Electric Motor Temperature Prediction System

Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru

Team ID:

Team Members Details:

S.No	Name of the Student
1	Arepalli Sudeepthi
2	Godavarthi Sasi Kumar
3	Chitturi Poojitha
4	Kollu Bindu Priya

AI&ML Project Documentation

Project Title:

Electric Motor Temperature Prediction System Using Machine Learning

Project Description:

This project aims to predict the temperature of an electric motor using Machine Learning techniques to improve operational safety and efficiency. Overheating in electric motors can lead to reduced performance, equipment damage, and increased maintenance costs.

The system analyzes sensor data such as voltage, current, torque, speed, and ambient conditions to predict motor temperature in real time. By leveraging predictive analytics, the application enables early detection of abnormal temperature rise and supports preventive maintenance decisions.

The solution automates temperature prediction, reduces manual monitoring efforts, and enhances reliability through data-driven insights.

Project Objectives:

- Develop a machine learning model to predict electric motor temperature
- Analyze sensor-based operational parameters affecting motor performance
- Reduce risk of motor failure through early prediction
- Improve maintenance planning using predictive analytics
- Automate data preprocessing and model evaluation
- Visualize prediction results for better understanding
- Enhance system reliability and operational efficiency

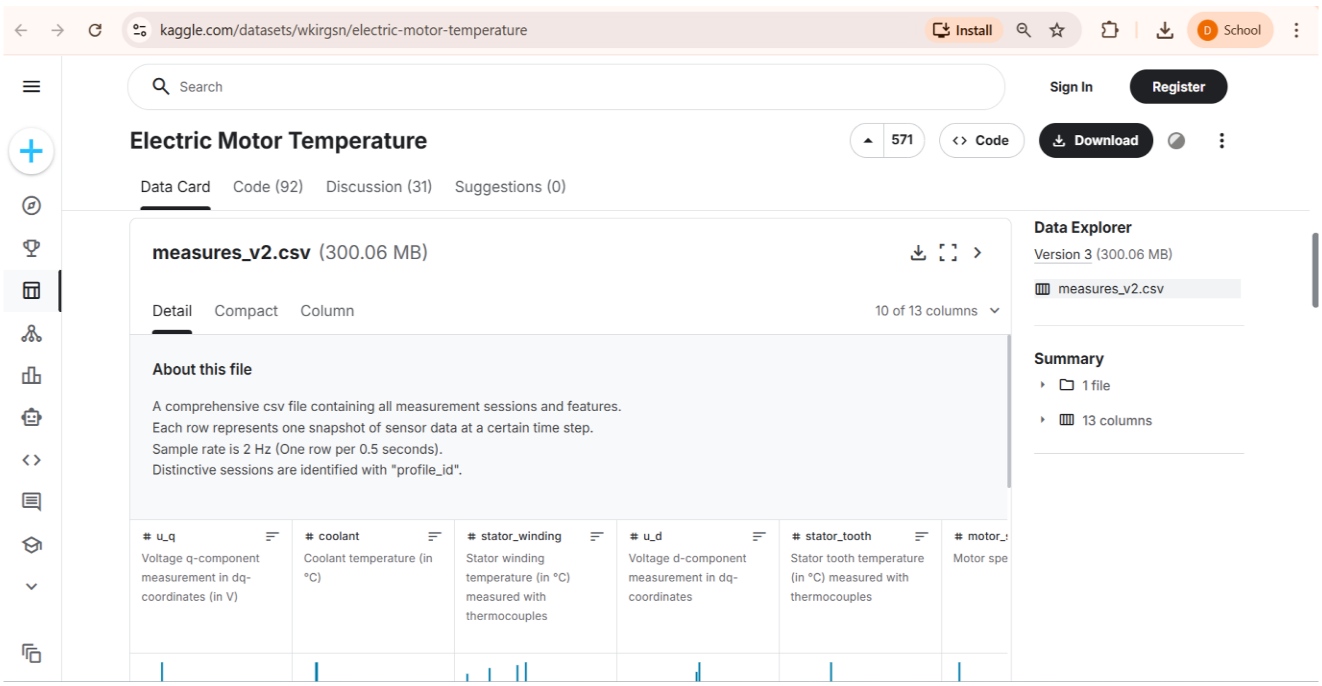
Project Features:

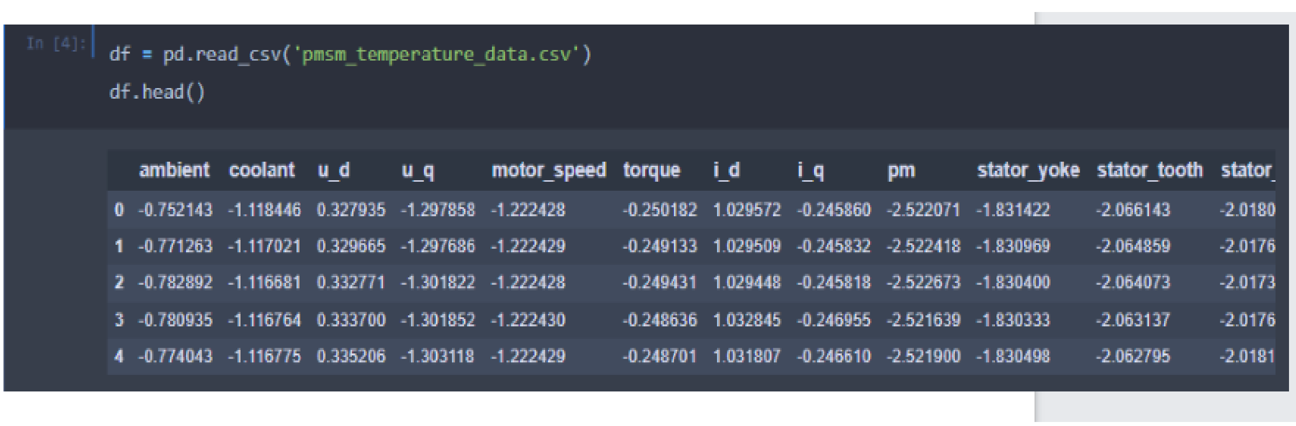
- Machine learning-based temperature prediction
- Data preprocessing and feature engineering
- Training and testing using real-world motor sensor datasets
- Model performance evaluation using regression metrics
- Visualization of temperature trends and predictions
- Automated prediction workflow
- Graphical insights using data visualization tools
- Accurate and scalable predictive system

Project Flow & Milestones:

1. Dataset Collection

Collected electric motor sensor data containing operational parameters such as voltage, torque, speed, and environmental conditions.





```
In [4]: df = pd.read_csv('pmsm_temperature_data.csv')
        df.head()
```

	ambient	coolant	u_d	u_q	motor_speed	torque	i_d	i_q	pm	stator_yoke	stator_tooth	stator_
0	-0.752143	-1.118446	0.327935	-1.297858	-1.222428	-0.250182	1.029572	-0.245860	-2.522071	-1.831422	-2.066143	-2.0180
1	-0.771263	-1.117021	0.329665	-1.297686	-1.222429	-0.249133	1.029509	-0.245832	-2.522418	-1.830969	-2.064859	-2.0176
2	-0.782892	-1.116681	0.332771	-1.301822	-1.222428	-0.249431	1.029448	-0.245818	-2.522673	-1.830400	-2.064073	-2.0173
3	-0.780935	-1.116764	0.333700	-1.301852	-1.222430	-0.248636	1.032845	-0.246955	-2.521639	-1.830333	-2.063137	-2.0176
4	-0.774043	-1.116775	0.335206	-1.303118	-1.222429	-0.248701	1.031807	-0.246610	-2.521900	-1.830498	-2.062795	-2.0181

2. Data Preprocessing

```
df.head()
```

ant	u_d	u_q	motor_speed	torque	i_d	i_q	pm	stator_yoke	stator_tooth	stator_winding	profile_id
446	0.327935	-1.297858	-1.222428	-0.250182	1.029572	-0.245860	-2.522071	-1.831422	-2.066143	-2.018033	4
021	0.329665	-1.297686	-1.222429	-0.249133	1.029509	-0.245832	-2.522418	-1.830969	-2.064859	-2.017631	4
681	0.332771	-1.301822	-1.222428	-0.249431	1.029448	-0.245818	-2.522673	-1.830400	-2.064073	-2.017343	4
764	0.333700	-1.301852	-1.222430	-0.248636	1.032845	-0.246955	-2.521639	-1.830333	-2.063137	-2.017632	4
775	0.335206	-1.303118	-1.222429	-0.248701	1.031807	-0.246610	-2.521900	-1.830498	-2.062795	-2.018145	4

Performed:

- Handling missing values

```
In [11]: df.isnull().sum()

ambient           0
coolant           0
u_d               0
u_q               0
motor_speed       0
torque            0
i_d               0
i_q               0
pm                0
stator_yoke       0
stator_tooth      0
stator_winding    0
profile_id        0
dtype: int64
```

- Data cleaning and normalization

```
mm = MinMaxScaler()
X = mm.fit_transform(X)
X_df_test = mm.fit_transform(X_df_test)
y = df['pm']
y_df_test = df_test['pm']
X = pd.DataFrame(X,columns = ['ambient', 'coolant', 'u_d', 'u_q', 'motor_speed', 'i_d','i_q'])
X_df_test = pd.DataFrame(X_df_test,columns = ['ambient', 'coolant', 'u_d', 'u_q', 'motor_speed', 'i_d','i_q
y.reset_index(drop = True,inplace = True)
y_df_test.reset_index(drop = True,inplace = True)
```

- Feature selection and transformation

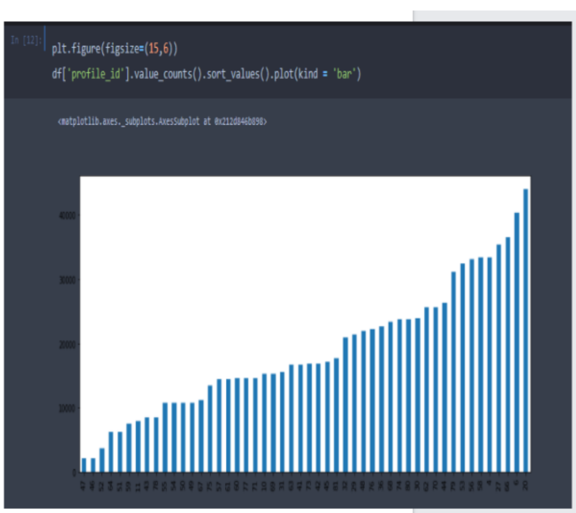
```
import joblib
joblib.dump(mm,'transform.save')
```

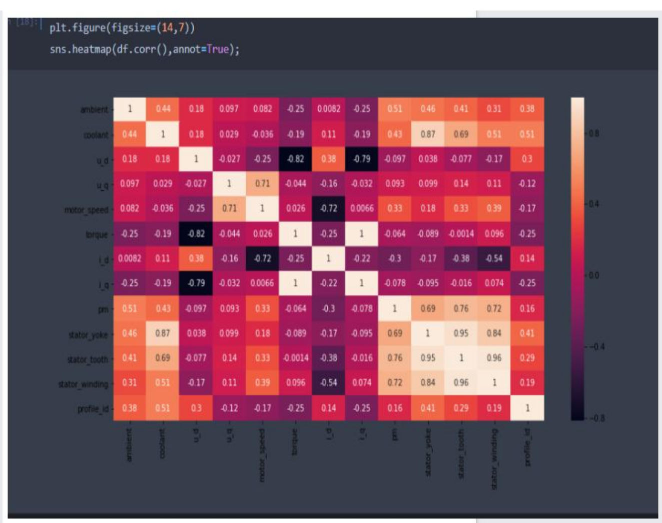
```
['transform.save']
```

3. Exploratory Data Analysis (EDA)

Analyzed relationships between variables using:

- Correlation analysis





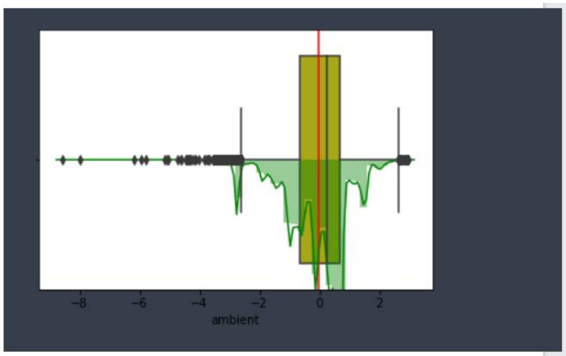
- Distribution plots

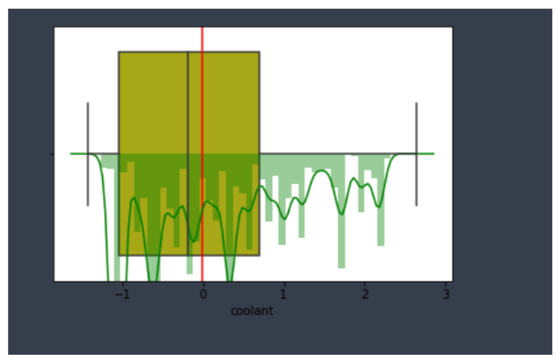
```
In [13]: df.columns

Index(['ambient', 'coolant', 'u_d', 'u_q', 'motor_speed', 'torque', 'i_d',
       'i_q', 'pm', 'stator_yoke', 'stator_tooth', 'stator_winding',
       'profile_id'],
      dtype='object')
```

```
In [14]: #Plotting Distribution and Boxplot for all the features to check for skewness
```

```
In [15]: for i in df.columns:
             sns.distplot(df[i],color='g')
             sns.boxplot(df[i],color = 'y')
             plt.vlines(df[i].mean(),ymin = -1,ymax = 1,color = 'r')#drawing the mean line
             plt.show()
```





- Trend visualization

```
In [50]: fig, axes = plt.subplots(2, 4, figsize=(20, 5),sharey=True)
         sns.scatterplot(df['ambient'],df['pm'],ax=axes[0][0])
         sns.scatterplot(df['coolant'],df['pm'],ax=axes[0][1])
         sns.scatterplot(df['motor_speed'],df['pm'],ax=axes[0][2])
         sns.scatterplot(df['i_d'],df['pm'],ax=axes[0][3])
         sns.scatterplot(df['u_q'],df['pm'],ax=axes[1][0])
         sns.scatterplot(df['u_d'],df['pm'],ax=axes[1][1])
         sns.scatterplot(df['i_q'],df['pm'],ax=axes[1][2])

<matplotlib.axes._subplots.AxesSubplot at 0x212d84f4f60>
```

4.Model Development & Training

Implemented machine learning regression models to predict motor temperature.

Trained the model using historical sensor data and optimized parameters for improved accuracy.

cing motor temperature and prepared training datasets.

```
In [51]: from sklearn.linear_model import LinearRegression
         from sklearn.tree import DecisionTreeRegressor
         from sklearn.ensemble import RandomForestRegressor
         from sklearn.svm import SVR
```

```
In [52]: lr=LinearRegression()
         dr=DecisionTreeRegressor()
         rf =RandomForestRegressor()
         svm =SVR()
```

```
In [*]: lr.fit(X_train,y_train)
        dr.fit(X_train,y_train)
        rf.fit(X_train,y_train)
        svm.fit(X_train,y_train)
```

```
from sklearn import metrics
```

```
print(metrics.r2_score(y_test,p1))
print(metrics.r2_score(y_test,p2))
print(metrics.r2_score(y_test,p3))
print(metrics.r2_score(y_test,p4))

0.9698725757690718
0.47757469778170836
```

6. Model Evaluation

Evaluated performance using:

- Mean Squared Error (MSE)

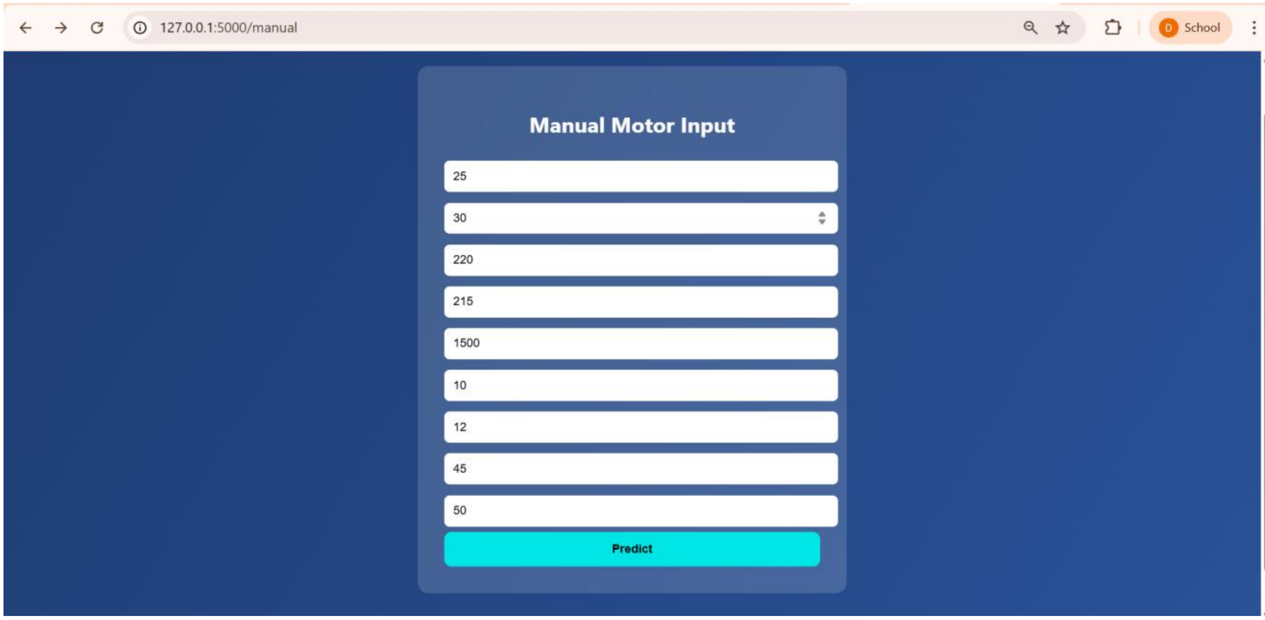
```
In [20]: from sklearn.metrics import mean_squared_error
```

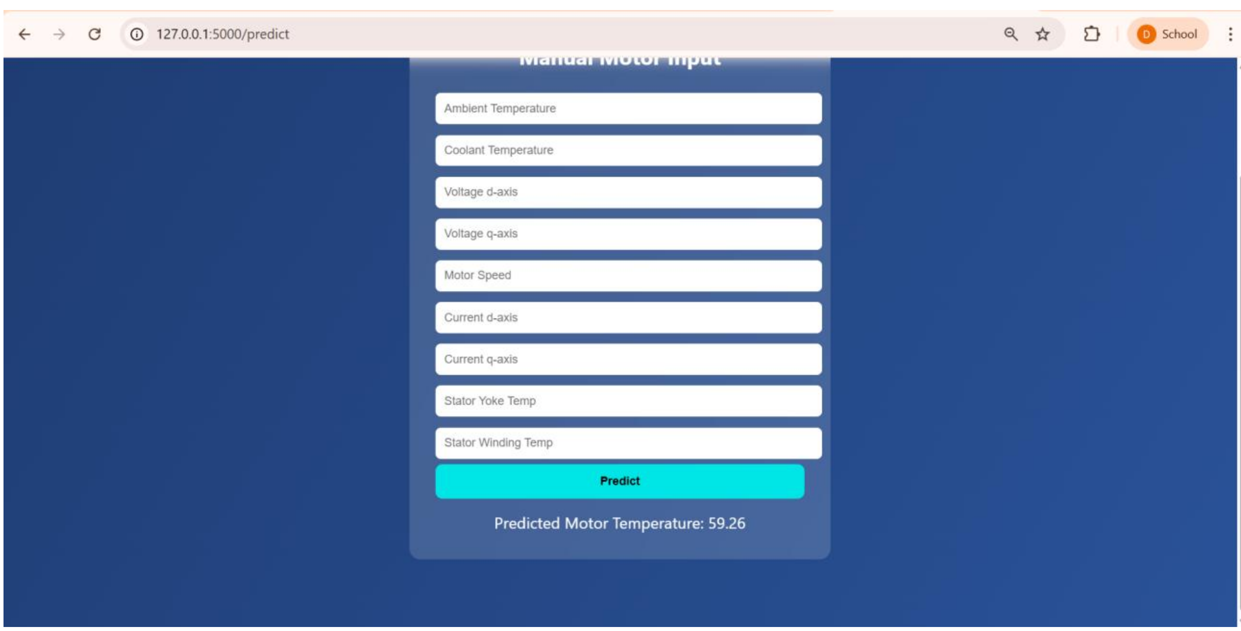
```
In [26]: print(mean_squared_error(y_test,p1))

0.030187256377134934
```

7. Prediction System

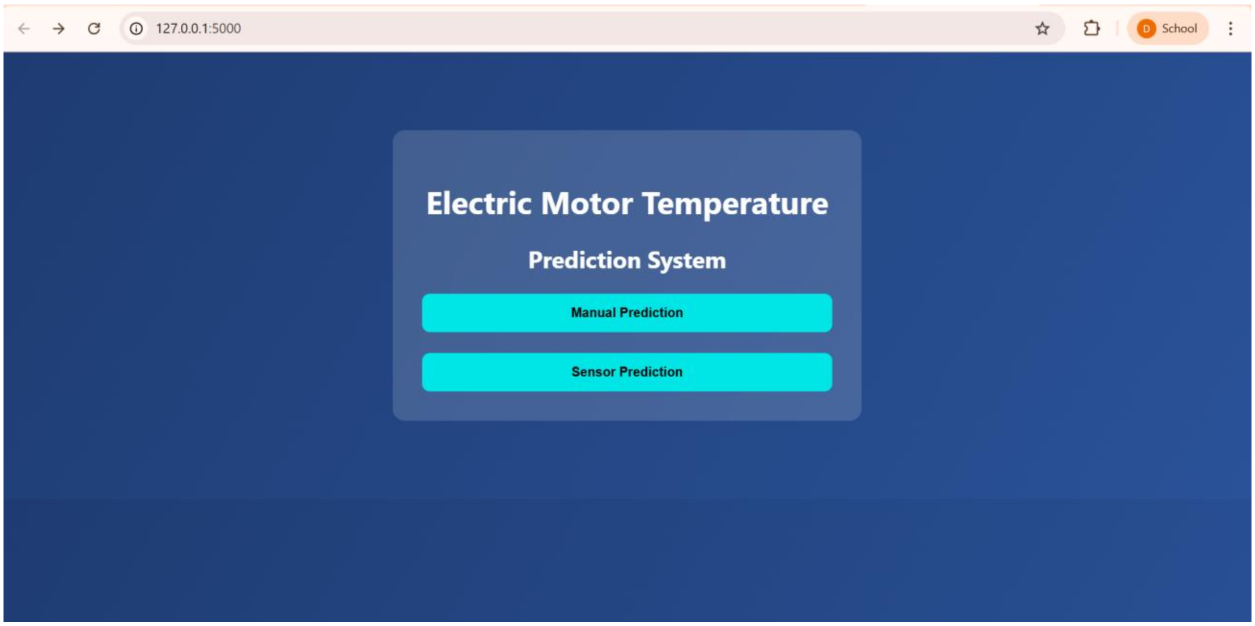
Built a system capable of predicting motor temperature based on new input sensor values.

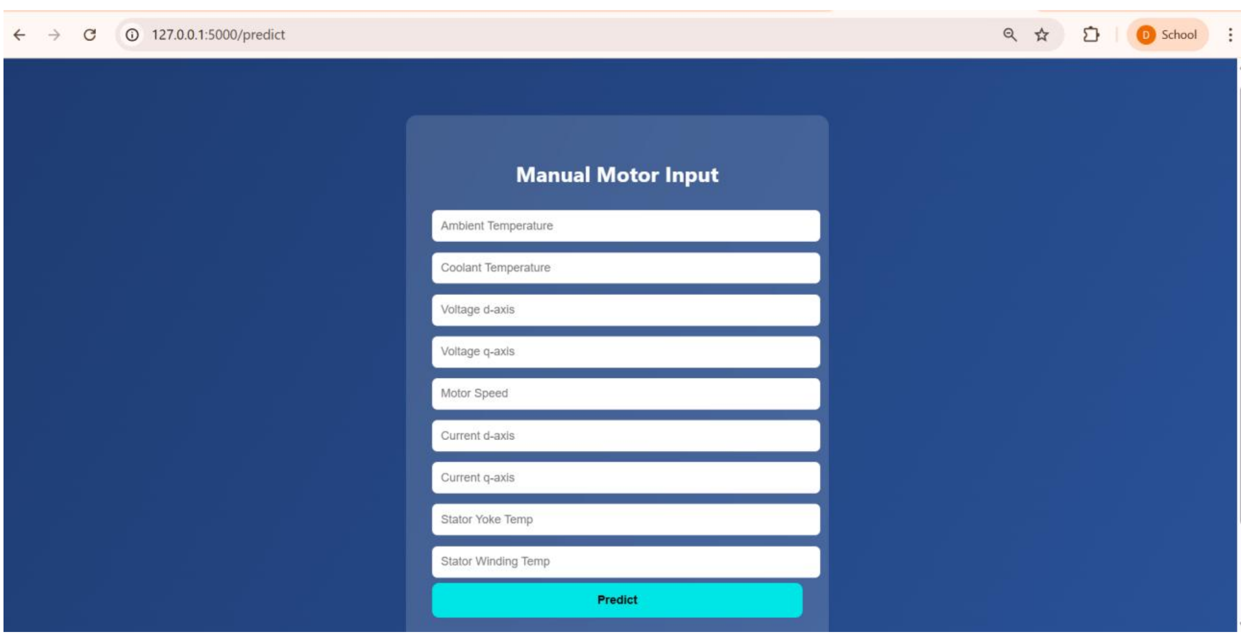




8 Final Output

A predictive analytics system that estimates electric motor temperature and assists in proactive maintenance decisions.





Technologies & Tools Used:

- Python
- Machine Learning Algorithms (Regression Models)
- Pandas & NumPy (Data Processing)
- Scikit-learn (Model Building)
- Matplotlib & Seaborn (Visualization)
- Jupyter Notebook / VS Code

Learning Outcomes:

- Understanding of machine learning regression techniques
- Experience in data preprocessing and feature engineering
- Model evaluation and performance optimization
- Data visualization and interpretation skills
- Practical knowledge of predictive maintenance systems

Conclusion:

The Electric Motor Temperature Prediction System provides an intelligent and automated approach to monitoring motor health. By predicting temperature using machine learning models, the system helps prevent overheating issues, improves equipment lifespan, and supports data-driven maintenance strategies. This solution demonstrates the practical application of AI and Machine Learning in industrial monitoring and smart manufacturing environments.