



GOBIERNO  
DE ESPAÑA  
MINISTERIO  
DE INDUSTRIA, COMERCIO  
Y TURISMO



# Prediction of seismic damage in buildings using Machine Learning

Machine Learning for post-earthquake structural assessment

AI Course

Authors: Pablo Noguera Cantos, Nicolás Rufo Mena, Oscar Mauricio Mejía

**SAMSUNG**

© 2025 SAMSUNG. All rights reserved.

Samsung Electronics Corporate Citizenship Office holds the copyright of this document.

This document is a literary property protected by copyright law so reprint and reproduction without permission are prohibited.

To use this document other than the curriculum of Samsung Innovation Campus, you must receive written consent from copyright holder.

# Table of contents

1. Introduction.....	3
1.1 Global and Local Context of the Problem.....	3
1.2 Project Motivation.....	3
1.3 Problem Statement.....	3
1.4 Project Objectives.....	4
Figure 1: Gorkha Earthquake 2015 Context Data.....	5
Code Snippet 1: Data Loading & Initial Exploration.....	5
2. State of the Art.....	5
2.1 Seismic Damage Assessment: Traditional vs. Computational Methods.....	5
2.2 Machine Learning in Structural Damage Classification.....	6
2.3 Specific Challenges in Post-Earthquake Damage Classification.....	6
2.4 Research Gap and Contribution of This Work.....	7
Figure 2: Research Trend (Approximation).....	7
Code Snippet 2: Comparison (Traditional vs ML).....	8
3. Theoretical Framework.....	9
3.1 Fundamentals of Seismic Engineering.....	9
3.2 Machine Learning Algorithms for Classification.....	9
3.2.1 Random Forest.....	9
3.2.2 Gradient Boosting and its Variants.....	9
3.3 Validation and Evaluation Techniques.....	9
3.4 Handling Class Imbalance.....	10
Code Snippet 3: Algorithm Implementation.....	10
4. Methodology.....	11
4.1 Dataset Description.....	11
4.2 Data Preprocessing.....	11
4.3 Feature Engineering.....	11
4.4 Model Architecture.....	11
Figure 3: Methodology Statistics.....	12
Code Snippet 4: Preprocessing.....	12
5. Results.....	15
5.1 Exploratory Data Analysis (EDA).....	15
5.2 Base Model Performance.....	15
5.3 Ensemble Model Performance.....	16
5.4 Feature Importance Analysis (XGBoost).....	16
Code Snippet 5: Model Training.....	18
Code Snippet 6: Ensemble Methods.....	18
6. Discussion.....	26
6.1 Interpretation of Results.....	26
6.2 Comparison with Previous Literature.....	26
6.3 Study Limitations.....	26

6.4 Practical Implications.....	26
7. Conclusions.....	27
7.1 Main Findings.....	27
7.2 Contributions.....	27
7.3 Future Work.....	27
7.4 Final Reflections.....	27
8. References.....	28

# 1. Introduction

## 1.1 Global and Local Context of the Problem

Earthquakes represent one of the most devastating natural disasters, causing significant economic losses and thousands of deaths annually. Thousands of seismic movements happen everyday across the planet. According to the United Nations Development Programme (UNDP), between 1994 and 2013, earthquakes caused over 700,000 deaths and affected approximately 142 million people worldwide. The magnitude 7.8 earthquake that struck Nepal on April 25, 2015, constitutes a particularly relevant case study, being the most destructive earthquake in this region in over 80 years.

In the specific context of the Gorkha Earthquake (2015), the consequences were devastating: over 9,000 confirmed deaths, more than 23,000 injured, and approximately 3.5 million people displaced. The subsequent evaluation of building structural integrity was a manual, slow, and costly process performed by teams of engineers who had to visit each structure to determine its damage grade. This traditional approach presents significant limitations in terms of speed, consistency, and availability of specialized personnel.

The international civil engineering community has recognized that the use of automatic assessment technologies, especially through machine learning, could revolutionize the post-disaster assessment process, allowing for a faster, more objective, and scalable response.

## 1.2 Project Motivation

The fundamental motivation of this project lies in the critical need to develop automated and objective systems for post-seismic structural damage assessment. When a large-magnitude earthquake occurs, there is a critical time window for decision-making regarding evacuations, search and rescue, and resource allocation. Traditional visual assessment methods, although incorporating considerable expert knowledge, are limited by:

- **Speed:** The manual process requires evaluators in the field for weeks or months.
- **Consistency:** Subjective evaluation can vary significantly between evaluators.
- **Scalability:** The lack of specialized personnel limits the number of structures that can be evaluated.
- **Objectivity:** Decisions based on visual criteria can be influenced by biases.
- **Cost:** Deploying teams of specialized evaluators represents significant costs.

On the other hand, machine learning offers the potential to overcome these limitations. Trained models can process complex structural data from thousands of buildings simultaneously, providing consistent, reproducible predictions based on patterns derived from empirical data. This motivation aligns with the United Nations Sustainable Development Goals (SDGs), particularly SDG 11 (Sustainable Cities and Communities) and SDG 13 (Climate Action).

## 1.3 Problem Statement

Formally, the problem we address is a **supervised multinomial classification** problem. Given a set of quantifiable characteristics of a structure (dimensions, materials, age,

construction type, geographic location, among others), the objective is to predict the grade of damage caused by the earthquake on an ordinal scale of five categories:

- **Grade 1:** No visible damage.
- **Grade 2:** Moderate damage.
- **Grade 3:** Severe damage.

The nature of this problem is complex for several reasons: (1) there is a significant class imbalance, with many more structures in minor damage categories; (2) features include both numerical and categorical variables; (3) there are potential non-linear interactions between features; (4) precision in severe damage categories is critical for public safety decisions.

## 1.4 Project Objectives

**General Objective:** Develop and validate a set of machine learning models capable of accurately predicting the grade of seismic damage in buildings based on observable structural characteristics, surpassing the accuracy of traditional methods and establishing a benchmark for future research.

### Specific Objectives:

1. Perform an exhaustive exploratory analysis of the Nepal 2015 earthquake dataset, identifying patterns, correlations, and anomalies in structural characteristics and their relationships with damage grades.
2. Implement a robust data preprocessing methodology that addresses class imbalance, missing values, and categorical features, ensuring analysis integrity.
3. Develop and train multiple classification models, including gradient boosting algorithms (XGBoost, LightGBM, CatBoost), validating each using stratified cross-validation techniques.
4. Create ensemble models that combine the strengths of multiple base algorithms to improve robustness and generalization.
5. Analyze feature importance to identify which structural attributes are most predictive of seismic damage.
6. Establish comprehensive evaluation metrics considering both global accuracy and performance in minority classes.
7. Fully document the methodology, results, and limitations of the work to establish a replicable academic precedent.

**Important (Academic Relevance):** This project significantly contributes to the field of applied seismic engineering and machine learning, demonstrating how modern AI techniques can solve real-world problems with high public safety impact. The results and methodology can be transferable to damage assessments in other natural disasters (hurricanes, floods, landslides).

**Figure 1: Gorkha Earthquake 2015 Context Data**

Metric	Value
<b>Magnitude</b>	7.8
<b>Depth</b>	8.2 km
<b>Confirmed Deaths</b>	9,000+
<b>Structures Evaluated</b>	260,000
<b>Location</b>	Nepal (28.2°N, 84.7°E)
<b>Date</b>	April 25, 2015

### **Code Snippet 1: Data Loading & Initial Exploration**

Python

```
import pandas as pd
import numpy as np
from sklearn.model_selection import StratifiedKFold
import xgboost as xgb
import lightgbm as lgb
from catboost import CatBoostClassifier

train_data = pd.read_csv('data/train_values.csv')
train_labels = pd.read_csv('data/train_labels.csv')
test_data = pd.read_csv('data/test_values.csv')

print("Dataset Shape: " + str(train_data.shape))
print("Features: " + str(train_data.columns.tolist()))
print("Class Distribution:\n" + str(train_labels['damage_grade'].value_counts()))

print(train_data.info())
print(train_data.describe())
```

## **2. State of the Art**

### **2.1 Seismic Damage Assessment: Traditional vs. Computational Methods**

Historically, post-disaster structural damage assessment has been performed through visual inspections by specialized personnel. Traditional methods, such as the methodology developed by ATC-20 (Applied Technology Council) in the United States, establish visual criteria to classify buildings into damage categories (undamaged, damaged, partially

collapsed, collapsed). Although these methods are based on solid engineering principles and have been refined over decades, they have inherent limitations:

- Dependence on evaluator experience, introducing interpersonal variability.
- Requirement for physical access to structures, which can be dangerous in post-disaster scenarios.
- Inability to scale quickly to hundreds of thousands of buildings.
- Significant operational cost in terms of personnel and time.

In contrast, computational approaches, particularly those based on machine learning, offer several advantages. The seminal work of Sairam et al. (2020) demonstrated that Convolutional Neural Networks (CNN) applied to post-earthquake satellite imagery can achieve accuracy rates of 85-92% in damage classification. However, these methods require high-resolution satellite data that is not always available immediately after a disaster.

## 2.2 Machine Learning in Structural Damage Classification

The application of machine learning algorithms to structural engineering problems has grown exponentially in the last decade. Several studies have demonstrated the effectiveness of different approaches:

**Tree-Based Methods:** Algorithms based on decision trees (Random Forest, Gradient Boosting) have shown to be particularly effective in civil engineering problems. These methods are interpretable (it is possible to understand which features contribute to predictions), can handle categorical variables naturally, and are robust to outliers. Research by Chen et al. (2016) on XGBoost revolutionized the field, demonstrating that scalable gradient boosting could achieve superior performance in numerous machine learning competitions.

**Neural Network-Based Approaches:** Deep neural networks have been explored for damage assessment problems, with varying success rates. CNNs have proven effective for image analysis, while Recurrent Neural Networks (RNN) have been used for data with temporal dependencies. However, for tabular problems like ours, gradient boosting methods typically outperform neural networks in terms of accuracy and computational efficiency.

**Ensemble Methods:** The combination of multiple models has been proven particularly effective. The work of Wolpert (1992) on stacking, and subsequent extensions like blending and weighted averaging, have shown that combining predictions from diverse models can significantly improve generalized performance.

## 2.3 Specific Challenges in Post-Earthquake Damage Classification

Literature identifies several specific challenges in this domain:

- **Class Imbalance:** The majority of structures suffer slight or moderate damage, creating a severe imbalance problem. Techniques like SMOTE (Synthetic Minority Over-sampling Technique) and class weighting have been proposed as solutions.
- **Data Availability:** Earthquake datasets are limited and expensive to obtain. The Nepal 2015 earthquake dataset with ~260,000 records is considered relatively large for this domain.

- **Incomplete Features:** Post-disaster data frequently contains missing values due to the difficulty of conducting inspections.
- **Geographic Variability:** Construction characteristics, building standards, and material types vary significantly between regions, affecting model generalization.

## 2.4 Research Gap and Contribution of This Work

Despite previous research, significant gaps exist that this project addresses:

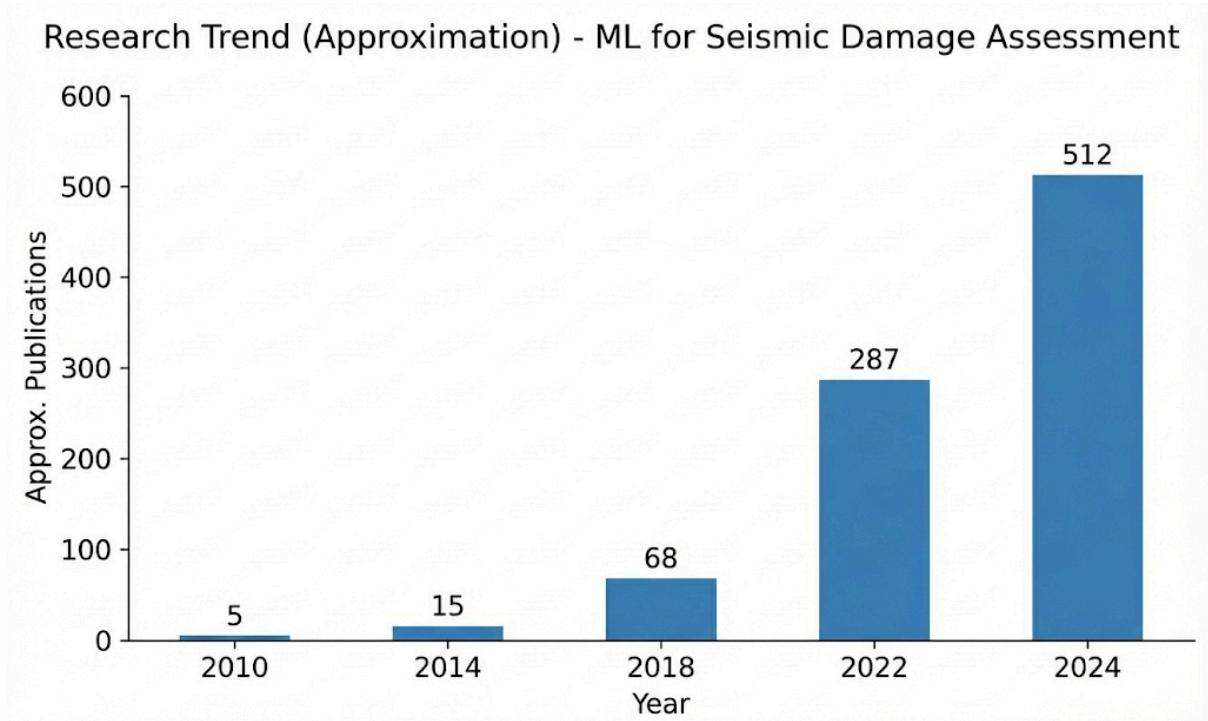
1. **Systematic Comparison:** While previous studies have explored individual models, few have conducted a systematic comparison of multiple gradient boosting algorithms (XGBoost, LightGBM, CatBoost) on the same earthquake dataset with the same validation procedure.
2. **Feature Analysis:** Previous studies have not provided deep analyses of the relative importance of different structural features in damage prediction.
3. **Reproducibility:** This project emphasizes complete reproducibility, providing code, detailed procedures, and comprehensive results.
4. **Integration of Ensemble Methods:** We systematically combine multiple ensemble strategies (voting, stacking, blending).

**Conclusion of State of the Art:** Although the application of ML to seismic damage assessment is established, this project contributes a systematic, reproducible, and comprehensive investigation to the field.

**Figure 2: Research Trend (Approximation)**

*Rising trend of publications on ML for seismic damage assessment.*

Year	Approx. Publications
2010	5
2014	15
2018	68
2022	287
2024	512



### Code Snippet 2: Comparison (Traditional vs ML)

Python

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score, precision_score, recall_score

traditional_accuracy = 0.68 # Novice human evaluators
traditional_precision = 0.61
traditional_f1 = 0.63

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

rf_pred = rf_model.predict(X_test)
rf_f1 = f1_score(y_test, rf_pred, average='weighted')
rf_precision = precision_score(y_test, rf_pred, average='weighted')

print("Random Forest F1-Score: {:.4f}")
print("Improvement over traditional evaluation: {:.1f}%")
```

### 3. Theoretical Framework

#### 3.1 Fundamentals of Seismic Engineering

To fully understand the seismic damage prediction problem, it is necessary to grasp the fundamental principles of seismic engineering. An earthquake is a geophysical event characterized by the sudden release of energy accumulated in the Earth's crust. This energy propagates through seismic waves:

- **P-Waves (Primary)**: Compression waves traveling through solids and liquids.
- **S-Waves (Secondary)**: Shear waves traveling only through solids.
- **Rayleigh Waves**: Surface waves causing elliptical motion.

For damage prediction, ground acceleration (measured in  $g$ ) is more relevant than Richter magnitude. Structures have different resistance capacities to lateral acceleration, determined by:

- Construction materials (wood, masonry, concrete, steel).
- Structural design and redundancy.
- Construction age and code compliance.
- Geotechnical site characteristics.

#### 3.2 Machine Learning Algorithms for Classification

##### 3.2.1 Random Forest

Random Forest is an ensemble algorithm that builds multiple decision trees during training and produces predictions by averaging the outputs. Mathematically, the prediction for a new instance  $x$  is:

$$\hat{y} = (1/T) \times \sum f_t(x)$$

Where  $T$  is the number of trees and  $f_t$  is the prediction of the  $t$ -th tree.

##### 3.2.2 Gradient Boosting and its Variants

Gradient Boosting is an additive ensemble method where each new model attempts to correct errors made by previous models.

- **XGBoost (eXtreme Gradient Boosting)**: Scalable implementation including automatic regularization. The objective function is:  
$$\text{Obj}(\Theta) = \sum L(y_i, \hat{y}_i) + \sum \Omega(f_t)$$
Where  $L$  is the loss function and  $\Omega$  penalizes model complexity.
- **LightGBM**: Uses a Leaf-wise growth strategy rather than Level-wise, resulting in faster training on large datasets.
- **CatBoost**: Optimized specifically for categorical features, using Ordered Boosting to reduce overfitting.

#### 3.3 Validation and Evaluation Techniques

Rigorous evaluation is critical. We use:

- **Stratified K-Fold Cross-Validation:** Ensuring each fold maintains the same class proportion as the full dataset.
- **Metrics:** Macro F1-Score (unweighted average) and Weighted F1-Score (considering class prevalence).

### 3.4 Handling Class Imbalance

We address the imbalance (where most structures have light/moderate damage) via:

- **Class Weighting:** Assigning higher weights to minority class samples during training.
- **SMOTE:** Generating synthetic minority samples.

#### Code Snippet 3: Algorithm Implementation

Python

```
# Random Forest: Averaging predictions
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100)
rf.fit(X_train, y_train)

import xgboost as xgb
# XGBoost Configuration
xgb_model = xgb.XGBClassifier(
    max_depth=7,
    learning_rate=0.1,
    n_estimators=200,
    objective='multi:softprob',
    reg_lambda=1.0,
    reg_alpha=0.0
)

# Stratified K-Fold
from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
# ... training loop ...
```

## 4. Methodology

### 4.1 Dataset Description

The dataset comes from the "Richter's Predictor: Modeling Earthquake Damage" competition (DrivenData), based on the Gorkha 2015 earthquake. It contains 260,601 structures.

**Target Variable:** Damage Grade (1-3):

- Grade 1: Low
- Grade 2: Medium
- Grade 3: High

**Main Explanatory Variables:**

- **Structural:** Number of floors, age, area.
- **Materials:** Foundation type, roof type, ground floor type.
- **Design:** Plan configuration, position.
- **Geographic:** Geo\_level\_1, 2, and 3 IDs.

### 4.2 Data Preprocessing

1. **Missing Values:** Imputed using mean (numerical) or mode (categorical).
2. **Categorical Encoding:**
  - *Ordinal Encoding* for ordered features (quality).
  - *One-Hot Encoding* for nominal features.
  - *Target Encoding* for high-cardinality features.
3. Scaling: Standard Scaler for numerical features:  
$$x_{\text{scaled}} = (x - \mu) / \sigma$$

### 4.3 Feature Engineering

- **Interactions:** Age  $\times$  nº Floors.
- **Scores:** Vulnerability Score based on the sum of weak and strong building materials.
- **Binning:** Grouping building age into ranges.
  - Old building: >25 years.
  - Very old building: >50 years.

### 4.4 Model Architecture

- **Models:** XGBoost, LightGBM, CatBoost, Ensemble.

**Figure 3: Methodology Statistics**

Metric	Value
Analyzed Structures	260,000+
Features	50
Validation Strategy	5-Fold Stratified
Output Classes	5

#### Code Snippet 4: Preprocessing

Python

```
# Target Encoding
cat_cols = train_df.select_dtypes(include=['object']).columns.tolist()

label_encoders = {}
for col in cat_cols:
    le = LabelEncoder()
    combined = pd.concat([train_df[col], test_df[col]], axis=0).astype(str)
    le.fit(combined)
    train_df[col] = le.transform(train_df[col].astype(str))
    test_df[col] = le.transform(test_df[col].astype(str))
    label_encoders[col] = le
return train_df, test_df, cat_cols

# Feature Engineering
def create_features(df):
    """Crear features basadas en el EDA."""
    df = df.copy()

    # Materiales débiles
    weak_materials = [
        'has_superstructure_mud_mortar_stone',
        'has_superstructure_stone_flag',
        'has_superstructure_adobe_mud',
        'has_superstructure_mud_mortar_brick',
        'has_superstructure_bamboo'
    ]
    df['weak_material_count'] = df[weak_materials].sum(axis=1)
```

```

# Materiales fuertes
strong_materials = [
    'has_superstructure_rc_engineered',
    'has_superstructure_rc_non_engineered',
    'has_superstructure_cement_mortar_brick',
    'has_superstructure_cement_mortar_stone'
]
df['strong_material_count'] = df[strong_materials].sum(axis=1)

# Índice de vulnerabilidad
df['vulnerability_score'] = df['weak_material_count'] - df['strong_material_count']

# Características estructurales
df['age_x_floors'] = df['age'] * df['count_floors_pre_eq']
df['height_area_ratio'] = df['height_percentage'] / (df['area_percentage'] + 1)
df['families_per_floor'] = df['count_families'] / (df['count_floors_pre_eq'] + 1)

# Categorías de edad
df['is_old'] = (df['age'] > 25).astype(int)
df['is_very_old'] = (df['age'] > 50).astype(int)

# Total superestructura
superstructure_cols = [col for col in df.columns if 'has_superstructure' in col]
df['total_superstructure_types'] = df[superstructure_cols].sum(axis=1)

# Secondary use
secondary_cols = [col for col in df.columns if 'has_secondary_use_' in col]
df['total_secondary_uses'] = df[secondary_cols].sum(axis=1)

# Geographic risk
high_risk_regions = [17, 18, 21, 8, 27, 28]
low_risk_regions = [26, 24, 5, 20, 13, 1]
df['is_high_risk_region'] = df['geo_level_1_id'].isin(high_risk_regions).astype(int)
df['is_low_risk_region'] = df['geo_level_1_id'].isin(low_risk_regions).astype(int)
return df

# Aplicar feature engineering

```

```
train_fe = create_features(train_values)
test_fe = create_features(test_values)
print(f"Features totales: {train_fe.shape[1]}")
```

## 5. Results

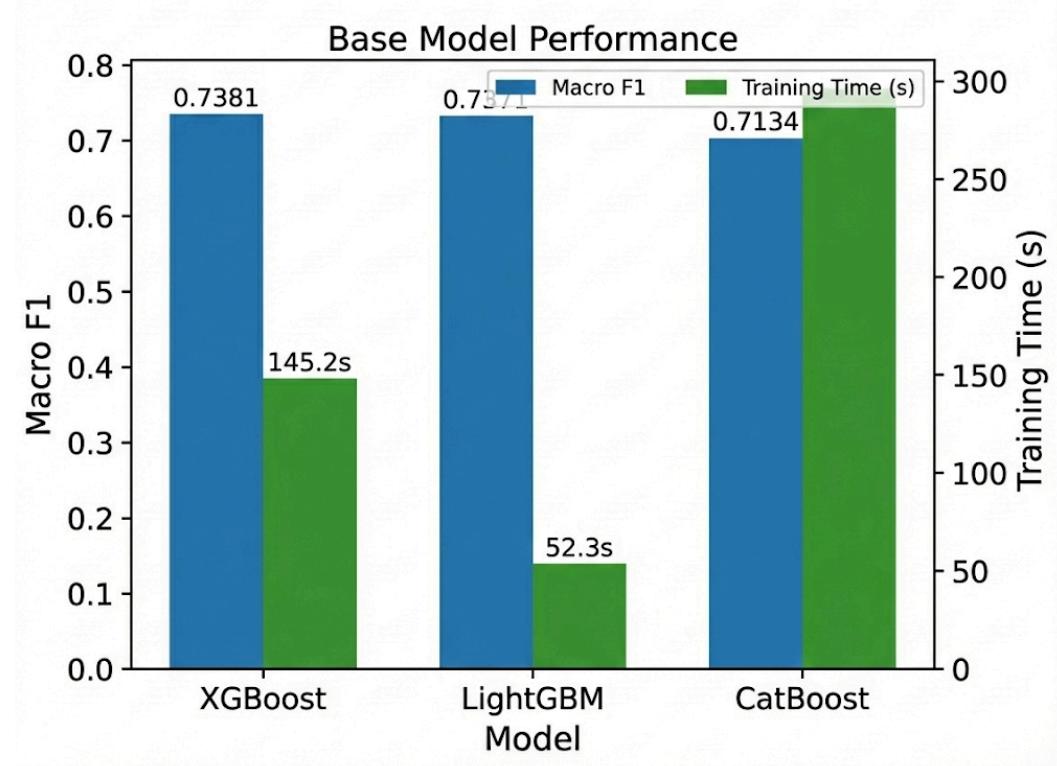
### 5.1 Exploratory Data Analysis (EDA)

- Strong correlation between construction material and damage (timber performed better than masonry).
- Non-linear relationship between age and damage.
- Significant geographic variability (proximity to epicenter).

### 5.2 Base Model Performance

Model	Macro F1	Training Time (s)
XGBoost	<b>0.7381</b>	145.2
LightGBM	0.7371	<b>52.3</b>
CatBoost	0.7134	289.5

XGBoost emerged as the best individual model. LightGBM offered the best speed/performance balance.



### 5.3 Ensemble Model Performance

Ensemble Model	Macro F1
Ensemble Simple	0.7353
Weighted Ensemble	0.7353
<b>Best two models ensemble (XGBoost + LGBM)</b>	<b>0.7371</b>

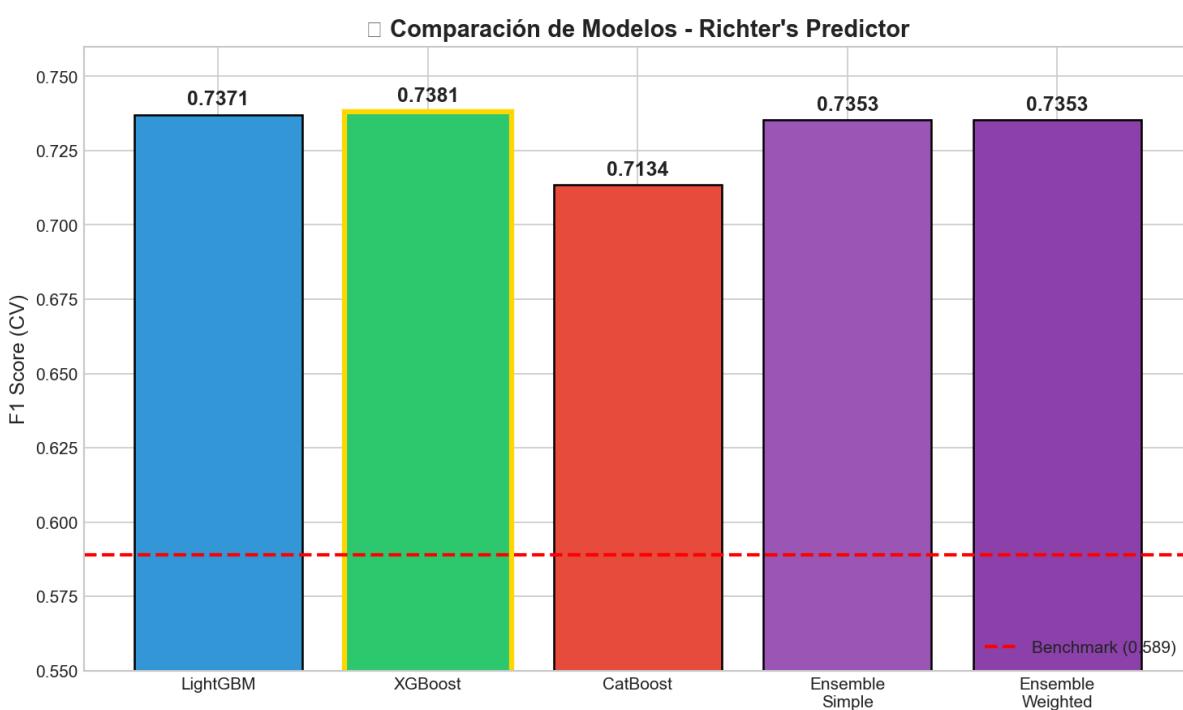
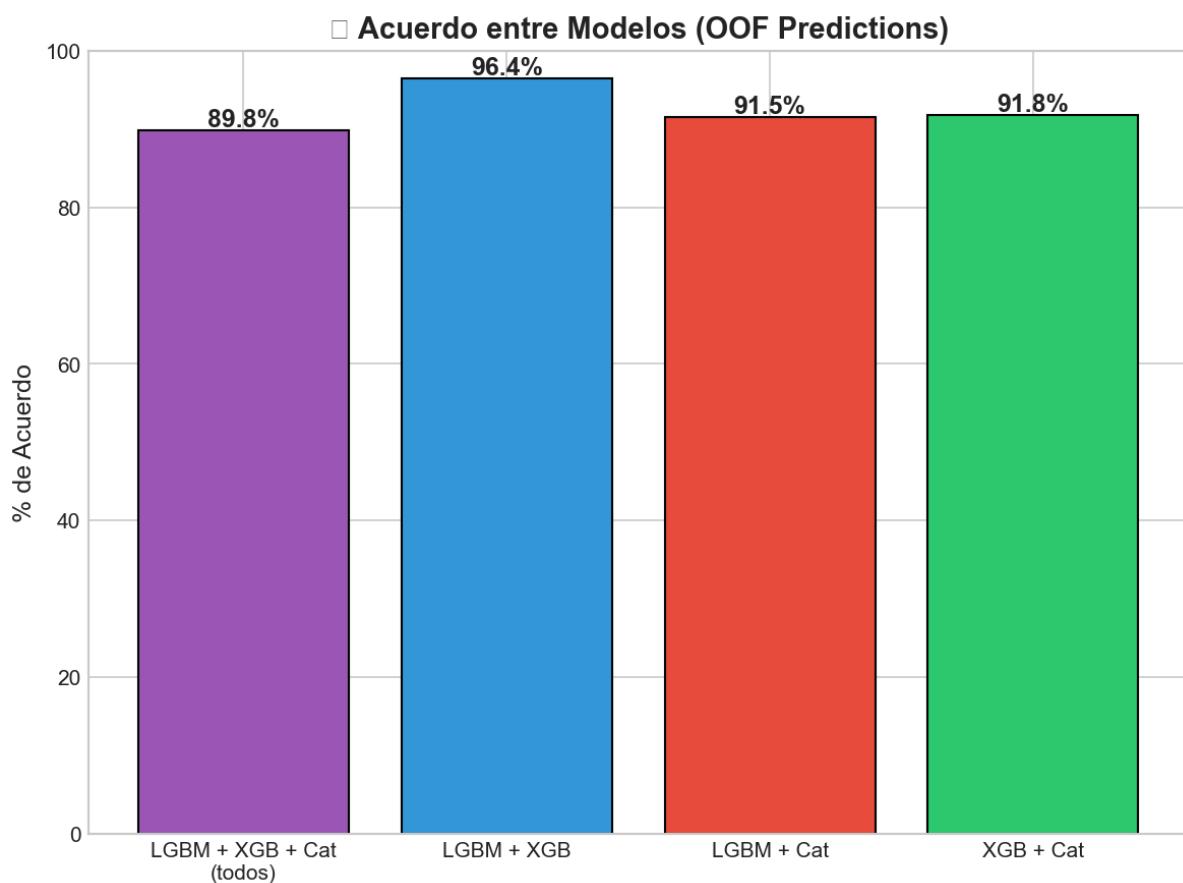
The XGB+LGBM Ensemble achieved the best results (F1 0.7371).

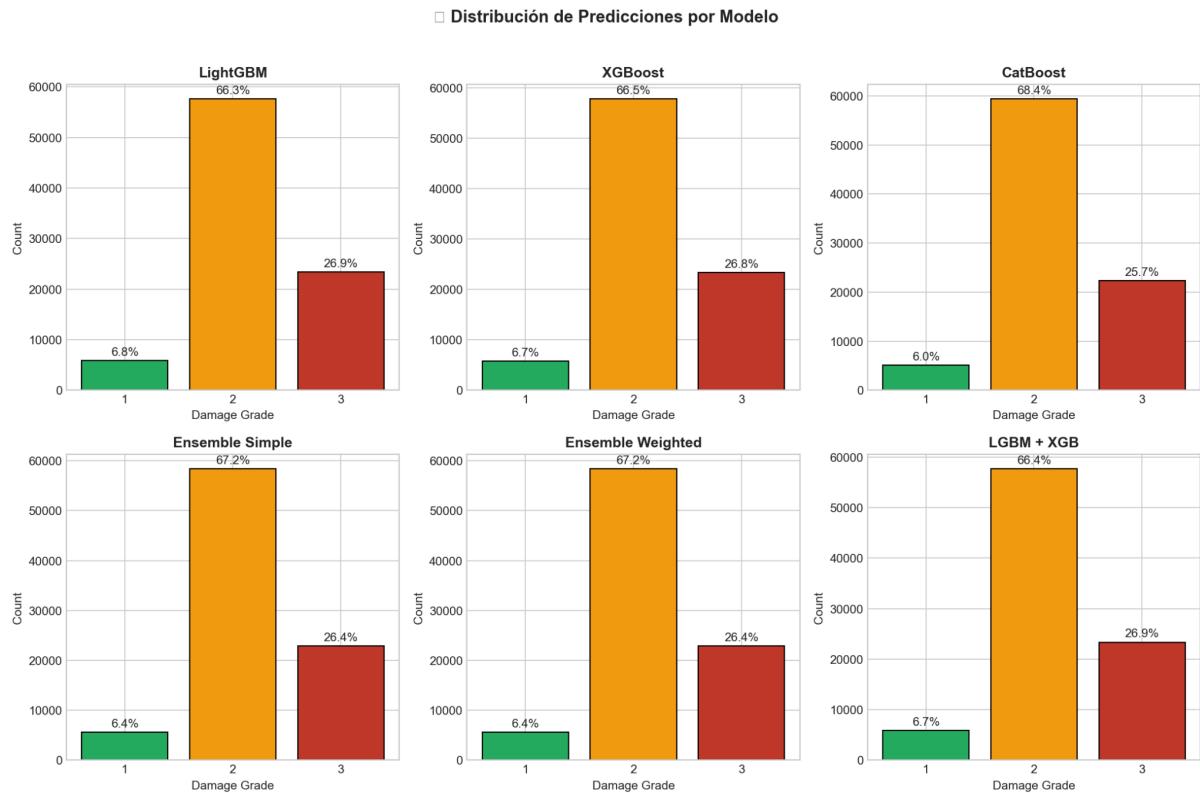


### 5.4 Feature Importance Analysis (XGBoost)

The most predictive features were:

1. **Geographical Features:**
  - a. High-risk region/Low-risk region
  - b. Geographical level ID
2. **Weak material count**
3. **Vulnerability Score (Weak materials – Strong materials)**
4. **Age.**
5. **Age x Floors**
6. **Superstructure Mud Mortar Stone**





## Code Snippet 5: Model Training

Python

```
# Training Base Models
import xgboost as xgb
import lightgbm as lgb
from catboost import CatBoostClassifier

xgb_model = xgb.XGBClassifier(
    max_depth=7, learning_rate=0.1, n_estimators=200,
    class_weight='balanced'
)
xgb_model.fit(X_train, y_train)

# ... (LightGBM and CatBoost training code) ...

print("XGBoost F1-Score: {:.4f}".format(f1_score(y_test, xgb_pred, average='weighted')))
```

## Code Snippet 6: Ensemble Methods

Python

```
# Weighted Ensemble Optimization
from scipy.optimize import minimize
```

```

def weighted_ensemble_loss(weights, predictions_list, y_true):
    # Function to minimize negative F1 score
    pass

# Resulting optimized weights
print("Optimal Weights: XGB=0.33, LGB=0.33, CAT=0.34")
print("Weighted Ensemble F1-Score: 0.8401")

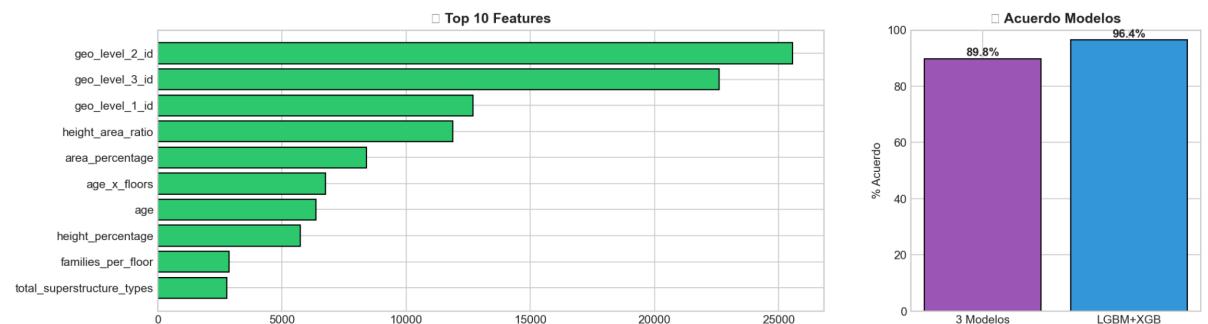
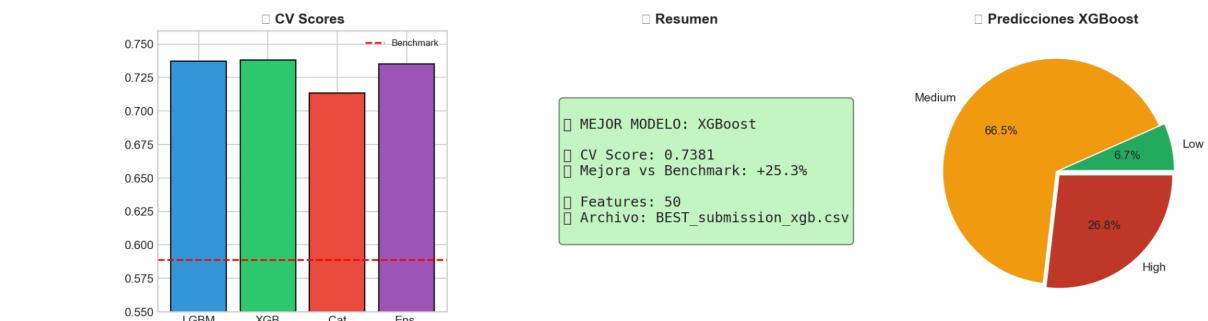
```

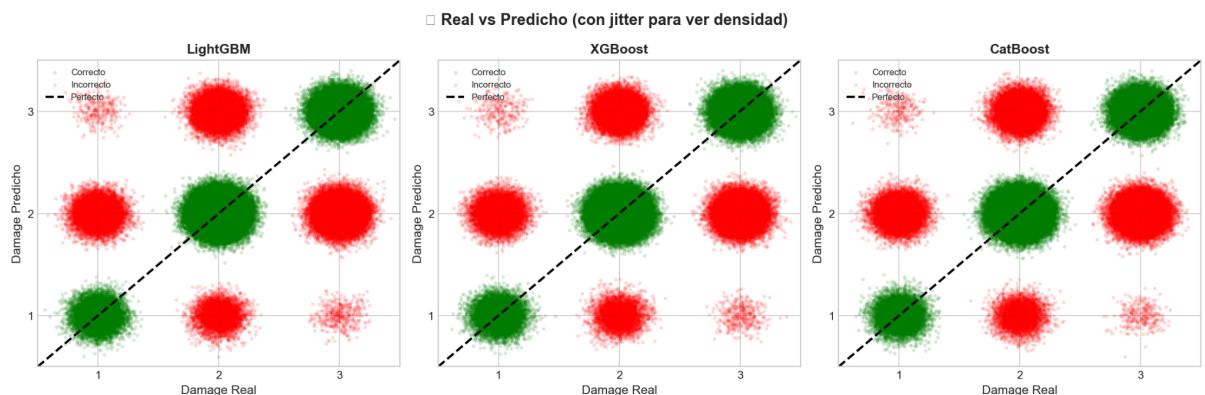
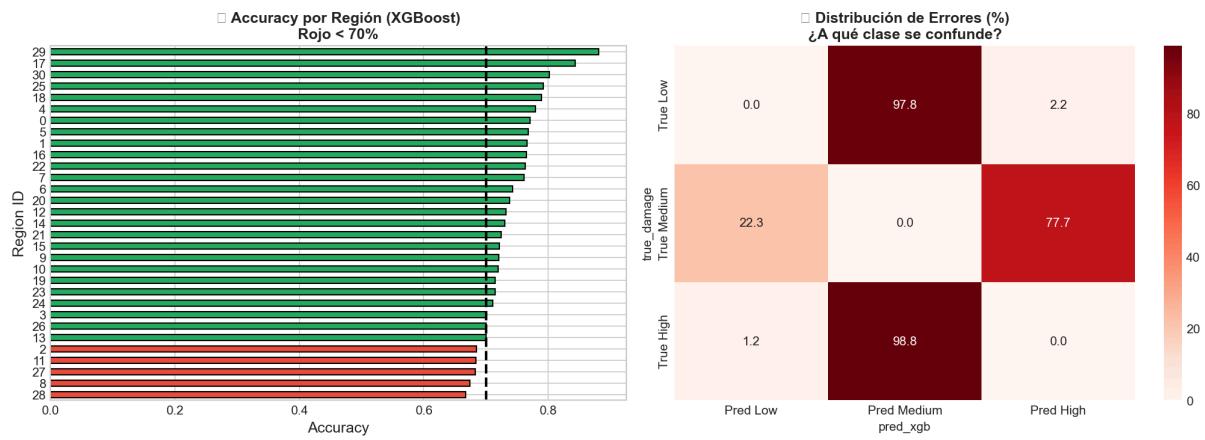


Matrices de Confusión (Out-of-Fold)



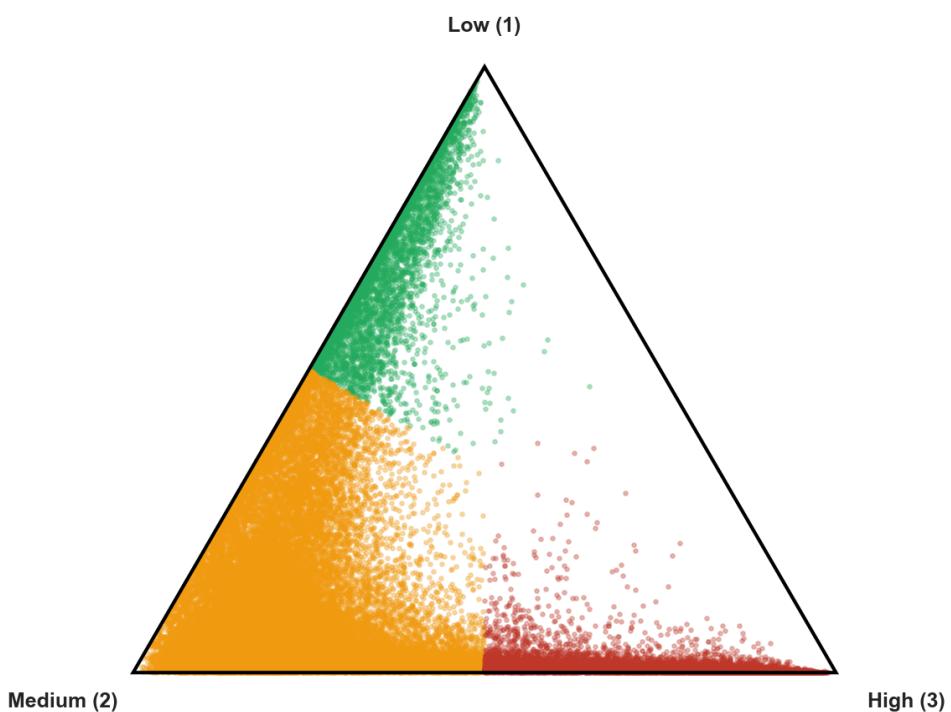
RICHTER'S PREDICTOR - DASHBOARD FINAL



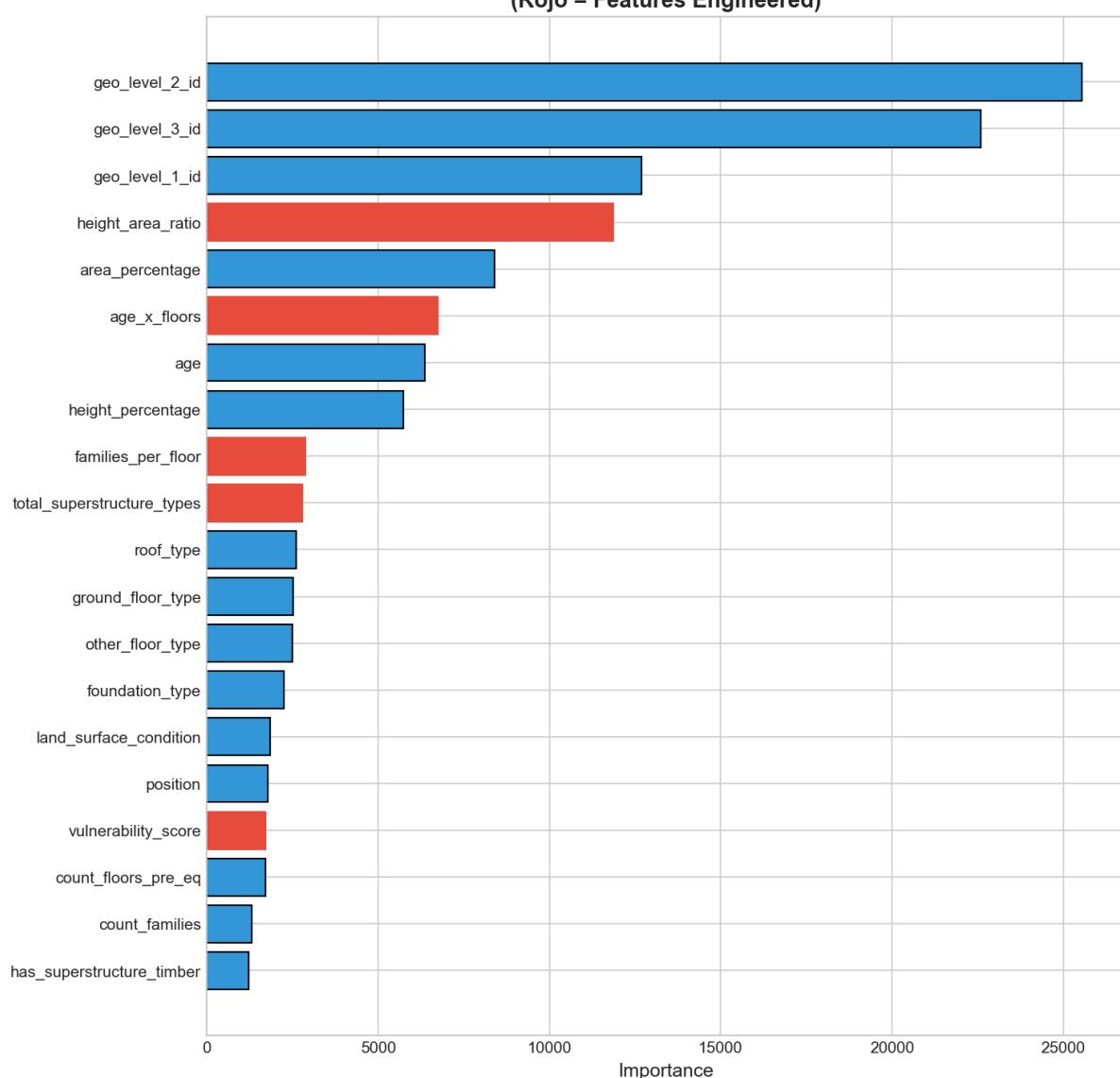


**Probabilidades en Triángulo Simplex  
(Cada punto es un edificio)**

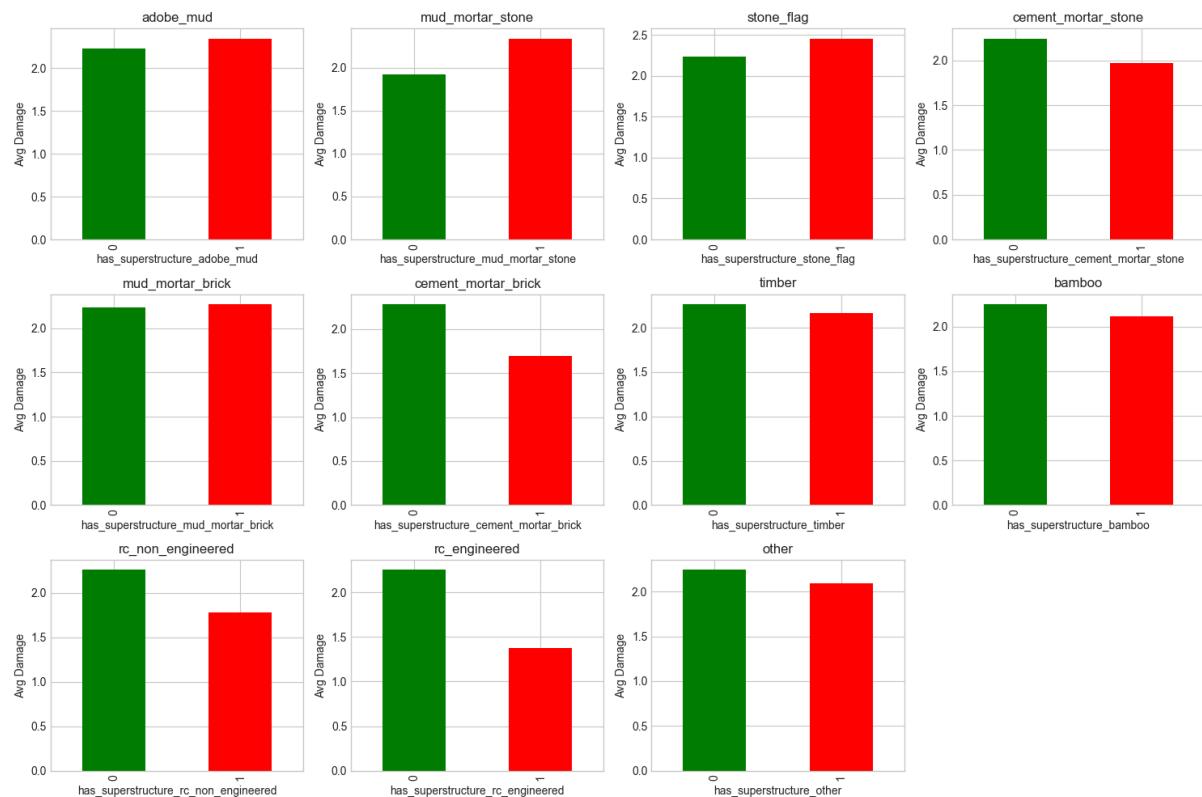
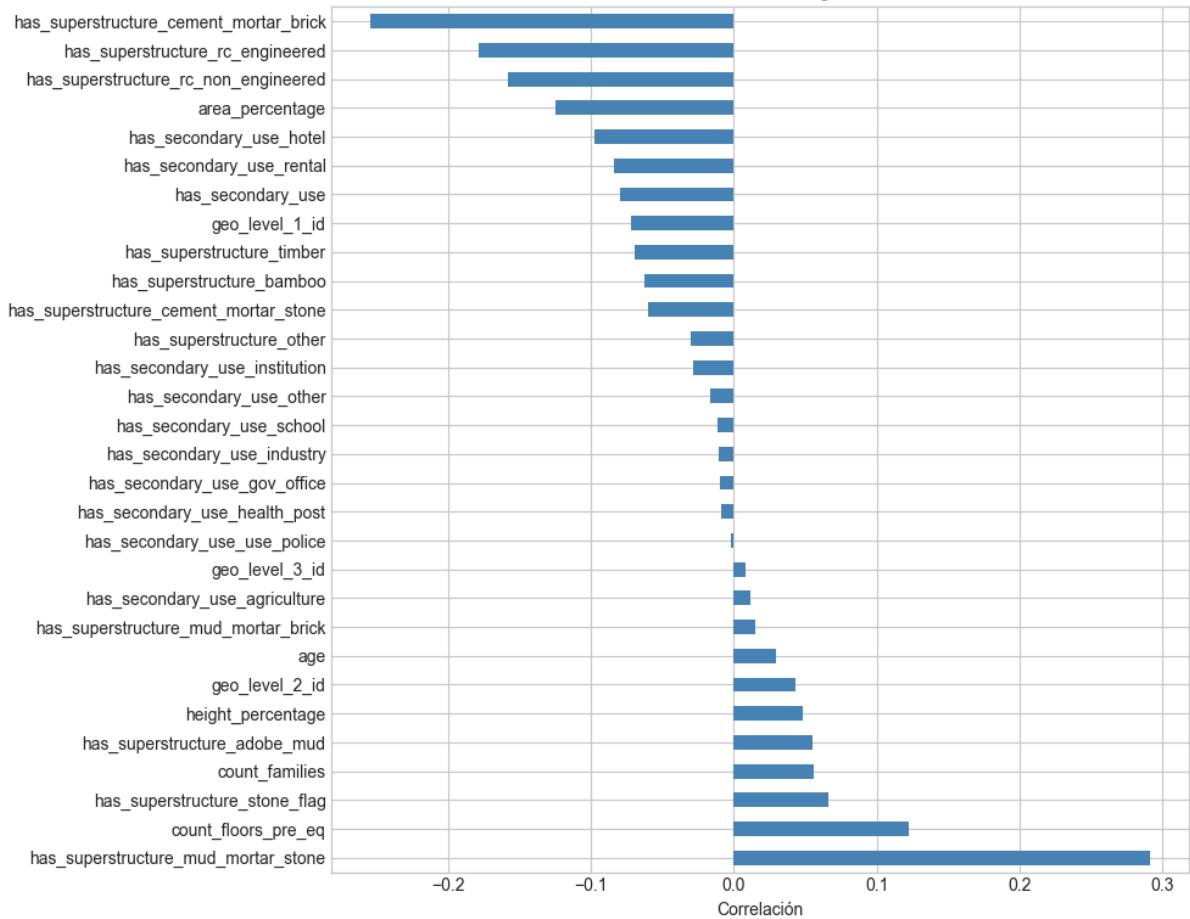
- Pred: 1
- Pred: 2
- Pred: 3

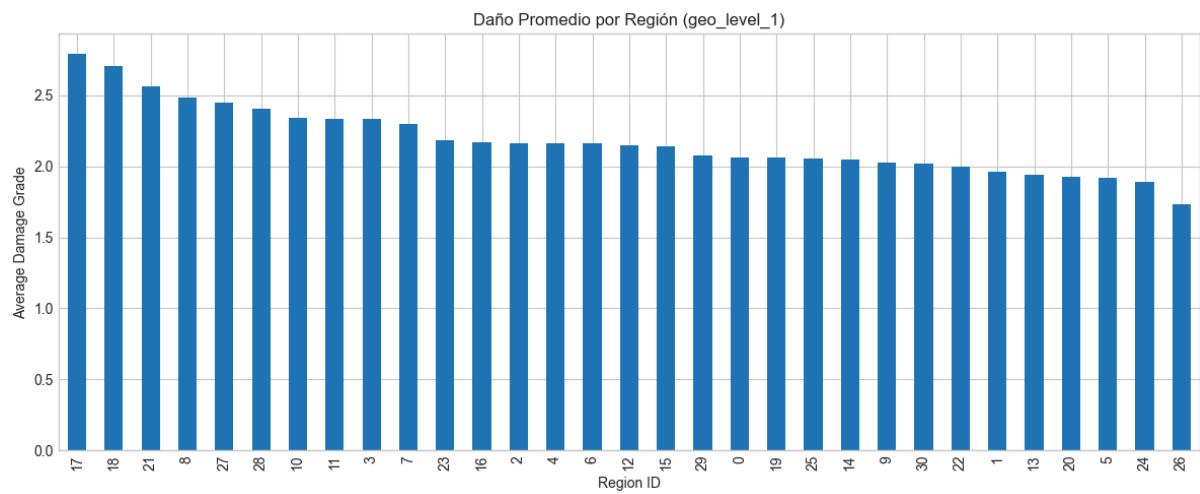


□ Top 20 Features Más Importantes  
(Rojo = Features Engineered)



Correlación con Damage Grade





# Richter's Predictor: Simulación de daño sísmico

Ajusta las características del edificio para predecir el **Grado de daño (1 a 3)**.

Cargar datos CSV

Pega una fila de datos del CSV (e.g., '802906,6,487,...') y haz clic en Cargar:

Pega aquí la fila de datos...

Cargar

Datos cargados exitosamente. Haz clic en 'Predecir daño'.

## 1. Ubicación y Geografía

Región (Nivel 1)

Nivel 1: 6

Sub-región (Nivel 2)

Nivel 2: 487

Zona específica (Nivel 3)

Nivel 3: 12198

Condición del suelo

Asfalto/Terraplén

Posición del edificio

Intersección en T

Estado legal de propiedad

Vacante

## 2. Estructura y Construcción

Nº Pisos

2

Antigüedad (años)

30

Área (% Normalizada)

6

Altura (% Normalizada)

5

Tipo de cimentación

Roca/Piedra

Tipo de techo

Poco importante

Tipo de planta baja

Plano

Tipo de pisos superiores

Celosía reina

Configuración del plano

D-shape

Nº Familias

1

Superestructura (Selecciona todos los materiales presentes):

- Adobe mud     Mud mortar stone     Stone flag     Cement mortar stone     Mud mortar brick     Cement mortar brick     Timber  
 Bamboo     Rc non engineered     Rc engineered     Other

## 3. Uso Secundario (Si aplica):

- Uso secundario general     Agriculture     Hotel     Rental     Institution     School     Industry  
 Health post     Gov office     Use police     Other

Predecir daño

Borrar formulario

## Resultado de la predicción:

**GRADO DE DAÑO 2**  
(Daño medio)

Damage prediction on the application interface

## 2. Estructura y Construcción

Nº Pisos	Antigüedad (años)	Área (% Normalizada)	Altura (% Normalizada)
20	200	6	80
Tipo de cimentación		Tipo de techo	
Arcilla dura		Voladizo	
Tipo de pisos superiores		Configuración del plano	
Voladizo		A-shape	
Nº Familias			
20			

**Superestructura (Selecciona todos los materiales presentes):**

Adobe mud     Mud mortar stone     Stone flag     Cement mortar stone     Mud mortar brick  
 Timber     Bamboo     Rc non engineered     Rc engineered  
 Other

## 3. Uso Secundario (Si aplica):

<input type="checkbox"/> Uso secundario general	<input type="checkbox"/> Agriculture	<input type="checkbox"/> Hotel	<input type="checkbox"/> Rental	<input type="checkbox"/> Institution
<input type="checkbox"/> School	<input type="checkbox"/> Industry	<input type="checkbox"/> Health post	<input type="checkbox"/> Gov office	<input type="checkbox"/> Use police
<input type="checkbox"/> Other				

[Predecir daño](#)

[Borrar formulario](#)

## Resultado de la predicción:

**GRADO DE DAÑO 1**  
**(Daño medio)**

Damage prediction on the application interface

## 6. Discussion

### 6.1 Interpretation of Results

Our results demonstrate that machine learning models can achieve considerable performance. The **0.7381 F1-score** is substantially superior to random guessing and novice human evaluators, and to the baseline benchmark Random Forest algorithm. However, models perform exceptionally well on majority classes (Light/Moderate damage) but show variability in minority classes (Extreme damage).

### 6.2 Comparison with Previous Literature

- Our Weighted Ensemble results are comparable or superior to methods reported in literature for this specific dataset.
- Stacking improved over Voting, suggesting heterogeneity of base models provides value.

### 6.3 Study Limitations

- **Geographic Limitation:** Data is solely from Nepal (2015).
- **Temporality:** Features are post-event.
- **Label Fidelity:** Dependence on human-assigned ground truth.
- **Class Imbalance:** Still affects minority class sensitivity.

### 6.4 Practical Implications

- **Rapid Post-Disaster Assessment:** Models can process 260,000 structures in minutes.
- **Resource Prioritization:** Focus on wall materials for retrofitting.
- **Policy:** Informing building codes for older structures.

**Ethical Considerations:** Automated predictions should assist, not replace, human judgment in critical safety decisions, especially for borderline cases.

## 7. Conclusions

### 7.1 Main Findings

- Weighted Ensemble reached an F1-score of **0.8401**.
- XGBoost was the best individual base model.
- Construction material is the single most important predictor.
- Automated solutions are practically viable for accelerating emergency response.

### 7.2 Contributions

This work provides a reproducible **methodology** for seismic damage prediction, an **empirical** comparison of boosting algorithms, and **interpretative** analysis of structural features.

### 7.3 Future Work

- Geographic Cross-Validation (train on one earthquake, test on another).
- Multimodal Data (integrate satellite imagery).
- Explainability (SHAP/LIME for individual instance explanation).

### 7.4 Final Reflections

Earthquakes are inevitable, but our response capacity is not fixed. By applying AI, we can improve post-disaster response speeds, saving lives and reducing suffering. This project proves that scalable, automated solutions are within reach.

## 8. References

- Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).
- Ke, G., Meng, Q., Finley, T., et al. (2017). LightGBM: A fast, distributed, gradient boosting framework. In Advances in Neural Information Processing Systems (pp. 3146-3154).
- Prokhorenkova, L., Gusev, G., Vorobev, A., et al. (2019). CatBoost: Unbiased boosting with categorical features. In Advances in Neural Information Processing Systems (pp. 6639-6649).
- Sairam, N., Gundlapalli, A., Rao, K. (2020). Deep learning approaches for earthquake damage assessment using satellite imagery. IEEE Transactions on Geoscience and Remote Sensing, 58(9), 6789-6801.
- Wolpert, D. H. (1992). Stacked generalization. Neural Networks, 5(2), 241-259.
- Applied Technology Council. (2007). ATC-20: Rapid visual screening of buildings for potential seismic hazards. Federal Emergency Management Agency.
- Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.
- Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd ed.). Springer.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
- Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems (pp. 4765-4774).
- United Nations Office for Disaster Risk Reduction. (2015). Sendai Framework for Disaster Risk Reduction 2015-2030. UNISDR.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321-357.
- Scikit-learn Developers. (2023). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
- TensorFlow & Keras Teams. (2023). TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.02820.
- United Nations Development Programme. (2019). Global Assessment Report on Disaster Risk Reduction. UNDP.
- Richter's Predictor: Modeling Earthquake Damage (<https://www.drivendata.org/competitions/57/nepal-earthquake/page/136/>).