Project Development Phase
Model Performance Test

Date	16 February 2026
Team ID	LTVIP2026TMIDS87521
Project Name	Rising Waters: A Machine Learning Approach to Flood Prediction
Maximum Marks	10 Marks

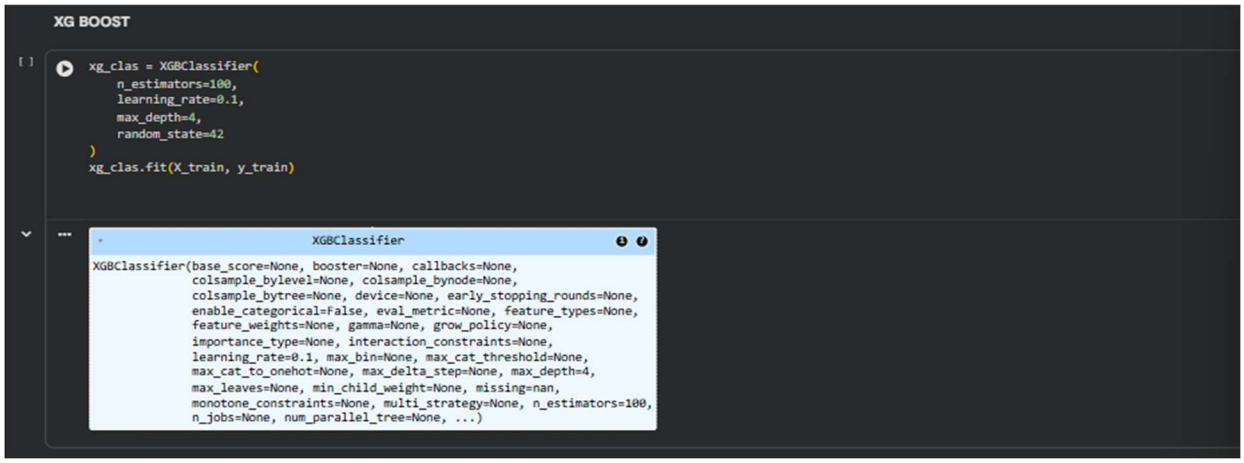
Model Building, Training, and Performance Testing

Model Building

The model building phase involved implementing multiple machine learning classification algorithms to predict flood risk based on environmental parameters. Algorithms such as Decision Tree, Random Forest, Support Vector Machine (SVM), Extra Trees, and XGBoost were developed using Python and relevant machine learning libraries.

The dataset features such as rainfall, temperature, humidity, and other climatic factors were used as input variables, while the flood occurrence label (High / Low risk) was used as the target variable. Each algorithm was configured with appropriate hyperparameters to ensure optimal learning performance.

A function named xgboost is created and train and test data are passed as the parameters. Inside the function, GradientBoostingClassifier algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.



Model Training

During the training phase, the dataset was divided into training and testing sets, typically using an 80:20 split ratio. The training dataset was used to allow the models to learn patterns and relationships between environmental parameters and flood risk.

For XGBoost, the training process involved sequentially building multiple decision trees using gradient boosting optimization. Each tree focused on reducing the errors made by the previous trees. Hyperparameters such as learning rate, number of estimators, and maximum tree depth were tuned to improve accuracy and reduce overfitting.

Performance Testing

After training, the models were evaluated using the testing dataset to measure their generalization performance. Several evaluation metrics were used, including:

- Accuracy to measure overall prediction correctness
- Precision to evaluate correctness of positive flood predictions
- Recall to measure the model's ability to detect actual flood cases
- F1-Score to balance precision and recall
- Confusion Matrix to analyze true positives, true negatives, false positives, and false negatives

```
[ ]
from sklearn import metrics
import pandas as pd

results = []

models = {
    "KNN": p1,
    "Decision Tree": p2,
    "Random Forest": p3,
    "XGBoost": p4
}

for name, pred in models.items():
    acc  = metrics.accuracy_score(y_test, pred)
    prec = metrics.precision_score(y_test, pred)
    rec  = metrics.recall_score(y_test, pred)
    f1   = metrics.f1_score(y_test, pred)

    results.append([name, acc, prec, rec, f1])

results_df = pd.DataFrame(results,
                          columns=["Model","Accuracy","Precision","Recall","F1-Score"])

print(results_df)
```

```
           Model  Accuracy  Precision    Recall  F1-Score
0            KNN  0.896552        0.5  1.000000  0.666667
1  Decision Tree  0.965517        1.0  0.666667  0.800000
2  Random Forest  0.965517        1.0  0.666667  0.800000
3        XGBoost  0.965517        1.0  0.666667  0.800000
```

Comparative analysis of all algorithms was performed. Among them, XGBoost achieved the highest accuracy and better overall performance metrics. It also demonstrated strong ability in handling nonlinear relationships and minimizing prediction errors.

Based on these results, XGBoost was selected as the final model and deployed into the web application for real-time flood risk prediction.