

Supervised Learning

Ignacio Martínez Martín, Pablo Valle Nieto, Pedro Rodríguez Viñuales

December 2022

Contents

1	Introduction	2
2	Load the data	2
3	Baseline	2
3.1	Feature selection	2
3.2	KNN model	2
3.3	Decision trees	3
4	Hyperparameter optimization	3
5	Competition results	4

1 Introduction

This report will describe how we have been applying different machine learning models (knn, decision trees, hyperparameter) with the aim of predicting the level of damage to buildings caused by the 2015 Gorkha earthquake in Nepal. We're trying to predict the ordinal variable *damage grade*, which represents a level of damage to the building that was hit by the earthquake.

For this, we have participated in a competition, which offers us the datasets with which we will work and will qualify our prediction of the damages.

2 Load the data

In this project we will use four datasets:

1. **Train labels:** It contains a *building id* along with its *damage grade*, which is the feature we're going to predict.
2. **Train values:** This contains the information of the buildings such as their height, age, ...

These two datasets serve as training for the model, the final tests will be done on the following datasets.

3. **Submission format:** We will use it to format our submission file.
4. **Test values:** It is the datasets with which we will test the model

The datasets train and test values mainly consist of information on the buildings' structure and their legal ownership. Each row in the dataset represents a specific building in the region that was hit by Gorkha earthquake. There are 39 columns in this dataset, where the *building id* column is a unique and random identifier.

3 Baseline

3.1 Feature selection

To optimize the work environment, we have made a selection of characteristics that consists of choosing those that have an integer type. Later, we realized that the features named "has" were boolean and in the dataset they appeared as integers (0 or 1) so we removed them from our choice. The final list would be this:

```
['geo_level_1_id',  
'geo_level_2_id',  
'geo_level_3_id',  
'count_floors_pre_eq',  
'age',  
'area_percentage',  
'height_percentage',  
'count_families',  
'damage_grade']
```

Figure 1: Features list

Before running the models, we need to split the data set into train/test sub sets.

3.2 KNN model

First of all, we use the metric f1 score because it is the metric of the competition. Additionally, we will use **KNeighborsClassifier** because it based on the k nearest neighbors of each query point.

Once we have identified the best parameterization we will go on to make an execution of the model and we will graph its results in a confusion matrix.

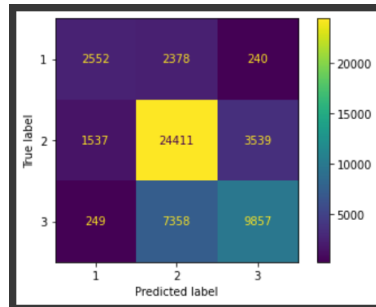


Figure 2: Confusion matrix knn

We can see that the prediction is not entirely accurate and this is confirmed by the competition metric, which rates this model (with test values) with **0.7076**.

3.3 Decision trees

This model has the same process as the previous one. We use **DecisionTreeClassifier** to construct the model, which when executed, we can see the following information.

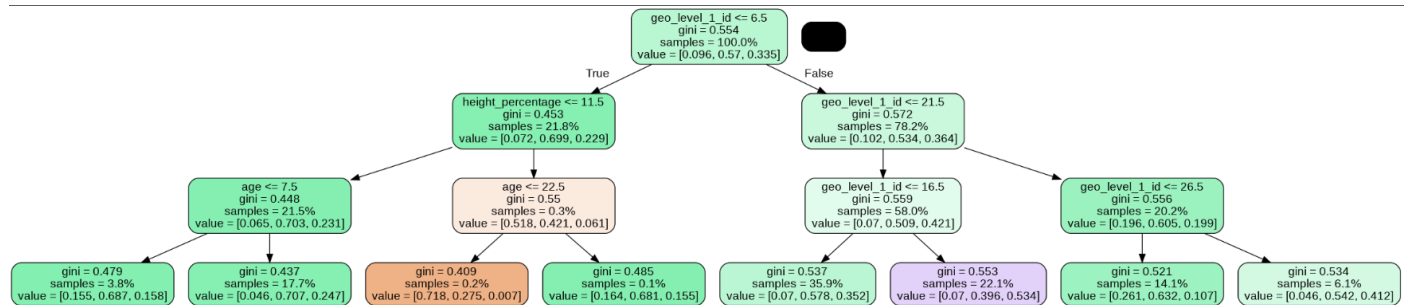


Figure 3: Decision tree

We can see that the relevant features are height, age, and geographic region 1. The competition rates this model (with test values) with **0.6482**.

4 Hyperparameter optimization

To optimize the model, we are going to use hyperparameters to find the best combination of algorithm parameters. These parameters are called hyperparameters, while the coefficients found by the learning algorithm itself are called parameters.

To carry out the optimization, we use **GridSearch**. It is a parameter fitting approach that allows you to methodically construct and evaluate a model for each combination of specified algorithm parameters in a grid.

After this, we execute the model plot the result in a confusion matrix.

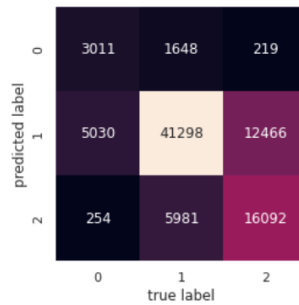


Figure 4: Confusion matrix hyperparameter

As we can see, something similar to the knn model happens. The prediction is good but it is not perfect, that is why the competition rates the model with **0.7003**.

5 Competition results

In this section we will see the results of the models in the competition in this order from top to bottom: Decision tree, knn and hyperparameter optimization.


0.6482	vallepablo14 	2022-12-13 19:00:39 UTC
0.7076	vallepablo14 	2022-12-13 19:01:10 UTC
0.7003	vallepablo14 	2022-12-13 19:09:01 UTC

Figure 5: Competition results