Predicting Earthquake Damage

Applying Machine Learning to predict damage from the 2015 Gorkha earthquake in Nepal

by

Alessio Ciullo

A report submitted for the Certificate of Advanced Studies

in Advanced Machine Learning at the University of Bern, Switzerland

April 2021

Tentative table of contents:

1. Introduction
2. Using Machine Learning to predict damage from natural hazards: a literature review
3. Case study and data description
4. Model selection and performance metrics
5. Results
6. Discussion

Abstract:

Predicting losses from natural hazard is crucial for both public and private agencies. The former, like planning authorities, need to implement damage reduction measures while the latter, like insurance companies, need to reliably assess risk premiums. This project aims at predicting earthquake damage from the 2015 Gorkha Nepali earthquake using one of the largest post-disaster datasets ever collected in the context of natural hazard risk assessment. The data were collected by the Kathmandu Living Labs and the Central Bureau of Statistics surveying 762.106 buildings affected by the earthquake. For each building, 42 features were collected, and each building is categorized per damage grade from 1 to 3, i.e. from low to damage to complete destruction. The goal is to build a machine learning algorithm which can predict buildings damage grades. This would allow to better prepare for the next earthquake. Raw data are available [here](https://eq2015.npc.gov.np/#/download), and an [online competition](https://www.drivendata.org/competitions/57/nepal-earthquake/) was also started on the same topic.

1. Introduction: using Machine Learning to predict damage from natural hazards: a literature review + case study

Intro to the problem, previous examples on ML on damage assessment, earthquakes, and floods. Then introduction to the case study and the problem: just what data you have and what is the goal

# Data description and pre-processing

The dataset can be freely downloaded from the 2015 Nepal Earthquake Open Data Nepal Portal (<http://eq2015.npc.gov.np/>). The portal provides several datasets regarding individuals, buildings, and households information. The goal of this project is to estimate damage levels based on information about buildings that can be acquired before the earthquake. The available info about buildings relate to the buildings’ structure, damage assessment as well as ownership and use. The supplementary material provides an overview of all features contained in these three datasets and it highlights those selected for the analysis, i.e., those which can be acquired *ex-ante*. The selected dataset amounts to 762’106 instances and 38 features. The target variable, i.e., *damage­\_grade*, is a categorical variable with 5 categories, corresponding to five damage levels. Of the 38 features, 8 are categorical, 3 are textual (also a type of categorical variable), 22 are binary (0 or 1) and 5 are numeric.

## 2.1 Treating categorical features

The 8 categorical variables are treated with the standard one-hot-encoding method, where, i.e., each feature is transformed into as many one-hot-encoded features as the number of categories in that feature. There is thus one one-hot-encoded feature per category. For each instance then, all one-hot-encoded features are zeros except for the one relative to the category value in that instance, which is a one. One-hot-encoding has the advantage of transforming categorical features into a numeric format, which most machine-learning algorithms can more easily deal with, but it comes with the cost of enlarging the number of features.

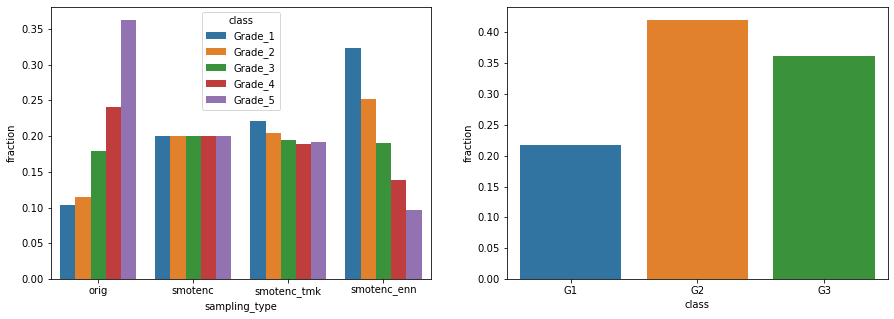
When dealing with high-cardinality categorical features, which have many categories, one-hot-encoding can be problematic as the newly created features may be too many. This is the case for the 3 textual features, i.e., *district\_id* (i.e., indicating in which of the 11 districts the building is located), *vdcmun\_id* (i.e., indicating in which of the 110 municipalities the building is located), *ward\_id* (i.e., indicating in which of the 945 ward numbers the building is located). For these features, impact (or target) encoding was used. Impact encoding was first introduce by Micci-Barreca (2001) and consists of assigning a numerical value to each category according to the target variable’s expected value. Assuming a binary classification, a categorical vector *x* and target variable *y*, the value of the *ith* category *xi* will be given by the probability *Pi* = P(*y* = 1| *x* = *xi*). Following Micci-Barreca (2001), this probability can be derived as:

where is the total number of instances, *ny* is the number of instances such that *y* = 1, *ni* is the number of instances where *x* = *xi*, *nyi* is the number of instances such that *y* = 1 and *x* = *xi*, and is a weighting factor that depends on *ni*. In case of multi-class classification problems, like in the present work, the procedure is applied to the one-hot-encoded target variables. By so doing, at the end of the procedure the 3 textual features were transformed into 15 numeric features (the 3 features times the 5 target classes). After transforming the categorical features into numeric ones, and dropping the original features, the number of features increases from 39 to 80.

## 2.2 Resampling training data

The data were split into train and test data, corresponding to 80 % (609’675) and 20 % (152’431) of the full dataset, respectively. As it can be inferred from the distribution of classes in the original training set (*orig*) in the left panel of Figure 1, it is clear that the dataset is quite imbalanced as class *Grade\_*5 alone accounts for 35 % of the instances and *Grade\_1* only accounts for 10 % of them. A typical way to deal with this problem is via resampling instances in order to better balance classes. Resampling can take the form of up-sampling, i.e., sample more of the minority classes, down-sampling, i.e., drop instances from the majority classes, or a combination of the two.

In this work, both up-sampling and a combination of up-sampling and down-sampling have been applied, and in the latter case two different down-sampling techniques are explored. In particular, up-sampling is performed via the *Synthetic Minority Over-sampling Technique for Nominal and Continuous* (SMOTENC) technique, which adapt *Synthetic Minority Over-sampling Technique* (SMOTE) introduced by Chawla et al. (2002) to deal with datasets that contains both numerical and categorical features (even if the categorical features where transformed into numeric, they are still discrete features). In brief, given an instance in the minority class, SMOTE creates new samples by considering the *k* nearest neighbours in the feature space to this instance, then taking the vector between one of those *k* neighbours and the instance itself and finally multiplying this vector by a random number between 0, and 1. The result represents a new, synthetic instance. As up-sampling can generate noisy samples during the interpolation process, i.e., points very close to each other which belong to different classes, cleaning down-sampling techniques can be applied. The applied techniques are Tomek links and Edited Nearest Neighbors (ENN). Tomek links were introduced in Tomek (1976) and are defined as pairs of nearest neighbor instances which belong to different classes. Once a Tomek link is identified, either both or only one instance (e.g., the one belonging to the majority class) is dropped. ENN was introduced by Wilson (1972) and it removes instances which do not agree (i.e., belong to the same class) with either the majority of all their nearest-neighbors. In this application, the majority rule has been adopted, which is obviously the more conservative, i.e., it drops less instances, than the two rules. The left panel of Figure 1 shows the prevalence of classes across the different datasets. Up-sampling generates and equal number of classes with a total number of training instances of 1’104’100, cleaning with Tomek links reduces the number of training instances to 897’616 but overall maintains an equal distribution of classes, cleaning with ENN reduces the number of training instances to 526’650 (lower than the original instnaces) and also provides an imbalanced set, this time with the majority class being *Grade\_1*.



Figure

## Reduction of the number of classes

Since not a

* Not all answers are equal wrong, prediction damage 1 when then damage 5 happens is the same as damage 4 and then 5 happens

1. Method

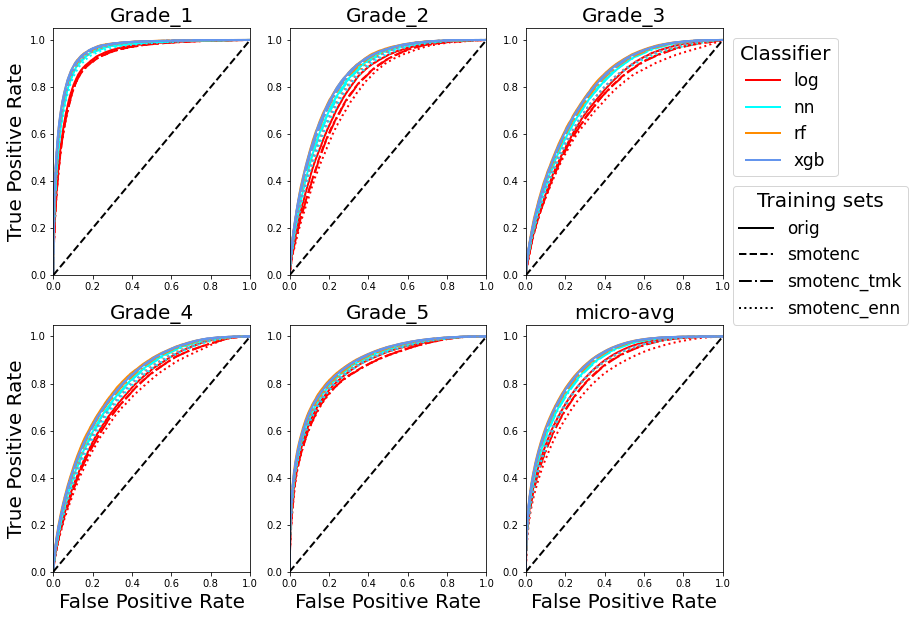
The algorithms used – with few words about their details, the grid search with all parameters (make single table, highlighting choice), the metrics selected for the evaluation

1. Results

F1 score

ROC, AUC

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **dummy** | **log** | **nn** | **rf** | **xgb** |
| **orig** | 0,199142 | 0,549492 | 0,571372 | 0,596724 | 0,592426 |
| **smotenc** | 0,199588 | 0,535314 | 0,575 | 0,586948 | 0,590576 |
| **smotenc\_tmk** | 0,198092 | 0,53351 | 0,568925 | 0,584724 | 0,58969 |
| **smotenc\_enn** | 0,200894 | 0,513742 | 0,544151 | 0,555029 | 0,55516 |
| **3Classes** | 0,335654 | 0,685708 | 0,70505 | 0,722449 | 0,720068 |



Conclusions

Few conclusive words

