Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2025) 000–000

Procedia
Computer Science

Natural Disaster Prediction using Machine Learning

A R Haripriya[a], K A Harinandana[a]

[a]Department of Computer Science and Engineering,School of Computing, Amrita Vishwa Vidyapeetham, Amritapuri

Abstract

Natural disasters, like earthquakes, floods, and cyclones, are serious dangers to people and buildings. It is very important to predict these events in time so that we can prepare for and respond to them well. This paper shows a machine learning system that uses a mixed way of modeling, including both grouping and estimating methods, to predict natural disasters. The system uses different data sources including location data and details specific to each kind of disaster. A careful process gets rid of unusual data, balances the data groups using SMOTE, and adjusts the data ranges to make the model work better. Grouping methods like Random Forest, Support Vector Machine, Logistic Regression, and Gradient Boosting are carefully adjusted using GridSearchCV to guess the kind of disaster based on location details. At the same time, estimating models like SVR, Random Forest Regressor, Ridge, Lasso, and XGBoost are used to guess how bad the disasters will be by measuring their size. The system is tested using measurements like Accuracy, F1-Score, and AUC-ROC for grouping, and R², RMSE, and MAE for estimating. It is shown as an easy-to-use web application using Streamlit, which gives up-to-date predictions, risk reviews, and visual data. The suggested answer shows very correct predictions and descriptions, making it a helpful tool for making choices to lower disaster risks and plan for emergencies.

Keywords: Disaster Prediction; Machine Learning; Classification; Regression; Streamlit; GridSearchCV; SMOTE

1. Introduction

Natural disasters like floods, earthquakes, and cyclones do a lot of damage to the environment. They hurt people, animals, and plants. These things happen suddenly, without much warning. So, predicting disasters is very important for keeping life going. If we can guess what kind of disaster will happen and how extreme it will be, people and governments can get ready early, which lowers damage and saves lives. Many researchers have tried different ways to guess disasters. Most of these ways can be put into three groups: Standard Machine Learning, Deep Learning, and Mixed Models. In Standard Machine Learning, methods like Decision Trees or Support Vector Machine (SVM) are good for data that is organized and easy to understand. But they cannot deal with difficult patterns. So, in this case, Deep Learning methods, like neural networks, are good at finding hidden patterns in big datasets, like images from satellites or weather maps. But they need a lot of data and strong machines to work. This is where Mixed Models are useful. They join Deep Learning and Machine Learning, like using deep learning to find features and then training a machine learning model to make predictions. But each of these has good and bad sides. Most studies only look at one kind of disaster, like guessing earthquakes, and do not predict many types. In our work, we tried to make a system that can guess the type of natural disaster (earthquake, cyclone, or flood) and how strong it might be. Our system only needs latitude and longitude as data. Then, it uses different machine learning models to guess what kind of disaster might happen in that place and how extreme it might be. There are different models for each job: grouping models to guess the type of disaster, and estimating models to guess how strong it could be. Before training models, we made the data cleaner, got rid of incorrect and high values, and adjusted some group sizes using SMOTE. Our project is special because we made one system that can guess many disasters. We joined and cleaned data from places like Kaggle and

USGS. we added value for the datasets where there were actual value missing and created a web app using Streamlit where you can type in a place and get the prediction. In the next parts, we will explain: Section 2 will talk about the datasets used. Section 3 will explain the ways we took and why they were needed. Section 4 will explain the results and talks. Section 5 will try to sum up the points we have made.

2. Dataset Collection

In this project, we used three location-based datasets related to earthquakes, floods, and cyclones. These datasets were taken from public sources. Each dataset has location details like latitude and longitude, along with disaster-specific signs of how bad it is, to make a single system for predicting many kinds of disasters. These datasets were worked on, adjusted, and put into a common form to do both grouping and estimating tasks.

Datasets used

- **Earthquake Dataset**: This dataset was from the United States Geological Survey (USGS). It has over 5000 records of past earthquakes. Each record includes latitude, longitude, and magnitude as details.
- **Flood Dataset**: The flood dataset was from Kaggle, which looks at flood-likely areas in India and has location coordinates. This dataset did not have the strength marks, so to match the earthquake and cyclone datasets, made-up magnitude values were created using a regular spread from 1.0 to 5.0. This dataset lacks the severity indicators; so to align with earthquake and cyclone datasets synthetic magnitude values are generated using a uniform distribution ranging between 1.0 and 5.0.
- **Cyclone Dataset**: This dataset has records of past cyclones from the Pacific area and includes details like latitude, longitude, and Maximum Wind. Because cyclones are measured by wind speed, this value is adjusted and used as a strength score for cyclones.

Table 1. Summary of Dataset Characteristics

Property	**Value**
Total Samples	Earthquake: 10,513, Flood: 10,001, Cyclone: 26,138
Disaster Types	Earthquake, Flood, Cyclone
Severity Type	Real (Earthquake), Synthetic (Flood), Proxy (Cyclone)
Features Used	Latitude, Longitude, Magnitude / Wind Speed
Feature Type	Continuous Numeric
Missing Values	None
Source	USGS, Kaggle, Public Cyclone Dataset

3. Approach Adopted

3.1. Data Preprocessing

We took datasets for earthquakes, floods, and cyclones from places like Kaggle and USGS. Each dataset had a different form and amount of data, so we used different steps to make the data more similar.

Data Cleaning: We got rid of bad entries like values missing, non-number data, and incorrect location ranges. Latitude values were given between -90 to +90 degrees and longitude from -180 to +180 degrees. Negative or zero powers were also gotten rid of. Clean data makes sure that the model learns real patterns without being confused by incorrect or not important data points.

Outlier Removal: The Interquartile Range (IQR) way was used to find and get rid of the outliers in latitude, longitude, and magnitude. This kept only the middle 50 percent of the data and got rid of values that were too far from the average range. Outliers can make models learn wrongly, especially in estimating tasks. Getting rid of them makes the models more right.

Synthetic Magnitude: Using synthetic values means a single estimation model can be trained across all disaster types. This keeps things the same and makes the model better. In our study, the flood dataset did not have a measuring power value. So, to make it more able to go with other datasets, we made synthetic data magnitudes using the uniform spread from 1 to 5

Class Balancing with SMOTE: Datasets that are not balanced may make the model prefer the bigger group. SMOTE makes sure that all datasets used are learned equally. In our study, earthquake records were much more than the other datasets, which made things unbalanced. We used SMOTE here to make synthetic examples of the smaller datasets, which were floods and cyclones.

Feature Scaling: Some models like SVM and SVR are more changed by the size of input data. Scaling stops one part from controlling the learning. Here, we used StandredScaler to make latitude and longitude regular, which were the parts. This gives all parts 0 as the middle and 1 as the change.

Train-Test Split and Cross Validation: Cross validating always makes sure the model guesses well and doesn't guess only on one test split. Here we split the dataset into 80 percent as training data and 20 percent as test split. We also did 3 fold cross validation to check model working more surely.

3.2. Models Used

We have used many machine learning models for the main two tasks: Grouping for guessing the kind of disaster that happened (Earthquake, Flood, and Cyclone) and the risk level (High, Medium, and Low), and Estimating for guessing the size of the disaster. To check the checking, we tried simple and complex models. This helped us learn which model worked best for different data patterns.

Classification Models used:

- **Random Forest Classifier**: It is a group model that makes many decision trees. It gives the most common guess. It lowers model guessing, handles the noisy data, and works well on linear and non linear forms of data. It tells how important parts are, which helps in checking how latitude and longitude work.

$$\hat{y} = \text{majority\_vote}(T_1(x), T_2(x), \ldots, T_n(x))$$

- **Support Vector Machine**: SVM makes a separation line that splits the groups in the dataset. It is strong in high space and good when the gap between the groups is clear. It sometimes does better than simple models on datasets that are worked on well.

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right)$$

- **Gradient Boosting Classifier**: Gradient Boosting Classifier learns hard connections between parts and guesses well. It makes a group of trees one after another, where each tree tries to fix the errors made in the trees before.

$$\hat{y} = \text{sign}\left(\sum_{m=1}^{M} \gamma_m h_m(x)\right)$$

- **Logistic Regression**: It is a linear model that uses a sigmoid function to put data into groups. It is fast, clear, and easy to use, making it a good model to measure against. It also checks if a straight decision line was good enough.

$$P(y = 1 \mid x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}}$$

- **K-Nearest Neighbour (KNN)**: It is a non linear and easy to use model. It checks how local and distance affect the disaster type guess. It groups a point based on the most common group among its k nearest neighbors

$$\hat{y} = \text{mode}(y_{(i)} \mid x_{(i)} \in \mathcal{N}_k(x))$$

Regression Models used:

- **Support Vector Regressor (SVR)**: It handles non straight guessing well using kernal functions, and it is useful even in small datasets. It tries to find a function that is away from an actual aim by a margin less than ϵ for most points.

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + b$$

$$|y_i - f(x_i)| \leq \epsilon$$

- **Random Forest Regressor**: It is a group of decision trees trained on different data pieces and averaged for the result. It is strong against outliers, lowers change, and does well on complex and non straight datasets. It also handles guessing stronger than a single tree.

$$\hat{y} = \frac{1}{n} \sum_{i=1}^{n} T_i(x)$$

- **Ridge and LASSO Regression**:They are linear models that use L2 and L1 regularization. They are useful when data parts are involved strongly. Ridges help in lowering the hardness of the model. LASSO chooses parts by making some numbers zero.

$$\hat{\beta} = \arg\min_{\beta} \left(\sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda \|\beta\|_2^2 \right)$$

$$\hat{\beta} = \arg\min_{\beta} \left(\sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda||\beta||_1 \right)$$

- **XGBoost Regressor**: It is a changed and regularized version of gradient boosting designed for good working and right guessing. It always gave the best results in power guessing because it can handle data that is not full, missing values, and model hardness well.

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F}$$

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2$$

We chose these models to compare how simple linear models, distance models and complex methods perform under same dataset and feature set. This helped to identify the best trade off between interpretability, training time and prediction accuracy for both classification and regression tasks.

3.3. Hyperparameter Tuning

In our study, we used a method **GridSearchCV** to find the best parameters for each model. Grid Search works by trying out every possible combination of the given options and using cross validation to measure how well each one performs. This helped us to make sure that each of the model was tuned specifically for our dataset and performed its best. Without this step, the models may have used default values that are not ideal and could lead to poor predictions. To give an example, for the Random Forest classifier, we tested different numbers of trees and depths and found that using 200 trees and a depth of 20 gave us the best results. Similar searches were done for all other models. To improve model performance, we used GridSearchCV to tune each model's hyperparameters. This method tests all combinations of given parameters using cross-validation to find the best settings.

Table 2. Hyperparameter Tuning - Classification Models

Model Name	**Tuned Hyperparameters**
Random Forest Classifier	n_estimators = 200, max_depth = 20, min_samples_split = 2
Support Vector Machine	C = 1, kernel = 'rbf', gamma = 'scale'
Gradient Boosting Class.	n_estimators = 100, learning_rate = 0.1, max_depth = 5
Logistic Regression	C = 1, penalty = 'l2', solver = 'lbfgs'
K-Nearest Neighbors	n_neighbors = 5, weights = 'uniform'

Table 3. Hyperparameter Tuning - Regression Models

Model Name	**Tuned Hyperparameters**
Support Vector Regressor	C = 1, kernel = 'rbf', gamma = 'scale'
Random Forest Regressor	n_estimators = 200, max_depth = 20, min_samples_split = 2
Ridge Regression	alpha = 1.0
Lasso Regression	alpha = 0.1
XGBoost Regressor	n_estimators = 200, max_depth = 5, learning_rate = 0.1

3.4. Evaluation Metrics and Relevance

We evaluated the models using appropriate metrics for both classification and regression. These helped us to understand how close the magnitude predictions are to the actual values. Since predicting severity of the disaster can affect planning decisions accurate estimation is essential.

Classification Metrics

- **Accuracy**: Proportion of the correct predictions

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision**: Proportion of positive identifications that were incorrect

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall**: Proportion of the actual positively identified

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1 Score**: Harmonic mean of precision and recall. It is useful when classes are imbalanced.

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics together provide a complete picture of classification performance. In disaster prediction, both false alarms which is the low precision and missed events which is the low recall are costly, so we use multiple metrics.

Regression Metrics

- **R Square Score**: Proportion of variance in the target variable explained by the model.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

- **RMSE**: Penalizes large errors, giving insight into prediction accuracy.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **MAE**: Average of all absolute differences between actual and predicted values.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **Explained Variance**: Measures how much of the variation in the target variable is captured by the model. A score close to 1.0 means the model explains most of the variability, while a score near 0 indicates limited explanatory power.

$$\text{Explained Variance} = 1 - \frac{\text{Var}(y - \hat{y})}{\text{Var}(y)}$$

These metrics help us understand how close our magnitude predictions are to the actual values. Since predicting disaster severity can influence planning decisions, accurate estimation is critical.

4. Results and Discussion

4.1. Results

4.1.1. Experiment 1: Disaster Type and Risk Level Classification

Table 4. Classification Performance of Different Models

Models	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Random Forest	89.05%	0.8900	0.8908	0.8905
SVM	81.70%	0.8165	0.8198	0.8100
Logistic Regression	65.53%	0.6551	0.6556	0.6550
KNN	**89.07%**	**0.8907**	**0.8911**	**0.8900**
Gradient Boosting	88.73%	0.8700	0.8882	0.8870

This summarizes the performance of five classification models evaluated on the task of predicting the type of natural disaster. The performance metrics considered are Accuracy, F1-score, Precision, and Recall. Among the models tested, the K-Nearest Neighbors (KNN) model demonstrated the best overall performance across all four evaluation metrics.

It achieved the highest accuracy of 89.07 percent, an F1-score of 0.8907, precision of 0.8911, and recall of 0.8900, narrowly outperforming the Random Forest and Gradient Boosting models.

The Random Forest Classifier also showed strong results, with an accuracy of 89.05 percent and an F1-score of 0.8900. Its precision and recall were also very close to those of KNN, indicating consistent and balanced predictions. The Gradient Boosting Classifier, while slightly behind in overall accuracy (88.73 percent), still maintained a competitive F1-score of 0.8700 and showed particularly high precision (0.8882), suggesting its predictions were often correct when a specific disaster class was identified.

On the other hand, the Support Vector Machine (SVM) and Logistic Regression models performed less favorably. SVM achieved an accuracy of 81.70 percent with an F1-score of 0.8165, while Logistic Regression lagged with an accuracy of only 65.53 percent and an F1-score of 0.6551. This indicates that simpler, linear models like Logistic Regression struggled to capture the complexity in the disaster classification problem, likely due to overlapping patterns in location-based features like latitude and longitude.

Overall, the results highlight that non-linear, ensemble, and distance-based models such as KNN, Random Forest, and Gradient Boosting are better suited for multi-class disaster classification. The superior performance of these models may be attributed to their ability to model complex decision boundaries and handle variations in the spatial distribution of disaster events.

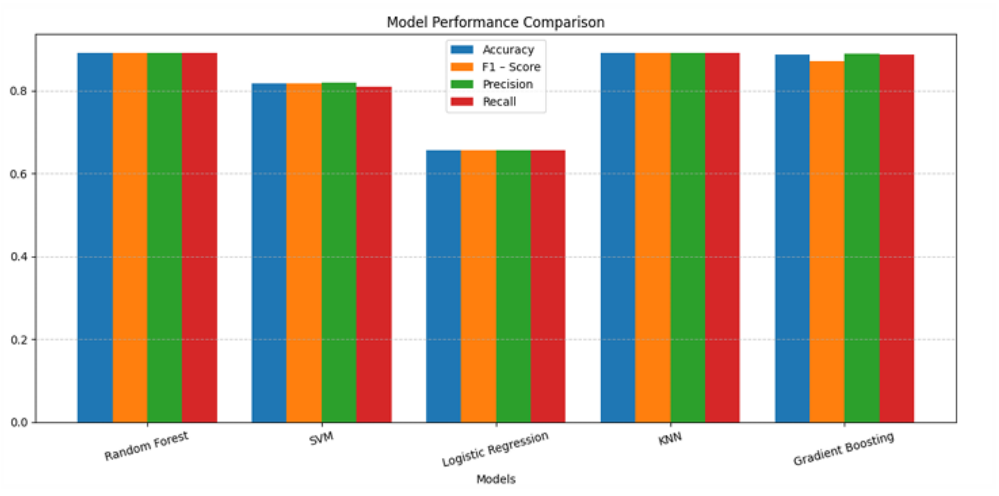

Figure 1. Histogram for the results of the classification model.

The above figure shows a side-by-side comparison of the classification models based on four key metrics: Accuracy, F1-Score, Precision, and Recall. As seen in the chart, K-Nearest Neighbors (KNN) achieved the best overall performance, closely followed by Random Forest and Gradient Boosting. These models performed consistently across all metrics, indicating strong generalization and balanced predictions. SVM also performed reasonably well but slightly lagged behind in recall and precision. Logistic Regression, on the other hand, showed the weakest results across all metrics, highlighting its limitations in capturing complex patterns in the dataset. Overall, the bar chart visually confirms that non-linear and ensemble models outperform simpler linear classifiers for multi-class disaster prediction.

4.1.2. Experiment 2: Severity Prediction using Regression

The above table presents the performance of five regression models used to predict the magnitude or severity of a disaster. The models were evaluated using four commonly used metrics: R² Score, RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), and Explained Variance. From the results, we can observe that Support Vector Regressor (SVR) delivered the best overall performance across all metrics. It achieved the highest R² Score of 0.2927, indicating that it could explain nearly 29 percent of the variance in the magnitude data. It also recorded the lowest RMSE (0.7213) and MAE (0.2874), which means the predicted magnitudes were on average very close to the actual values. Its Explained Variance score, also 0.2927, further confirms that SVR was most effective in capturing the variability of disaster severity.

Table 5. Regression Performance of Different Models

Models	**R^2 Score**	**RMSE**	**MAE**	**Explained Variance**
Random Forest	0.2120	0.7327	0.3122	0.2292
SVR	**0.2927**	**0.7213**	**0.2874**	**0.2927**
Ridge Regression	0.0759	0.8657	0.3499	0.0759
Lasso Regression	0.0622	0.8854	0.3570	0.0622
XGBoost	0.2838	0.7271	0.2904	0.2838

The XGBoost Regressor followed closely, with an R^2 Score of 0.2838, RMSE of 0.7271, and MAE of 0.2904, making it a very competitive option. Although slightly behind SVR, it performed consistently well and is known for handling complex patterns, missing values, and noisy data effectively.

The Random Forest Regressor performed moderately, with an R^2 Score of 0.2120, RMSE of 0.7327, and MAE of 0.3122. While not as strong as SVR or XGBoost, it still demonstrated a decent ability to generalize.

In contrast, Ridge and Lasso Regression, both linear models, showed significantly lower performance. Their R^2 Scores were 0.0759 and 0.0622 respectively, and they had the highest RMSE and MAE among all models. This suggests that linear models struggled to capture the non-linear relationships between location and disaster severity.

Overall, the results indicate that non-linear and ensemble-based models, especially SVR and XGBoost, are better suited for predicting disaster severity. Their ability to model complex interactions in the data makes them more reliable for this regression task compared to simpler linear methods.

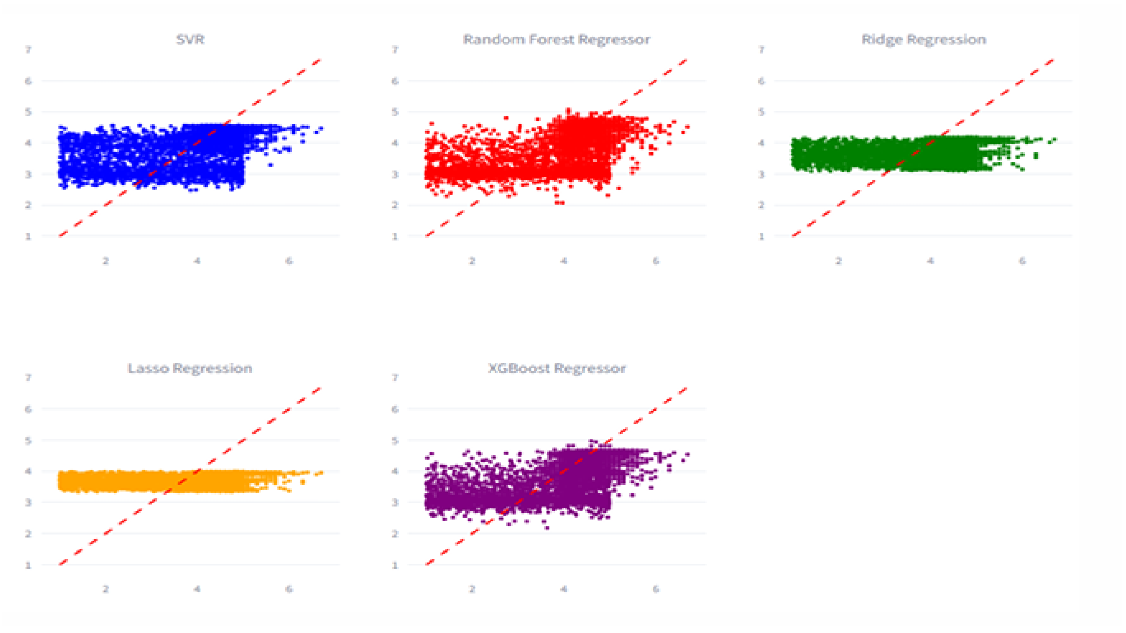

Figure 2. Scatter plots of predicted vs actual values for different regression models. The red dashed line represents ideal predictions.

This scatter plot presents the comparing the predicted magnitudes versus the actual values for each regression model. The red dashed line represents the ideal scenario where predicted values exactly match the actual ones (perfect prediction). Among the five models, Support Vector Regressor (SVR) and XGBoost Regressor show the best alignment with this reference line, indicating higher predictive accuracy. The SVR plot shows a compact cluster of points near the diagonal, reflecting consistent predictions close to true values. XGBoost also follows the trend line well, though with slightly more spread.

In contrast, Ridge and Lasso Regression exhibit flat and narrow bands of predictions, suggesting that they failed to capture the variance in the data, resulting in underfitting. The Random Forest Regressor produces a broader spread but still manages to follow the general trend. These visual patterns align with the numeric results shown earlier, reinforcing the conclusion that non-linear and ensemble models outperform simple linear ones for severity prediction in this context.

4.2. Discussion

This project aimed to evaluate multiple machine learning models on the basis of two main tasks:**disaster type classification and severity prediction**.The results clearly indicates that non-linear models and ensemble techniques which perform better than traditional linear classifiers and regressors in both scenarios.

Classification Task: Among the five models tested,**K-Nearest Neighbors (KNN)** showed the highest performance,slightly outperforming **Random Forest** and **Gradient Boosting** classifiers.**KNN** achieved an accuracy of 89.07%.This suggests the **distance based** and **ensemble models** are well suited for capturing the complex spatial patterns of natural disaster data (latitude,longitude and magnitude).

The **Support Vector Machine (SVM)** and **Logistic Regression** models lagged behind.They showed relatively lower performance(SVM accuracy: 81.70%,Logistic Regression:65.53%)reflects the limitations of **linear boundaries** in classifying real world,high variance data. This emphasizes the importance of using models capable of **non-linear decision-making** in spatial classification problems. **Regression Task**: In the second experiment which focused on predicting disaster severity,**Support Vector Regressor (SVR)** and **XGBoost** outperformed other models across all four evaluation metrics.**SVR** recorded the best R^2 (0.2927) suggesting it captured approximately 29% of the variance in disaster magnitude,the highest among all models. While **Random Forest REgressor** showed moderate performance,**Ridge** and **Lasso Regression** underperformed significantly. These linear models were unable to capture the non-linear dependencies between input features.Their low R^2 scores and high error values show that these models were not able to learn patterns in the data well,means they couldn't make reliable predictions when faced with new or unseen examples.

5. Conclusion

In this paper, we proposed a machine learning based approach for predicting natural disasters which happened in a location, focusing on the type and severity of the disaster. Our system takes latitude and longitude as input from the user and makes a combination of classification and regression models to predict the outputs. For ensuring more reliability we have done a comprehensive preprocessing pipeline which incldudes cleaning the data, outlier removal, class balancing and feature scaling. We used various machine learning models ranging from simple models like logistic regression to complex one like XGBoost. GridSearchCV was done to optimize the hyper parameters which ensured the model was fine tuned for best performance. The classification results showed that KNN performed best with 89 percentage and in the regression task SVR and XGBoost gave the most accurate estimate, which out performed linear models. We also visualized the regression outputs through scatter plot, which shown the alignment of predictions with the actual values. This also confirmed that SVR and XGBoost have done the best. To make our system practical and accessible, we have developed a stream lit based interface which helps the users to input the latitude and longitude and it outputs the predictions with visualization. In future, we would aim to improve the system by integrating real time featues like weather data, satellite images etc. Overall this study lays the ground work for an effective disaster prediction system that can decision makers in disaster impacts.