CSCU9M5 ASSIGNMENT - 3115875

PROJECT METHODOLOGY

1. Loaded the dataset (517 rows, 13 variables).
2. Converted target (area) to binary (T >4%, F ≤4%).
3. Explored distributions, correlations and patterns.
4. Split data: 70% training, 30% testing.
5. Removed rain, wind, X and Y due to low variance/weak correlation and cleaned ISI.
6. .Preprocessed the data using a ColumnTransformer:
   a. Standardised numeric variables
   b. One-hot encoded the categorical variable month
7. Built ML pipelines combining preprocessing + classifier.
8. Evaluated models using 5-fold cross-validation (F1).
9. Tuned top models (Random Forest, Gradient Boosting, Decision Tree) using GridSearchCV.
10. Selected Gradient Boosting based on best F1.
11. Evaluated final model and interpreted results.

VARIABLES/ DATA UNDERSTANDING

The dataset contains 517 fire records from Montesinho National Park. It includes spatial coordinates, weather measurements, and Fire Weather Index (FWI) components. Most variables are numeric but on different scales, which is why scaling is later applied.

Variable	Type
FFMC, DMC, DC, ISI	Numeric
temp, RH	Numeric
month	Categorical
area (target)	Categorical → binary

Month is a categorical variable and was one-hot encoded before modelling. The target variable (area) was originally labelled T/F and was converted to a binary numeric class for classification.

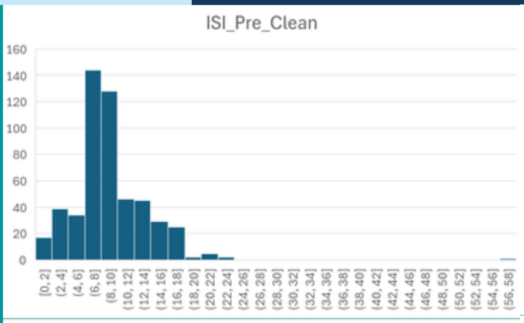
DATA PREPARATION

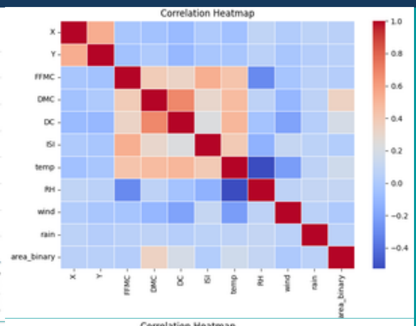
Before Cleaning

- Target variable stored as letters (T/F).
- Month was a string and not usable in modelling.
- Numeric features were on different scales.

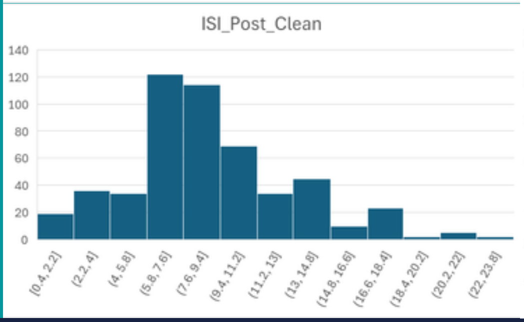
After Cleaning

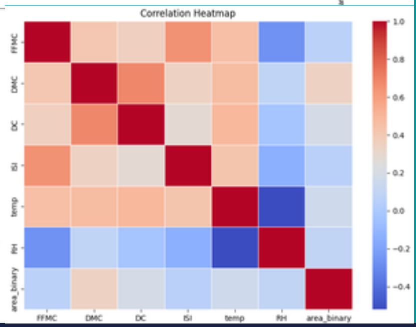
- Converted area → binary (0 = F, 1 = T).
- One-hot encoded month.
- Standardised numeric variables.
- Removed rain, wind, X, Y, and day due to very low variance or no meaningful correlation with the target.
- Removed extreme ISI outliers (top and bottom).
- Final cleaned dataset: **515 rows, 7 features.**





Pre-Cleaning Correlation Heatmap





Post-Cleaning Correlation Heatmap

Model	Mean F1	Std F1
Random Forest	0.894	0.039
Gradient Boosting	0.885	0.044
Decision Tree	0.867	0.03
KNN	0.806	0.042
SVM	0.79	0.031
Logistic Regression	0.726	0.069

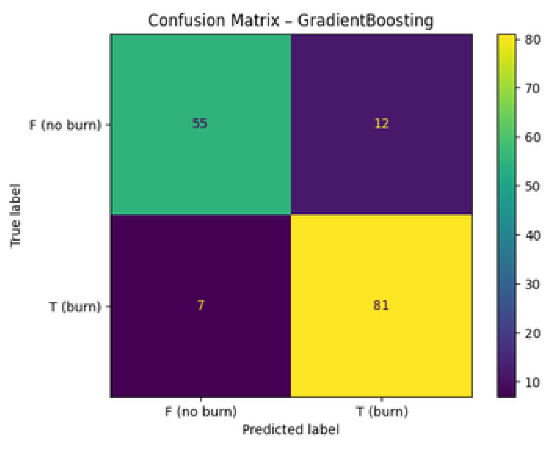
MODEL TRAINING AND HYPER PARAMETERS

I first trained all models with default settings to compare baseline F1-scores. The three best performers Random Forest, Gradient Boosting, and Decision Tree were selected for hyperparameter tuning. For each model, I tuned the hyperparameters most likely to influence performance:

- Random Forest: number of trees, max depth, min samples split
- Gradient Boosting: n_estimators, learning rate, tree depth
- Decision Tree: max depth, min samples split, min samples leaf

GridSearchCV with 5-fold cross-validation was used to test combinations of these parameters and identify the best settings. Gradient Boosting achieved the strongest CV score (F1 = **0.903**, with n_estimators=100, learning_rate=0.05, max_depth=5). Random Forest also performed strongly (F1 = **0.899**), followed by Decision Tree (F1 = **0.867**).

FINAL MODEL AND RESULTS



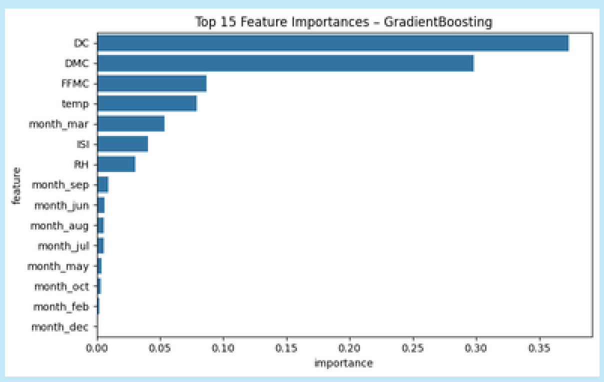
The model predicts burn and non-burn cases with high accuracy:

- Correctly predicted 82% of no-burn cases (55/67)
- Correctly predicted 92% of burn cases (81/88)
- False negatives: 8% (7/88)
- False positives: 18% (12/67)

This shows Gradient Boosting is especially strong at detecting actual fires (high recall) and provides robust overall performance (F1 = 0.903, ROC-AUC = 0.94).

INSIGHT ABOUT DATA

Feature importance analysis showed that the FWI indices (DC, DMC, FFMC, ISI) are the strongest predictors of fire occurrence, reflecting fuel dryness and heat buildup, key factors in ignition. Temperature and humidity also contribute, but to a lesser extent.



Month adds seasonal context but is less important than the core FWI variables. Gradient Boosting's feature importances highlight the environmental conditions most linked to fire events. No model bias was detected, with balanced recall and precision. The small dataset (515 rows) may limit generalisation, so more data or resampling could improve robustness of the model.

References

NumPy – Harris, C. R. et al. (2020). Array programming with NumPy. https://numpy.org/
pandas – McKinney, W. (2010). Data structures for statistical computing in Python. https://pandas.pydata.org/
scikit-learn – Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. https://scikit-learn.org/
seaborn – Waskom, M. L. (2021). Seaborn: Statistical data visualization. https://seaborn.pydata.org/