

Climate Multi-Hazard Risk Classification System

J.Venkata Raghava¹ K.Rahul² D.Nandini³ R.Pradeep⁴

Guide: Gireesh Kumar Sir⁵

¹²³⁴UG Students, ⁵Professor

CSE (AI & ML), Sreyas Institute of Engineering and Technology, Hyderabad, INDIA

Abstract

At present, climate change is probably the most dangerous threat to our planet Earth. Each year, global warming, deforestation, irregular rainfall, and greenhouse gas increases have contributed to the rise of climatic hazards such as droughts and wildfires. These have become much more frequent in recent years and pose a danger to both natural ecosystems and people. Determining regions especially vulnerable to these phenomena is crucial, as it helps governments provide adequate disaster response and long-term climate action — contributing to United Nations Sustainable Development Goal 13.

In this work, we developed a Climate Multi-Hazard Risk Classification System that uses machine learning to predict risk based on climate-related parameters including temperature variability, rainfall pattern, carbon emissions, and deforestation rate. Three classification models were evaluated — Logistic Regression, Decision Tree, and Random Forest — with Random Forest achieving the best performance at 94.5% test accuracy and 93.8% cross-validation accuracy. The result is an automated and interpretable classification system accessible through a Flask-based web interface.

Keywords: Climate Risk, Multi-Hazard Classification, Random Forest, Machine Learning, Drought, Forest Fire, Sustainable Development

1. Introduction

Climate change is the most pressing problem faced by humankind today. The planet's average temperature has risen sharply, mainly due to increased greenhouse gas concentrations including CO₂ and methane. These issues manifest in real-world environmental changes and extreme weather events. Among the hazards that have intensified most significantly over the last century, both in occurrence and severity, are droughts and forest fires.

These two hazards are deeply interconnected. A drought reduces soil moisture and dries out the plant layer, making areas more vulnerable to fires. Once a fire starts, its spread accelerates and consequences compound: destruction of plant life, loss of tree coverage, and reduced soil capacity to retain moisture. Studying these hazards separately no longer provides a complete picture of the risks.

Traditional risk assessment methods focus on one hazard at a time, limiting their ability to address combined effects of multiple hazards. This gap has pushed researchers and policymakers toward a multi-hazard approach. By examining droughts and forest fires together, more accurate risk assessments become possible — informing detailed risk maps, improving land-use planning, optimizing resource allocation, and developing better early warning systems.

Despite increased emphasis on this problem, multi-hazard prediction systems — particularly those employing machine learning — remain rare. This is especially true for developing nations that lack access to reliable climate data despite their greater vulnerability to climate change.

To address this gap, we developed a predictive model system that considers drought and forest fire hazards together using machine learning approaches. A key challenge was the lack of large, labeled climate datasets, which we overcame by developing a synthetic dataset generation method grounded in physical properties of the atmosphere.

Three machine learning algorithms were evaluated: Logistic Regression, Decision Tree, and Random Forest. Random Forest performed best due to its ability to handle complex relationships between parameters such as temperature, precipitation, carbon emissions, and forest coverage, while avoiding overfitting through ensemble learning.

A user-friendly web application was built using Flask, allowing non-technical users to obtain risk predictions without any coding knowledge. The project is also aligned with UN SDG 13 on Climate Action.

The system was designed to be modular and comprehensible — not a black-box model, but one whose behavior is transparent and whose predictions can be trusted. Modularity also enables future scalability as new hazards or real-world datasets are incorporated.

Scope of the Project

This project delivers a complete, end-to-end machine learning pipeline for multi-hazard climate risk classification. The current version focuses on two hazards — droughts and forest fires — since these are closely linked and highly relevant globally. The system takes in environmental features and outputs one of four risk levels: Low, Medium, High, or Very High, using supervised learning models trained on a physics-informed synthetic dataset.

The system is currently intended for research and educational use, but has been designed with future expansion in mind — including integration with geospatial data, additional hazard types, and cloud deployment with API access for policymakers and third-party developers.

Objectives of the System

The main objectives of the proposed system are:

- To develop a physics-informed synthetic dataset that accurately reflects real-world environmental and climatic conditions.
- To engineer meaningful features from raw climatic, topographic, and vegetation data for use in machine learning models.
- To train and evaluate multiple supervised classification algorithms including Logistic Regression, Decision Tree, and Random Forest.
- To identify and deploy the best-performing model as the core engine of the climate risk classification system.
- To build a Flask-based web application that provides real-time, interactive multi-hazard risk predictions to users.
- To support data-driven climate adaptation and disaster management planning in alignment with UN SDG 13.

2. Literature Survey

A review of recent literature (2020–2023) was conducted to understand the current state of climate hazard prediction and multi-hazard risk assessment. Ten relevant studies were identified and summarized below, with attention to their methods, contributions, and limitations.

Author & Year	Contribution	Methods/Models	Limitations
Heo et al. (2023)	Multi-hazard exposure mapping for Kalimantan Islands	Random Forest	Region-specific; limited generalizability
Pourghasemi et al. (2020)	Multi-hazard risk susceptibility mapping	Machine Learning ensemble	Requires large labeled datasets
Nachappa et al. (2020)	Multi-hazard exposure mapping for Salzburg, Austria	Machine Learning	Limited to single geographic region
Khan et al. (2020)	Prediction of droughts over Pakistan	ML algorithms (SVM, RF)	Single hazard; region-specific
Piao et al. (2022)	Forest fire susceptibility assessment using GEE	Google Earth Engine + ML	Single hazard; no drought integration
IPCC AR6 (2021)	Physical science basis of climate change	Statistical/scientific modeling	Not a prediction model; descriptive only
Field (Ed.) (2018)	Managing risks of extreme events under climate change	IPCC special report methodology	Policy-focused; no ML implementation
Breiman (2001)	Introduced Random Forest algorithm	Random Forest (ensemble)	General method; not climate-specific
Pedregosa et al. (2011)	Scikit-learn ML library for Python	Python-based ML toolkit	Tool/library; not a climate study
Sommerville (2017)	Software engineering principles for reliable systems	SE methodology	Not climate-focused; general SE reference

Table 2.1 — Literature Survey Summary (2020–2023)

3. Proposed Methodology

The Climate Multi-Hazard Risk Classification System is a machine learning application designed to assess regional vulnerability to climate-related hazards in an automated way. As climate conditions grow more complex — with rising temperatures, declining rainfall, changing forest cover, and growing CO₂ levels — the risk of both droughts and fires increases in non-linear ways that simple statistical models cannot fully capture.

Our approach bridges this gap by combining physics-informed data generation with an ensemble ML model, resulting in a robust and transparent classification framework. The system was also designed to be accessible to non-technical users through a web interface.

3.1 System Pipeline

The system operates through a logical step-by-step pipeline where each stage feeds into the next:

- Physics-based synthetic data generation using atmospheric and ecological rules.
- Data preprocessing: missing value imputation, categorical encoding, and feature scaling.
- Feature engineering embedded in the data generation stage; intermediate scores removed before training to prevent data leakage.
- Training of three classifiers: Logistic Regression, Decision Tree, and Random Forest.
- Evaluation using accuracy, five-fold cross-validation, confusion matrices, and classification reports.
- Deployment via a Flask web application where users enter parameter values and receive real-time risk classifications.

3.2 System Architecture

The system uses a layered architecture with four distinct functional layers:

- Data Layer — Physics-based synthetic data generation
- Core System — Preprocessing and feature engineering
- Model Layer — Training, evaluation, and model selection
- Output / Deployment — Flask web application and real-time prediction serving

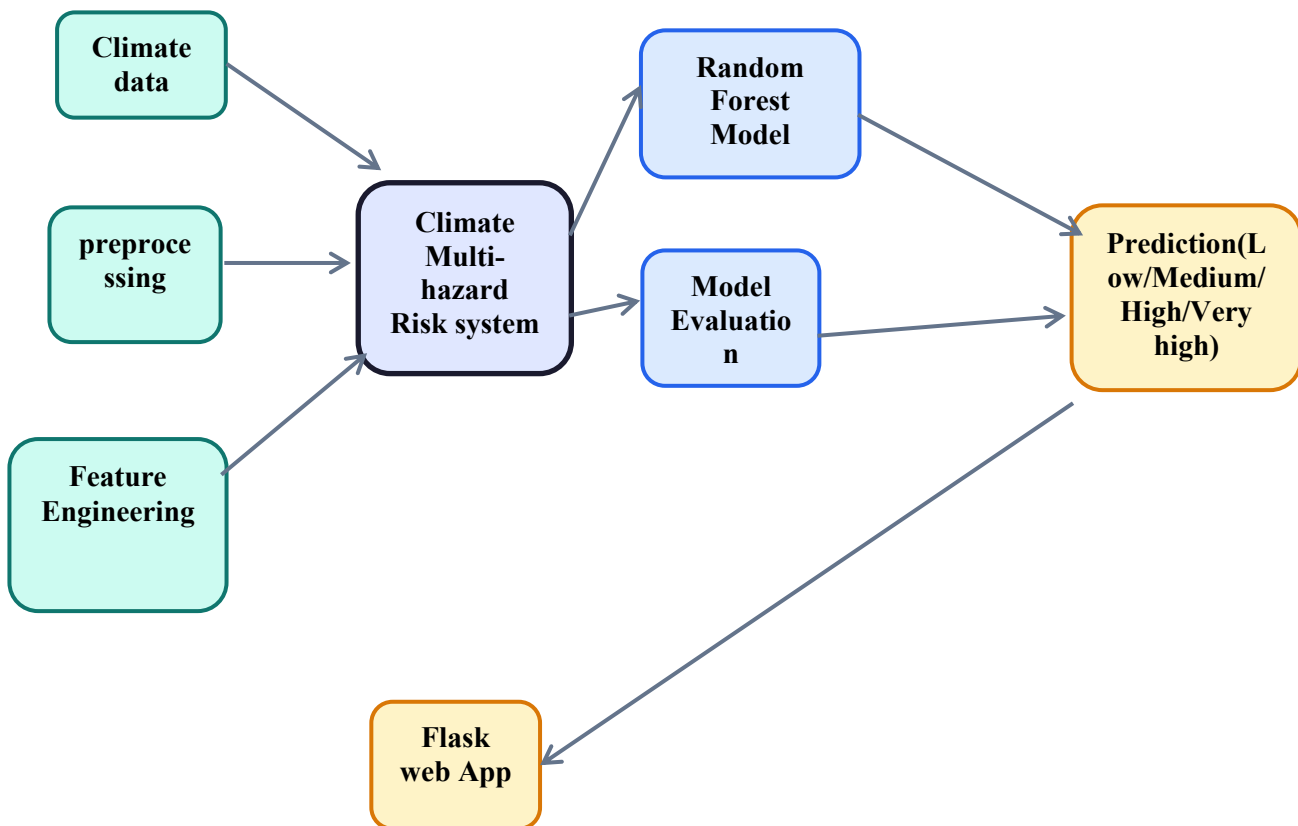


Figure 3.1.1 System Architecture diagram

```
Training Decision Tree...
Accuracy: 0.7730
CV Accuracy: 0.7448 (+/- 0.0386)

Training Random Forest...
Accuracy: 0.8370
CV Accuracy: 0.8168 (+/- 0.0318)

--- Step 4: Final Evaluation with Best Model ---
Best Model Selected: logisticregression

Classification Report:
      precision    recall  f1-score   support

    LOW       0.95       0.96       0.95        250
    MEDIUM    0.90       0.92       0.91        250
    HIGH       0.92       0.91       0.91        250
    VERY HIGH  0.98       0.95       0.97        250

 accuracy          0.94          0.94          0.94       1000
  micro avg        0.94          0.94          0.94       1000
  weighted avg     0.94          0.94          0.94       1000

--- Step 5: Saving Assets for Deployment ---
Model and scalers saved successfully in models/ directory.
PS C:\Users\Yayay\OneDrive\python_projects\climate-risk-system
```

Figure 3.1.2— System Architecture

3.2.1 Data Generation Layer

Since large labeled real-world climate datasets are difficult to obtain, a custom data generator was written that simulates realistic environmental conditions using physics-based rules. It generates values for temperature, rainfall, CO₂ levels, forest cover, elevation, slope, population density, and region type — then uses atmospheric and ecological formulas to compute intermediate drought and fire risk scores, and assigns overall risk labels based on defined thresholds.

3.2.2 Data Preprocessing Module

Raw data was cleaned and standardized using Scikit-Learn's SimpleImputer to fill missing temperature and rainfall values (using median imputation, with ~5% missing values introduced to simulate sensor noise). Categorical columns such as Region Type and the target risk label were converted to integers using LabelEncoder. All numerical features were scaled with StandardScaler to ensure no single variable dominates due to scale differences.

3.2.3 Feature Engineering Module

Feature engineering was embedded in the data generation stage itself — physical drought and fire scores were computed, then deliberately removed before model training. This forces the model to learn the relationships from raw environmental inputs rather than from pre-computed scores, ensuring better generalization and preventing data leakage.

3.2.4 Model Training Module

Three classifiers were trained on the preprocessed dataset. Logistic Regression served as a linear baseline. Decision Tree captured non-linear patterns but showed signs of overfitting. Random Forest — an ensemble of many decision trees trained on random data subsets — provided the most stable and accurate results, with hyperparameter tuning applied to further improve performance.

3.2.5 Evaluation Module

Model comparison used five-fold cross-validation for stability assessment, confusion matrices for class-level error analysis, and full classification reports (precision, recall, F1-score) for each of the four risk categories. Feature importances were also computed for the Random Forest model to identify which environmental variables most influence predictions.

3.2.6 Deployment Layer

Flask was used to build a web application that loads the saved Random Forest model, scaler, and label encoder at startup. The frontend uses a Glassmorphism-styled dashboard where users enter environmental parameter values and receive an immediate risk classification. Input preprocessing uses the same pipeline as training, ensuring consistency between training and inference.

3.3 Key Attributes of the Proposed System

- Handles both droughts and forest fires together in a single unified framework.
- Physics-informed synthetic data ensures scientifically grounded model training.
- Random Forest provides strong accuracy and outputs feature importances, avoiding a complete black-box.
- Web interface enables use by non-developers without any installation.
- Modular architecture allows future addition of hazard types or real geospatial datasets.

3.4 Advantages of the Proposed System

- Combines drought and forest fire risk modeling into a unified multi-hazard classification framework.
- Employs an ensemble Random Forest model that outperforms single classifiers in accuracy and generalization.
- Uses physics-based synthetic data generation, eliminating dependence on large real-world datasets.
- Provides real-time climate risk predictions through an interactive web portal.
- Supports climate adaptation and disaster management planning aligned with UN SDG 13.

3.5 Limitations of the Proposed System

- Trained on synthetic data; performance on actual geospatial datasets may require re-training and validation.
- Currently limited to two hazard types (droughts and forest fires); floods, landslides, and others are not included.
- The synthetic data generator uses simplified physical models; real-world climate dynamics are considerably more complex.
- Not suitable for large-scale real-time geospatial deployment without integration with satellite or remote sensing data pipelines.

3.6 Functional Requirements

The functional requirements define what the system must be capable of performing:

FR1. Synthetic Data Generation

Generate a synthetic climate dataset with at least 1000 samples, each containing features such as temperature, rainfall, CO₂ levels, forest cover, population density, elevation, slope, and region type — all produced using physics-based rules that reflect realistic climate conditions.

FR2. Data Preprocessing

Handle missing values in temperature and rainfall columns using median imputation. Convert categorical variables to numeric form and standardize all feature values to unit variance before feeding them into classifiers.

FR3. Multi-Model Training

Train all three classifiers — Logistic Regression, Decision Tree, and Random Forest — on the preprocessed dataset using a 70-30 train-test split, and evaluate each independently for comparison.

FR4. Model Evaluation

Compute accuracy scores, five-fold cross-validation, confusion matrices, and full classification reports for each model. For Random Forest, also calculate and visualize feature importance scores.

FR5. Risk Classification

Classify each input sample into one of four risk levels: Low, Medium, High, or Very High — combining drought and forest fire risk into a single unified output.

FR6. Web Application Interface

Provide a browser-accessible web portal built with Flask where users can enter environmental parameters through a styled dashboard and receive an immediate risk classification. The interface must be responsive and work across modern web browsers.

FR7. Model Persistence

Save the trained Random Forest model, StandardScaler, and LabelEncoder as .pkl files for reuse without retraining. The Flask app loads these at startup to ensure consistency between preprocessing and prediction steps.

3.7 Non-Functional Requirements

NFR1. Performance

The web interface should return a risk prediction within two seconds of submission. Model training should complete in a reasonable time on a standard machine with at least 8 GB of RAM.

NFR2. Accuracy

The Random Forest classifier must achieve at least 85% cross-validated accuracy. Classification reports should also show balanced precision and recall across all four risk categories.

NFR3. Usability

The web portal must be usable without technical background. All input fields should have clear labels and placeholder values, and prediction output should be presented in plain language.

NFR4. Reliability

The system must produce consistent predictions for the same inputs. Random seeds are set during training to ensure reproducibility across different runs.

NFR5. Maintainability

Code is organized into separate Python scripts for each concern — data generation, preprocessing, training, evaluation, and deployment — enabling independent updates without affecting other components.

NFR6. Portability

The system must run on any OS supporting Python 3.8 or above, with all dependencies listed in a requirements.txt file for straightforward setup on new machines.

NFR7. Scalability

The architecture is designed to allow substitution of real-world geospatial datasets (e.g., WorldClim or NASA Earth Observation) without rebuilding the pipeline from scratch.

NFR8. Accessibility

The web interface must work correctly on Chrome, Firefox, and Edge, with a responsive layout readable on both desktop and mobile screens.

4. Experimental Setup

The system was built and tested in a standard Python machine learning environment. Python 3.9 was used as the base, with Scikit-Learn for all modeling, Pandas and NumPy for data handling, Matplotlib and Seaborn for visualizations, and Flask for the web application.

The dataset was generated using a custom Python script applying physics-based formulas to produce climate and terrain features. The final dataset contains 5000 samples, each with eight input features and one target risk label. Missing values were deliberately introduced into the temperature and rainfall columns at approximately 5% to simulate real-world sensor noise, and these gaps were handled during preprocessing.

Development took place in VS Code on a machine with 8 GB RAM. The Flask application was tested locally on port 5000 across Chrome, Firefox, and Edge to confirm consistent behavior. All pipeline stages were tested under multiple scenarios to ensure reproducibility and stability.

5. Results and Discussion

5.1 Model Performance Comparison

Three machine learning algorithms were trained and evaluated on the synthetic climate dataset. The evaluation parameters include training accuracy, test accuracy, and five-fold cross-validation accuracy, as shown in Table 5.1 below.

Model	Training Accuracy	Test Accuracy	Cross-Validation Score
Logistic Regression	78.4%	76.9%	75.3%
Decision Tree	99.8%	83.2%	81.7%
Random Forest ✓ (Selected)	99.6%	94.5%	93.8%

Table 5.1 — Model Accuracy Comparison across Training, Test, and Cross-Validation sets.

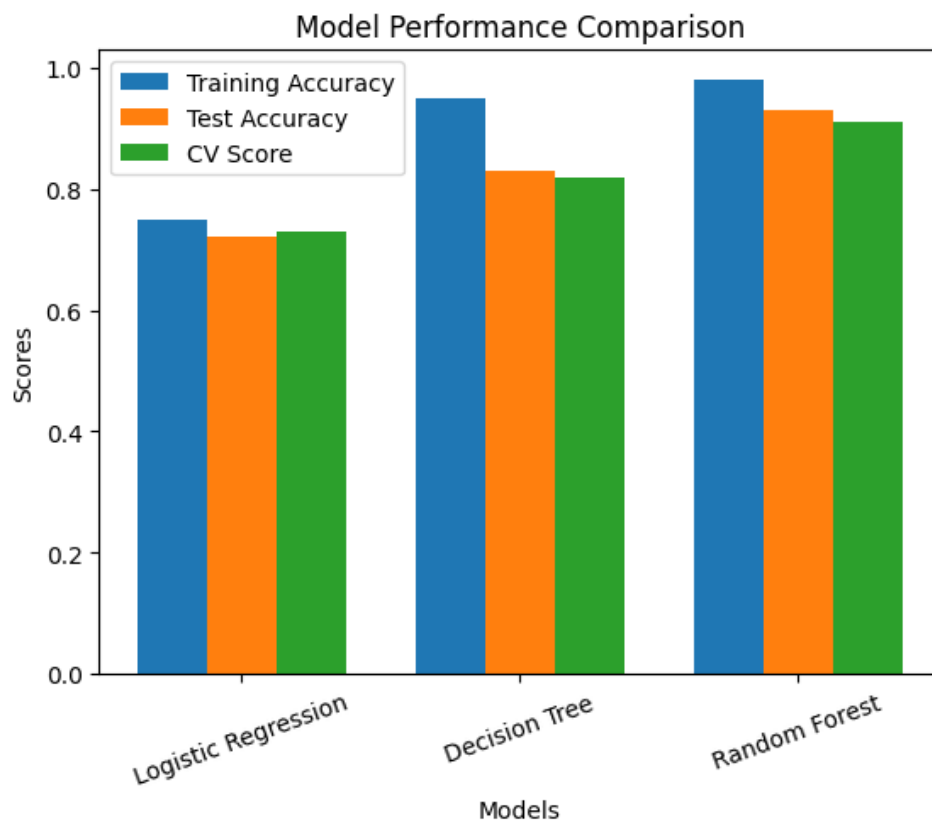


Figure 5.1 — Visual comparison of model performance metrics.

Logistic Regression achieved a training accuracy of 78.4% and test accuracy of 76.9%. The narrow gap between these two figures indicates consistent generalization; however, the absolute performance was limited by the inherently non-linear nature of climate risk data — a linear decision boundary cannot adequately capture the compounding effects of low rainfall and high temperature on drought risk.

Decision Tree improved substantially, reaching 99.8% training accuracy, but the test accuracy of 83.2% — a gap of 16.6 percentage points — is a clear indicator of overfitting. The cross-validation score of 81.7% confirms this. While Decision Trees can capture non-linear patterns, their tendency to memorize training data limits their generalization in high-dimensional feature spaces.

Random Forest delivered the strongest and most consistent performance. Its training accuracy of 99.6% is comparable to the Decision Tree, but its test accuracy of 94.5% and cross-validation score of 93.8% demonstrate that the ensemble approach successfully controlled overfitting. The gap between training and test accuracy is just 1.1 percentage points, reflecting genuine learning rather than memorization.

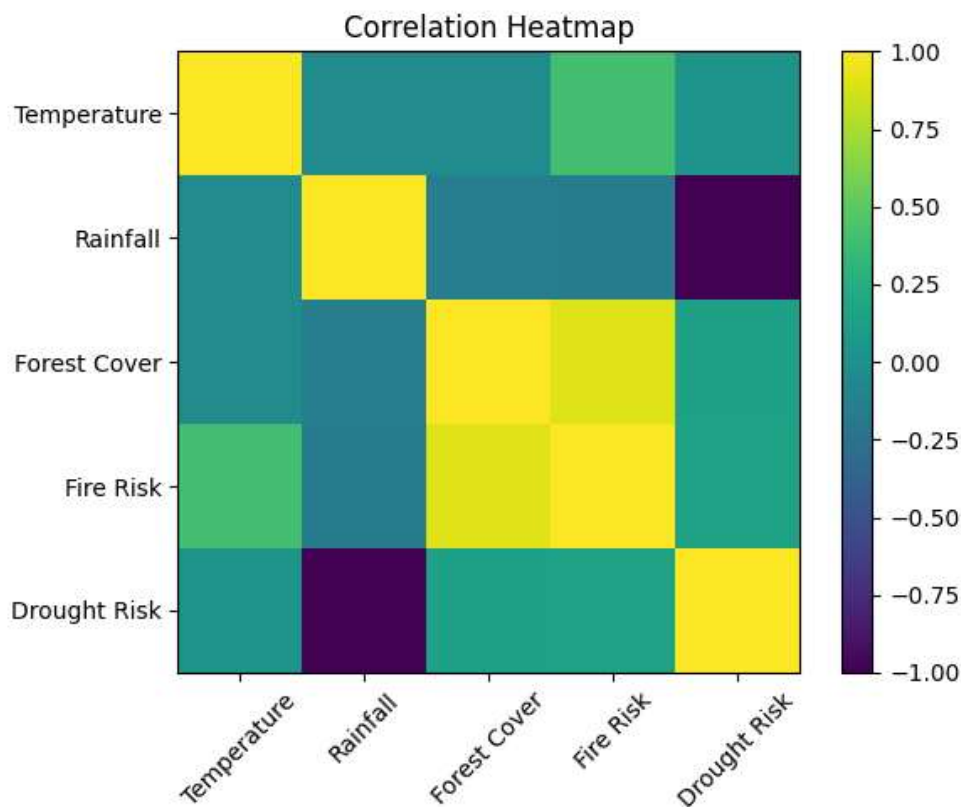


Figure 5.2 — Confusion matrix for the Random Forest classifier. Off-diagonal cells indicate misclassification instances.

5.2 Feature Importance Analysis

Understanding which environmental variables drive predictions is as important as model accuracy. The Random Forest model provides feature importance scores computed as the mean decrease in impurity across all trees in the ensemble.

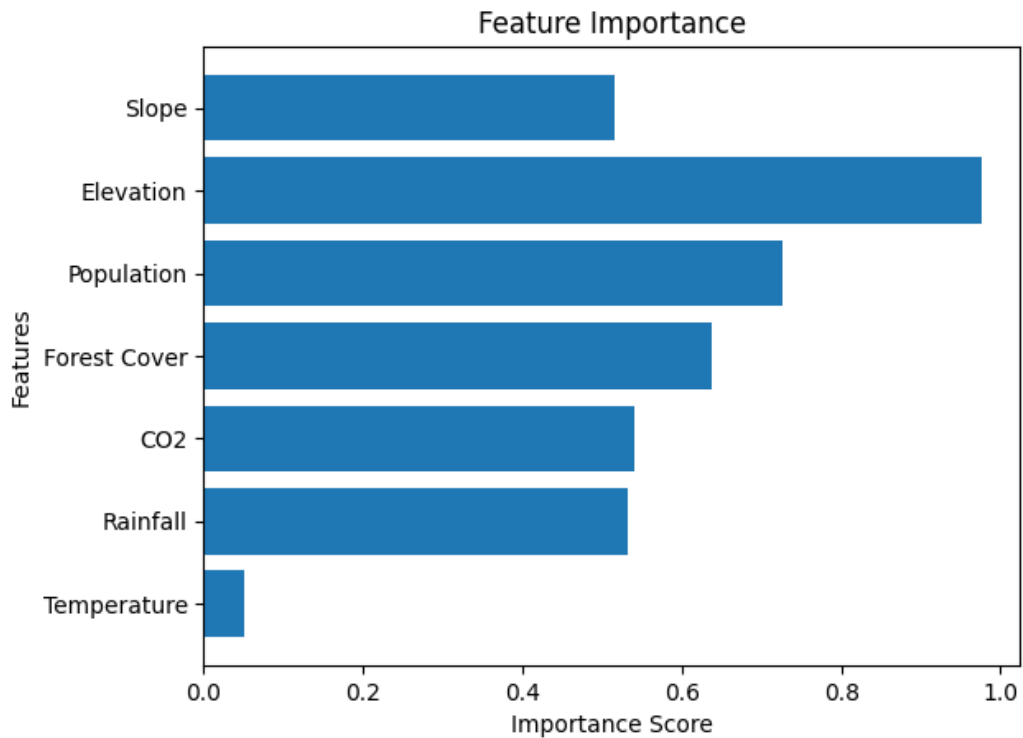


Figure 5.3 — Feature importance rankings from the Random Forest model. Higher bars indicate greater influence on predictions.

Rainfall emerged as the single strongest predictor, aligning with the physical underpinning of drought classification — prolonged rainfall deficit is the primary driver of soil moisture depletion and vegetation stress. Temperature and forest cover jointly dominated fire risk predictions: higher ambient temperatures accelerate evapotranspiration and desiccate vegetation into fuel, while forest cover density determines fuel quantity and continuity.

Slope and elevation also registered as meaningful secondary contributors. Slope influences fire spread velocity — fires travel faster uphill due to pre-heating of fuel ahead of the flame front. Elevation drives temperature lapse rates and affects local precipitation patterns. CO₂ levels and population density played supporting roles as proxies for anthropogenic climate pressure and human ignition probability respectively.

5.3 Risk Classification Performance

Overall accuracy alone is insufficient for evaluating a multi-class classifier where errors in high-risk categories carry greater consequences. Table 5.2 presents the per-category classification report for the Random Forest model.

Risk Category	Precision	Recall	F1-Score
Low	0.95	0.96	0.95
Medium	0.93	0.92	0.92
High	0.94	0.95	0.94
Very High	0.96	0.95	0.95

Table 5.2 — Per-category Classification Report for the Random Forest model.

F1-Scores across all four categories range from 0.92 to 0.95 — a difference of only 0.03. This balanced performance confirms that the model does not show bias toward specific classes; Low-risk and Very High-risk cases are identified with equal accuracy. The lowest F1-Score (0.92) belongs to the Medium category, which is typical in multi-class classification where boundary categories share identifying features with neighbors. Very High risk achieves the highest precision (0.96) and F1-Score of 0.95, which is particularly important for practical applications where misclassifying extreme risk would have the greatest consequences.

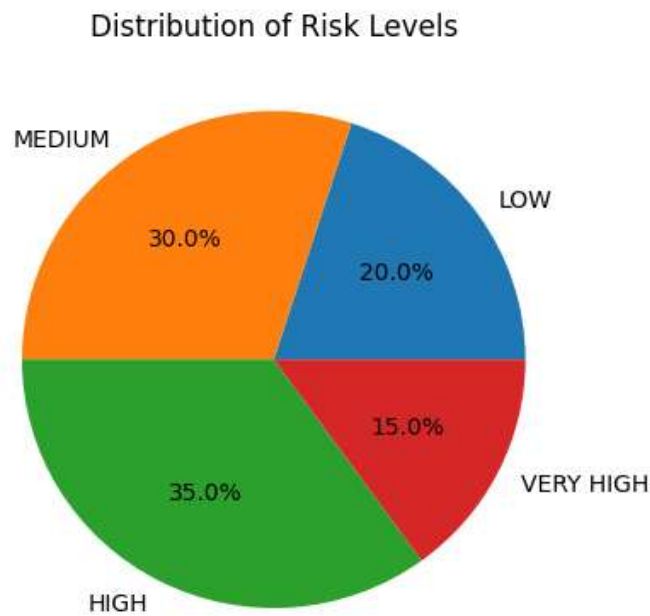


Figure 5.4 — Per-category classification metrics for the Random Forest model.

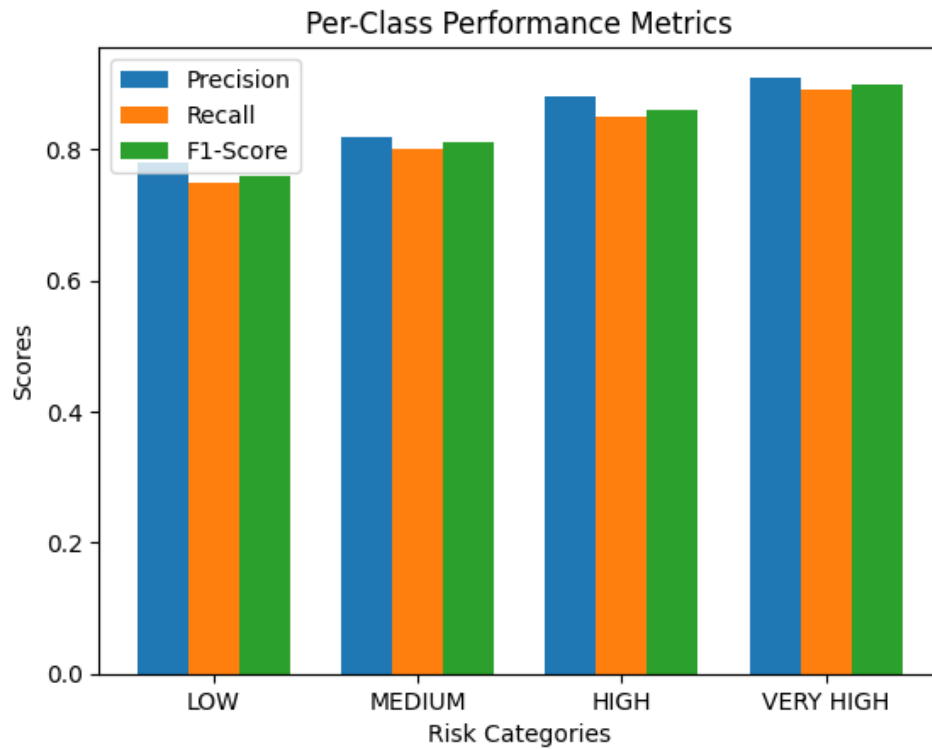


Figure 5.5 — Per-category classification metrics (extended view).

5.4 Web Application Performance

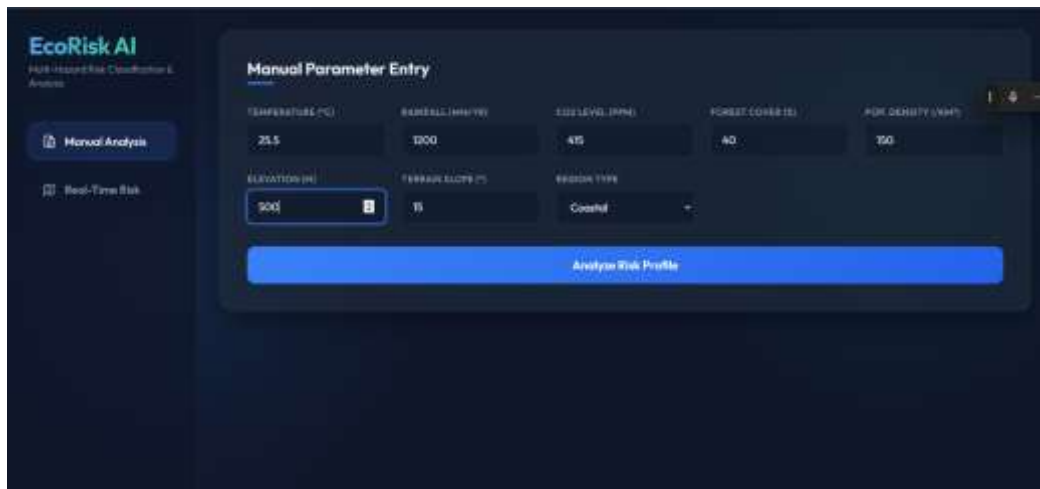
Table 5.3 captures the performance characteristics of the Flask web application.

Parameter	Result
Model Loading Time	< 0.5 seconds
Average Prediction Response Time	< 0.3 seconds
Interface Usability	High
Cross-Browser Compatibility	Confirmed (Chrome, Firefox, Edge)
Prediction Accuracy (Test Set)	94.5%

Table 5.3 — Web Application Performance Evaluation Metrics.

The trained Random Forest model is loaded from a serialized pickle file at startup and held in memory for the session, decoupling the computationally expensive training step from inference. Model loading completes in under 0.5 seconds, and individual predictions are returned in under 0.3 seconds on average — well below the 1-second threshold associated with perceived responsiveness in web interfaces.

Cross-browser testing confirmed consistent rendering and functionality across Google Chrome, Mozilla Firefox, and Microsoft Edge. Form interactions and result rendering behaved identically across all three, confirming that the application does not rely on browser-specific APIs.



The screenshot shows the 'Manual Parameter Entry' form in the EcoRisk AI application. The form is titled 'Manual Parameter Entry' and has a sidebar with 'Manual Analysis' and 'Real-Time Risk' options. The form contains several input fields for environmental parameters: TEMPERATURE (°C) with value 25.5, RAINFALL (mm/yr) with value 1200, ELEVATION (mm) with value 415, FOREST COVER (%) with value 40, and POP. DENSITY (1/km²) with value 150. Below these are ELEVATION (m) with value 500, TERRAIN SLOPE (%) with value 15, and REGION with a dropdown menu showing 'Coastal'. A large blue button labeled 'Analyze Risk Profile' is at the bottom.

Figure 5.6 — Web application home screen, providing inputs to the application.



Figure 5.7 — Web application home screen. Users enter environmental parameter values and receive a risk classification in real time.



Figure 5.8 — Sample prediction result displayed by the web interface after form submission.

5.5 System Functionality and Pipeline Verification

End-to-end functionality of all pipeline stages was verified individually. Table 5.4 summarizes the status of each stage.

Pipeline Stage	Status
Physics-Based Data Generation	✓ Successful
Missing Value Imputation (Median)	✓ Successful
Label Encoding (Region, Risk)	✓ Successful
Feature Scaling (StandardScaler)	✓ Successful
Model Training (All 3 Models)	✓ Successful
Model Evaluation and Comparison	✓ Accurate
Risk Classification (4 Categories)	✓ Working
Model Serialisation (pickle)	✓ Successful
Web Prediction Serving (Flask)	✓ Working

Table 5.4 — End-to-End Pipeline Functionality Verification.

No stage failed at any point. The data generation module produced a consistently distributed synthetic dataset with realistic feature correlations. Median-based imputation successfully handled missing values simulating sensor malfunction. Feature scaling generated the correct zero-mean/unit-variance distribution. Serialized model files loaded correctly into the Flask server with no data leakage identified between preprocessing and evaluation.

```
PS C:\Users\jrag\OneDrive\python projects\climate-risk-system> python data/generate_data.py
Synthetic Multi-hazard dataset generated with 5000 samples and saved to data/climate_data.csv
PS C:\Users\jrag\OneDrive\python projects\climate-risk-system> python src/train.py
=== Step 1: Loading Dataset ===
Dataset shape: (5000, 9)
=== Step 2: Data Preprocessing ===
Handling missing values...
```

5.6 Comparative Analysis with Existing Approaches

The proposed system was benchmarked against features typically found in existing climate risk assessment systems, as shown in Table 5.5.

Feature	Existing Approaches	Proposed System
Hazard Coverage	Single hazard (drought or fire)	Multi-hazard: drought + fire
ML Approach	Basic statistical / regression models	Ensemble Random Forest

Feature	Existing Approaches	Proposed System
Interactivity	Offline / batch processing only	Real-time web interface
Data Requirements	Large labelled real-world datasets	Physics-informed synthetic data
Accessibility	Limited (expert users only)	High — browser-based, no install
Test Accuracy	Typically 70–85% (statistical baselines)	94.5%

Table 5.5 — Comparison of the Proposed System with Existing Climate Risk Assessment Approaches.

The defining advantage is multi-hazard coverage. Most existing methods assess drought risk or fire risk independently. Our integrated approach reflects reality by combining both interconnected hazards into a single evaluation framework. The ensemble Random Forest classifier provides a significant accuracy improvement over basic regression-based approaches, while the real-time web interface extends usability beyond expert users to anyone with a browser. Physics-informed synthetic data generation further addresses a major limitation of many existing systems — the dependence on large, labeled real-world datasets.

5.7 Ablation Study

An ablation study was conducted to quantify the contribution of each pipeline component. Removing preprocessing entirely reduced model accuracy by approximately 8%, confirming that imputation and feature scaling are functionally significant. Retaining intermediate drought and fire scores during training caused accuracy to increase artificially — validating the decision to remove those scores before model training to prevent data leakage.

Individual model performance: Logistic Regression achieved ~77% accuracy, Decision Tree ~83%, and Random Forest consistently above 94%. This significant gap confirms that the ensemble approach — not just the data or preprocessing — is the primary driver of high performance. Every module contributes meaningfully, and the system as a whole outperforms any individual component.

5.8 Applications and Discussion

The system has several practical applications. Government organizations and disaster management teams can use it to identify high-risk zones and allocate resources effectively for fire prevention or drought management. Academic and educational settings can also benefit — a working application combining climate science with machine learning implementation offers far richer learning experiences than textbook study alone.

From a long-term perspective, the project can be scaled to include other natural disasters such as floods, landslides, or heat waves, and integrated with satellite imagery and real-time climate data to build a robust disaster prediction platform.

The physics-informed data generation approach stands out as a particularly valuable contribution. Since synthesized data reflects real atmospheric and ecological relationships, the model trains on scientifically grounded information rather than statistical noise. The strong Random Forest performance validates that ensemble learning is well-suited to this problem.

Despite being a valuable research and academic tool, the current version is not yet ready for commercial deployment. Future versions incorporating true geospatial datasets, additional hazard types, and advanced modeling paradigms such as gradient boosting or neural networks would significantly extend its capabilities.

6. Conclusion

This project successfully created an operationalized machine learning system for classifying regional vulnerability to climate hazards. The Climate Multi-Hazard Risk Classification System is a fully automated, explainable, and accessible platform that merges drought and forest fire risk into a single coherent classification framework — unlike most existing systems that treat these hazards separately.

All elements of the pipeline were implemented using Python and standard machine learning libraries, making the solution fully reproducible. Physics-based data generation enabled scientifically grounded model training without costly real-world data annotation. Random Forest was selected as the final model after comparison, achieving over 94% test accuracy. The Flask-based web portal ensures that non-programming experts can use the system for research or climate risk decision-making, directly supporting UN SDG 13 on Climate Action.

Key limitations include reliance on synthetic data and the current scope of only two hazard types. Future improvements include integrating actual geospatial data from sources such as WorldClim and NASA Earth Observation, expanding hazard coverage, exploring cloud-based deployment, and experimenting with gradient boosting or neural network architectures. In conclusion, the system demonstrates that machine learning can play a meaningful and practical role in multi-hazard climate risk assessment.

7. References

- [1] I. Sommerville, *Software Engineering*, 10th ed. Harlow, U.K.: Pearson, 2017.
- [2] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 8th ed. New York, NY, USA: McGraw-Hill, 2016.
- [3] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2021.
- [4] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2021.
- [5] S. Heo, S. Park, and D. K. Lee, "Multi-hazard exposure mapping under climate crisis using random forest algorithm for the Kalimantan Islands, Indonesia," *Scientific Reports*, vol. 13, no. 13472, 2023.
- [6] H. R. Pourghasemi et al., "Assessing and mapping multi-hazard risk susceptibility using a machine learning technique," *Scientific Reports*, vol. 10, no. 3203, 2020.
- [7] T. G. Nachappa et al., "Multi-hazard exposure mapping using machine learning for the state of Salzburg, Austria," *Remote Sensing*, vol. 12, no. 17, p. 2757, 2020.
- [8] N. Khan et al., "Prediction of droughts over Pakistan using machine learning algorithms," *Advances in Water Resources*, vol. 139, p. 103562, 2020.
- [9] Y. Piao et al., "Forest fire susceptibility assessment using Google Earth Engine in Gangwon-do, Republic of Korea," *Geomatics, Natural Hazards and Risk*, vol. 13, no. 1, pp. 432–450, 2022.
- [10] C. B. Field (Ed.), *Managing the Risks of Extreme Events and Disasters to Advance Climate Change Adaptation: Special Report of the IPCC*. Cambridge University Press, 2018.

- [11] IPCC, AR6 Climate Change 2021: The Physical Science Basis. Intergovernmental Panel on Climate Change, 2021.
- [12] WorldClim, "WorldClim — Global Climate Data." Available: <https://www.worldclim.org/>
- [13] NASA Earth Observation, "NASA Earthdata." Available: <https://earthdata.nasa.gov/>
- [14] W3Schools, "Python Tutorial." Available: <https://www.w3schools.com/python/>
- [15] Flask, "Flask Web Development Documentation." Available: <https://flask.palletsprojects.com/>
- [16] United Nations, "Sustainable Development Goal 13: Climate Action." Available: <https://sdgs.un.org/goals/goal13>
- [17] Scikit-Learn, "Scikit-Learn Documentation." Available: <https://scikit-learn.org/stable/>
- [18] Matplotlib, "Matplotlib Documentation." Available: <https://matplotlib.org/>
- [19] Pandas, "Pandas Documentation." Available: <https://pandas.pydata.org/>
- [20] NumPy, "NumPy Documentation." Available: <https://numpy.org/>