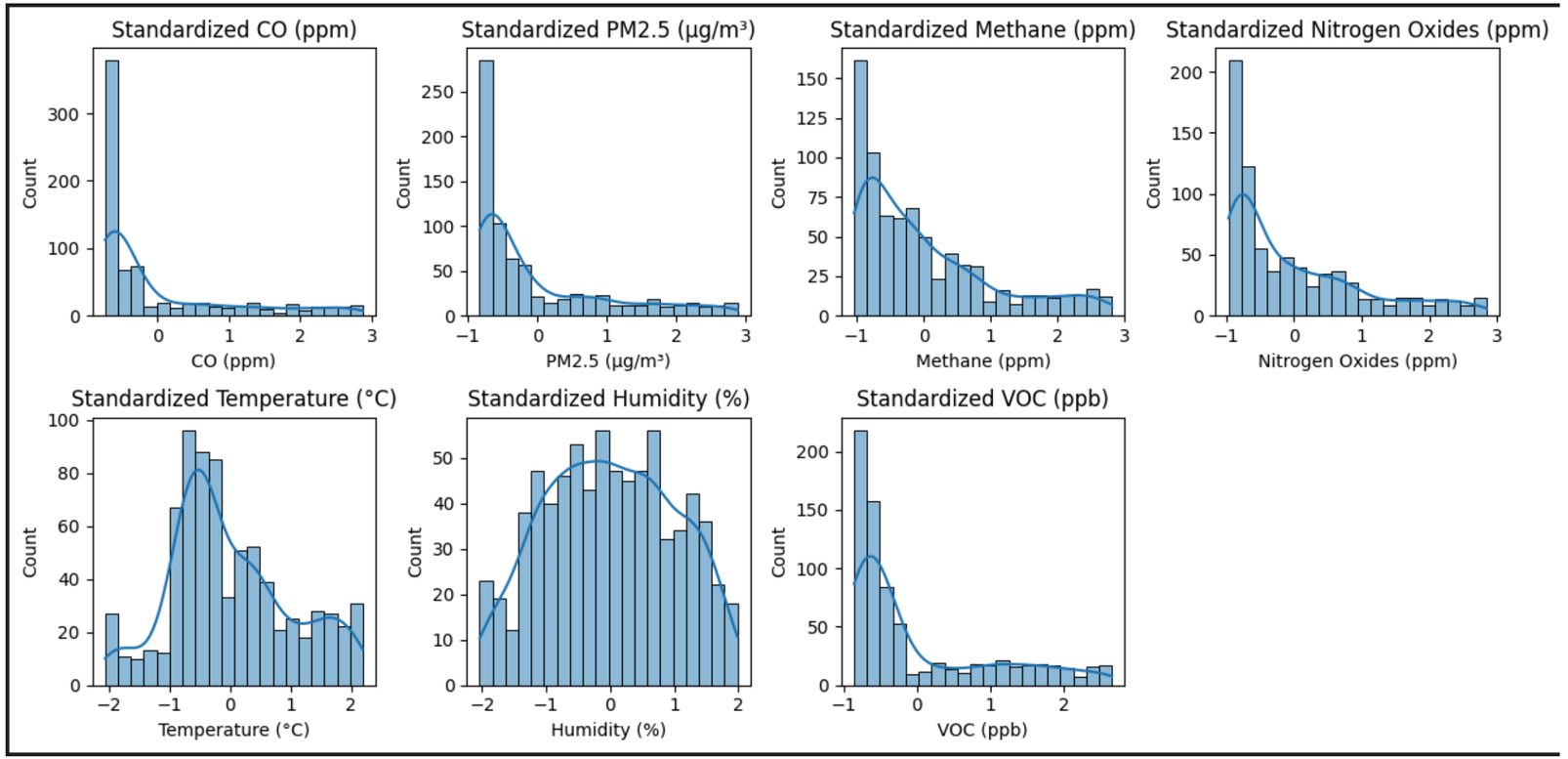


Data Preprocessing



Before Feature Scaling

Data Preprocessing



After Feature Scaling

Model Architecture

Algorithms Used :

- Random Forest
- SVM
- Linear Regression

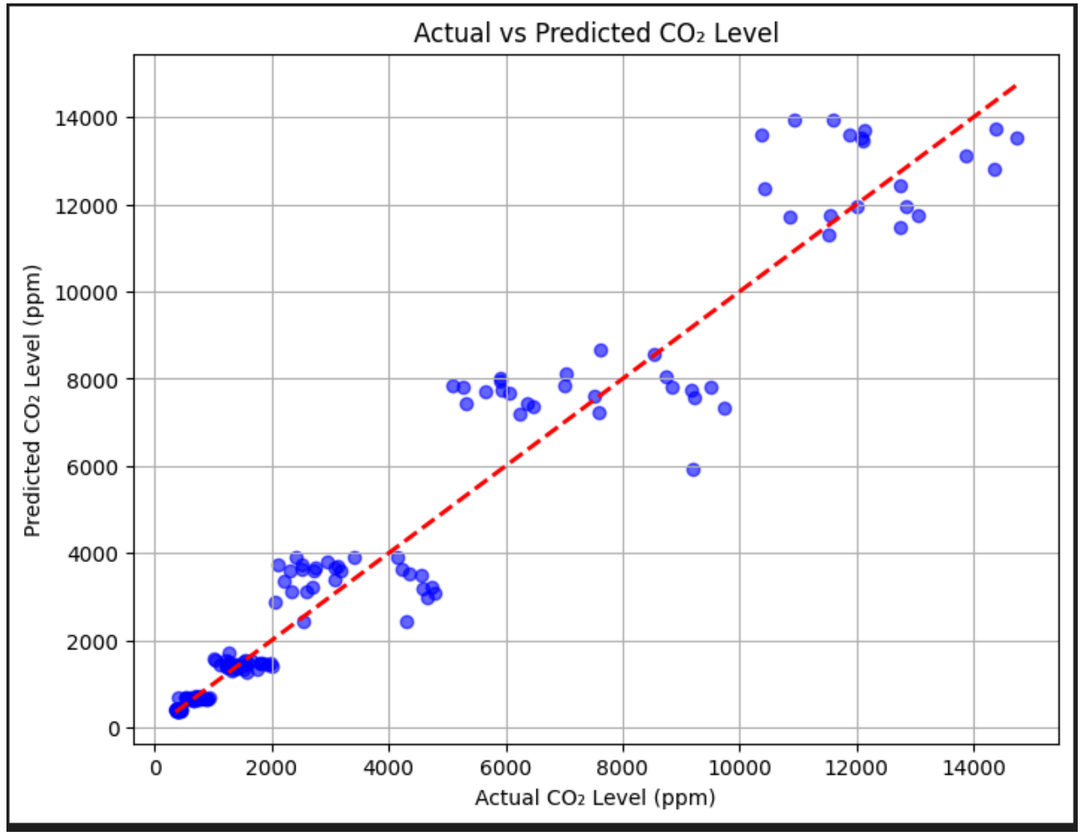
- Select Best Accurate Model by comparing the testing and Training Accuracies
- Use The Grid Search CV to hyper tune the parameter

IT21164330 | Arandara S.D. | 24-25J-224

Accuracy Details

```python
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

best_model_tuned = RandomForestRegressor(
    max_depth=10,
    min_samples_leaf=6,
    min_samples_split=10,
    n_estimators=50
)

best_model_tuned.fit(X_train, y_train)

y_pred_tuned = best_model_tuned.predict(X_test)

rmse_tuned = np.sqrt(mean_squared_error(y_test, y_pred_tuned))

r2_tuned = r2_score(y_test, y_pred_tuned)

print(f"RMSE (Tuned Model): {rmse_tuned}")
print(f"R² Score (Tuned Model): {r2_tuned}")
```

```
RMSE (Tuned Model): 972.0629663492767
R² Score (Tuned Model): 0.9437284285231736
```



IT21164330 | Arandara S.D. | 24-25J-224

Example Usage

```
feature_names = ['CO (ppm)', 'PM2.5 (µg/m³)', 'Methane (ppm)', 'Nitrogen Oxides (ppm)',
                 'Temperature (°C)', 'Humidity (%)', 'VOC (ppb)']


sample_json = """
{
    "CO (ppm)": 0.998560806585628,
    "PM2.5 (µg/m³)": 8.567704903,
    "Methane (ppm)": 0.16570818,
    "Nitrogen Oxides (ppm)": 0.301487774,
    "Temperature (°C)": 26.60405769,
    "Humidity (%)": 59.92036684,
    "VOC (ppb)": 133.8367866
}

"""
input_data = json.loads(sample_json)

input_df = pd.DataFrame([input_data])


scaler = StandardScaler()
scaler = joblib.load('scaler.pkl')

scaled_input = scaler.transform(input_df)

predicted_co2 = best_model.predict(scaled_input)

print(f"Predicted CO₂ Level (ppm): {predicted_co2[0]}")
```

```
Predicted CO₂ Level (ppm): 399.22701656251974
```

Technologies to be Used

Machine Learning:

- Supervised Models such as Random Forest, SVM , Linear Regression

IoT Sensors:

- Raspberry Pi , MQ-2 , e.t.c

Data Visualization:

- D3.js, Chart.js

Programing languages :

- python , MERN stack , Flask

Requirements

System Requirements

- Hardware: IoT sensors for CO2 and PM2.5 data collection, high-performance computing for model training.
- Software: Machine learning frameworks (TensorFlow, PyTorch), data visualization tools.

Software Requirements

- **Functional Requirements**
  - Air quality monitoring
  - Data analysis and predictions
  - Recommondations for good air quality
- **Non-Functional Requirements**
  - Performance
  - Scalability
  - Reliability
  - Usability

References

[1] V. Tikkiwal et al., "An IoT Based Air Pollution Monitoring System for Smart Cities," ResearchGate, 2019. [Online]. Link

[2] A. Gunathilaka et al., "Real-Time Air Quality Monitoring System using IoT," Journal of Physics: Conference Series, 2021. [Online]. Link

[3] P. Wang et al., "Machine learning-based artificial intelligence method for predicting the air pollution index PM2.5" Journal of Cleaner Production, 2024. [Online]. Link

[4] S. Kumar et al., "An Integrated IoT and Machine Learning Approach for Environmental Monitoring and Management," Environmental Research, 2021. [Online] Link

Team Member 3

Noise Guard AI

Karunarathne R. Y. D.
IT21169144

Noise Guard AI

Smart Noise monitoring with IoT sensors,
Classification and Prediction

Background

Urban noise pollution harms public health and quality of life. Effective noise management is crucial for livable cities, yet current methods are inadequate due to fragmented data and limited analysis tools.

Uses AI to Analyze Noise Data for Better Management

- Leverages machine learning to analyze real-time noise levels using IoT data.

Classification and Prediction

- Uses advanced ML and DL models to classify noise sources and predict future trends based on historical data.

Informed Decision-Making

- Identifies high-noise areas and provides real time insights.

Research Problem

Urban areas struggle with managing noise pollution.

Need for a system to analyze and improve noise monitoring using advanced AI and IoT technologies.

IT21169144 | Karunarathne R. Y. D. | 24-25J-224

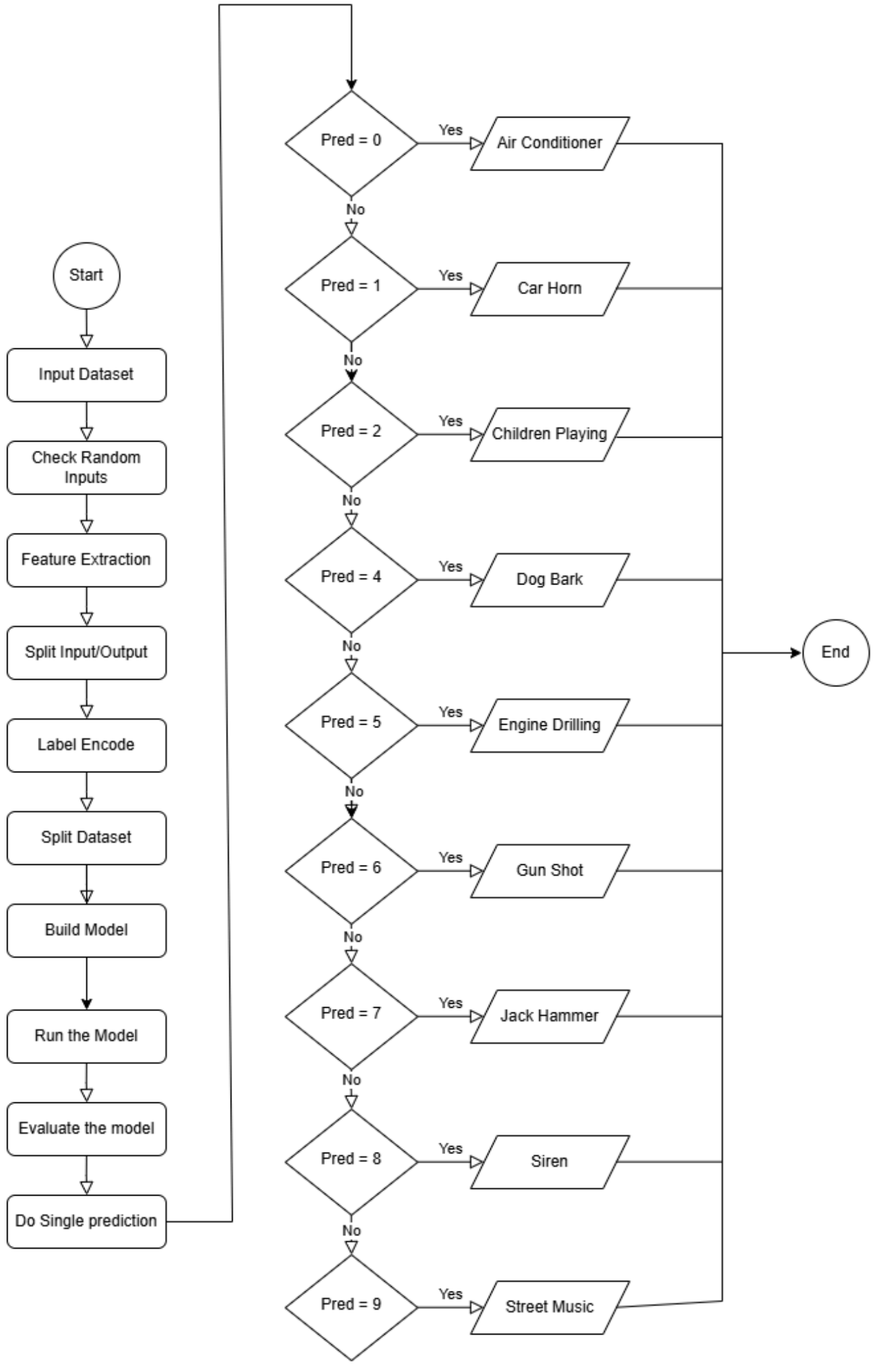
Obejctives

Main Objectives

- Develop a comprehensive system for real-time
  - Noise Level Monitoring
  - Noise Classification
  - Prediction.

Sub-Objectives

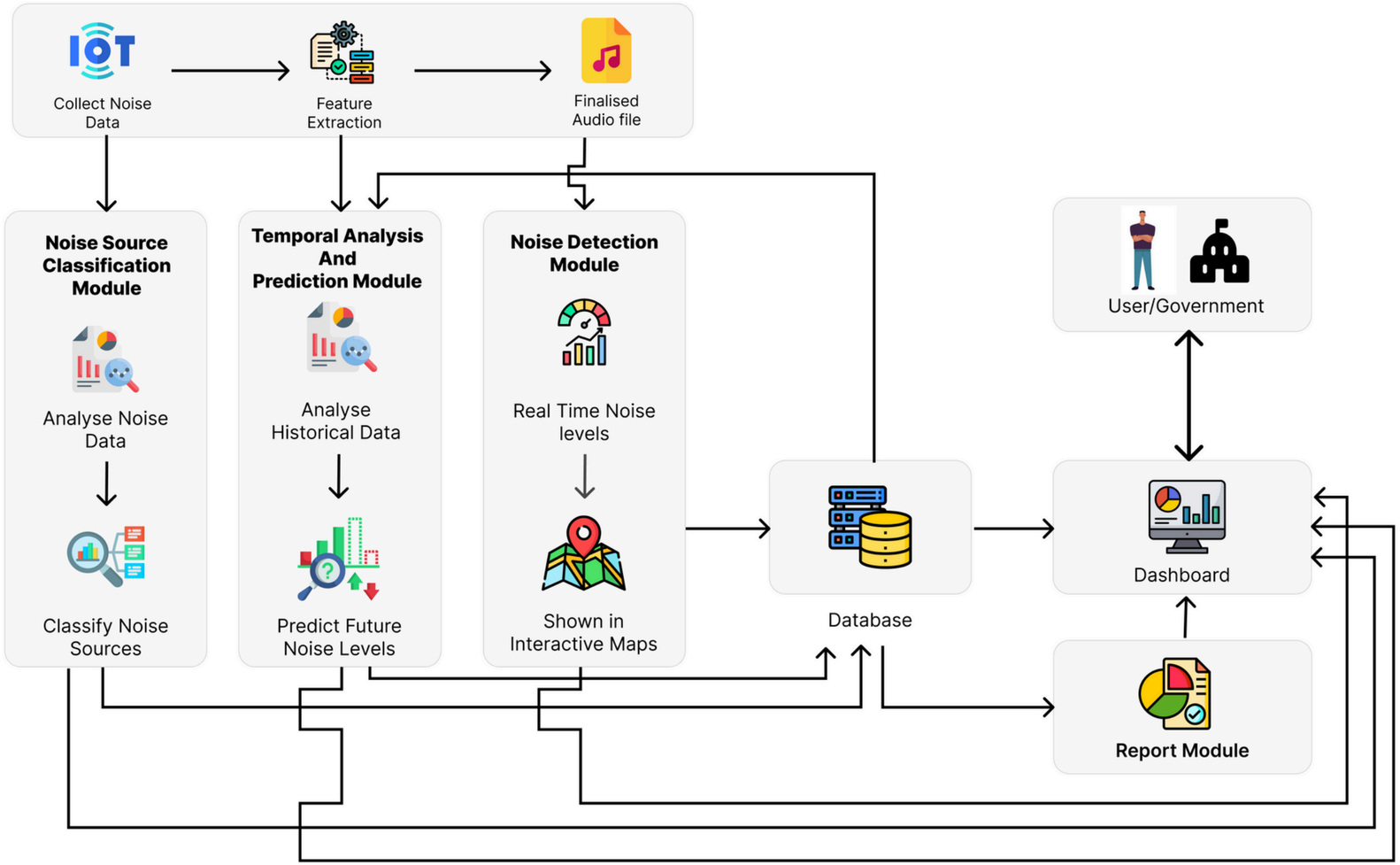
- Collect noise data using IoT and mobile devices.
- Classify noise sources using ML and DL models.
- Predict future noise levels based on historical data.



Flowchart

IT21169144 | Karunarathne R. Y. D. | 24-25J-224

System Diagram



IT21169144 | Karunarathne R. Y. D. | 24-25J-224

Dataset Overview

Audio Classification Data set

Source: Urban 8k - Kaggle .

Key Features:

- 8732 audio
- 10 classes .

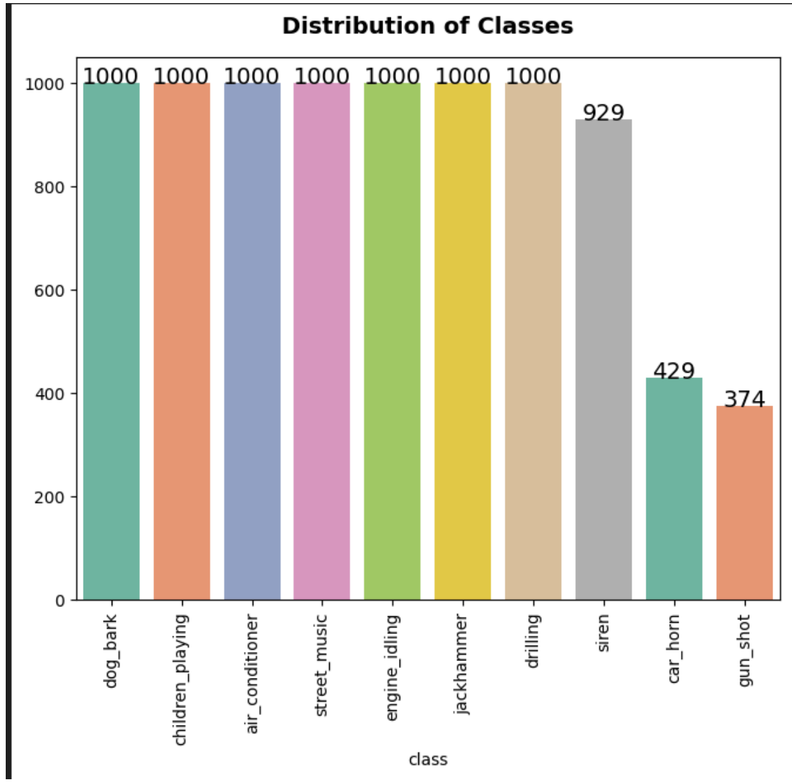
Contains a CSV file

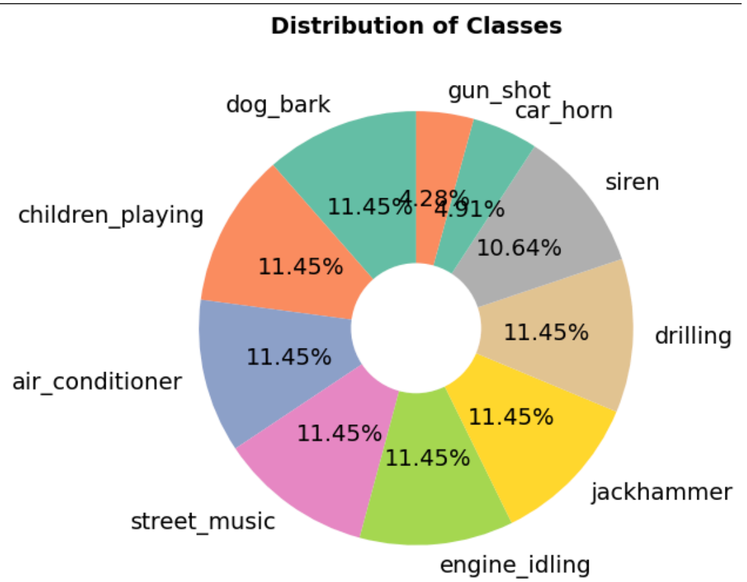
```
metadata.head(10)
```

	slice_file_name	fsID	start	end	salience	fold	classID	class
0	100032-3-0-0.wav	100032	0.000000	0.317551	1	5	3	dog_bark
1	100263-2-0-117.wav	100263	58.500000	62.500000	1	5	2	children_playing
2	100263-2-0-121.wav	100263	60.500000	64.500000	1	5	2	children_playing
3	100263-2-0-126.wav	100263	63.000000	67.000000	1	5	2	children_playing
4	100263-2-0-137.wav	100263	68.500000	72.500000	1	5	2	children_playing
5	100263-2-0-143.wav	100263	71.500000	75.500000	1	5	2	children_playing
6	100263-2-0-161.wav	100263	80.500000	84.500000	1	5	2	children_playing
7	100263-2-0-3.wav	100263	1.500000	5.500000	1	5	2	children_playing
8	100263-2-0-36.wav	100263	18.000000	22.000000	1	5	2	children_playing
9	100648-1-0-0.wav	100648	4.823402	5.471927	2	10	1	car_horn

Head of CSV file

Dataset Overview





IT21169144 | Karunarathne R. Y. D. | 24-25J-224

Data Preprocessing

- Used the **Librosa** library to extract features
- Create new Data frame with features

	features	class
0	[-211.93698, 62.58121, -122.813156, -60.745293...	dog_bark
1	[-417.0052, 99.336624, -42.995583, 51.073326, ...	children_playing
2	[-452.39316, 112.36253, -37.57807, 43.19586, 8...	children_playing
3	[-406.47922, 91.1966, -25.043556, 42.78452, 11...	children_playing
4	[-439.63873, 103.86223, -42.658787, 50.690285,...	children_playing
5	[-441.0859, 108.92061, -25.777704, 47.518364, ...	children_playing
6	[-469.41965, 109.28011, -18.45655, 46.043015, ...	children_playing
7	[-457.1464, 106.52766, -19.124525, 39.87179, 4...	children_playing
8	[-466.2641, 118.13225, -29.881447, 50.494637, ...	children_playing
9	[-188.62564, 102.429115, -2.3306007, -10.94615...	car_horn

Data Preprocessing

- Used the Label Encoding to label the Classes

```
label_mapping = dict(zip(labelencoder.classes_, labelencoder.transform(labelencoder.classes_)))
print("Label Mapping:")
print(label_mapping)
```

```
Label Mapping:
{'air_conditioner': 0, 'car_horn': 1, 'children_playing': 2, 'dog_bark': 3, 'drilling': 4, 'engine_idling': 5, 'gun_shot': 6, 'jackhammer': 7, 'siren': 8, 'street_music': 9}
```

- Split the Data set

Model Architecture

Building the Model

- Keras Sequential Model
- Used Relu Activation Function
- Used the Dense layers
- Used Dropout layer

Tuning Hyperparameters

- Used Random Search to maximize the accuracy

Used the Early Stopping

- Stops training if the validation loss does not improve

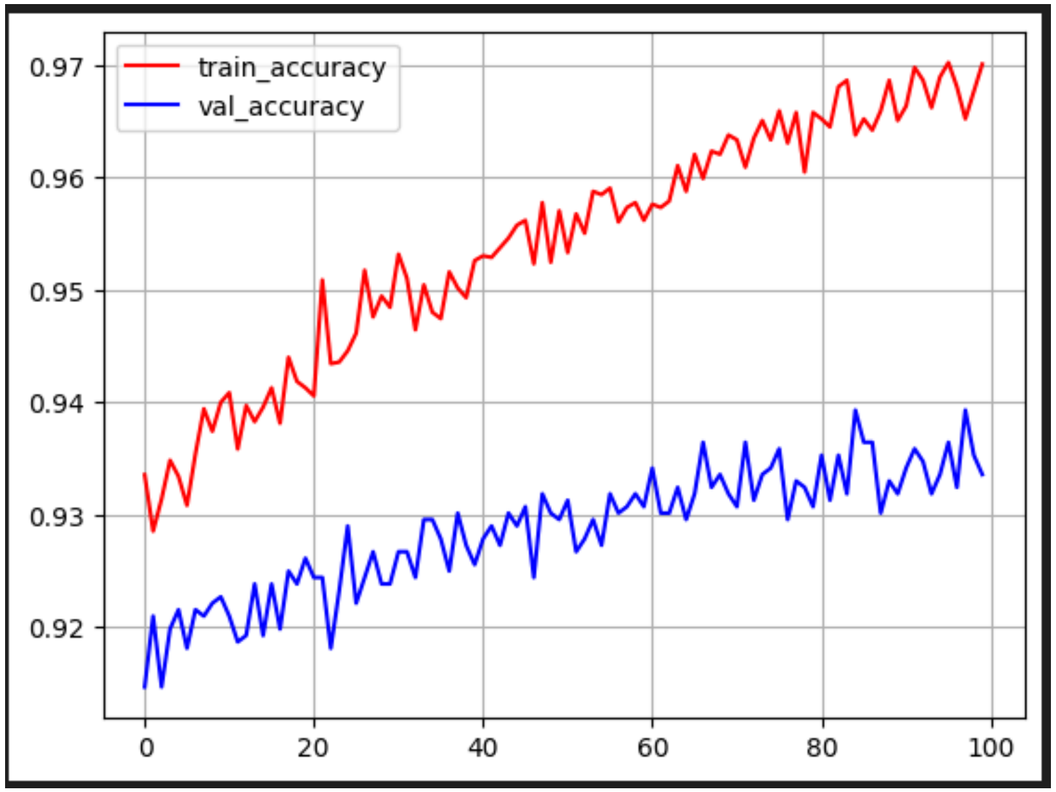
```
[36] model1.summary()

Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 512)	33,280
dropout_3 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 10)	5,130

```
Total params: 38,410 (150.04 KB)

Trainable params: 38,410 (150.04 KB)

Non-trainable params: 0 (0.00 B)
```

IT21169144 | Karunarathne R. Y. D. | 24-25J-224

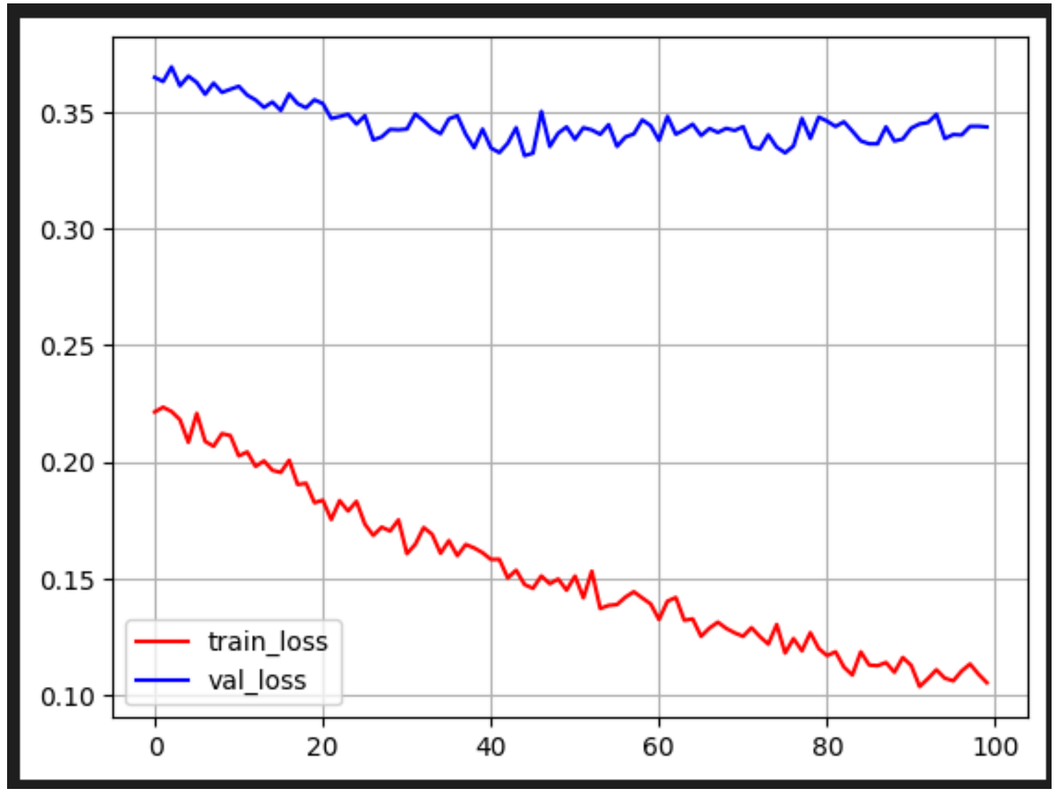
Accuracy Details




```
Epoch 13/100
...
Epoch 99/100
219/219 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9698 - loss: 0.1018 - val_accuracy: 0.9353 - val_loss: 0.3440
Epoch 100/100
219/219 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.9711 - loss: 0.1020 - val_accuracy: 0.9336 - val_loss: 0.3437
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Traing Accuracy and loss in last epochs

Accuracy Details

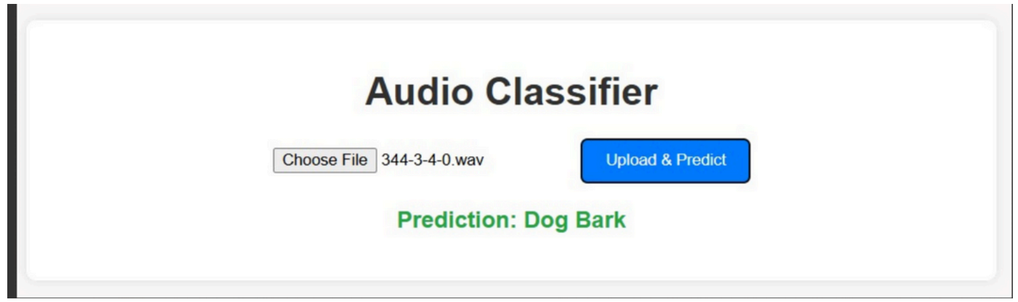


Train Loss and Val loss

Example Usage

```
filename="/kaggle/input/urbansound8k/fold2/106015-5-0-7.wav"
#preprocess the audio file
audio, sample_rate = librosa.load(filename)
mfccs_features = librosa.feature.mfcc(y=audio, sr=sample_rate, n_mfcc=64)
mfccs_scaled_features = np.mean(mfccs_features.T,axis=0)

#Reshape MFCC feature to 2-D array
mfccs_scaled_features=mfccs_scaled_features.reshape(1,-1)

#predicted_label=model.predict_classes(mfccs_scaled_features)
x_predict=model1.predict(mfccs_scaled_features)
predicted_label=np.argmax(x_predict,axis=1)
print(predicted_label)
prediction_class = labelencoder.inverse_transform(predicted_label)
print(prediction_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
[5]
['engine_idling']
```



Technologies to be Used

- **IoT, Mobile phone Microphone**
  - For real-time noise data collection.
- **ML/DL Models**
  - ANN with dense layer
  - Hyper tuners to tune model

1. **Data Collection:** Deploy IoT sensors in various locations and Pre Data-sets
2. **Pre-processing:** Clean and preprocess collected data.
3. **Model Training:** DL models for classification and prediction.
4. **System Integration:** Integrate models with real-time data for monitoring.

IT21169144 | Karunarathne R. Y. D. | 24-25J-224

TOOLS & TECHNOLOGIES

IEMS

System Requirements

System Requirements

- **Hardware**: IoT sensors, mobile phone's microphone for noise data collection, high-performance computing for model training.
- **Software**: TensorFlow, PyTorch, data visualization tools.

Software Requirements

- **Functional Requirements**
  - Noise Data Collection
  - Noise Source Classification
  - Temporal Analysis And Prediction Module
- **Non-Functional Requirements**
  - Performance
  - Scalability
  - Reliability
  - Usability

References

[1] M. A. Basuni and S. H. Alzahrani, "Urban Sound Classification using CNN," ResearchGate, 2021. [Online]. Available: Link

[2] M. F. Khan, M. Al-Kahtani, S. H. Islam, N. Kumar, and P. W. C. Prasad, "A Machine Learning Driven IoT Solution for Noise Classification in Smart Cities," ResearchGate, 2018. [Online]. Available: link

[3] D. Gómez-Gutiérrez, J. Pascual-Gaspar, J. M. Lanza-Gutiérrez, and E. Sancristobal, "Internet of Things for Noise Mapping in Smart Cities: State-of-the-Art and Future Directions," ResearchGate, 2020. [Online]. Available: link

[4] S. A. Kumar, K. R. Sudheesh, and S. M. S. Islam, "Noise Prediction Using Machine Learning with Measurement," ResearchGate, 2020. [Online]. Available: link

Team Member 4

Eco go AI

Kodithuwakku C.K.
IT21156960

Eco go

Predicting and Reducing
CO2 Emissions from Vehicles.

Background

Eco Go, a component of the Integrated Environmental Monitoring System (IEMS), leverages advanced machine learning to predict and reduce vehicle CO2 emissions, offering personalized strategies and insights for urban emission challenges.

Predictive Emission Analysis

- Uses machine learning to predict vehicle CO2 emissions by analyzing vehicle attributes.
- Use two models to predict CO2 emissions accurately.

Personalized Recommendation Engine

- Use TOPSIS algorithm(multi-criteria decision making) to recommend best vehicle

Research Problem

Urban areas experience significant CO2 emissions from vehicles, contributing to climate change and air pollution.

A system is needed that uses advanced machine learning to predict CO2 emissions and offer customized reduction strategies for each driver.

IT21156960 | Kodithuwakku C.K. | 24-25J-224

Obejctives

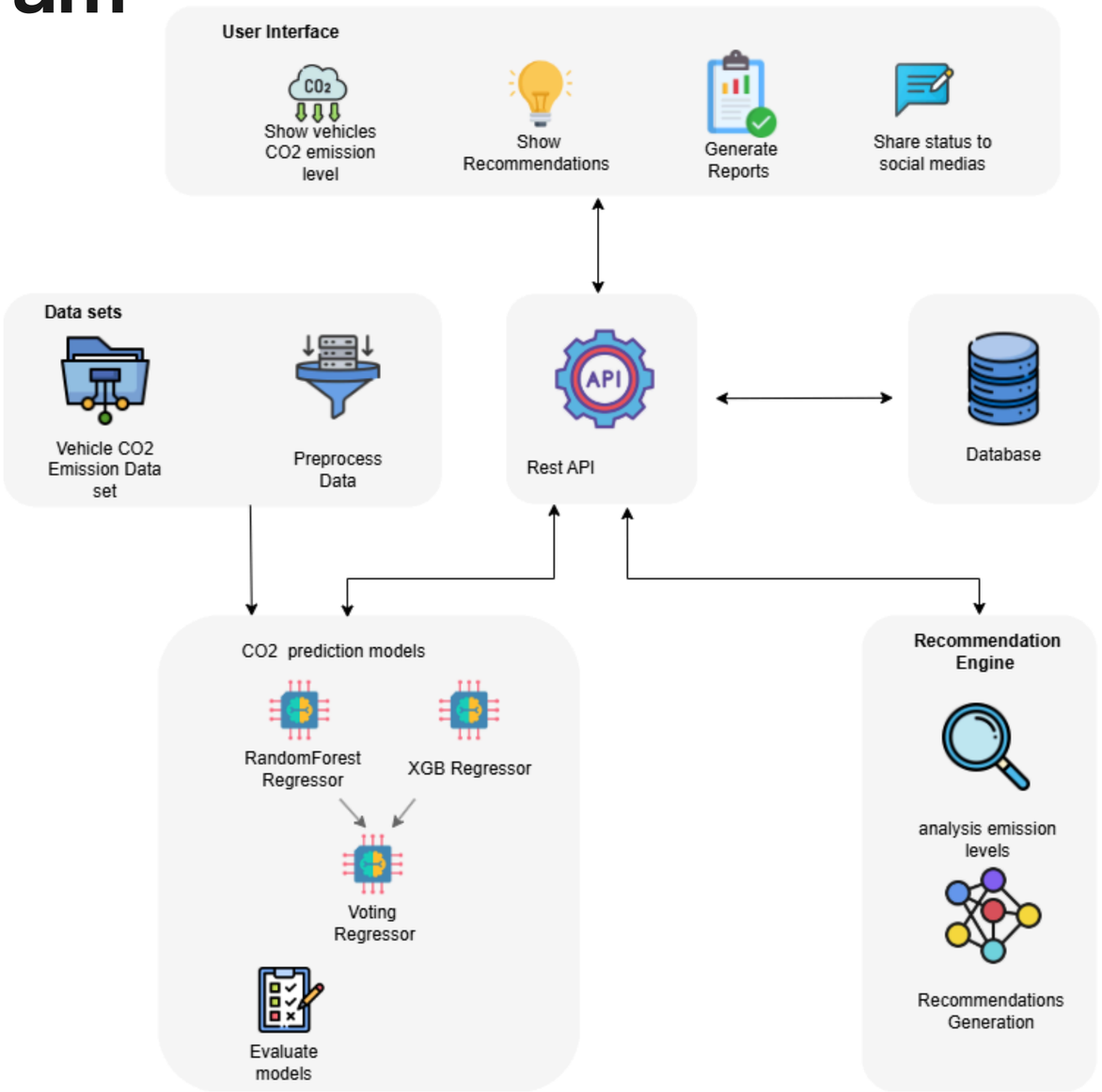
Main Objective

- Improve urban sustainability by predicting vehicle CO2 emissions and delivering personalized recommendations for effective emission reduction.

Sub-Objectives

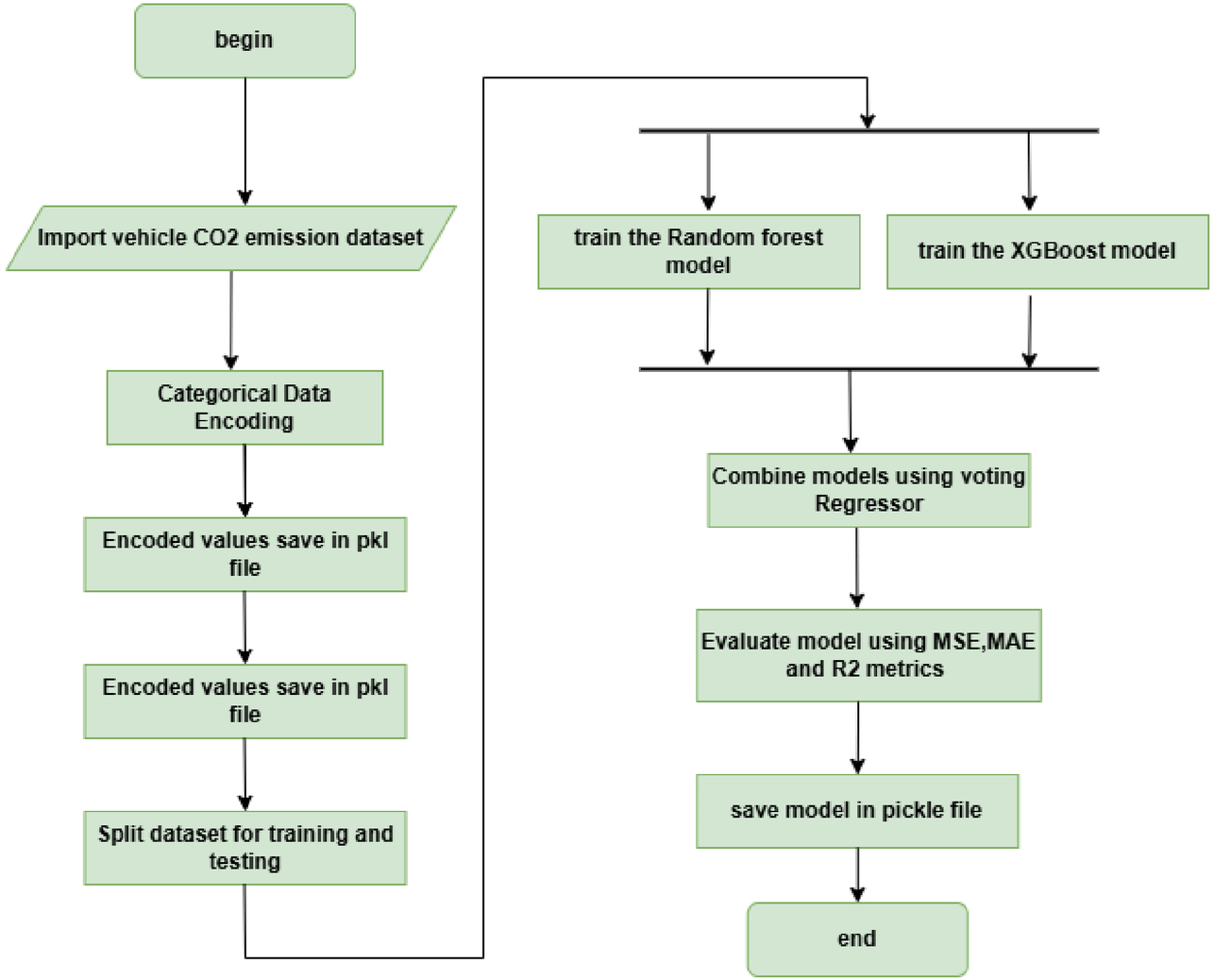
- Predict Vehicle CO2 emission.
- Develop personalized strategies.
- Develop a recommendation engine that suggest best vehicle

System Diagram



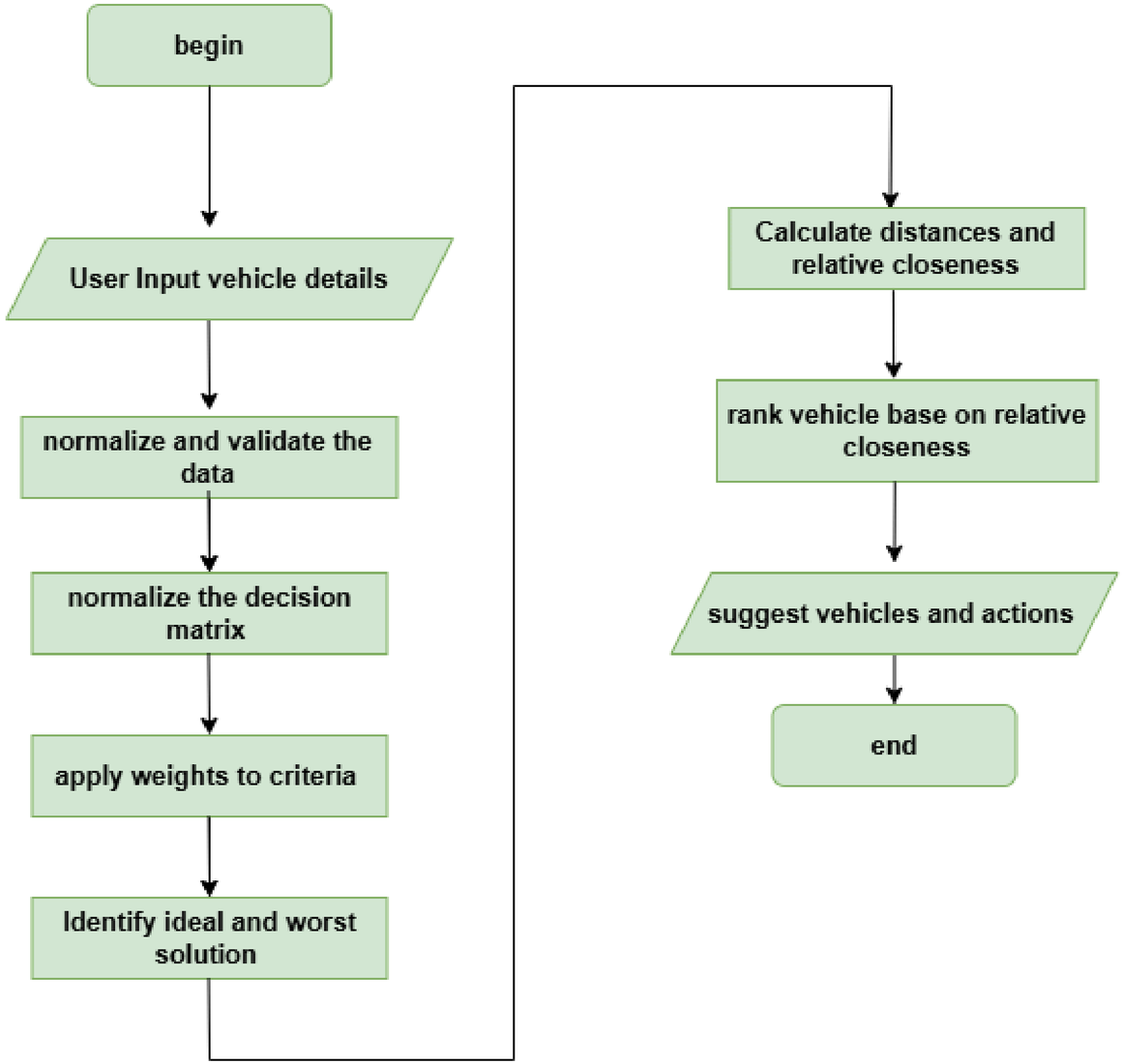
IT21156960 | Kodithuwakku C.K. | 24-25J-224

Flowchart of CO2 prediction model



IT21156960 | Kodithuwakku C.K. | 24-25J-224

Flowchart of recommendation engine



IT21156960 | Kodithuwakku C.K. | 24-25J-224

Dataset

	Manufacturer	Model	Transmission	Vehicle_Type	Engine_Capacity	Fuel_Type	Powertrain	Engine_PowerPS	Engine_Power	Emissions_CO2	THC0Emissions0[mg/km]	Emissions NOx [mg/km]
0	ABARTH	595 MY22	M5	Manual	1368	Petrol	Internal Combustion Engine (ICE)	165	121	688	66.0	28
1	ABARTH	595 MY22	M5	Manual	1368	Petrol	Internal Combustion Engine (ICE)	165	121	688	66.0	28
2	ABARTH	595 MY22	M5	Manual	1368	Petrol	Internal Combustion Engine (ICE)	165	121	688	66.0	28
3	ABARTH	595 MY22	M5	Manual	1368	Petrol	Internal Combustion Engine (ICE)	165	121	688	66.0	28
4	ABARTH	595 MY22	M5	Manual	1368	Petrol	Internal Combustion Engine (ICE)	165	121	688	66.0	28

Data Preprocessing

- Data Loading and Basic Inspection
- Feature Selection
- Handling Missing Values
- Data Encoding for Categorical Variables
- Split Data Set

Model Architecture

- Random Forest model
- XGBoost model
- Voting Regression model

- Input Features - Transmission, Vehicle Type, Engine Capacity, Fuel Type, Powertrain, and Engine PowerPS.
- Combine models two models using voting Regression
- Output final CO_2 emission prediction for the given vehicle

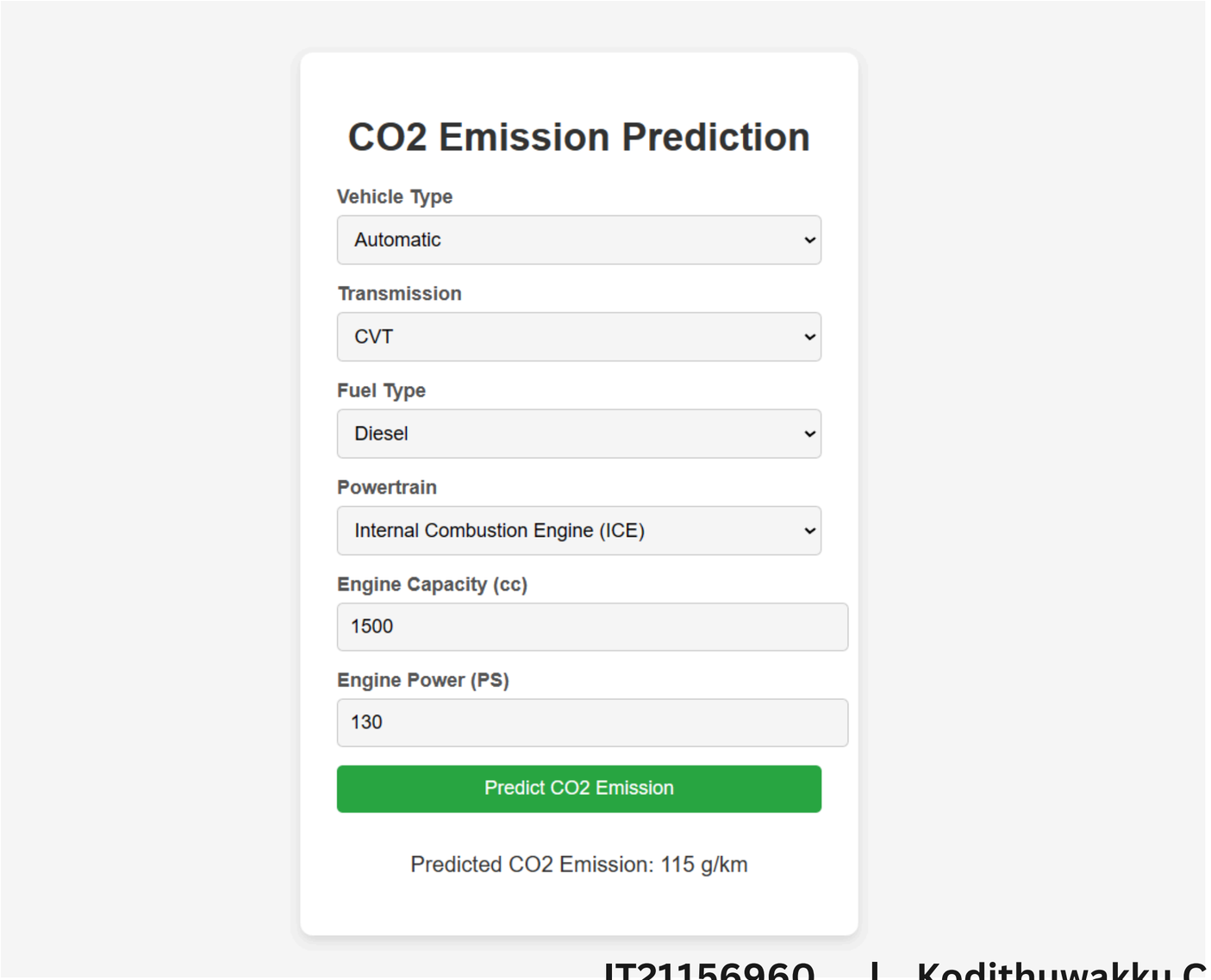
Accuracy Details

```
mse_train = mean_squared_error(Ytrain, Ptrain)
mse_test = mean_squared_error(Ytest, Ptest)

mae_train = mean_absolute_error(Ytrain, Ptrain)
mae_test = mean_absolute_error(Ytest, Ptest)

r2_train = r2_score(Ytrain, Ptrain)
r2_test = r2_score(Ytest, Ptest)

df_result = pd.DataFrame({
                          "Train": [mse_train, mae_train, r2_train],
                          "Test": [mse_test, mae_test, r2_test]
                          }, index=["MSE", "MAE", "R2"])
print(df_result)
```

[21]

```
            Train         Test
MSE  3546.556667  4410.599049
MAE    34.385795    37.518633
R2      0.904137     0.876075
```

Example usage



Technologies

Machine Learning

- Supervised Models such as Random Forest, Gradient Boosting
- Ridge Regression, OLS Regression, Lasso Regression
- Metrics such as RMSE,R2

Data Visualization

- D3.js, Chart.js

Programing languages

- python , MERN stack , Flask

TOOLS & TECHNOLOGIES

IEMS

IT21156960 | Kodithuwakku C.K. | 24-25J-224

Requirements

IEMS

System Requirements

- High-Performance Computing

Software Requirements

- **Functional Requirements**
  - CO2 Prediction Model
  - Recommendation Engine
  - Visualization and Reporting
- **Non-Functional Requirements**
  - Performance
  - Scalability
  - Reliability
  - Usability

References

[1] Navarajan Subramaniam, Norhakim Yusof, "Modelling of CO2 Emission Prediction for Dynamic Vehicle Travel Behavior Using Ensemble Machine Learning Technique," IEEE, 25 November 2021. [Online]. Available: link . (Accessed: Jul. 22, 2024.)

[2] S. Tsiakmakis, G. Fontaras, J. Dornoff, and V. Valverde, "From lab-to-road & vice-versa: Using a simulation-based approach for predicting real-world CO2 emissions," Energy, vol. 177, pp. 495–508, 15 February 2019. [Online]. Available: link . (Accessed: Jul. 22, 2024.)

[3] S. Li, Z. Tong, and M. Haroon, "Estimation of transport CO2 emissions using machine learning algorithm," Transportation Research Part D: Transport and Environment, vol. 122, August 2024. [Online]. Available: link. (Accessed: Jul. 23, 2024.)

[4] H. Zhong, K. Chen, C. Liu, and M. Zhu, "Models for predicting vehicle emissions: A comprehensive review," Science of The Total Environment, vol. 791, 1 May 2024. [Online]. Available: link. (Accessed: Jul. 23, 2024.)

Commercialization

Market Demand

- Market demand analysis involves understanding the need for the Intelligent EcoUrban Monitoring System (IEMS) in the market. This includes identifying the target audience, market size, growth trends, and potential market opportunities.

Target Audience

- Municipalities and city planners ,Environmental agencies , Goverment , Citizens

Market Opportunities

- Integration with existing smart city infrastructure. ,Partnership with governments for large-scale implementation , Custom solutions for different urban environments.

24-25J-224

THANK YOU !

SLIIT UNI
THE KNOWLEDGE UNIVERSITY

24-25J-224

Q & A

SLIIT UNI
THE KNOWLEDGE UNIVERSITY