# Earthquake Prediction Model Using Python

## *TEAM MEMBER*

963521106025-BRAMUEL NIJOE GS

**Phase 3: Document Submission**

**Project Title: Earthquake Prediction**

**Phase 3: Development Part 1**

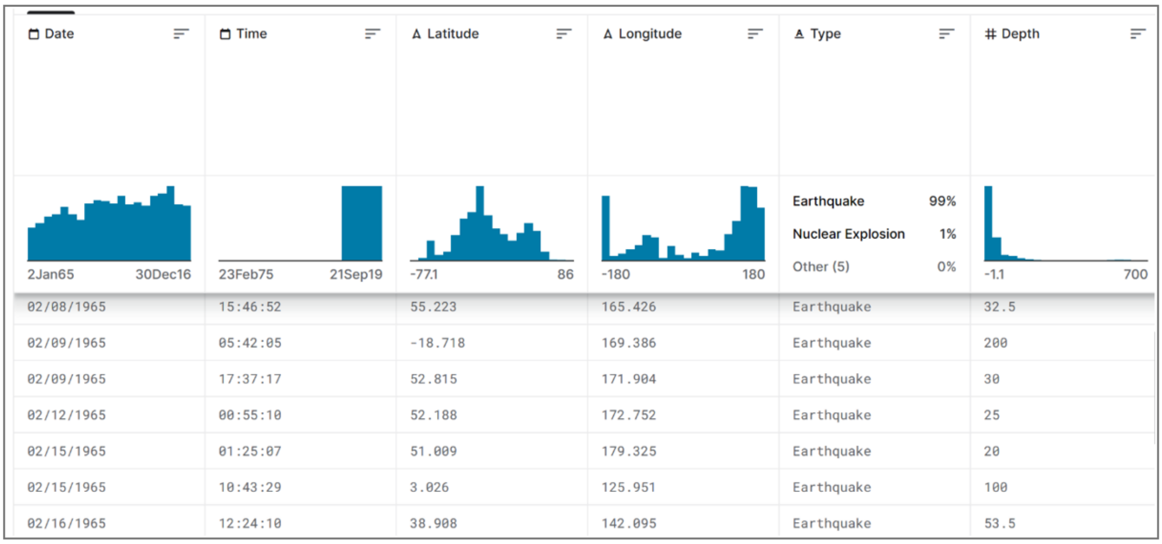
**Topic: Start building the earthquake prediction model by loading and pre-processing the dataset**

**Earthquake Prediction Model**

**INTRODUCTION:**

- Earthquake prediction is a complex and critical field of study, as it holds the potential to save lives and protect infrastructure from the devastating impact of seismic events.
- While predicting the exact time and location of earthquakes with high precision remains a formidable challenge, scientists and researchers have made significant progress in understanding the underlying patterns of seismic activity, enabling the development of models to assess earthquake risk and provide early warnings.
- This introduction will focus on the crucial steps of loading and preprocessing seismic data using Python, essential for building robust earthquake prediction models.

**Given Data Set:**




| Date | Time | Latitude | Longitude | Type | Depth |
|---|---|---|---|---|---|
| 02/08/1965 | 15:46:52 | 55.223 | 165.426 | Earthquake | 32.5 |
| 02/09/1965 | 05:42:05 | -18.718 | 169.386 | Earthquake | 200 |
| 02/09/1965 | 17:37:17 | 52.815 | 171.904 | Earthquake | 30 |
| 02/12/1965 | 00:55:10 | 52.188 | 172.752 | Earthquake | 25 |
| 02/15/1965 | 01:25:07 | 51.009 | 179.325 | Earthquake | 20 |
| 02/15/1965 | 10:43:29 | 3.026 | 125.951 | Earthquake | 100 |
| 02/16/1965 | 12:24:10 | 38.908 | 142.095 | Earthquake | 53.5 |

## Necessary step to follow:

### 1.Import Libraries:

Start by importing the necessary libraries:

### Program:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import os
```

### 2.Load the Dataset:

Load your dataset into a Pandas DataFrame. You can typically findhouse price datasets in CSV format, but you can adapt this code to otherformats as needed.

### Program:

```
data = pd.read_csv(‘ E:\earth_database.csv ’)

pd.read()
```

### 3.Exploratory Data Analysis (EDA):

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

**Program:**

```
data.head()

data.columns
```

## 4.Feature Engineering:

Depending on your dataset, you may need to create new features or transform existing ones. This can involve one-hot encoding categorical variables, handling date/time data or scaling numerical features.

**Program:**

```
model = Sequential()

model.add(Dense(16, activation='relu', input_shape=(3,)))

model.add(Dense(16, activation='relu'))

model.add(Dense(2, activation='softmax'))


model.compile(optimizer='SGD', loss='squared_hinge',
metrics=['accuracy'])

model.fit(X_train, y_train, batch_size=10, epochs=20, verbose=1,
validation_data=(X_test, y_test))
```

## 5.Split the Data:

Split your dataset into training and testing sets. This helps you evaluate your model's performance later.

**Program:**

```
X = final_data[['Timestamp', 'Latitude', 'Longitude']]
```

y = final_data[['Magnitude', 'Depth']]

X_train,X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

## 6.Feature Scaling:

Apply feature scaling to normalize your data, ensuring that all features have similar scales. Standardization (scaling to mean=() and std=1) is a common choice.

## Program:

reg = RandomForestRegressor(random_state=42)

reg.fit(X_train, y_train)

reg.predict(X_test)

## Importance of loading and preprocessing dataset:

Loading and preprocessing an earthquake prediction dataset are crucial steps in the data analysis pipeline, as they lay the foundation for accurate and meaningful analysis and modelling.

## Challenges involved in loading and preprocessing a house price Dataset:

There are a number of challenges involved in loading and preprocessinga house price dataset, including:

- **Handling Missing Data:**
  Earthquake prediction datasets might have missing data, and deciding how to handle it (imputation, removal, etc.) is important to avoid biases in your analysis.
- **Feature Scaling and Normalization:**
  Many machine learning algorithms require feature scaling or normalization to perform optimally. Preprocessing helps in

standardizing the data, which can lead to improved model convergence and performance.

## How to overcome the challenges of loading andpreprocessing aearthquake prediction dataset:

There are a number of things that can be done to overcome the challenges of loading and preprocessing a earthquake prediction dataset, including:

- **Useadata preprocessing library:**
  There are a number of libraries available that can help with datapreprocessing tasks, such as handling missing values, encodingcategorical variables, and scaling the features.
- **Carefully consider the specific needs of your model:**
  The best way to preprocess the data will depend on the specific machine learning algorithm that you are using. It is importanttocarefully consider the requirements of the algorithm and to preprocessthe data in a way that is compatible with the algorithm.

- **Validate the preprocessed data:**
  It is important to validate the preprocessed data to ensure that it isin a format that can be used by the machine learning algorithm and thatit is of high quality. This can be done by inspecting the data visually orby using statistical methods.

## 1.Loading the dataset:

- Loading the dataset using machine learning is the process of bringingthe data into the machine learning environment so that it can be usedto train and evaluate a model.
- Thespecific steps involved in loading the dataset will vary dependingon the machine learning library or framework that is

being used.However, there are some general steps that are common to mostmachine learning frameworks:

**a. Identify the dataset:**

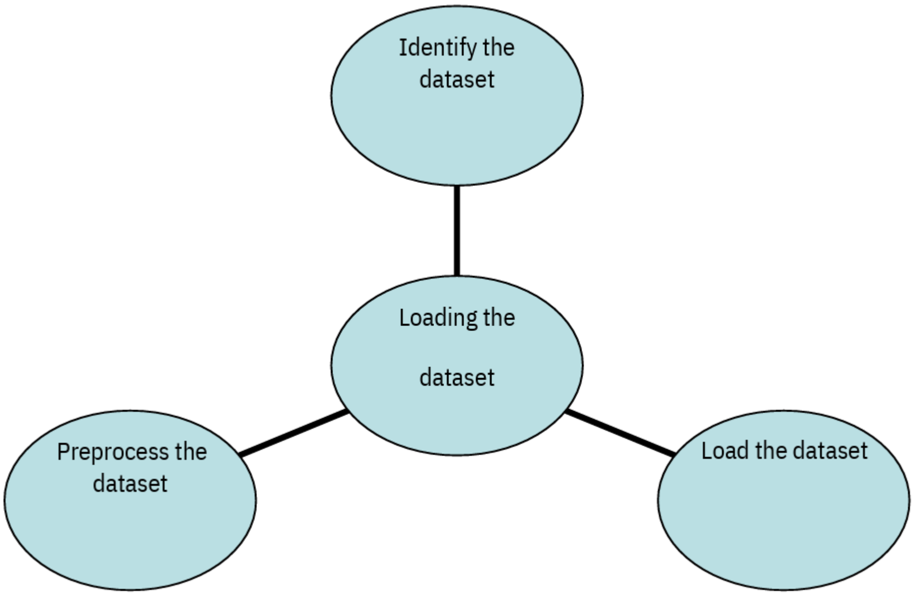
The first step is to identify the dataset that you want to load. Thisdataset may be stored in a local file, in a database, or in a cloud storageservice.

**b.Load the dataset:**

Once you have identified the dataset, you need to load it into the machine learning environment. This may involve using a built-infunction in the machine learning library, or it may involve writingyourown code.

**c.Preprocess the dataset:**

Once the dataset is loaded into the machine learningenvironment,you may need to preprocess it before you can start training andevaluating your model. This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training andtest sets.




Here, how to load a dataset using machine learning in Python

**Program:**

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import os

import datetime

import time

from mpl_toolkits.basemap import Basemap

from sklearn.cross_validation import train_test_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.model_selection import GridSearchCV

from keras.models import Sequential

from keras.layers import Dense
```

from keras.wrappers.scikit_learn import KerasClassifier

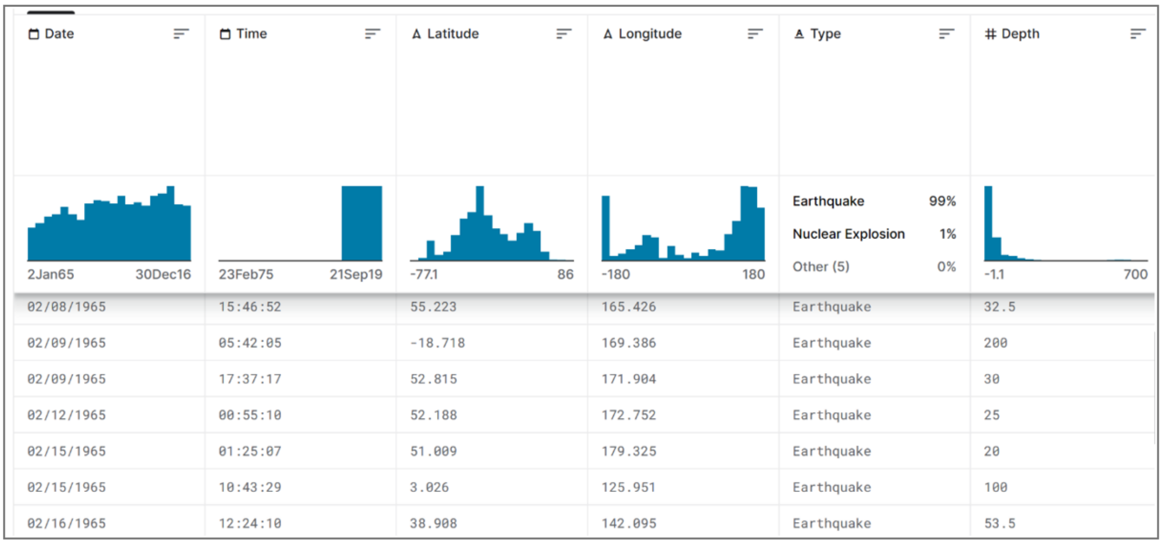
## Loading Dataset:

data = pd.read_csv('E:/earth_database.csv')

## Data Exploration:

### Dataset:

### Output:




| Date | Time | Latitude | Longitude | Type | Depth |
|---|---|---|---|---|---|
| 02/08/1965 | 15:46:52 | 55.223 | 165.426 | Earthquake | 32.5 |
| 02/09/1965 | 05:42:05 | -18.718 | 169.386 | Earthquake | 200 |
| 02/09/1965 | 17:37:17 | 52.815 | 171.904 | Earthquake | 30 |
| 02/12/1965 | 00:55:10 | 52.188 | 172.752 | Earthquake | 25 |
| 02/15/1965 | 01:25:07 | 51.009 | 179.325 | Earthquake | 20 |
| 02/15/1965 | 10:43:29 | 3.026 | 125.951 | Earthquake | 100 |
| 02/16/1965 | 12:24:10 | 38.908 | 142.095 | Earthquake | 53.5 |

## 2.Preprocessing the dataset:

- Data preprocessing is the process of cleaning, transforming, andintegrating data in order to make it ready for analysis.
- Thismayinvolve removing errors and inconsistencies, handlingmissing values, transforming the data into a consistent format, andscaling the data to a suitable range.

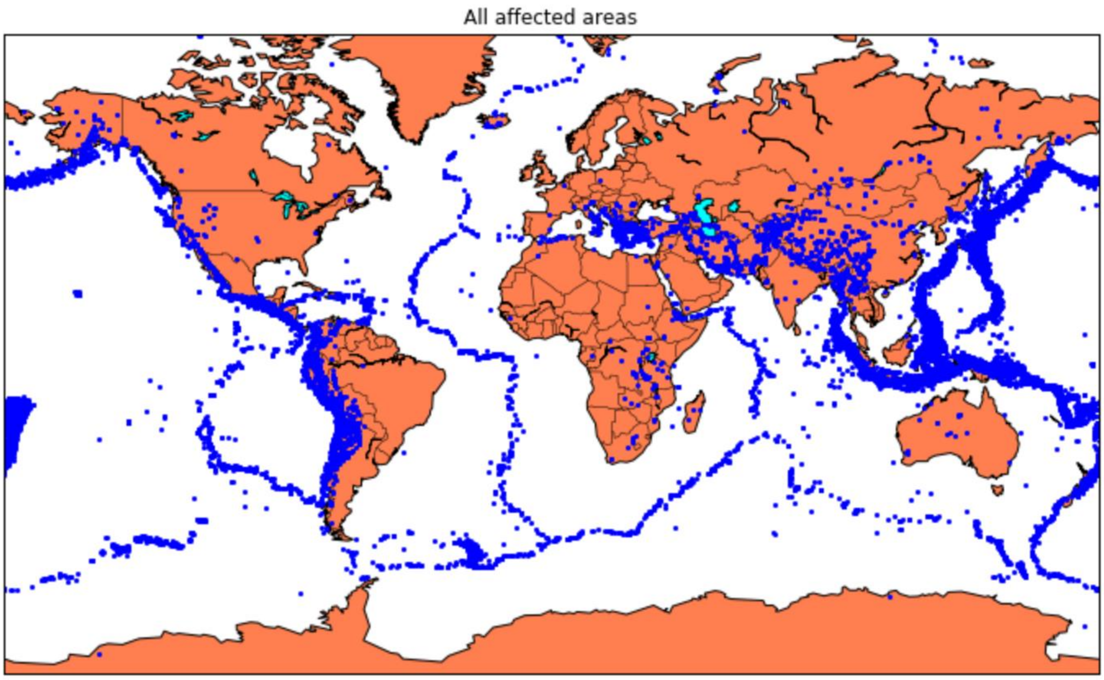
## Visualisation and Pre-Processing of Data:

## In [1]:

fig = plt.figure(figsize=(12,10))

```
plt.title("All affected areas")
m.plot(x, y, "o", markersize = 2, color = 'blue')
m.drawcoastlines()
m.fillcontinents(color='coral',lake_color='aqua')
m.drawmapboundary()
m.drawcountries()
plt.show()
```

Out [1]:

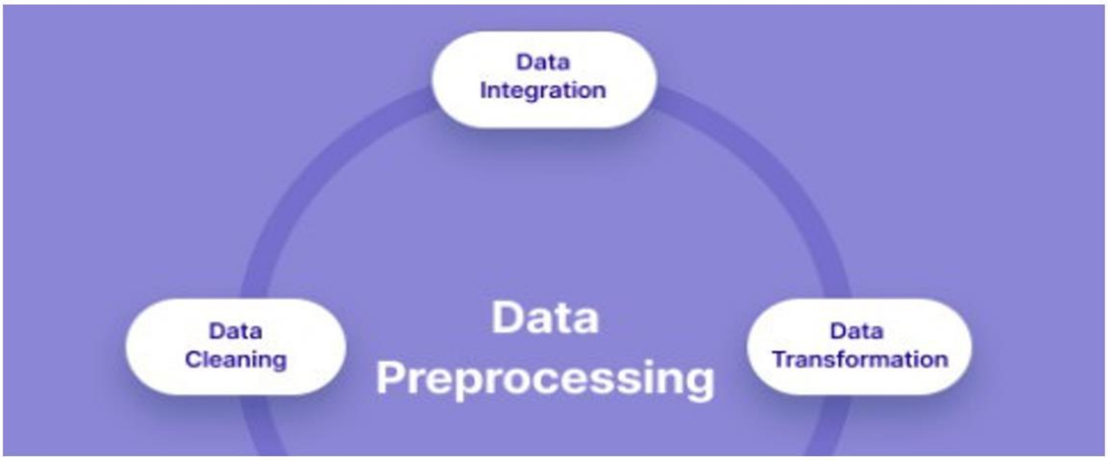



## Some common data preprocessing tasks include:

- **Datacleaning**:
  This involves identifying and correcting errors andinconsistencies in the data. For example, this may involveremoving duplicate records, correcting typos, and filling in missingvalues.
- **Datatransformation:**

This involves converting the data into aformat that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scalingthe data to a suitable range.

- **Feature engineering:**
  This involves creating new features from the existing data. Forexample, this may involve creating featuresthat represent interactions between variables, or features thatrepresent summary statistics of the data.
- **Dataintegration:**
  This involves combining data from multiple sources into a single dataset. This may involve resolvinginconsistencies in the data, such as different data formats or different variable names.Data preprocessing is an essential step in many datascience projects. By carefully preprocessing the data, data scientists canimprove the accuracy and reliability of their results.




**Program:**

```
# Importing necessary libraries

import numpy as np

import pandas as pd
```

```
import matplotlib.pyplot as plt
import datetime
import time
from mpl_toolkits.basemap import Basemap
from sklearn.cross_validation import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier

# Step 1: Load the dataset
data = pd.read_csv('E:/earth_database.csv')

# Step 2: Exploratory Data Analysis (EDA)
data.head()
data.columns

# Step 3: Feature Engineering
grid = GridSearchCV(estimator=model, param_grid=param_grid,
n_jobs=-1)
grid_result = grid.fit(X_train, y_train)
```

```
print("Best: %f using %s" % (grid_result.best_score_,
grid_result.best_params_))

means = grid_result.cv_results_['mean_test_score']

stds = grid_result.cv_results_['std_test_score']

params = grid_result.cv_results_['params']

for mean, stdev, param in zip(means, stds, params):

    print("%f (%f) with: %r" % (mean, stdev, param))



# Step 4: Data Splitting

X = final_data[['Timestamp', 'Latitude', 'Longitude']]

y = final_data[['Magnitude', 'Depth']]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

print(X_train.shape, X_test.shape, y_train.shape, X_test.shape)


# Step 5: Preprocessing and Feature Scaling

reg = RandomForestRegressor(random_state=42)

reg.fit(X_train, y_train)

reg.predict(X_test)
```

**Output:**

**Exploratory Data Analysis**

| | Date | Time | Latitude | Longitude | Type | Depth | Depth Error | Depth Seismic Stations | Magnitude | Magnitude Type |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 01/02/1965 | 13:44:18 | 19.246 | 145.616 | Earthquake | 131.6 | NaN | NaN | 6.0 | MW |
| 1 | 01/04/1965 | 11:29:49 | 1.863 | 127.352 | Earthquake | 80.0 | NaN | NaN | 5.8 | MW |
| 2 | 01/05/1965 | 18:05:58 | -20.579 | -173.972 | Earthquake | 20.0 | NaN | NaN | 6.2 | MW |
| 3 | 01/08/1965 | 18:49:43 | -59.076 | -23.557 | Earthquake | 15.0 | NaN | NaN | 5.8 | MW |
| 4 | 01/09/1965 | 13:32:50 | 11.938 | 126.427 | Earthquake | 15.0 | NaN | NaN | 5.8 | MW |

| Magnitude Error | Magnitude Seismic Stations | Azimuthal Gap | Horizontal Distance | Horizontal Error | Root Mean Square | ID | Source | Location Source | Magnitu Source |
|---|---|---|---|---|---|---|---|---|---|
| NaN | NaN | NaN | NaN | NaN | NaN | ISCGEM860706 | ISCGEM | ISCGEM | ISCGEM |
| NaN | NaN | NaN | NaN | NaN | NaN | ISCGEM860737 | ISCGEM | ISCGEM | ISCGEM |
| NaN | NaN | NaN | NaN | NaN | NaN | ISCGEM860762 | ISCGEM | ISCGEM | ISCGEM |
| NaN | NaN | NaN | NaN | NaN | NaN | ISCGEM860856 | ISCGEM | ISCGEM | ISCGEM |
| NaN | NaN | NaN | NaN | NaN | NaN | ISCGEM860890 | ISCGEM | ISCGEM | ISCGEM |

Index(['Date', 'Time', 'Latitude', 'Longitude', 'Type', 'Depth', 'Depth Error',

'Depth Seismic Stations', 'Magnitude', 'Magnitude Type',

'Magnitude Error', 'Magnitude Seismic Stations', 'Azimuthal Gap',

'Horizontal Distance', 'Horizontal Error', 'Root Mean Square', 'ID',

'Source', 'Location Source', 'Magnitude Source', 'Status'],

dtype='object')

Best: 0.666684 using {'activation': 'sigmoid', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16, 'optimizer': 'SGD'}

0.666684 (0.471398) with: {'activation': 'sigmoid', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16, 'optimizer': 'SGD'}

0.000000 (0.000000) with: {'activation': 'sigmoid', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16, 'optimizer': 'Adadelta'}

0.666684 (0.471398) with: {'activation': 'relu', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16, 'optimizer': 'SGD'}

0.000000 (0.000000) with: {'activation': 'relu', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16, 'optimizer': 'Adadelta'}

## Data Splitting

(18727, 3) (4682, 3) (18727, 2) (4682, 3)

```
array([[  5.96,  50.97],
       [  5.88,  37.8 ],
       [  5.97,  37.6 ],
       ...,
       [  6.42,  19.9 ],
       [  5.73, 591.55],
       [  5.68,  33.61]])
```

## Conclusion:

the process of loading and preprocessing data for an earthquake prediction model in Python is a critical and foundational step in the development of a reliable and accurate predictive system. This phase is essential for ensuring that the model can effectively learn from the available data and make informed predictions.

Loading the data involves collecting, importing, and organizing relevant datasets that contain seismic and geological information. This data can be obtained from various sources, including seismometers, geological surveys, and satellite imagery, among others. It is imperative to ensure data quality, accuracy, and consistency during this step, as any discrepancies or anomalies in the dataset can greatly affect the model's performance.

Preprocessing the data is equally crucial, as it involves cleaning, transforming, and feature engineering to make the data suitable for machine learning algorithms. Common preprocessing tasks include handling missing values, scaling features, encoding categorical variables, and splitting the data into training and testing sets. Additionally, data augmentation techniques may be employed to increase the model's robustness.

Furthermore, domain expertise is essential in this process, as it can guide the selection of relevant features and the creation of meaningful input representations for the model. It is also important to keep in mind the temporal and spatial aspects of earthquake data, as these can play a significant role in model design and feature engineering.