1. Student ID: 3326080

2. Project Methodology

1. Loaded the raw forest fore dataset and inspected the size and variables
2. Explored the data using summary statistics and histograms to understand distributions and spot outliers
3. Cleaned the data by removing an outlier in the 'rain' variable (values > 2.0) and converting 'area' from F/T into 0/1.
4. Split the data into 70% training, 15% validation, and 15% test, using a stratified split to preserve the class balance
5. Applied preprocessing: one-hot encoding for month and day, and standard scaling for numeric features (fitted to training set only)
6. Trained 3 models (Logistic Regression, random Forest, Neural Network) using the training set and tuned hyperparameters based on validation accuracy
7. Selected random Forest as the final model, retrained it on the combined training + validation data, and evaluated it on the test set using accuracy and a confusion matrix.

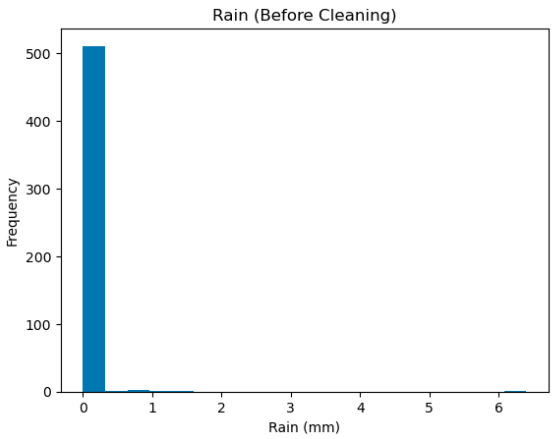
3. Variables/ Data Understanding

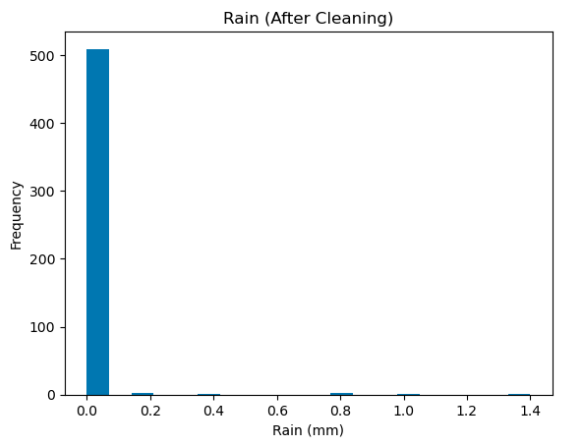
Month and day are categorical, while the remaining features are numerical. 'Area' was converted from F/T to 0/1. A consequence is that categorical variables need encoding, while numerical variables can be used directly in modelling.

Variable	Type	Variable	Type
X	Numerical (Discrete)	ISI	Numerical (Continuous)
Y	Numerical (Discrete)	temp	Numerical (Continuous)
month	Categorical	RH	Numerical (Discrete)
day	Categorical	wind	Numerical (Continuous)
FFMC	Numerical (Continuous)	rain	Numerical (Continuous)
DMC	Numerical (Continuous)	area	Categorical -> Numerical (Discrete)
DC	Numerical (Continuous)		

4. Data Preparation

Used a histogram to detect an outlier in the 'rain' variable. A value of 6.4mm was far outside the typical range (0-1.4mm), so I removed observations with rain > 2.0mm. The histograms below show the distribution before and after cleaning.





5. Model Training and Hyper Parameters

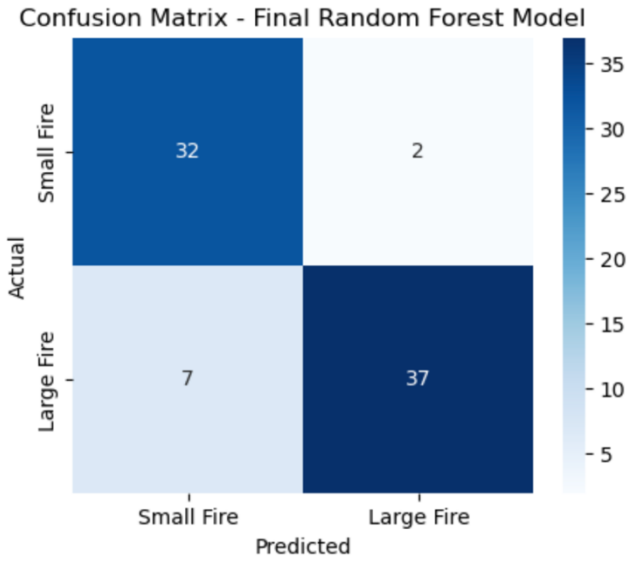
I chose simple and commonly used hyperparameter values for each model, like different C values, tree sizes, and hidden layer sizes. This gave me a good range to compare without making the search too complicated.

I trained each model on the training set and checked its accuracy on the validation set, changing one hyperparameter at a time. This made it easy to compare results and see which settings worked best.

Model	Hyper Parameters Tested	Best Hyper Parameter	Validation Metric
Logistic Regression	C = {0.1, 1, 10}	C = 10	71.4%
Random Forest	n_estimators = {50, 100, 200} max_depth = {5, 10, 20, None} min_samples_split = {2, 5, 10}	n_estimators = 100 max_depth = 10 min_samples_split = 2	92.2%
Neural Network	hidden_layer_sizes = {(20,), (50,), (100,)}	hidden_layer_sizes = (50,)	77.9%

6. Final Model and Results

Random Forest gave the highest validation accuracy, so I selected it as the final model. I retrained it on the full training + validation data and then tested it on the unseen test set.



The model performs well at identifying both small and large fires. It makes very few false alarms (2 cases predicted a large fire when there wasn't) and a moderate number of missed large fires (7 cases). Overall, the model could be used for supporting fire risk decisions, but missed large fires pose a risk and should be considered.

7. Insight about data or models gained

- Preprocessing made a clear difference: Encoding and scaling improved model accuracy, especially for Logistic Regression and the Neural Network.
- Random Forest handled the dataset best, suggesting tree-based models are more suitable for mixed and non-linear forest fire data.
- The confusion matrix showed useful performance, but some large fires were still missed, which would be important to address in real-world fire management.

8. References

- Skicit-learn: Machine Learning Library for Python
- Pandas: Data Structures for Statistical Computing in Python
- Matplotlib: Plotting 2D Graphics in Python