









Seismic Intelligence: Machine Learning Models for Earthquake Magnitude Estimation

Kanchon Kumar Bishnu¹, Biswajit Chandra Das², Md. Shafikul Islam³, Md. Shafiul Alam Chowdhury^{3*},
Shoma Islam⁴, Md Azam Khan⁴

¹ Department of Computer Science, California State University Los Angeles, Los Angeles 90032, CA, USA

² Department of Computer Science, Los Angeles City College, Los Angeles 90029, CA, USA

³ Department of Computer Science and Engineering, Uttara University, Dhaka 1230, Bangladesh

⁴ Management Information System Department, School of Business, International American University, Los Angeles 90012, CA, USA

Corresponding Author Email: shafiul.a.chowdhury@gmail.com

Copyright: ©2025 The authors. This article is published by IETA and is licensed under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://doi.org/10.18280/mmep.121011>

ABSTRACT

Received: 27 August 2025

Revised: 5 October 2025

Accepted: 11 October 2025

Available online: 31 October 2025

Keywords:

machine learning, ridge, lasso, support vector machine, random forest, gradient boosting, XGBoost, LightGBM, CatBoost, ElasticNet, ANNs

Earthquakes pose serious risks to human life, infrastructure, and economic stability across the globe. This experiment explores the use of machine learning (ML) to improve the accuracy of earthquake magnitude prediction. A dataset from the U.S. Geological Survey (USGS), containing over 8,000 records with attributes such as time, location, depth, and magnitude type, was used for model development. The data underwent preprocessing to handle missing values, remove anomalies, and eliminate irrelevant features. Feature engineering techniques, including one-hot encoding and normalization, were applied to enhance model performance. Several regression models were evaluated, including Ridge, Lasso, SVR, Random Forest, Gradient Boosting, XGBoost, LightGBM, CatBoost, and ElasticNet. Among these, a stacked ensemble model combining CatBoost, LightGBM, and Random Forest delivered the most accurate results. It achieved a mean squared error (MSE) of 0.0976, a coefficient of determination (R^2) of 0.9392, and a normalized root mean squared error (NRMSE) of 4.6%. The findings demonstrate that ensemble learning significantly improves earthquake magnitude estimation. This approach offers valuable insights for seismic risk assessment and supports the development of more effective disaster preparedness strategies.

1. INTRODUCTION

Earthquakes are among the most destructive natural hazards, posing serious risks to lives, infrastructure, and economies worldwide. Accurately predicting earthquake magnitude remains a complex challenge due to the nonlinear and multidimensional nature of seismic phenomena. While traditional empirical and deterministic models offer foundational insights, they often struggle to capture intricate patterns within seismic data.

This study aims to enhance earthquake magnitude prediction by applying machine learning (ML) techniques to a Kaggle-sourced dataset containing over 8,000 records with attributes such as time, location, depth, and magnitude type. The primary objective is to evaluate and compare multiple regression models, with a focus on developing a robust stacked ensemble that integrates CatBoost, LightGBM, and Random Forest to improve predictive accuracy.

The methodology includes data preprocessing to address missing values and anomalies, feature engineering to optimize input variables, and exploratory data analysis (EDA) to uncover relevant patterns. Model performance is assessed using metrics such as mean squared error (MSE), root mean

squared error (RMSE), and coefficient of determination (R^2).

By combining geophysical data with advanced ML techniques, this research contributes to more reliable seismic hazard assessment and supports the development of data-driven strategies for disaster preparedness.

2. LITERATURE REVIEW

In the latest literature, different ML methods for earthquake magnitude prediction have been investigated and shows their own strengths and weaknesses. A remarkable work used Random Forest models on LA, Istanbul, and San Diego seismic data with high sensitivity rates of 91.65%, 98.53% and 97.97%. Of 16 tested ML and neural network models, Random Forest was the most successful. Nevertheless, as a few drawbacks still exist, it is hard to estimate the exact time and location of an earthquake and to apply in early warning systems [1].

In a different study, Artificial Neural Networks (ANNs) with Levenberg-Marquardt Backpropagation were applied for short-term forecasting (between 1–15 days) at regions covering Japan, Turkey, Greece and Indian Subcontinent. The

model showed flexibility and low false alarm rates with the accuracy ranging from 78.36 to 87.65%. Nevertheless, it still relied on the past and its prediction power was not accurate in time [2].

Advanced models including Long Short-Term Memory (LSTM), Support Vector Machines (SVMs), Gradient Boosting, and hybrid stacking have also been used for prediction of magnitude and depth. These methods effectively capture temporal and geological characteristics, which help in preventing overfitting. However, they have the same limitations: computationally expensive and lack generalization to different seismic zones [3].

A comparative study assessed the performance between Random Forest Regressors and Neural Networks with heterogeneous seismic features. Whereas Random Forest performed well only with a variety of data, and Neural Network could recognize intricate trends but it has limitations on overfitting and in terms of computational time [4]. More recently Inverse Boosting Pruning Trees (IBPT) based on satellite infrared and hyperspectral to detect pre-earthquake anomalies were proposed. After evaluation with 1,371 events, IBPT surpassed six benchmarks (including Random Forest and SVM), and was promising for its short-term prediction. Nevertheless, the reliance on satellite information may hinder its scale and usage in resource-deprived areas [5].

In line with these works, Ding and Hu [6] showed how deep learning and computer vision could be employed in

monitoring geological hazards, thus indicating that ML techniques can easily be adapted to the detection of hazard. JayaLakshmi et al. [7] employed transformer-based multimodal tweet classification for crisis management, highlighting the promise of ML in real-time events analysis. Ojeda et al. [8] improved seismic vulnerability assessment of confined brick masonry houses with the use of assemblage algorithms, cementing ML’s place within structural risk analysis. Also, Xing [9] developed a real-time monitoring and early warning system based on image analysis that is equivalent to the earthquake early warning application. Shantaf and Kutucu [10] also solved the issue of seismic sensor reliability by the use of ML in detecting signal distortion, Putra and Sunarno [11] presented a Monte Carlo aggregating technique to radon precursor-based earthquake prediction in Java, Indonesia.

Although these studies as shown in Table 1 highlight the promise of ML in seismic prediction, they mostly concentrate on single-model approaches with limitations either in terms of generalization, timing precision or data availability. This study fills these gaps by conducting a stacked ensemble method with CatBoost, LightGBM and Random Forest. By leveraging strengths of various models and selecting feature engineering systematically, the proposed method is expected to enhance accuracy and robustness of magnitude prediction for making better decision of seismic risk assessment and disaster preparedness.

Table 1. Comparison table (prior studies)

Study	Method	Accuracy	Limitations
[1]	Random Forest	91.65%–98.53%	Relies on historical data; lacks timing/location prediction
[2]	ANN (Levenberg-Marquardt)	78.36%–87.65%	Limited temporal precision; historical dependency
[3]	LSTM, SVM, Gradient Boosting, Hybrid Stacking	High (varied)	Computational cost; generalizability issues
[4]	Random Forest vs. Neural Networks	High (varied)	Overfitting; resource-intensive
[5]	IBPT	Outperformed 6 models	Satellite data dependency; limited scalability
[6]	Deep Learning + Computer Vision (Geological Disaster Monitoring)	High detection accuracy	Focused on geological hazards; indirect application to seismic magnitude
[7]	Transformer-based Multimodal Tweet Classification	Effective in disaster event detection	Dependent on social media data; not seismic-specific
[8]	Assembly Algorithms for Seismic Vulnerability Estimation	Reliable structural vulnerability assessment	Limited to confined masonry dwellings
[9]	Image-based Early Warning System	Real-time hazard monitoring	Applied to engineering safety hazards; adaptation needed for seismic data
[10]	ML for Seismic Sensor Security	Improved signal distortion detection	Focused on sensor reliability; indirect to magnitude prediction
[11]	Monte Carlo Aggregating Method (Radon Precursors)	Effective in parameter prediction	Requires radon data; limited scalability

3. DATASET

The dataset used in this study was published on Kaggle by Rahman [6] and is sourced from the U.S. Geological Survey (USGS). It contains 8,017 records of global seismic events that occurred between November 4 and December 4, 2024 (Figure 1). Each record includes detailed attributes suitable for statistical, geographical, and machine learning analyses.

Key features relevant to this study include:

- **Magnitude (mag):** Represents the energy released during the earthquake.
- **Depth (km):** Indicates the depth of the epicenter beneath the Earth's surface.
- **Magnitude Type (magType):** Specifies the scale

used to measure magnitude (e.g., Mw, Mb, MI).

- **Time and Location:** Timestamp and geographic coordinates of each event.
- **Number of Seismic Stations (nst):** Reflects the number of stations that recorded the event, influencing data accuracy.

Additional features such as error margins, event type, and network identifiers are included in the dataset mentioned in the Table 2.

A full description of all 22 features is provided in the Appendix I. This dataset offers a rich foundation for training and evaluating machine learning models aimed at improving earthquake magnitude prediction.

```

Df.info( )
< class 'pandass.core.frame.DataFrame'>
RangeIndex: 7576 entries, 0 to 7575
Data columns (total 22 columns):
#      Column      Non-Null Count  Dtype
---  -
0      time          7576 non-null   object
1      latitude      7576 non-null   float64
2      longitude     7576 non-null   float64
3      depth         7576 non-null   float64
4      mag           7574 non-null   float64
5      magType       7574 non-null   object
6      nst           6489 non-null   float64
7      gap           6481 non-null   float64
8      dmin          7574 non-null   float64
9      rms           7576 non-null   float64
10     net           7576 non-null   object
11     Id            7576 non-null   object
12     updated       7576 non-null   object
13     place         7576 non-null   object
14     type          6004 non-null   object
15     horizontalError 7574 non-null   float64
16     depthError    6467 non-null   float64
17     magError      6489 non-null   float64
18     magNst        7576 non-null   float64
19     status        7576 non-null   object
20     locationSource 7576 non-null   object
21     magSource     7576 non-null   object
dtypes: float64(12), object(10)
Memory usage: 1.3+ MB

```

Figure 1. Dataset from Kaggle.org

Table 2. Feature description of a few datasets (summarize)

Feature Name	Description
Latitude and Longitude	Epicenter coordinates in degrees
Depth	Epicenter depth in kilometers
Magnitude	Earthquake strength
Magnitude Type	Measurement scale used
Time	Timestamp of occurrence
Number of Stations (nst)	Count of stations that recorded the event
Azimuthal Gap	Angular gap between stations
RMS	Seismic wave intensity across stations
Event Type	Classification (e.g., earthquake, explosion)
Error Metrics	Horizontal, depth, and magnitude uncertainties
Network and Source Info	Reporting and magnitude estimation sources

4. DOMAIN KNOWLEDGE

The magnitude of an earthquake is the measurement of energy release in an earthquake, and it is measured using a scale that can be correlated to other earthquakes.

4.1 Domain earthquake magnitude scales

The magnitude of an earthquake is a measure of the energy released during a seismic event and is an important parameter for predictive modeling. Mag Type (magType) is of particular interest for machine learning applications as source scales measure seismic energy differently, affecting model performance and transferability.

For larger earthquakes, the Moment Magnitude Scale (Mw)

is commonly used due to its uniformity across fault types and depths [12, 13]. On the other hand, Local Magnitude (ML) or Body Wave Magnitude (Mb) may be used for smaller or shallow earthquakes and their use could add to statistical variability associated with regional calibration and station distance [14, 15]. Despite the fact that the Richter Scale was designed to be used for magnitude, these are usually determined using Mw for present seismic analysis [16]. In the context of machine learning, considering magType as a categorical feature is one way to allow models to control for such methodological differences. It is important to at least take it into account, because if not one might obtain biased predictions, especially when the data contain mixed-scale measures. Careful encoding and normalization of magType is thus crucial to obtain accurate magnitudes.

4.2 Earthquake magnitude ranges

Earthquake Mahufacture sizes can vary enormously and are usually ranged into categories depending upon their size, the amount of energy released, and the potential for damage. The classification of seismic magnitudes with the resulting consequences are presented in Table 3 [17-19].

Table 3. Earthquake magnitude classification

Magnitude Range	Classification	Characteristics
< 2.0	Micro Earthquakes	Typically undetectable without instruments; no impact on people or infrastructure.
2.0–3.0	Very Minor	Rarely felt; no damage; may resemble distant rumbling.
3.0–4.0	Minor	Felt in populated areas; unlikely to cause damage.
4.0–5.0	Light	Noticeable shaking; may cause minor damage to weak structures.
5.0–6.0	Moderate	Can cause minor to moderate damage; may disrupt daily life.
6.0–7.0	Strong	Capable of severe structural damage and triggering landslides.
7.0–8.0	Major	Catastrophic potential; can devastate cities and cause secondary disasters.
> 8.0	Great	Extremely rare; global-scale destruction possible.

5. METHODOLOGY

This section outlines the strategic approach used to process and analyze seismic data using ML techniques. The workflow includes (Figure 2) data preparation, model development, and performance evaluation, designed to ensure reliable and accurate earthquake magnitude prediction [20-22].

5.1 Data preparation

The dataset was first cleaned and organized to ensure consistency and usability. Pre-processing steps include handling missing values, removing outliers, and transforming features. Numerical variables were normalized, and categorical variables such as magnitude type (magType) were

one-hot encoded to improve model compatibility. Feature selection and reduction techniques were applied to minimize redundancy and enhance model efficiency.

5.2 Supervised learning and regression models

Supervised learning was employed to train models on labeled data, where each input corresponds to a known output. The focus was on regression tasks, which predict continuous values such as earthquake magnitude. Algorithms evaluated include linear regression, Random Forest, Gradient Boosting, CatBoost, and other ensemble methods [23, 24]. These models were chosen for their ability to capture both linear and nonlinear relationships and to generalize across diverse seismic conditions.

5.3 Feature correlation analysis

To reduce multicollinearity and improve model interpretability, a Pearson correlation matrix was computed. Highly correlated features were identified and visualized using a heatmap, aiding in feature selection and simplification [25, 26].

5.4 Model evaluation

Model performance was assessed using standard regression metrics: MSE, RMSE, NRMSE, and R² score [27]. These metrics quantify prediction accuracy and model fit. Detailed formulas and interpretations are provided in the Table 4.

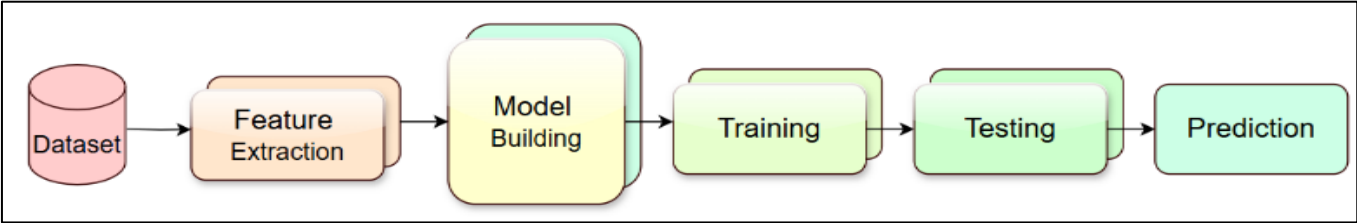


Figure 2. Procedure of experiment

Table 4. Evaluation metric formulas and interpretations [27]

Metric	Formula	Description
MSE	$MSE = \sum (y_i - \hat{y}_i)^2 / n$	Measures the average squared difference between actual and predicted values. Penalizes larger errors more heavily. Lower MSE indicates better model fit.
RMSE	$\sqrt{\sum (y_i - \hat{y}_i)^2 / n}$	Square root of MSE, expressed in the same units as the target variable. Easier to interpret and useful for understanding average error magnitude.
NRMSE	$NRMSE = RMSE / \text{Normalization Factor}$	Adjusts RMSE for the scale of the target variable. Useful for comparing model performance across datasets with different ranges or units.
R ² Score	$R^2 = 1 - SS_{res} / SS_{tot}$	Indicates how well the model explains variance in the target variable. Ranges from 1 (perfect fit) to negative values (poor fit). Higher R ² suggests stronger predictive power.

6. IMPLEMENTATION

Figure 3 illustrates the implementation workflow for the earthquake-magnitude prediction project. Key stages comprise data acquisition, pre-processing, transformation, model testing, training, evaluation, stacking/ensemble learning, final deployment, prediction, and feedback.

6.1 Tools and libraries

The project was developed in python 3.12 and run in a Jupyter-style environment on Google Colab, leveraging libraries for data preprocessing, model development, evaluation, and visualization.

6.2 Key libraries

Table 5 is about Key Libraries used in the study.

6.3 EDA

Through EDA, we examined the dataset’s statistical properties, visualized feature interactions, and identified correlations and anomalies. This process revealed patterns that were crucial for feature selection and model design, ultimately

shaping a more accurate and reliable predictive framework.

6.3.1 Distribution of earthquake magnitudes

Figure 4 illustrates the distribution of earthquake magnitudes. The x-axis denotes magnitude, and the y-axis denotes frequency. The predominance of low-magnitude events (approximately 1–2) is evident from the tall bars on the left, while the overlaid curve represents a smoothed trend of the distribution.

6.3.2 Distribution of earthquake depths

Figure 5 illustrates the distribution of earthquake depths. The x-axis denotes depth (km), and the y-axis denotes frequency. A predominance of shallow events (near-surface) is evident from the tall bar near zero, while the overlaid curve indicates that deeper earthquakes are relatively rare.

6.3.3 Earthquake locations by magnitude with global outline

Figure 6 presents a global distribution of earthquake epicenters, indicated by dots. Dot color corresponds to earthquake magnitude, as defined in the adjacent color bar (yellow = high magnitude; dark purple = low magnitude). Longitude and latitude are shown on the x- and y-axes, respectively. Earthquake activity is concentrated along tectonic plate boundaries, with notable clustering in the pacific “ring of fire” region.

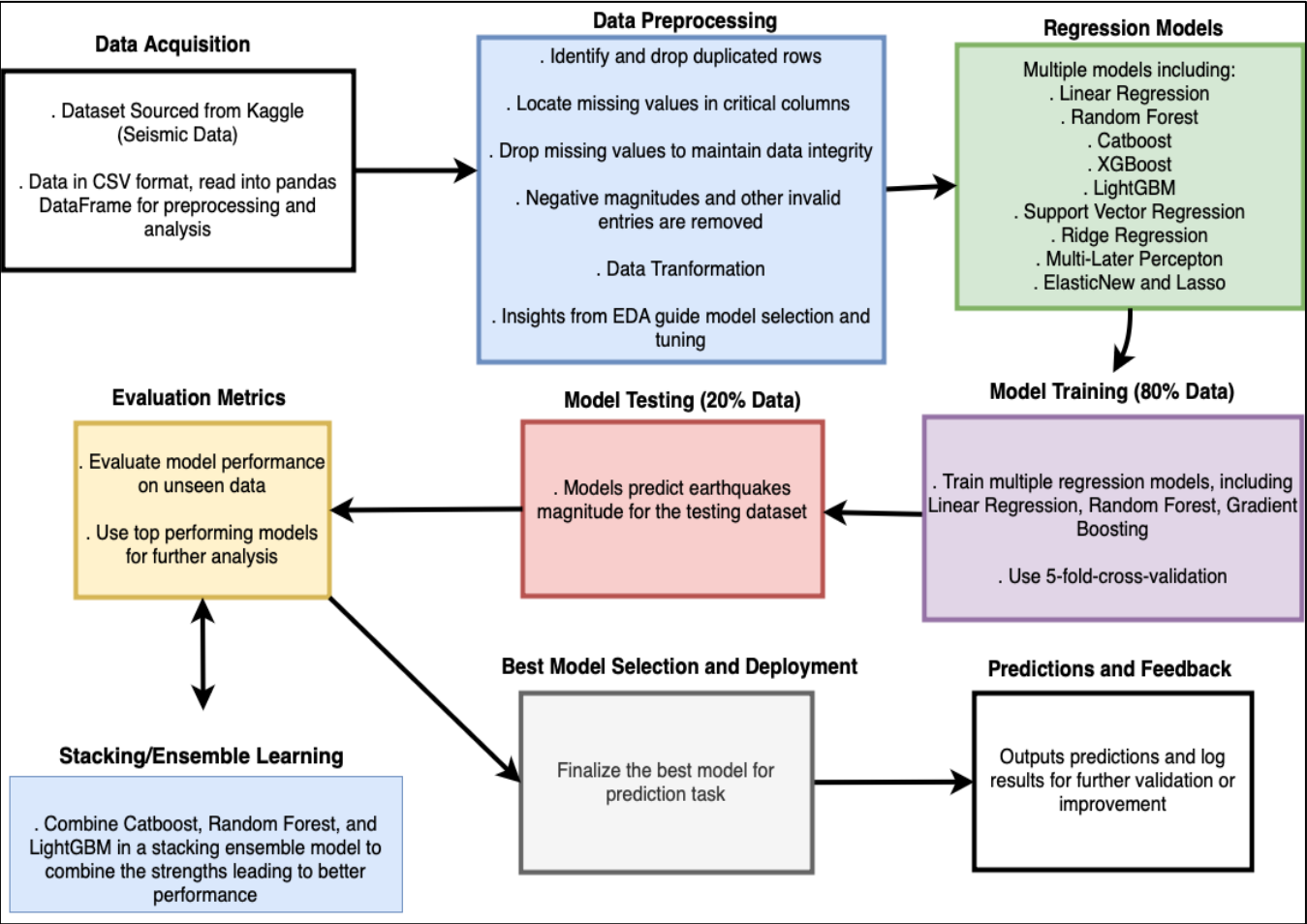


Figure 3. Implementation diagram

Table 5. Key libraries

Library	Purpose
pandas	Employed for data manipulation, preprocessing, and exploration.
numpy	Applied for numerical computations and array manipulations.
scikit-learn	Utilized for preprocessing (e.g., standardization), regression models, and evaluation metrics (e.g., MSE).
matplotlib	Used to create static visualizations, such as histograms and scatter plots.
seaborn	Used to create enhanced visualizations for better insights.
xgboost	Implemented for advanced gradient boosting regression tasks.
lightgbm	Used for efficient and scalable gradient boosting on large datasets.
catboost	Optimized for handling categorical features in regression models.
cartopy	Used for geospatial data visualization, primarily focused on creating maps and plotting spatial data.

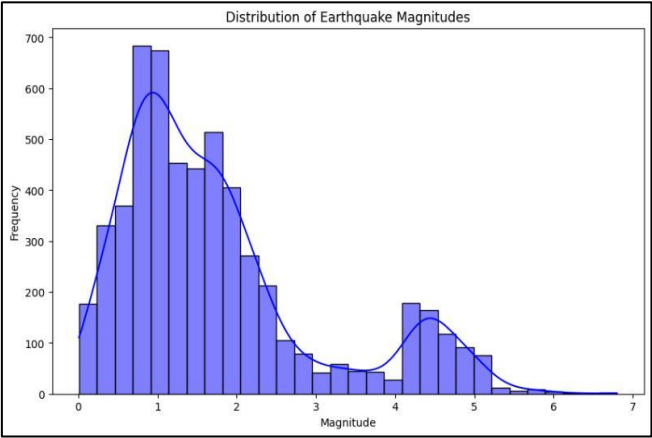


Figure 4. Distribution of earthquake magnitudes

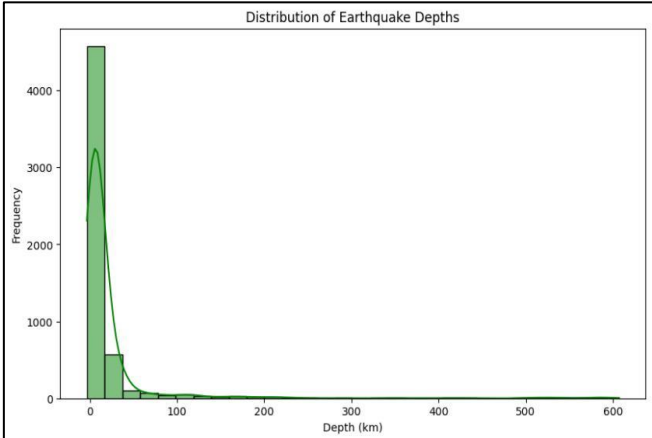


Figure 5. Distribution of earthquake depths

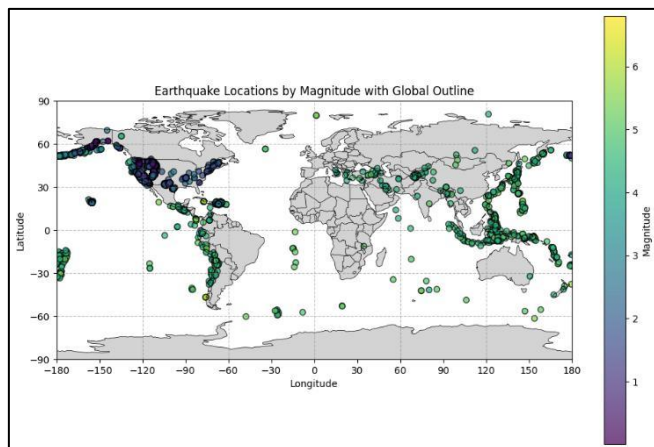


Figure 6. Earthquake locations by magnitude with global outline

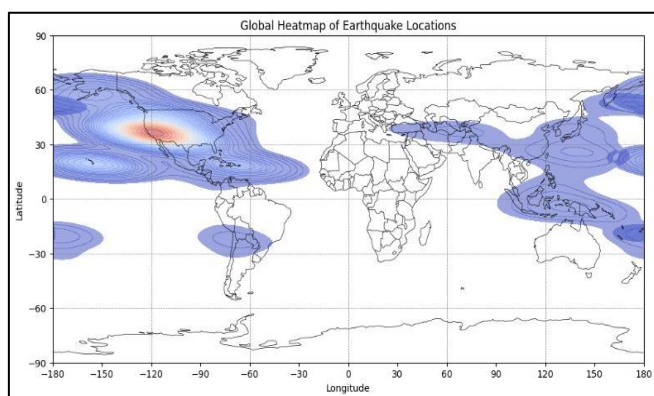


Figure 7. Global heatmap of earthquake locations

6.4 Global heatmap of earthquake locations

Figure 7 presents a global heatmap showcasing earthquake frequency, where dark red indicates high-density zones and light blue marks low-density areas. The map plots longitude and latitude along the x- and y-axes, respectively. Notably, seismic activity aligns closely with tectonic plate boundaries, with prominent clusters observed in the Pacific *Ring of Fire*, as well as regions near South America, Japan, and Indonesia.

6.4.1 Magnitude type vs. depth

Figure 8 illustrates the relationship between earthquake magnitude and depth in a scatter plot. Depth (km) is shown along the x-axis and magnitude along the y-axis. Colors represent different magnitude measurement types, as defined in the legend. Shallow earthquakes exhibit a broad magnitude range, while deep-focus events are comparatively rare and generally exceed magnitude 4.

6.4.2 Distribution of magnitude types

Figure 9 displays the distribution of different earthquake magnitude measurement types. The x-axis lists magnitude types (e.g., ml, md, mb), and the y-axis indicates their count or frequency. The most common types are ml and md, with ml occurring most frequently. Other types such as mb, Mww, Mwr, and mB_LG are comparatively rare.

6.4.3 Magnitude vs. magnitude error

Figure 10 displays a scatter plot illustrating the relationship between earthquake magnitude and measurement error. The x-

axis represents magnitude, while the y-axis shows the corresponding error. Lower-magnitude earthquakes (near zero) exhibit greater variability in error, whereas higher magnitudes (above 4) show smaller, more consistent errors indicating improved measurement accuracy with increasing magnitude.

6.4.4 Earthquake distribution by state (with other places)

Figure 11 shows that California leads in earthquake occurrences (42.8%), followed by Hawaii (9.9%) and Alaska (7.0%). Less activity is noted in other states, with “Other Places” making up 17.4%. The data highlights a strong concentration of seismic events in specific regions, especially California and Hawaii.

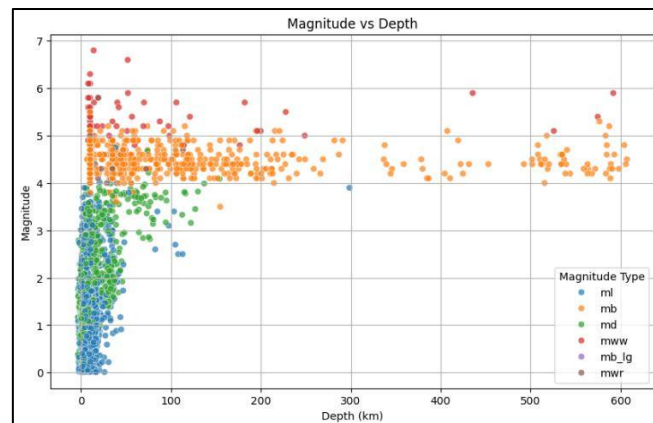


Figure 8. Magnitude type vs. depth

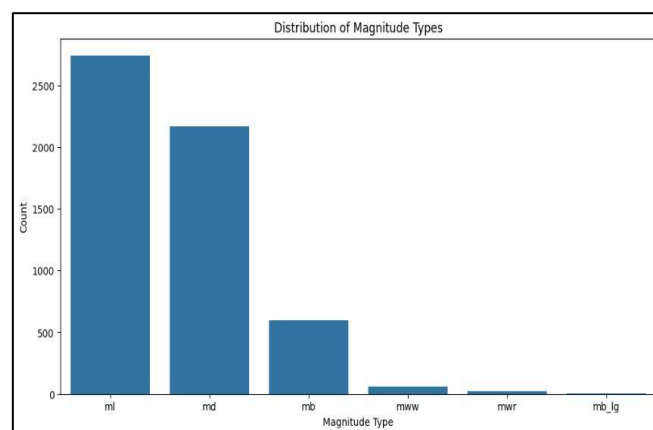


Figure 9. Distribution of magnitude types

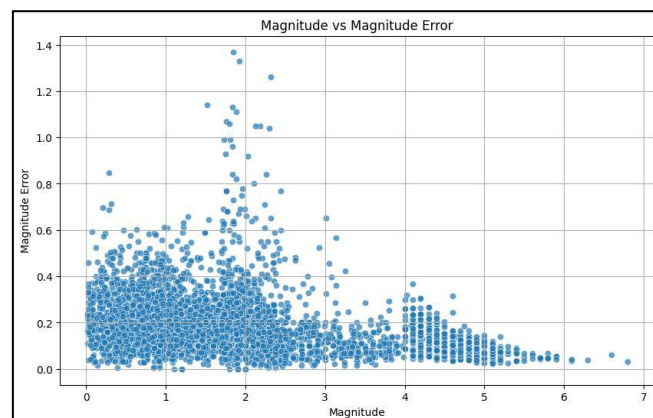


Figure 10. Magnitude vs. magnitude error

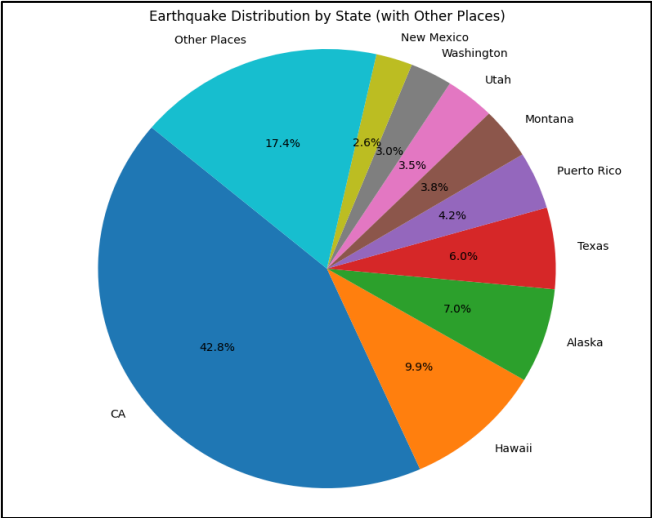


Figure 11. Earthquake distribution by state (with other places)

6.5 Data preprocessing

To ensure the dataset's relevance, integrity, and suitability for machine learning modeling, a comprehensive data pre-

processing pipeline was implemented. Initially, data filtering was performed by retaining only rows labeled as 'earthquake' in the type column, excluding other seismic events such as quarry blasts, explosions, and sonic booms. Missing data were addressed by identifying columns with NaN values specifically nst, gap, dmin, horizontalError, magError, and magNst - and removing rows with incomplete entries, reducing the dataset from 7,884 to 6,146 records. Duplicate rows were checked using the df.duplicated() function, confirming no redundancies. Redundant columns such as net and location source were dropped to streamline the dataset. Outlier removal was conducted by excluding records with negative earthquake magnitudes, which lack physical meaning in standard seismic interpretation.

For categorical transformation, the mag_type feature was one-hot encoded into binary columns representing each magnitude type (e.g., magType_mb, magType_ml), enabling effective model input. Finally, feature standardization was applied using Z-score normalization via StandardScaler, ensuring consistent feature scaling across training and test sets to enhance model convergence and performance.

6.5.1 Algorithmic breakdown and discussion

Table 6 provides a detailed overview of the algorithmic breakdown along with relevant discussions.

Table 6. Algorithmic breakdown and discussion

Step	Process	Algorithm	Discussion
i	Data Filtering	df = df[df['type'] == 'earthquake']	Retains only earthquake events, excluding irrelevant seismic types to focus the analysis.
ii	Missing Data Handling	df = df.dropna(subset=['nst', 'gap', 'dmin', 'horizontalError', 'magError', 'magNst'])	Removes rows with missing values in critical columns to maintain data integrity and prevent bias.
iii	Duplicate Rows Removal	df = df[~df.duplicated()]	Ensures each record is unique, avoiding redundancy that could distort model training.
iv	Drop Redundant Columns	df = df.drop(columns=['net', 'locationsource'])	Eliminates non-essential metadata to streamline the dataset and reduce dimensionality.
v	Outlier Removal	df = df[df['mag'] > 0]	Filters out invalid negative magnitudes, ensuring meaningful seismic data representation.
vi	One-hot Encoding	df = pd.get_dummies(df, columns=['magType'], prefix='magType')	Converts categorical magnitude types into binary features for compatibility with ML models.
vii	Standardization (Z-Score)	scaler = StandardScaler() scaler.fit_transform(X_train) X_train_scaled = scaler.fit_transform(X_train) X_test_scaled = scaler.transform(X_test)	Normalizes feature scales to improve model convergence and performance across datasets.

6.6 Correlation analysis

Pearson's correlation coefficient measures the strength and direction of the linear relationship between two continuous variables such as the mag feature and other numerical attributes. Its value ranges from -1 (indicating a perfect negative correlation) to +1 (indicating a perfect positive correlation), with 0 representing no linear association. The coefficient is calculated using the following formula:

$$r = \frac{\sum((x_i - \text{mean}(x)) * (y_i - \text{mean}(y)))}{\sqrt{(\sum(x_i - \text{mean}(x))^2 * \sum(y_i - \text{mean}(y))^2)}}$$

where,

x_i, y_i: individual data points for the two variables respectively.

x, y: Two variables of dataset.

R: Pearson correlation coefficient.

The resulting matrix has the correlation coefficients between all pairs of numerical features.

The correlation matrix (Figure 12) provides insight into

which variables are strongly associated with earthquake magnitude and which exhibit minimal or negative correlations, potentially indicating limited or no predictive value with respect to the target variable. The accompanying heatmap visualizes these relationships among the dataset's numerical features. Strong positive or negative correlations are depicted in dark red or dark blue, respectively, while weaker correlations appear in lighter shades.

Values near 1 indicate strong positive correlations, while those near -1 indicate strong negative correlations. The diagonal represents perfect correlations, as each variable is correlated with itself. This visualization enables rapid assessment of feature relationships, supporting effective feature selection for modeling.

6.7 Standardization and scaling

Feature scaling was performed to ensure uniformity and prevent dominance of features with larger numerical ranges. Table 7 represents the overview of regression methods for earthquake magnitude prediction.

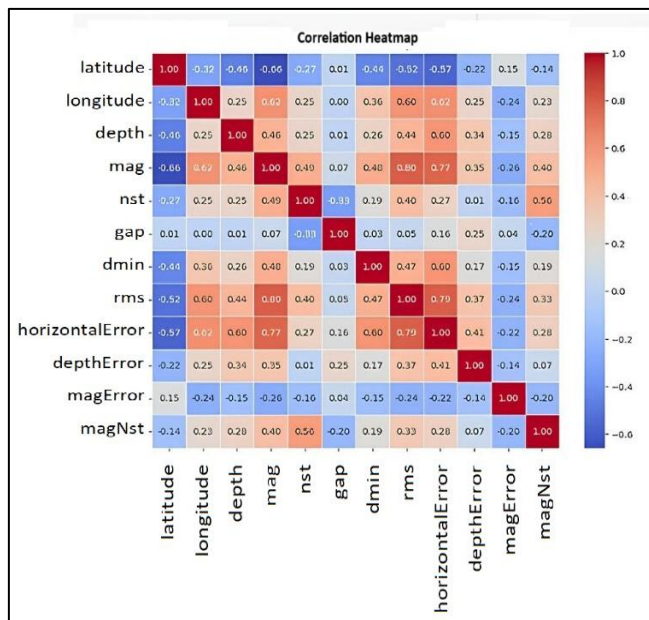


Figure 12. Heatmap image (confusion matrix)

6.8 Cross validation

The study used 5-fold cross-validation to assess model generalizability by splitting data into five parts, training on four, and validating on one ensuring all data points were used. Preprocessing steps like standardization were applied within each fold via pipelines to prevent data leakage. This method provided reliable performance estimates, reduced overfitting, and enabled consistent evaluation using metrics such as MSE, RMSE, NRMSE, and R^2 .

Table 7. Overview of regression methods for earthquake magnitude prediction

Model	Description
Random Forest Regressor	Constructs multiple decision trees on random subsets of data and averages predictions to reduce overfitting and capture non-linear relationships.
XGBoost	Implements gradient boosting with level-wise tree growth, L1/L2 regularization, and efficient handling of missing data.
LightGBM	Uses leaf-wise gradient boosting and histogram-based learning for fast and scalable performance on large or sparse datasets.
CatBoost	Optimized for categorical features with native encoding and ordered boosting, reducing overfitting and improving accuracy.
Support Vector Regression (SVR)	Applies kernel functions (e.g., RBF) to fit a hyperplane in higher-dimensional space, minimizing prediction error within a defined margin.
Ridge Regression	A linear model with L2 regularization that mitigates multicollinearity and helps prevent overfitting.
Multi-Layer Perceptron (MLP)	A feedforward neural network with hidden layers and non-linear activations, capable of modeling complex patterns.
ElasticNet and Lasso	ElasticNet blends L1 and L2 penalties for balanced regularization; Lasso (L1) performs feature selection by shrinking some coefficients to zero.
Bayesian Ridge Regression	A probabilistic linear regression model that incorporates prior distributions over parameters, yielding predictions with uncertainty estimates.

Table 8. Rationale for selected regression models

Model	Selection Rationale
CatBoost	Chosen for its strong handling of categorical features and robust individual predictive performance.
LightGBM	Selected for its speed, scalability, and efficient gradient boosting capabilities.
Random Forest Regressor	Included for its ability to model non-linear relationships and reduce overfitting through an ensemble of decision trees.
Meta-Regressor (Ridge Regression)	Used for prediction aggregation due to its simplicity, stability, and resistance to overfitting.

6.9 Training and testing

Models were trained on a normalized dataset using 80% of the data, with preprocessing ensuring consistency. A 5-fold cross-validation approach guided performance evaluation and hyperparameter tuning, emphasizing key metrics and ensemble model selection. The remaining 20% was reserved for testing, with prediction accuracy validated through error metrics and Predicted-vs-Actual plots.

The code defines `evaluate_model_cv`, which takes a model, training data (`x_train`, `y_train`), and a cross-validation parameter (default = 5). It evaluates models using two scorers MSE (lower is better) and r^2 (higher is better) via `cross_val_score`, returning their mean values. Models are wrapped in a pipeline for consistent application and potential preprocessing. A loop iterates through models, creates pipelines, calls the function, and stores results (MSE and r^2) in a dictionary, enabling systematic comparison of model performance on the same dataset.

6.10 Ensemble learning

Ensemble learning techniques were applied to combine the strengths of multiple models (Table 8), improving predictive accuracy and generalization. The stacking model implemented in the notebook incorporated three base learners and a meta-regressor to aggregate their predictions effectively.

This line creates a stacking regressor that combines three machine learning models CatBoost Regressor (with verbose output disabled), LightGBM Regressor, and Random Forest Regressor. In the stacking approach, each base model generates its own predictions, which are then combined by a meta-model to produce the final output.

7. RESULTS

7.1 Ensemble learning (stacking ensemble)

The stacking ensemble, combining CatBoost, LightGBM, and Random Forest, achieved the best overall performance, with an MSE of 0.097583, an r^2 of 0.939, and a NRMSE of 4.6%. These results, obtained via cross-validation, confirm the model’s generalization and robustness. By leveraging the complementary strengths of multiple advanced algorithms, the ensemble mitigates individual model limitations, delivering a more accurate, robust, and reliable prediction system.

7.2 Model performance summary

Table 9 presents a comprehensive summary of the performance metrics for various regression models applied to earthquake magnitude prediction. It highlights key indicators

such as MSE, R^2 Score, and NRMSE, offering insights into each model’s accuracy, generalization capability, and suitability for capturing complex seismic patterns.

7.3 Result analysis

The study achieved major improvements in earthquake magnitude prediction through careful data preprocessing and model optimization. Key steps included removing irrelevant or faulty data, applying one-hot encoding, and selecting meaningful features to boost accuracy and reduce overfitting. The most effective model was a stacked ensemble combining CatBoost, Random Forest Regressor, and LightGBM, which outperformed individual models. With five-fold cross-validation, the final model showed excellent performance metrics (e.g., $R^2 = 0.9587$), confirming its precision and reliability. Overall, the approach highlights the power of advanced machine learning in geoscientific prediction.

Table 9. Model performance summary

Model	MSE	R ² Score	NRMSE	Remarks
CatBoost	0.0998	0.937	4.65%	Strong generalization; close to ensemble model; suitable as standalone predictor.
LightGBM	0.1073	0.933	4.83%	Efficient and scalable; slightly below top performers.
Random Forest	0.1075	0.933	4.83%	Overfitting-resistant; performance nearly identical to LightGBM.
XGBoost	0.1130	0.929	4.95%	Robust and versatile; marginally behind top models.
Gradient Boosting	0.1289	0.919	5.29%	Accuracy decline suggests overfitting or weaker feature handling.
MLP	0.1472	0.909	5.65%	Captures non-linearities; underperforms compared to tree-based models.
SVR	0.1725	0.892	6.12%	Struggles with complex non-linear patterns.
Bayesian Ridge / Ridge	≈0.371	0.769	8.97%	Limited by linear assumptions.
ElasticNet	1.013	0.371	14.82%	Poor performance; unable to model complex interactions.
Lasso	--	−0.002	18.72%	Poorest performer; fails to capture meaningful patterns.

8. VISUAL ANALYSIS

The stacking ensemble model (Figure 13), integrating CatBoost, LightGBM, and Random Forest, delivered the most accurate predictions with the lowest MSE. In contrast, the standalone Lasso regression model showed the highest error,

reflecting its relatively limited predictive performance. The stacking ensemble (Figure 14) learning model (CatBoost, LightGBM, and Random Forest) achieved the highest R^2 value, reflecting superior predictive performance, while the Lasso regression model recorded the lowest R^2 value, highlighting its weaker predictive accuracy.

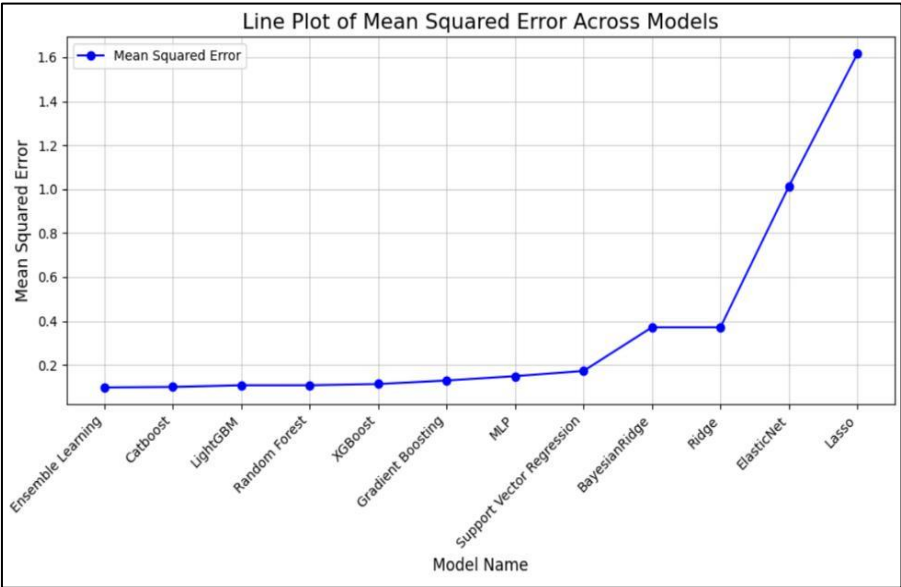


Figure 13. MSE access models

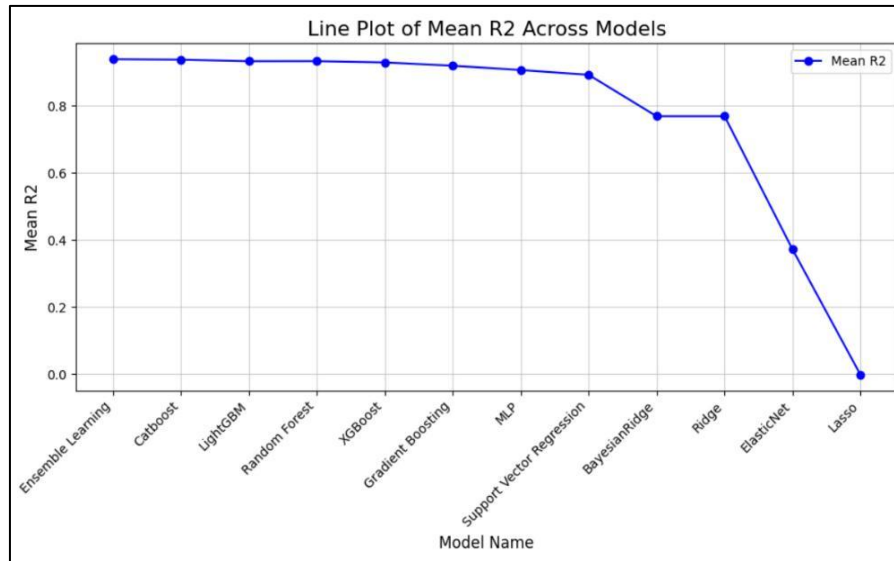


Figure 14. MSE access models

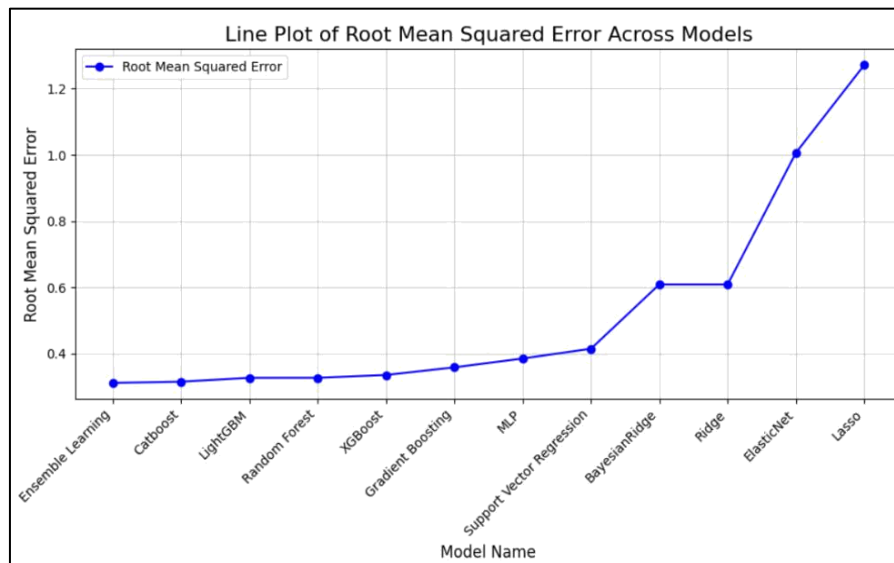


Figure 15. RSME models

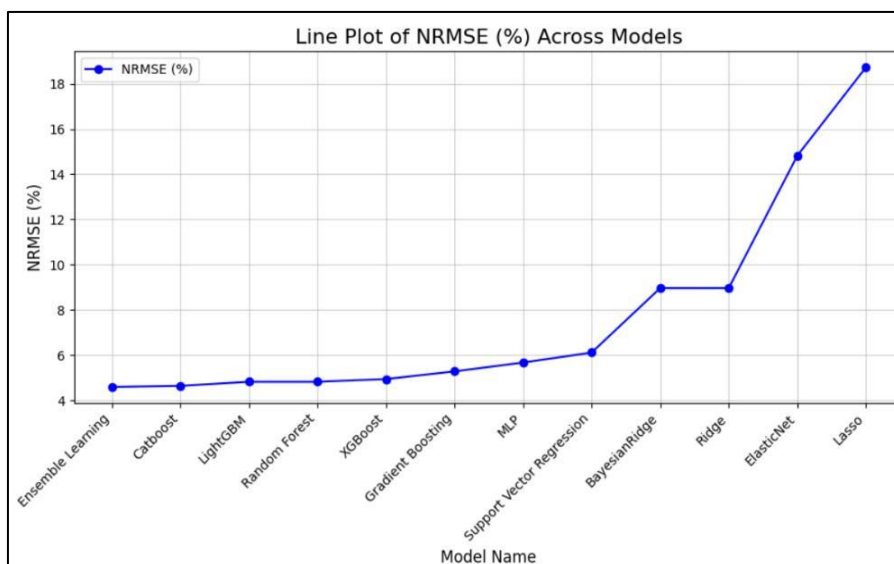


Figure 16. NRMSE models

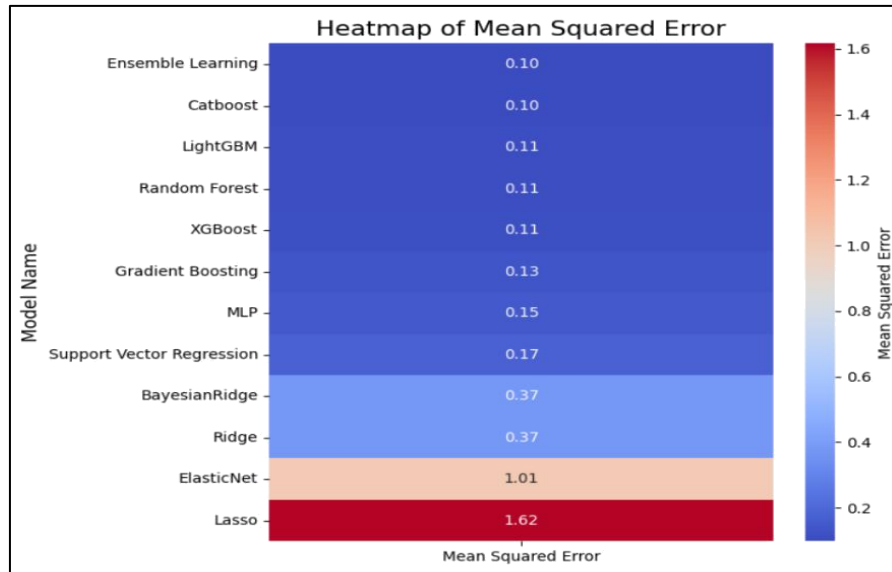


Figure 17. The heatmap (MSE in various machine learning models)

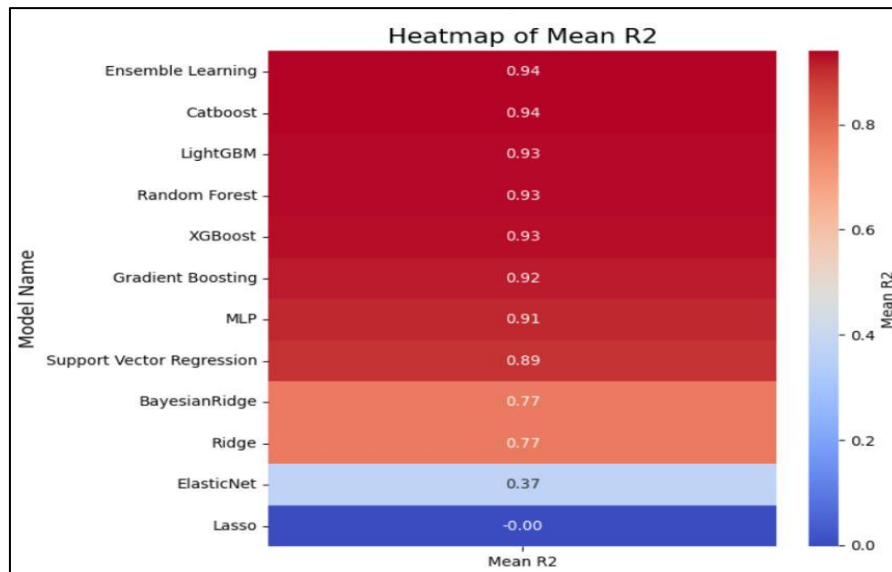


Figure 18. The heatmap (R-squared values in various machine learning models)

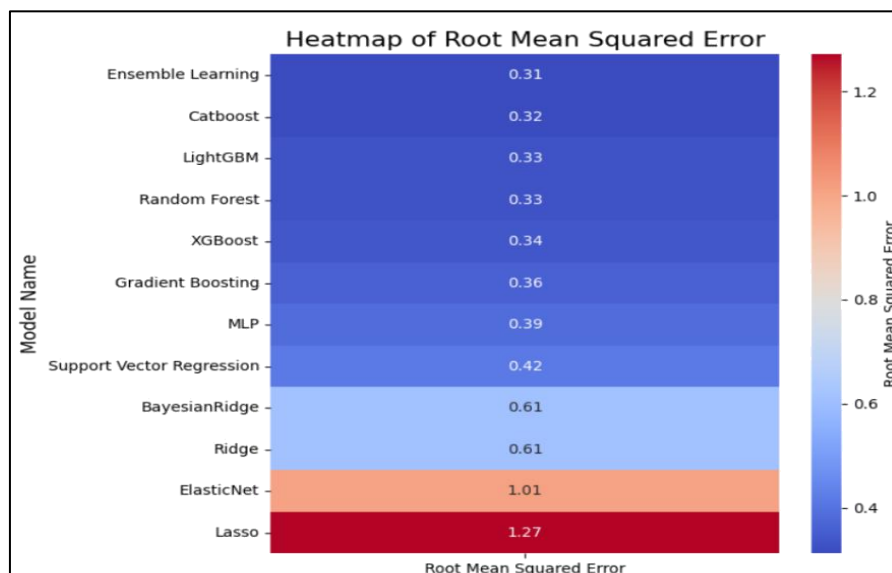


Figure 19. RMSE across various machine learning models

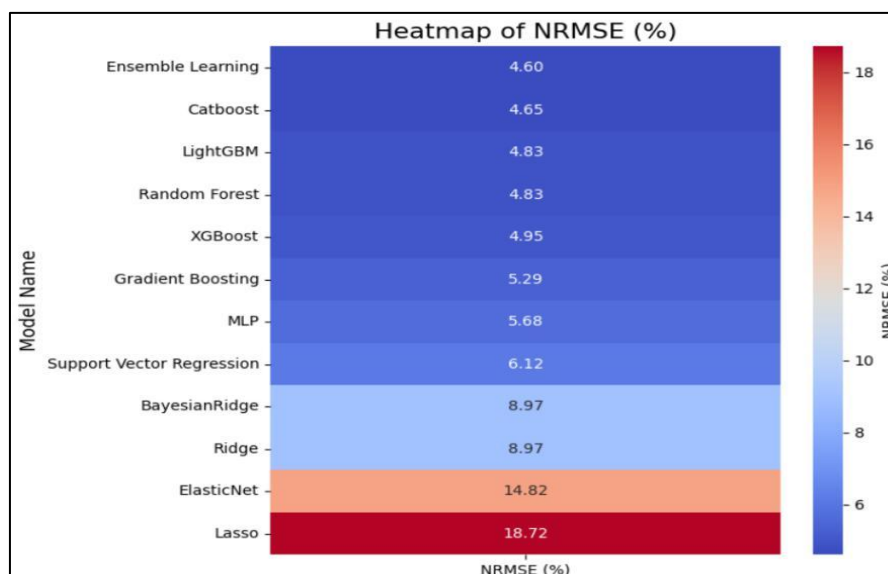


Figure 20. NRMSE models

A low RMSE (Figure 15) on the left side of the plot reflects greater predictive accuracy and a better data fit for models such as Ensemble Learning and CatBoost, whereas higher RMSE values seen in models like Lasso indicate reduced effectiveness in minimizing errors.

The line plot illustrates (Figure 16) the NRMSE across various machine learning models, ranging from approximately 4% for the high-performing Ensemble Learning model to over 17% for the low-performing Lasso model. This variation underscores marked differences in predictive accuracy and suggests potential trade-offs between model complexity and error minimization.

The heatmap highlights substantial differences (Figure 17) in MSE across the evaluated machine learning models, with Ensemble Learning and CatBoost achieving the lowest MSE values, while Lasso records the highest, indicating pronounced variation in predictive performance.

The heatmap (Figure 18) illustrates the variation in mean R^2 values across the models, with Ensemble Learning and CatBoost demonstrating the highest model fit and explaining the greatest proportion of variance in the target variable. In contrast, Lasso is an outlier, displaying a near-zero R^2 score.

The heatmap presents (Figure 19) RMSE values for the different machine learning models, showing that Ensemble Learning achieves the lowest RMSE, whereas Lasso produces the highest, reflecting notable differences in predictive accuracy.

The heatmap compares prediction accuracy (Figure 20) across machine learning models, showing that ensemble-based methods (CatBoost + LightGBM + Random Forest Regressor) with an NRMSE of 4.6% consistently outperform linear regression models such as Lasso and ElasticNet.

9. DISCUSSION AND CONCLUSION

This study presents a machine learning framework for predicting earthquake magnitudes using over 8,000 global seismic records from the U.S. Geological Survey. Through rigorous data pre-processing including the removal of missing values, invalid entries, and non-earthquake events alongside targeted feature engineering and normalization, the dataset

was prepared for robust model training and evaluation. A diverse set of regression models was explored, ranging from linear approaches to advanced ensemble techniques and neural networks. Among these, a stacking ensemble combining CatBoost, LightGBM, and Random Forest achieved the highest predictive accuracy, with an R^2 of 0.9392 and NRMSE below 5%, demonstrating the effectiveness of ensemble learning in capturing complex seismic patterns.

Future work will focus on expanding the feature set to include temporal and geophysical variables, integrating real-time data streams, and exploring deep learning architectures for spatiotemporal modeling. Additionally, deploying the model in operational settings and evaluating its performance in early warning systems could further validate its utility in seismic risk mitigation.

REFERENCES

- [1] Yavas, C.E., Chen, L., Kadlec, C., Ji, Y. (2024). Improving earthquake prediction accuracy in Los Angeles with machine learning. *Scientific Reports*, 14: 24440. <https://doi.org/10.1038/s41598-024-76483-x>
- [2] Hoque, A., Raj, J., Saha, A., Bhattacharya, P. (2020). Earthquake Magnitude Prediction Using Machine Learning Technique. In *Trends in Computational Intelligence, Security and Internet of Things. ICCISIoT 2020. Communications in Computer and Information Science*, Springer, Cham. https://doi.org/10.1007/978-3-030-66763-4_4
- [3] Cui, B., Guo, J., Han, G., Liu, X. (2024). Earthquake magnitude and depth prediction based on machine learning and multiple linear regression models. In *IEEE 2nd International Conference on Sensors, Electronics, and Computer Engineering (ICSECE)*, Jinzhou, China, pp. 1056-1060. <https://doi.org/10.1109/ICSECE61636.2024.10729410>
- [4] Manral, G.S., Chaudhary, A. (2023). Prediction of earthquake using machine learning algorithms. In *International Conference on Intelligent Engineering and Management (ICIEM)*, London, United Kingdom, pp. 1-5. <https://doi.org/10.1109/ICIEM59379.2023.10166658>

- [5] Xiong, P., Tong, L., Zhang, K., Shen, X., Battiston, R., Ouzounov, D., Iuppa, R., Crookes, D., Long, C., Zhou, H. (2021). Towards advancing earthquake forecasting by machine learning of satellite data. arXiv preprint, arXiv:2102.04334. <https://doi.org/10.48550/arXiv.2102.04334>
- [6] Ding, X.Y., Hu, W.J. (2023). Advancements in geological disaster monitoring and early warning systems: A deep learning and computer vision approach. *Traitement du Signal*, 40(3): 1195-1202. <https://doi.org/10.18280/ts.400336>
- [7] JayaLakshmi, G., Madhuri, A., Vasudevan, D., Thati, B., Sirisha, U., Praveen, S.P. (2023). Effective disaster management through transformer-based multimodal tweet classification. *Revue d'Intelligence Artificielle*, 37(5): 1263-1272. <https://doi.org/10.18280/ria.370519>
- [8] Ojeda, J.M.P., Huatangari, L.Q., Arce, J.R., Fernández, N.A., Santisteban, M.A.G., Serrano, M.A.M. (2024). Assembly algorithms for seismic vulnerability estimation in confined masonry dwellings. *International Journal of Safety and Security Engineering*, 14(3): 967-984. <https://doi.org/10.18280/ijss.140327>
- [9] Xing, H.R. (2024). Design of a real-time monitoring and early warning system for engineering safety hazards using image analysis technology. *Traitement du Signal*, 41(5): 2381-2390. <https://doi.org/10.18280/ts.410513>
- [10] Shantaf, A.M., Kutucu, H. (2025). Assessing signal distortion in seismic sensor security systems: A machine learning approach to target identification. *Traitement du Signal*, 42(1): 467-484. <https://doi.org/10.18280/ts.420140>
- [11] Putra, W.S., Sunarno, Mustika, I.W. (2025). Monte Carlo Aggregating method for radon precursor-based earthquake parameter prediction in Java, Indonesia. *International Journal of Safety and Security Engineering*, 15(2): 359-367. <https://doi.org/10.18280/ijss.150217>
- [12] Rahman, F. (2023). Earthquake. Kaggle. <https://www.kaggle.com/datasets/farazrahman/earthquake>
- [13] Earthquake Magnitude, Energy Release, and Shaking Intensity. USGS Earthquake Hazards Program. U.S. Geological Survey. <https://www.usgs.gov/programs/earthquake-hazards/earthquake-magnitude-energy-release-and-shaking-intensity>
- [14] U.S. Geological Survey. Moment magnitude, Richter scale – What are the different magnitude scales and why are there so many? USGS Magnitude Scales. <https://www.usgs.gov/faqs/moment-magnitude-richter-scale-what-are-different-magnitude-scales-and-why-are-there-so-many>
- [15] Event Catalogue – Magnitude Data. International Seismological Centre. <http://www.isc.ac.uk/iscbulletin/search/catalogue/>
- [16] USGS Earthquakes 2024 – Kaggle Dataset. U.S. Geological Survey. <https://www.kaggle.com/datasets/rupindersinghrana/usgs-earthquakes-2024>
- [17] Seismic magnitude scales. Wikipedia. https://en.wikipedia.org/wiki/Seismic_magnitude_scale
- [18] Education and Public Outreach – Earthquake Resources. IRIS Consortium. <https://www.iris.edu/hq/programs/epo>
- [19] Earthquake magnitude classes. Alaska Earthquake Center. <https://earthquake.alaska.edu/earthquake-magnitude-classes>
- [20] Data preprocessing for ML: Options and recommendations. TensorFlow TFX Guide. https://www.tensorflow.org/tfx/guide/tft_bestpractices
- [21] GeeksforGeeks. (2025). What is Machine Learning Pipeline? <https://www.geeksforgeeks.org/blogs/machine-learning-pipeline>
- [22] Chugani, V. (2025). From features to performance: crafting robust predictive models. *Machine Learning Mastery*. <https://machinelearningmastery.com/from-features-to-performance-crafting-robust-predictive-models/>
- [23] GeeksforGeeks. (2025). Supervised machine learning. <https://www.geeksforgeeks.org/machine-learning/supervised-machine-learning/>
- [24] GeeksforGeeks. (2025). Regression in machine learning. <https://www.geeksforgeeks.org/machine-learning/regression-in-machine-learning/>
- [25] Talent500. (2023). Data preprocessing: Cleaning, transforming, and normalizing data for analysis. <https://talent500.com/blog/data-preprocessing/>
- [26] Heatmap of correlation matrix. CodeSignal. <https://codesignal.com/learn/courses/feature-engineering-and-correlation-analysis-in-pandas/lessons/heatmap-of-correlation-matrix>
- [27] Evaluation metrics for regression models. MachineLearning-Basics.com. <https://machinelearning-basics.com/evaluation-metrics-for-regression-models/>

APPENDIX

Appendix I. Dataset feature descriptions

Feature Name	Description
time	Timestamp of the seismic event (UTC format).
latitude	Latitude of the earthquake epicenter (in degrees).
longitude	Longitude of the earthquake epicenter (in degrees).
depth	Depth of the epicenter below Earth's surface (in kilometers).
mag	Magnitude of the earthquake, indicating its strength.
magType	Type of magnitude measurement (e.g., md, ml, mb, mw).
nst	Number of seismic stations that recorded the event.
gap	Azimuthal gap between the two most widely spaced stations (in degrees).
dmin	Minimum distance between epicenter and nearest seismic station (in kilometers).
rms	Root Mean Square of seismic wave amplitudes across stations.
net	Seismic network that reported the event (e.g., USGS, IRIS).
id	Unique identifier assigned to each event.
updated	Timestamp of the latest update to the event record.
place	Textual description of the event location (e.g., nearby city or region).
type	Classification of the event (e.g., earthquake, explosion, quarry blast).
horizontalError	Uncertainty in the horizontal location of the epicenter (in kilometers).

depthError	Uncertainty in the depth measurement (in kilometers).
magError	Uncertainty in the magnitude estimate.
magNst	Number of stations used to calculate the magnitude.
status	Indicates whether the event was reviewed or

	automatically generated.
locationSource	Organization that reported the location (e.g., USGS, regional agencies).
magSource	Source or method used to estimate the magnitude.