Prediction of earthquakes: A Step Towards Predicting the Unpredictable

by

Maheen Chowdhury
18201069
Md Abdur Rahaman
18201093
Tafhim Sadman Islam
18301220
Rahanuma Sultana
18201105
Anjolika Rahman
21101347

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
January 2023

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

[illegible]

Md Abdur Rahaman
18201093

[illegible]

Rahanuma Sultana
18201105

[illegible]

Tafhim Sadman Islam
18301220

Anjolika

Anjolika Rahman
21101347

MAHEEN

Maheen Chowdhury
18201069

Approval

The thesis titled "Predicting the Unpredictable: A Machine Learning Approach to Earthquake Prediction" submitted by

1. Anjolika Rahman (21101347)
2. Tafhim Sadman Islam (18301220)
3. Rahanuma Sultana (18201105)
4. Md Abdur Rahaman (18201093)
5. Maheen Chowdhury (18201069)

of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 19, 2023.

Examining Committee:

Supervisor:
(Member)

Moin Mostakim

Lecturer
Department of Computer Science and Engineering

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD

Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

One of the most catastrophic natural disasters is an earthquake, especially because they typically occur without warning. It has catastrophic effects on both the economy of a nation and human life. Our paper is a step towards taking the challenge and complexity of predicting earthquakes and implies that the research here aims to make progress in this field. In this paper, we have proposed a hybrid model combining multiple algorithms which analyses already existing datasets. We have done our work in two steps. Firstly, we trained our model on multiple algorithms such as KNN, SVM, XGBOOST and ADABOOST. Then we created a hybrid model out of these algorithms which gave us the best results in terms of accuracy and precision. The use of machine learning techniques for earthquake prediction is examined in this thesis. We concentrate on the use of multiple algorithms: k-Nearest Neighbors (KNN), AdaBoost, and Support Vector Machines (SVM). We start by going over the state of earthquake forecasting right now and how machine learning is applied in this area. We next go over our study, which involves feeding our models input features made up of both seismological and geodetic data. We assess each algorithm's performance using a range of evaluation indicators and contrast the outcomes with conventional statistical techniques. We explore the significance of our results for future research in this field and show how these machine learning techniques have the potential to be used to forecast earthquakes. Overall, this thesis makes a positive contribution to the current work to increase the precision and dependability of earthquake prediction utilizing cutting-edge machine learning methods.

Keywords: Earthquake; K-Nearest Neighbors(KNN); Support Vector Machine(SVM); Extreme gradient Boosting(XGBOOST); Adaptive boosting(ADABOOST); Machine Learning; Prediction/Forecasting

Acknowledgement

We want to extend our heartfelt thanks to our thesis adviser, Moin Mustakim, for his guidance, support, and encouragement throughout the research and writing process of this group thesis. Their expertise and valuable feedback helped us to shape our ideas and to ensure the quality of our work.

We would also like to acknowledge the support of our families, friends, and loved ones, who have always been there for us, providing us with the encouragement and motivation we needed to complete this endeavor.

Last but not least, we would like to express our gratitude to all of the participants who gave their time to help with our study; without their participation, this thesis would not have been possible.

Thank you all for being a part of our journey.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Thoughts behind the Prediction Model

A abrupt release of energy in the Earth's lithosphere, which then results in seismic waves, is what causes an earthquake. An earthquake is defined as this shaking of the earth's surface. The earliest documented earthquake for which we have detailed information took place in China in 1177 B.C. With the technology that is now available, it is predicted that 500,000 earthquakes may be detected annually. Approximately 100,000 of these are detectable. In addition to California and Alaska in the United States, El Salvador, Guatemala, Chile, Peru, Indonesia, the Philippines, Iran, Pakistan, the Azores in Portugal, Turkey, New Zealand, Greece, Italy, India, Nepal, and Japan all experience minor earthquakes on a regular basis. Some of these locations are found in the USA. Because of the exponential relationship between earthquake size and frequency, stronger earthquakes happen less often. For instance, there are around ten times as many earthquakes with a magnitude larger than 4 as there are earthquakes with a magnitude greater than 5 during a same time frame. For instance, it has been found that the average frequency of earthquakes in the United Kingdom, which has a low seismicity, is as follows: an earthquake of magnitude 3.7 to 4.6 every year, an earthquake of magnitude 4.7 to 5.5 every 10 years, and an earthquake of magnitude 5.6 or higher once every 100 years. Consequently, a model for early prediction will be helpful in lowering the severity and scope of the calamity. One of the various techniques that have been created can be used to forecast earthquakes' times and locations with a high degree of accuracy. Despite the huge amount of study seismologists have done, it is currently not feasible to create projections for a certain day or month that can be verified scientifically. Despite the fact that forecasting is sometimes considered to be a sort of prediction, it is customary practice to differentiate between earthquake prediction and earthquake forecasting. The probabilistic assessment of general earthquake hazards, including the frequency and magnitude of catastrophic earthquakes in a given location over a certain time span of years or decades, is fundamental to the science of earthquake forecasting.

The size or magnitude of earthquakes is determined by measuring the amplitude of the seismic waves recorded on a seismograph and the distance between the seismograph and the earthquake. These are put into a formula to produce an earthquake's magnitude, which is a measure of the energy it released. For every unit increase in magnitude, the amount of energy released almost triples. The magnitude of an earthquake was normally estimated using the Richter scale. It is currently often calculated using the seismic moment, which is equivalent to the fault area times the average displacement on the fault. For properly described faults, the probability that a segment may rupture during the next many decades can be predicted. These measurements will serve as the primary source of data for the prediction model. With the help of the prediction model, we can identify the most precise predictions.

1.2 Problem Statement

Those attempting to monitor earthquakes still face a special problem. So why are earthquakes so difficult to forecast, and how may we become more adept at doing so? We must comprehend certain notions about how earthquakes happen in order to respond to it. The tectonic plates, which make up Earth's crust, are large, angular rock masses that each rest on a heated, partly molten layer of the planet's mantle. As a result, the plates spread extremely slowly—anywhere between 1 and 20 millimeters every year. But these minute motions are strong enough to split the interacting plates apart. And in vulnerable areas, the escalating pressure might eventually cause an earthquake. It's challenging enough to keep track of these minute fluctuations, but there are many more variables at play when shifts become seismic occurrences. Earthquakes come in four varieties: tectonic, volcanic, collapse, and explosion.

- A tectonic earthquake is one that occurs when the earth's crust splits as a consequence of forces acting on neighboring plates and rocks that change their physical and chemical makeup.
- The earth's crust ruptures during tectonic earthquakes as a result of geological forces acting on nearby plates and rocks, which cause physical and chemical changes.
- Small earthquakes known as collapse earthquakes are caused by seismic waves produced by rock explosions on the surface and happen underground in mines and caves.
- Explosive earthquakes are those brought on by the explosion of a nuclear or chemical weapon.

Different fault lines are constructed from a variety of rocks, some of which are stronger than others. These rocks respond differently to friction and high temperatures while under pressure. Some may discharge lubricating fluids consisting of superheated minerals that lessen friction along fault lines when they partly melt. However, some are left dry and vulnerable to harmful pressure builds-ups. Aside from the currents of hot rocks traveling through the Earth's mantle, all of these faults are also susceptible to a variety of gravitational pressures. What should we be looking at among these hidden factors, and how do they fit into our expanding toolbox for making predictions? The behavior of the plates is somewhat cyclical as a result of several of these forces occurring at essentially constant rates. Additionally, several seismic wave types are created during earthquakes, and they move through rock at various speeds:

- Longitudinal P-waves (shock- or pressure waves)
- Transverse S-waves (both body waves)
- Surface waves – (Rayleigh and Love waves)

Additionally, these waves don't provide very useful information for predicting earthquakes. Today, long-term forecasting, which considers when and where earthquakes have already occurred, provides many of our most trustworthy cues. This enables us to forecast when very active faults, such as the San Andreas, are poised for a significant earthquake on a millennia scale. However, since there are so many factors at play, this technique can only provide extremely broad temporal predictions.

Researchers have looked at the vibrations Earth makes before a quake to forecast more recent disasters. Seismometers have been used by geologists for a long time to monitor and record these minute alterations in the earth's crust. A thorough fusion of geological data may be used by NASA's Quakesim program to pinpoint vulnerable areas. Recent research suggests that the most important earthquake indicators could be imperceptible to all of these sensors. Researchers in the area discovered remarkably high amounts of the radioactive isotope pair, radon and thoron, in 2011, right before an earthquake devastated Japan's east coast. These vapors are released to the surface via microfractures when the crust becomes stressed just before an earthquake.

Researchers have looked at the vibrations Earth makes before a quake to forecast more recent disasters. Seismometers have been used by geologists for a long time to monitor and record these minute alterations in the earth's crust. A thorough fusion of geological data may be used by NASA's Quakesim program to pinpoint vulnerable areas. Recent research suggests that the most important earthquake indicators could be imperceptible to all of these sensors. Researchers in the area discovered remarkably high amounts of the radioactive isotope pair, radon and thoron, in 2011, right before an earthquake devastated Japan's east coast. These vapors are released to the surface via microfractures when the crust becomes stressed just before an earthquake. The most important point to emphasize is that since precursors are not only restricted to earthquakes, they are challenging to employ for short-term forecasting (for example, unusual lights in the atmosphere may arise before geomagnetic storms or have a technogenic origin). Additionally, several antecedents existed before various earthquake kinds that happened in various locations.

There is currently no universal earthquake prediction methodology. Furthermore, the scientific community has not been able to find any solution to this challenge. However, the rapid development of machine learning algorithms and their successful application to a variety of problems suggests that these technologies may be useful in detecting hidden patterns and making accurate predictions.

1.3 Research Objectives

The goal of this project is to provide a system for calculating earthquake magnitude and location using machine learning classifier methods. The model will be constructed using a number of machine learning classifier techniques in order to do that. In addition, we will train this model using a variety of datasets. The objectives of this research are:

1. To deeply understand the core reasons for earthquakes.
2. To deeply understand relations among these elements.
3. To develop a different model with different algorithms to find out the best prediction result.
4. To evaluate the model.
5. To combine all models to build a sustainable prediction system.

Chapter 2

Literature Review

Correct prediction of earthquakes is still an impossible task. No way of accurately predicting earthquakes is the reason which makes this the most dangerous natural disaster among all others. First of all, they frequently take place with little prior notice, leaving no opportunity for preparation. The problem is further complicated by the fact that earthquakes frequently trigger other natural disasters including tsunamis, snowfalls, and landslides. [4] They could potentially lead to industrial catastrophes (for instance, the Fukushima Daiichi nuclear disaster). There isn't a standard approach for predicting earthquakes right now. [18] Analysis and signs that a large earthquake is about to occur should be included in reliable forecasts. It is very difficult to identify these signals in the seismic analysis since they are rarely present before earthquakes have taken place. Due to its distinct advantages, such as quick imaging and a large image-acquisition range, remote sensing has been one of the most widely utilized technologies for earthquake research. However, early research on pre-earthquake anomalies and anomalies detected by remote sensing has mostly focused on the detection and interpretation of abnormalities in relation to a particular physical characteristic. Since earthquake signals are typically masked by background noise, many analyses are focused on isolated events, which leaves us with a poor comprehension of these complicated natural phenomena. On a global level, the applicability of such analyses is still not being proven. [13] Due mostly to the availability of massive training data sets, data-driven machine learning has lately made significant advancements in its ability to forecast instantaneous and future fault-slip in laboratory trials. However, on Earth, the intervals between earthquakes can be as long as 100 years, and only a small section of each earthquake cycle is normally covered by geophysical data [1]. The training of machine learning algorithms for forecasting fault slides in the Earth is severely hampered by sparse data. Here is a transfer learning method for training a convolutional encoder-decoder that forecasts fault-slip behavior in lab trials using numerical simulations. [15] The three sections of another study are as follows: (1) evaluating the significance of influencing features on seismic fatality based on the random forest to select indicators for the prediction model; (2) categorizing the study area into different grades of risk zones using a strata fault line dataset and WorldPop population dataset; and (3) developing a zoning support vector regression model (Z-SVR) with optimal parameters that are suitable for various risk areas.

The positives and negatives of using a variety of machine learning approaches to forecast ground-motion intensity data, taking into account the features of the source, the distance between the source and the site, and the characteristics of the regional site [19]. [14] In practice, models based on linear regression, complete with predetermined equations and coefficients, are utilized in order to forecast ground motion. However, the limitations of the linear regression models may prohibit them from being able to extract the complex nonlinear patterns that are hidden within the data. This article examines the possible advantages of using a few different machine learning approaches, such as Support Vector Machine, Random Forest, and Artificial Neural Network, as statistical methods for ground motion prediction. Specifically, the study focuses on the Support Vector Machine. [17] Assessing the response statistics of nonlinear systems is one of the most significant components of engineering design. As a result, using the Monte Carlo method has proven to be beneficial despite the high amount of computational resources it requires. Around the design point of a sophisticated and high-dimensional system, it is particularly hard to get a meaningful assessment of the statistics. This point is quite near to the point at which the system will collapse structurally. A neural network is used to learn the response behavior of a structure when it is subjected to an initial nonstationary ground excitation subset. This learning is done based on the spectrum features of a ground acceleration record that has been chosen. The higher processing efficiency of the neural network may then be used to make predictions about the entire sample set, which is a far larger dataset than the one used for the initial training. Researchers Adeli et al. estimated the magnitude of an earthquake by using seismic indicators in conjunction with a probabilistic neural network. Our model provides accurate forecasts for magnitudes ranging from 4.5 to 6 in the southern California area. There have been two case studies that have ultimately shown that NN can predict accurately when given the right data to work with, as stated by Moustra et aluseof .'s artificial neural networks to forecast earthquake activity in the Greece region using time series data and seismic electric signals. According to their research, there have been two case studies that have ultimately shown this. cite article6. When applied to prediction tasks, the upgraded version of the flower pollination algorithm (FPA) that was developed by Hegazy et al. has a higher degree of accuracy. In this study, Ma et al. recommended using a neural network that was based on a genetic algorithm. The GA NN model was applied to six different seismic indicators in order to assess how closely those indicators are related to the earthquakes that occur most frequently in China. The problem of earthquakes is solved by using SVM in the Maceda et al. study. This research came to the conclusion that SVM is an excellent tool for dealing with classification problems despite having a limited training set. The information that was acquired from earthquake stations was relied on by Li et al. so that they could differentiate between earthquakes and other types of seismic activity. [3] In spite of the complexity of the problem, Rajguru et al. were able to successfully employ ANN (artificial neural network) and GA (genetic algorithm) to predict the source location of earthquakes as well as their magnitudes. A deep learning method was developed by Wang et al. in order to discover the association between earthquakes in various areas, which is beneficial in earthquake prediction. This relationship was determined by Wang et al (LSTM). An ANN model was applied by Reyes et al. in order to make a prediction about the magnitude of an earthquake within a certain time window or within a predetermined threshold over the course of the following five days. Through the utilization of statistical tests and trials, it was proved that this method had a higher success rate than other machine learning classifiers [2]. In a variety of seismic zones, Martnezlvarez et al. used a variety of seismic indicators as inputs to an artificial neural network (ANN). The results of these experiments revealed that seismic indicators are the most useful qualities for earthquake prediction. Zhou et al. demonstrated how accurate the neural network could be in predicting the magnitude of an earthquake by utilizing the LM BP approach. The authors Morales-Esteban et al. (cite article7) applied ANN and statistical methods to the study of earthquakes in two different seismic zones. Whether the magnitude is limited by an interval or determined by a threshold, the Alborán Sea and the Western Azores-Gibraltar Fault. Wang et al. proposed using a neural network called RBF (Radial Basis Function) for the goal of predicting earthquakes, and their findings demonstrated that RBF is a good technique for addressing this non-linear problem. The results of an experiment in which Alarifi et al. used ANN to anticipate earthquakes in the Red Sea region indicated that ANN's predictive powers are better to those of all other statistical models. Asencio-Cortés et al. proposed a model that uses machine learning to anticipate the size of an earthquake seven days from now using cloud infrastructure as their data source. Tan et al. came up with the idea of using support vector machines (SVM), and their findings were encouraging, leading to the development of a novel method for predicting earthquakes. The analysis of data relating to earthquakes in Chile is of interest to Florido and others. [9] The data were labeled after being clustered using various approaches. Large earthquakes, those with a magnitude of at least 4.4, were not difficult to locate. Last et al. utilized a multi-objective infofuzzy network algorithm and a range of data mining techniques and time series to anticipate the magnitude of the greatest seismic occurrence that would take place in the following year based on data from the records that were kept in the previous years [5]. Rafiei et al. combined a classification algorithm (CA) with a mathematical optimization algorithm (OA) in order to provide an early warning several weeks before the most significant seismic event. This was done in order to determine the location of the earthquake that would have the greatest magnitude. [10] In order to evaluate how the Taiwan earthquake reacted, Kerh et al.citeartcle15 combined a genetic algorithm with a neural network. The combined model produced findings that were superior to those produced by a neural network model used on its own. Mirrashid et al. used the approach of adaptive neuro-fuzzy inference that was built by three algorithms—subtractive clustering (SC), grid partition (GP), and fuzzy Cmeans—to estimate future earthquakes with a magnitude of 5.5 or higher (FCM). The ANFIS-FCM model was shown to have the highest level of accuracy for calculating the magnitude of earthquakes, as evidenced by the findings [7]. The feed-forward neural network fared the best out of all of the computational intelligence models that Asim and his colleagues examined and evaluated to anticipate earthquakes in northern Pakistan. Frequency Ratio (FR) and Logistic Regression were linked by Umar et al. in article 16 so that they could circumvent the limitations posed by each method on its own (LR). This model had a substantial amount of success in Indonesia and delivered accurate earthquake forecasts. Asim et al. focussed on mathematically analyzing the seismic indicators and then applying tree classifiers to them in order to anticipate the earthquake magnitude in the Hindu Kush region. They demonstrated that the rotation forest generated a better prediction than the random forest did. [11] The three machine learning (ML) approaches that are most often used in classification and regression are extreme learning machines (ELMs), artificial neural networks (ANNs), and support vector machines (SVMs). On the other hand, due to the fact that these approaches are employed, methods that use local optimization training processes that are comparable to the gradient descent process used in ANN may encounter problems with overfitting and local minima. [5] Swarm intelligence methods like particle swarm optimization (PSO), follower pollination algorithm (FPA), ant colony optimization (ACO), and artificial bee colony may be able to overcome the constraints of machine learning models like ANN, SVM, and ELM approaches (ABC). By incorporating meta-heuristic algorithms or swarm intelligence into the process of training and improving conventional machine learning models, it is feasible to increase the precision and generalizability of these approaches.

Furthermore, there is still disagreement among scientists as to whether it is possible to solve this issue. However, the quick development of machine learning techniques and their successful application to a variety of issues suggest that these technologies might aid in the discovery of hidden patterns and the generation of precise forecasts.

2.1 Surface Movement Geo Location

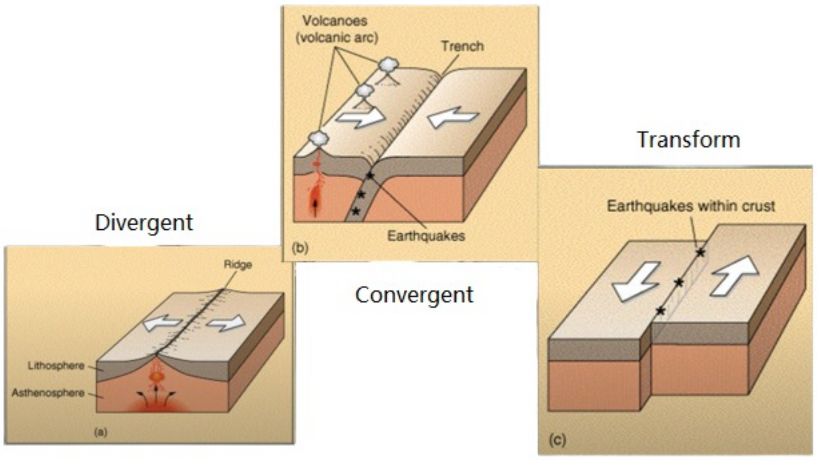


Figure 2.1: Tectornic plate boundaries

An earth quake is defined as a vibration in the ground that results in an instantaneous release of energy in the Earth's lithosphere and generates seismic waves [8]. Therefore, tracking Surface mobility and examining its pattern could help us develop a more accurate prediction model. Divergent, convergent, and transform tectonic plate borders are the three different kinds.

1. When two tectonic plates diverge, they create divergent borders. Magma from the Earth's mantle rises to the surface and solidifies to create new crust at these borders. Mid-ocean ridges, where new oceanic crust is produced, are the result of this process.

2. Convergent boundaries occur when two tectonic plates move towards each other. At these boundaries, one plate is usually forced beneath the other, a process known as subduction. This can lead to the formation of volcanic islands and mountain ranges, such as the Andes in South America.

3. When two tectonic plates glide past one another horizontally, transform borders are created. There is no crustal material movement along these borders, yet there may be substantial seismic activity, including earthquakes. The San Andreas Fault in California serves as an illustration of this kind of border.

It is worth noting that plate boundaries are not fixed, they can change over time. For example, convergent boundary can change to divergent boundary over time.[7] At earthquake data collection facilities, several streams of seismic data [16] have been compiled by sensors. Another significant problem is getting this massive amount of seismic data to the central server. Major difficulties with data privacy, accessibility, security, data protection, and dependability are also brought up. [6] An ML approach called FL can speed up the process of predicting earthquakes by onsite training local data models and ensuring data privacy at local stations. It is tested using a dataset of the Western Himalayan zone that was instrumentally recorded and contains thirty million recordings from the last 35 years [16]. FL has been used to forecast earthquakes, allowing dispersed clients to cooperatively train a common prediction model. It dissociates the necessity for cloud storage of huge data from the capacity to do machine learning. Data is kept safe at its original location in the proposed architecture because it does not move via the network [16]. The suggested architecture is more efficient in terms of speed and accuracy since massive data is not transferred over the network; instead, a small-sized data model is.

The techniques that are utilized to extract data models from client computers must, however, meet the proposed system's efficiency criteria.

2.2 Wave Observation

Earthquakes generate a variety of waves. [14] The seismic body-wave velocities underwent a noticeable precursory shift before to the earthquake at San Fernando, California. The effect is instantly linked to the dilatancy phenomena in fluid-filled rocks once it is realized that the P-wave velocity is the main cause of this shift. [2] The time-volume relation that was created by integrating the data from the current inquiry with those from earlier ones lends credence to this concept. [7] It appears that a fluid-flow or diffusive phenomenon regulates the amount of time that elapses between the start of dilatancy and the return to the fully saturated state necessary for rupture because the length of the precursor period is inversely proportional to the square of the effective fault dimension. By examining this wave type, length, time difference, and effect, we may create a test set to train multiple ML algorithms on and examine their prediction output.

2.3 Previous Earthquake Record

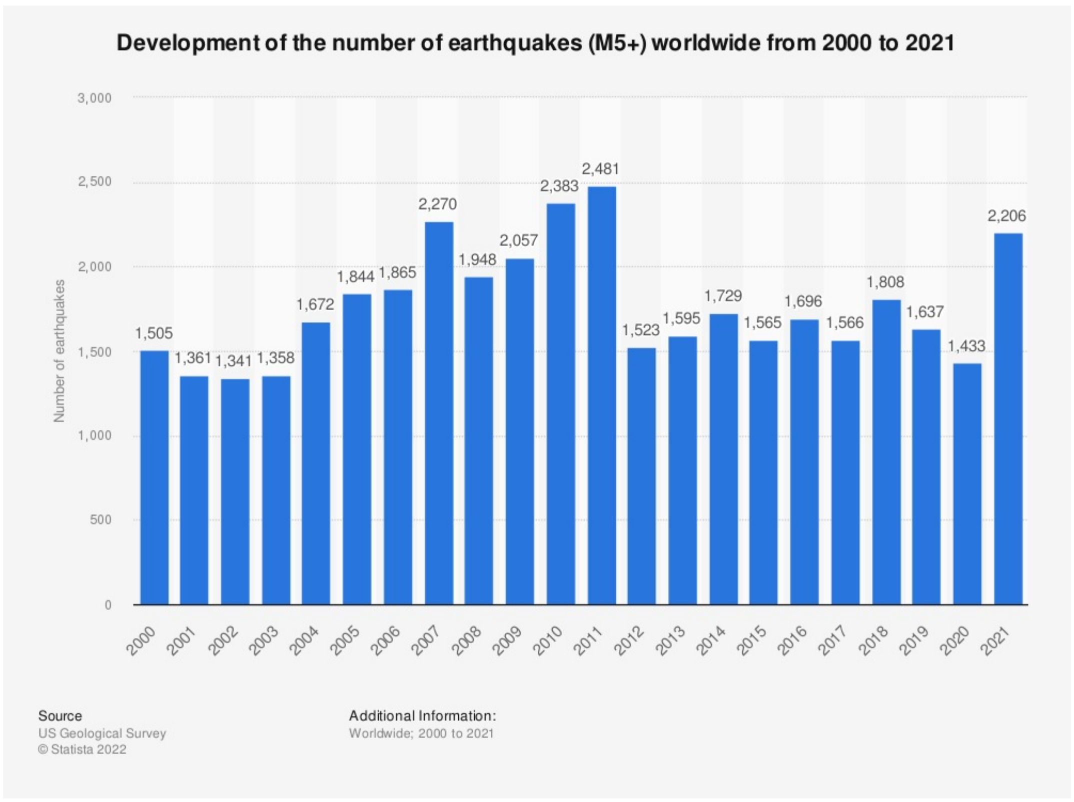


Figure 2.2: Number of earthquakes globally 2000-2021

By studying previous earthquake records and using that as a test subject for the ML algorithm we may find a pattern to predicting earthquakes early. Another similar paper [17] shows machine learning algorithms have been used to predict the magnitude of future earthquakes. This study investigates the effect of spatial parameters on four [6] ML algorithms' performance, Support
Another study examines the advantages and disadvantages of several machine learning methods for creating ground motion prediction models. Typically, regression-

based supervised learning is used in ground motion models [14]. In recent geoscience investigations, the usage of machine learning methods has significantly increased. The possible alternatives to the traditional linear regression-based ground motion prediction models are assessed in this paper [14].The possible limitations of the technique based on linear regression served as the driving force for this investigation. Three distinct machine learning algorithms—specifically, a combination of human and machine learning techniques—are employed to accomplish this. In this study, support vector machines (SVM), AdaBoost, and XGBoost, KNN and Hybrid model are all taken into account. 4528 ground-motion recordings connected to 376 distinct earthquakes are used to train the algorithms. Results show that all machine learning approaches perform better when there is enough data available [12].

Chapter 3

Description of model

Our primary goal is to analyze different seismic activity patterns and build a classification model for earthquake prediction on a particular place in a selected time duration. Our system will be retrained with multiple data set of the previously occurred earthquakes and their seismic activity pattern. There will be multiple algorithms used for this purpose. So that we can compare the accuracy and precision. Once the model is trained then we can present a data set of recently recorded seismic activity and the system will be able to output a classification model.

Proposed Model

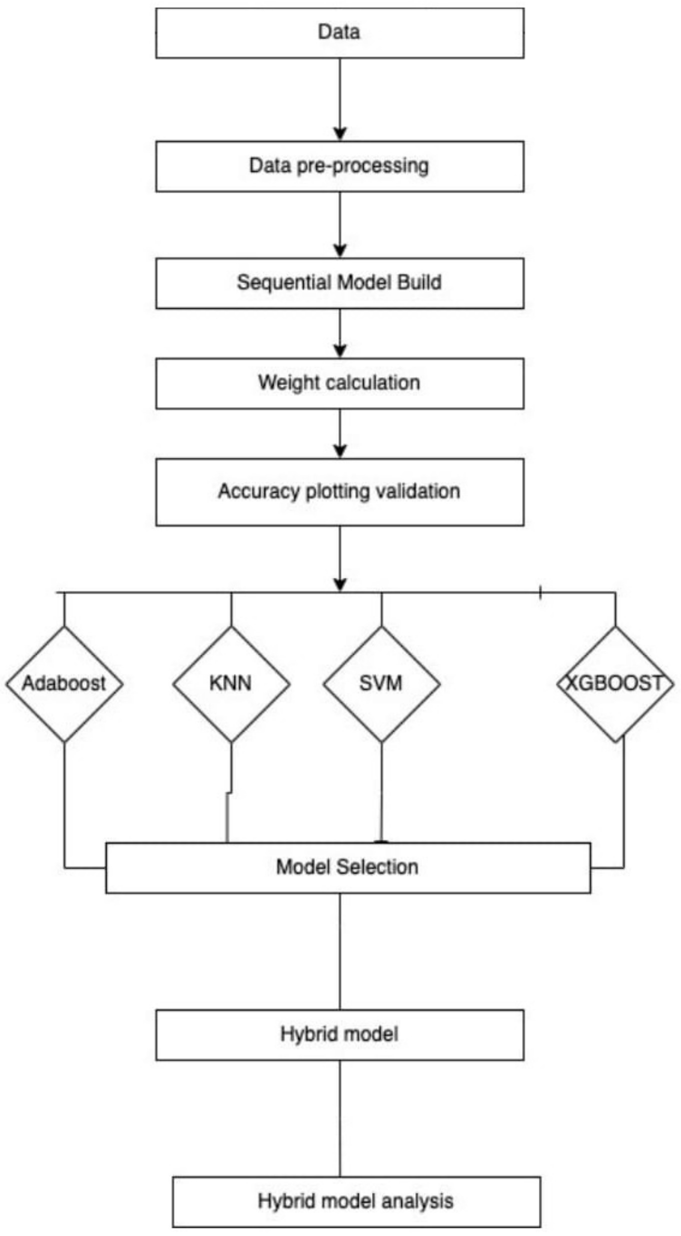


Figure 3.1: System overview

Earthquake detection/classification using machine learning models involves the use of algorithms to analyze seismic data and identify patterns that indicate an earthquake is occurring or about to occur. According to our model, after collecting the dataset we preprocessed the data according to the algorithms' needs. Here we divided the dataset into two categories because it is not possible to hot encode numerical values along with objects during the preprocessing. Then we built a sequential model to use machine learning algorithms. Then we calculated the weight of the model so that we can further use it while building a web application or building a predictive model. Then the dataset is split for training and testing. After that, we did the model fitting. Then we calculated the model accuracy and validation. At this stage, we calculated the accuracy, precision, recall, and f1-score for all our selected algorithms. Now, we analyze the outcomes of all of these algorithms to find out the most accurate ones. And finally, we used these algorithms to build our hybrid classification model.

3.1 Data Collection

Our dataset was obtained from Kaggle's "Significant Earthquakes, 1965-2016" [1]. It has 23,412 distinct data points, out of which 23,232 are earthquakes, 180 are due to other events, including 4 explosions, 175 nuclear explosions, and 1 rockburst, as illustrated in figure 3.2. We solely used the dataset's earthquake data for our study.

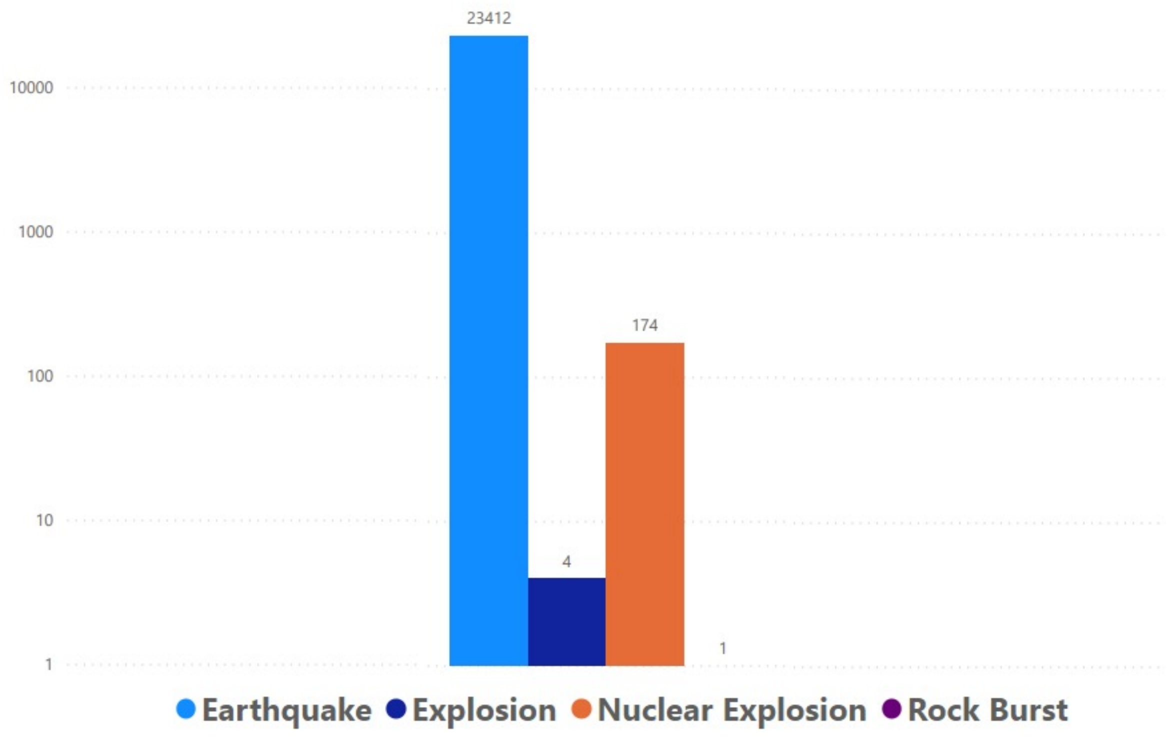


Figure 3.2: Different Labels in Dataset

The term "seismic data" means all geologic, geophysical, seismic, and related technical data and information (including all such data and information obtained during and resulting from any seismic surveys, such as unprocessed and processed data, interpretive data and analyses, maps, electric logs, core data, pressure data, decline curves, and related reports and analysis) relating to or covering all or any part of the Leases, Lands, Wells, or Units, including all such data and information obtained during and

Seismometers have been installed by scientists at several seismic stations throughout the globe in a variety of locations to research ground motion and track earthquakes. An earthquake may occur as a consequence of a seismic event, which is energy that manifests as waves. The research area's geography, the people' participation, the time crunch, and the availability of ground data are some of the difficulties encountered during data collecting. Data gathering is an extremely important step since it will influence the research's conclusions and decide how accurate the results will be. These seismic stations capture the information and transmit it to their own servers. Additionally, their data is extracted for analysis, reporting, and other types of study.

3.1.1 Seismic Data Collection

There are many seismic stations across the world in many different sites where scientists have placed a Seismometer to study ground movements and record earthquakes. Since seismic energy manifests as waves, it will induce ground trembling and eventually an earthquake. The research area's geography, the people' participation, the time crunch, and the availability of ground data are some of the difficulties encountered during data collecting. Data gathering is an extremely important stage since it will influence the research's conclusions and decide how accurate the results will be. These seismic stations record and send the data to their own servers. And from there data is pulled for analyzing, reporting and other research purposes.

The National Earthquake Information Center (NEIC) locates and measures all large earthquakes that take place across the globe, and it instantly notifies government and non-governmental organizations, scientists, important infrastructure, and the general public of this information. With the use of cutting-edge digital national and worldwide seismograph networks and collaborative international agreements, the NEIC gathers and makes available to researchers, the general public, and scientists a vast seismic database that serves as a basis for their work. The NEIC is the country's repository and data center for seismic data.

This dataset contains information on every earthquake with a recorded magnitude of 5.5 or greater since 1965, including the date, time, location, depth, and source.

3.2 Data Pre-processing

Data preprocessing is the initial stage in the process, and we are starting with Machine Learning models. While processing, it's possible that not all of the data will be necessary to accomplish our goal, or that some data will be incomplete, includes irrelevant information, or lack some essential factors. As a result, in order to provide the most effective output, we must preprocess or turn raw input into meaningful data. Our dataset has many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling missing data, noisy data, etc.

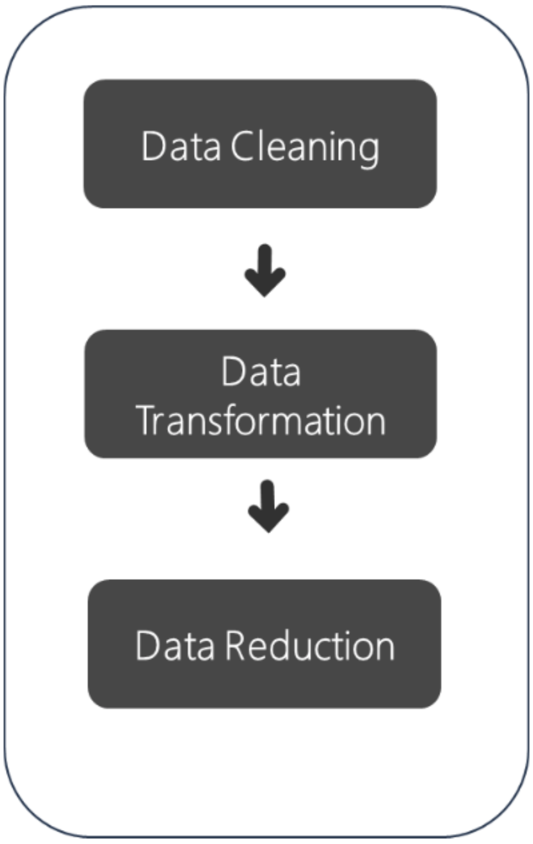


Figure 3.3: Steps of Data Preprocessing

3.2.1 Data Cleaning

We started the data cleaning part by dropping some columns that were not relevant to our purpose. Following that, checking for missing data is done, and filled those missing values with the most probable values.

There are several strategies for handling null values during data preprocessing, including:

- Deleting the entire row or column that contains the null value: This strategy is appropriate if the null value is in an important feature and the missing data is not possible to be replaced.
- Replacing the null value with a predefined value: This strategy is appropriate when the data is missing randomly, and the missing value can be replaced with the mean, median, or most frequent value of the feature.
- Interpolating the missing value: This strategy is appropriate when the data is missing in a time series, and the missing value can be replaced with the value of nearby rows.
- Using a machine learning algorithm for imputation: Some machine learning algorithms are capable of imputing missing data, like KNN imputation.

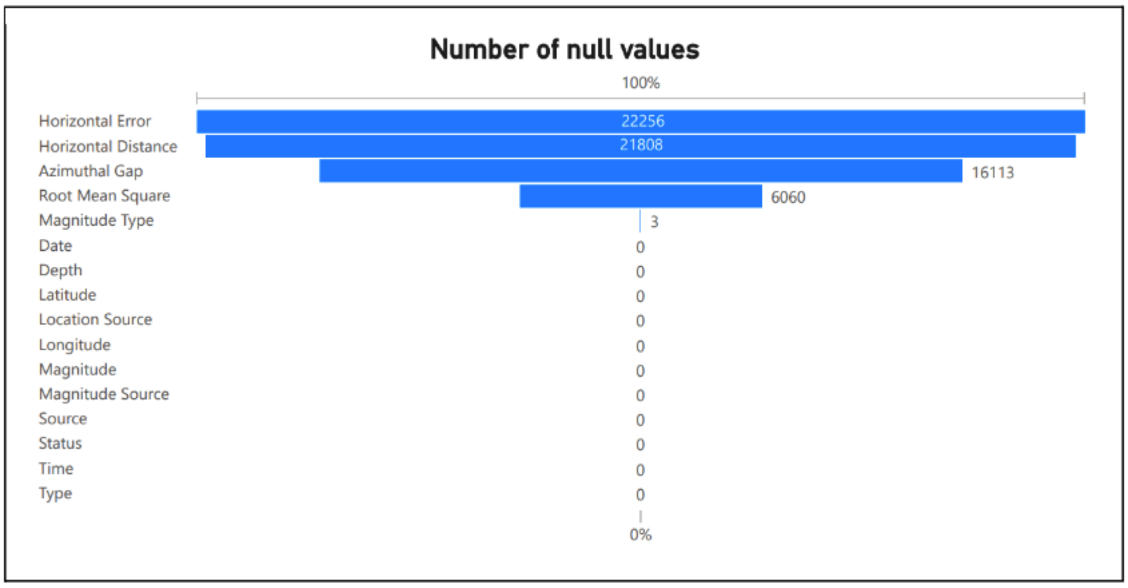
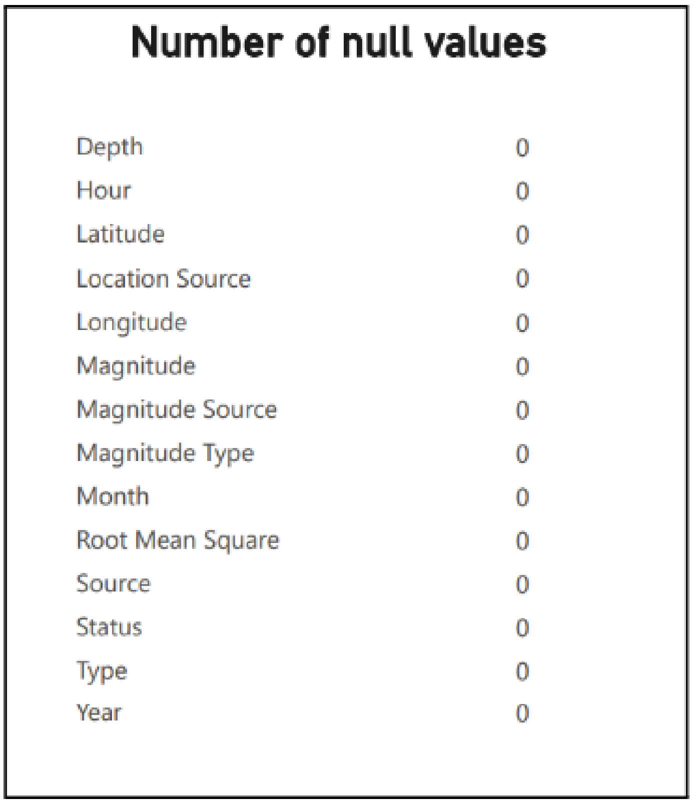


Figure 3.4: Null values checking (before)



Number of null values

Depth	0
Hour	0
Latitude	0
Location Source	0
Longitude	0
Magnitude	0
Magnitude Source	0
Magnitude Type	0
Month	0
Root Mean Square	0
Source	0
Status	0
Type	0
Year	0

Figure 3.5: Null values checking (after)

3.2.2 Data Transformation

Various approaches to data transformation can be utilized, based on the particular task at hand and the tools at hand. Here are a few typical procedures for carrying out data transformation we have applied on our dataset:

- Identify the data that needs to be transformed: This step involves identifying which columns or features of the data need to be transformed, and what kind of transformation is needed.
- Prepare the data: This step involves cleaning and preprocessing the data to ensure that it is in a format that can be easily transformed. This may include filling in missing values, removing outliers, or converting data types.
- Apply the transformation: This step involves using a specific transformation method to convert the data into the desired format. For example, this could involve scaling the data, normalizing it, or encoding categorical variables.
- Validate the transformed data: This step involves checking that the transformed data is in the correct format and that the transformation has been applied correctly.
- Use the transformed data for modeling: Once the data has been transformed, it can be used for building machine learning models, visualization, or further exploration.

It's important to note that these steps may need to be repeated multiple times, and the order in which they are applied may vary depending on the specific task and the tools being used. It's also important to consider the performance of the model after each step of transformation, to check if the model's performance is improving or not.

3.2.3 Data Reduction

Data reduction is a process of reducing the number of features in the data to make it more manageable, and to improve the performance of machine learning models. Here are a few common methods for implementing data reduction:

- Identify the number of features in the dataset: Determine the total number of features in the dataset and consider if it is necessary to reduce the number of features.
- Examine the correlation between features: Use a correlation matrix or heatmap to identify highly correlated features. You can then remove one of the correlated features to reduce the dimensionality of the data.
- Use feature selection methods: Apply feature selection methods such as mutual information, correlation-based feature selection, and wrapper methods to select the most informative features from the data.
- Use regularization methods: Apply regularization methods such as Lasso and Ridge to set some of the coefficients to zero, effectively removing those features from the model.
- Evaluate the performance of the model: After each step of data reduction, evaluate the performance of the model to ensure that it is not negatively impacted by the reduction in the number of features.

It's important to keep in mind that data reduction should be done with care, as removing too many features could negatively impact the performance of the model. Additionally, it is important to validate the model's performance with different evaluation metrics and cross-validation techniques to ensure that the feature selection is robust.

3.3 Sequencial Model building

As we know that a sequential model is a type of deep learning model that we are using in our machine learning approach to process our sequential datasets, which is basically a time series data containing text and numerical data. It is a linear stack of layers, where the output from one layer is used as the input to the next layer. The sequential model is a simple and efficient way to build deep learning models, and it is particularly useful for tasks such as image classification, text generation, and speech recognition.

To use our sequential model in machine learning, we first needed to import the relevant libraries and modules, such as Keras, and TensorFlow. Next, we had to define the architecture of our model by creating an instance of the Sequential class and adding layers to it. There are several types of layers that can be added to a sequential model, including dense layers, convolutional layers, and recurrent layers. Once the architecture of our model is defined, we then compile it by specifying the optimizer, loss function, and metrics that we will be using. After that, we train the model on our dataset using the fit() method and then evaluate its performance on a validation or test dataset.

This model allows us the easy construction and training of deep learning models. They are called ”sequential” because they are composed of a linear stack of layers, where the output of one layer is used as the input for the next. This allows for the model to learn increasingly complex representations of the input data as it progresses through the layers.

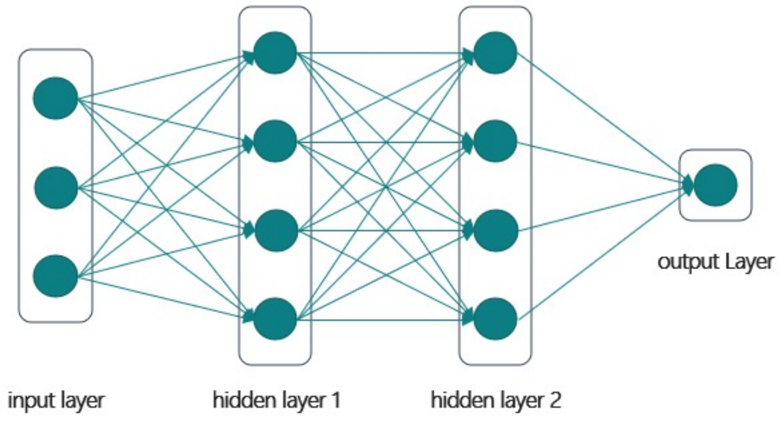


Figure 3.6: Sequential Model Architecture

The sequential model in question is a linear stack of layers in deep learning, where each layer’s output feeds into the next. The layers are added to the model using the ”add” technique and might be fully connected layers, convolutional layers, or recurrent layers. A sequential model’s training procedure begins with a base set of random weights for each layer. After that, the model is trained using a dataset made up of input-output pairs. The model’s layers process the input, and the result is compared to the real result. Using a loss function, such as mean squared error, the difference between the expected and actual output is determined. The layers' weights are then modified to reduce the inaccuracy. An optimization technique, such as stochastic gradient descent, is used to do this (SGD). The weights' gradient with respect to the loss function is calculated by the optimization process, and the weights are updated in the gradient's opposite direction. To reduce the mistake, this procedure is done across a number of epochs, or iterations. Once trained, the model may be used to make predictions based on fresh inputs. The model's layers process the input, and the result is the anticipated result. The sequential model is considered one of the most popular approaches in deep learning, as it allows for easy and modular construction of deep learning models. It is particularly useful for simple feedforward neural networks, where the input is passed through the layers in a single pass, without any loops or recursions. However, it's important to note that the sequential model isn't suitable for all types of tasks, for example in the case of natural language processing tasks where the context of the previous words is important, recurrent neural networks are more suitable. Additionally, the sequential model can be improved by adding regularization techniques such as Dropout, Batch Normalization, and early stopping to prevent overfitting. In summary, a sequential model is a linear stack of layers in deep learning, where the output of one layer is fed as input to the next. The layers can be added to the model using the "add" method. The model is trained using a dataset, and the weights of the layers are adjusted during training to minimize the error between the predicted output and the actual output. Once trained, the model can be used for making predictions on new inputs.

3.4 Weight Calculation

In machine learning, the process of figuring out the parameters' (also known as weights') values in a model that minimizes the error between the projected output and the actual output is referred to as weight calculation. Finding a system of weights that can be used to generate precise predictions on brand-new, untainted data is the aim of supervised learning. Training is the process of locating these weights. The model's initial collection of random weights serves as the basis for the training process. The model is then given the input data, and it provides a prediction. Using a loss function, such as mean squared error, the difference between the expected and actual output is determined. The weights are then changed to reduce the inaccuracy. An optimization technique, such as stochastic gradient descent, is used to do this (SGD). The weights' gradient with respect to the loss function is calculated by the optimization process, and the weights are updated in the gradient's opposite direction. To reduce the mistake, this procedure is done across a number of epochs, or iterations. Using the weights that were discovered during training, the model may then be utilized to generate predictions on fresh, unforeseen data.

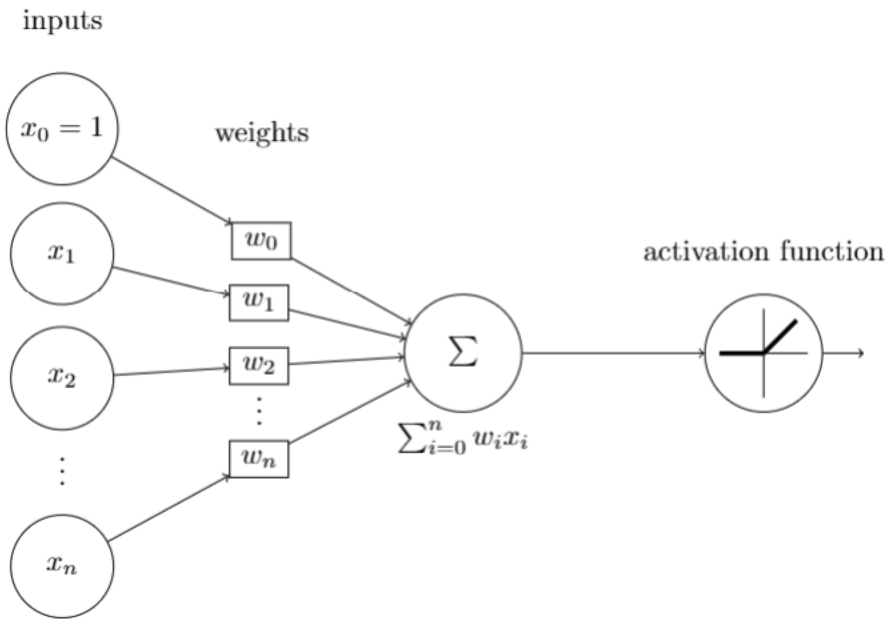


Figure 3.7: Weight Calculation

There are various methods for weight calculation in machine learning, including gradient descent, stochastic gradient descent, and the L-BFGS algorithm. Gradient descent is a first-order optimization algorithm that is commonly used for weight calculation in machine learning. It iteratively updates the weights in the direction of the negative gradient of the loss function, which measures the difference between the predicted output and the true output. The learning rate is a hyperparameter that determines the size of the step taken in the direction of the negative gradient. Stochastic gradient descent is a variant of gradient descent that updates the weights using a single example (or "batch") at a time, rather than using the entire dataset. This can be more computationally efficient, but it can also result in more noise in the weight updates. The L-BFGS algorithm is a second-order optimization algorithm that is commonly used for weight calculation in machine learning. It uses information about the Hessian matrix of the loss function (the matrix of second derivatives) to make more informed updates to the weights. In all the above methods, the goal is to minimize the loss function which is the difference between the predicted output and the true output. The weights are updated in a way that the loss is minimized over time.

In addition to these strategies, regularization techniques like L1 and L2 regularization may also be employed to avoid overfittings. The penalty term added to the loss function by L1 regularization is proportional to the weights' absolute value, while the penalty term added by L2 regularization is proportional to the square of the weights.

3.5 Data Fitting

To create predictions or comprehend the underlying link between the input characteristics and the output variables, our model is trained on the chosen dataset in this context. This process is referred to as "data fitting."

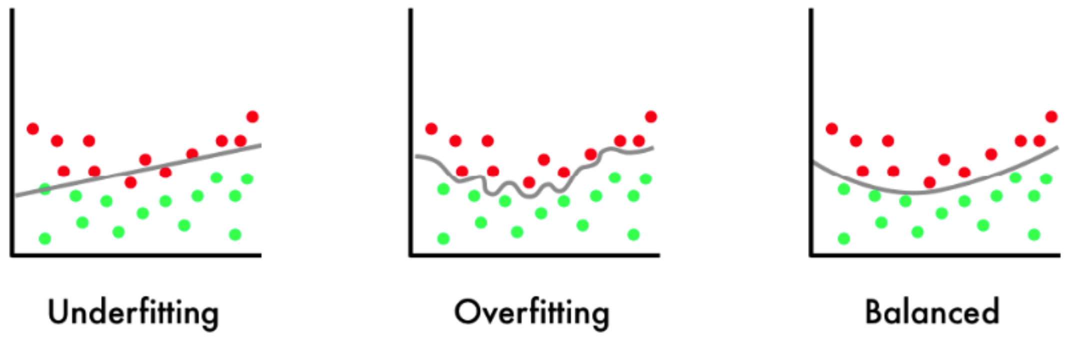


Figure 3.8: Model Fitting

It is used to train our model to learn the underlying patterns and relationships present in the data, so that it can make predictions on unseen data. Data fitting is a crucial step in supervised learning, where the goal is to train a model that can generalize to new data. The process of data fitting involves several steps:

- Defining the model architecture: This step involves choosing the type of model that is most appropriate for the task at hand. For example a neural network or support vector machine.
- Training the model: Once the model architecture is defined, the model is trained on our selected dataset. This step involves providing the model with input features and corresponding output variables, and iteratively updating the model's parameters (such as weights) in order to minimize the difference between the predicted output and the true output.
- Evaluating the model: Once the model is trained, it is evaluated on a separate dataset to measure its performance. This step is used to check the model's ability to generalize to new data.

Data fitting can be performed using various optimization algorithms such as gradient descent, stochastic gradient descent, and L-BFGS. Regularization techniques such as L1 and L2 regularization can also be used to prevent overfitting of the model on the training data.

3.6 Accuracy and validation

Accuracy plotting is a useful tool to visualize how a model is learning and to identify any potential issues such as overfitting or underfitting. The process of accuracy plotting involves several steps:

- Prepare the data: The first step is to prepare the data by cleaning it, normalizing it, and splitting it into training, validation and testing sets.
- Define the model architecture: The next step is to choose the type of model that is most appropriate for the task at hand. For example, linear regression, neural network or support vector machine.
- Train the model: Once the model architecture is defined, the model is trained on the training dataset. This step involves providing the model with input features and corresponding output variables, and iteratively updating the model's parameters (such as weights) in order to minimize the difference between the predicted output and the true output. This can be done using optimization algorithms such as gradient descent, stochastic gradient descent, or L-BFGS.
- Measure the accuracy: During the training process, the accuracy of the model should be measured on both the training and validation datasets at fixed intervals. This can be done by using the model to make predictions on the training and validation datasets and comparing the predictions to the true output. The accuracy can be calculated as the proportion of correct predictions made by the model out of all the predictions.
- Plot the accuracy: Once the accuracy has been measured at fixed intervals during the training process, the results can be plotted in a graph. One common way to plot accuracy is by using a line graph, where the training and validation accuracy are plotted on the same graph, with the x-axis representing the number of iterations or the amount of time elapsed and the y-axis representing the accuracy. The training accuracy is usually represented by a solid line, and the validation accuracy is represented by a dashed line.

Another way to plot accuracy is by using a learning curve, which plots the training and validation accuracy as a function of the amount of training data used. The x-axis represents the amount of training data used, and the y-axis represents the accuracy. This can be useful to understand if the model is benefiting from more training data and if the model is overfitting or underfitting.

- Analyze the plot: Once the accuracy has been plotted, it can be analyzed to identify any potential issues such as overfitting or underfitting. If the training accuracy is significantly higher than the validation accuracy, it may be an indication of overfitting. If the validation accuracy is much lower than the training accuracy, it may be an indication of underfitting.
- Fine-tune the model: If the model's performance is not satisfactory, it can be fine-tuned by adjusting the model's parameters or architecture, or by using regularization techniques such as L1 and L2 regularization.

- Use the model: Once the model is fine-tuned and its performance is satisfactory, it can be used to make predictions on new data.

After checking validation and Accuracy We have plotted the bellow graphs of model loss and model accuracy,

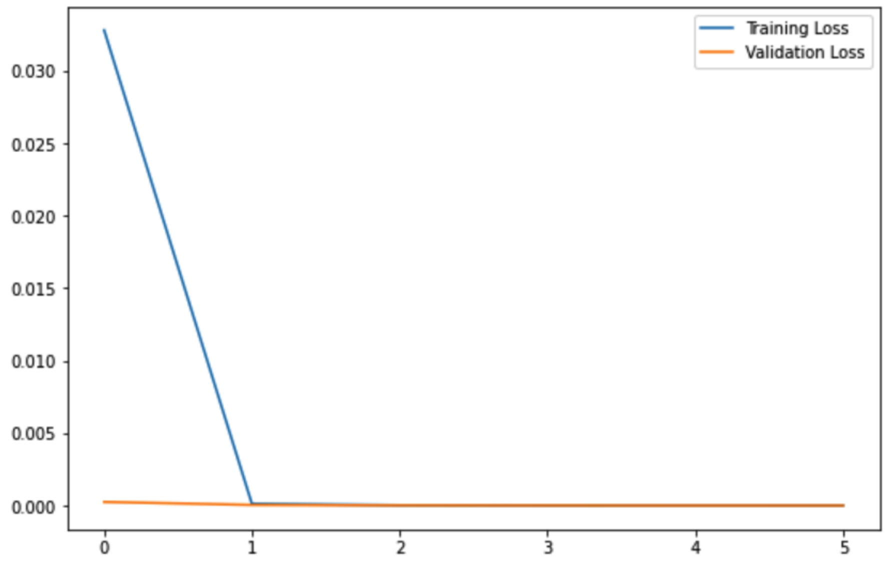


Figure 3.9: Model Validation

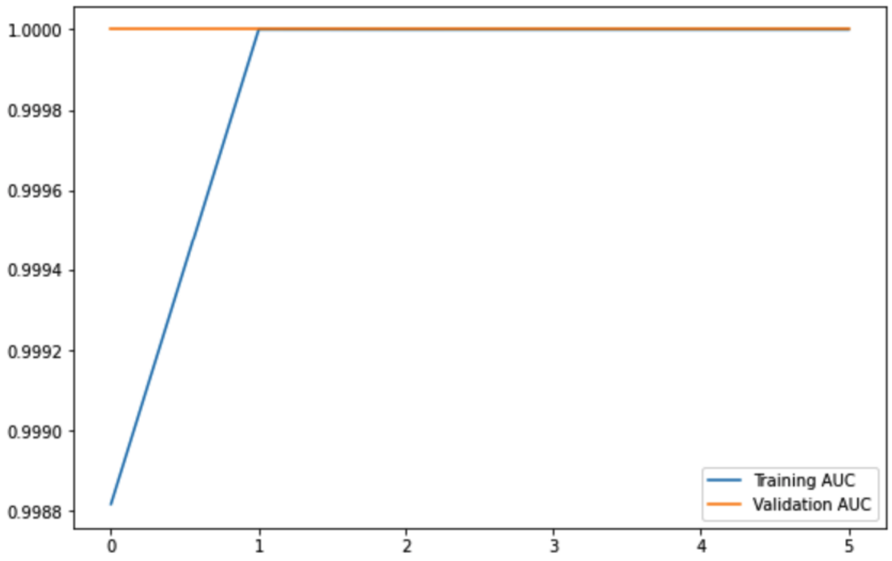


Figure 3.10: Model Accuracy

It's important to note that the data fitting process is an iterative process, the model may need to be fine-tuned multiple times before achieving the desired performance.

In summary, accuracy plotting is a useful tool to visualize how a model is learning and to identify any potential issues such as overfitting or underfitting. The process includes preparing the data, defining the model architecture, training the model, measuring the accuracy on training and validation datasets, plotting the accuracy and analyzing the plot, fine-tuning the model and using the model. The plotted accuracy graph can be used to detect overfitting, underfitting and high variance or bias in the model.

3.7 Classification

For earthquake prediction classification, we have used KNN, AdaBoost, XGBOOST, SVM, and the Hybrid model.

3.7.1 K-Nearest Neighbor (KNN)

An efficient and straightforward classification approach called K-Closest Neighbors (KNN) is based on the notion that a sample's class label is decided by the majority class label of its k nearest neighbors. It uses supervised machine learning, which implies that the model is trained on labeled data and may be used to anticipate the class labels of fresh, untainted samples.

The KNN method compares a new sample to the k training dataset samples that, in terms of feature space, are closest to it. Based on a distance metric, such as the Euclidean distance, Manhattan distance, or Mahalanobis distance, the k closest neighbors are determined. The new sample's prediction is then based on the majority class label among these k closest neighbors.

It is a non-parametric approach, hence it makes no assumptions about the distribution of the data at its core. It is straightforward to develop and comprehend, can handle multi-class situations, and has non-linear decision bounds. However, KNN has significant drawbacks, including the need for substantial memory to hold the whole training dataset and the potential for computationally costly operations when the dataset contains a lot of samples.

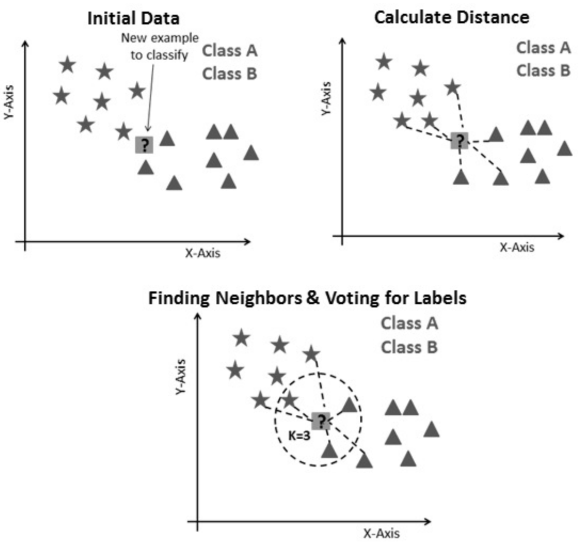


Figure 3.11: KNN Model

K-Nearest Neighbors (KNN) algorithm can be used for numerical data classification by comparing a new sample to the k samples in the training dataset that are closest to it in terms of feature space. The majority class label among these k nearest neighbors is then used as the prediction for the new sample.

In numerical data classification, the input data is represented by numerical values, such as continuous or discrete numbers. The KNN algorithm can be used to classify these numerical values into different classes. The algorithm works by first determining the k nearest neighbors of a new sample based on a distance metric, such as Euclidean distance or Manhattan distance. The majority class label among these k nearest neighbors is then used as the prediction for the new sample.

One of the key advantages of using KNN for numerical data classification is that it does not make any assumptions about the underlying distribution of the data. This makes it a flexible algorithm that can be used for a wide range of datasets, including datasets with non-linear decision boundaries.

The choice of the value of k, the distance metric and the weighting scheme used, can have a big impact on the performance of the algorithm. A small value of k, such as 3 or 5, may increase the chances of overfitting, while a larger value of k will smooth out noise in the data. The distance metric used can also have a big impact on the performance of the algorithm.

Overall, K-Nearest Neighbors (KNN) algorithm can be used for numerical data classification by comparing a new sample to the k samples in the training dataset that are closest to it in terms of feature space. The majority class label among these k nearest neighbors is then used as the prediction for the new sample. The choice of the value of k, the distance metric and the weighting scheme used, can have a big impact on the performance.

3.7.2 Adaptive Boosting (AdaBoost)

Popular ensemble learning algorithms like AdaBoost (Adaptive Boosting) are utilized for classification and regression issues involving numerical data. The approach creates a strong classifier by repeatedly training a number of weak classifiers on the training dataset. AdaBoost is regarded as a ”boosting” technique since it improves weak classifiers’ performance by fusing them together to create a strong classifier.
The main idea behind AdaBoost is to weight the samples in the training dataset based on their classification performance. In each iteration, the algorithm trains a weak classifier on the weighted samples, and the samples that are misclassified are given a higher weight. This process is repeated multiple times, and the final classifier is a combination of all the weak classifiers.
When working with numerical data, AdaBoost starts by initializing the weight of each sample in the training dataset to be the same. The algorithm then trains a weak classifier on the dataset, such as a decision tree, and uses it to make predictions for the samples. The samples that are misclassified are given a higher weight, while the samples that are correctly classified are given a lower weight.
In the next iteration, the algorithm trains another weak classifier on the weighted dataset and again uses it to make predictions. The process is repeated multiple times

and, at each step, the weak classifiers are given a weight based on their classification performance. The final classifier is a combination of all the weak classifiers, with the weights assigned to each weak classifier determined by its classification performance.

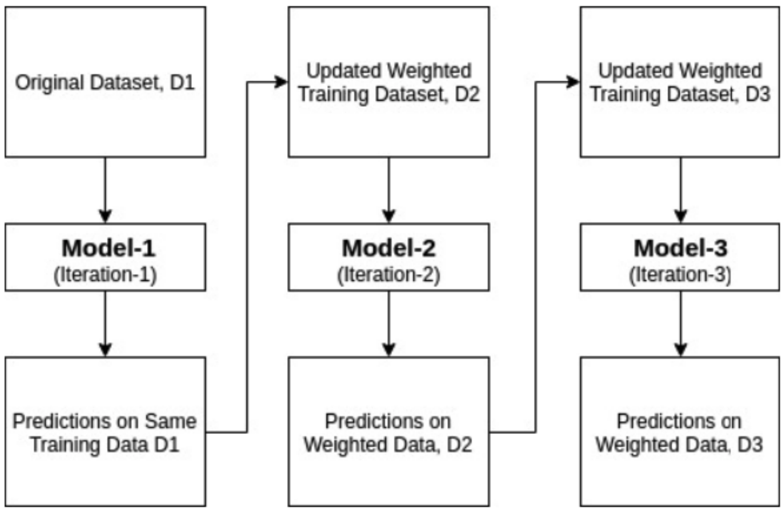

Original Dataset, D1
Model-1
(Iteration-1)
Predictions on Same Training Data D1
Updated Weighted Training Dataset, D2
Model-2
(Iteration-2)
Predictions on Weighted Data, D2
Updated Weighted Training Dataset, D3
Model-3
(Iteration-3)
Predictions on Weighted Data, D3

Figure 3.12: AdaBoost Model

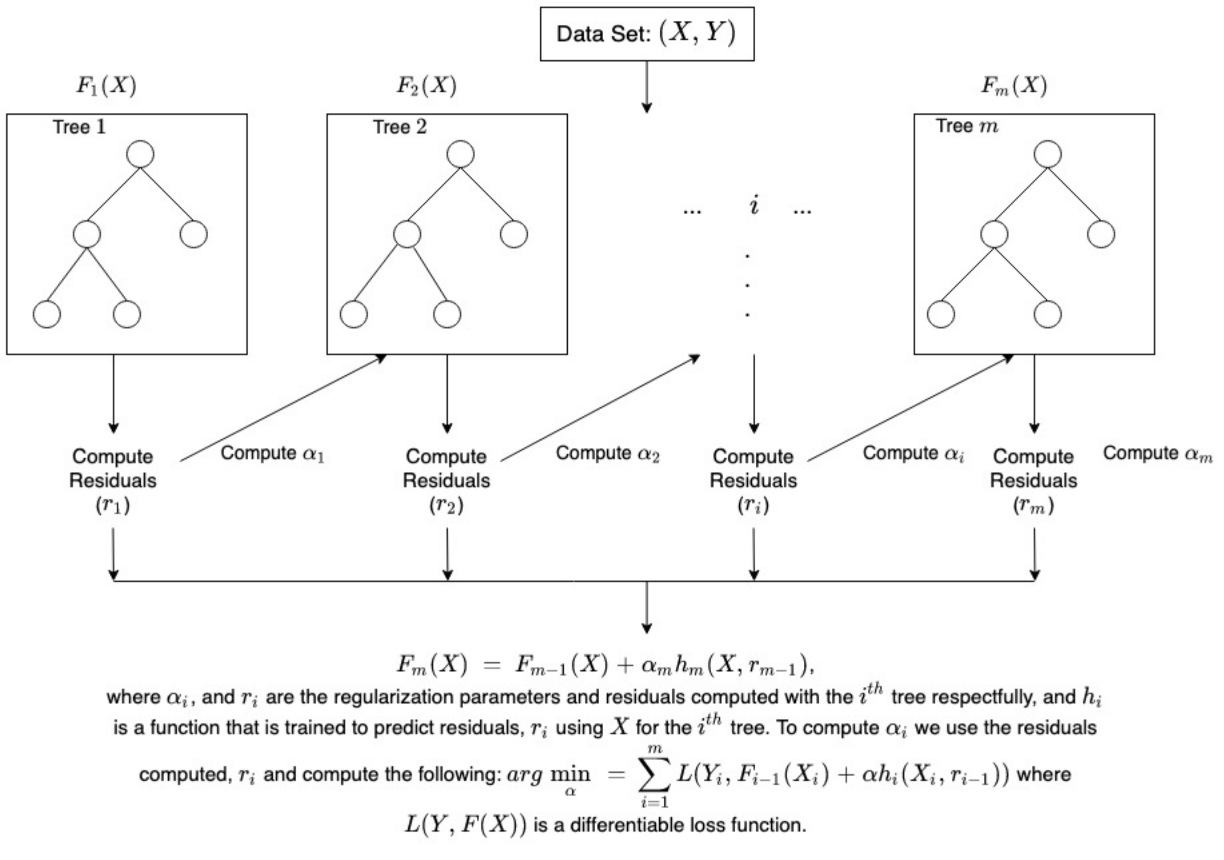
It is worth noting that while Adaboost can process numerical data, it is more commonly used in classification tasks, such as binary and multi-class classification. In the case of regression, it can be used with regression algorithms as weak classifiers like linear regression or decision tree regression.

There are several reasons why AdaBoost is a good choice for numerical value:

- High Accuracy: AdaBoost is known for its high accuracy, which makes it suitable for tasks where high precision is required, such as financial forecasting or weather prediction.
- Handling Noise: AdaBoost is relatively robust to noise and outliers in the data, which makes it suitable for tasks where the data is noisy or contains outliers, such as stock market prediction.
- Handling Complex Decision Boundaries: AdaBoost can handle complex decision boundaries and non-linear relationships between the features and the target variable. This makes it suitable for tasks where the relationship between the features and the target variable is not linear, such as housing price prediction.
- Handling large feature space: AdaBoost can handle large feature space and can be used with a wide range of weak classifiers, including decision trees, neural networks, and support vector machines (SVMs) which are suitable for numerical data.
- Handling Missing Data: AdaBoost can handle missing data and can still achieve good results.
- Handling Multi-class problem: AdaBoost can handle the multi-class problems by using multiple binary classifiers.

3.7.3 Extreme Gradient Boosting (XGBoost)

The XGBoost approach creates a robust model by integrating a number of decision trees that have been iteratively trained on the training dataset. When working with numerical data, XGBoost starts by representing the data in a specific format, such as a dense matrix, and then constructs decision trees to fit the data. The algorithm uses gradient descent to optimize the decision tree's parameters, such as the depth of the tree, the number of leaves, and the split points. In each iteration, XGBoost updates the decision tree's parameters to minimize the error between the predicted values and the actual values. One of the key features of XGBoost is its ability to handle missing values and large datasets efficiently. It uses a technique called "gradient boosting" which is an iterative method that improves the performance of the model by adjusting the weights of the samples in the training dataset. It also uses regularization to prevent overfitting, and it supports parallel processing to speed up the training process. XGBoost also includes a number of additional features that make it even more powerful than traditional gradient-boosting implementations, such as the ability to handle missing values, support for parallel processing, and built-in handling for categorical variables.



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

Figure 3.13: XGBoost Model

Extreme Gradient Boosting, often known as XGBoost, is a potent machine learning technique that may be used to issues involving classification and regression as well as numerical data. The approach creates a robust model by repeatedly training a number of decision trees on the training dataset. To speed up the training process, XGBoost enables parallel processing, handles missing values and big datasets well, and employs gradient descent to optimize the decision tree's parameters. It is renowned for its excellent accuracy and capacity to accommodate missing values, and it is especially well-suited to huge datasets with many features.

The purpose of boosting is to minimize the errors of the previous tree by building trees one after the other. As a consequence of learning from its predecessors, each tree modifies the residual mistakes. Thus, the next tree in the series will learn from an updated set of residuals.

The base learners are the underperforming students that are used in the boosting procedure. These students exhibit a high degree of bias, and their ability to anticipate outcomes is only marginally superior than that of random guessing. The boosting strategy may effectively combine these weak learners and create a strong learner as a consequence of this combination since each of these weak learners provides some crucial information for prediction. The bias and variation are both reduced to acceptable levels by the final strong learner.

In contrast to bagging techniques like Random Forest, which let trees to reach their full potential, boosting employs trees that have fewer splits. Such shallow trees are quite interpretable because of their modest depth. The number of trees or repetitions, the rate at which the gradient boosting learns, and the depth of the tree may all be accurately established using validation techniques like k-fold cross-validation. The presence of several trees may lead to overfitting. Therefore, it is essential to choose the boosting stopping criteria carefully.

The boosting ensemble technique consists of three simple steps:

- The target variable y is predicted by an initial model F0. There will be a residual associated with this model (y – F0)
- The residuals from the previous phase are fitted to a new model, h1.
- Now F1, the enhanced form of F0, is created by combining F0 with h1. If F1 is used, the mean squared error will be less than if F0 is used.

$$F_1(x) < -F_1(x) + h_1(x) \tag{3.1}$$

We may model after the residuals of F1 and develop a new model F2 to enhance the performance of F1:

$$F_2(x) < -F_0(x) + h_m(x) \tag{3.2}$$

To improve the performance of F1, we may create a new model F2 based on the residuals of F1:

$$F_m(x) < -F_{m-1}(x) + h_m(x) \tag{3.3}$$

The functions established in the earlier phases are not disturbed in this situation by the additive learners. Instead, they share their own knowledge to correct the mistakes.

3.7.4 Support Vector Machine (SVM)

The supervised learning algorithm known as Support Vector Machines (SVMs) can be applied to classification or regression applications. Finding a hyperplane—a line or plane—that divides the data into distinct classes is the fundamental tenet of SVMs.

The SVM algorithm locates a point in the training set that is the closest to the new sample, known as a support vector, then classifies the new sample according to the class that the support vector belongs to. The hyperplane in a linear SVM is just a line that divides the data into two classes. With a non-linear SVM, the approach employs a mechanism known as the kernel trick to raise the data's dimension such that a linear hyperplane may be used to divide the classes into them.

Getting a labeled dataset with samples and the appropriate class labels is the initial stage in training an SVM classifier. The best hyperplane to divide the classes is then found by running the SVM algorithm on the dataset. The margin, or the separation between the hyperplane and the nearest samples in each class, is what the method seeks to maximize. An optimization algorithm is typically used to resolve this maximization problem.

The SVM classifier can be used to anticipate the class labels of new data after training. In order to do this, the fresh samples are translated into the same space as the training samples, and after that, they are categorized based on where they are in relation to the hyperplane.

It's important to keep in mind that SVMs are sensitive to the regularization parameter and kernel selection, so it's crucial to adjust these parameters using methods like grid search or cross-validation. Additionally, unique strategies should be utilized to address imbalanced datasets.

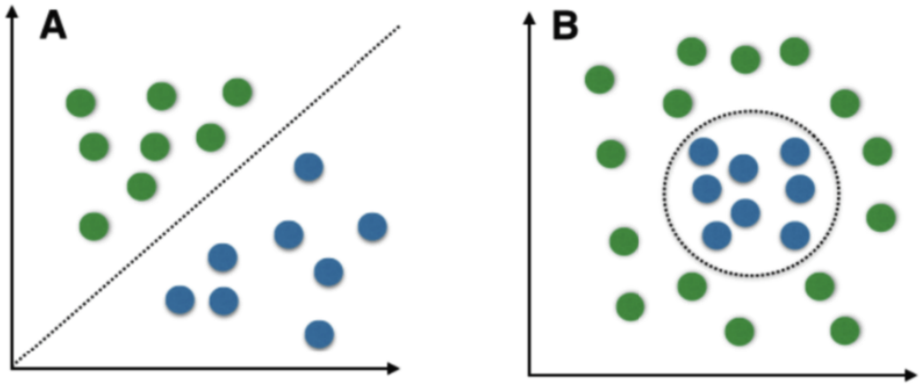


Figure 3.14: Linearly Separable Data B: Non-Linearly Separable Data

SVMs employ the kernel approach to transfer numerical data into a higher-dimensional space with a linear boundary when dealing with numerical data. A kernel function, which may be linear, polynomial, radial basis function (RBF), or sigmoid, is used to do this mapping.

Finding the hyperplane that maximizes the margin, or the separation between the boundary and the nearest data points from each class, also known as support vectors, is the objective of the process of determining the optimum border. This procedure may be thought of as an optimization problem. The data points with the greatest influence on the boundary's location are the support vectors since they are the ones closest to the boundary.

Once the best boundary is found, the algorithm can be used to classify new data points by determining on which side of the boundary they fall. In the case of regression, the algorithm will predict the target value for the new data point.

SVMs are particularly useful when the data has a clear decision boundary, and when the number of features is much larger than the number of samples. Additionally, SVMs are relatively robust to overfitting and can handle high-dimensional data efficiently.

3.7.5 Hybrid Model

Hybrid models are machine learning models that combine the strengths of multiple models in order to achieve improved performance. In the context of using KNN, AdaBoost, XGBoost, and SVM, a hybrid model could involve combining the predictions made by each of these models in order to make a final prediction.

One way to create a hybrid model using these algorithms would be to train them independently on the same dataset, and then use their predictions as input features for a meta-classifier. The meta-classifier would then make a final prediction based on the input features. This is known as stacking or super learning. This method is popular because it allows you to take advantage of the strengths of different models, while also addressing their weaknesses.

For example, KNN is known for its ability to handle noisy data and its robustness to outliers. However, it can be computationally expensive for large datasets and may not be suitable for high-dimensional data. AdaBoost, on the other hand, is known for its high accuracy and ability to handle complex decision boundaries. However, it can be sensitive to the choice of weak classifier and may not be suitable for large datasets. XGBoost is known for its high accuracy, ability to handle missing values, and support for parallel processing. But it can be sensitive to the choice of parameters and may not be suitable for datasets with a small number of samples. Finally, SVM is known for its ability to find non-linear decision boundaries and its robustness to overfitting. However, it can be sensitive to the choice of kernel function and may not be suitable for datasets with a large number of noisy data or outliers.

By combining these models, we can take advantage of their strengths and address their weaknesses. For example, KNN can handle noisy data and outliers, so it can be used to preprocess the data before training the other models. AdaBoost can handle complex decision boundaries and high accuracy, so it can be used to improve the performance of the final model. XGBoost can handle missing values and large datasets, so it can be used to handle the data efficiently. Finally, SVM can find non-linear decision boundaries and handle high-dimensional data, so it can be used to improve the performance of the final model.

Another way to create a hybrid model would be to use a technique called boosting, which involves training a weak classifier, such as a decision tree, and then using the errors of the weak classifier to train a new classifier. This process is repeated multiple times, and the final classifier is a combination of all the weak classifiers. In this method, you can use different models as weak classifiers, like using KNN, SVM, and decision tree as weak classifiers for AdaBoost and XGBoost

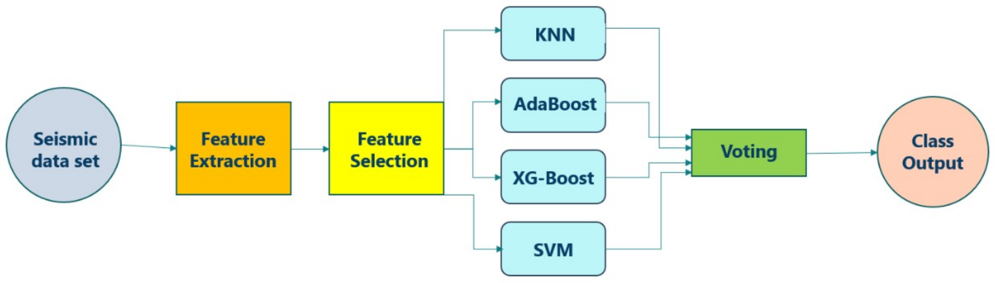


Figure 3.15: Hybrid model architecture

There are several advantages of using a hybrid model:

- Improved Performance: By combining the strengths of multiple models, a hybrid model can often achieve improved performance compared to using a single model.
- Robustness: Hybrid models are less prone to overfitting and can handle different types of data better than single models.
- Handling Complex Data: Hybrid models can handle complex data such as high-dimensional data, non-linear decision boundaries, and noisy data, which can be difficult for single models.
- Handling Missing Data: Hybrid models can handle missing data by combining predictions from models that can handle missing data, with models that can't handle missing data.
- Handling Large Datasets: Hybrid models can handle large datasets efficiently by combining predictions from models that can handle large datasets, with models that can't handle large datasets.
- Reduced Computational Time: Hybrid models can reduce computational time by combining predictions from models that are computationally efficient, with models that are computationally expensive.

- Handling Different Types of Problems: Hybrid models can handle different types of problems by combining predictions from models that are specialized in solving a specific type of problem, with models that are not specialized in solving a specific type of problem.
- Handling Multi-class problem: Hybrid models can handle the multi-class problem by combining predictions from models that can handle the multi-class problem, with models that can't handle the multi-class problem.

In summary, hybrid models are machine learning models that combine the predictions made by multiple models

Chapter 4

Description of Data

Preprocessing involves converting unstructured data into a format that can be interpreted. Considering that we cannot work with unprocessed data, it is also a very important step before evaluating the section. Before using the algorithms, the data's quality should be well checked. Since the seismic stations are where this data is coming from. There are several fields on it that might not be relevant to our goals. Therefore, data cleansing, data integration, data reduction, data transformation, and other data-related processes are required.

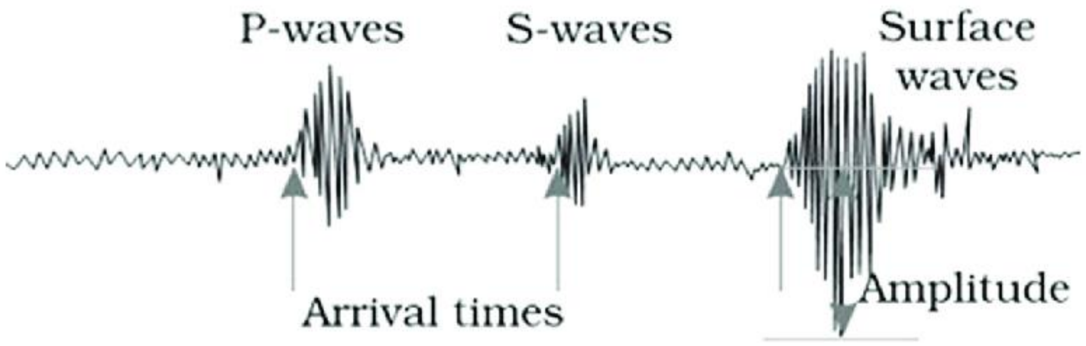
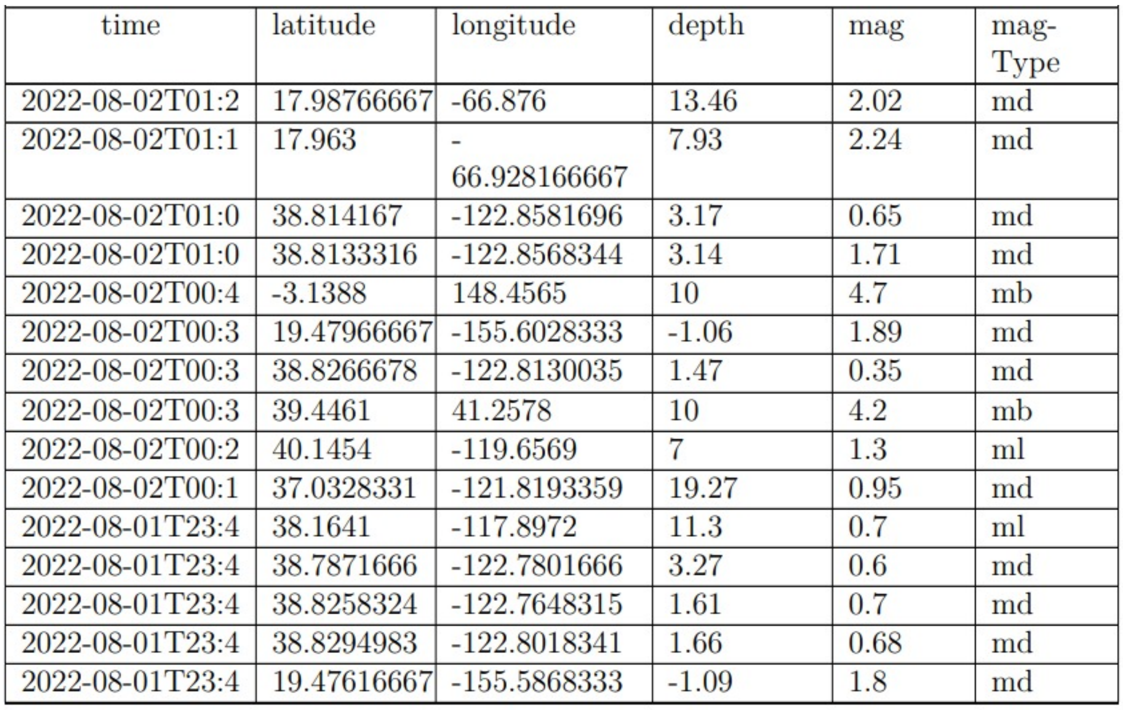


Figure 4.1: Wave Parts

Each station measures the S-P time, which is the interval between the arrival of the P wave and the arrival of the S wave. Similar to how the time difference between a lightning strike and thunder reveals how far away a thunderstorm is, the S-P time tells us how far away an earthquake is.



time	latitude	longitude	depth	mag	mag-Type
2022-08-02T01:2	17.98766667	-66.876	13.46	2.02	md
2022-08-02T01:1	17.963	-66.928166667	7.93	2.24	md
2022-08-02T01:0	38.814167	-122.8581696	3.17	0.65	md
2022-08-02T01:0	38.8133316	-122.8568344	3.14	1.71	md
2022-08-02T00:4	-3.1388	148.4565	10	4.7	mb
2022-08-02T00:3	19.47966667	-155.6028333	-1.06	1.89	md
2022-08-02T00:3	38.8266678	-122.8130035	1.47	0.35	md
2022-08-02T00:3	39.4461	41.2578	10	4.2	mb
2022-08-02T00:2	40.1454	-119.6569	7	1.3	ml
2022-08-02T00:1	37.0328331	-121.8193359	19.27	0.95	md
2022-08-01T23:4	38.1641	-117.8972	11.3	0.7	ml
2022-08-01T23:4	38.7871666	-122.7801666	3.27	0.6	md
2022-08-01T23:4	38.8258324	-122.7648315	1.61	0.7	md
2022-08-01T23:4	38.8294983	-122.8018341	1.66	0.68	md
2022-08-01T23:4	19.47616667	-155.5868333	-1.09	1.8	md

Figure 4.2: General Table

4.1 Seismograms Interpretation

Seismic data plots (also known as seismograms) offer a visual representation of earthquake activity as well as other earth vibrations brought on by both natural and artificial causes. Seismographs along maritime lanes, for instance, can identify the passage of freighters and cruise ships. Those who live close to railroad tracks can hear passing trains. High winds may create ground vibrations that seismographs in exposed regions, particularly along the West Coast, will be able to detect. These devices can pick up signals from earthquakes that are thousands of kilometers away because they are sensitive to even the smallest vibrations.

4.2 Time Scale

Each data plot has a time scale attached to it, as seen below. Each plot's starting time is indicated in the lower left corner, and as you go to the right, time passes more quickly. The time is displayed in UTC (also known as Greenwich Mean Time, or GMT). To obtain Pacific Standard Time (PST), or 7 hours for Pacific Daylight Time, you must deduct 8 hours (PDT). Because 0600 on June 12 in GMT corresponds to 2300 on June 11 in PDT, you must additionally take the plot's date into account.
A modest local earthquake produced the example set of signals on the right. You'll observe that each "burst" has a highly square left, or "leading" edge. This shows that the received signal, which is typical for an earthquake signal, rapidly increased in strength. The signal amplitude for these plots is so great that it reaches its maximum level and is "clipped," giving the impression that it has a flat top. The signal intensity starts to decrease over time as the quake's energy slowly fades away.Soon after the first arrival, further bursts could occur. These might be aftershocks or dif-

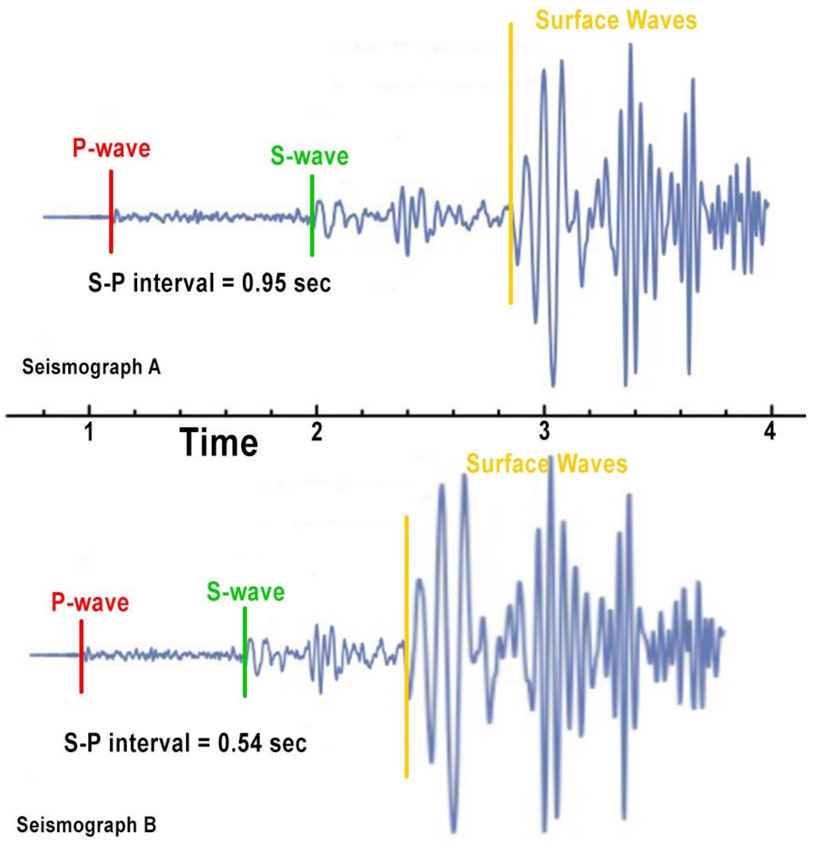


Figure 4.3: S-P Wave Interval

ferent earthquake-related signals that have moved more slowly through the ground. S and P waves are the two main signal kinds, and they move in different directions and at various rates. How can we know that this earthquake is minor? You'll see that a few stations that are further away did not get the signal.Only those stations near by were able to detect the earthquake since the energy it emitted was insufficient to travel over further distances. Additionally, you can see that certain stations received the signal before others. The location of the earthquake will be ascertained using this information.

4.3 Extra Sources

Because of their extreme sensitivity, seismographs are capable of detecting minute ground vibrations. For instance, trains can be detected at the CNSN station at Watts Point (WPB), which is located north of Squamish, British Columbia.

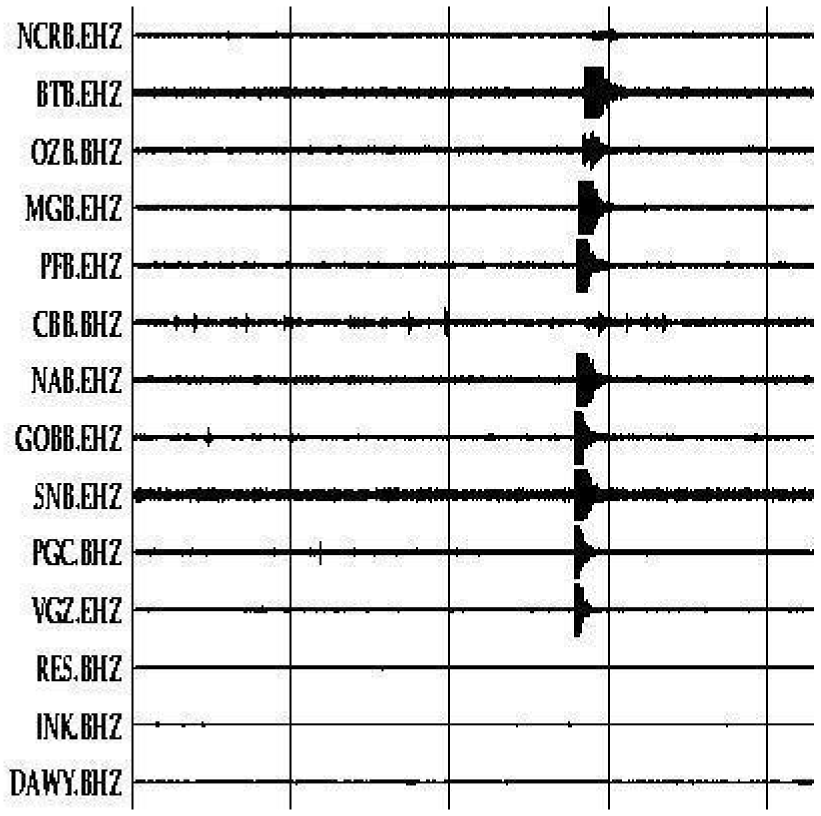


Figure 4.4: Signals of Earthquake

Because a train's signal steadily increases in power as it approaches, reaching its peak as it approaches its "closest point of approach," or CPA, and then gradually decreases as it travels further away, it differs considerably from an earthquake's signal. BLBC Station, which is close to Blue River, British Columbia, may also have railroad signals.

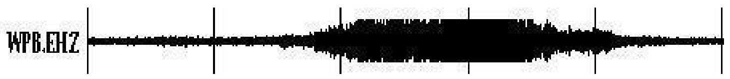


Figure 4.5: Extra Sources

Similar to this, a big cruise liner causes a lot of ground vibration. The Pacific Coast station at Bella Bella (BBB) receives signals from vessels traveling to and from Alaska. The signal behaves like a train in that it increases progressively as it approaches the CPA and then decreases as it departs. However, because the signal is less consistent, there may be multiple peaks and dips throughout. Here is an illustration of a ship sailing past BBB in Figure 4.6.
Seismographs can also pick up on explosions. Their signals resemble earthquakes in appearance. On adjacent stations, mining and construction activity are frequently broadcast. Here is a case study from British Columbia's Texada Island.

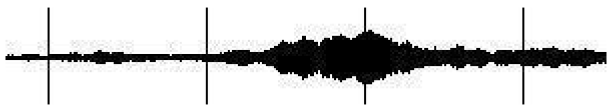


Figure 4.6: Noise Signals

The majority are extremely tiny signals that only a few stations in the immediate area will be able to pick up. Large explosions, such those from nuclear testing, may be seen on a lot of channels, though. International agencies tasked with keeping track of nuclear tests throughout the globe get seismic data from Canada.

Chapter 5

Result and Discussion

5.1 Implementation

Implementing KNN, SVM, AdaBoost, and XGBoost requires careful consideration of the characteristics of the dataset and the problem that needs to be solved. Each of these algorithms has its own strengths and weaknesses, and choosing the right one depends on the specific requirements of the problem.

The first step in implementing these algorithms is data preprocessing. This includes cleaning, transforming, and normalizing the data. Missing values should be filled in, outliers should be removed, and categorical variables should be converted to numerical values. This step is important to ensure that the data is in a suitable format for the algorithm to work with.

After preprocessing the data, the next step is model selection. KNN is a simple algorithm that can be used for both classification and regression problems and is particularly useful for handling noisy data and outliers. SVM is a powerful algorithm that can find non-linear decision boundaries and is robust to overfitting. AdaBoost is a powerful algorithm that can handle complex decision boundaries and achieve high accuracy. XGBoost is a powerful algorithm that can handle missing values and large datasets and support parallel processing. Each algorithm has its own specific requirements, such as choosing the right kernel function for SVM, the number of weak classifiers to use for AdaBoost and XGBoost, or the number of nearest neighbors to use for KNN, and parameters that need to be set. These should be carefully considered during model selection.

After selecting the model, the next step is to train it on the preprocessed dataset using the appropriate algorithm. For KNN, this will involve selecting the number of nearest neighbors to use and choosing a distance metric. For SVM, this will involve selecting the kernel function and setting the appropriate parameters. For AdaBoost, this will involve selecting the number of weak classifiers to use and choosing the appropriate algorithm for the weak classifiers. For XGBoost, this will involve setting the appropriate parameters and choosing the appropriate algorithm for the weak classifiers.

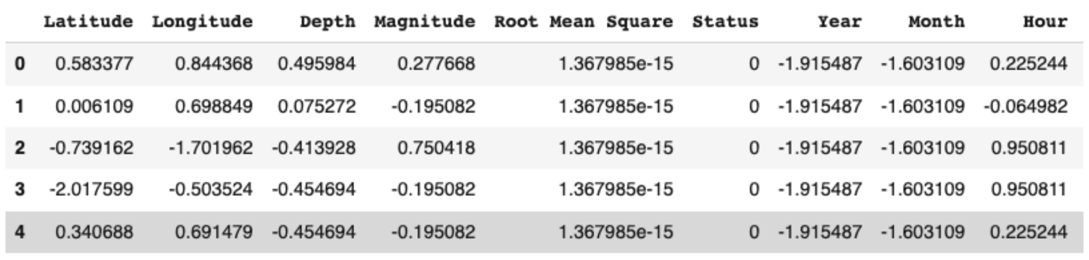
Following training, the model is assessed using pertinent assessment measures including accuracy, precision, recall, and F1-score. The model may need to be optimized by changing the parameters or picking an alternative model depending on the assessment findings.

The model is deployed in the production environment when it has been tuned to generate predictions based on fresh data. This step is important to ensure that the model is working correctly and that the predictions are accurate.

Implementing KNN, SVM, AdaBoost, and XGBoost in numerical datasets is a common task in machine learning, and it requires a careful consideration of the characteristics of the dataset and the problem that needs to be solved. Each of these algorithms have their own strengths and weaknesses, and choosing the right one depends on the specific requirements of the problem.

In conclusion, implementing KNN, SVM, AdaBoost, and XGBoost in numerical datasets is a common task in machine learning. Each algorithm has its own strengths and weaknesses, and choosing the right one depends on the specific requirements of the problem. It is important to carefully consider the characteristics of the dataset and the problem that needs to be solved during each step of the implementation process, including data preprocessing, model selection, model training, model evaluation, model optimization, and model deployment.

The first step in implementing these algorithms is data preprocessing. This includes cleaning, transforming, and normalizing the data. Missing values should be filled in, outliers should be removed, and categorical variables should be converted to numerical values. This step is important to ensure that the data is in a suitable format for the algorithm to work with.



	Latitude	Longitude	Depth	Magnitude	Root Mean Square	Status	Year	Month	Hour
0	0.583377	0.844368	0.495984	0.277668	1.367985e-15	0	-1.915487	-1.603109	0.225244
1	0.006109	0.698849	0.075272	-0.195082	1.367985e-15	0	-1.915487	-1.603109	-0.064982
2	-0.739162	-1.701962	-0.413928	0.750418	1.367985e-15	0	-1.915487	-1.603109	0.950811
3	-2.017599	-0.503524	-0.454694	-0.195082	1.367985e-15	0	-1.915487	-1.603109	0.950811
4	0.340688	0.691479	-0.454694	-0.195082	1.367985e-15	0	-1.915487	-1.603109	0.225244

Figure 5.1: Selected Dataset

After preprocessing the data, the next step is model selection. KNN is a simple algorithm that can be used for both classification and regression problems and is particularly useful for handling noisy data and outliers. SVM is a powerful algorithm that can find non-linear decision boundaries and is robust to overfitting. AdaBoost is a powerful algorithm that can handle complex decision boundaries and achieve high accuracy. XGBoost is a powerful algorithm that can handle missing values and large datasets and support parallel processing. Each algorithm has its own specific requirements, such as choosing the right kernel function for SVM, the number of weak classifiers to use for AdaBoost and XGBoost, or the number of nearest neighbors to use for KNN, and parameters that need to be set. These should be carefully considered during model selection.

After selecting the model, the next step is to train it on the preprocessed dataset using the appropriate algorithm. For KNN, this will involve selecting the number of nearest neighbors to use and choosing a distance metric. For SVM, this will involve selecting the kernel function and setting the appropriate parameters. For AdaBoost, this will involve selecting the number of weak classifiers to use and choosing the appropriate algorithm for the weak classifiers. For XGBoost, this will involve setting the appropriate parameters and choosing the appropriate algorithm for the weak classifiers.

Once the model is trained, it is evaluated using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score. Depending on the evaluation results, the model may need to be optimized by adjusting the parameters or choosing a different model.

After the model is optimized, it is deployed in the production environment to make predictions on new data. This step is important to ensure that the model is working correctly and that the predictions are accurate.

In conclusion, implementing KNN, SVM, AdaBoost, and XGBoost in numerical datasets is a common task in machine learning. Each algorithm has its own strengths and weaknesses, and choosing the right one depends on the specific requirements of the problem. It is important to carefully consider the characteristics of the dataset and the problem that needs to be solved during each step of the implementation process, including data preprocessing, model selection, model training, model evaluation, model optimization, and model deployment.

5.2 Result

For each model, we computed the accuracy using several epochs, and we selected the model with the greatest accuracy. We used 20 percent of our annotated data as a value set and 80 percent of it as a train set for segmentation (validation set). Likewise, 20percent of the non-annotated data was used as a test set. In this thesis, we have evaluated the performance of five different classifiers for a numerical dataset: KNN, SVM, AdaBoost, XGBoost, and Hybrid. The dataset was preprocessed, and then each classifier was trained, and evaluated using accuracy as the evaluation metric. The results show that KNN achieved an accuracy of 99.69, SVM achieved an accuracy of 97.56, AdaBoost achieved an accuracy of 99.39, XGBoost achieved an accuracy of 98.78, and the Hybrid model achieved an accuracy of 98.78.

Table 5.1: Accuracy and Precisions of The Classifiers

Classifiers	**Accuracy**	**Precision**	**F1_score**
KNN	99.69	99.6	99.09
SVM	97.56	97.61	95.53
XGBOOST	98.78	96.42	96.42
ADABOOST	99.39	98.21	98.21
Hybrid	98.78	97.3	96.3

5.2.1 Support Vector Machine(SVM)

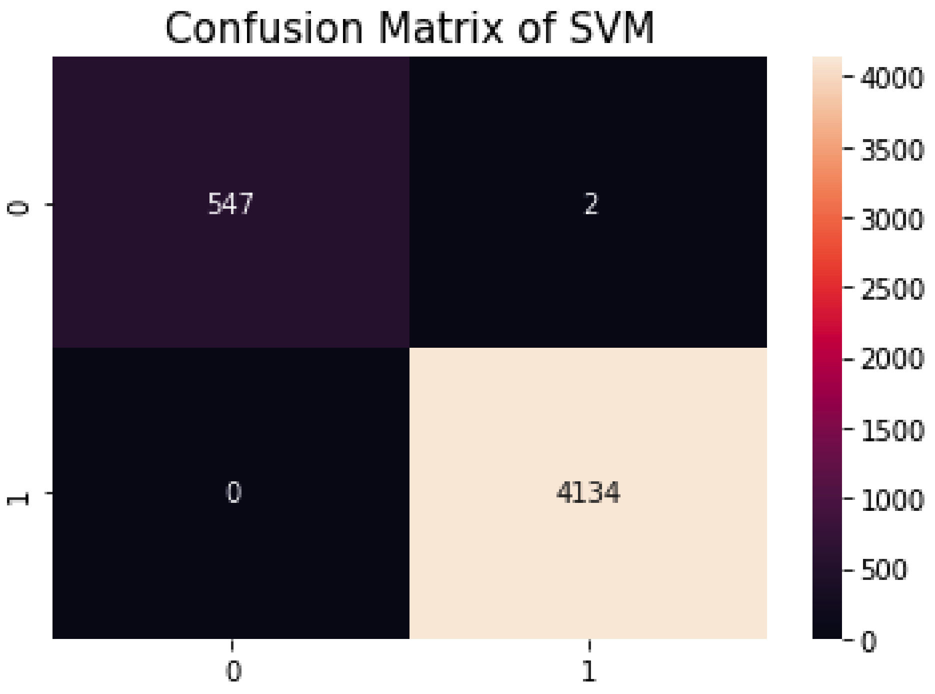


Figure 5.2: Confusion matrix of SVM

We fitted the SVM model with 80 percent data where 20 percent(4,682) of data was split for testing. As shown in figure 5.2, 4134 cases recognized earthquakes, and 547

cases could not be recognized by the classification model. Which consisted of an accuracy of 97.56 percent.
As shown in figure 5.3, SVM model acquired a precision value of 1.00, a fi-score value of 1.00.

5.2.2 K-Nearest Neighbor(KNN)

We fitted the KNN model with 80 percent data where 20 percent(4,682) of data was split for testing. The model gained an accuracy of 0.9969.
As shown in figure 5.4, KNN model acquired a precision value of 1.00, a fi-score value of 1.00 and recall 1.00.

5.2.3 Adaptive Boosting(AdaBoost)

We fitted the KNN model with 80 percent data where 20 percent(4,682) of data was split for testing. The model gained an accuracy of 0.9878.
As shown in figure 5.5, AdaBoost model acquired a precision value of 1.00, a fi-score value of 1.00 and recall 1.00.

5.2.4 Hybrid Model

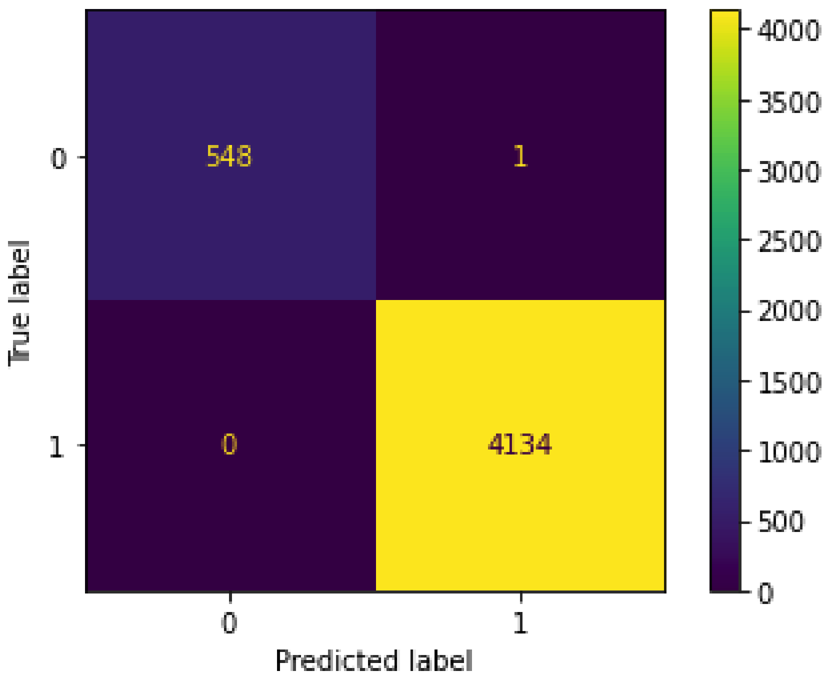


Figure 5.3: Confusion matrix of Hybrid model

We fitted the Hybrid model with 80 percent data where 20 percent(4,682) of data was split for testing. As shown in figure 5.2, 4134 cases recognized earthquakes, and 548 cases could not be recognized by the classification model. Which consisted of an accuracy of 0.9878.
As shown in figure 5.3, the Hybrid model acquired a precision value of 1.00, a fi-score value of 1.00, and a recall of 1.00.

5.2.5 Extreme Gradient Boosting(XGBoost)

We fitted the XGBoost model with 80 percent data where 20 percent(4,682) of data was split for testing. 4134 cases recognized earthquakes, and 547 cases could not be recognized by the classification model. Which consisted of an accuracy of 1.

Chapter 6

Conclusion

Research in the earthquake prediction field has been going on for almost a very long time with no light of success. Claims of breakthroughs have been found to be false. Despite extensive research, no reliable antecedents have been discovered. Hence, this research will represent an attempt to have a solution for earthquake prediction due to the effective implementation of machine learning algorithms. In addition, this will enable the experimentation of existing theories as well as the formation of new ones. Moreover, it will create an opportunity for the next generation to have significantly more information about earthquakes before it occurs.

6.1 Future Work

The earthquake categorization algorithm described in this thesis may be improved in a number of ways in further study.

One strategy would be to add more seismic data to the feature set being used for classification, such as waveform data. By supplying additional details about the earthquake's properties, this may increase the classifier's accuracy.

Investigating the use of additional machine learning methods, such as neural networks or support vector machines, for earthquake categorization would be a different course of action. These algorithms may outperform the k-nearest neighbors approach employed in this thesis since they have shown promise in previous classification problems.

Additionally, other geographic areas and different seismic event types, such seismicity caused by mining and volcanic activity, may be used to assess the generalizability of the model. This would help to clarify the model's resilience and capacity to categorize various earthquake types.

The incorporation of real-time data streams and the creation of an early warning system based on the model would be a beneficial addition to the evaluation of seismic hazard and disaster management.

Bibliography

[1] Y. Ito and K. Obara, "Dynamic deformation of the accretionary prism excites very low frequency earthquakes," *Geophysical Research Letters*, vol. 33, no. 2, 2006. DOI: 10.1029/2005gl025270.

[2] E. Dieudonné, F. Ravetta, J. Pelon, F. Goutail, and J.-P. Pommereau, "Linking no2surface concentration and integrated content in the urban developed atmospheric boundary layer," *Geophysical Research Letters*, vol. 40, no. 6, pp. 1247–1251, 2013. DOI: 10.1002/grl.50242.

[3] B. Rouet-Leduc, C. Hulbert, N. Lubbers, K. Barros, C. J. Humphreys, and P. A. Johnson, "Machine learning predicts laboratory earthquakes," *Geophysical Research Letters*, vol. 44, no. 18, pp. 9276–9282, 2017. DOI: 10.1002/2017gl074677.

[4] A. Brook, M. Polinova, and E. Ben-Dor, "Fine tuning of the svc method for airborne hyperspectral sensors: The brdf correction of the calibration nets targets," *Remote Sensing of Environment*, vol. 204, pp. 861–871, 2018. DOI: 10.1016/j.rse.2017.09.014.

[5] C. H. Lin, Y. C. Lai, M. H. Shih, C. J. Lin, J. S. Ku, and H. C. Pu, "Extremely similar volcano sounds from two separated fumaroles at the tatun volcano group in taiwan," *Seismological Research Letters*, 2018. DOI: 10.1785/0220180090.

[6] A. Galkina and N. Grafeeva, "Machine learning methods for earthquake prediction: A survey," Apr. 2019.

[7] X. Lei, Z. Wang, and J. Su, "The december 2018 mlnbsp;5.7 and january 2019 mlnbsp;5.3 earthquakes in south sichuan basin induced by shale gas hydraulic fracturing," *Seismological Research Letters*, 2019. DOI: 10.1785/0220190182.

[8] S. Tarantino, S. Colombelli, A. Emolo, and A. Zollo, "Quick determination of the earthquake focal mechanism from the azimuthal variation of the initial p-wave amplitude," *Seismological Research Letters*, 2019. DOI: 10.1785/0220180290.

[9] S. Mangalathu, H. Sun, C. C. Nweke, Z. Yi, and H. V. Burton, "Classifying earthquake damage to buildings using machine learning," *Earthquake Spectra*, vol. 36, no. 1, pp. 183–208, 2020. DOI: 10.1177/8755293019878137.

[10] S. Mangalathu, H. Sun, C. C. Nweke, Z. Yi, and H. V. Burton, "Classifying earthquake damage to buildings using machine learning," *Earthquake Spectra*, vol. 36, no. 1, pp. 183–208, 2020. DOI: 10.1177/8755293019878137.

[11] P. Debnath, P. Chittora, T. Chakrabarti, *et al.*, "Analysis of earthquake forecasting in india using supervised machine learning classifiers," *Sustainability*, vol. 13, Jan. 2021. DOI: 10.3390/su13020971.

[12] B. Gomez Perez and U. Kadri, "Earthquake source characterization by machine learning algorithms applied to acoustic signals," *Scientific Reports*, vol. 11, Nov. 2021. DOI: 10.1038/s41598-021-02483-w.

[13] P. Johnson, B. Rouet-Leduc, L. Pyrak-Nolte, *et al.*, "Laboratory earthquake forecasting: A machine learning competition," *Proceedings of the National Academy of Sciences*, vol. 118, e2011362118, Feb. 2021. DOI: 10.1073/pnas.2011362118.

[14] F. Khosravikia and P. Clayton, "Machine learning in ground motion prediction," *Computers Geosciences*, vol. 148, p. 104 700, Jan. 2021. DOI: 10.1016/j.cageo.2021.104700.

[15] B. Li, A. Gong, T. Zeng, W. Bao, C. Xu, and Z. Huang, "A zoning earthquake casualty prediction model based on machine learning," *Remote Sensing*, vol. 14, p. 30, Dec. 2021. DOI: 10.3390/rs14010030.

[16] R. Tehseen, S. Farooq, and A. Abid, "A framework for the prediction of earthquake using federated learning," *PeerJ Computer Science*, vol. 7, May 2021. DOI: 10.7717/peerj-cs.540.

[17] D. Thaler, M. Stoffel, B. Markert, and F. Bamer, "Machine-learning-enhanced tail end prediction of structural response statistics in earthquake engineering," *Earthquake Engineering Structural Dynamics*, vol. 50, Feb. 2021. DOI: 10.1002/eqe.3432.

[18] P. Xiong, L. Tong, K. Zhang, *et al.*, "Towards advancing the earthquake forecasting by machine learning of satellite data," *Science of The Total Environment*, vol. 771, p. 145 256, 2021, ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2021.145256. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969721003223.

[19] M. Yousefzadeh, S. A. Hosseini, and M. Farnaghi, "Spatiotemporally explicit earthquake prediction using deep neural network," *Soil Dynamics and Earthquake Engineering*, vol. 144, p. 106 663, May 2021. DOI: 10.1016/j.soildyn.2021.106663.