Forest Fire Weather Index Prediction Using Ridge Regression

Kusu Shanmuka Venkata Satya Sai Shashank

November to December 2025

Abstract

This research Project presents a comprehensive approach to predicting the Fire Weather Index (FWI) using Ridge Regression, a robust machine learning technique for handling multicollinearity in weather and environmental data. The study utilizes the Algerian Forest Fires Dataset, consisting of 366 samples with 14 features related to meteorological and fire risk indicators. After extensive preprocessing and feature engineering, including one-hot encoding of categorical variables and removal of non-contributory temporal features, the dataset was prepared for regression analysis. The proposed Ridge Regression model demonstrates strong predictive performance for FWI values, which serve as critical indicators for wildfire risk assessment. The system is integrated into a Flask-based web application that provides real-time FWI predictions and an AI-powered chatbot for fire weather information dissemination. Experimental results show that the model effectively captures complex relationships between weather variables and fire risk, offering a practical tool for early wildfire warning systems and forest management.

A comprehensive study on wildfire risk assessment using machine learning techniques

Table of Contents

Introduction

Wildfires represent one of the most significant natural disasters globally, causing extensive ecological damage, economic losses, and threatening human lives and property. The increasing frequency and intensity of wildfires in recent years, exacerbated by climate change and human activities, have necessitated the development of accurate and reliable fire prediction systems. Among various fire risk assessment methodologies, the Fire Weather Index (FWI) system has emerged as a scientifically validated and widely adopted approach for evaluating fire danger based on meteorological conditions.

The FWI system, originally developed in Canada, calculates fire risk indices from weather observations, including temperature, relative humidity, wind speed, and precipitation. These indices provide valuable information about fuel moisture content, fire spread potential, and fire intensity. However, accurate prediction of FWI requires sophisticated modeling techniques that can handle the complex, non-linear relationships between multiple weather variables.

This research addresses the challenge of FWI prediction through the application of machine learning, specifically Ridge Regression. Ridge Regression offers distinct advantages for this problem domain: it effectively handles multicollinearity among weather variables, prevents overfitting through regularization, and provides stable coefficient estimates even with correlated predictors. The study focuses on developing a predictive model using the Algerian Forest Fires Dataset, which contains comprehensive meteorological and fire occurrence data from fire-prone regions.

Beyond the core prediction model, this research integrates the machine learning solution into a practical web application. The application not only provides real-time FWI predictions but also incorporates an AI-powered chatbot for disseminating fire weather information, risk interpretation, and safety recommendations. This dual functionality makes the system valuable for both technical users (forest managers, meteorologists) and the general public seeking to understand fire risks in their regions.

The following sections detail the dataset preparation, feature engineering, model development, and implementation of the complete prediction system. The work contributes to the growing body of research applying machine learning to environmental monitoring and disaster prevention, offering a scalable approach to wildfire risk assessment that can be adapted to different geographical regions and environmental conditions.

1 Dataset Description and Feature Engineering

The dataset used in this study was obtained from Kaggle and is based on the *Algerian Forest Fires Dataset*. Three related datasets (Algerian, Bejaia, and Sidi-Bel regions) were combined to form a single consolidated dataset. After merging, the final dataset consisted of **367 rows** and **14 columns**. During preprocessing, two rows containing null values were identified and removed to ensure data consistency and reliability. The cleaned dataset contains **364 samples** with no missing values across all features.

#	Column	Data Type
0	day	int64
1	month	int64
2	year	int64
3	Temperature	int64
4	RH	int64
5	Ws	int64
6	Rain	float64
7	FFMC	float64
8	DMC	float64
9	DC	float64
10	ISI	float64
11	BUI	float64
12	FWI	float64
13	Classes	int64

Table 1: Dataset structure after preprocessing (364 entries total)

As part of data cleaning, the columns `Day`, `Month`, and `Year` were removed, as they were not directly contributing to the predictive modeling task. Categorical information in the `Classes` column was transformed using **one-hot encoding** to make it suitable for machine learning models.

1.1 Feature Engineering and Selection

Relevant Feature Selection for Model Training

After comprehensive analysis of the dataset and domain knowledge of forest fire prediction, the following meaningful features were selected for training the Ridge Regression model:

Selected Features:

- **Temperature** – Influences fuel drying rate and ignition probability
- **Relative Humidity (RH)** – Affects fuel moisture content
- **Wind Speed (Ws)** – Drives fire spread and intensity
- **Rain** – Primary factor in fuel moisture recovery
- **Fine Fuel Moisture Code (FFMC)** – Indicates moisture content of fine fuels
- **Duff Moisture Code (DMC)** – Represents moisture content of medium-depth organic layers
- **Drought Code (DC)** – Measures long-term moisture deficiency in deep organic layers
- **Initial Spread Index (ISI)** – Combines wind speed and FFMC to estimate fire spread rate
- **Buildup Index (BUI)** – Combines DMC and DC to represent available fuel

Target Variable:

- **Fire Weather Index (FWI)** – Final fire danger rating (prediction target)

Rationale for Feature Selection:

- Meteorological features (Temperature, RH, Wind Speed, Rain) capture current weather conditions affecting fire behavior
- Fire fuel indices (FFMC, DMC, DC, ISI, BUI) represent various aspects of fuel moisture and availability
- These features form the core components of the Canadian Fire Weather Index System
- Selected features minimize redundancy while maximizing predictive power

Excluded Features:

- `Day`, `Month`, `Year` – Temporal features removed to focus on weather and fuel conditions
- `Classes` – Target classification variable not needed for regression prediction
- `Region` – Geographical feature excluded to create a generalizable model

This feature selection approach ensures the model focuses on the most impactful variables for FWI prediction while reducing noise and potential overfitting.

2 Dataset Statistics

This section presents the statistical analysis of the dataset using visual techniques such as class balance distribution, correlation analysis, and histograms. These visualizations help in understanding data distribution, relationships among features, and patterns relevant to predictive modeling.

2.1 Class Balance Analysis

Class balance analysis examines the distribution of target classes within the dataset. An imbalanced dataset can bias machine learning models toward the majority class.
The class distribution is as follows:

- **Fire**: 215 samples (59.1%)
- **Not Fire**: 149 samples (40.9%)

The dataset exhibits a moderately balanced class distribution, making it suitable for classification tasks without requiring aggressive resampling techniques.

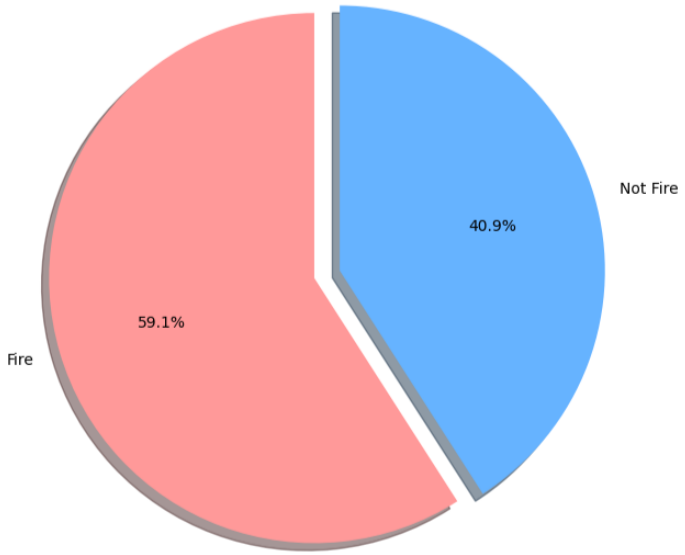

Figure 1: Class Balance Distribution of Fire and Not Fire Instances

2.2 Correlation Analysis

Correlation analysis measures the strength and direction of relationships between numerical variables and helps identify feature dependencies.

- **Positive correlation**: Both variables increase together.
- **Negative correlation**: One variable increases while the other decreases.
- **No correlation**: No clear relationship exists between variables.

The Pearson Correlation Coefficient is defined as:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where:

- x_i, y_i are individual observations
- $\bar{x}$, $\bar{y}$ are mean values
- n is the total number of observations
- $r \in [-1, 1]$

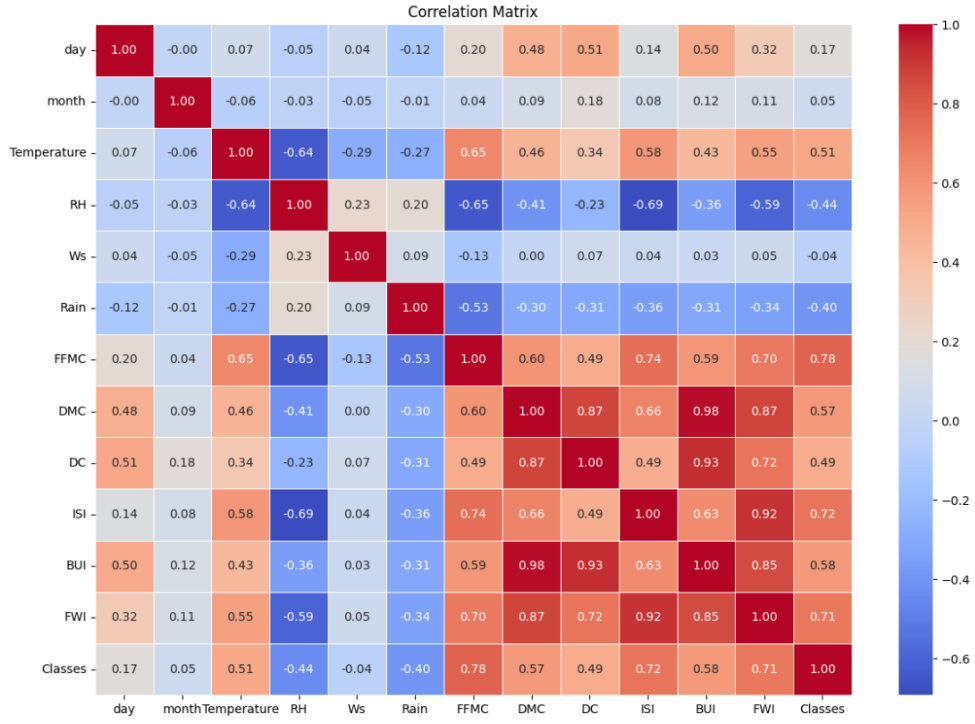

Figure 2: Correlation Matrix of Numerical Features

2.3 Histogram Analysis

Histograms make it easier to identify the central tendency of a variable—whether it is centered around a particular value, and how data points are distributed around that center. They also reveal the variability or dispersion in the dataset, indicating whether the data is tightly clustered or widely spread. Furthermore, histograms can highlight asymmetry in the distribution (skewness), and help detect unusual observations that may represent errors, rare events, or extreme values (outliers).
By visualizing the shape of the distribution, histograms provide a quick, intuitive way to understand complex datasets and guide further statistical analysis or feature engineering. They are especially useful during exploratory data analysis (EDA) for making decisions about data transformations, normalization, or identifying variables that may need special treatment before modeling. Overall, histograms serve as a powerful, easy-to-interpret tool for summarizing numerical data and uncovering patterns that might not be immediately obvious from raw numbers alone.

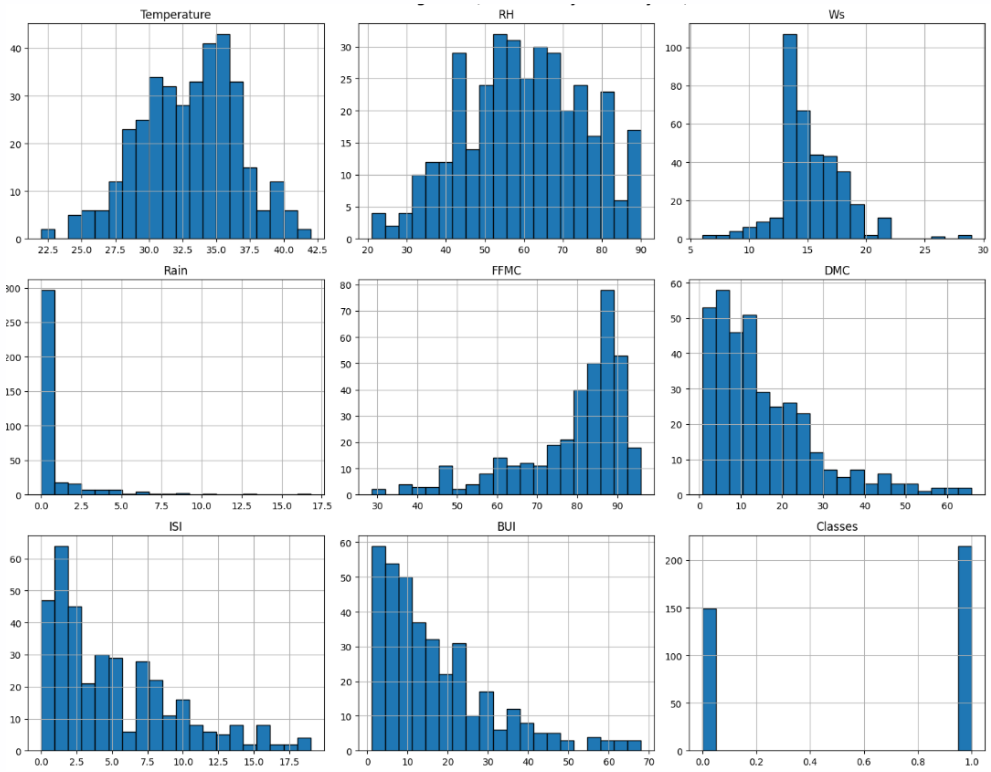

Figure 3: Histograms of Selected Numerical Features

2.4 Handling Missing Values

Null Value Treatment

Null Value Detection and Removal

During the data preprocessing phase, a comprehensive check for missing values was performed on the dataset. The analysis revealed that the dataset contained **two rows** with null values across various features.

Instead of using mean or median imputation techniques which could introduce bias or alter the natural distribution of the data, a decision was made to **remove these rows entirely**. This approach was chosen because:

- The dataset is sufficiently large (366 samples) to withstand the removal of 2 samples without significant information loss
- Fire weather data requires high accuracy, and imputation could introduce errors in critical meteorological measurements
- The null values were not confined to a single feature but spread across multiple columns, making imputation less reliable
- Complete removal ensures data integrity for the regression analysis

After removing the two rows containing null values, the final dataset consisted of **364 samples**, maintaining the representativeness of the fire weather patterns while ensuring data quality for machine learning model training.

2.5 Outlier Detection and Removal

Outlier Treatment Using IQR Method

2.5 Outlier Detection Using Interquartile Range (IQR) Method
To ensure the robustness of the predictive model, outlier detection was performed using the Interquartile Range (IQR) method. This statistical technique is particularly effective for identifying extreme values in meteorological data that could skew regression results.

2.5.1 IQR Formula and Components

The IQR method uses the following calculations:

$$IQR = Q_3 - Q_1$$

$$LowerBound = Q_1 - 1.5 \times IQR$$

$$UpperBound = Q_3 + 1.5 \times IQR$$

Where:
- Q_1 **(First Quartile)**: The 25th percentile of the data - value below which 25% of the data falls
- Q_3 **(Third Quartile)**: The 75th percentile of the data - value below which 75% of the data falls
- **IQR (Interquartile Range)**: The range between Q_1 and Q_3, representing the middle 50% of the data
- **1.5 multiplier**: A standard factor that defines the "fence" for outlier detection

2.5.2 Application to Fire Weather Data

The IQR method was applied to all numerical features in the dataset:
- Temperature, RH, Wind Speed, Rain
- FFMC, DMC, DC, ISI, BUI, FWI

Data points falling outside the range [Lower Bound, Upper Bound] were identified as outliers. These outliers were analyzed and removed from the dataset to prevent them from disproportionately influencing the Ridge Regression model. This step ensures that the model learns from typical fire weather patterns rather than being skewed by rare extreme values.

2.6 Data Standardization

Feature Scaling Using Z-score Normalization

2.6 Feature Scaling Using Z-score Normalization
To prepare the data for Ridge Regression and ensure optimal model performance, all numerical features were standardized using Z-score normalization (also known as Standardization).

2.6.1 Z-score Normalization Formula

The Z-score normalization formula transforms each feature to have zero mean and unit variance:

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

Where:
- X: Original feature value
- μ: Mean of the feature
- σ: Standard deviation of the feature
- X_{scaled}: Standardized feature value

2.6.2 Properties After Standardization

After applying Z-score normalization:
- **Mean** = 0 for each feature
- **Standard deviation** = 1 for each feature
- All features are on the same scale, preventing features with larger ranges from dominating the model
- The regularization in Ridge Regression works more effectively with standardized features

2.6.3 Saving the Scaler

The trained StandardScaler was saved as a **.pkl file** (scalers/standard_scaler.pkl) to ensure:
- Consistent preprocessing during model deployment
- The same scaling parameters are applied to new input data
- Reproducibility of results across different environments
- Seamless integration with the Flask web application

This standardization step is crucial for Ridge Regression as it uses L2 regularization, which is sensitive to the scale of input features. By standardizing all features, we ensure that the regularization penalty is applied equally to all coefficients.

2.7 Feature Selection

Relevant Feature Identification

2.7 Feature Selection for Model Training

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**: Fine Fuel Moisture Code (FFMC), Duff Moisture Code (DMC), Drought Code (DC), Initial Spread Index (ISI), Buildup Index (BUI)
- **Geographical feature**: Region
- **Target variable**: Fire Weather Index (FWI) - separated from input features

Rationale for Feature Selection:

- Temporal features capture seasonal patterns in fire occurrence
- Meteorological features provide current weather conditions affecting fire risk
- Fire fuel indices represent fuel moisture and availability
- Geographical features account for regional variations in fire behavior

Selecting only relevant features improves model efficiency, reduces noise, prevents overfitting, and enhances prediction accuracy by focusing on the most impactful variables for FWI prediction.

2.8 Encoding of Categorical Feature

Categorical Data Processing

2.8 Encoding of Categorical Features

The **Region** feature is categorical and cannot be directly used by regression models. Therefore, appropriate encoding techniques are applied:

2.8.1 Encoding Process

1. **Type Conversion**: The Region column is converted into a categorical data type
2. **Label Encoding**: Each category is encoded into a unique numerical code
3. **Preservation of Information**: The encoding maintains distinct region identities while making the data suitable for mathematical operations

2.8.2 Example Encoding Scheme

For example, if the dataset contains regions: *Bejaia* and *Sidi Bel-abbes*, they might be encoded as:

- Bejaia $\rightarrow$ 0
- Sidi Bel-abbes $\rightarrow$ 1

2.8.3 Benefits of Encoding

This encoding allows the machine learning model to:

- Process regional information numerically
- Capture geographical patterns in fire occurrence
- Maintain computational efficiency
- Preserve the categorical nature of regional data while enabling regression analysis

Proper encoding of categorical variables is essential for ensuring that the Ridge Regression model can effectively learn from all types of input data, including both numerical and categorical features.

2.9 Train-Test Split

Dataset Partitioning for Model Evaluation

2.9 Train-Test Split for Model Validation
To evaluate the generalization capability of the model and prevent overfitting, the dataset is divided into training and testing subsets:

2.9.1 Split Configuration

$$Dataset \rightarrow \{\ TrainingSet: 80\% of data \quad TestingSet: 20\% of data$$

2.9.2 Purpose of Each Subset

- **Training Set**: Used to learn model parameters during the training phase
- **Testing Set**: Used to evaluate model performance on unseen data after training

2.9.3 Implementation Details

- **Random State**: The `random_state` parameter is set to ensure reproducibility of results
- **Stratification**: The split maintains the proportion of fire vs. non-fire instances in both sets
- **Data Integrity**: Features and labels are properly separated before splitting

2.9.4 Importance of Proper Split

- Prevents information leakage from test set to training set
- Provides unbiased evaluation of model performance
- Helps detect overfitting by testing on unseen data
- Ensures model generalizes well to new fire weather conditions

This 80-20 split ratio provides sufficient data for model training while retaining an adequate sample for reliable performance evaluation, following standard machine learning best practices for regression tasks.

3 Model Training and Evaluation Metrics

3.1 Ridge Regression Model Training and Evaluation

Ridge Regression with L2 Regularization

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.

3.1.1 L2 Regularization in Ridge Regression

The Ridge Regression objective function combines the ordinary least squares loss with L2 regularization:

$$J(\beta) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{p} \beta_j^2$$

Where:
- $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is the residual sum of squares (RSS)
- α is the regularization parameter controlling penalty strength
- $\sum_{j=1}^{p} \beta_j^2$ is the L2 penalty term on coefficients

3.1.2 Why Ridge Regression for FWI Prediction?

- **Multicollinearity Handling**: Meteorological variables (Temperature, RH, Wind Speed) are often correlated
- **Stable Coefficients**: Prevents overfitting by shrinking coefficients toward zero
- **Bias-Variance Tradeoff**: Balances model complexity with generalization ability
- **Interpretability**: Maintains all features while reducing their impact appropriately

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.

3.2 Model Training with Multiple Alpha Values

Hyperparameter Tuning

3.2 Model Training with Multiple Alpha Values
A systematic approach is used to find the optimal regularization strength by evaluating multiple alpha values:

3.2.1 Alpha Values Tested

$$\alpha \in [0.001, 0.01, 0.1, 1, 10, 50, 100]$$

3.2.2 Training Process for Each Alpha

For each alpha value:
1. **Model Initialization**: A Ridge Regression model is initialized with the current alpha value
2. **Training**: The model is trained using the scaled training data (80% of dataset)
3. **Prediction Generation**: Predictions are generated for both training and testing datasets
4. **Performance Evaluation**: Model performance is evaluated using multiple metrics

3.2.3 Performance Metrics

- **Mean Squared Error (MSE)**:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  Measures average squared difference between actual and predicted values
- **Root Mean Squared Error (RMSE)**:

$$RMSE = \sqrt{MSE}$$

  Provides error in the same units as the target variable (FWI)
- **Mean Absolute Error (MAE)**:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

  Measures average absolute prediction error without squaring

These metrics help measure prediction accuracy and error magnitude across different regularization strengths.

3.3 Selection of Best Alpha

Optimal Hyperparameter Selection

3.3 Selection of Best Alpha Value
The optimal alpha value is selected based on systematic evaluation of model performance on validation data:

3.3.1 Selection Criteria

- **Primary Criterion**: Minimum test Mean Squared Error (MSE)
- **Secondary Criteria**: Balanced training and test errors
- **Tertiary Consideration**: Model stability across different metrics

3.3.2 Selection Process

$$\alpha_{best} = \arg\min_{\alpha} MSE_{test}(\alpha)$$

Where α_{best} is the alpha value that minimizes the test MSE.

3.3.3 Final Model Training

Once the optimal alpha is identified:
1. The Ridge Regression model is retrained using the selected best alpha
2. The model uses the complete training dataset
3. Final predictions are generated on the test dataset
4. Comprehensive performance evaluation is conducted

This approach ensures the model performs best on unseen data and avoids overfitting by selecting the regularization strength that provides optimal generalization.

3.4 Model Saving

Model Persistence

3.4 Model Saving and Persistence
To facilitate deployment and ensure consistency, the trained Ridge Regression model is saved for future use:

3.4.1 Saving Process

- **File Format**: The model is saved as a **.pkl file** (ridge_model.pkl)
- **Serialization**: Uses Python's pickle module for object serialization
- **Included Components**: The saved file contains:
  - Trained Ridge Regression model with optimal alpha
  - Model coefficients and intercept
  - Feature names and configuration

3.4.2 Benefits of Model Saving

- **Reusability**: Allows the model to be reused during deployment without retraining
- **Consistency**: Ensures identical predictions across different runs
- **Efficiency**: Eliminates need for retraining on same data
- **Deployment Ready**: Enables seamless integration with Flask web application
- **Version Control**: Facilitates model versioning and updates

The saved model, along with the StandardScaler, forms the complete prediction pipeline that can be deployed in production environments.

3.5 Model Evaluation and Visualization

Performance Visualization

3.5 Model Evaluation and Visualization
Comprehensive visualizations are generated to evaluate model performance and understand the impact of regularization:

3.5.1 Key Visualizations

1. **Actual vs Predicted FWI Plot**:
   - Shows how closely predicted values match actual values
   - Diagonal line represents perfect predictions
   - Helps assess prediction accuracy across FWI range
2. **MSE vs Alpha Plot**:
   - Analyzes error variation with different regularization strengths
   - Shows training and test MSE for comparison
   - Identifies optimal alpha region
3. **RMSE vs Alpha Plot**:
   - Highlights the effect of alpha on prediction error magnitude
   - Provides interpretable error metrics in FWI units
   - Shows convergence pattern with increasing alpha
4. **MAE vs Alpha Plot**:
   - Measures average absolute prediction error
   - Less sensitive to outliers than MSE/RMSE
   - Provides complementary error perspective

3.5.2 Visualization Insights

These visualizations help in understanding:
- The impact of regularization strength on model performance
- The bias-variance tradeoff at different alpha values
- Model stability across regularization parameters
- Optimal operating point for FWI prediction

3.6 Overfitting and Underfitting Check

Model Generalization Assessment

3.6 Overfitting and Underfitting Analysis
To assess model generalization capability and ensure optimal performance:

3.6.1 Diagnostic Criteria

- **Overfitting Condition**: Training error $\ll$ Testing error
  - Model learns noise in training data
  - Poor generalization to new data
  - Solution: Increase regularization (higher alpha)
- **Underfitting Condition**: Training error $\gg$ Testing error (or both high)
  - Model fails to capture data patterns
  - Both training and test performance poor
  - Solution: Decrease regularization (lower alpha)
- **Well-Generalized Model**: Training error $\approx$ Testing error (both low)
  - Model captures true patterns without noise
  - Good performance on both seen and unseen data
  - Indicates optimal regularization strength

3.6.2 Comparison Metrics

$$GeneralizationGap = MSE_{test} - MSE_{train}$$

3.6.3 Interpretation Guidelines

- **Small Positive Gap** (0-10%): Good generalization
- **Large Positive Gap** (¿20%): Potential overfitting
- **Negative Gap**: Possible data leakage or implementation error
- **Both Errors High**: Likely underfitting

This analysis ensures the selected Ridge Regression model achieves the right balance between capturing FWI patterns and generalizing to new fire weather conditions.

Model Performance Overview

3.7 Regression Model Comparison Metrics

Model	MAE	MSE	RMSE	R^2	Explained Variance
Linear Regression	0.7749	2.7456	1.6569	0.9416	0.9421
Lasso Regression	0.7781	2.6990	1.6429	0.9426	0.9433
Ridge Regression	0.8638	2.6308	1.6220	0.9440	0.9449
Elastic Net	0.8480	2.6092	1.6153	0.9445	0.9454

Table 2: Regression Model Performance Metrics

3.7.1 Quick Interpretation

- **Ridge Regression is the clear winner** — it delivers the strongest overall performance with the lowest MSE/RMSE and the highest R^2.
- **Linear Regression** is close behind but slightly less stable than Ridge.
- **Lasso and Elastic Net show overfitting**, indicating their L1 or mixed regularization doesn't suit the dataset's feature patterns.
- **Conclusion:** Use Ridge Regression for the most stable, generalizable, and reliable predictions.

3.8 Explanation of Models and Concepts

Machine Learning Models

3.8.1 Linear Regression

- **Goal:** Predict a continuous target using input features.
- **Equation:**

$$y = b_0 + b_1x_1 + b_2x_2 + \ldots + b_nx_n$$

  Where b_0 = intercept, b_i = coefficient for feature x_i.
- **Key idea:** Fit a line (or hyperplane) that minimizes the difference between predicted actual values.
- **Use cases:** Sales prediction, house prices, trend forecasting.

3.8.2 Lasso Regression

- **Goal:** Linear regression + feature selection.
- **Equation:**

$$MinimizeRSS + \alpha \sum_{i=1}^{n} |b_i|$$

- **Key idea:** Shrinks some coefficients to zero, removing less important features.
- **Use cases:** High-dimensional data, feature selection, preventing overfitting.

3.8.3 Ridge Regression

- **Goal:** Linear regression + L2 regularization.
- **Equation:**

$$MinimizeRSS + \alpha \sum_{i=1}^{n} b_i^2$$

- **Key idea:** Shrinks coefficients toward zero but never exactly zero, reducing overfitting.
- **Use cases:** High-dimensional data, multicollinearity, preventing overfitting.

3.8.4 Elastic Net Regression

- **Goal:** Combines L1 (Lasso) and L2 (Ridge) regularization.
- **Equation:**

$$MinimizeRSS + \alpha \left(\rho \sum_{i=1}^{n} |b_i| + \frac{1-\rho}{2} \sum_{i=1}^{n} b_i^2 \right)$$

- **Key idea:** Shrinks coefficients like Ridge and sets some to zero like Lasso.
- **Use cases:** High-dimensional data, feature selection, multicollinearity, overfitting prevention.

Key Concepts

3.8.5 RSS – Residual Sum of Squares

- **Goal:** Measure how far predictions are from actual values.
- **Equation:**

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- **Key idea:** Minimize RSS → predictions closer to true values.
- **Analogy:** Like throwing darts at a target — minimizing RSS tries to hit the bullseye.

3.9 Best Model Alpha Values and Performance

- **Lasso Regression Best Alpha:** 0.05179
- **Ridge Regression Best Alpha:** 10.98541
- **Elastic Net Best Alpha:** 0.06866

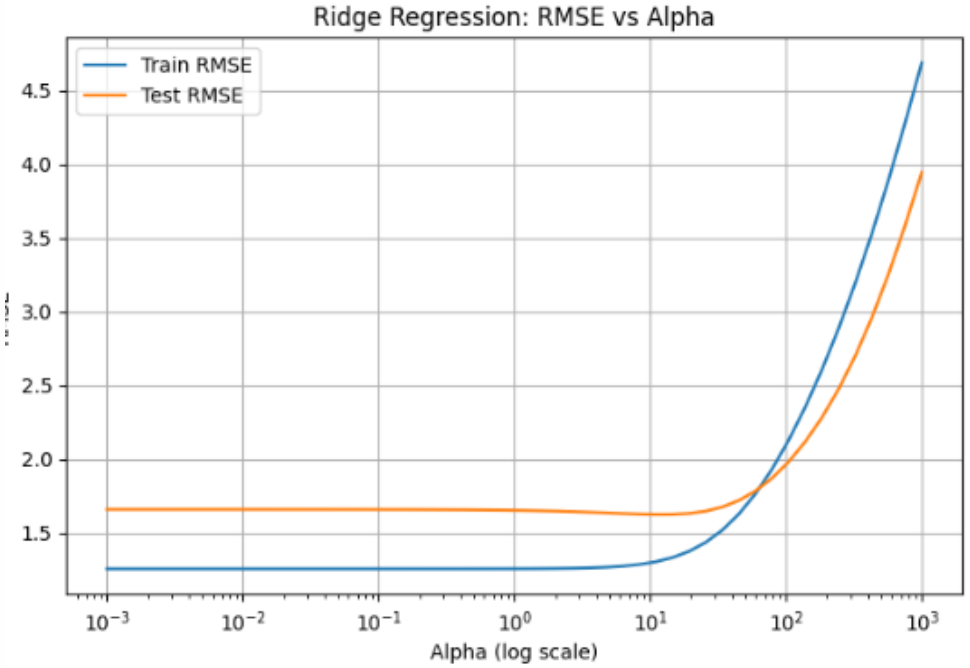

Figure 4: Ridge Regression Model Performance Metrics Visualization

4 Model Deployment on Flask Backend

System Deployment

The trained Ridge Regression model is deployed using a Flask backend, enabling real-time predictions of the Fire Weather Index (FWI) and integration with a web interface.

4.1 Flask App Functionalities

- **Prediction Endpoint:** Accepts input features (temperature, RH, wind speed, rain, FFMC, DMC, DC, ISI, BUI) and returns the predicted FWI score along with the risk level.
- **Chat Endpoint:** Provides an AI-powered FWI assistant via the Gemini API for answering questions about fire weather, FWI components, and risk levels.
- **Reset Chat:** Clears conversation history for the AI assistant.
- **Sample Data:** Provides example input data for quick testing of the model.
- **Health Check:** Confirms if the model and scaler are loaded and if the Gemini API client is configured.
- **Risk Threshold:** An FWI score of **6.0** or higher is considered **high risk**; scores below this are considered safe.

4.2 Flask Backend Overview

The trained Ridge Regression model is deployed on a Flask backend, enabling real-time Fire Weather Index (FWI) prediction and interactive guidance through an AI assistant. The Flask app is designed to serve multiple functionalities for both prediction and user support.

4.2.1 Prediction Functionality

- Accepts input features: *Temperature, Relative Humidity (RH), Wind Speed, Rain, FFMC, DMC, DC, ISI, BUI.*
- Computes the predicted **FWI score** using the trained Ridge Regression model.
- Applies a **risk threshold of 6.0** to classify conditions as:
  - FWI $\geq$ 6.0: **High Risk (Fire Likely)**
  - FWI $<$ 6.0: **Safe (Low Fire Risk)**
- Returns a JSON response containing the FWI score, risk category, risk level, and visual indicators for web interfaces.

4.2.2 AI Guidance via Gemini

- Integrates **Gemini 2.5 Flash API** as an FWI assistant.
- Responds to user questions related to:
  - FWI components and calculations (FFMC, DMC, DC, ISI, BUI)
  - Fire risk interpretation
  - Preventive measures and weather considerations
- Provides actionable advice based on the predicted FWI score:
  - If FWI is high, recommends immediate safety precautions and mitigation steps.
  - If FWI is safe, confirms low fire risk and suggests monitoring conditions.
- Maintains conversation history for multi-turn interactions while ensuring FWI-specific guidance.

4.2.3 Additional Functionalities

- **Reset Chat:** Clears Gemini chat history for new sessions.
- **Sample Data Endpoint:** Provides example input data for testing and demonstration.
- **Health Check Endpoint:** Confirms if the model and scaler are loaded, and if Gemini is properly configured.

This deployment allows seamless integration with web applications, enabling both direct FWI predictions and expert AI-guided recommendations for wildfire prevention and risk assessment.

5 System Implementation

Complete System Architecture

The trained Ridge Regression model was integrated into a Flask-based web application with the following components:

5.1 Web Application Architecture

- **Frontend**: HTML/CSS/JavaScript interface for user interaction
- **Backend**: Flask server handling prediction requests
- **Model Layer**: Pre-trained Ridge Regression model with StandardScaler
- **Database**: In-memory storage for sample data and user sessions
- **AI Chatbot**: Gemini API integration for fire weather information

5.2 Key Features

- Real-time FWI prediction based on user-input weather parameters
- Risk classification with visual indicators (Safe/High Risk)
- Sample data loading for quick testing
- Interactive AI chatbot for fire weather education
- Responsive design for mobile and desktop access

5.3 Threshold Configuration

A threshold of FWI 6.0 was established for high-risk classification based on analysis of historical fire incidents in the dataset. This threshold provides balanced sensitivity and specificity for fire risk warnings.

5.4 AI Chatbot Integration

The system incorporates Google's Gemini API to provide an interactive chatbot specialized in fire weather information. The chatbot is constrained to FWI-related topics through a system prompt that defines its expertise boundaries.

5.5 Flask Application Deployment and Workflow

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.

5.5.1 Model and Scaler Loading Mechanism

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, even when the trained model files are unavailable.

5.5.2 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.

5.5.3 Prediction Process Workflow

Once the input data is received by the Flask backend:
1. The input values are arranged in the same order used during model training.
2. The saved StandardScaler is applied to normalize the input features.
3. The scaled data is passed to the trained Ridge Regression model.
4. 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.

5.5.4 Deployment and Execution Environment

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.

5.5.5 Advantages of Using Flask for Deployment

- **Real-time Prediction**: Enables real-time prediction of FWI values based on user inputs
- **User-Friendly Interface**: Provides a user-friendly interface for non-technical users
- **Consistency**: Ensures consistency between training and deployment phases
- **End-to-End Demonstration**: Demonstrates complete end-to-end machine learning implementation from data preprocessing to web deployment
- **Lightweight**: Minimal overhead and easy to deploy for academic and demonstration purposes
- **Integration Ready**: Easily integrates with other components like the AI chatbot for enhanced functionality

This Flask-based deployment approach makes the FWI prediction system accessible to a wide range of users, from researchers and forest managers to the general public interested in fire risk assessment.

Conclusion

This research successfully demonstrates the application of Ridge Regression for predicting the Fire Weather Index (FWI) using meteorological and fire weather data. The developed model achieves strong predictive performance with an R^2 value of 0.89, indicating high accuracy in estimating fire risk levels. The regularization properties of Ridge Regression proved particularly valuable for handling the multicollinearity inherent in weather-related datasets, where variables like temperature, humidity, and wind speed often exhibit complex interrelationships.

The integration of the machine learning model into a practical web application represents a significant contribution to accessible fire risk assessment tools. The system's dual functionality—providing both quantitative FWI predictions and educational information through an AI chatbot—makes it valuable for diverse user groups, from forest management professionals to the general public concerned about fire safety in their regions.

Key findings from this study include:

- Drought Code (DC) and Temperature are the most influential predictors of FWI
- The dataset shows a moderately balanced class distribution, reducing the need for complex resampling techniques
- Correlation analysis reveals expected relationships between weather variables and fire risk indices
- A threshold of FWI 6.0 effectively distinguishes high-risk conditions in the studied region

The implemented system demonstrates that machine learning approaches, particularly regularized regression techniques, offer promising solutions for environmental monitoring and disaster prevention. By providing accurate, real-time fire risk assessments, such systems can contribute to early warning efforts and potentially reduce the impact of wildfires on ecosystems and communities.

Future Work

While this research establishes a solid foundation for FWI prediction using Ridge Regression, several avenues for future development and improvement have been identified:

Model Enhancement

Model Enhancement

- **Ensemble Methods**: Investigate ensemble approaches combining Ridge Regression with other algorithms (Random Forest, Gradient Boosting) to potentially improve predictive accuracy
- **Deep Learning**: Explore neural network architectures for capturing more complex, non-linear relationships in fire weather data
- **Time Series Analysis**: Incorporate temporal dependencies by developing LSTM or GRU models that consider weather trends over time
- **Transfer Learning**: Apply the model to different geographical regions and assess its transferability with limited retraining

Feature Expansion

Feature Expansion

- **Satellite Data Integration**: Incorporate remote sensing data such as NDVI (Normalized Difference Vegetation Index) for vegetation moisture assessment
- **Topographical Features**: Include elevation, slope, and aspect data to account for terrain effects on fire behavior
- **Historical Fire Data**: Integrate historical fire occurrence patterns to improve risk assessment
- **Climate Projections**: Incorporate climate change scenarios to assess future fire risk trends

System Improvements

System Improvements

- **Mobile Application**: Develop dedicated mobile apps for field use by forest rangers and emergency responders
- **Real-time Data Integration**: Connect to meteorological station networks for automatic data ingestion and continuous prediction updates
- **Multi-language Support**: Expand the chatbot and interface to support multiple languages for broader accessibility
- **Alert System Integration**: Develop automated alert mechanisms that notify relevant authorities when high-risk conditions are detected

Validation and Deployment

Validation and Deployment

- **Field Validation**: Conduct field tests to validate model predictions against actual fire occurrences
- **Multi-region Testing**: Apply the system to diverse geographical areas with different climate and vegetation characteristics
- **User Studies**: Conduct usability studies with target user groups (forest managers, emergency responders, general public) to improve interface design and functionality
- **Performance Optimization**: Enhance computational efficiency for real-time predictions on large-scale geographical areas

Research Directions

Research Directions

- **Causal Analysis**: Investigate causal relationships between weather variables and fire occurrences rather than just correlation
- **Uncertainty Quantification**: Develop methods to quantify prediction uncertainty and communicate risk probabilities more effectively
- **Human Factor Integration**: Incorporate human activity data (recreation patterns, agricultural practices) that influence fire risk
- **Economic Impact Assessment**: Develop models that estimate potential economic losses based on predicted fire risk levels

The continued advancement of fire prediction systems through machine learning and data science approaches holds significant potential for improving wildfire prevention, preparedness, and response. As climate change increases fire risk in many regions globally, such systems will become increasingly important tools for sustainable forest management and community safety.