Artificial Intelligence and Machine Learning

Project Documentation

1. Introduction

• **Project Title: Electric Motor Temperature Prediction using Machine Learning**

• **Team Members:** Usha Kiran Illa
B. Sravya
D. Srinu
Anusuri Satish

2. Project Overview

• **Purpose:** The purpose of this project is to predict the operating temperature of an electric motor using machine learning techniques. Accurate temperature prediction helps in preventing motor overheating, reducing maintenance costs, and improving operational efficiency.

• **Features:**

- Predicts electric motor temperature using machine learning models
- Supports both manual input and sensor-based input
- Web-based interface for easy interaction
- Real-time prediction results
- Simple and user-friendly design

3. Architecture

• Frontend: The frontend is developed using HTML and CSS and is integrated with the Flask backend. It provides pages for manual input prediction and sensor-based prediction, allowing users to submit motor parameters and view predicted temperature outputs.

• Backend: The backend is implemented using Python Flask. It handles HTTP requests, manages routing between pages, processes user inputs, loads trained machine learning models, and returns prediction results to the frontend.

• Machine Learning Model: The system uses trained machine learning models developed using scikit-learn. Input data is preprocessed using a saved scaler, and predictions are generated using serialized models stored as .pkl files.

4. Setup Instructions

· Prerequisites:

- Python 3.x
- Flask
- scikit-learn
- NumPy
- Joblib

• Installation:

1. Download or clone the project repository.
2. Install required Python libraries using pip.
3. Ensure trained model and scaler files (.pkl) are present in the project directory.
4. Run the Flask application using the command: python app.py

5. Folder Structure

Application Files:

- app.py – Main Flask application
- model files (.pkl) – Trained machine learning models
- scaler file – Preprocessing scaler

Templates Folder:

- index.html – Home page
- manual.html – Manual input prediction page
- sensor.html – Sensor-based prediction page

Static Folder:

- CSS files for styling

6. Running the Application

Steps:

1. Open terminal or command prompt.
2. Navigate to the project directory.
3. Run the command: python app.py
4. Open a browser and access the application through the local server URL.
5. API Documentation

7. API Documentation

The backend exposes the following routes using Flask:

• Home Route: URL: / Method: GET Description: Loads the home page of the application.

• Manual Prediction Route: URL: /manual Method: GET Description: Displays the manual input prediction page.

• Sensor Prediction Route: URL: /sensor Method: GET Description: Displays the sensor-based prediction page.

• Prediction Route: URL: /predict Method: POST Description: Receives input parameters from the user, applies preprocessing using the saved scaler, loads the trained machine learning model, and returns the predicted electric motor temperature.

• Request Parameters:

- Motor input parameters (for manual mode)
- Sensor values (for sensor mode)

• Response:

- Predicted electric motor temperature value displayed on the result page.

8. Authentication

• Authentication is not implemented in this project.

• The application is designed for academic and demonstration purposes and allows open access.

9. User Interface

• The user interface consists of simple web pages for input and output.

• Users can enter motor parameters manually or provide sensor values.

• Predicted temperature results are displayed clearly on the result page.

10. Testing

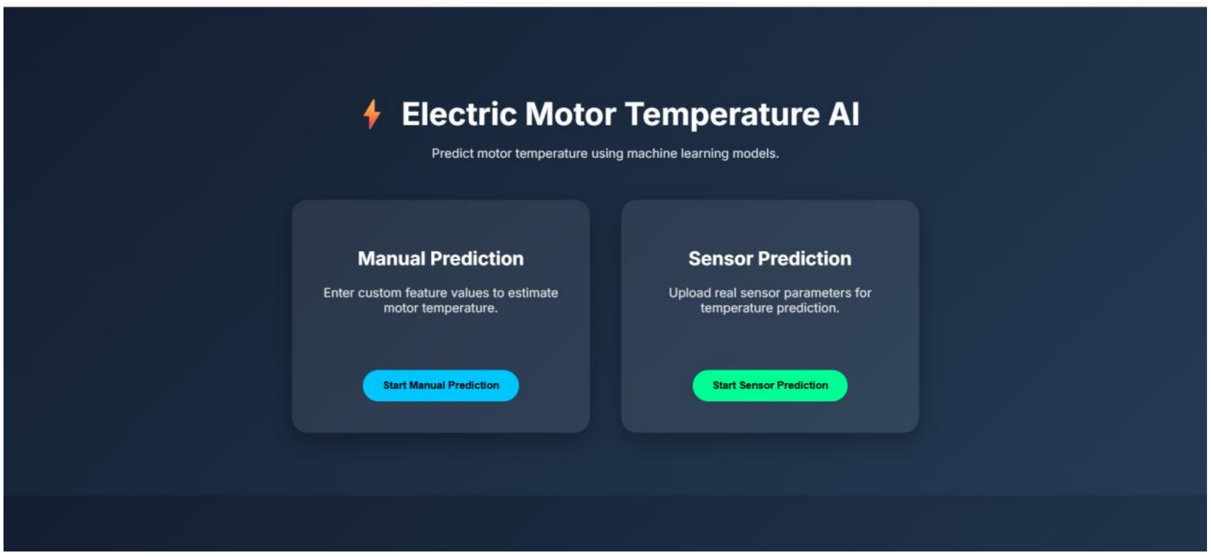
• The application is tested using sample input values.

• Model accuracy is validated during the training phase.

• Manual testing is performed to verify correct predictions and UI flow.

11. Screenshots or Demo

• Provided screenshots the application.



Electric Motor Temperature AI

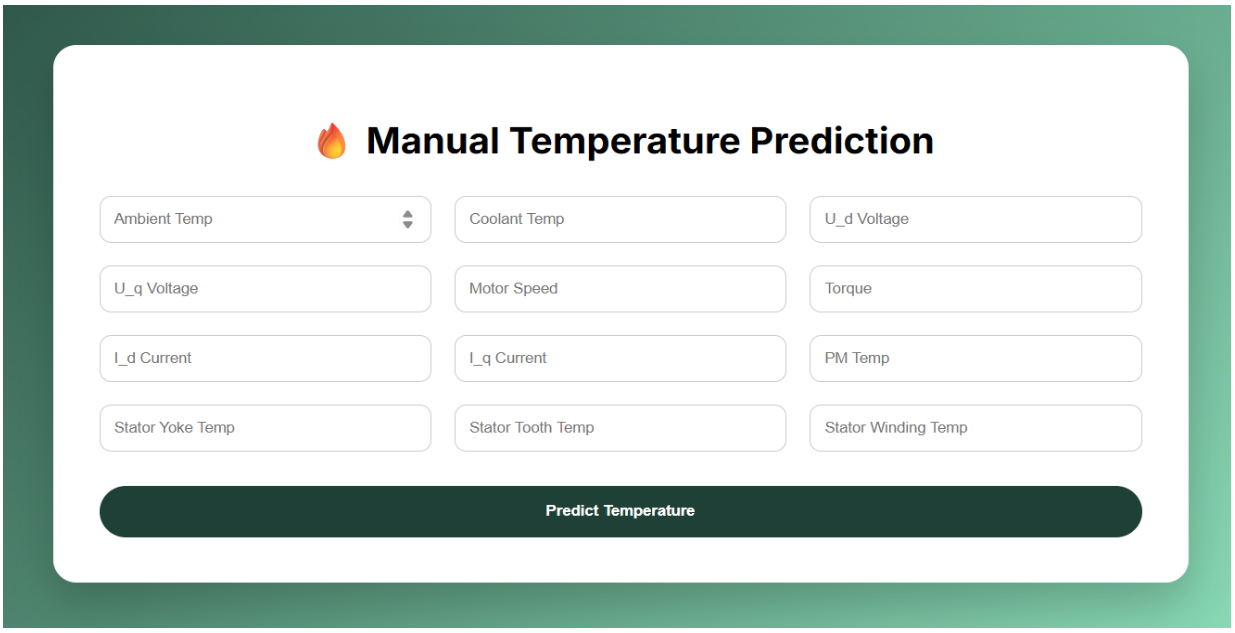
Predict motor temperature using machine learning models.

Manual Prediction

Enter custom feature values to estimate motor temperature.

Sensor Prediction

Upload real sensor parameters for temperature prediction.



Manual Temperature Prediction

Ambient Temp

Coolant Temp

U_d Voltage

U_q Voltage

Motor Speed

Torque

I_d Current

I_q Current

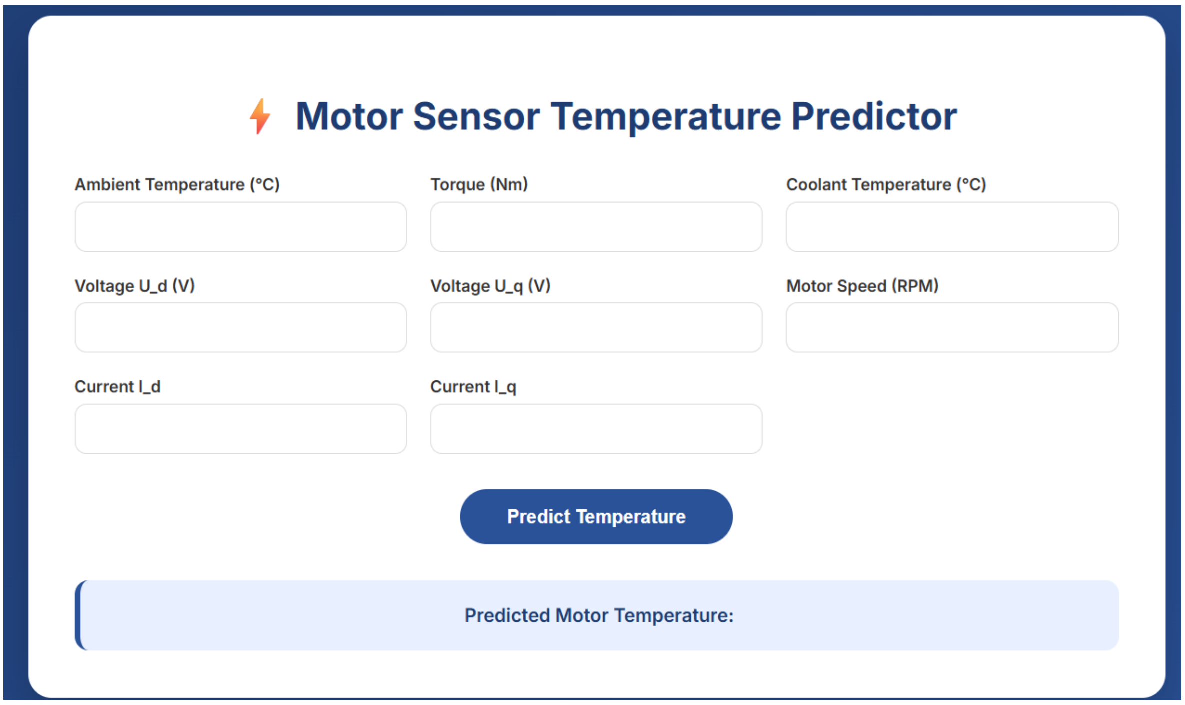
PM Temp

Stator Yoke Temp

Stator Tooth Temp

Stator Winding Temp

Predict Temperature



Motor Sensor Temperature Predictor

Ambient Temperature (°C)

Torque (Nm)

Coolant Temperature (°C)

Voltage U_d (V)

Voltage U_q (V)

Motor Speed (RPM)

Current I_d

Current I_q

Predict Temperature

Predicted Motor Temperature:

12. Known Issues

- Prediction accuracy depends on the quality of input data.
- Real-time sensor integration is simulated using manual input.

13. Future Enhancements

- Integrate real-time IoT sensors for live data collection
- Improve model accuracy using advanced algorithms
- Add graphical visualization of temperature trends
- Implement user authentication and role management