Earthquake Prediction using XAI

Rishabh Jain

IIT Delhi
Research Intern
National University of Singapore

Guide: Vishal Srivastava

National University of Singapore

Summary Report

13th June 2024

NUS
National University
of Singapore

Outline

1. Paper Review 1- SPA in Arabia (XAI)
2. Paper Review 2- AI in Predicting Earthquakes: State-of-the-Art
3. Data Exploration on our data

Paper Review- Spatial Probability Assessment (SPA) in Arabia (XAI)

1. **Novelty**: Application of an XAI framework to estimate earthquake spatial probability and identify the contributing factors, hidden interaction, and their relative importance.

2. **Key Features**: 1. Hybrid Inception-v3 XgBoost
   2. SHAP to understand predictors
   3. Output variation with change in factors

3. **Data**:

3.1 *Data Collection*: 1. NEIC, USGS, NCS databases
   2. GIS environment data
   3. Remote Sensing images

3.2 *Features*: Slope (Degree), Elevation (m), Curvature (radians/m), Magnitude variation (Mw), Depth variation (m), Epicenter density, Seismic gap (km), Earthquake frequency, PGA (cm/s^2), Proximity to thrust faults (km), Tectonic contacts density (km)

4. **Methodology**: Inception v3-XGBoost model is used to predict the targets, such as earthquake points and non-earthquakes.

Then, the SHAP library was implemented using JavaScript functions to estimate the contribution of factors towards hazard assessment and factors interaction.

5. **Results**: The hybrid model performed well in classifying earthquakes. As per the results, PGA and magnitude variation contribute the highest with the SHAP values of 4.3 and 5, respectively.

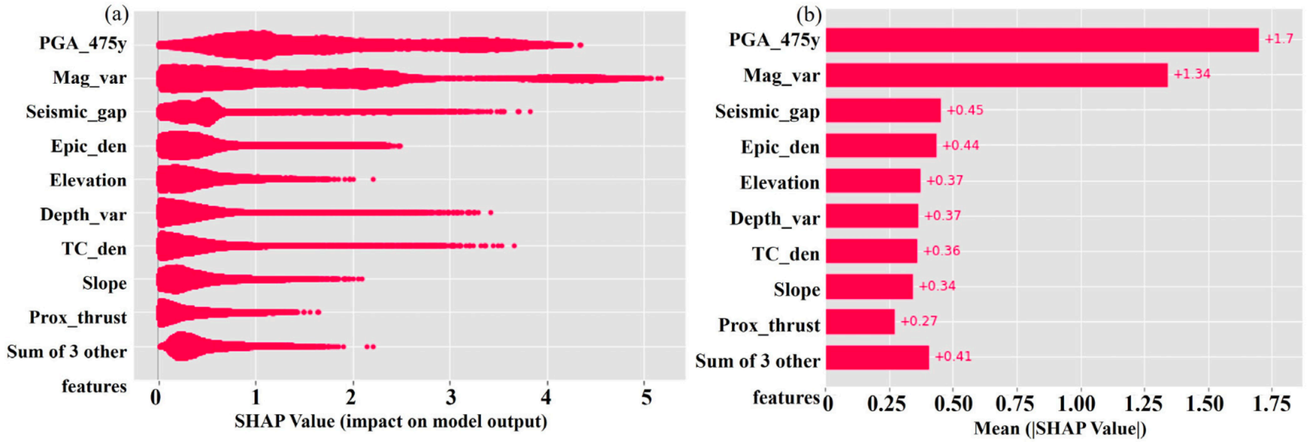


Figure 6. SHAP interpretation for earthquake SPA: (**a**) bee swarm plot portrays SHAP importance, and (**b**) bar plot shows importance based on mean SHAP values.

A comparison between the probability maps without and with three important factors was conducted. The result shows that the one with three important factors portrays better output than the one without PGA, magnitude variation, and seismic gap as shown above.

The highest weights were achieved by PGA (14%), magnitude variation (13%), seismic gap (12%), and epicenter density (10%). These are the four highly recommended and globally stable factors for the SPA in the Arabian Peninsula.

Continued

Models	Accuracy	References
PRNN	58%	Asim et al. [60]
RNN	64%	
RF	62%	
LPBoost	65%	
ANN	84%	Jena et al. [15]
CNN	90%	Huang et al. [61]
Inceptionv3-XGBoost	87.9%	Proposed model

After the pre-processing stage, 1,000,000 points were derived from the study area for testing purposes. **The hybrid model achieved an overall accuracy of 87.91%.** The macro average was achieved by the model with a precision of 0.8805, while the weighted average with a precision of 0.8806. The earthquake probability classification achieved a precision of 0.8557, while the earthquake non-probability assessment achieved a precision of 0.9053. This shows that the trained model **classifies the non-earthquake points more efficiently than earthquake points due to the skewness in data.**

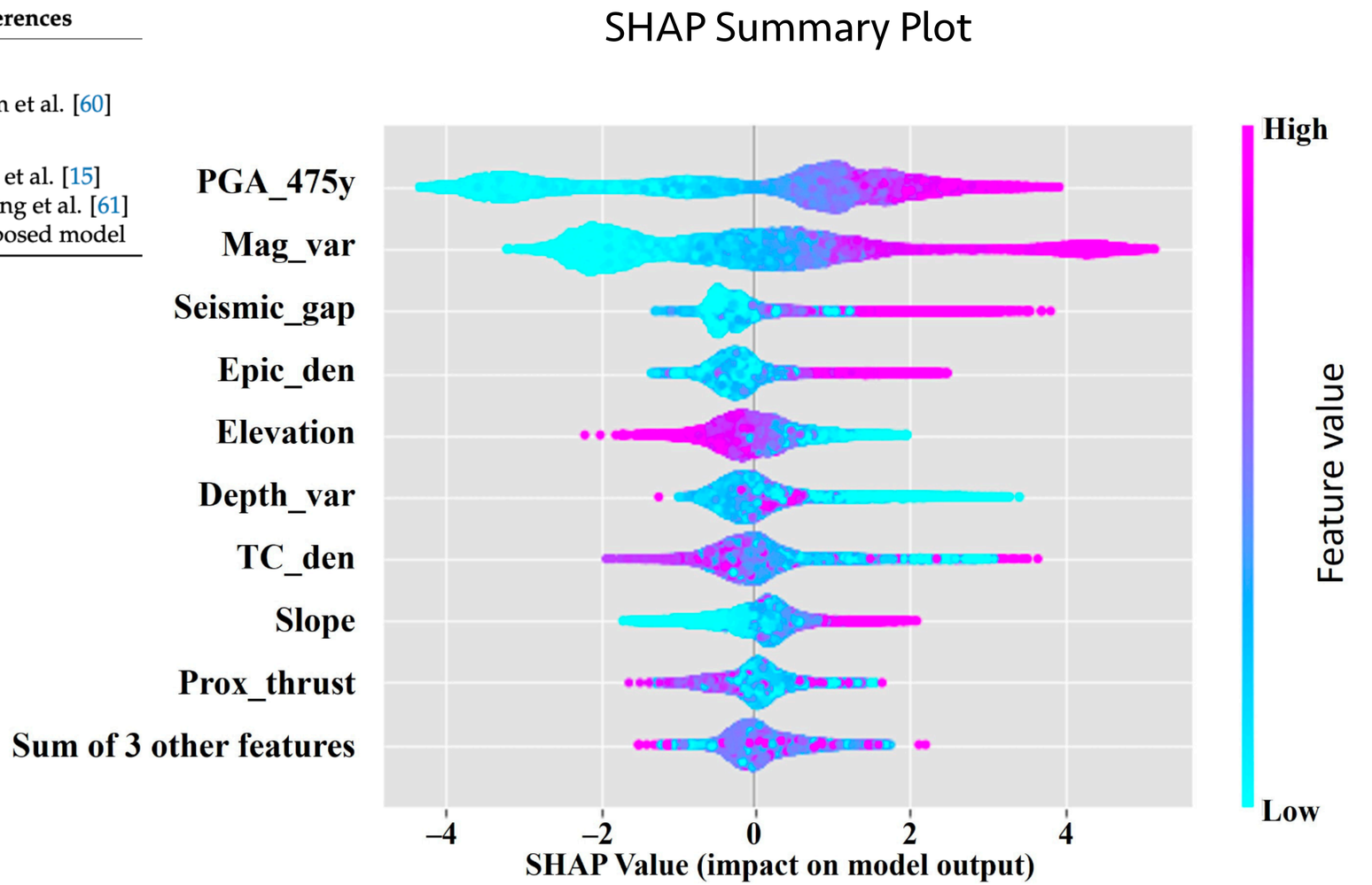
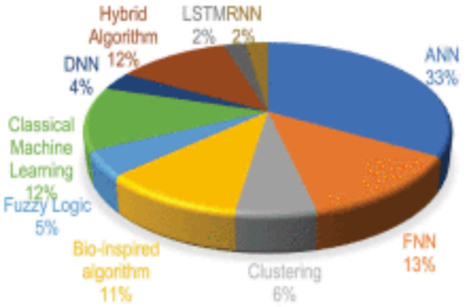
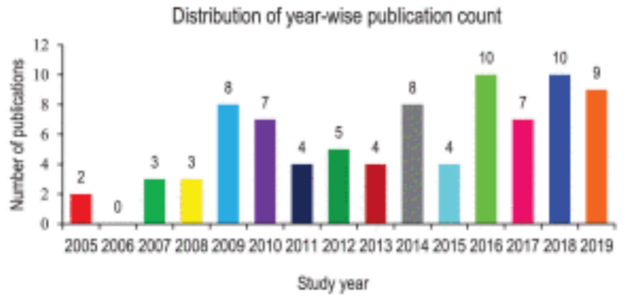


Figure 8. The summary plot shows the factors' interaction and importance based on Inception v3-XGBoost model.

Paper Review- AI in Predicting Earthquakes: State-of-the-Art

1. Objective: This work systematically explores the contributions made to date in earthquake prediction using AI-based techniques. A total of **84** scientific research papers, which reported the use of AI-based techniques in earthquake prediction, have been selected from different academic databases. Covering all existing AI-based techniques in earthquake prediction, this article provides an account of the available methodologies and a **comparative analysis** of their performances.





2. Observations: Here are some salient observations which are derived from the analysis of all the papers:

2.1 *Prediction Accuracy Limited to Magnitude:* All earthquake prediction models performed very well in predicting magnitudes **between 3 and 5**. Naturally, an earthquake with a magnitude **greater than magnitude 6 is rare**. When an earthquake with a magnitude higher than 6 happens, the AI-based models show poor performance because of **data scarcity.**

2.2 *No Benchmark Datasets:* The earthquake prediction datasets typically contain **earthquake catalog, SES, seismic waves, precursory parameters, or animal behavior**. Consequently, for a specific area, data is not sufficient. SES for the historical earthquake catalog is not available. VAN team managed to record SES for 29 events in quest of creating an SES dataset. Though this effort is not enough for machine learning or deep learning researches. The DL-based approach cannot be used due to the scarcity of large benchmark dataset. An earthquake dataset should be created on which every earthquake model can be tested for easing the comparison process of the different models.

2.3 *Insufficient Effective Parameters:* Different studies use different parameter sets. The magnitude of completeness is different for different datasets as well. Based on these, b-value parameters are calculated. In a single dataset, different earthquake's magnitude techniques (Local-scale, Richter scale) are evaluated by different techniques. In most cases, the **same method performs very differently, just because the dataset is different**. The best set of earthquake features need to be defined for different geological locations. While recording the earthquake data, a specific magnitude scale should be adopted all over the world.

Continued

3. **Performance Analysis**: This paper divides ML approaches into broadly 3 categories i.e. Rule-Based, Shallow ML (Classical & Neural Network Based) and Deep ML.

Category	High Performance	Medium Performance	Low Performance
Rule-Based	ANFIS based methods [55] and [47] provided relative RMSE of 17.96% and 24.95%. The ANFIS with ELM method [58] achieved relative RMSE of 19.8% and GFCV method [48] achieved relative RMSE of 0%.	The NFC-based method [51] got relative accuracy of 42.86% and the ANFIS-based method [57] provided relative RMSE of 28.62%.	ANFIS-based method [54] achieved relative accuracy of 0% and the ANFIS-based method [52] had relative RMSE of 0%.
Classical Shallow ML	The SVR-based method [61] was high performing with relative accuracy of 87.9%. HWT-based method [64] for feature transformation was found successful (relative accuracy 100%). DT-based[66] & KMC-TBS methods [69] were high performing.	The bio-inspired PSO-based clustering method [77] achieved relative accuracy of 46.77%, which makes it a medium performing method. The SVM-SVD method [60] produced relative accuracy of 29.32%.	The SVM based method [59] low performing as the relative accuracy of these models is 0%. The KMC method [68] and the HKMC with ANN-based methods [71], [72] were also low performings with a relative accuracy of 3.23%, 9.68% and 19.35%.
NN-Based Shallow ML	BP-AdaBoost [37] & NDC-NDAP methods [97] had relative accuracy of 98.59% and 98.27%. The bio-inspired IABC-MLP method achieved relative accuracy of 100%. The RBFNN model [106] also performed well with a relative accuracy of 86.30%.	ANN-based models [82]–[84], [90] were medium performing based on accuracy with relative accuracies 44.21%, 42.48%, 34.17%, & 29.68%. PNN method [3], and the DT-regressor method [109] had relative accuracy of 58.12% and 65.16%.	ANN-based methods [80], [81] achieved relative accuracy of 16.24% and 0% respectively. The ANN-SVM model [95] and the RF-based method [112] had 6.59% and 12.33% relative accuracy, respectively.
Deep ML	The DNN based models [39], [114] showed high performance with a relative accuracy of 75.53% and 80.85%. The LSTM model [42] performed best with a relative accuracy of 100%	The PRNN method [116] achieved relative accuracy of 57.97%. Therefore, this method was labelled as medium performing.	The vanilla RNN-based method [65] performed worst among the DL-based methods. This method provided a relative accuracy of 0% and reported as a low performing model.

***Note**: All methods have been numbered and can be referred-to in the paper.

Exploratory analysis on our data

What our data looks like:

Target:
0-No Earthquake
1-Earthquake

In [93]:
```
# Combining Data into one dataframe

df = pd.concat([df0,df1,df2,df3,df4,df5,df6,df7], ignore_index=True)
df.sample(5)
```

Out[93]:

	noun_id	signal_names	signal	target	signal_length	sigma_n0	epsilon_t	epsilon_s
8117	signal_15654	b"\x93NUMPY\x01\x00v\x00{'descr': '|S9', 'fort...	b'\x93NUMPY\x01\x00v\x00{\'descr\': \'<f4\', \...	0	500	19400000.0	6000.0	7000.0
5911	signal_9509	b"\x93NUMPY\x01\x00v\x00{'descr': '|S9', 'fort...	b'\x93NUMPY\x01\x00v\x00{\'descr\': \'<f4\', \...	0	500	19500000.0	6000.0	6000.0
12346	signal_17379	b"\x93NUMPY\x01\x00v\x00{'descr': '|S9', 'fort...	b'\x93NUMPY\x01\x00v\x00{\'descr\': \'<f4\', \...	0	500	16100000.0	4000.0	5000.0
3385	signal_4315	b"\x93NUMPY\x01\x00v\x00{'descr': '|S9', 'fort...	b'\x93NUMPY\x01\x00v\x00{\'descr\': \'<f4\', \...	0	500	18900000.0	1000.0	10000.0
7059	signal_25799	b"\x93NUMPY\x01\x00v\x00{'descr': '|S9', 'fort...	b'\x93NUMPY\x01\x00v\x00{\'descr\': \'<f4\', \...	1	500	20300000.0	5000.0	4000.0

Data Processing:

1. The signal needs to be decoded.
2. Drop irrelevant columns, only decoded signal is relevant.

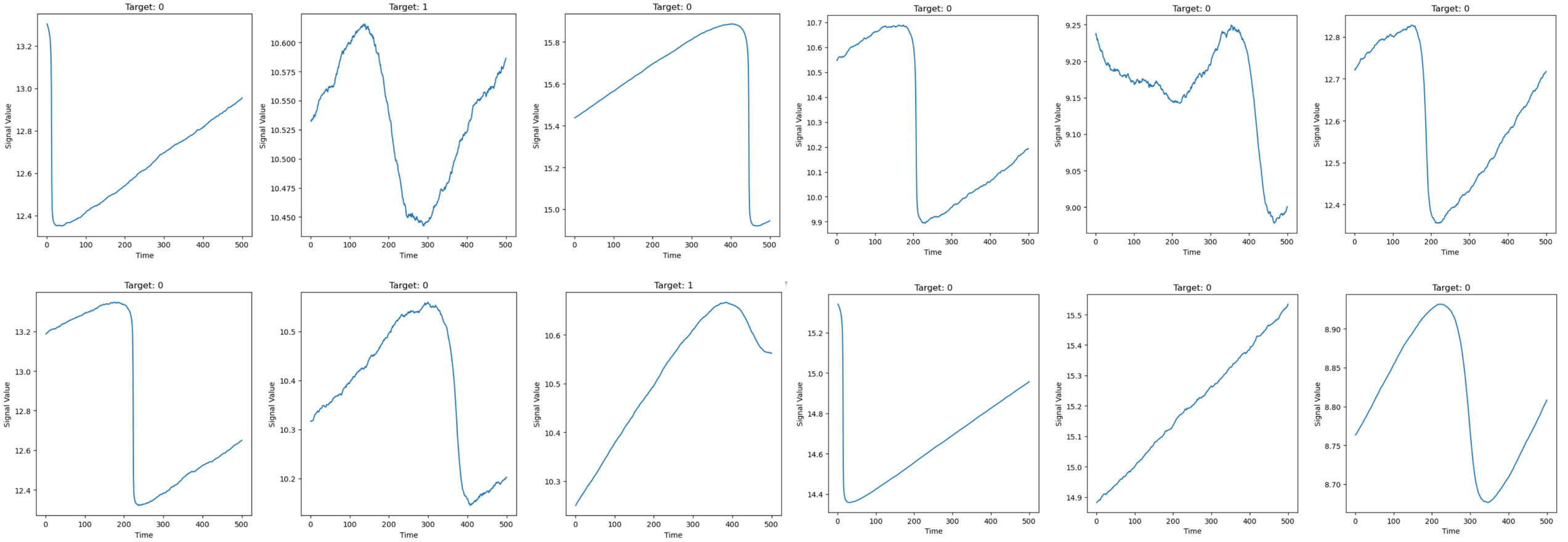
In [123]:
```
# Remove remaining columns

df.drop(['signal_names', 'signal', 'signal_length', 'sigma_n0', 'epsilon_t', 'epsilon_s'], axis=1, inplace=True)
df.sample(5)
```

Out[123]:

	noun_id	target	decoded_signal_names	decoded_signal
2343	signal_966	1	[b'feature_0']	[[9.663074], [9.664831], [9.66563], [9.66734],...
5249	signal_10072	0	[b'feature_0']	[[15.996807], [15.9978895], [15.999032], [16.0...
12218	signal_8965	1	[b'feature_0']	[[12.242362], [12.243764], [12.245162], [12.24...
14248	signal_23338	1	[b'feature_0']	[[11.397999], [11.399357], [11.400076], [11.40...
10378	signal_5273	0	[b'feature_0']	[[11.512595], [11.514619], [11.517252], [11.51...

Continued

Each signal (corresponding to one earthquake) consists of a set of 500 values. Hence, we have the signal magnitudes for 500 times steps.
I have plotted the signal vs time graphs (for some sample data points) to gain more perspective towards the relation between earthquake occurance (target=1) and corresponding signals.



Continued

Observations: Visually, I have not been able to distinguish between earthquake and non-earthquake events.
We can train the data to various models and check the result. For instance, I have trained an XGBoost Classifier using Mean of the signals as the input feature (even though this may be a poor approach from the perspective of the subject i.e. Seismology, also confirmed by the sub-par accuracy for output of training data).

```
In [130]: df['target'] = df['target'].astype(int)

          X = np.array([np.mean(sig, axis=0) for sig in df['decoded_signal']])
          y = df['target']



          model = XGBClassifier()
          model.fit(X, y)

          y_pred = model.predict(X)

          accuracy = accuracy_score(y, y_pred)
          print(f"Accuracy: {accuracy * 100:.2f}%")

          Accuracy: 79.40%
```

Next Steps:

1. Need to find better ways to create a model that utilises only the signal data as input.
2. Possibly find other data and broaden the scope of our analysis.
3. More features will also allow us to work on **Explainability** and understand which features affect the model prediction more.