University of Science and Technology Chittagong (USTC)
Faculty of Science, Engineering & Technology
Department of Computer Science & Engineering

Course code: CSE 324

Course Title: Artificial Intelligence and Expert Systems Lab

Project title:

AI Smart Bin - Intelligent Waste Management Solution

By

Team Elite

Submitted by:

Name	ID
Abhilash Chowdhury (L)	22010110
Tanmay Das	22010105
Tazrian Binte Mahfuz	22010116
Pritom Gupta	22010134

Semester: 6th
Batch: 38th
Dept.: CSE, FSET

Submitted to:

Debabarata Mallick
[Lecturer]
CSE, FSET, USTC

Problem Statement

With increasing environmental concerns about waste management, improper segregation of waste into organic and inorganic types continues to be a critical issue. Traditional methods of waste classification rely on manual sorting, which is time-consuming and inefficient. The need for an automated solution to effectively classify waste at the disposal level is essential to improving recycling processes and waste treatment. This project addresses the challenge by developing a smart dustbin that uses sensor data and machine learning to automatically classify waste as organic or inorganic.

Abstract

The Smart Bin AI is designed to automate waste classification into organic and inorganic categories based on sensor input. The system utilizes an ultrasonic sensor to detect when waste is placed in the bin, a soil moisture sensor to determine moisture levels (indicating organic waste), and an infrared (IR) sensor for object detection. A machine learning model, specifically a RandomForestClassifier, is trained using moisture level data to classify waste in real time. The waste type and moisture information are logged into a CSV file for monitoring and analysis. The system includes real-time feedback mechanisms through a buzzer and LED indicators. The project presents a solution to improve waste segregation at the source, thereby promoting efficient waste management practices.

Features

Automatic Waste Detection: The system opens the dustbin door automatically when an object is detected within a certain range (via ultrasonic sensor).

Waste Type Classification: The AI model classifies the waste as organic or inorganic based on real-time moisture sensor data.

Moisture-Level-Based Classification: Moisture levels are read and used as the primary feature to distinguish organic (high moisture) from inorganic (low/no moisture) waste.

Real-Time Data Logging: Waste type predictions, along with moisture levels and status checks, are logged with timestamps into a CSV file for tracking and analysis.

User Feedback Mechanism: A buzzer sounds if moisture (organic waste) is detected, and an LED indicates dry waste.

System Reset: After waste is placed and classified, the system resets itself, ready to detect and classify new waste.

Methodology/Architecture

The Smart Bin AI is built using the following methodology -

Hardware Components:

Ultrasonic Sensor: Detects the proximity of waste within 30 cm of the dustbin.

Servo Motor: Opens the dustbin lid when waste is detected and closes it after waste is removed.

Soil Moisture Sensor: Measures the moisture content of the waste, serving as the primary feature for classification.

IR Sensor: Detects the presence of waste in the bin and signals when the object is removed.

Buzzer and LED: Provide real-time feedback based on the moisture level and waste classification.

Software Components:

Machine Learning Model: A RandomForestClassifier is trained with labeled moisture level data to classify waste into organic or inorganic categories.

Arduino Communication: Data is transmitted from the Arduino sensors to a Python script via a serial connection.

Data Logging: A CSV file stores waste type predictions, moisture levels, and timestamped records.

Workflow:

Waste Detection: The ultrasonic sensor detects an object and opens the dustbin using the servo motor.

Moisture and IR Detection: The soil moisture sensor reads the moisture content of the waste, while the IR sensor confirms the presence of waste in the bin.

AI-Based Classification: The RandomForestClassifier predicts the waste type based on the moisture level.

Data Logging: The system logs the predicted waste type, moisture level, and match status into a CSV file.

System Reset: After detecting waste removal, the system resets, closing the dustbin and preparing for the next detection.

Implementation of Arduino Code

◆ Why we use Arduino Code in our Project?

The Arduino code is essential for automating and controlling the smart bin ai by integrating sensors (ultrasonic, IR, and soil moisture) and actuators like the servo motor and buzzer. It enables the dustbin to detect objects, identify waste types (organic or inorganic), and operate the lid autonomously. The code processes real-time sensor inputs, making decisions like opening the lid when waste is detected and closing it once the object is removed. It also communicates with a PC to send waste data for further analysis. Overall, the Arduino code makes the dustbin efficient, user-friendly, and capable of functioning without manual intervention.

◆ How it works?

STEP 1: Object Detection (Ultrasonic Sensor)

The ultrasonic sensor measures the distance to objects in front of the dustbin. If an object is detected within 30 cm, the system considers that something is approaching the bin.

```
// Step 1: Check if an object is detected within 30 cm
if (distance > 0 && distance <= 30) {
  if (!objectDetected) {
    objectDetected = true;  // Object detected

    Serial.print("Object detected within 30 cm! Distance: ");
    Serial.print(distance);
    Serial.println(" cm");
```

Step 2: Opening the Dustbin (Servo Motor)

Once the ultrasonic sensor detects an object, the servo motor moves the dustbin door to the open position (90°). This allows the user to place waste into the bin.

```
    // Step 2: Open the dustbin
    myServo.write(90);  // Door open position
    doorOpen = true;    // Mark door as open
    wastePlaced = false;  // Reset waste placed status
    moistureDetected = false;  // Reset moisture detected status
    irWasteDetected = false;  // Reset IR sensor detection status
    wasteTypeSent = false;    // Reset the flag to allow new waste type detection
    Serial.println("Dustbin door opened. Please place the waste.");
  }
}
```

Step 3: Waste Detection (IR Sensor and Soil Moisture Sensor)

After the door is open, the IR sensor monitors for waste. Once waste is placed, it checks for the presence of moisture using the soil moisture sensor. If moisture is detected, the system assumes the waste is organic.

```
// Step 3: Check the IR sensor and soil moisture sensor only if the door is open
if (doorOpen) {
  irSensorValue = digitalRead(irSensorPin);  // Read IR sensor value
```

Step 4: Moisture Detection and Buzzer/LED Alerts

If moisture is detected, the system classifies the waste as organic, turns on the buzzer, and logs the moisture level. If no moisture is found, the waste is considered dry (inorganic), and the LED is turned on to indicate dry waste.

```
    // Step 4: Check if the waste is placed (IR sensor triggered)
    if (irSensorValue == LOW && !irWasteDetected) {
      irWasteDetected = true;  // Waste detected by IR sensor
      Serial.println("Waste detected by IR sensor. Waiting for moisture check.");

      // Start checking moisture sensor
      unsigned long wastePlacedTime = millis();  // Record the time when waste is placed

      // Monitor for 10 seconds or until moisture is detected
      while (millis() - wastePlacedTime < 5000) {
        sensorValue = analogRead(sensorPin);  // Read soil moisture sensor value

        // Condition: If moisture is detected (sensor value < dry air reading)
        if (sensorValue < 900) {
          moistureLevel = map(sensorValue, 1023, 0, 0, 100);
          moistureDetected = true;  // Mark that moisture is detected
          Serial.print("Moisture Level: ");
          Serial.println(moistureLevel);

          if (moistureLevel > 0) {
            Serial.println("Moisture detected! Keeping buzzer on.");
            digitalWrite(buzzerPin, HIGH);  // Turn on buzzer if moisture is detected
            break;  // Exit the 10-second countdown early if moisture is detected
          }
        }
      }

      // If no moisture is detected after 10 seconds, the IR sensor gives output
      if (!moistureDetected) {
        Serial.println("No moisture detected within 10 seconds. Waste is dry.");
        digitalWrite(ledPin, HIGH);  // Turn on LED to indicate waste is dry (or perform other action)
      }
    }
  }

// Waste type detection and sending data to PC
if (!wasteTypeSent) {  // Check if waste type has already been sent
  bool isOrganic = detectOrganicWaste();  // Replace with actual logic
  bool isInorganic = detectInorganicWaste();  // Replace with actual logic
```

```
    // If organic waste is detected, send organic data to the PC with moisture level
    if (isOrganic) {
      sendToPC(true, moistureLevel);  // Sends "1,0,moistureLevel"
      wasteTypeSent = true;  // Mark waste type as sent
    }
    // If inorganic waste is detected, send inorganic data to the PC with moisture level as 0
    else if (isInorganic) {
      sendToPC(false);  // Sends "0,1,0"
      wasteTypeSent = true;  // Mark waste type as sent
    }
  }
```

Step 5: Waste Removal Detection (IR and Soil Moisture Sensor)

After the waste is placed, the system continuously monitors both the IR and soil moisture sensors. If the waste is removed from the bin (IR sensor is high, and moisture sensor detects no moisture), the system resets the states.

```
// Step 5: Check if waste is removed from the moisture sensor and IR sensor
if (irWasteDetected || moistureDetected) {
  sensorValue = analogRead(sensorPin);  // Continuously read soil moisture sensor
  irSensorValue = digitalRead(irSensorPin);  // Continuously read IR sensor

  // Condition: If waste is removed from both sensors (no moisture and IR sensor HIGH)
  if (sensorValue >= 900 && irSensorValue == HIGH) {
    Serial.println("Waste removed from both IR and moisture sensors. Resetting system.");
    digitalWrite(buzzerPin, LOW);  // Turn off the buzzer
    digitalWrite(ledPin, LOW);     // Turn off the LED
    moistureDetected = false;      // Reset moisture detected flag
    irWasteDetected = false;       // Reset IR waste detected flag
    wasteTypeSent = false;         // Reset waste type sent flag
  }
}
```

Step 6: Object Removal from Detection Range

After placing the waste, if the object (e.g., user's hand) is removed from the detection range of the ultrasonic sensor, a 10-second countdown begins to close the dustbin.

```
// Step 6: Object is removed from detection range
if (objectDetected && distance > 30) {
  if (!objectRemoved) {
    objectRemoved = true;           // Object was just removed
    objectRemovedTime = millis();   // Store the time when object is removed
    Serial.println("Object removed, starting 10-second timer to close dustbin.");
  }
}
```

Step 7: Closing the Dustbin (Servo Motor)

After 10 seconds without any new objects being detected, the servo motor moves the door to the closed position (0°), sealing the bin. If a new object is detected within this time, the countdown resets to keep the door open.

```
// Step 7: Close the door 10 seconds after the object is removed, only if no new object is detected
if (objectRemoved) {
  // Recheck the ultrasonic sensor within the 10-second countdown
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;

  // If a new object is detected, reset the countdown and keep the door open
  if (distance > 0 && distance <= 30) {
    Serial.println("New object detected, resetting 10-second timer.");
    objectRemoved = false;  // Reset removal flag to keep the door open
  }
  else if (millis() - objectRemovedTime >= 10000) {
    Serial.println("10 seconds passed, closing dustbin.");
    myServo.write(0);  // Close the dustbin
    doorOpen = false;  // Mark door as closed
    objectDetected = false;  // Reset object detected state
    objectRemoved = false;   // Reset removal flag
    wasteTypeSent = false;   // Reset the flag to allow new waste type detection
    digitalWrite(buzzerPin, LOW);  // Ensure the buzzer is off
    digitalWrite(ledPin, LOW);  // Turn off the LED
  }
}

  delay(100);  // Small delay between readings
}
```

Implementation of Random Forest

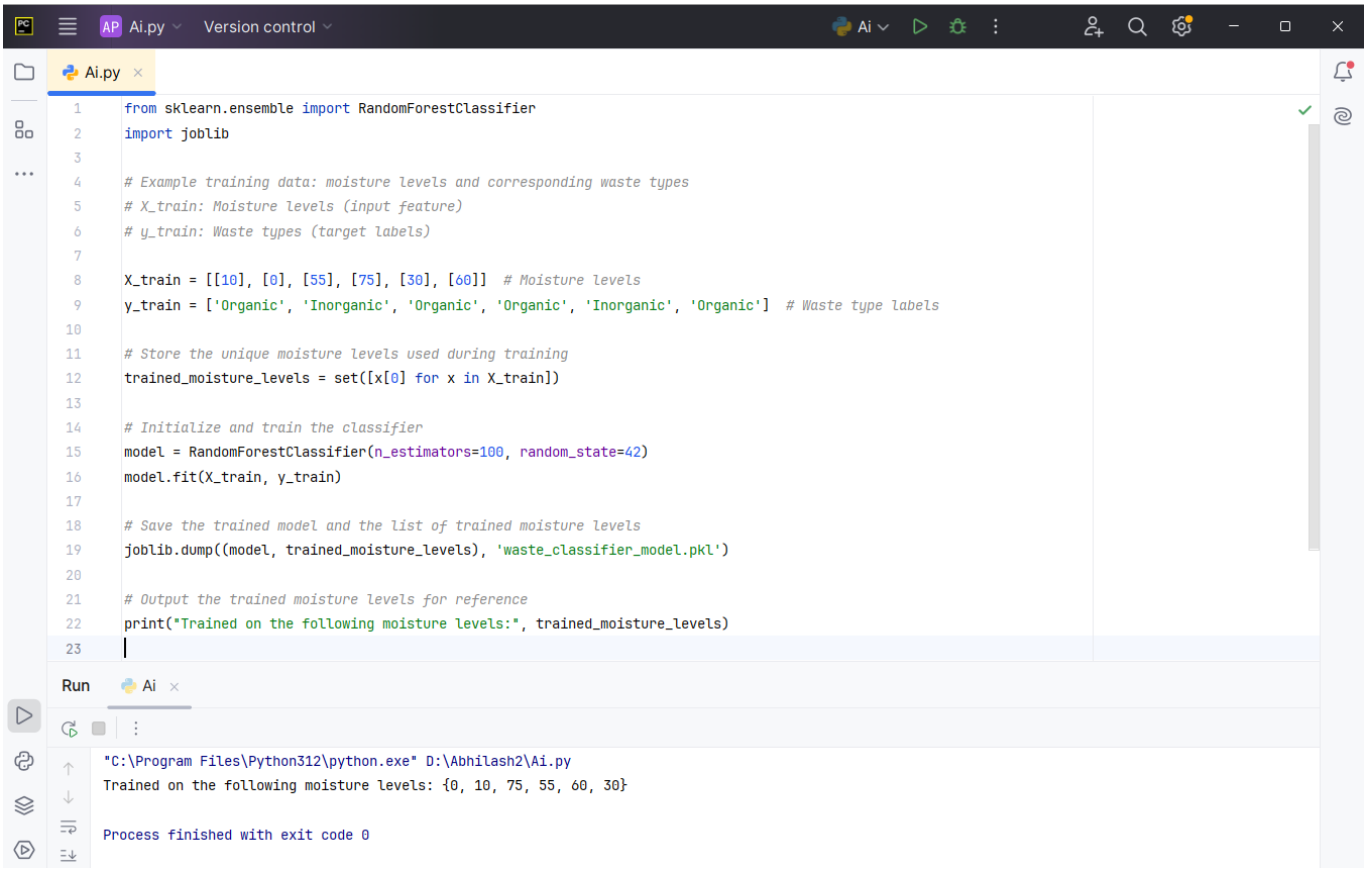
◆ Why we use Random Forest in our Project?

- Random Forest is an ensemble machine learning algorithm that builds multiple decision trees during training.
- Each decision tree makes a prediction, and the majority vote (or average in regression tasks) across all trees is taken as the final output.
- For our project, the input is the moisture level of the waste, and the output is whether the waste is classified as organic or inorganic.

◆ How it works?

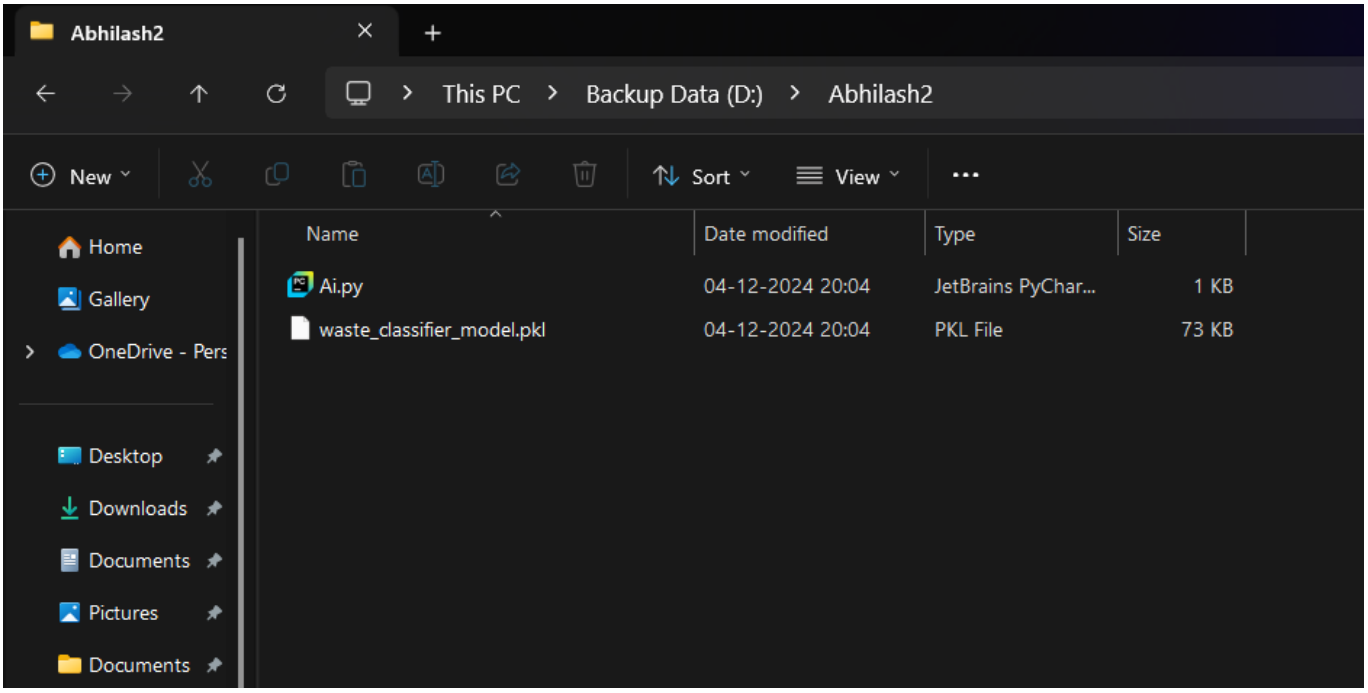
Training the Model:

- We have training data consisting of moisture levels and corresponding waste types. These values are used to train the model.
- The moisture levels (X_train) are the input features, and the waste types (y_train) are the target labels.
- A Random Forest Classifier is trained on this data to predict the waste type based on new moisture levels.



Saving the Model:

- After training, the model is saved using joblib so that it can be loaded and used later to classify new moisture levels in real-time as the Arduino provides them.



Implementation of Machine Learning

◆ Why we use ML in our Project?

We use Machine Learning (ML) in our smart dustbin project to enhance the efficiency and accuracy of waste classification. By leveraging ML, the system can analyze data from various sensors, such as the soil moisture sensor and IR sensor, to identify patterns and classify waste more intelligently. An ML model can learn to distinguish between organic and inorganic waste based on sensor readings and other inputs, improving detection accuracy.

The implementation of ML allows the dustbin to adapt and make more informed decisions, enabling continuous improvement in waste identification. This helps in better sorting of materials, leading to more effective waste management, reduced environmental impact, and optimized recycling processes. By utilizing ML, our project moves toward a more advanced and responsive waste management system.

◆ How it works?

Data Preprocessing:

- The Arduino sends data in the format moisture_level.
- This data is read via the serial connection and split into individual components.

```
11
12    # Set up the serial connection (ensure the correct COM port is used)
13    arduino = serial.Serial('COM3', 9600)  # Change COM3 to the correct port for your Arduino
14
```

- For our ML model, the primary input is likely the moisture_level, which the model will use to predict whether the waste is organic or inorganic.

Using the Pre-trained ML Model:

- You have loaded a pre-trained model (waste_classifier_model.pkl) using joblib.load. This model takes moisture levels as input and predicts the waste type (organic or inorganic).

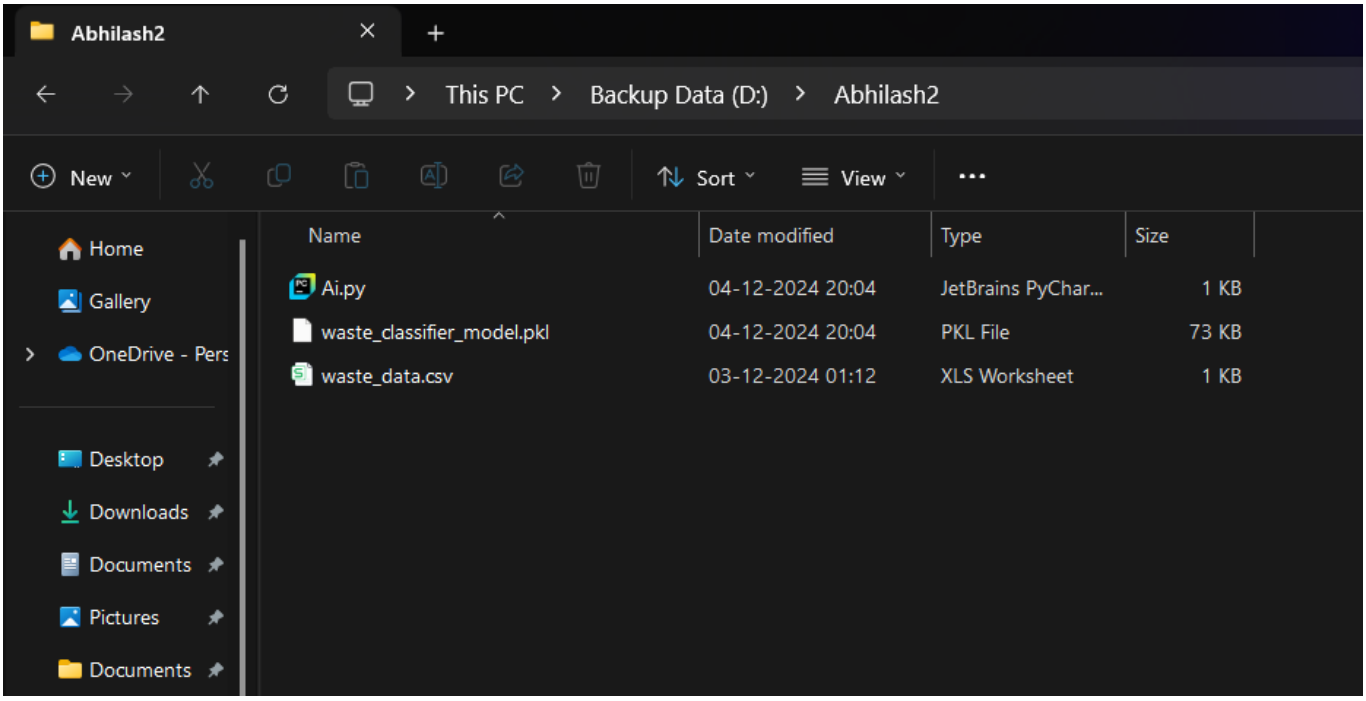
```
8
9     # Load pre-trained ML model (replace with your model file path)
10    model, trained_moisture_levels = joblib.load('waste_classifier_model.pkl')
11
```

- We will pass the moisture level to the model and let it predict the waste type.

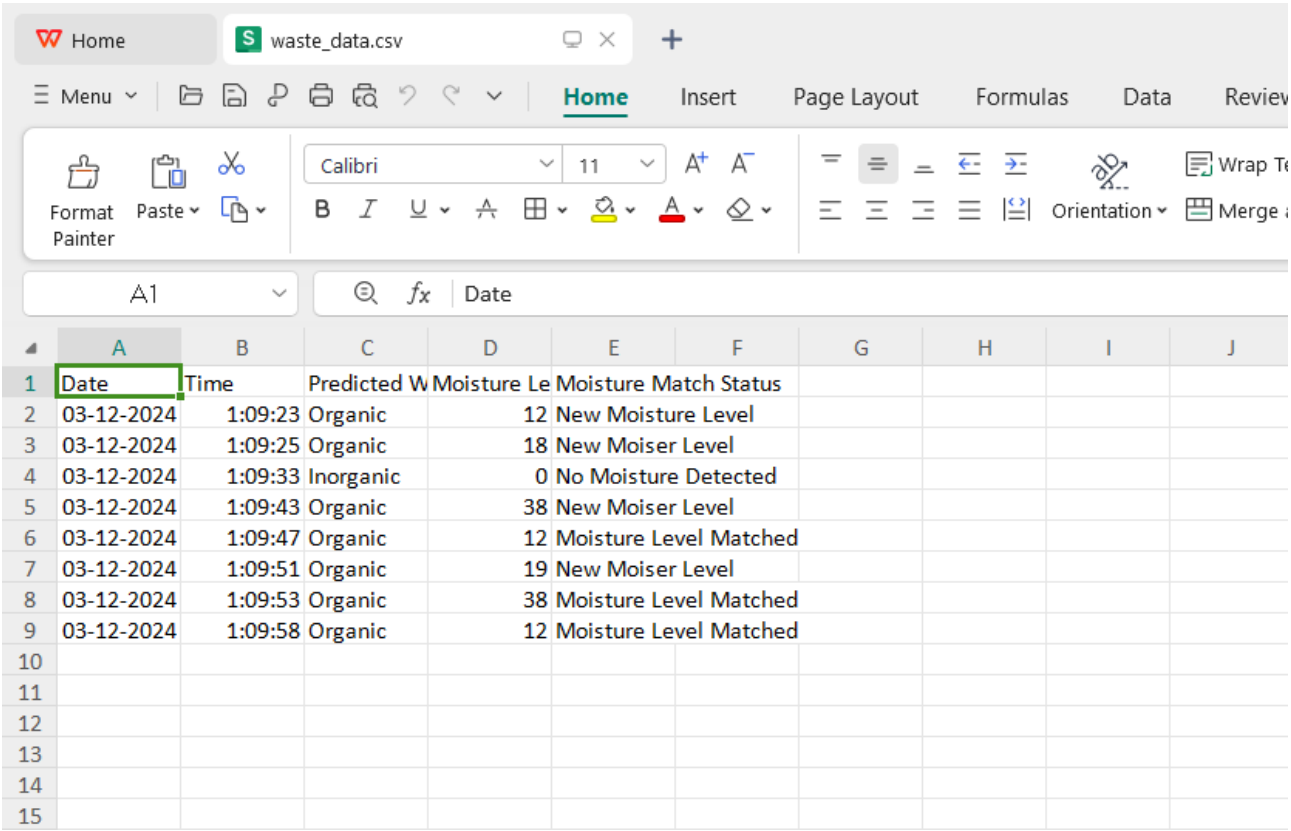
```
42
43      while True:
44          try:
45              # Read a line from the serial input
46              line = arduino.readline().decode('utf-8').strip()
47              print(f"Received: {line}")  # Debugging print to see what is received
48
49              # Check if the line contains the expected comma-separated values
50              if ',' in line:
51                  try:
52                      # Split the line into organic, inorganic, and moisture values
53                      organic, inorganic, moisture_level = line.split(',')
54
55                      # Ensure that moisture level is properly converted to an integer
56                      moisture_level = int(moisture_level)
57
58                      # Use the ML model to predict waste type (organic/inorganic)
59                      features = np.array([[moisture_level]])
60                      predicted_waste_type = model.predict(features)[0]  # 'Organic' or 'Inorganic'
61
62                      # Get the current date and time
63                      current_date = time.strftime('%Y-%m-%d')  # Extract date
64                      current_time = time.strftime('%H:%M:%S')  # Extract time
65
66                      # Determine the moisture match status using pandas DataFrame
67                      moisture_status = check_moisture_status(moisture_level, data)
68
69                      # Log the new data entry in the CSV file
70                      csvwriter.writerow([current_date, current_time, predicted_waste_type, moisture_level, moisture_status])
71
72                      # Update the DataFrame with the new entry
73                      new_row = {
74                          'Date': current_date,
75                          'Time': current_time,
76                          'Predicted Waste Type': predicted_waste_type,
77                          'Moisture Level': moisture_level,
78                          'Moisture Match Status': moisture_status
79                      }
80                      data = pd.concat([data, pd.DataFrame([new_row])], ignore_index=True)
81
82                      # Flush the file to ensure data is written immediately
83                      csvfile.flush()
84
85                      # Display the logged information in the console
86                      print(f"Logged: {current_date} {current_time}, Predicted Waste Type: {predicted_waste_type}, "
87                            f"Moisture Level: {moisture_level}, Status: {moisture_status}")
88
89                  except ValueError:
90                      # Handle cases where the data cannot be split or converted correctly
91                      print(f"Error: Unable to process the line: {line}")
92              else:
93                  print(f"Unexpected data format: {line}")
94
95          except Exception as e:
96              print(f"Error: {e}")
```

Logging Data:

- After predicting the waste type using the model, the predicted type, along with the moisture level, will be logged into a CSV file.



- The system also checks if the current moisture level matches any previously recorded levels (to track if the same waste type has been classified before).



	A	B	C	D	E	F	G	H	I	J
1	Date	Time	Predicted W	Moisture Le	Moisture Match Status					
2	03-12-2024	1:09:23	Organic	12	New Moisture Level					
3	03-12-2024	1:09:25	Organic	18	New Moiser Level					
4	03-12-2024	1:09:33	Inorganic	0	No Moisture Detected					
5	03-12-2024	1:09:43	Organic	38	New Moiser Level					
6	03-12-2024	1:09:47	Organic	12	Moisture Level Matched					
7	03-12-2024	1:09:51	Organic	19	New Moiser Level					
8	03-12-2024	1:09:53	Organic	38	Moisture Level Matched					
9	03-12-2024	1:09:58	Organic	12	Moisture Level Matched					
10										
11										
12										
13										
14										
15										

Summary

The Smart Bin AI successfully integrates machine learning with embedded sensors to automate waste classification at the point of disposal. The system uses real-time sensor data, particularly moisture levels, to distinguish between organic and inorganic waste, with feedback provided to the user via buzzer and LED indicators. By logging data into a CSV file, the system allows for performance tracking and data analysis. The project demonstrates an effective approach to improving waste segregation, contributing to more efficient waste management and recycling process.

References

1. Smart Dustbin by using Arduino and ultrasonic sensor (Youtube) - https://www.youtube.com/watch?v=9yrP1CZN3Ds
2. How to make Wet Dry Metal waste segregation project || Smart dustbin project using Arduino (Youtube) - https://youtu.be/4XedfXtPxLQ?si=2F1v6EaSIa1xGRud
3. Smart Bin (Arduino project Hub) – https://projecthub.arduino.cc/mohammadsohail0008/how-to-make-smart-dustbin-for-your-home-2fd84f
4. Ultrasonic Sensor (Last Minute Engineers) – https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/
5. Soil Moisture Sensor (Circuit Digest) - https://circuitdigest.com/microcontroller-projects/interfacing-soil-moisture-sensor-with-arduino-uno
6. IR Sensor (Circuit Digest) - https://circuitdigest.com/microcontroller-projects/interfacing-ir-sensor-module -with-arduino
7. Servo Motor (Last Minute Engineers) - https://lastminuteengineers.com/servo-motor-arduino-tutorial/
8. “A Brief Study on Random Forest Using Python” Madhav, Shubham Kohli, Himanshu Rawat, Priyanshu Joshi (Department of Information Technology), Dr. Akhilesh Das Gupta (Institute of Technology and Management, New Delhi) - https://ijaem.net/issue_dcp/A%20Brief%20Study%20on%20Random%20Forest%20Using%20Python.pdf

◆ GitHub Link - https://github.com/T-a-n-m-a-y-D-a-s-TD/Project--AI-SmartBin_2024