Predictive Modeling of Tsunami Risk from Seismic Data

Lynda HAMMOUCHE,Imane EL QCHIRI,
Quentin WENDLING, Nicolas Franceschini
Université Paris Cité

2024-2025

Abstract :

Tsunamis, although infrequent, are among the most catastrophic natural disasters, with the potential to cause significant human and material losses within minutes. This work presents a data-driven approach to predicting tsunami occurrence based on earthquake characteristics. Using over 3 million global seismic events recorded between 1990 and 2023, we develop and evaluate a machine learning pipeline centered around the Random Forest Classifier (RFC). Key challenges such as class imbalance, geospatial complexity, and data quality are addressed through careful preprocessing, feature engineering (including the computation of coastline distance), and balanced sampling strategies (SMOTE and undersampling). Our study also compares RFC with Logistic Regression, SVM, and XGBoost to assess robustness across models. Results demonstrate a substantial improvement in tsunami prediction when domain knowledge and resampling techniques are combined.

Keywords: tsunami prediction, earthquake modeling, machine learning, Random Forest, SMOTE, class imbalance, geospatial data

1 Introduction:

Tsunamis are waves caused by sudden oceanic disturbances, often triggered by undersea earthquakes. Despite decades of scientific research, real-time identification of tsunami-generating earthquakes remains an open challenge, especially in scenarios where magnitude alone is not a sufficient indicator. Traditional early warning systems rely on predefined thresholds (e.g., magnitude > 7.0, shallow depth, or coastal proximity), which, while simple and fast, often fail to capture the non-linear patterns behind tsunami occurrences.

Thanks to open-access seismic datasets and advances in supervised learning, it is now possible to revisit this problem from a data-centric perspective. The goal of this project is to design and implement a machine learning model that can predict whether a given earthquake will generate a tsunami, based on features available immediately after the event. Our methodology includes robust preprocessing, construction of geospatial features, treatment of class imbalance, and empirical validation using state-of-the-art classifiers.

2 Related work:

The several research articles we read have addressed the tsunami prediction problem using a variety of effective approaches. These studies explore different modeling strategies, data preprocessing techniques, and feature engineering methods to improve prediction accuracy.

1) Asunción (2024) proposed using deep neural networks to directly predict tsunami alert levels from earthquake parameters. The model achieves results in less than one second and supports uncertainty estimation through probabilistic outputs.

2) Mulia et al. (2022) trained a neural network using offshore pressure data to predict tsunami inundation along the Japanese coast. Their model replaces physics-based simulations and provides accurate predictions using real-time sensor input.

3) In a related paper, Mulia et al. (2020) applied deep learning to enhance low-resolution tsunami simulations into high-resolution inundation maps. This allows for fast and accurate mapping without traditional computation.

4) The article SMOTEHashBoost: Ensemble Algorithm for Imbalanced Dataset Pattern Classification introduces SMOTEHashBoost, a method combining SMOTE, hash-based undersampling, and AdaBoost to tackle class imbalance. It improves fairness and minority class detection in imbalanced datasets

5)The article Architecture-Oriented Agent-Based Simulations and Machine Learning Solution proposes a modular tsunami response system combining machine learning (SVM, XGBoost), agent-based modeling, and GIS data. It uses CRISP-DM for data mining, CFD tools for wave estimation, and Google Maps for navigation to aid evacuation planning and decision-making

6) The article Supervised Machine Learning Algorithms: Classification and Comparison reviews algorithms like SVMs, neural networks, decision trees, and Naïve Bayes.it compares their accuracy, error, and build time. SVMs show the best accuracy.

7) In the article « Machine Learning for Tsunami Waves Forecasting Using Regression Trees » searchers trained

regression tree models using simulated DART buoy data to estimate tsunami wave heights at coastal stations. The method provides fast and accurate forecasts, supports sensor importance analysis, and offers a lightweight alternative to traditional numerical simulations.

8) Górriz et al. (2024) proposed K-fold Cross Upper Bound Validation (CUBV), a statistical test combining K-fold cross-validation with concentration inequalities to estimate conservative confidence intervals for machine learning classifiers. This approach addresses over-optimism in traditional CV, particularly with small and heterogeneous datasets, and reduces false positives by bounding the actual risk under worst-case scenarios.

9) In the article « A Free From Local Minima Algorithm for Training Regressive MLP Neural Networks » searchers proposed a derivative-free learning algorithm for training MLPs on regression tasks, avoiding local minima without using gradient descent. The method relies on directional search for weight updates and outperforms standard backpropagation in terms of speed, generalization, and robustness.

10) Abdallah & Saporta (1998) proposed a method to classify and reduce sets of qualitative variables using relational analysis. Their approach introduces similarity criteria and relational clustering to group variables, enabling either representative variable selection or aggregation into synthetic variables, thus simplifying high-dimensional qualitative datasets.

11) In the article « Tsunami tide prediction in shallow water using recurrent neural networks: model implementation in the Indonesia Tsunami Early Warning System » searchers integrated recurrent neural networks into Indonesia's tsunami early warning system to predict coastal tide levels using simulated data from COMCOT. The RNNs provide accurate and fast forecasts across multiple seismic scenarios, offering a real-time alternative to traditional physical models.

12) Hossen et al. (2024) developed a convolutional neural network to detect seismic P-wave arrivals in real time using single-station waveforms. Their model outperforms traditional algorithms and demonstrates high accuracy even with low signal-to-noise ratios, enabling early earthquake warning systems.

3 Problem Definition:

Tsunamis are rare but devastating natural disasters, often triggered by undersea earthquakes. Detecting whether a seismic event could lead to a tsunami is critical for issuing timely alerts and reducing the risk to human life and infrastructure. While traditional early warning systems rely on fixed thresholds (e.g., earthquake magnitude or location), they can struggle to capture complex, real-world patterns.

In this project, we formulate the challenge as a binary classification task:

Given data about a seismic event, predict whether it is likely to generate a tsunami.

Key challenges include:

- **Data Storage**: Handling and processing large volumes of data can be challenging. Efficient storage and access to the seismic event data, We need to ensure that the data is stored efficiently. Additionally, enabling real-time collaboration among team members is essential to ensure smooth workflow and faster model development.
- **Geographical complexity:** The risk of a tsunami depends not only on the earthquake's strength but also on its location relative to the coastline.
- **Feature selection:** Choosing the appropriate features is a key step in addressing the tsunami detection problem.Some features, if not used carefully, could introduce information from the future into the model and artificially boost performance. Care must be taken to avoid this.
- **Severe class imbalance:** The vast majority of seismic events do not result in tsunamis, making positive cases rare.
- **Model selection**: We need to identify a machine learning model that can best classify seismic events, especially under strong class imbalance and limited input features.

4 Solution:

In this section, we will comprehensively explore the approaches we employed to address the underlying problem statement. Each approach will be dissected, highlighting its rationale, methodology, and relevance to effectively tackle the identified issues.

4.1 Data Storage:

To store and manage our seismic dataset, which contains over 3 million records, we opted for Snowflake's online cloud data platform. Snowflake offers a robust and scalable solution that can easily handle large volumes of data. its powerful infrastructure ensured smooth data access and loading during our experiments.

Additionally, the online version of Snowflake enabled our team to collaborate seamlessly in real time. This was especially useful during data exploration and preprocessing phases, allowing everyone to stay synchronized and work efficiently.

4.2 Data Description:

Our dataset was obtained from Kaggle, and it provides a comprehensive record of earthquake events worldwide from 1990 to 2023. This dataset contains approximately 3 million rows, with each row representing a unique earthquake event.

Key attributes in the dataset include:

- Time (in milliseconds): Timestamp of the earthquake event
- Place: Name of the location affected by the earthquake
- Status: Represents the current state or condition of the event, which could be reviewed or automatic
- Tsunami: Boolean value indicating whether the earthquake generated a tsunami
- Significance: Denotes the importance or impact level of the event, which could be used to assess the potential consequences.
- Data type: Specifies the type of data being referenced
- Magnitude: Refers to the measurement of the size or intensity of an earthquake, typically measured on the Richter or moment magnitude scale.
- State: The state or region where the earthquake occurred
- Longitude: Geographical longitude of the epicenter
- Latitude: Geographical latitude of the epicenter
- Depth: Depth of the earthquake's epicenter.
- Date: Date when the earthquake occurred.

4.3 Data preprocessing:

The preprocessing steps applied to the dataset are as follows:

- No missing values were present in the dataset, so no imputation was required.
- The dataset was filtered to retain only entries where data_type == 'earthquake', which represented approximately 98% of all records.
- The following columns were dropped as they were not relevant for our classification task or could have caused data leakage: status, time, date, place, significance, and data_type.
- The state column was cleaned by removing leading and trailing spaces and standardizing capitalization (e.g., " california" and "California" were unified).
- To manage high cardinality in the state feature, only the 50 most frequent states were retained; the remaining states were grouped into a single 'other' category.
- The categorical feature “state” were one-hot encoded using OneHotEncoder

4.4 Feature Engineering

To enhance our dataset and better capture the geographical influence on tsunami risk, we engineered two new features: the shortest distance between each earthquake and the nearest coastline and wether the earthquake happens on land or not. Using the GeoPandas library, we first converted the earthquake coordinates into a GeoDataFrame. We then projected those data on a land shapefile (sourced from Natural Earth) to know if the coordinates were on land or not. We then reprojected both the earthquake points and the coastline shapefile (sourced from Natural Earth) into a metric coordinate system (EPSG:3857) to enable accurate distance calculations. For each earthquake, we computed the minimum distance to the coast in kilometers and added this as a new feature in the dataset.

This computation required approximately six hours on a machine equipped with 32 GB of RAM and an NVIDIA RTX 4070 SUPER GPU.

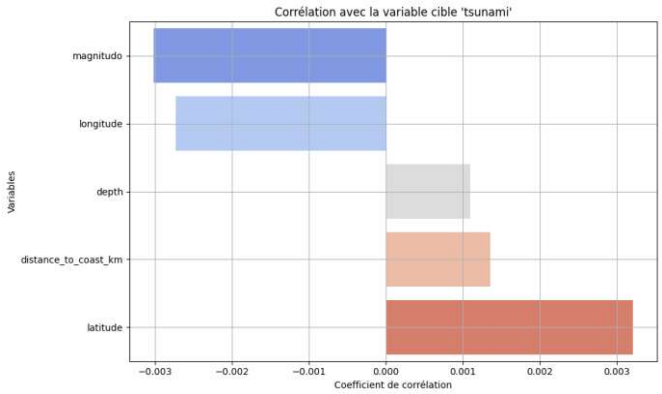
4.5 Data Correlation

Data correlation is a statistical tool used to measure the strength and direction of the relationship between two or more variables. It helps to identify patterns, trends, and potential dependencies in data. By understanding correlations, we can make more informed decisions, improve predictions, and uncover hidden insights in complex datasets.

In our case, we started by checking correlation between every variable remaining and we discovered that correlations were not high, thus we decided to keep every variable remaining.

Then we checked correlation between target variable ‘tsunami’ and the others. We found that correlation for every variable is very low, so we confirmed that we wanted to keep every variable remaining.

Correlation matrix – No state variable



Correlation plot – Correlation between 'tsunami' variable and the others

As we can observe on this graph, coastal distance emerged to not be as important as we thought in predicting if an earthquake will end up creating a tsunami.

4.6 Model selection :

We employed a diverse set of machine learning algorithms specifically crafted for classification purposes. In this section, we will delve into each of these models:

4.6.1 Random Forest :

Random Forest is a robust and widely-used ensemble learning algorithm for classification tasks. It builds multiple decision trees on different random subsets of the dataset and combines their predictions to improve accuracy and reduce overfitting.

- **Ensemble Learning**: Aggregates the results of many decision trees to produce a final prediction via majority vote.
- **Robustness**: Handles noisy data well and is less sensitive to overfitting compared to a single decision tree.
- **Feature Importance**: Offers valuable insights into which features contribute most to the predictions, aiding in interpretability.

4.6.2 Logistic Regression:

Logistic Regression is a fundamental algorithm for binary classification problems. It models the probability that a given input belongs to a particular class using a logistic function.

- **Simplicity and Interpretability**: Easy to implement and interpret, especially useful when relationships between variables are approximately linear.
- **Efficiency**: Fast to train and requires fewer computational resources, making it suitable for large datasets.
- **Probabilistic Output**: Provides probabilities for each class, which is useful for threshold tuning and decision-making.

4.6.3 Support Vector Machines (SVM):

SVM is a powerful classifier that aims to find the optimal hyperplane that separates data into different classes with the maximum margin.

- **Effective in High Dimensions**: Performs well when the number of features is high.
- **Flexibility**: Supports different kernel functions (e.g., linear, RBF) to handle both linear and non-linear classification.
- **Robustness to Overfitting**: Particularly effective in cases where the margin of separation is clear.

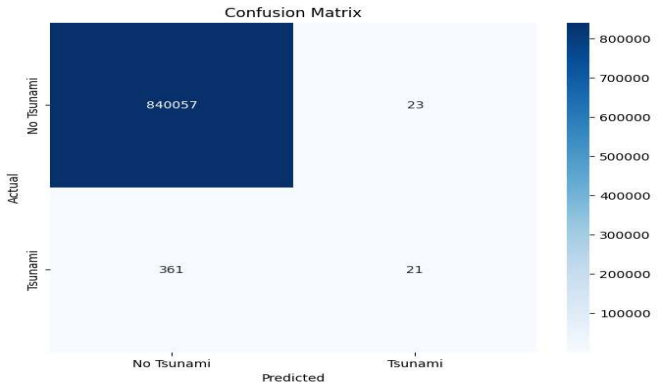
4.6.4 XGBoost (Extreme Gradient Boosting):

XGBoost is a high-performance gradient boosting algorithm known for its speed and accuracy. It builds trees sequentially, where each new tree corrects the errors of the previous ones.

- **Gradient Boosting Framework**: Combines multiple weak learners (typically decision trees) to create a strong classifier.
- **Regularization**: Includes built-in L1/L2 regularization to prevent overfitting.
- **Scalability**: Optimized for speed and performance, making it suitable for large-scale datasets.

4.7 Data oversampling and undersampling:

One major challenge we encountered was the severe imbalance in our target variable: only 1,527 out of over 3.36 million recorded earthquake events were labeled as tsunamis. This corresponds to less than 0.05% of the dataset, making it highly skewed. To assess the impact of this imbalance, we trained a Random Forest classifier on the raw data without any resampling. As expected, the model performed well in detecting non-tsunami events—which make up most of the dataset—but struggled significantly to identify actual tsunamis. This confirmed the need for appropriate resampling strategies to ensure the model learns to detect the minority class more effectively.



Random Forest Confusion Matrix – Original Imbalanced Dataset

4.7.1 Undersampling :

Undersampling is a technique used to address class imbalance in datasets by reducing the number of samples from the majority class. The goal is to balance the distribution of classes, which can improve the performance of machine learning models, especially when the model tends to be biased toward the majority class. By undersampling, we decrease the number of majority class instances, making it easier for the model to learn patterns from the minority class.

In our case, we specifically used **random undersampling**. This method randomly selects a subset of the majority class (non-tsunami events) and reduces it to a specified size. We set the `sampling_strategy=0.1`, meaning we reduced the number of non-tsunami events to 10 times the number of tsunami events.

4.7.2 Oversampling :

Oversampling is a technique used to address class imbalance by increasing the number of samples in the minority class. This is achieved by replicating or generating synthetic samples of the minority class to balance the distribution of classes. The goal is to give the model a better representation of the minority class, ensuring it learns the patterns more effectively and improving the performance on imbalanced datasets.

In our case, we used **SMOTE (Synthetic Minority Over-sampling Technique)**, an advanced oversampling technique. SMOTE works by generating synthetic samples of the minority class based on the existing ones. It does so by selecting a minority class sample, finding its k-nearest neighbors, and generating new synthetic examples by interpolating between the selected sample and its neighbors. This technique allows the model to learn from new, synthetic examples rather than just duplicated instances, enhancing its ability to generalize.

5 Experiments :

5.1 Hardware Description :

The heavy computations were conducted on a personal computer equipped with 32 **GB of RAM**, an AMD Ryzen 5 7500F **series processor**, and an **NVIDIA GeForce RTX 4070 SUPER GPU**.

5.2 Software Tools :

Multiple libraries were used in our experiments, this is the list of the most important ones:

- pandas : Powerful data manipulation and analysis library for Python.
- NumPy : Foundation for numerical computing in Python
- Matplotlib : Widely used library for creating static visualizations..
- Sickit-learn: General purpose machine learning library (potentially for pre processing or other model)
- XGBoost: Efficient Gradient Boosting implementation.
- Imbalanced-learn : Specialized tools for handling imbalanced datasets. We used it to perform oversampling (with SMOTE) and undersampling (with random undersampling) to improve class distribution in the training data.
- GeoPandas: Geospatial extension of Pandas that enables spatial operations. It was used to compute the shortest distance from each earthquake to the nearest coastline.

5.3 Results :

After applying both undersampling and oversampling techniques in our training data, we proceeded to train and evaluate the machine learning models.

Model	Sampling	Precision (Class 0)	Precision (Class 1)	Recall (Class 0)	Recall (Class 1)	F1-score (Class 0)	F1-score (Class 1)
Random Forest Classifier	None	0,48	1	0,05	1	0,1	1
Random Forest Classifier	RandomOverSampler	0,48	1	0,15	1	0,22	1
Random Forest Classifier	SMOTE	0,29	1	0,49	1	0,36	1
Random Forest Classifier	Undersampling Method 1	0	1	0,31	0,6	0	0,75
Random Forest Classifier	Undersampling Method 2	0,55	0,59	0,71	0,42	0,62	0,49
SVM	SMOTE	0	1	0,82	0,79	0	0,89
xgboost	RandomOverSampler	0,17	1	0,84	1	0,28	1

The table above shows the performance of several classification models (Random Forest, XGBoost and SVM) evaluated with different resampling techniques (SMOTE, RandomOverSampler and two undersampling methods) to deal with class imbalance. The results show that undersampling methods, while effective in improving detection of the minority class (class 1), struggle to predict the majority class (class 0) well, with F1-scores often very low. Conversely, some undersampling techniques, notably the second method, achieve a better balance between performance on the two classes. SVM with SMOTE has good recall for both classes, but zero precision for the majority class, limiting its usefulness. The XGBoost model with RandomOverSampler shows high recall on class 0 but very low precision, suggesting numerous false positives.

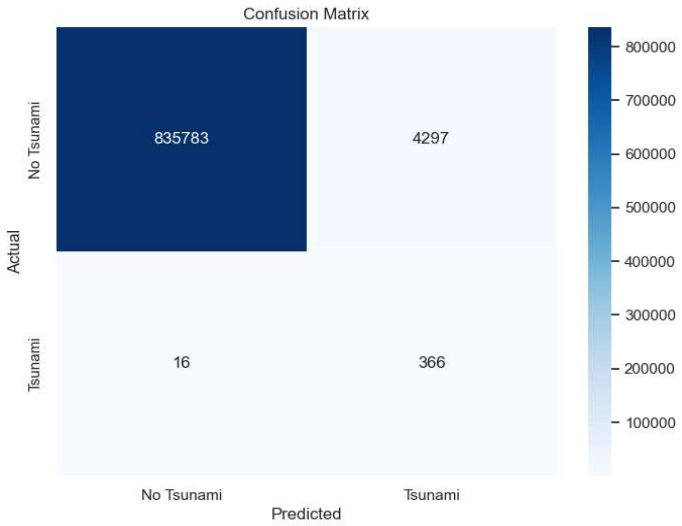
6. Visualizations and Interpretation

To better understand model behavior and validate performance, we employed several visualization techniques:

6.1 Confusion Matrices

Confusion matrices allow us to assess classification performance by showing true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

- In the original unbalanced dataset, the matrix revealed an overwhelming number of TNs with almost no TPs, meaning the model failed to detect tsunami events.
- After applying SMOTE, the matrix showed a significant increase in TPs. Although the number of FPs increased, this trade-off was acceptable due to the higher priority on catching real tsunami events.
- Undersampling led to a more balanced matrix with moderate precision and recall.



Random Forest Confusion Matrix – After applying undersampling and oversampling in training data

6.3 Feature Importance

For both RFC and XGBoost, feature importance scores help us understand which features most influence predictions:

- **Magnitude** was the most significant predictor, as expected.
- **Distance to coastline** ranked second, validating the usefulness of geospatial feature engineering.
- Other relevant features included **depth**, **longitude**, and **latitude**.

These visualizations not only support model evaluation but also provide interpretability, which is critical in high-stakes domains like natural disaster prediction.

7. Conclusion

7.1 Summary of Contributions

This project explored the application of machine learning to predict tsunami occurrence based on seismic event data. Our work addressed several key challenges:

- **Class imbalance** was mitigated using SMOTE, oversampling and undersampling, leading to improved recall.
- **Geospatial engineering** introduced coastal distance as a key feature, enhancing model performance.
- **Model comparison** showed Random Forest with SMOTE outperformed simpler methods like Logistic Regression.

We demonstrated that predictive modeling of tsunamis is not only feasible but significantly improved through thoughtful data preprocessing and domain-aware feature engineering.

7.2 Limitations

- **Label noise**: Some tsunami labels in historical data may be incorrect or missing, impacting training quality.
- **Feature limitations**: The dataset lacks oceanographic and tectonic context (e.g., bathymetry, sea-floor displacement).
- **Data latency**: Real-time application would require immediate access to seismic features and coastline proximity calculations.

8. References

- Asunción (2024) – Prediction of Tsunami Alert Levels Using Deep Learning
- Mulia et al. (2022) – Machine learning-based tsunami inundation prediction derived from offshore observations
- Mulia et al. (2020) – Applying a Deep Learning Algorithm to Tsunami Inundation Database of Megathrust Earthquakes
- Gorriz, J.M., Martin Clemente, R., Segovia, F., Ramirez, J., Ortiz, A., & Suckling, J. (2024). Is K-Fold Cross Validation the Best Model Selection Method for Machine Learning?
- Abdallah, H., & Saporta, G. (1998). Classification of a set of qualitative variables.
- Li et al. (2024) – Dual-scale spatiotemporal graph neural network for traffic flow forecasting
- Dharmawan et al. (2024) – Tsunami tide prediction in shallow water using recurrent neural networks
- Asunción, A. M. (2024). Prediction of tsunami alert levels using deep learning.

- Augusto Montisci (2024) - A Free From Local Minima Algorithm for Training Regressive MLP Neural Networks.
- Yadav, S., Joshi, D., Mulye, S., Udmale, S. S., & Bhole, G. P. (2025). SMOTEHashBoost: Ensemble algorithm for imbalanced dataset pattern classification. *IEEE Access*.
- Čech, P., Mattoš, M., Anderková, V., Babič, F., Alhasnawi, B. N., Bureš, V., Kořínek, M., Štekerová, K., Husáková, M., Zanker, M., Manneela, S., & Triantafyllou, I. (2023). Architecture-oriented agent-based simulations and machine learning solution: The case of tsunami emergency analysis for local decision makers. *Information*, 14(3), 172.
- Akinsola, J. E. T. (2017). Classification and comparison of supervised machine learning algorithms. *International Journal of Computer Trends and Technology*, 48(3), 128–138.