PROJECT

FLAME GUARD

TEAM

Site:AlgorithmicAlchemists

https://github.com/tomoe-495/flameguard

Contents

Background and Necessity for the Application

Forest fires are a significant environmental concern, causing devastating damage to ecosystems, wildlife, and human communities. The frequency and intensity of these fires have been increasing due to climate change, prolonged droughts, and human activities. There is an urgent requirement for advanced technological solutions that can predict and monitor forest fires to enable faster and more efficient intervention.

FlameGuard, leveraging cutting-edge predictive analytics and Machine Learning algorithms, addresses this critical necessity. By analyzing vast amounts of data from various sources such as weather patterns, vegetation indices, and historical fire occurrences, **FlameGuard** can forecast potential fire outbreaks with high accuracy. This approach not only helps in safeguarding natural habitats and biodiversity, but also minimizes economic losses and protects human lives. The necessity for such an advanced application is underscored by the growing impact of climate change and the increasing occurrence of extreme weather events, making forest fire prediction and early warning systems more crucial than ever.

Proposed Solution

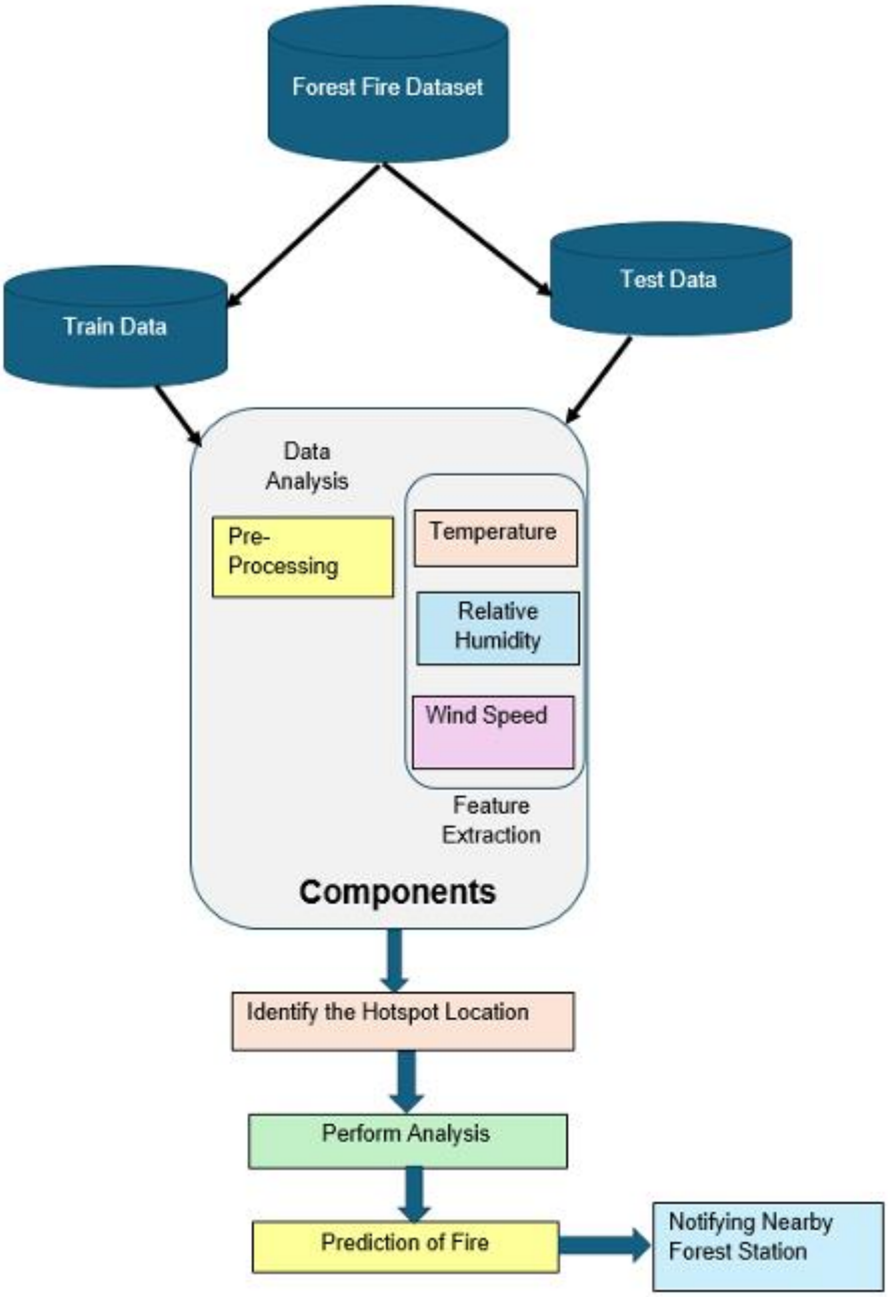
The proposed solution, 'FlameGuard', for the Forest Fire Prediction project involves creating a robust Machine Learning model to predict the likelihood of forest fires based on various environmental factors. The primary objective is to develop an accurate and efficient prediction application that can be easily accessed and used by users to mitigate the risks associated with forest fires.

Dataset and Features

The dataset comprises a data matrix where columns represent variables and rows represent instances, specifically focusing on the northeast region of Portugal. The dataset includes 515 instances divided into training (333 instances), generalization (45 instances), and testing (45 instances) subsets. Key variables in the dataset are as follows:

- **Month:** The month of the year when the data was recorded.
- **Fine Fuel Moisture Code (FFMC):** Indicates the moisture content of surface litter and fine fuels, influencing how easily fires can start and spread.
- **Duff Moisture Code (DMC):** Represents the moisture content of loosely compacted organic layers, affecting how well a fire can sustain itself.
- **Drought Code (DC):** Measures the moisture content of deep, compact organic layers, indicating the likelihood of sustained burning.
- **Initial Spread Index (ISI):** Combines wind speed and the FFMC to estimate the initial spread rate of a fire.
- **Temperature:** The temperature in degrees Celsius.
- **Relative Humidity (RH):** The relative humidity in percentage.
- **Wind Speed:** The wind speed in km/h.
- **Class:** The target variable indicating the presence (1) or absence (0) of fire.

The sample architecture of application can be as follows:



Explaining the Project

FlameGuard project involves developing a Machine Learning model to predict forest fires based on various environmental factors. The project will utilize the provided dataset containing variables such as temperature, humidity, wind speed, and moisture indices. The solution includes several key steps and serializes the trained model using Pickle to save it for future use. This allows the model to be loaded and used for predictions without retraining. Users can input temperature, humidity, and other relevant values to predict the likelihood of a forest fire, providing a user-friendly interface to assess fire risk. It can also notify the nearby Forest Station to assess fire risk and facilitate a timely response if the forest is in danger.

Steps to build the model for forest fire prediction are as follows:

1. Data Collection

The data was given to us by the project provider as a csv data file.

2. Data Pre-Processing

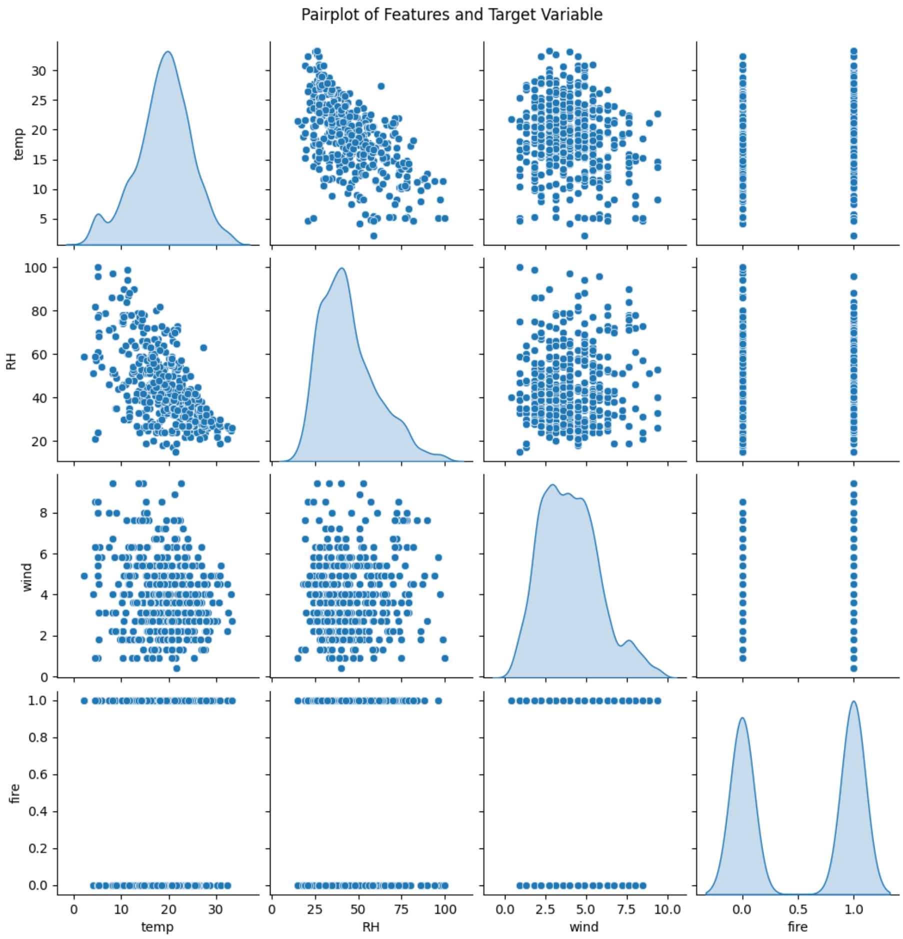
On data pre-processing, first the columns which weren't needed were discarded and were only proceeded with the necessary columns. Such as temperature, RH (relative humidity), wind and area.
The data of area was reformed as in binary just 0s and 1s.
At the end the data was divided in *features* (temp, humidity, wind) and *target* (area).
Finally the data was normalized with a scaler.

3. Exploratory Data Analysis (EDA)

	temp	RH	wind	fire
count	2068.000000	2068.000000	2068.000000	2068.000000
mean	18.889168	44.288201	4.017602	0.522244
std	5.802410	16.305624	1.790352	0.499626
min	2.200000	15.000000	0.400000	0.000000
25%	15.500000	33.000000	2.700000	0.000000
50%	19.300000	42.000000	4.000000	1.000000
75%	22.800000	53.000000	4.900000	1.000000
max	33.300000	100.000000	9.400000	1.000000

This image provides a summary of a dataset with temperature, humidity, wind speed, and fire occurrence:

- **Temperature**: Average is 18.89°C, ranging from 2.2°C to 33.3°C.
- **Humidity**: Average is 44.28%, ranging from 15% to 100%.
- **Wind Speed**: Average is 4.02 m/s, ranging from 0.4 to 9.4 m/s.
- **Fire Occurrence**: About 52% of cases involve a fire.

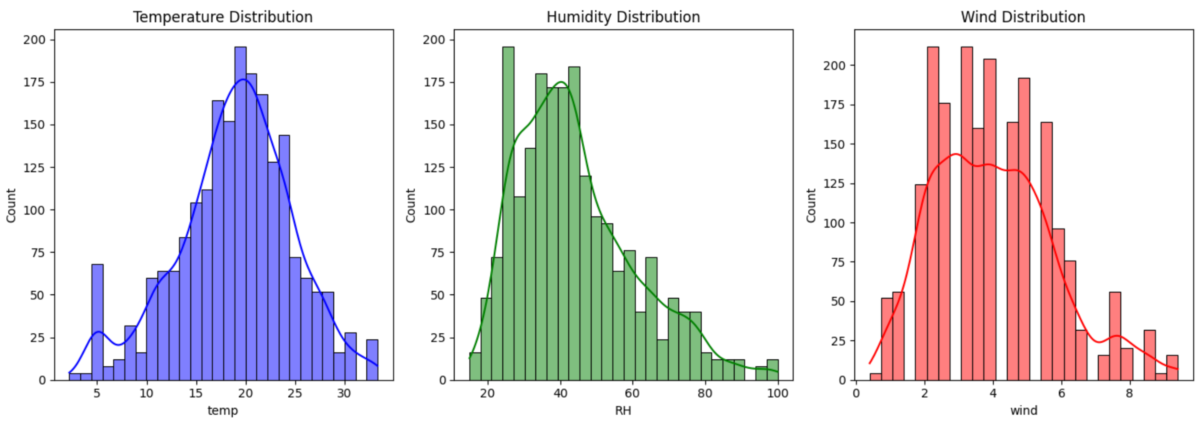


This image is a pairplot, which visualizes relationships between the features (temperature, relative humidity, wind) and the target variable (fire) in the dataset. Here's what each part shows:

- **Diagonal plots**: These are histograms that show the distribution of individual variables (temp, RH, wind, fire).
    - Temperature is normally distributed with a peak around 20°C.
    - Relative humidity has a peak near 40-50%.
    - Wind speed is skewed towards lower values, with most values under 5 m/s.
    - Fire has two distinct peaks at 0 and 1, representing binary outcomes.
- **Scatter plots**: The off-diagonal plots show pairwise relationships between the variables.

- **Temperature vs RH**: As temperature increases, relative humidity tends to decrease, indicating a negative correlation.
- **Temperature vs Wind**: There doesn't seem to be a clear relationship.
- **Temperature vs Fire**: Fire occurrence seems to increase with higher temperatures.
- **RH vs Wind**: No strong relationship is visible.
- **RH vs Fire**: Fire occurrence is higher when relative humidity is lower.
- **Wind vs Fire**: No strong correlation between wind and fire occurrence is apparent.

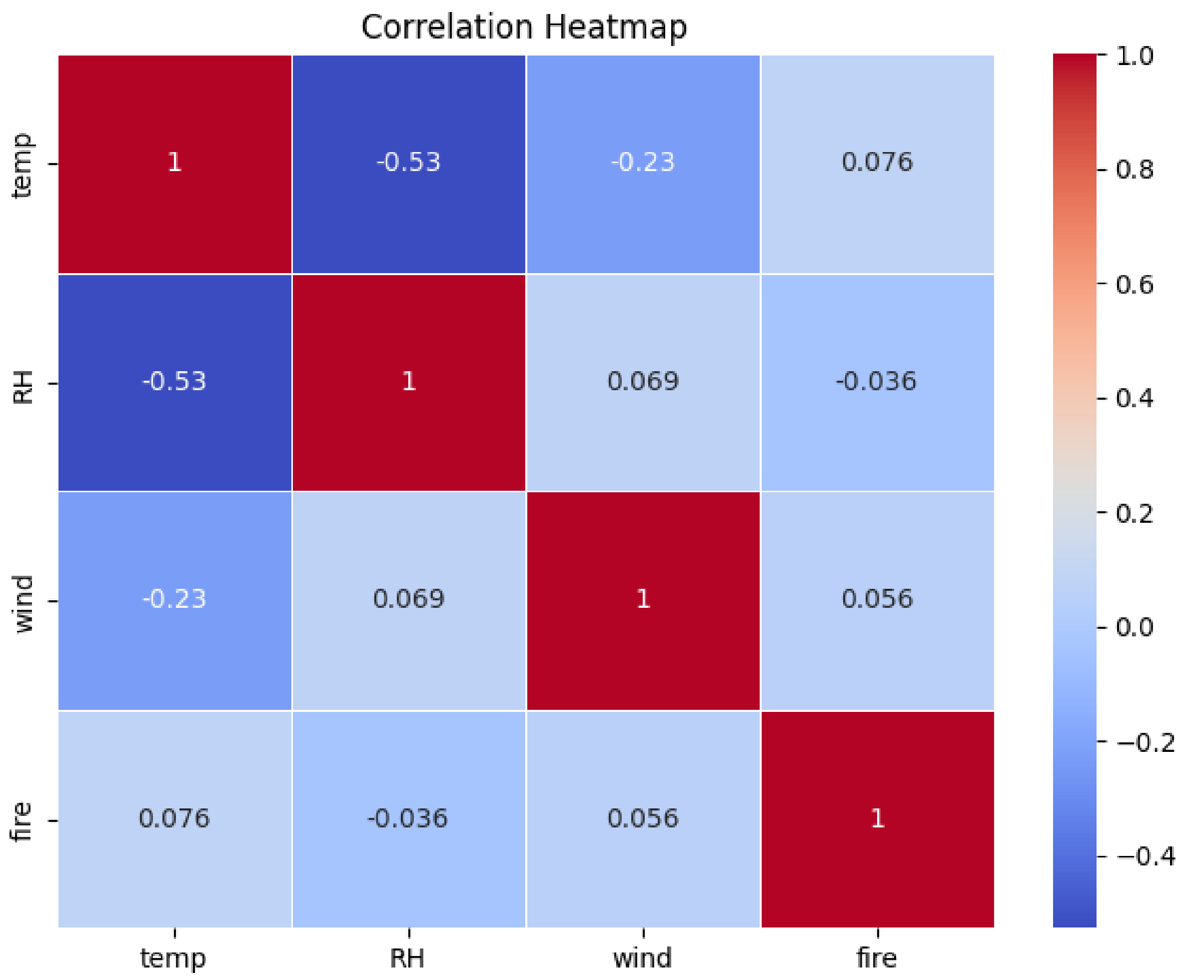
  Overall, the pairplot helps in understanding how the variables relate to each other and how they influence fire occurrence.



This image shows the distribution of temperature, humidity, and wind in the dataset:

- **Temperature**: Normally distributed, with most values between 18–22°C.
- **Humidity**: Skewed, with most values between 35–55%, and fewer extreme high or low values.
- **Wind**: Peaks around 2–5 m/s, with few values above 6 m/s.

These distributions give insight into how the variables are spread across the dataset.



4. Feature Engineering

In our fire prediction model, we applied feature engineering techniques to optimize the dataset. Missing values were imputed using mean values for temperature, humidity, and wind. We normalized the features using Min-Max Scaling and added polynomial features (e.g., temp^2, wind^2) to account for non-linear relationships. Temperature and humidity were discretized into categorical bins, and class imbalance was handled using random oversampling to ensure a balanced dataset, leading to more accurate predictions.

5. Feature Selection

we focused on identifying the most relevant environmental factors for predicting fire occurrence. The initial features—**temperature**, **humidity (RH)**, and **wind speed**—were chosen based on their direct relationship with fire risk. After exploring the dataset, we decided to retain these core features, as they demonstrated strong correlations with the target variable (fire occurrence). No additional features or derived combinations were included, ensuring the model remains interpretable and avoids overfitting on unnecessary complexity. This selection provided a balance between model performance and simplicity.

6. Model Building

In this project, several machine learning models were employed to predict the occurrence of fires based on environmental factors such as temperature, humidity, and wind speed. The models chosen include Logistic Regression, RandomForestClassifier, DecisionTreeClassifier, and Support Vector Machine (SVM). These models were selected due to their effectiveness in classification tasks and their ability to capture complex relationships between features. Logistic Regression was used as a baseline model, while RandomForestClassifier and DecisionTreeClassifier were chosen for their strength in handling non-linear data and feature importance evaluation. SVM was selected for its capability to handle high-dimensional spaces and provide robust classification boundaries. The performance of these models was compared to identify the best approach for accurate fire prediction.

7. Model Selection

we built fire prediction models using the given data processed of temperature, humidity, and wind speed. Four machine learning algorithms were implemented: Logistic Regression, Random Forest Classifier, Decision Tree Classifier, and Support Vector Machine (SVM). Each model was trained on the dataset to predict fire occurrence, and their performance was evaluated based on accuracy and other metrics. By comparing the results, we identified the most effective approach for accurately forecasting fire risks, which will aid in early fire detection and prevention efforts.

Their evaluation are as follow:

#	Logistic Regression	Random Forest Classifier	Decision Tree Classifier	SVC
Accuracy	0.53	0.95	0.93	0.60
Precision	0.53	0.96	0.94	0.58
Recall	0.74	0.95	0.93	0.78
F1 Score	0.62	0.95	0.94	0.67

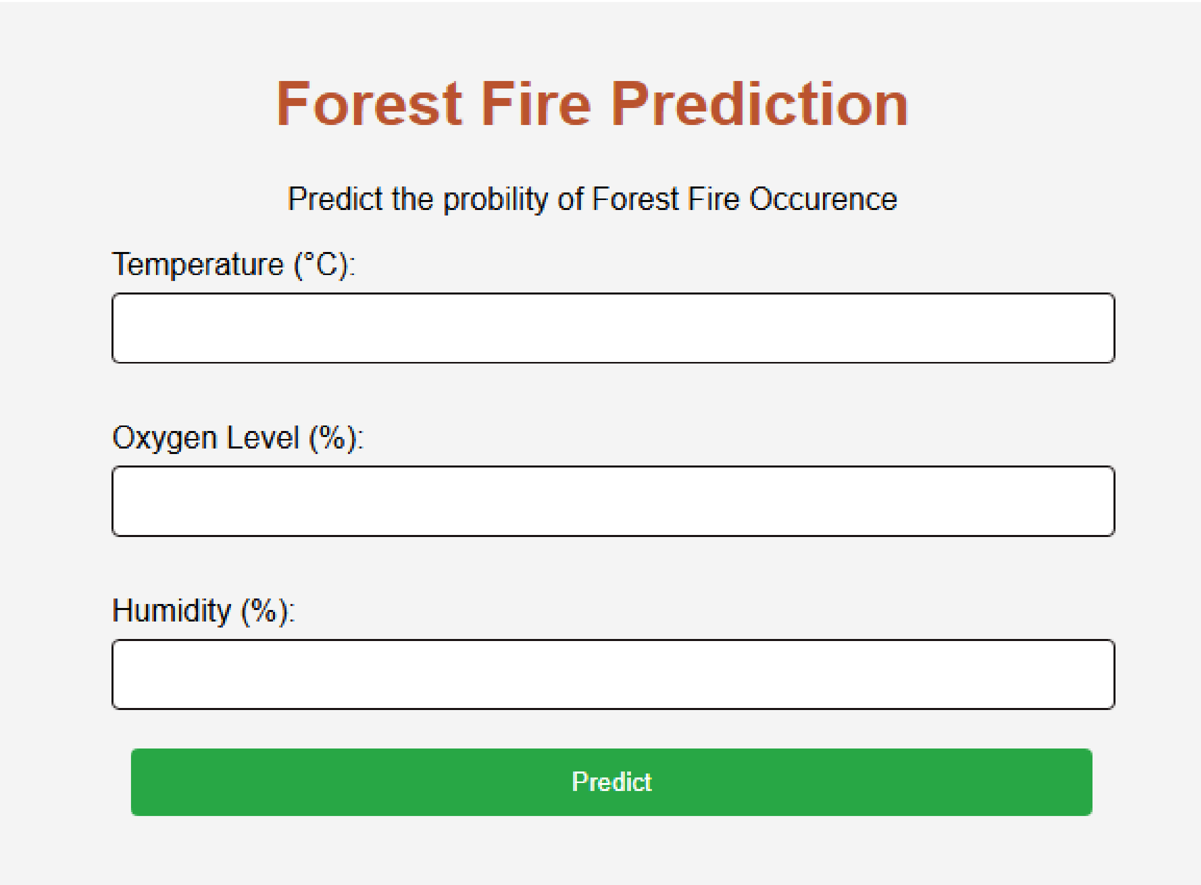
RandomForestClasifier was chosen because of it's highest accuracy

8. Model Serialization (pickle the model)

The model was serialized and then used on the Django app.

How to use the Django app

First make sure you have python installed, and then go to the directory of 'flameguard_django'. Open a terminal/cmd, type the following command 'pip install -r requirements.txt' this'll install all the necessary python packages then you can simply run project with 'python manage.py runserver' the project will now be accessible on 'localhost:8000' or '127.0.0.1:8000'



This concludes the Documentation