Richter's Predictor: Modeling Earthquake Damage

Shambhavi Sinha
sinhasha@usc.edu

Anushka Patil
anushkap@usc.edu

Shriya Padhi
spadhi@usc.edu

Abstract

Earthquake damage modeling is an essential tool for understanding the potential impact of earthquakes and for developing effective strategies for risk reduction and disaster response. This study aims to predict the level of damage to buildings caused by the 2015 Gorkha earthquake in Nepal based on the aspects of building location and construction as a part of the DrivenData Competition (3). The earthquake resulted in significant damage and loss of life, highlighting the importance of effective risk assessment and mitigation strategies. The study employs LightGBM (9), a gradient boosting model, with Optuna (1), a hyperparameter optimization algorithm, and 5-fold cross-validation to achieve the best predictive performance. The results suggest that this approach has practical applications for improving building codes and retrofitting existing structures in earthquake-prone areas, ultimately reducing the impact of future earthquakes on vulnerable communities.

1 Introduction

Over 22,000 people were injured and over 9,000 people lost their lives as a result of the 2015 Gorkha earthquake in Nepal. In response, the Nepali government used mobile technology to perform a thorough household survey to evaluate the damage in the impacted districts. While identifying households eligible for government aid served as the survey's main goal, the socio-economic data that was gathered will also help researchers and local governments better understand how the earthquake affected people's lives and communities.

This project is a submission for Richter's Predictor: Modeling Earthquake Damage Machine Learning Competition hosted by DrivenData (3) with a total of 2064 participants. The project's goal is to forecast the degree of structure damage caused by the Nepalese earthquake that struck Gorkha in 2015, using machine learning techniques. The damage grade (an ordinal variable with three categories:low damage, medium damage, and nearly total destruction) represents the extent of the building's damage. The project is significant because it applies machine learning to the extensive dataset gathered to show how these methods may be used in order to generate insights that can guide future disaster response operations in order to alleviate the damage caused by finding patterns and correlations in the data.

We organize our report as follows:

1. The first section covers exploratory data analysis which aims to uncover relationships between different features and the damage grade, which can guide structural design choices and risk assessments. Additionally, the section will cover outlier treatment, damage grade distribution, missing values, and dimensionality reduction techniques employed in the analysis. Overall, the section will provide insights into the dataset's characteristics and the potential factors affecting building damage during earthquakes.
2. The second section covers the feature engineering approach where various techniques used to prepare the dataset for modeling will be discussed, including dropping the building ID feature, creating a new volume feature, and applying geo-embedding (2). The section

will also cover the analysis of feature importances and how it can aid in selecting relevant features for better model performance.

3. The last section is concerned with modeling, goes over the various algorithms that were employed, starting with Logistic Regression (6) and continuing on to Random Forest (8), CatBoost (7), XGBoost (5), and LightGBM (9).

The overall goal of the project is to give a thorough methodology that makes use of a variety of machine learning approaches to find the most effective model for the classification assignment.

2 Data

The dataset contains detailed information about the construction and ownership of structures in the Gorkha earthquake-affected area (3).The dataset has 260,000 entries and 38 features, which include the region where the structure is located, the number of floors within the structure, and the dimensions of its footprint. The unique identifier for each building is building_id, and it represents a particular building affected in the earthquake.

Other important features of the dataset include the type of construction materials used in certain sections of the structure and its plan configuration. The dataset also includes flags indicating secondary usage of the superstructure and the building, such as agricultural, industrial, or institutional uses. The ownership status of the property on which the structure was constructed is also given.

The report evaluates algorithms for forecasting damage levels, which range from 1 to 3, using the micro averaged F1 score as the performance parameter. This variant of the F1 score is used since there are three possible labels.

In summary, the dataset provides a comprehensive view of the structures in the Gorkha earthquake-affected area and their characteristics. The features captured in the dataset can be used to evaluate the damage levels of the buildings using machine learning algorithms.

Here is an overview of the building features and characteristics in the dataset.

Geographic regions: Variables indicate the geographic location of the building, with three levels of specificity from the largest region to the smallest sub-region. (*geo_level_1_id, geo_level_2_id, geo_level_3_id*)

Building characteristics: Variables describing the building's physical properties, such as the number of floors, age, area, and height.(*count_floors_pre_eq, age, area_percentage, height_percentage*)

Categorical variables: Variables provide categorical information about the building, such as land surface condition, foundation type, roof type, and others. (*land_surface_condition, foundation_type, roof_type, ground_floor_type, other_floor_type, position, plan_configuration, legal_ownership_status*)

Superstructure materials: Binary flag variables indicate the types of materials used in the building's superstructure, such as Adobe/Mud, Timber, etc. (all the *has_superstructure* variables)

Building usage: Variables representing the number of families living in the building and binary flags for various secondary uses, like agriculture, hotel, school, or government office, among others. (count_families and all the *has_secondary_use* variables)

3 Exploratory data analysis (EDA)

In our EDA, we aim to uncover patterns, trends, and relationships within the Nepal Earthquake 2015 dataset (3), as well as pre-process the data for further analysis.

1. **Identifying different Trends:**
    - **Number of Floors:** Analyzing the distribution of the number of floors before the earthquake for each damage grade, we can observe distinct patterns across different severity levels. While there are buildings in the dataset with up to 9 floors, the majority of the affected buildings had only 2 floors. This observation suggests that lower-rise buildings may have been more vulnerable to damage during the earthquake.

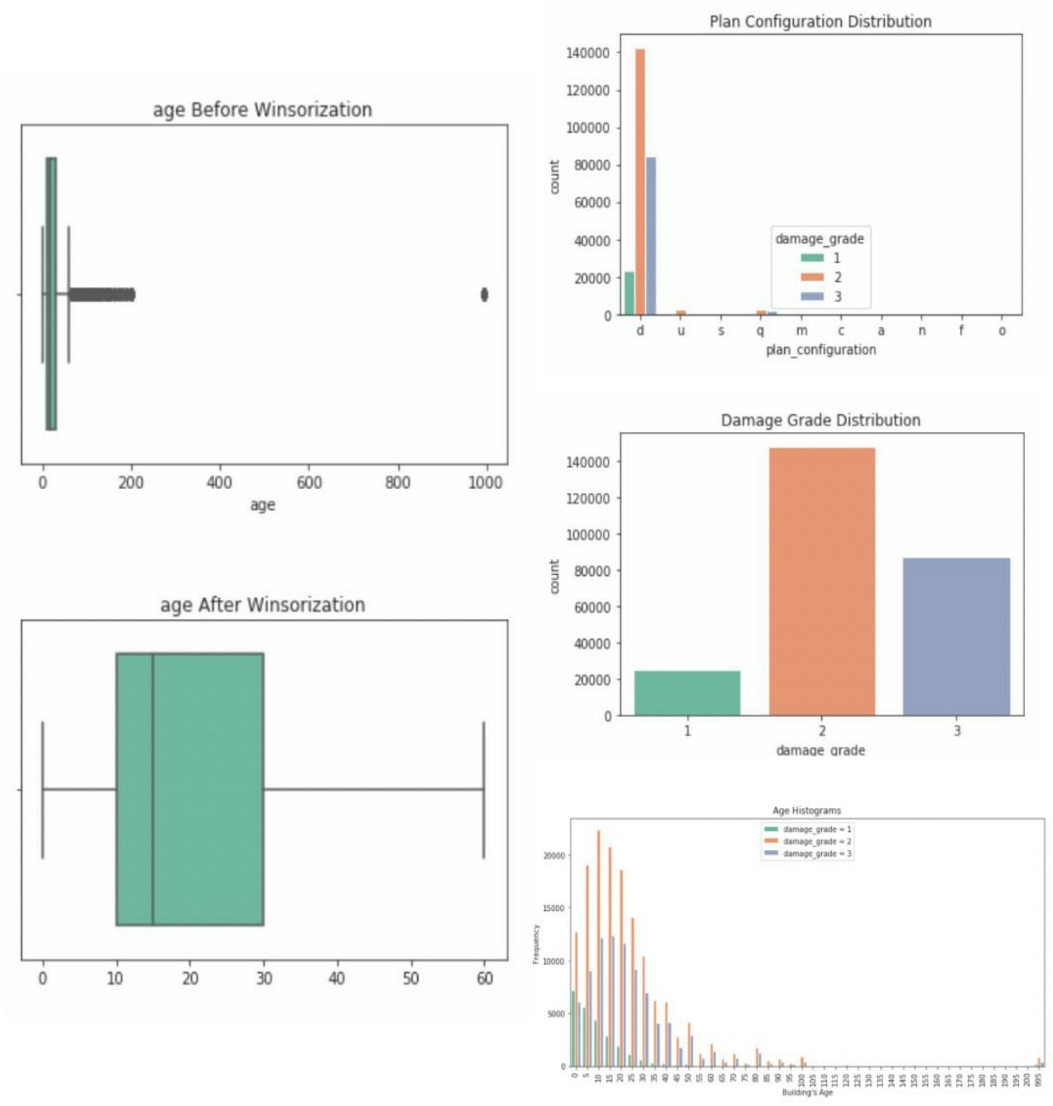

Figure 1: EDA

- **Age:** By plotting a histogram of building ages for each damage grade, we can identify possible correlations between building age and the extent of the damage. Buildings in the dataset range from new constructions to those around 995 years old, which initially seems like a data collection error. However, to avoid skewed results due to extreme values, we perform Winsorization (4) on the *age* feature, capping it at the 95th percentile.
- **Land Surface Condition:** Investigating the distribution of land surface conditions for each damage grade can reveal potential relationships between these conditions and the earthquake's impact. We can assess if certain land surface conditions are more prone to damage and consequently devise mitigation strategies to minimize future risks.
- **Plan Configuration:** Examining the plan configuration distribution for each damage grade can help us understand if specific configurations are more susceptible to damage. For instance, buildings with a *D* type configuration might be more vulnerable to earthquake impact, which can guide structural design choices and risk assessments.
- **Superstructure and Secondary Usage Features:** Analyzing the distribution of different superstructure and secondary usage features, such as

has_superstructure_adobe_mud, *has_superstructure_mud_mortar_stone*, *has_secondary_use_agriculture*, and others, helps us better understand the impact of building materials and secondary usages on the damage grade. By visualizing the frequency of these features across different damage grades, we can identify patterns and potential vulnerabilities associated with specific building materials and usages.

2. **Outlier Treatment:** We apply the Winsorization technique (4) to the *age*, *height_percentage*, and *area_percentage* features to minimize the impact of extreme values on our model's performance. By capping these features at the 95th percentile, we ensure that outliers do not overly influence our analysis.
3. **Damage Grade Distribution:** To address class imbalance issues, we considered resampling techniques, such as oversampling the minority class or undersampling the majority class, ensuring our model can learn from a more balanced dataset. However, resampling did not work well for us.
4. **Missing Values:** Upon inspection, we found no missing values in the dataset, eliminating the need for imputation or other missing data handling techniques.
5. **Dimensionality Reduction:** Due to the presence of binary variables that do not adhere to the coordinate system, techniques like Principal Component Analysis (PCA) were not suitable for reducing the dimensions of the dataset.

4 Feature Engineering

In addition to the EDA, we performed feature engineering to prepare the dataset for further analysis and modeling.

1. **Dropping Building ID:** We remove the *building_id* feature from the dataset to avoid misleading the model, as this feature is a unique identifier and does not contribute any valuable information for predicting the damage grades.
2. **Studying Feature Importances:** After building a model, we analyze feature importances to understand which features contribute the most to the model's performance. Examining the correlation between features and the damage grade target variable helps identify potential multicollinearity issues and provides insights into the relationships between variables, allowing for better feature selection and model interpretability. It lead to loss of information and did not give up improved results.
3. **Creating the Volume Feature:** To better represent the size of each building, we create a new *volume* feature by multiplying *area_percentage* and *height_percentage*. This new feature can provide additional insights into the relationship between building size and earthquake damage.
4. **Geo Embedding:** We adopted geo-embeddings (2) to capture the hierarchical relationship between the *geo_level_1_id*, *geo_level_2_id*, and *geo_level_3_id* features. We create a neural network-based embedding model that maps each geo-level to a low-dimensional space. This approach can help us extract meaningful spatial information from these features, improving our model's ability to predict the damage grades.

In summary, our EDA and feature engineering involve analyzing various trends in the dataset, treating outliers, addressing the class imbalance, selecting relevant features, and confirming the absence of missing values. Additionally, we found that PCA is not a viable option for dimensionality reduction due to the presence of binary variables in the dataset. These steps provide a robust foundation for building and evaluating predictive models on the Nepal Earthquake 2015 dataset (3).

5 Experiments

5.1 Modeling

In this section, we present our approach to modeling for the task of classification using machine learning algorithms. Our goal was to develop a highly accurate model that could classify the data with high precision and recall.

Iteration	Incremental Improvements	F1 score
Logistic Regression	EDA, PCA for Noise Reduction	0.5150
Random Forest	Grid Search on Benchmark code	0.5815
Catboost	Visual Feature Engineering , Feature Importance	0.7423
XGBoost	Winsorization , GBDT and DART boosting types	0.7400
LightGBM	Resampling, Geo-embedding using a neural network	0.7497
LightGBM with Optuna and Cross Validation	5 fold cross validation, Optuna (Bayesian Hyperparameter-tuning method)	**0.7519**

Table 1: Experimental results

Initially, we explored the use of Logistic Regression (6), a commonly used classification algorithm. However, we found that it did not perform well on our data, possibly due to the complexity and non-linearity of the features.

Next, we tried Random Forest (8) with Grid Search (10) on Benchmark code, which improved our results slightly. However, we wanted to try other algorithms as well to see if we could further improve the accuracy.

We then used CatBoost (7) with visual feature engineering and feature importance techniques to identify the most important features for classification. This resulted in a slight improvement in our results.

Additionally, we used XGBoost (5) with Winsorization (4) and GBDT (12) and DART (11) boosting types. These techniques allowed us to handle outliers and improve the performance of our model. However, we still did not achieve the desired level of accuracy.

To further improve our results, we tried using LightGBM (9) with resampling and geo-embedding (2) using a neural network. This approach helped us to deal with class imbalance and resulted in a significant improvement in the accuracy of our predictions.

Finally, we used LightGBM (9) with 5-fold cross-validation and Optuna (1) (Bayesian Hyperparameter-tuning method) to fine-tune our model and optimize the hyperparameters to achieve the best possible results.

Overall, our modeling approach involved a combination of various machine learning algorithms and techniques to identify the best possible model for our classification task. The use of multiple algorithms allowed us to leverage the strengths of each and improve the overall performance of our model.

5.2 Results

We have achieved competitive results with our approach, which are shown in table 1. We can see that Logistic regression, Random Forest, Catboost and XGBoost achieved F1 scores of 0.5150, 0.5815, 0.7423, 0.7400 respectively. We see significant improvements with LightGBM, which achieved a score of 0.7497. Finally, we see that our approach which uses Optuna and 5-fold cross validation achieved the best score of 0.7519.

6 Conclusion

In conclusion, the project involved manual feature engineering and hyperparameter tuning, which requires domain expertise and multiple trials to achieve the best results. The team tried various classification models, including Logistic Regression (6), Random Forest (8), CatBoost (7), XGBoost (5), and LightGBM (9), and found that geo-embedding (2) with LightGBM (9) with Optuna (1) and 5 fold Cross Validation models were more effective in classifying the data that gave the highest F1 score. Moving forward, there are several avenues for future work based on the results of this project. One potential direction is to explore the use of ensemble models to improve the classification performance to create a more accurate and robust system. Overall, there is still much room for improvement and refinement in this area of classification, and continued research is needed to push the boundaries of what is possible with these methods.

References

[1] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. CoRR **abs/1907.10902** (2019), `http://arxiv.org/abs/1907.10902`

[2] Baran, K.: Goodsea/richter-s-eye: Richter's predictor: Modeling earthquake damage challenge 0.7521 scored solution (2020), `https://github.com/Goodsea/Richter-s-Eye`

[3] Bull, P., Slavitt, I., Lipstein, G.: Harnessing the power of the crowd to increase capacity for data science in the social sector (2016)

[4] Chambers, R., Kokic, P., Smith, P., Cruddas, M.: Winsorization for identifying and treating outliers in business surveys. Proceedings of the Second International Conference on Establishment Surveys pp. 717–726 (01 2000)

[5] Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794. KDD '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939785, `http://doi.acm.org/10.1145/2939672.2939785`

[6] Cox, D.R.: The regression analysis of binary sequences. Journal of the Royal Statistical Society: Series B (Methodological) **20**(2), 215–232 (1958)

[7] Dorogush, A.V., Ershov, V., Gulin, A.: Catboost: gradient boosting with categorical features support. CoRR **abs/1810.11363** (2018), `http://arxiv.org/abs/1810.11363`

[8] Ho, T.K.: Random decision forests. In: Proceedings of 3rd international conference on document analysis and recognition. vol. 1, pp. 278–282. IEEE (1995)

[9] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems **30**, 3146–3154 (2017)

[10] LaValle, S.M., Branicky, M.S., Lindemann, S.R.: On the relationship between classical grid search and probabilistic roadmaps. The International Journal of Robotics Research **23**(7-8), 673–692 (2004)

[11] Rashmi, K.V., Gilad-Bachrach, R.: DART: dropouts meet multiple additive regression trees. CoRR **abs/1505.01866** (2015), `http://arxiv.org/abs/1505.01866`

[12] Shi, Y., Ke, G., Chen, Z., Zheng, S., Liu, T.Y.: Quantized training of gradient boosting decision trees (2023)

A Appendix

This appendix provides an overview of the directory structure and file locations used in the code. Consider that the submitted code is present in a folder named CSCI567.

1. **Data Files:** The data files are located in the Data folder CSCI567/Data/file_name.csv
   - Train Values,Train Labels,Test Values,Submission Format
2. **Model Files:**
   - GEO Embedding Model: geo_embed.h5 (Saved locally in the Colab runtime)
   - LightGBM Models: CSCI567/Data/models/modeli.txt (where i is the index of the model in the ensemble, ranging from 0 to 4)
3. **Submission File:**
   - Submission File: submission.csv (Saved locally in the Colab runtime, and downloaded to the local machine using files.download("submission.csv"))

The code should ideally run on Google Colab GPU with the above-mentioned directory structure on the drive.