# Wildfire Prediction Challenge Documentation

## Overview and Objectives

### Overview:

This project focuses on developing a machine learning model to predict the burned area in different locations based on a set of climatic, geographic, and environmental features. The solution is designed to help in better understanding and managing the risk of wildfires, which are increasingly becoming a concern due to climate change. The primary challenge addressed by this project is accurately predicting the extent of wildfire damage, which is crucial for resource allocation, disaster preparedness, and mitigation strategies.

### Objectives:

1. Accurate Prediction: Develop a robust machine learning model capable of predicting the burned area with high accuracy.
2. Feature Selection: Identify and select the most important features that significantly contribute to the prediction, optimizing model performance.
3. Model Interpretability: Ensure that the model is interpretable, allowing stakeholders to understand which factors most influence wildfire spread and intensity.
4. Scalability: Design the solution to be scalable and adaptable to different geographic regions with varying environmental conditions.
5. Resource Optimization: Provide insights that can help in optimizing resource allocation for wildfire management and emergency response.
6. Collaborative Effort: Leverage the diverse expertise within the team to explore various approaches, test multiple models, and continuously improve the solution.

## Data Modeling

### 1. Description of the Data Model(s) Used:

In this project, two different approaches were utilized for data modeling:

- The first approach involved generating new features from the old features and training a LightGBM model to output the predictions.
- The second approach implemented Winsorization for outlier handling, followed by feature engineering and robust scaling, and ultimately training a LightGBM model. Both approaches aimed to predict the burned area (burn_area) using a dataset containing various climatic, landcover, and geographical features.

**Assumptions/Theoretical Foundations:**

- **Linear Regression:** Assumes a linear relationship between the independent variables and the target variable. The model minimizes the sum of squared differences between the observed and predicted values.
- **LightGBM:** A gradient boosting framework based on decision trees that builds models sequentially, optimizing them by minimizing a loss function. LightGBM assumes that the data is not strictly linear and can capture complex, non-linear relationships.

## 2. Feature Selection, Engineering, and Normalization Processes:

- **Feature Selection:** For both approaches, specific features that were either redundant or had multicollinearity were dropped. The features `climate_tmmn`, `climate_tmmx`, `climate_vap`, `climate_aet`, and others were removed after careful evaluation.

- **Feature Engineering:**
  - **First Approach:** Focused on adding new features from pre existing features
  - **Second Approach (Winsorization):** In addition to scaling and outlier handling, several new features were engineered to capture interactions between the existing features, such as:
    - `combined_temp_vap`: Average of temperature and vapor pressure.
    - `combined_vegetation`: Sum of landcover features related to vegetation.
    - `elevation_sun_interaction`: Product of elevation and solar radiation.
    - `drying_effect`: Interaction between vapor pressure deficit, solar radiation, and soil moisture.
- **Normalization:** In second approach, RobustScaler was used to normalize the features, making the model less sensitive to outliers and ensuring that all features contributed equally to the model.

## 3. Model Training:

- **Algorithms Used:**
  - **First Approach:** LightGBM (Light Gradient Boosting Machine).
  - **Second Approach:** LightGBM (Light Gradient Boosting Machine).
- **Hyperparameters:**
  - **LightGBM:** Default settings were also used here, focusing on the core strength of LightGBM's ability to handle large-scale data with high efficiency.
- **Training Process:**
  - For the first model training approach, the data was splited using train_test_split ()function to t() he ratio of 80/20 then it was validated

    - In the second approach, after Winsorization and feature engineering, LightGBM was trained on the processed data.
- **Evaluation Metrics:**
    - **Mean Squared Error (MSE):** Measures the average squared difference between actual and predicted values.
    - **Root Mean Squared Error (RMSE):** Provides a measure of the error in the same units as the target variable.
    - **R-squared:** indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

### 4. Model Validation and Performance Measurement:

- **Validation:**
    - Both models were validated using a separate validation set, not seen during training, to assess their generalization performance.
    - The evaluation metrics were computed on the validation set to ensure the models were not overfitting to the training data.

- **Performance Measurement:**
    - The first approach showed better performance than the second approach eventhough both approaches used LightGBM model.
    - The final predictions were made by averaging the predictions from both models, providing a more robust output by leveraging the strengths of each model.

This approach ensured that the final model was well-tuned, capturing both linear and non-linear relationships in the data, while also handling outliers and interactions between features effectively.

## Performance Metrics

### 1. Metrics and KPIs :

- **Data Completeness:** Ensured that the dataset was complete, with no missing values in critical columns.

- **Data Consistency:** Verified that data types were consistent across the dataset, and no anomalies existed after transformations, such as date formatting and geographic coordinate validation.

- **Outlier Handling Efficiency:** Used metrics such as the percentage of data points removed or adjusted during outlier handling to ensure that significant information was not lost.

## 2. Metrics for Model Accuracy:

- **Mean Squared Error (MSE):** Calculated for both the training and validation datasets to measure the average squared difference between the predicted and actual values. This helped in determining how well the models were performing in terms of prediction accuracy.
- **Root Mean Squared Error (RMSE):** Used as a standard measure to compare the errors of different models since it is in the same units as the target variable (`burn_area`). The RMSE provided a clear understanding of how far off the predictions were on average.
- **R-squared ($R^2$):** Measured the proportion of variance in the dependent variable (`burn_area`) that was predictable from the independent variables. Higher $R^2$ values indicated better model performance in explaining the variance.

## 3. Inference Outcomes:

- **Public and Private Scores:**
  - **Public Score:** 0.019308177—This score was obtained by submitting the predictions to the competition's public leaderboard,which used a subset of the test data.
  - **Private Score: 0.01821023**—This score was revealed after the competition ended and was based on the remaining unseentest data. It provided a final assessment of the model's performance.

## 4. Additional Metrics:

- **Cross-Validation Scores:** During model development, cross-validation was used to ensure that the model's performance was consistent across different subsets of the data. This helped in preventing overfitting and provided a more reliable estimate of model performance.
- **Model Training Time:** The time taken to train the models was monitored, particularly for LightGBM, to ensure that the solution was not only accurate but also computationally efficient.

## 5. Commentary on Metrics:

The combination of metrics provided a comprehensive evaluation of the models, balancing accuracy (RMSE, MSE, $R^2$) with practical considerations like computational efficiency and generalization performance (cross-validation scores). The use of public and private leaderboard scores gave insight into how well the models would perform in real-world, unseen scenarios.

## Additional Resources

For more details on the code, data, and analysis, you can access the full project on the GitHub repository.