Infosys Springboard Virtual Internship 6.0 Completion Report

Project Title

Tempest FWI Predictor - A ML Model to Predict Fire Weather Index

Batch Number: 6

Start date: 10.11.2025

Name: Shruti Vilas Wani

Internship Duration: 8 Weeks

Section- 1

Dataset – FWI Dataset

1.1 Overview of the Dataset

The dataset used in this project is the Fire Weather Index (FWI) Dataset, which is designed to evaluate and predict the potential risk and intensity of forest fires based on meteorological conditions and fuel moisture indicators.

The Fire Weather Index is a widely used environmental index that helps in estimating how weather variables such as temperature, humidity, wind speed, and rainfall influence fire behavior. The dataset contains daily recorded observations and is suitable for regression-based machine learning models, as the target variable is continuous in nature.

In this project, the dataset is used to train a Ridge Regression model to predict the Fire Weather Index (FWI) and deploy the trained model using a Flask web application.

1.2 Feature Description

The dataset consists of the following features, each of which plays a significant role in determining forest fire behavior:

1. Day

- Represents the **day of the month** when the observation was recorded.
- Helps in capturing **daily weather variations**.
- Useful for identifying short-term trends in fire risk.

2. Month

- Indicates the **month of the year**.
- Fire occurrences are often **seasonal**, with higher risks during dry summer months.
- Enables the model to learn **seasonal fire patterns**.

3. Year

- Represents the **year of observation**.
- Helps analyze **long-term trends** in fire activity.
- Useful when data spans across multiple years.

4. Temperature (°C)

- Measures the **ambient air temperature**.
- Higher temperatures dry vegetation and fuels, increasing fire risk.
- Strongly influences the ignition and intensity of fires.

5. Relative Humidity (%)

- Represents the **amount of moisture present in the air**.
- Lower humidity results in drier fuels, increasing fire probability.

- Higher humidity generally reduces fire spread.

6. Wind Speed (km/h)

- Indicates how fast the wind is blowing.
- Wind significantly affects:
  - Fire spread speed
  - Fire direction
- Higher wind speeds lead to more aggressive fire behavior.

7. Rain (mm)

- Represents the **amount of rainfall** received.
- Rainfall increases moisture in vegetation and soil.
- Even small rainfall amounts can significantly reduce fire risk.

8. FFMC (Fine Fuel Moisture Code)

- Measures the **moisture content of surface litter and fine fuels**.
- High FFMC values indicate dry fuels that ignite easily.
- Important for predicting **fire ignition probability**.

9. DMC (Duff Moisture Code)

- Represents the **moisture content of loosely compacted organic layers** below surface litter.
- Reflects medium-term drying effects.
- Higher DMC values increase the likelihood of sustained fires.

10. DC (Drought Code)

- Measures **long-term moisture deficiency** in deep soil layers.
- High DC values indicate prolonged drought conditions.
- Strong indicator of severe and large-scale fires.

11. ISI (Initial Spread Index)

- Estimates the **expected rate of fire spread** immediately after ignition.
- Influenced by wind speed and FFMC.
- Higher ISI values indicate faster-spreading fires.

12. BUI (Buildup Index)

- Represents the **total amount of fuel available for combustion**.
- Calculated using DMC and DC.
- High BUI values lead to more intense fires.

13. Region

- Categorical feature representing **different geographical regions**.
- Different regions have different climate conditions and vegetation types.
- Helps the model capture **regional fire behavior variations**.

1.3 Data Standardization

1. Introduction to Data Preprocessing

Data preprocessing is an essential step in machine learning to improve data quality and model performance. Raw datasets may contain missing values, duplicates, and features with different scales, which can negatively affect predictions.

In the **Tempest FWI Predictor** project, preprocessing ensures that the Fire Weather Index (FWI) dataset is clean, consistent, and suitable for **Ridge Regression**, which is sensitive to feature scaling.

---

2. Dataset Loading and Initial Exploration

The Fire Weather Index dataset is loaded using the Pandas library. Initial exploration is performed to understand the dataset structure and statistical properties.

This includes:

- Viewing initial records
- Inspecting data types
- Generating descriptive statistics

This step helps identify missing values and abnormal data ranges.

---

3. Data Cleaning

3.1 Handling Missing Values

- Missing numerical values are replaced using the **mean of the respective column**.
- Mean imputation preserves the overall data distribution.

3.2 Removing Duplicate Records

- Duplicate rows are removed to avoid biased learning and maintain data integrity.

---

4. Exploratory Data Analysis (EDA)

4.1 Histogram Visualization

- Histograms are used to analyze feature distributions, skewness, and variability.

4.2 Correlation Analysis

- A correlation matrix and heatmap are generated to study relationships between features and the FWI target.
- This helps identify positively and negatively correlated variables and supports feature selection.

---

5. Feature Selection and Encoding

5.1 Selection of Input Features

The selected input features include:

- Day, Month, Year
- Temperature, Relative Humidity, Wind Speed, Rain
- FFMC, DMC, DC, ISI, BUI
- Region

The **FWI** column is chosen as the target variable.

5.2 Encoding of Categorical Feature

- The categorical **Region** feature is converted into numerical form using category encoding.

---

6. Train–Test Split

- The dataset is split into **80% training data** and **20% testing data**.
- This ensures fair evaluation and better generalization of the model.

---

7. Feature Scaling and Standardization

7.1 Need for Standardization

- Features are measured in different units, which can cause dominance of large-scale values.
- Since Ridge Regression uses **L2 regularization**, feature scaling is mandatory.

7.2 StandardScaler Technique

StandardScaler is used to transform features as:

$$X_{scaled} = \frac{X - \mu}{\sigma}$$

After standardization:

- Mean = 0
- Standard deviation = 1

7.3 Saving the Scaler

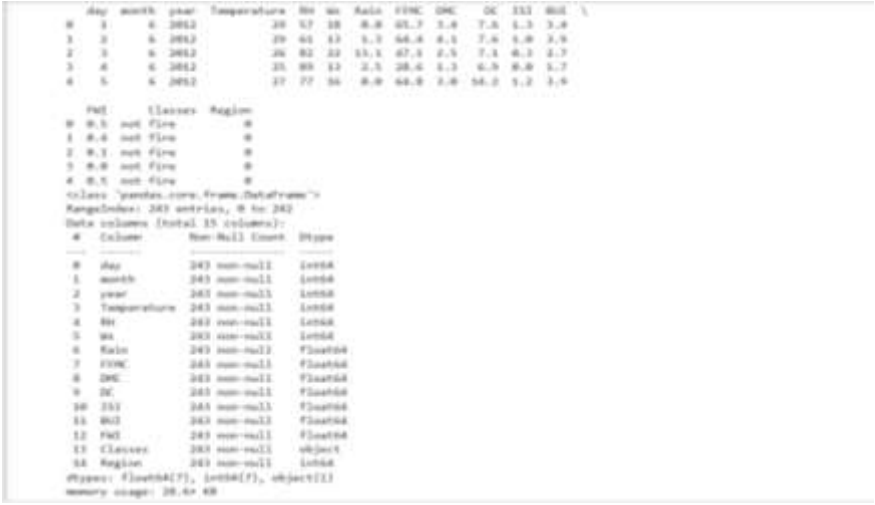
- The trained scaler is saved as a .pkl file for consistent preprocessing during deployment.

---

8. Pseudocode for Data Preprocessing and Standardization

BEGIN
Load dataset
Explore dataset structure and statistics
Handle missing values using mean
Remove duplicate records
Perform histogram and correlation analysis
Select relevant features
Encode categorical feature (Region)
Split data into training and testing sets
Apply StandardScaler
Save scaler
END

9. Snapshots



```
memory usage: 28.6+ KB
None
              day       month    year  Temperature          RH          Ws  \
count  243.000000  243.000000   243.0   243.000000  243.000000  243.000000
mean    15.761317    7.502058  2012.0    32.152263   62.041152   15.493827
std      8.842552    1.114793     0.0     3.628039   14.828160    2.811385
min      1.000000    6.000000  2012.0    22.000000   21.000000    6.000000
25%      8.000000    7.000000  2012.0    30.000000   52.500000   14.000000
50%     16.000000    8.000000  2012.0    32.000000   63.000000   15.000000
75%     23.000000    8.000000  2012.0    35.000000   73.500000   17.000000
max     31.000000    9.000000  2012.0    42.000000   90.000000   29.000000

             Rain        FFMC         DMC          DC         ISI         BUI  \
count  243.000000  243.000000  243.000000  243.000000  243.000000  243.000000
mean     0.762963   77.842387   14.680658   49.430864    4.742387   16.690535
std      2.003207   14.349641   12.393040   47.665606    4.154234   14.228421
min      0.000000   28.600000    0.700000    6.900000    0.000000    1.100000
25%      0.000000   71.850000    5.800000   12.350000    1.400000    6.000000
50%      0.000000   83.300000   11.300000   33.100000    3.500000   12.400000
75%      0.500000   88.300000   20.800000   69.100000    7.250000   22.650000
max     16.800000   96.000000   65.900000  220.400000   19.000000   68.000000

              FWI      Region
count  243.000000  243.000000
mean     7.035391    0.497942
std      7.440568    0.501028
min      0.000000    0.000000
25%      0.700000    0.000000
50%      4.200000    0.000000
75%     11.450000    1.000000
max     31.100000    1.000000
```

Section- 2

1.Histogram Visualization

Histogram visualization is used to understand the **distribution of numerical features** in the Fire Weather Index (FWI) dataset. In this project, histograms are plotted for all numerical columns after data cleaning.

Each histogram represents the frequency distribution of a single feature, allowing analysis of:

- **Data spread** (range of values)
- **Skewness** (left-skewed or right-skewed distributions)
- **Presence of outliers**
- **Variability of weather parameters**

By observing histograms, it becomes easier to identify whether features follow a normal distribution or exhibit skewed behavior. This step is important because non-uniform distributions can influence model learning and may require normalization. The use of plt.tight_layout() ensures proper spacing between plots for better readability.

---

2. Correlation Heatmap

A correlation heatmap is generated to examine the **linear relationships between numerical features** and the target variable, Fire Weather Index (FWI). The correlation matrix is computed using only numerical columns to avoid incompatibility with categorical data.

The heatmap provides a visual representation where:

- **Positive correlation** indicates that an increase in one feature leads to an increase in another
- **Negative correlation** indicates an inverse relationship
- **Color intensity** represents the strength of correlation

Key insights from correlation analysis include:

- Strong positive correlation of FWI with **Temperature, ISI, FFMC, and BUI**
- Negative correlation of FWI with **Rain and Relative Humidity**
- Identification of inter-feature relationships that may cause multicollinearity

This analysis supports **feature selection** and justifies the use of **Ridge Regression**, which handles correlated features effectively.

3.Feature Selection

Feature selection involves choosing the most relevant variables that influence the prediction of the Fire Weather Index. Based on **domain knowledge** and **correlation analysis**, the following features are selected:

- Temporal features: day, month, year
- Meteorological features: Temperature, Relative Humidity (RH), Wind Speed (Ws), Rain
- Fire fuel indices: FFMC, DMC, DC, ISI, BUI
- Geographical feature: Region

The target variable **FWI** is separated from the input features. Selecting only relevant features improves model efficiency, reduces noise, and enhances prediction accuracy.

4.Encoding of Categorical Feature

The **Region** feature is categorical and cannot be directly used by regression models. Therefore:

- The Region column is converted into a categorical data type
- Each category is encoded into a unique numerical code

This encoding allows the machine learning model to process regional information numerically while preserving distinct region identities.

5.Train–Test Split

To evaluate the generalization capability of the model, the dataset is divided into:

- **80% training data**
- **20% testing data**

The training set is used to learn model parameters, while the testing set is used to evaluate model performance on unseen data. The random_state parameter ensures reproducibility of results.

---

6. Feature Scaling and Standardization

The selected features have different units and value ranges, such as temperature in degrees, rainfall in millimeters, and index values with larger magnitudes. Without scaling, features with higher numerical ranges may dominate the learning process.

To address this issue, **StandardScaler** is applied:

- The scaler is fitted only on the training data to prevent data leakage
- Both training and testing data are transformed using the same scaler

Standardization ensures:

- Mean of each feature = 0
- Standard deviation = 1

This step is especially important for **Ridge Regression**, as it relies on distance-based calculations and L2 regularization.

---

7. Saving the Scaler

The fitted StandardScaler is saved as a **.pkl file**. This ensures that:

- The same preprocessing is applied during model deployment
- Predictions remain consistent with training conditions
- The deployed model produces reliable outputs

---

8. Pseudocode

Plot histograms for numerical features

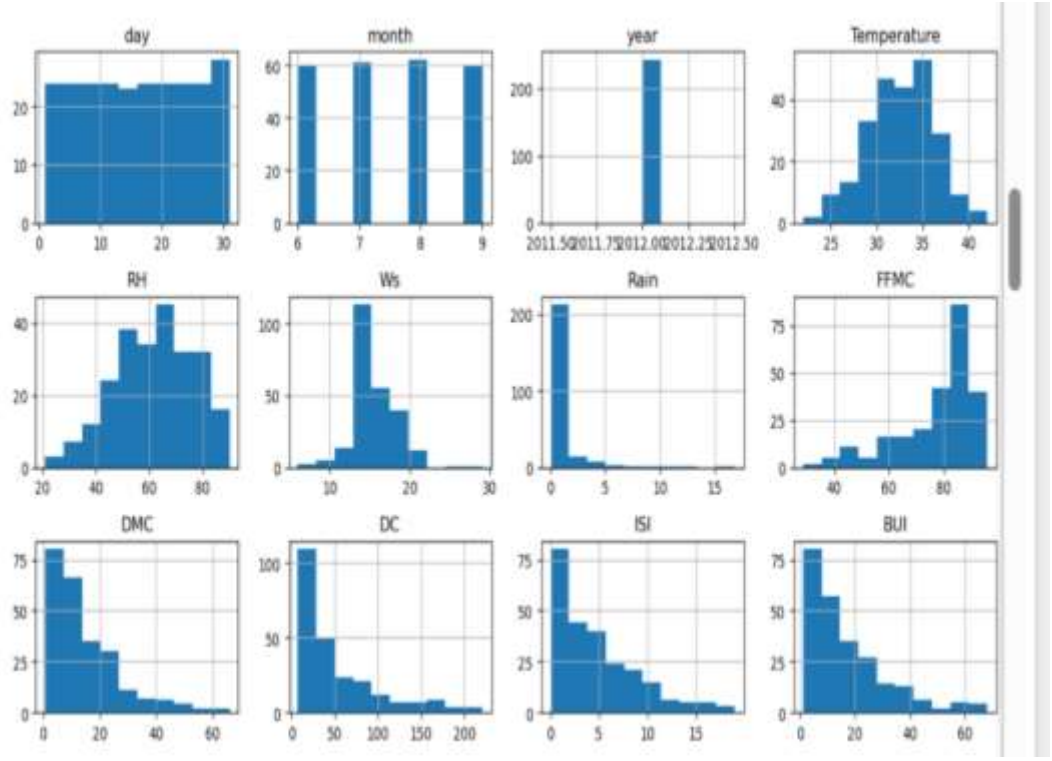
Compute and visualize correlation heatmap

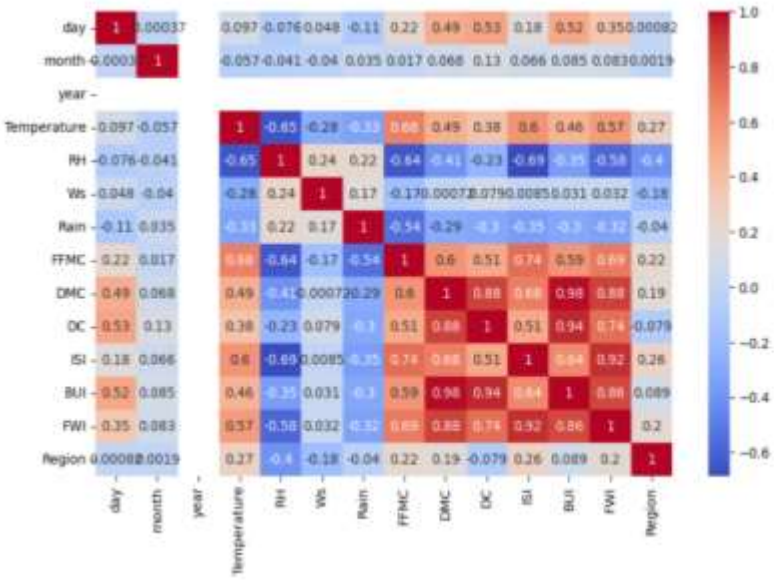
Select relevant features and encode categorical variable

Split dataset into training and testing sets

Apply StandardScaler and save the scaler

9.Snapshots


day
month
year
Temperature
RH
Ws
Rain
FFMC
DMC
DC
ISI
BUI


day
month
year
Temperature
RH
Ws
Rain
FFMC
DMC
DC
ISI
BUI
FWI
Region

Section- 3

1.Ridge Regression Model Training and Evaluation

Ridge Regression is a linear regression technique that uses **L2 regularization** to reduce model complexity and handle **multicollinearity** among input features. In the Fire Weather Index (FWI) dataset, several meteorological and fuel-related variables are highly correlated. Ridge Regression is therefore suitable as it penalizes large coefficients and improves model generalization.

In this project, multiple values of the regularization parameter **alpha (α)** are tested to identify the optimal balance between bias and variance. Smaller alpha values behave like simple linear regression, while larger alpha values apply stronger regularization.

2.Model Training with Multiple Alpha Values

A set of alpha values $[0.001, 0.01, 0.1, 1, 10, 50, 100]$ is evaluated. For each alpha value:

- A Ridge Regression model is trained using the scaled training data.
- Predictions are generated for both training and testing datasets.
- Performance is evaluated using:
  - **Mean Squared Error (MSE)**
  - **Root Mean Squared Error (RMSE)**
  - **Mean Absolute Error (MAE)**

These metrics help measure prediction accuracy and error magnitude.

3.Selection of Best Alpha

The optimal alpha value is selected based on the **minimum test Mean Squared Error (MSE)**.
This ensures the model performs best on unseen data and avoids overfitting.

The Ridge model is then retrained using the selected best alpha and used to generate final predictions on the test dataset.

4.Model Saving

The trained Ridge Regression model is saved as a **.pkl file**. This allows the model to be reused during deployment without retraining and ensures consistency in predictions.

5.Model Evaluation and Visualization

The following plots are generated to evaluate model performance:

- **Actual vs Predicted FWI:** Shows how closely predicted values match actual values.
- **MSE vs Alpha:** Analyzes error variation with different regularization strengths.
- **RMSE vs Alpha:** Highlights the effect of alpha on prediction error magnitude.
- **MAE vs Alpha:** Measures average absolute prediction error.

These visualizations help in understanding the impact of regularization on model performance.

6.Overfitting and Underfitting Check

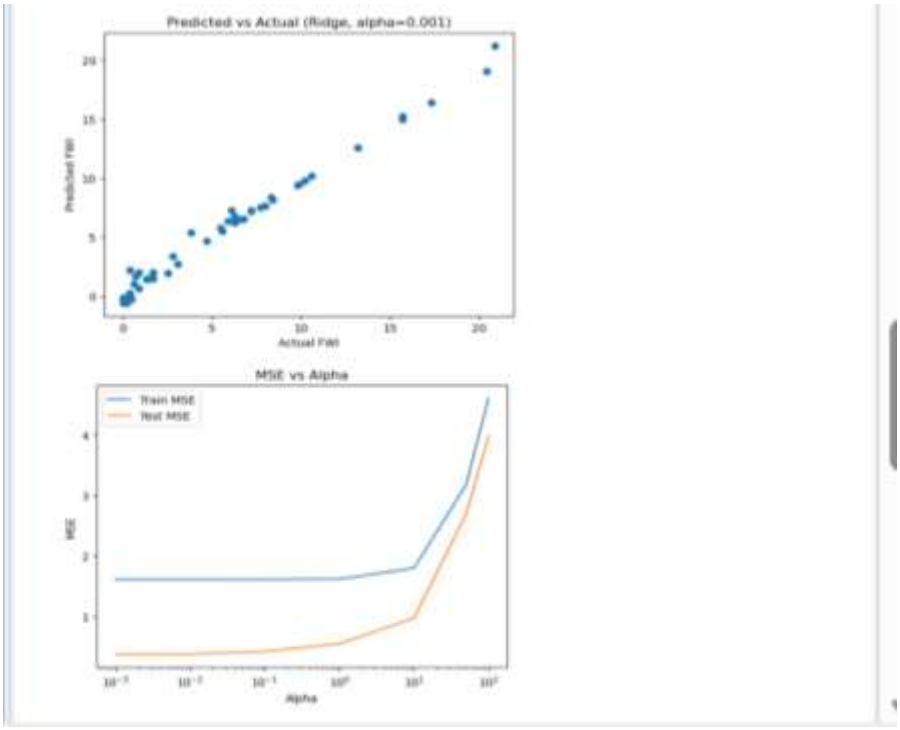
To assess model generalization:

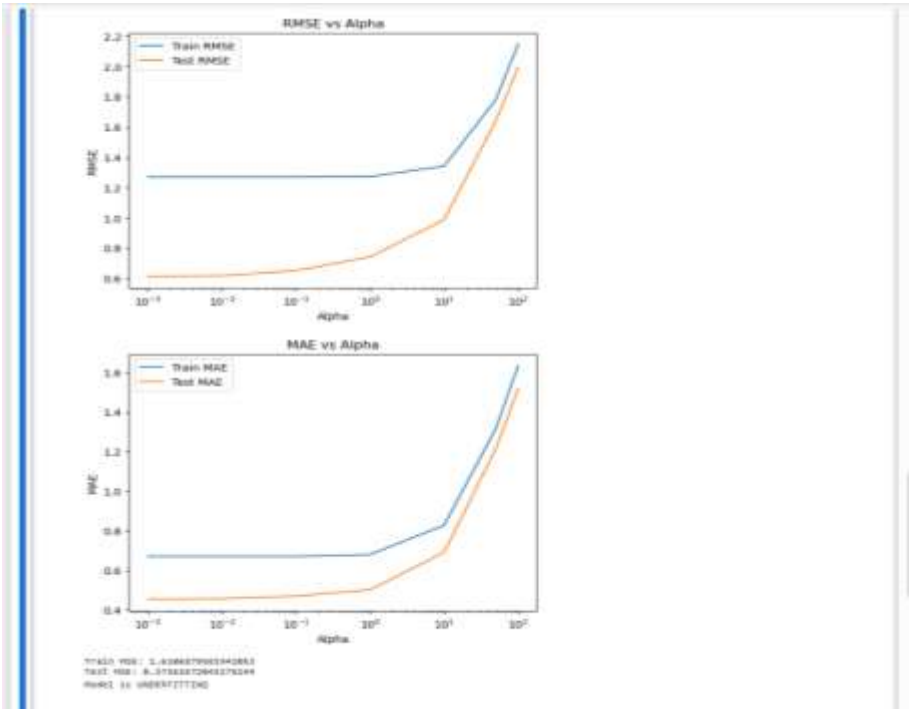
- Training MSE is compared with Testing MSE.
- If training error is much lower than testing error, the model is overfitting.
- If training error is much higher than testing error, the model is underfitting.
- A balanced error indicates a well-generalized model.

7.Short Pseudocode: Ridge Regression

```
BEGIN

Define list of alpha values

FOR each alpha value DO
   Train Ridge Regression model
   Predict on training and testing data
   Compute MSE, RMSE, and MAE
END FOR

Select alpha with minimum test MSE
Train final Ridge model using best alpha
Predict FWI values on test data

Save trained Ridge model

Evaluate model using plots and error comparison
Check for overfitting or underfitting

END
```

8.Snapshots


Predicted vs Actual (Ridge, alpha=0.001)
Predicted FWI
Actual FWI
MSE vs Alpha
Train MSE
Test MSE
MSE
Alpha


RMSE vs Alpha
Train RMSE
Test RMSE
RMSE
Alpha
MAE vs Alpha
Train MAE
Test MAE
MAE
Alpha

Section- 4

1.Flask Application for Fire Weather Index (FWI) Prediction

Flask is a lightweight Python web framework used to deploy machine learning models as web applications. In the **Tempest FWI Predictor** project, Flask is used to integrate the trained **Ridge Regression model** with an interactive web-based dashboard, enabling real-time Fire Weather Index (FWI) prediction.

The Flask application loads the previously saved **ridge.pkl** model and **scaler.pkl** to ensure that the same preprocessing and trained parameters are used during prediction. This guarantees consistency between the training and deployment phases.

2.Model and Scaler Loading

At application startup, the Flask backend checks for the presence of the trained Ridge Regression model and StandardScaler files:

- If ridge.pkl and scaler.pkl are found, they are loaded into memory.
- If not found, a temporary demo model and scaler are created to allow the dashboard to function.

This mechanism ensures robustness and uninterrupted execution of the web application.

3.User Interface and Input Handling

The Flask application provides a **single-page interactive dashboard** designed using HTML, CSS, and Bootstrap.
The dashboard allows users to enter meteorological and fuel-related parameters such as:

- Date information (day, month, year)
- Temperature, humidity, wind speed, and rainfall
- Fire fuel indices (FFMC, DMC, DC, ISI, BUI)
- Region information

Users submit the input values through a form, which sends the data to the Flask backend using a **POST request**.

4.Prediction Process

Once the input data is received:

- The input values are arranged in the same order used during model training.
- The saved StandardScaler is applied to normalize the input features.
- The scaled data is passed to the trained Ridge Regression model.
- The model predicts the **Fire Weather Index (FWI)** value.

The predicted FWI value is returned to the frontend in **JSON format** and displayed instantly on the dashboard.

5.Prediction Visualization and History

The dashboard also maintains an **in-memory prediction history**, which is visualized using a dynamic line chart.
Each new prediction is appended to the chart, allowing users to observe changes in predicted FWI values over time. This enhances interpretability and user interaction.

6.Deployment and Execution

The Flask application runs locally on:

http://127.0.0.1:5000/

It operates without external tunneling services (such as ngrok), making it suitable for local demonstration and academic project evaluation.

7.Advantages of Using Flask for Deployment

- Enables real-time prediction of FWI values
- Provides a user-friendly interface for non-technical users
- Ensures consistency between training and deployment
- Demonstrates end-to-end machine learning implementation

8.Short Pseudocode: Flask Application

BEGIN

Load trained Ridge model and scaler
Initialize Flask application

Create dashboard for user input
Accept input data from user

Scale input using saved scaler

Predict FWI using trained model

Return prediction to dashboard
Display prediction and update history chart

END

9.Snapshots