Tempest Fire Weather Index (FWI) Predictor Report

Infosys Springboard Virtual Internship

SHAIK MOHAMMAD IRFAN

December,2025

Contents

1 Dataset Description and Preprocessing

1.1 Dataset Overview

The dataset used in this project is derived from forest fire observations in Algeria, specifically from the Bejaia and Sidi-Bel Abbes regions during June to September 2012. It consists of 244 entries with 15 columns, focusing on meteorological and fire weather index components to predict the Fire Weather Index (FWI), a key metric for assessing fire danger.

The features include:

- **Region**: Binary encoded (0 for Bejaia, 1 for Sidi-Bel Abbes), indicating the geographical area.
- **day, month, year**: Temporal features (all entries from 2012, months 6-9).
- **Temperature**: Air temperature in °C (range: 22-42°C).
- **RH**: Relative humidity in % (range: 21-90%).
- **Ws**: Wind speed in km/h (range: 6-29 km/h).
- **Rain**: Rainfall in mm (range: 0-16.8 mm).
- **FFMC**: Fine Fuel Moisture Code (range: 28.6-96.0), indicating moisture in fine fuels.
- **DMC**: Duff Moisture Code (range: 0.7-65.9), for deeper organic layers.
- **DC**: Drought Code (range: 6.9-177.3), tracking long-term dryness.
- **ISI**: Initial Spread Index (range: 0.0-19.0), estimating fire spread rate.
- **BUI**: Buildup Index (range: 1.1-68.0), combining DMC and DC.
- **FWI**: Fire Weather Index (target variable, range: 0.0-31.1), overall fire danger rating.
- **Classes**: Binary (1 for fire, 0 for no fire), derived from observations.

1.2 Preprocessing Steps

The raw dataset required cleaning and preparation for modeling:

- **Column Stripping and Numeric Conversion**: Column names were stripped of whitespace, and features were converted to numeric types, coercing errors to NaN.
- **Handling Missing Values**: Rows with missing 'day' were dropped. Remaining missing values in numeric columns were filled using linear interpolation (mean of neighbors) with both forward and backward limits.
- **Encoding Categorical Variables**: 'Region' mapped to 0 (Bejaia) or 1 (Sidi-Bel Abbes). 'Classes' stripped and converted to binary (1 for 'fire', 0 otherwise).
- **Type Casting**: Integers for day, month, year, Temperature, RH, Ws; floats for others.
- **Standardization/Normalization**: For modeling, features were standardized using StandardScaler from scikit-learn, ensuring mean=0 and standard deviation=1 for each feature in the training set. This was crucial for Ridge regression to handle scale differences and multicollinearity.

The cleaned dataset was saved as 'Cleaned_FWI_dataset.csv' for further analysis.

1.2.1 Pseudo-Code for Data Preprocessing

Algorithm 1 Data Preprocessing Pipeline

Require: Raw CSV: 'FWI-UPDATE.csv'
Ensure: Cleaned CSV: 'Cleaned-FWI-dataset.csv'

Load dataframe
$df \leftarrow$ pd.read-csv('FWI-UPDATE.csv', skiprows=1)
Strip column names
$df.columns \leftarrow [col.strip()$ for col in $df.columns]$
Define numeric columns
$numericcols \leftarrow [' day',' month',' year',' Temperature',' RH',' Ws',$
$' Rain',' FFMC',' DMC',' DC',' ISI',' BUI',' FWI']$
for each col in $numericcols$ **do**
    $df[col] \leftarrow$ pd.to-numeric($df[col]$, errors='coerce')
end for
Drop missing 'day' rows
$df \leftarrow df.dropna(subset = [' day']).resetindex(drop = True)$
Insert and encode Region
$df.insert(0,' Region',' Bejaia')$
$df.loc[122 :,' Region'] \leftarrow' Sidi - Bel'$
Encode Classes
$df[' Classes'] \leftarrow df[' Classes'].astype(str).str.strip()$
$df[' Classes'] \leftarrow np.where(df[' Classes'] ==' fire', 1, 0)$
Interpolate missing values
$df[numericcols] \leftarrow df[numericcols].interpolate(method =' linear', limitdirection =' both')$
Cast integer types
df[['day','month','year','Temperature','RH','Ws']] $\leftarrow df[[' day',' month',' year',' Temperature',' RH',' Ws']].astype(int)$
Encode Region numerically
$df[' Region'] \leftarrow df[' Region'].map(\{' Bejaia' : 0,' Sidi - Bel' : 1\}).astype(int)$
Save cleaned dataset
$df.tocsv(' Cleaned - FWI - dataset.csv', index = False)$

2 Dataset Statistics

2.1 Descriptive Statistics

The dataset's key statistics are summarized below (from pandas describe()):

	Region	day	month	year	Temperature	RH	Ws	Rain
count	244.00	244.00	244.00	244.00	244.00	244.00	244.00	244.00
mean	0.50	15.75	7.50	2012.00	32.17	62.04	15.50	0.76
std	0.50	8.83	1.11	0.00	3.63	14.83	2.81	2.00
min	0.00	1.00	6.00	2012.00	22.00	21.00	6.00	0.00
25%	0.00	8.00	7.00	2012.00	30.00	52.75	14.00	0.00
50%	0.50	16.00	7.50	2012.00	32.00	63.00	15.00	0.00
75%	1.00	23.00	8.00	2012.00	35.00	73.25	17.00	0.50
max	1.00	31.00	9.00	2012.00	42.00	90.00	29.00	16.80

Table 1: Descriptive statistics for basic features.

	FFMC	DMC	DC	ISI	BUI	FWI	Classes
count	244.00	244.00	244.00	244.00	244.00	244.00	244.00
mean	77.89	14.68	49.43	4.74	16.66	7.05	0.56
std	14.23	12.39	47.67	4.15	14.20	7.43	0.50
min	28.60	0.70	6.90	0.00	1.10	0.00	0.00
25%	72.08	5.80	12.30	1.40	6.00	0.70	0.00
50%	83.50	11.30	33.10	3.50	12.25	4.45	1.00
75%	88.30	20.80	69.08	7.20	22.53	11.38	1.00
max	96.00	65.90	177.30	19.00	68.00	31.10	1.00

Table 2: Descriptive statistics for fire indices and classes.

The dataset comprises 244 observations across two regions in Algeria during the fire-prone summer months of 2012. Key descriptive statistics highlight the central tendencies, dispersion, and distribution shapes of the variables.

From the pandas .describe() output and additional computations:
- Mean Values: Average Temperature is approximately 32.17°C, RH 62%, Wind Speed 15.5 km/h, and Rainfall very low at 0.76 mm/day (indicating predominantly dry conditions). Fire indices show moderate means: FFMC 77.9, DMC 14.7, DC 49.4, ISI 4.7, BUI 16.7, with target FWI averaging 7.05.
- Variability: Standard deviations are notable for indices like DC (47.67) and BUI (14.20), reflecting seasonal buildup variability. Temperature has lower variability (std 3.63°C).
- Extremes: Maximum Temperature reaches 42°C, Rain up to 16.8 mm (rare events), and FWI up to 31.1 (extreme danger).
- Class Distribution: 137 fire occurrences (Classes = 1, 56%) vs. 107 non-fire days, providing a reasonably balanced binary classification target alongside the regression task.

Additional moments reveal non-normality: - Skewness: Strong positive skew in Rain (¿4, highly right-tailed due to many zero-rain days), DC (1.5), DMC, BUI, ISI, and FWI (1.2–1.8), indicating long tails toward higher fire risk values. Temperature near-symmetric (-0.3), RH slightly left-skewed.y.

These non-normal distributions justify preprocessing steps like standardization (to mean=0, variance=1) for regularization-sensitive models like Ridge, and potential log-transformations for skewed features in advanced modeling.

These statistics reveal skewed distributions (e.g., Rain mostly 0), moderate temperatures, and variable fire indices.

2.2 Histograms

Histograms of key variables show distributions: Temperature is bimodal (around 30-35°C), RH skewed right (50-80%), Ws moderate (10-20 km/h), Rain heavily skewed to 0, and fire indices like FFMC left-skewed (high values common), others right-skewed.

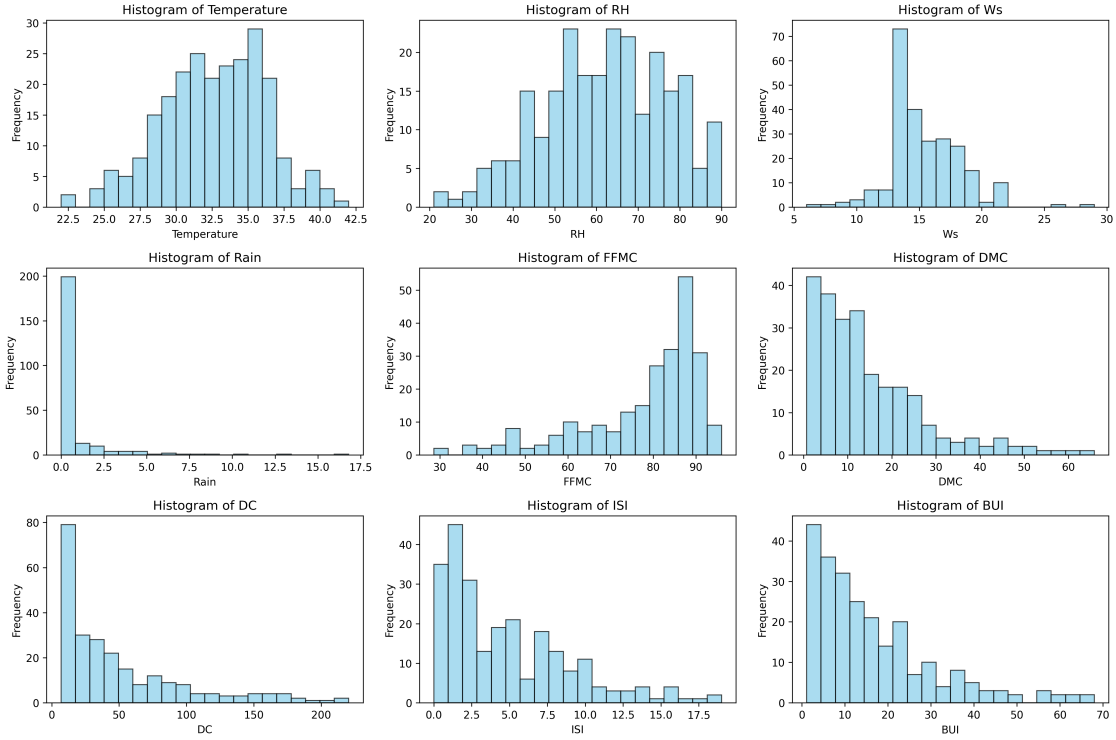

Figure 1: Histograms of meteorological variables in a forest fire dataset.

2.3 Correlation Analysis

The correlation matrix highlights strong positive correlations between FWI and ISI (0.92), BUI (0.86), DMC (0.88), DC (0.74), FFMC (0.69), and Temperature (0.57). Negative correlations with RH (-0.58) and Rain (-0.32). Multicollinearity is evident among indices (e.g., DMC-BUI: 0.98).

Key correlations with FWI:

- ISI: 0.919
- BUI: 0.857
- DMC: 0.875
- DC: 0.738
- FFMC: 0.691
- Temperature: 0.566
- RH: -0.580
- Rain: -0.325
- Ws: 0.033
- Region: 0.198

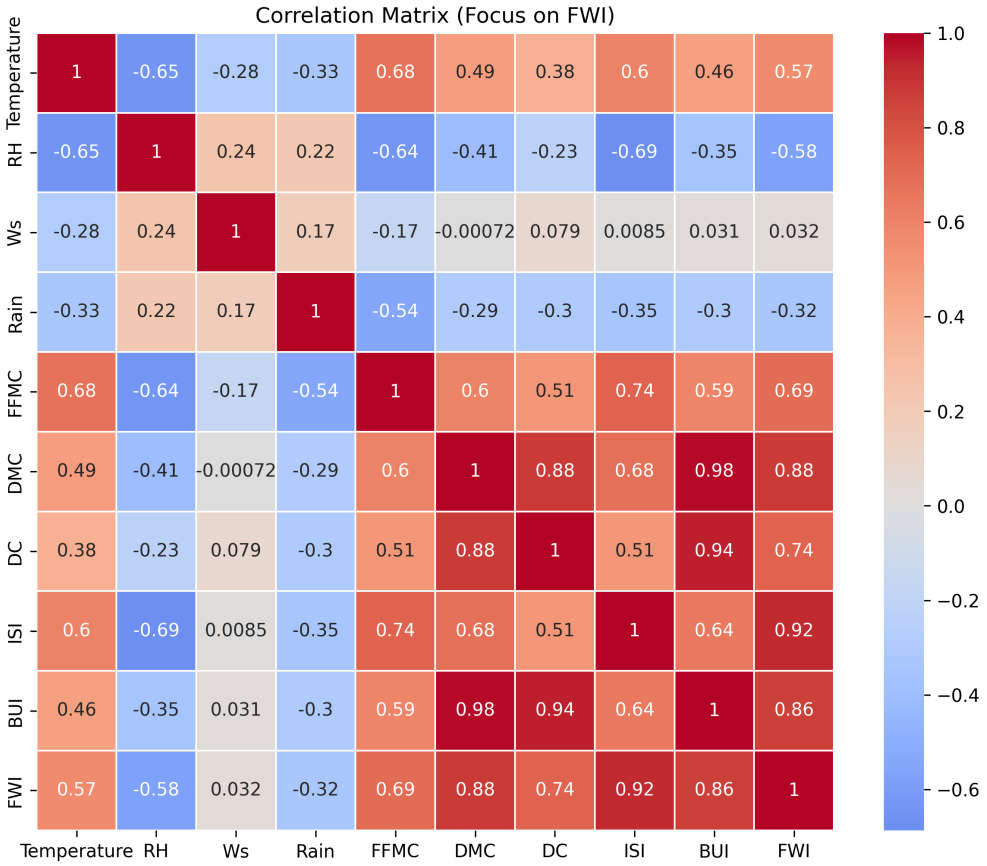

Figure 2: Correlation heatmap for FWI dataset features.

2.3.1 Pseudo-Code for Exploratory Data Analysis

Algorithm 2 EDA: Statistics, Histograms, and Correlation Heatmap

Require: 'Cleaned-FWI-dataset.csv'
Ensure: Plots saved

1: $df \leftarrow$ pd.read-csv('Cleaned-FWI-dataset.csv')
2: $print(df.describe().round(2))$
3: $corr \leftarrow df.corr()$
4: $print(corr['FWI'].sortvalues(ascending = False).round(3))$
5: Generate histograms (grid 2x5 for 10 features)
6: Save 'histograms-features.png'
7: Generate heatmap with annotations
8: $sns.heatmap(corr, annot = True, cmap =' coolwarm', fmt =' .3f')$
9: Save 'correlation-heatmap.png'

3 Ridge Regression Model

3.1 Model Description

Ridge regression was employed to predict FWI, addressing multicollinearity via L2 regularization. The model minimizes:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \alpha\sum_{j=1}^{n}\theta_j^2$$

where α is the regularization parameter.

Features: Region, Temperature, RH, Ws, Rain, FFMC, DMC, DC, ISI, BUI. Target: FWI.

Data split: 80/20 train/test. Features standardized.

3.2 Hyperparameter Tuning

Alpha values tested: logspace from 10^{-3} to 10^{3} (100 values). Best $\alpha \approx 0.132$, with MSE ≈ 0.401, MAE ≈ 0.483.

3.3 MSE and MAE vs. Alpha

MSE decreases initially with increasing alpha, reaching a minimum, then increases (over-regularization). MAE follows a similar trend but plateaus earlier.

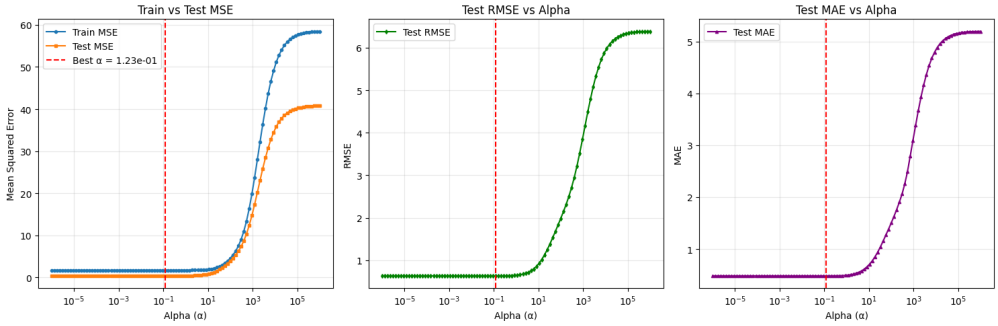

Figure 3: Performance metrics vs. regularization strength.

3.3.1 Pseudo-Code for Training and Tuning

Algorithm 3 Ridge Regression Training

Require: Cleaned *df*

Ensure: 'ridge.pkl', 'scaler.pkl', plots

1: $X \leftarrow df[features]$; $y \leftarrow df['FWI']$
2: Train-test split (20% test)
3: $scaler \leftarrow StandardScaler()$; fit on train, transform both
4: $alphas \leftarrow np.logspace(-3, 3, 100)$
5: RidgeCV for best alpha (5-fold CV)
6: Train Ridge with best alpha
7: Predict, compute MSE/MAE
8: Plot validation curve and predicted vs actual
9: Save model and scaler

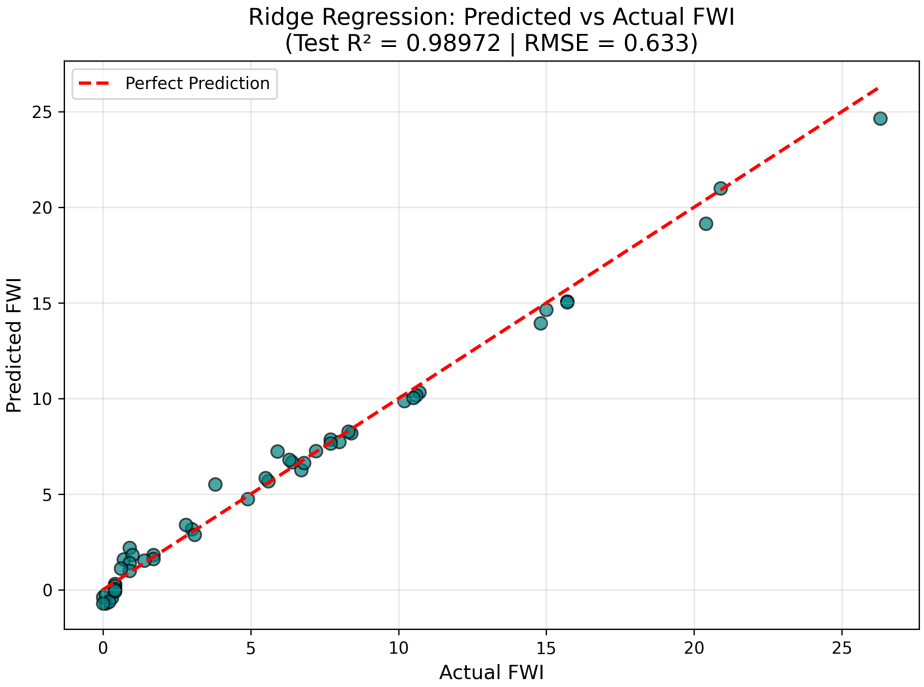

Figure 4: Actual Values vs. Predicted Plots

4 Flask Web Application

The model was deployed as a Flask web app for FWI prediction. Key components:

- **Loading Model and Scaler**: Ridge model and StandardScaler loaded from pickles.
- **Routes**: '/' for input form (index.html), '/predict' for processing inputs and rendering results (home.html).
- **Input Features**: Matches model inputs; predictions rounded to 3 decimals.
- **Risk Categorization**: FWI mapped to risk levels ($VeryLow < 2$, $Low - Moderate < 5$, $High < 12$, $VeryHigh < 25$, $Extreme \geq 25$).
- **UI**: HTML forms for inputs, card-style results with FWI value and risk.

The app runs locally (debug mode) and can be exposed via ngrok.

4.0.1 Pseudo-Code for Flask App

Algorithm 4 Flask Prediction Server

Require: Pickles and templates
Ensure: Running app
1: Load model and scaler
2: Define feature names
3: Route '/' : render input form
4: Route '/predict' (POST): extract data, scale, predict, map risk, render result
5: Risk levels: Very Low (< 2), Low–Moderate (< 5), High (< 12), Very High (25), Extreme (≥ 25)
6: Run in debug mode (ngrok for public access)