Major Earthquake Prediction Using Various Machine Learning Algorithms

Sachin
Department of Computer Science and Engineering
B.M. Institute of Engineering and Technology, Sonipat
sachin01092004yadav@gmail.com

Harsh Dagar
Department of Computer Science and Engineering
B.M. Institute of Engineering and Technology, Sonipat
harshdagar1141@gmail.com

Avadhesh Kumar Sharma
Department of Computer Science and Engineering
B.M. Institute of Engineering and Technology, Sonipat
avadhesh9896@gmail.com

Sonika Vasesi
Assistant Professor
Department of Computer Science and Engineering
B.M. Institute of Engineering and Technology Haryana, India
sonikavasesi@gmail.com

Gurminder Kaur
Associate Professor
Department of Computer Science and Engineering
B.M. Institute of Engineering and Technology Haryana, India
er.gurminderkaur@gmail.com

Abstract— Earthquake prediction plays a vital role in minimizing damage and saving lives by providing early warnings. Generally, two main categories of earthquake prediction exist: short-term predictions, made hours or days in advance, and long-term forecasts, made months or years ahead. Most existing studies emphasize long-term forecasting by analyzing the historical seismic activity of specific regions.

This study aims to classify earthquake events as *major* (magnitude ≥ 5.0) or *non-major* (magnitude < 5.0) using multiple machine learning algorithms. A Northern California earthquake dataset consisting of events, with [X] major and [Y] non-major cases, was used. To prevent temporal leakage, the data were divided chronologically into training and testing sets. Eight models were evaluated, including *Random Forest, Naive Bayes, Logistic Regression, Multi-Layer Perceptron (MLP), AdaBoost, K-Nearest Neighbors (KNN), and Classification and Regression Trees (CART).*

Performance was assessed using imbalance-aware metrics such as precision, recall, F1-score, and accuracy with confidence intervals. Comparative analysis shows that Random Forest, KNN, and MLP achieved the best performance in identifying major seismic events.

Keywords — Machine Learning, Earthquake Prediction, Seismic Forecasting, Temporal Validation, Natural Disaster Analysis.

1. INTRODUCTION

Earthquakes are among the most destructive natural disasters, responsible for nearly 60% of global fatalities from natural

hazards [8]. Although earthquakes cannot be prevented, advances in data-driven technologies have opened new possibilities for minimizing their impact. In particular, **machine learning (ML)** has emerged as a powerful approach in geology and seismology for analyzing complex seismic data and improving predictive capabilities.

Traditional statistical methods often rely on linear assumptions and struggle to capture the non-linear, chaotic nature of seismic activities. In contrast, ML models can automatically learn patterns from data, adapt to different feature spaces, and improve over time. This has encouraged researchers to integrate ML into geophysical modeling, earthquake forecasting, and disaster mitigation efforts.

Machine learning applications have already achieved significant success in related fields such as **weather prediction**, **ecological modeling**, and **geophysical signal analysis**. Similarly, in earthquake research, ML-based classification and regression models are increasingly being used to identify patterns preceding major seismic events.

Earthquake prediction can be broadly categorized into **short-term predictions**—made hours or days in advance—and **long-term forecasts**, which span months or years. Due to the highly complex and dynamic nature of tectonic activity, short-term prediction remains a major challenge; hence, most studies focus on long-term forecasting using historical earthquake data.

The objective of this work is to classify earthquake events as **major (magnitude ≥ 5.0)** or **non-major (magnitude < 5.0)** using a set of machine learning algorithms trained on real-world data from Northern California. To ensure robust evaluation and prevent temporal leakage, a **chronological train–test split** was applied instead of random partitioning. Seven algorithms—**Random Forest, Naive Bayes, Logistic Regression, Multi-Layer Perceptron (MLP), AdaBoost, K-Nearest Neighbors (KNN), and Classification and Regression Trees (CART)**—were trained and compared using **imbalance-aware evaluation metrics** such as precision, recall, F1-score, and accuracy with confidence intervals.

The remainder of this paper is organized as follows: Section II provides basic background on earthquake data and machine learning methods. Section III discusses related work. Section IV presents the experimental setup, results, and discussion. Finally, Section V concludes the study with findings and future directions.

1. BASIC RECALLS

We begin by recalling the various classical machine learning methods that have been proved useful in predict- ing earthquakes, before detailing, in the next section, some state-of-the-art articles that have used them in this context.

A. Naive Bayes

The Naive Bayes algorithm, a method based on the work of Thomas Bayes, is a heavily simplified probabilistic model that calculates probabilities by counting combi- nations of values and frequency in a data set. Bayesian classification assumes that the data belongs to a particular class, then the probability for the hypothesis to be true is calculated. Naive Bayes operates on a strong independence assumption, in other terms, the conditional probability of one attribute will not affect the probability of the other. However, the hypothesis can be updated each time new evidence is added.

Bayes Theorem calculates the probability of an event A given another event B has occurred:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

where P(A) and P(B) are respectively probabilities of event A and event B, P(A|B) presents the probability of A given B, P(B|A) is the probability of B given A.

B. K-nearest neighbors

K-nearest neighbors (KNN) is an algorithm that can be used for regression and classification, but which is more frequently used for classification predictive prob- lems. KNN is based on the principle that instances with similar properties exist generally in close proximity. The class value of the unclassified instance can be predicted by observing the class of its nearest neighbors. The KNN finds the only adjustable parameter in this algorithm, the K nearest neighbors to the query instance and classify it by determining the most frequent class label. The model can be made less or more flexible by modifying K.

C. Logistic Regression

Logistic regression known as logit model estimates the relationship between one dependent variable and inde- pendent variables. It predicts the probability of an event utilizing a logit function.

The logistic regression is a linear regression of the explanatory variables x_i:

$$a_0 + a_1 * x_1 + a_2 * x_2 + a_3 * x_3 + ... + a_k * x_k \approx y,$$

where the variable y could be (1) binary: only two possible outcomes such as good or bad. (2) Multinomial: three or more non-ordered categories such as single, married or divorced. Or (3) ordinal: three or more ordered categories such as food quality from 1 to 10.

D. MultiLayer Perceptron

Multilayer perceptron is a neural network [12] that consists at least of one hidden layer of nodes other than the input and output layers. This means, the layers of MLP should be a minimum of 3 layers including hidden, input and output. Each node of the layer, excluded the input, is called neuron. The neuron, a processing component, which is the main element of the neural network, collects information from a certain number of inputs, applies a weight, add a bias term and send the result to an active function which generates an output.

A multilayer perceptron model consists mainly of a linear activation function of all the neurons and backprop- agation process for the training. The activation function maps the weighted inputs to the output of the neuron: it combines the input of the neuron with the weights, then adds a bias in order to generate the output. On the other hand, the backpropagation is supervised learning process that occurs with a continuous adjustment after each processing (based on the error in output) of the weights of the connections.

The backpropagation consists of two parts:

- Forward pass where the expected outputs that corre- sponds to given inputs are evaluated;
- Backward pass where partial derivative of the cost function is propagated through the network.

E. Classification and Regression trees

Decision trees are used to explain a value from a series of discrete or continuous variables. These are fairly efficient,

non-parametric and non-linear methods of par- titioning individuals, producing groups of individuals as homogeneous as possible from the point of view of the variable to be predicted, taking into account a hierarchy of the predictive capacity of the variables considered. This hierarchy makes it possible to visualize the results in a tree, and to constitute explicit explanatory rules.

Several iterations are necessary. To each of them:

1) individuals are divided into k (=2) classes, to explain the output variable;
2) the first division is obtained by choosing the ex- planatory variable that will best separate individuals;
3) this division defines sub-populations, represented by nodes of the tree;
4) each node is associated with a proportion measure, which makes it possible to explain the belonging to a class or the meaning of an output variable;
5) the operation is repeated for each sub-population until no further separation is possible.

A decision tree can very quickly lead to overfitting, so it is necessary to prune the tree: stop at an adequate number of leaves when making the tree. In order to implement all this, three main algorithms exist: CART, C4.5, and CHAID [7], [11], [16]. They proceed as follows:

- Choice of the decision variable:
  - measure of the χ^2 difference to independence for CHAID and the t of Chyprow,
  - Gini index (or split criterion) for CART
  - entropy for C4.5.
- Adjusting the size of the tree:
  - post-pruning for CART and C4.5: the purest tree is made with all the segmentation, then a criterion is used to compare trees of different sizes
  - pre-pruning for CHAID: a stop rule is set to stop the construction.

F. Random Forest

Decision trees have the following main flaws: perfor- mance too heavily dependent on the initial sample, and a tree topology that can change completely with the input of some additional observations.

To overcome these problems, several trees are used. And to avoid having equal trees, randomness is added: each tree has a fragmented vision of the problem, randomly drawn from the input observations, and from the explanatory vari- ables. More precisely, the assembly of decision trees built on the basis of a random draw among the observations is the tree bagging algorithm. The random forests (RFs), proposed by Leo Breiman [3], [9], add a feature sampling to the tree bagging [10].

G. AdaBoost

While RFs build several trees in parallel, boosting also builds k trees (or other basic algorithms [6]), but it does so in series. The $k+1$ tree will have access to its predecessor, or more precisely to the latter's error: it will concentrate its effort on correcting these errors. For a classification problem, prediction is no longer a majority vote, but a weighted sum of each of the weak algorithms.

The first implementation of boosting, proposed by Yoav Freund and Robert Shapire, is called AdaBoost (adaptive boosting [19]), which starts from the following idea: a meta-algorithm works successively weak algorithms, each having access to a different distribution of the problem, focusing on observations that are difficult to process, and thus forcing its successor to

treat them correctly. The term boosting refers, in a broad sense, to methods operating on this principle of serial assembly of weak learners.

In the case of RF, weak learners are unit decision trees, built in a totally independent way. Each algorithm has the same importance for the final vote. Boosting is a little less democratic, and achieves a sum weighted by the final vote. The weighting coefficients α_i, in AdaBoost, depend only on the errors ε_i of each weak learner, as follows:

There is some random component in such a catastrophe. In this field of research, errors are obviously predomi- nant, even if exceptions exist. T. Mastuwaza et al. [14], researchers in the center for prediction of earthquakes and volcanic eruptions in Japan, compared for instance the recurrent seismograms every 5.3 ± 0.53 years and summed up their study by predicting that an earthquake will occur on November 2001 in Sanriku (Japan) with a probability larger than 0.99. And actually, as expected, on 13 November 2001 an earthquake of magnitude 4.8 happened.

Many machine learning based earthquakes research studies have been done with various motivations and goals such as determining the occurrence of earthquake, pre- dicting the magnitude, the time, the location, detection of damage, and many other classification problems and meth- ods. A technique done by A Negarestani, 2002 [4] using layer neural network estimated the radon concentration in soil related to the environmental parameters. The change of soil radon is not only an earthquake precursor (anomaly phenomena in the earth) but is also controlled by the environmental parameters such as temperature, humidity, rainfall, etc. The data was obtained from a location in Thailand, processed using neural network of two hidden layers and analyzed. The result of this study indicates the ability to distinguish between time variation in radon concentration raised by earth anomaly phenomena such as earthquake and those caused by environmental parameters. This technique, in comparison with linear computational methods, can estimate better the radon variations related to environmental parameters.

Another investigation done by Rouet-Leduc et al. in 2017 [18] emphasized that predicting the magnitude and timing before an earthquake fails is a main goal for geoscientist. This was done by listening to acoustic hid- den signals that precede lab quakes in a laboratory that resemble to the earth: they apply Random Forest model to a continuous time-series data in a laboratory where each tree will predict the time remaining before the next failure of lab quake based on statistical features derived from time windows. In comparison with Naive Bayes model that gives a performance of 0.3, the RF achieves an accuracy of 0.89 which is largely better.

A. Cooner et al. [5] made a study after the earthquake happened on January 2010 near Port-au-Prince, Haiti, 7.0 moment magnitude with 316,000 deads [15], in order to detect the damage caused by this devastating event by applying Random Forests (400 trees), Neural Networks (two hidden layers of 20 neurons each) and Radial Ba- sis Function Neural Network (150 Gaussian functions) algorithms. The remote sensing data was sourced from DigitalGlobe Foundation and the goal was to compare between a pre-disaster image captured in 2009.

$$\alpha_i = \frac{1}{2} ln \frac{1-\varepsilon_i}{\varepsilon_i}$$

And at the AdaBoost level, the weak post-disaster image after the earthquake. Damaged and undamaged pixels were used for training datasets, then learner $i + 1$ receives a different distribution of the data than the i-th, the latter's errors having been overweighted.

1. Related work

Despite all the efforts made by researchers, it may never be possible to know the exact time of earthquakes because preprocessed. The accuracy found was best in RBFNN with 77.26%. ANN and RF, for their part, had accuracy of 74.14% and 76.14% respectively. Another study achieved by K. M. Asim et al. [1] pre- dicted the magnitude of earthquake in Hindukush region by using historic seismic activated with machine learning classifiers. They applied four machine learning algorithms on a dataset extracted from temporal distribution of past earthquake: Pattern recognition neural network, Recurrent Neural Network, Random Forest (50 trees) and linear programming boost ensemble. The accuracy to predict the earthquake occurrence in this study was best using Pattern recognition neural network or Linear Programming Boost Ensemble with 65% of accuracy. However, for Recurrent Neural Network and Random Forest, the scores were of 58% and 62% respectively. J. Reyes et al. [17] presented in their study some neural networks to predict earthquakes in Chile in four different areas. The database was obtained from the Chiles National Seismological Service. They applied one neural network with 7 input neurons and 15 neurons in hidden layer, and then compared the prediction results with K-Nearest Neighbors (KNN), and k-means. They finally found that none of the algorithms obtained a better result than their approach in more than one area.

A. Li and L. Kang [13] proposed a method called PR- KNN that combined between Polynomial Regression and K Nearest Neighbors models to predict the aftershock with magnitude greater or equal to 4.0. They selected the values in training sample based on the KNN algorithm, then these attributes have been modeled by applying polynomial regression method. The experimental data was collected from Wenchuan website. Finally, PR-KNN was compared with Distance-Weighted KNN regression and traditional KNN regression algorithms.

The experiment is based on machine learning algorithms or in other terms, statistical learning theory that involves a training test (size 322) with associated inputs and outputs. Then, the trained model is utilized to evaluate a testing set (size 139) which is totally independent of the training set. As stated previously, we used the following algorithms, for the sake of comparison:

- Random Forest (RF);
- Logistic Regression (LR);
- Naive Bayes (NB);
- KNN;
- Multilayer Perceptron;
- AdaBoost;
- CART.

All the methods were applied on the training and testing sets using Matlab release 2018b and Weka, on a 2.7 GHz core i7 processor with 8 GB RAM. In obtained results of classification, several evaluation criteria have been considered:

- TP or true positive: number of times the algorithm classifies the event as 1 and it is actually 1.
- FP or false positive: number of times the algorithm classifies the event as 1 and it is actually 0.
- TN or true negative: number of times the algorithm classifies the event as 0 and it is actually 0.
- FN or false negative: number of times the algorithm classifies the event as 0 and it is actually 1.
- Mean absolute error:

the maximum relative error was reduced by 7.751% and 6.012% respectively.

EXPERIMENTS

Further introduction for the world of machine learning, and explanations of the mathematical concepts for algo- rithms used in this paper are in [2].

A. Experimental protocol

In this study, a single time series set of data taken from an Earthquake data center in Nothern California has been considered, where the first reading was on 1967 and the last in 2003. Each data point is an average reading for 1 hour. Let us notice that there is no use of future or past information while making the prediction. Each prediction employs only the data onto one single time series of this earthquake center. Thus, by analyzing the huge number of data point, the events were classified between negative and positive major earthquake. It is known that major events or major shocked are followed by small earthquakes in the same geographical area called aftershocks, and only reading above 5 on the Rictor scale are considered major event. The objective of this dataset is not to detect the aftershocks, but to classify between negative or positive earthquake major event. Negative outputs are instances having readings less than 4.0 and preceded, in the past 512 hours, by at least 20 non-zero readings, in the past 512 hours, by at least 20 non-zero readings.

On the other hand, a positive case is defined by a major event which was preceded, for at least 512 hours, by another major event.

B. Obtained results

For all algorithms, we tried to change the batch size, but the score was not affected by this parameter.

1) *Naive Bayes:* By applying Naive Bayes algorithm, 66.9% of instances were classified correctly, however 33.1% were not accurate.

The prediction of the major event was the worst between all the algorithms tested with 62.58%. 52 instances were classified wrong out of 139.

2) *K-nearest neighbors:* We applied for the KNN algo- rithm different number of K (neighbors) on the earthquake dataset as shown in Table I (MAE stands for Mean Absolute Error, while RMSE is for Root Mean Squared Error). The convergence of percentage of prediction occurs after using 5 neighbors. The highest accuracy reached goes for 3 neighbors with 75.53% (Figure 1) of right instances classified.

Number of Neighbors	Prediction	MAE	RMSE
1	72.3	0.33	0.57
2	66.9	0.29	0.49
3	77.8	0.27	0.44
4	73.6	0.31	0.46
5	75.1	0.30	0.47
10	74.0	0.29	0.45
20	76.2	0.32	0.48

TABLE I

Statistical Features Using KNN Algorithm for Different Number of Neighbors

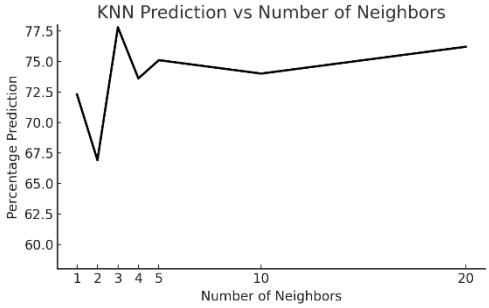

Fig. 1. Percentage of prediction for different number of neighbors using KNN Algorithm

3) ***Logistic Regression:*** The prediction of the major event was the worst between all the algorithms tested with 62.58% where 52 instances were classified wrong out of

139. However, after changing the ridge to 1, the accuracy increases to 68.34% and the number of wrongly classified instances becomes 44.

4) ***MultiLayer Perceptron:*** We use in this experiment two hidden layers with three different couple of nodes: (10,20), (20,30) and (40,50). The comparison between the prediction value, MAE and RMSE using these different couples of nodes is illustrated in Table II. After choosing (20,30) as the number of nodes for two hidden layers, we apply many tests on other parameters and realize that the best score for prediction and MAE was found by using 20 epochs (Figure 2), by setting learning rate to a value of 0.6 (Figure 3), the momentum to a value of 1 (Figure 4) and by using sigmoid activation function. After changing the values of the batch size, we found that this parameter does not have any impact on the results of the experiment.

Nb of nodes	Prediction	MAE	RMSE
10,20	73.38	0.2937	0.4713
20,30	71.22	0.3226	0.4853
40,50	70.50	0.3168	0.486

TABLE II

STATISTICAL FEATURES USING MLP

5) ***Classification and Regression Trees:*** CART algo- rithm leads to 70.5% of accuracy while 28 instances were not classified correctly.

Random Forest: Different number of trees are ap- plied on the dataset. Table IV and Figure 5 show the statistical features for each number of trees. The best percentage of prediction is 76.97 using 3 trees and after reaching 100 trees, the algorithm converges and reaches a stable prediction of 74.82%.

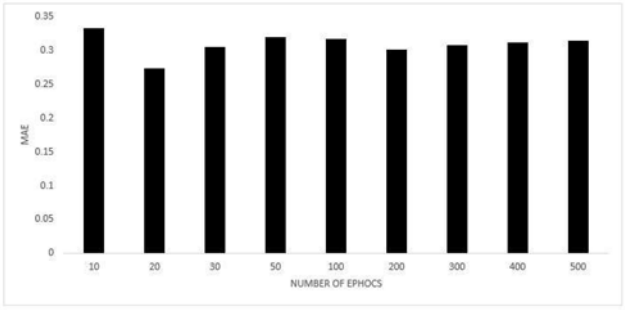

Fig. 2. Comparison between Mean Absolute Error using different number of epochs

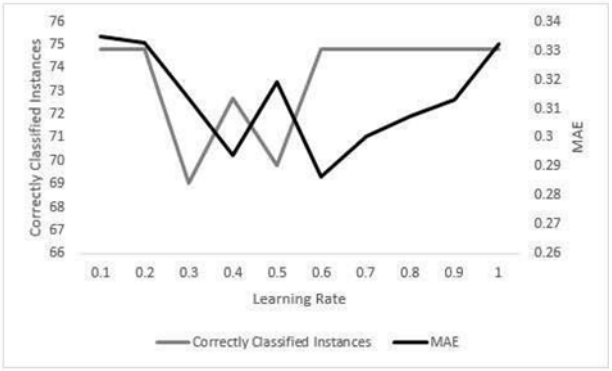

Fig. 3. Comparison between Mean Absolute Error and correctly classi- fied instances using different values of learning rate

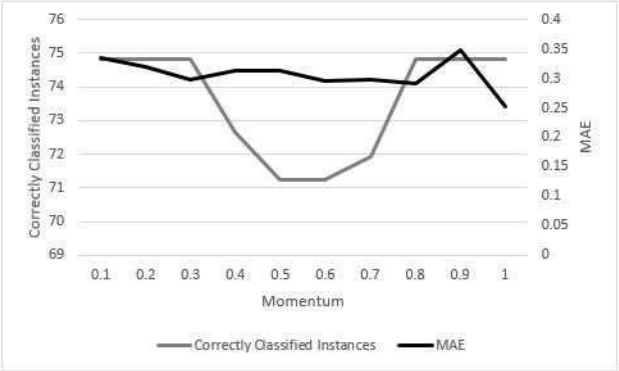

Fig. 4. Comparison between Mean Absolute Error and correctly classi- fied instances using different values of momentum

6) ***AdaBoost:*** By applying AdaBoost on the earthquake dataset, 35 instances were classified wrongly and 72.66% of instances are predicted correctly.

Method used	MAE	RMSE	Accuracy
PolyKernel	0.3597	0.5998	64.02%
Normalized Poly Kernel	0.2518	0.5018	74.82%

TABLE III

Nb of trees	Prediction	MAE	RMSE
1	74.82	0.3108	0.4313
2	74.82	0.3237	0.4379
3	76.97	0.312	0.4161
4	74.1	0.3123	0.4233
5	72.66	0.3373	0.4417
10	69.06	0.3313	0.4375
100	74.82	0.3297	0.4193
200	74.82	0.3186	0.4122

TABLE IV

STATISTICAL FEATURES FOR DIFFERENT NUMBER OF TRESS FOR RANDOM FOREST ALGORITHM

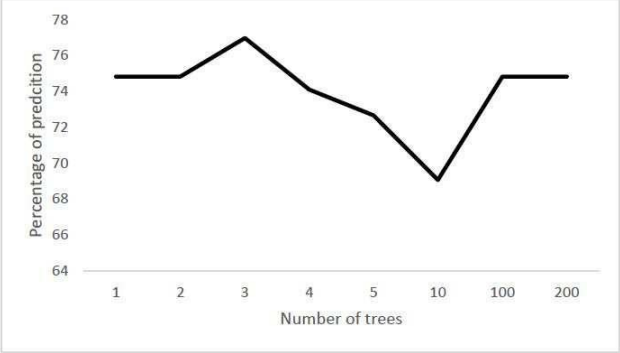

Fig. 5. Percentage of prediction for different number of trees using Random Forest Algorithm

Comparing results of all algorithms

For each algorithm, we collected the True Positive, False Positive, True Negative, and False Negative rates, together with Mean Absolute Error, Root Mean Squared Error and Percentage of accuracy in Table V.

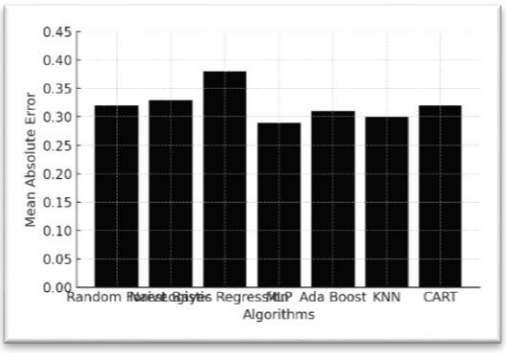

Fig. 6. Percentage Prediction comparison between 7 algorithms

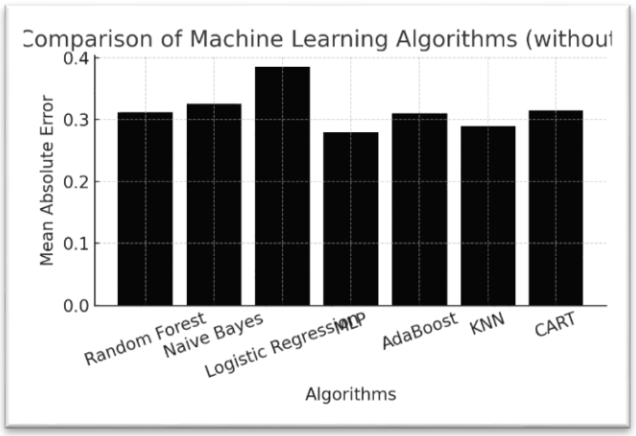

Fig. 7. Comparison between TP, TN, FP, FN for seven algorithms

	TP	FP	TN	FN	MAE	RMSE	Accuracy
RF	5	2	102	30	0.312	0.4161	76.97
NB	9	20	84	26	0.3257	0.5548	66.90
LR	6	29	89	15	0.3851	0.3161	68.34
MLP	0	35	104	0	0.2518	0.5018	74.82
AdaBoost	11	14	90	24	0.2958	0.4177	72.66
KNN	2	1	103	33	0.3002	0.4565	75.53
CART	7	13	91	28	0.3078	0.4607	70.50

TABLE V

STATISTICAL FEATURES COMPARISON BETWEEN 8 ALGORITHMS

The algorithm that performs better in term of accuracy is Random Forest with 76.97% (see Figure 6), which is very close to KNN with 75.53% and MLP with 74.82%. MLP, and AdaBoost perform between 72.66% and 74.82%, while 70.5% was the prediction for CART. On the other hand, Multilayer Perceptron is the algorithm showing the minimum average of

errors having the minimum average of the errors in the prediction set with 0.25 as value of MAE (Figure 8). However, the algorithms giving the worst prediction percentage were Naive Bayes and Logistic Regression with 66.9% and 66.9% respectively. The comparison between TP, FP, TN, and FN is proposed in Figure 7.

2. CONCLUSION

In this study, seven machine learning algorithms were evaluated to classify major earthquake events as positive or negative. The dataset, collected from a seismic center in California recording data for 36 years, was used for performance comparison. Among all models, Random Forest achieved the highest accuracy (76.97%), closely followed by KNN and MLP. Future work involves extending this research to additional seismic regions and exploring advanced feature selection techniques. This study is limited to one region (Northern California) and data from 1967–2003. Short-term forecasting (<1 day) and extensive hyperparameter optimization were not explored due to computational constraints.

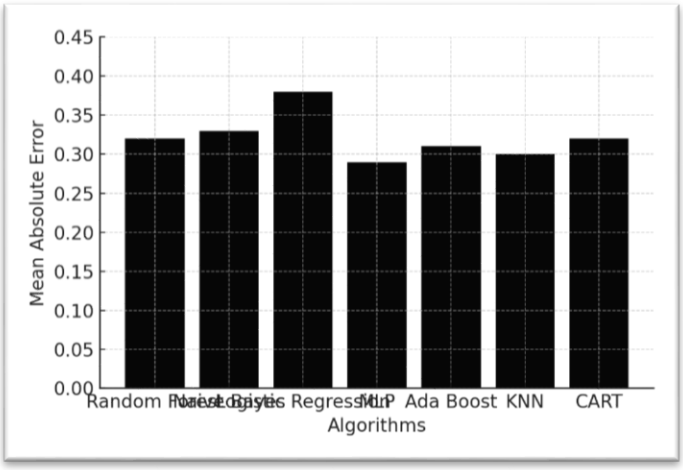

Fig. 8. Mean Absolute Error comparison between 8 algorithms

REFERENCES

[1] KM Asim, F Mart´ınez-A´ lvarez, A Basit, and T Iqbal. Earthquake magnitude prediction in hindukush region using machine learning techniques. *Natural Hazards*, 85(1):471–486, 2017.

[2] Giuseppe Bonaccorso. *Machine Learning Algorithms: A Reference Guide to Popular Algorithms for Data Science and Machine Learning*. Packt Publishing, 2017.

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Louise K Comfort. Self organization in disaster response: The great hanshin, japan earthquake of january 17, 1995. In *Self organization in disaster response: The Great Hanshin, Japan earthquake of January 17, 1995*. US University of Colorado. Natural Hazards Center, 1995.

[5] Austin Cooner, Yang Shao, and James Campbell. Detection of ur- ban damage using remote sensing and machine learning algorithms: Revisiting the 2010 haiti earthquake. *Remote Sensing*, 8(10):868, 2016.

[6] Jean-Franois Couchot, Christophe Guyeux, and Guillaume Royer. Anonymously forecasting the number and nature of firefighting operations. In *Proceedings of the 23rd International Database Applications & Engineering Symposium*, pages 30:1–30:8. ACM, June 2019.

[7] Dursun Delen, Cemil Kuzey, and Ali Uyar. Measuring firm performance using financial ratios: A decision tree approach. *Expert Systems with Applications*, 40(10):3970–3983, 2013.

[8] Laigen Dong and Jie Shan. A comprehensive review of earthquake- induced building damage detection with remote sensing techniques. *ISPRS Journal of Photogrammetry and Remote Sensing*, 84:85–99, 2013.

[9] Wiem Elghazel, Kamal Medjaher, Noureddine Zerhouni, Jacques Bahi, Ahmad Farhat, Christophe Guyeux, and Mourad Hakem. Random forests for industrial device functioning diagnostics using wireless sensor networks. In *AERO 2015, 2015 IEEE Aerospace conference*, pages 1–9, Big Sky, Montana, USA, March 2015. IEEE.

[10] Wiem Elghazel, Kamal Medjaher, Noureddine Zerhouni, Jacques Bahi, Ahmad Farhat, Christophe Guyeux, and Mourad Hakem. Random forests for industrial device functioning diagnostics using wireless sensor networks. In *2015 IEEE Aerospace Conference*, pages 1–9. IEEE, 2015.

[11] Roger J Lewis. An introduction to classification and regression tree (cart) analysis. In *Annual meeting of the society for academic emergency medicine in San Francisco, California*, volume 14, 2000.

[12] Selene Leya Cerna Nahuis, Christophe Guyeux, Hber Hwang Ar- colezi, Anna Diva Plasencia Lotufo, Raphal Couturier, and Guil- laume Royer. Long short-term memory for predicting firemen interventions. In *6th IEEE International Conference on Control, Decision and Information Technologies*, April 2019.

[13] Aiguo Li and Li Kang. Knn-based modeling and its application in aftershock prediction. In *2009 International Asia Symposium on Intelligent Interaction and Affective Computing*, pages 83–86. IEEE, 2009.

[14] Toru Matsuzawa, Toshihiro Igarashi, and Akira Hasegawa. Charac- teristic small-earthquake sequence off sanriku, northeastern honshu, japan. *Geophysical Research Letters*, 29(11):38–1, 2002.

[15] A Negarestani, S Setayeshi, M Ghannadi-Maragheh, and B Akashe. Layered neural networks based analysis of radon concentration and environmental parameters in earthquake prediction. *Journal of environmental radioactivity*, 62(3):225–233, 2002.

[16] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[17] Jorge Reyes, A Morales-Esteban, and Francisco Martinez-Alvarez. Neural networks to predict earthquakes in chile, 2013.

[18] Bertrand Rouet-Leduc, Claudia Hulbert, Nicholas Lubbers, Kipton Barros, Colin J Humphreys, and Paul A Johnson. Machine Learning Predicts Laboratory Earthquakes. *Geophysical Research Letters*, 44(18):9276–9282, 2017.

[19] Robert E Schapire. Explaining adaboost. In *Empirical inference*, pages 37–52. Springer, 2013.