Electric Motor Temperature Prediction using Machine Learning

1. INTRODUCTION

Project title: Electric Motor Temperature Prediction Using Machine Learning.

Team ID: LTVIP2026TMIDS74581

Team Members:

Satya Siva Sai Ganesh Valluri
Marise Radha Vaishnawe
Velagala Jyothi Ayyappa Swarupa Reddy
Kommanapalli Yeswanth Ganesh

2. Project Overview

Electric motors are widely used in industrial systems, electric vehicles, manufacturing plants, and automation sectors. Overheating electric motors can result in equipment failure, operational downtime, increased maintenance cost, and safety risks.

This project focuses on developing a Machine Learning-based predictive system to estimate motor temperature using operational and environmental parameters. By analyzing historical motor data, a predictive model is trained to forecast motor temperature accurately.

The system is deployed using a Flask web application that allows users to input motor parameters and receive real-time temperature predictions.

2.1 Purpose

The purpose of this project is:

- To develop an intelligent system for motor temperature prediction
- To prevent overheating and equipment damage
- To reduce downtime in industrial operations
- To implement predictive maintenance
- To deploy a real-time ML web application

3. IDEATION PHASE

3.1 Problem Statement

Electric motors in industries frequently experience overheating due to excessive load, poor cooling mechanisms, and environmental conditions. Traditional monitoring systems rely on threshold-based alerts, which react only after the temperature crosses dangerous levels.

Problem:
How can we develop a machine learning-based solution to predict electric motor temperature in advance to prevent overheating and improve reliability?

3.2 Empathy Map Canvas

Target User: Maintenance

Engineer Thinks:

- Wants reliable temperature monitoring
- Concerned about unexpected Motor Failures

Feels:

Pressure to reduce downtime

- Responsible for equipment safety

Sees:

- Rising maintenance costs
- Frequent equipment wear and tear

Hears:

- Complaints about production delays

Pain Points:

- No early warning system
- High repair and replacement cost

Gains:

- Predictive monitoring
- Improved operational efficiency

3.3 Brainstorming

Various approaches were analyzed:

- Manual inspection
- Basic threshold alert systems
- IoT monitoring devices
- Machine Learning predictive model

The ML approach was selected due to higher accuracy and scalability.

4. REQUIREMENT ANALYSIS

4.1 Customer Journey Map

1. User enters motor operational parameters
2. System processes the data
3. Machine learning model predicts temperature
4. Predicted result is displayed
5. Maintenance decision is taken

4.2 Solution Requirements

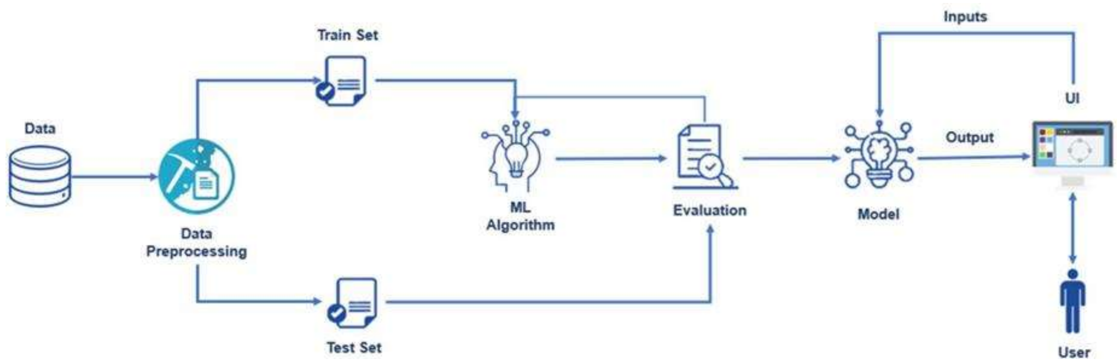
Functional Requirements

- Accept motor parameter inputs
- Process and clean data
- Predict motor temperature
- Display result on web page
- Store trained model

Non-Functional Requirements

- High accuracy
- Fast prediction response
- User-friendly interface
- Reliable system performance

4.3 Data Flow Diagram



4.4 Technology Stack

Category	Technology
Language	Python
ML Libraries	NumPy, Pandas, Scikit-learn
Visualization	Matplotlib, Seaborn
Model	Random Forest Regressor
Web Framework	Flask
Deployment	Render
Model Saving	Joblib
Environment	Jupyter Notebook

PROJECT DESIGN

4.4 Problem Solution Fit

The proposed machine learning solution effectively addresses the problem of motor overheating by predicting temperature before it reaches critical levels.

The high R^2 score (≈ 0.9999) indicates excellent prediction performance, validating the solution's effectiveness.

4.5 Proposed Solution

The system uses the following features:

1. u_q
2. Coolant
3. u_d
4. motor_speed
5. i_d
6. i_q
7. Pm
8. stator_yoke
9. Ambient
10. torque
11. stator_tooth

A Random Forest model is trained and saved as:

motor_temperature_model.pkl

The Flask application loads this model and provides real-time prediction.

4.6 Solution Architecture

The solution architecture of the Electric Motor Temperature Prediction system follows a layered and modular approach consisting of data processing, machine learning, and deployment components.

The first layer is the Data Layer, where historical motor operational data is collected and preprocessed. This includes data cleaning, feature selection, handling missing values, and splitting the dataset into training and testing sets.

The second layer is the Machine Learning Layer, where the Random Forest Regressor model is trained using Scikit-learn. The model is evaluated using performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R^2 score. After achieving high accuracy, the trained model is saved as a serialized file (motor_temperature_model.pkl) for deployment.

The third layer is the Application Layer, developed using Flask. The Flask backend loads the trained model and processes user input received from the web interface. When the user enters motor parameters, the system converts them into the required format and sends them to the trained model for prediction.

The final layer is the Presentation Layer, which consists of an HTML-based user interface. The predicted motor temperature is displayed instantly to the user.

The system can be deployed locally or on cloud platforms such as Render, ensuring scalability and real-time accessibility.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

The project was executed in a systematic and phased manner to ensure proper implementation and testing. The development process followed a structured approach beginning with problem identification and ending with deployment and documentation.

Initially, the dataset was collected and analyzed to understand its structure, features, and target variable. During this phase, exploratory data analysis (EDA) was performed to study relationships between parameters such as motor speed, torque, coolant temperature, and stator temperature.

In the second phase, data preprocessing was carried out. This included handling missing values, removing irrelevant features, detecting outliers, and splitting the dataset into training and testing sets. Proper preprocessing ensured that the machine learning model received clean and structured data.

The third phase focused on model development. Multiple algorithms were considered, and Random Forest Regressor was selected due to its high accuracy and robustness. The model was trained, evaluated using MAE, RMSE, and R^2 metrics, and fine-tuned to achieve optimal performance.The fourth phase involved application development. A Flask-based web application was developed to integrate the trained model with a user interface. The model file was serialized and loaded into the backend to enable real-time prediction functionality.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Model evaluation results:

- Mean Absolute Error (MAE): ~1.32
- Root Mean Square Error (RMSE): ~1.83
- R^2 Score: ~0.9999

The high R^2 value confirms excellent predictive performance.

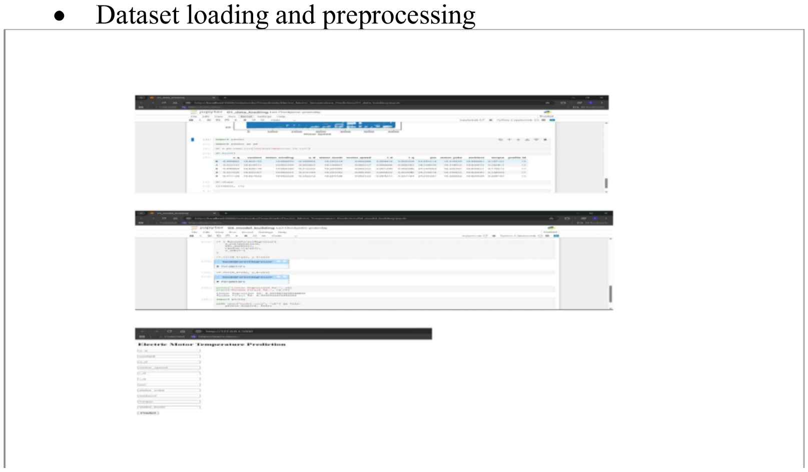
System response time for prediction is less than 1 second.

7. RESULTS

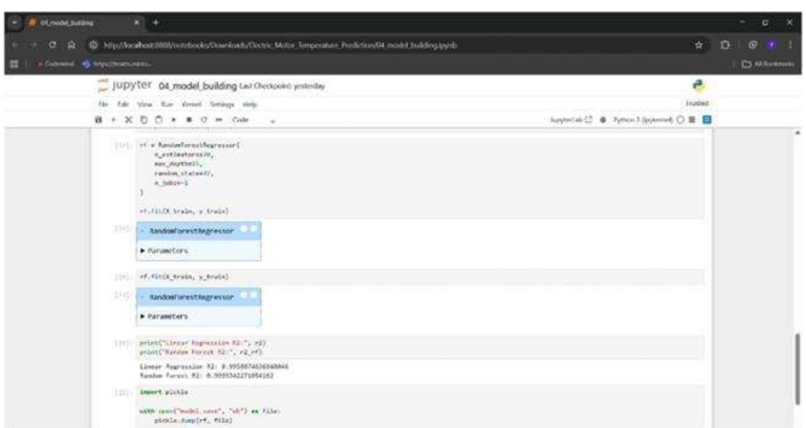
7.1 Output Screenshots

The following outputs were generated:

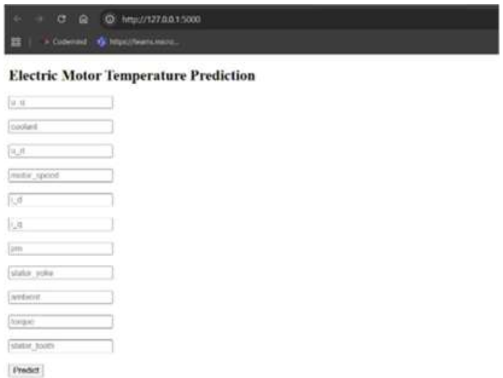
- Dataset loading and preprocessing



- Model evaluation metrics



- Flask web application interface



7.ADVANTAGES & DISADVANTAGES

Advantages

- Highly accurate prediction
- Prevents motor overheating
- Reduces maintenance cost
- Supports predictive maintenance
- Scalable solution

Disadvantages

- Requires historical dataset
- Not connected to live sensors
- Model accuracy depends on input quality

8. CONCLUSION

The Electric Motor Temperature Prediction system successfully demonstrates the application of Machine Learning in predictive maintenance. The Random Forest model achieved excellent performance and was deployed using a Flask web application.

This solution helps industries prevent motor failures, reduce downtime, and improve operational reliability.

9. FUTURE SCOPE

- Integration with IoT sensors
- Mobile app development
- Cloud scalability enhancement
- Multi-motor monitoring dashboard
- Anomaly detection integration

10. APPENDIX

Source Code:
Available in GitHub repository

Dataset Link:
https://www.kaggle.com/datasets/wkirgsn/electric-motor-temperature

GitHub Repository:
https://github.com/yeswanthganesh/Electric-Motor-Temperature-Prediction

Project Demo Video:
https://drive.google.com/drive/folders/1m_vXdKkujfkVq1x57h6bZ3Kj_07hdh4T?usp=sharing