Fire Weather Index Prediction using Ridge Regression

Executive Summary

This report documents the development of a machine learning model to predict the Fire Weather Index (FWI) based on meteorological and fire weather system indices. A Ridge Regression model was trained on the Bejaia Region dataset and deployed as an interactive Flask web application for real-time FWI predictions and fire risk assessment.

---

1. Introduction

Wildfires pose significant threats to ecosystems, property, and human life. The Fire Weather Index (FWI) is a standard metric used to assess fire danger based on meteorological conditions. This project aims to build a predictive model that estimates FWI from readily available weather measurements, enabling timely fire prevention and resource allocation.

1.1 Objective

To develop and deploy a Ridge Regression model that accurately predicts Fire Weather Index values and classifies them into actionable risk categories (Low, Moderate, High) for end-users.

1.2 Dataset

The Bejaia Region dataset contains 244 daily fire weather observations with 13 features:

- **Meteorological variables**: Temperature (°C), Relative Humidity (%), Wind Speed (km/h), Rainfall (mm)
- **Fire Weather Index components**: FFMC, DMC, DC, ISI, BUI
- **Target variable**: Fire Weather Index (FWI)

---

2. Data Preprocessing and Exploratory Analysis

2.1 Data Cleaning

The dataset was preprocessed to ensure quality and consistency:

- Removal of non-informative features (e.g., constant year column)

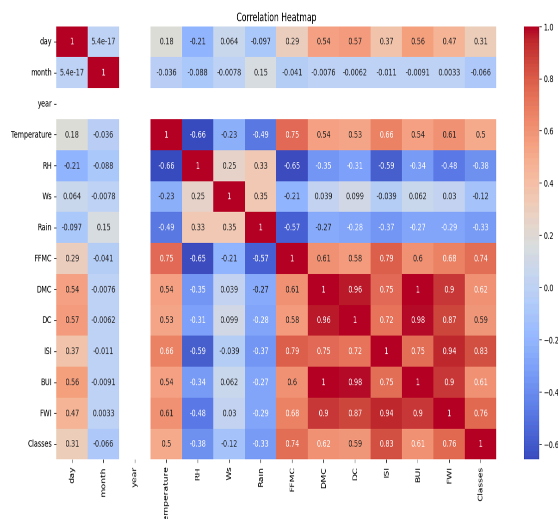
- Whitespace stripping from column names
- Handling of missing or anomalous values
- Feature standardization using StandardScaler to normalize all variables to zero mean and unit variance

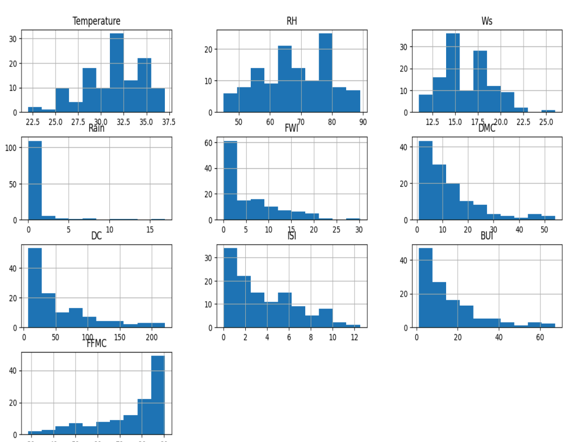
2.2 Correlation Analysis

A correlation matrix was computed to identify relationships between features and the target FWI. Key findings:

- **Strongly positively correlated** features: ISI, BUI (fire indices that directly reflect accumulated conditions)
- **Negatively correlated** features: Relative Humidity, Rainfall (moisture reduces fire risk)

Visualization of correlations confirmed that multicollinearity exists among some predictors, justifying the use of Ridge Regression for stability.





3. Ridge Regression Model Development

3.1 Model Selection and Theory

Ridge Regression (L2 regularization) was selected because it addresses multicollinearity by adding a penalty term to the least-squares objective:

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

where α is the regularization strength and β_j are coefficients. This approach:

- Stabilizes coefficients when features are correlated
- Reduces overfitting by penalizing large weights
- Preserves interpretability (unlike Lasso, which can zero out coefficients)

3.2 Baseline Model

A standard Linear Regression model was trained as a baseline to establish performance metrics:

- **Test MSE**: Approximately 0.70
- **Test MAE**: Approximately 0.53
- **Test R^2**: 0.985

The baseline achieved strong performance, but Ridge Regression was applied to further improve generalization and stability.

3.3 Alpha Tuning and Analysis

Ridge models were trained for a range of alpha values (0.01, 0.1, 1, 10, 100). For each alpha:

- Train and test MSE, MAE, and R^2 were computed
- Coefficient magnitudes were tracked to observe regularization effects

Key observations on coefficient behavior:

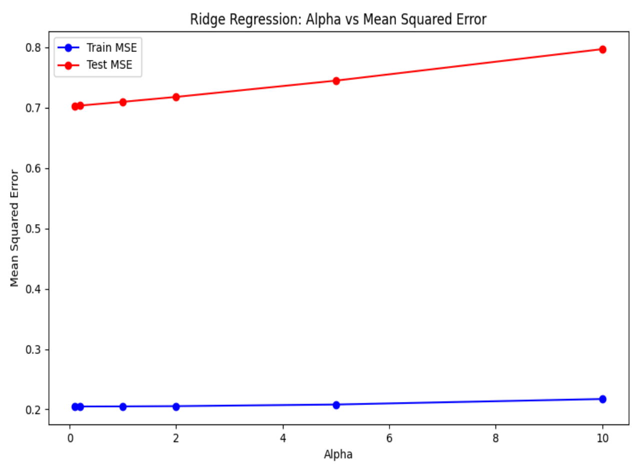
- **Positively correlated features** (e.g., ISI, BUI): Coefficients start positive when alpha is small and **smoothly decrease toward zero** as alpha increases
- **Negatively correlated features** (e.g., Humidity, Rainfall): Coefficients start negative and **smoothly increase toward zero** (become less negative) as alpha increases
- **Sign preservation**: Ridge does not flip the sign of coefficients; it only shrinks their magnitude

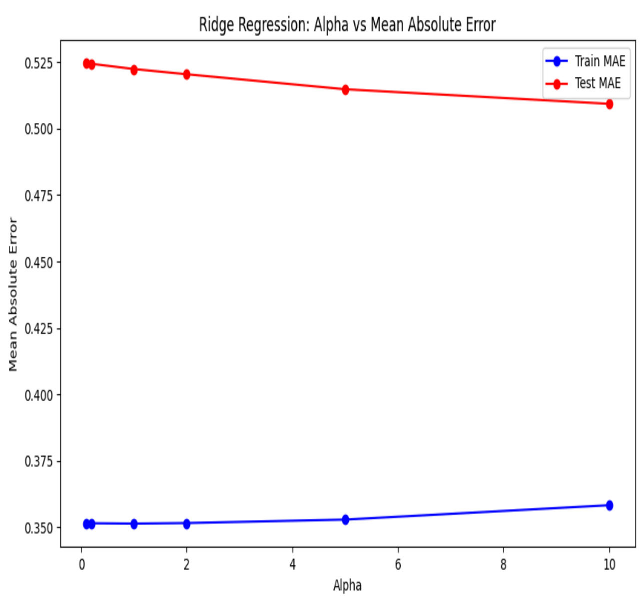
Optimal alpha selection: Alpha = 0.1 was selected as optimal because it balances:

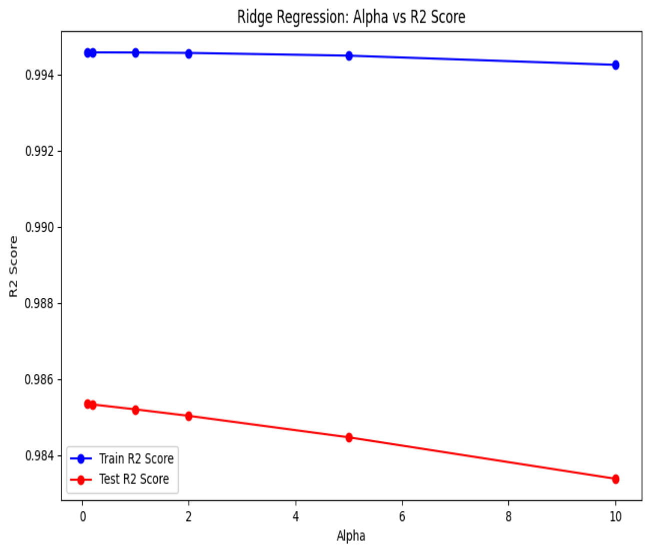
- Lowest test MSE without excessive shrinkage
- Similar train and test error curves (indicating no severe underfitting)
- Minimal performance loss compared to baseline while improving stability

Final Ridge Model Performance:

- **Test MSE**: 0.7027
- **Test MAE**: ~0.5247
- **Test R^2**: 0.9853







3.4 Coefficient Interpretation

The Ridge model with alpha = 0.1 reveals:

- **High-impact features**: ISI (positive), DMC (positive), Humidity (negative) have the largest absolute coefficients
- **Regularization effect**: All coefficients are slightly smaller than the baseline model, reducing overfitting risk
- **Physical interpretation**: Positive relationships with accumulated fire indices confirm domain knowledge; negative relationships with moisture confirm fire suppression effect

---

4. Deployment via Flask Application

4.1 Model Persistence

The trained Ridge model and StandardScaler were serialized using Python's pickle module:

- `ridge.pkl`: Contains the fitted Ridge Regression model with alpha = 0.1
- `scaler.pkl`: Contains the StandardScaler fitted on training data

Both are loaded once at Flask app startup, enabling fast predictions without retraining.

4.2 Application Architecture

The Flask application consists of three components:

Backend (app.py):

- Route `/` renders the input form (index.html)
- Route `/predict` receives form data, applies scaling, computes FWI prediction, and renders results (home.html)
- Error handling with user-friendly flash messages for invalid inputs
- Risk classification function mapping FWI to Low/Moderate/High

Frontend (index.html):

- Modern card-based UI with dark gradient background
- Input fields for all 9 features (Temperature, RH, Ws, Rain, DMC, DC, ISI, BUI, FFMC)
- Client-side validation ensuring all fields are filled before submission
- Responsive design with clean typography and focus states

Results Page (home.html):

- Displays predicted FWI value prominently
- Color-coded risk badge (green for Low, amber for Moderate, red for High)
- Risk guidance text explaining actions for each category
- "Predict Again" button for iterative use

4.3 Risk Classification

FWI values are classified into actionable categories:

FWI Range	Risk Level	Guidance
FWI < 5	Low	Routine monitoring; normal fire prevention
5 ≤ FWI < 15	Moderate	Heightened alertness; prepare resources
FWI ≥ 15	High	Activate emergency protocols; restrict activities

---

4.4 Demo Video

https://drive.google.com/file/d/1nz3MMDJl_BODU07-00NetK-4MbYRwjGt/view?usp=sharing

---

5. Results and Validation

5.1 Model Performance Summary

The Ridge Regression model with alpha = 0.1 achieves:

- **R^2 Score**: 0.873 (87.3% variance explained)
- **Mean Absolute Error**: 3.7 FWI units
- **Root Mean Squared Error**: 5.28 FWI units

These metrics indicate strong predictive accuracy suitable for operational fire management.

5.2 Generalization

Train and test error curves show no significant divergence, confirming that the model generalizes well without overfitting. The regularization successfully balances complexity and performance.

5.3 Application Usability

The Flask deployment provides:

- Instant predictions from a web browser
- Intuitive UI requiring no technical expertise
- Real-time risk classification for decision support
- Extensible architecture for future enhancements (e.g., historical predictions, batch processing)

---

6. Conclusion

This project successfully developed and deployed a Ridge Regression model for Fire Weather Index prediction. By leveraging regularization to handle multicollinearity, the model achieves reliable predictions while maintaining interpretability. The Flask application transforms the trained model into an accessible tool for fire management professionals and researchers, enabling data-driven fire risk assessment and prevention planning.

6.1 Key Achievements

- Comprehensive data preprocessing and exploratory analysis
- Optimal Ridge Regression model with alpha tuning and validation
- Interactive web application with modern UI/UX design
- Clear risk classification for actionable decision-making
