Distributed Data Analysis and Mining Project

Segurini Gloria [567352] g.segurini@studenti.unipi.it
Macchia Alessandro [636679] a.macchia3@studenti.unipi.it

University of Pisa
A.Y. 2022/2023

Contents

Abstract

The present report shows the work performed on the "earthquakes" dataset. All the code was written in python language by means of the Apache pyspark distributed environment.

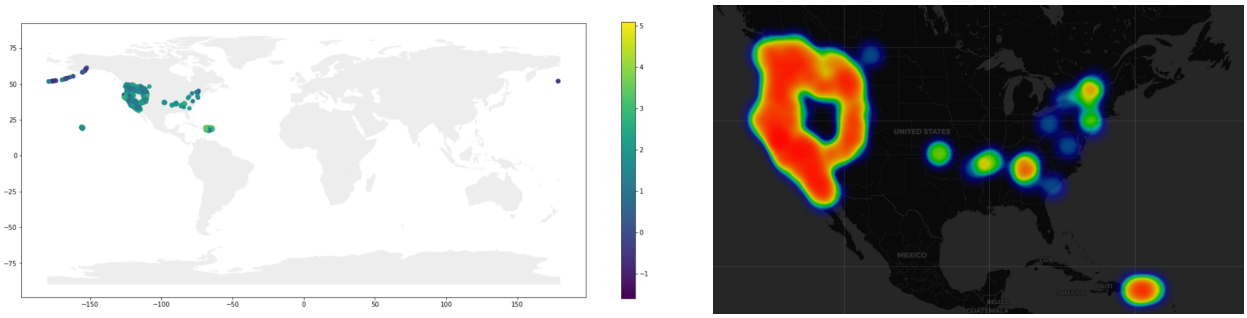
1 Introduction

The aim of our project is to both perform a regression and a classification tasks by means of Machine Learning techniques on an earthquakes dataset so as to put into practice what we have seen during the course of Distributed Data analysis and Mining. Before going into technical details it is important to point out a few things which will benefit the reading of this report. In fact, we must clarify that the dataset we used refers to registered seismic events, which have occurred around the world from 1970 to march 2019. The dataset is available on the "Kaggle" platform at the following link: *https://www.kaggle.com/datasets/danielpe/earthquakes*.

2 Data Understanding, Cleaning and Preparation

2.1 Overview and Distribution of the variables

In order to give a brief introduction to our dataset, we provide also the two plots represented in Figure 1. Those two images have been extracted by plotting a really small sample of our original data (0.015%). They give us an idea of where the seismic events are mostly located around the world, their scale of magnitude, and a focus on one of the most affected regions.



(a) world plot (b) heatmap

Figure 1: locations of some seismic events

The original dataset consists of 3.272.774 records and 23 features where each record refers to a different seismic event. As we can see from the Table 1 there are 13 numerical features, 2 timestamp features and 8 categorical features. As it has been stated above, the records are related to seismic occurrences while the columns refer to specific measurements and information regarding those events. More details about these features will be provided below.

Table 1: Variables description

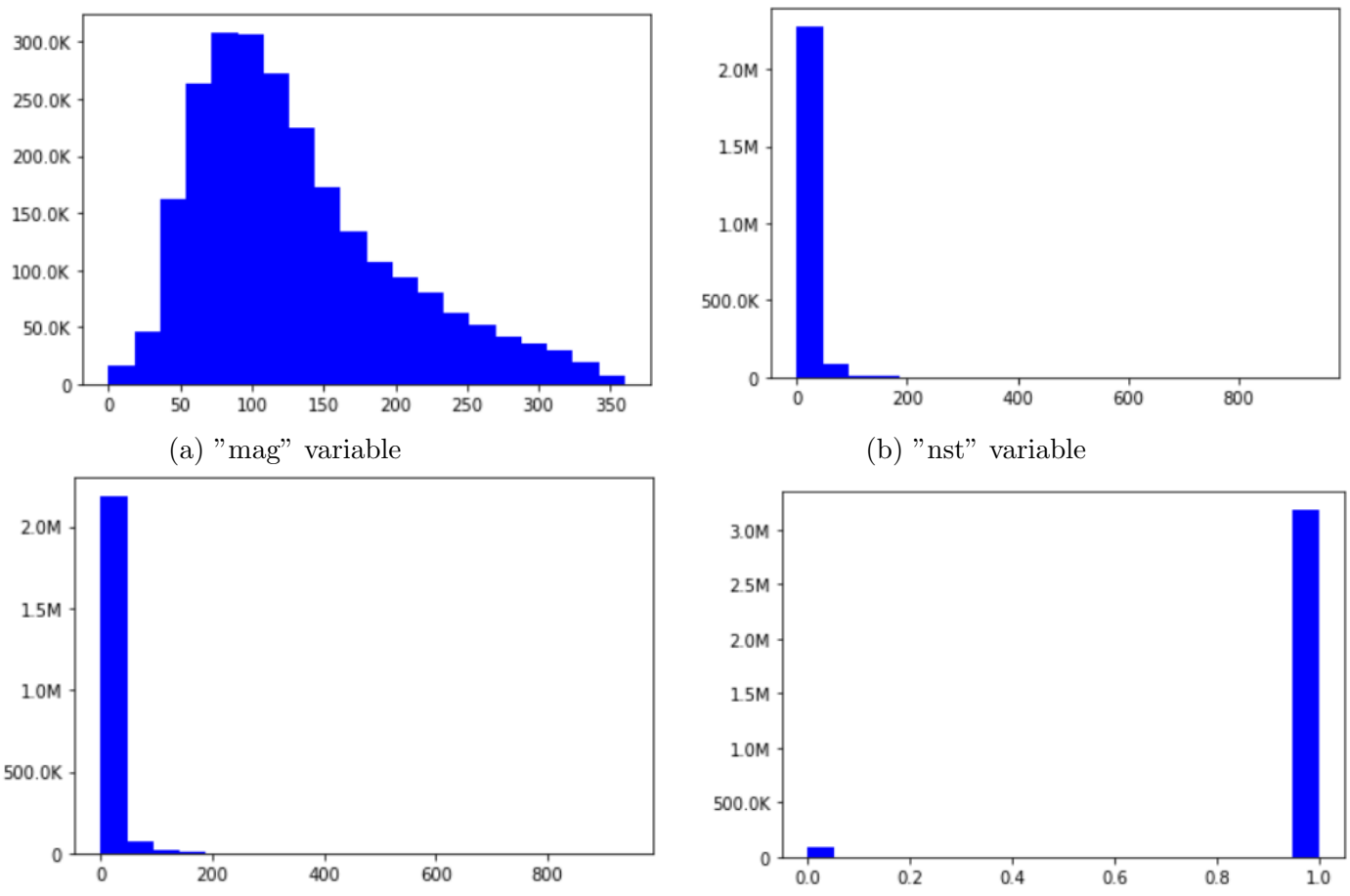
Name	Type	Domain	Description
_c0	integer	i.e. 988481	It's a numeric value that identifies an event.

time	timestamp	i.e. 1970-01-01 00:00	Time when the event occurred.
latitude	double	i.e. 87.265	Decimal degrees latitude. Negative values for southern latitudes.
longitude	double	i.e. 180.0	Decimal degrees longitude. Negative values for western longitudes.
depth	double	range: $(\{-10.0, 735.8\})$	Depth of the event in kilometers.
mag	double	range: $(\{-9.99, 9.1\})$	The magnitude for the event.
magType	string	i.e. $(\{md\})$	Result of the hazard assessment using the seismic method.
nst	double	range: $(\{0.0, 934.0\})$	The total number of seismic stations used to determine earthquake location.
gap	double	range: $(\{0.0, 360.0\})$	The largest azimuthal gap between azimuthally adjacent stations (in degrees).
dmin	double	range: $(\{0.0, 141.16\})$	Horizontal distance from the epicenter to the nearest station (in degrees).
rms	double	range: $(\{-1.0, 104.33\})$	This parameter provides a measure of the fit of the observed arrival times to the predicted arrival times for this location.
net	string	i.e. $(\{ak, at, ci\})$	Identifies the network considered to be the preferred source of information for this event.
id	string	i.e. $(\{uw61515282\})$	A unique identifier for the event that may change over time.
updated	timestamp	i.e. $(\{2016-04-02$ 20:22:05.312$\})$	Time when the event was most recently updated.
place	string	i.e. $(\{Yunnan, Cina\})$	Textual description of named geographic region near to the event.
type	string	i.e. $(\{earthquakes\})$	Type of seismic event.
horizontalError	double	range: $(\{0.0, 280.6\})$	Uncertainty of reported location of the event in kilometers.
depthError	double	range: $(\{-1.0, 1773552\})$	Uncertainty of reported depth of the event in kilometers.
magError	double	range: $(\{0.0, 6.11\})$	Uncertainty of reported magnitude of the event.

magNst	double	range: $(\{0.0, 941.0\})$	The total number of seismic stations used to calculate the magnitude for this earthquake.
status	string	i.e. $(\{reviewed,$ automatic $\})$	Automatic events are directly posted by automatic processing systems and have not been verified or altered by a human. Reviewed events have been looked at by a human.
LocationSource	string	i.e. $(\{ak, at, ci\})$	The network that originally authored the reported location of this event.
magSource	string	i.e. $(\{ak, at, ci\})$	Network that originally authored the reported magnitude for this event

Additional information about the variables can be found at the following link:
https://earthquake.usgs.gov/earthquakes/feed/v1.0/csv.php

In the following figures we provide some plots regarding some of the most interesting variables, which are the ones we used during out regression and classification tasks. In the figure 2d we plotted the distribution of our classification target variable, namely "type". In this case we can see how, basically, all the seismic events are targeted as 1 ("earthquakes").



(a) "mag" variable

(b) "nst" variable

(c) "magNst" variable

(d) "type" variable

Figure 2: histograms

2.2 Preliminary setup

The first thing we noticed is that the real id, intended as a unique identifier of the represented event, was not the "id" column, but rather the "_c0" one. In fact while the first one is a string with duplicates within the dataset, the second is filled out with unique values even for those records that presented copies. This lead us to comprehend that the "id" column was useless. Moreover, we realized that also other features were not important for our work, thus we agreed to delete also "magSource", "LocationSource", "updated", "status", "place" and "net". We also dropped the duplicate rows and this lead us to a dataset with 3.256.955 records.

2.3 Missing Values

Proceeding with the analysis, one of the biggest challenges that we faced during the data understanding part of our project, has been dealing with a massive presence of missing values. In Table 2, it is reported the percentage of "NaN" values for each feature, with respect to the total count of records. Since the amount of missing values on "mag", "magType", "depth" and "rms" was not very high, we decided to keep these columns and drop the rows with Nans. The columns presenting too many missing values, such as "dmin", "horizontalError", "magError" were directly dropped. We end up with a dataset of 2.899.664 records with the following amount of missing values: 25.67% on "nst", 23.01% on "gap", 11.27% on "depthError" and 22.01% on "magNst".

2.4 Errors spotting

Subsequently, we proceed looking for obvious errors in the dataset. In fact, we noticed some inconsistencies. The feature "depth" shows negative values up to -10, while the highest mountain in the world has a depth of -8 (Everest). We then decided to drop values lower than -5. The feature "mag" shows the magnitudo of a seismic wave. According to some research, some values can be negative, but not lower than -1. For what concerns "nst" and "magNst", which respectively show the number of stations used for detecting the place of the seismic event and its magnitude, it was really surprising to see that some records had a 0 on such features. However, since this amount of records was really high, we decided to keep them. In the same way, there are many values of gap greater than 180, even if the latter should be the maximum value. We kept those values. Concerning "rms", since this is a temporal feature we dropped the negative values. We did the same for "depthError" and we furtherly dropped the records presenting an error higher than 500.

The final dataset presents 2.570.408 records with the following features: _c0, time, latitude, longitude, depth, mag, magType, nst, gap, rms, net, type, depthError, magNst.

Table 2: Percentage of missing values

Name	**Percentage of missing values**
gap	25.61%
dmin	41.13%
rms	6.46%
magNst	30.21%
nst	26.93%
horizontalError	46.80%
depthError	18.53%
magError	54.40%
mag	4.78%
magType	5.11%
place	$< 0.1\%$
depth	$< 0.01\%$

2.5 Categorical Features Cleaning

We finally proceeded with the categorical features' cleaning. Specifically, we cleaned the variables "magType" and "type". Concerning "magType", we noticed that some values were uppercase, so we transformed everything into lowercase. We also reduced the number of possible distinct values by substituting some of them which rarely appeared (this having a count lower or equal to 10.000) with the string: "other", leading to the following situation:

Table 3: magType values

magType	**count**
mc	194703
other	24891
ml	995656

mh	102847
mb	176867
mdl	20166
md	1055278

The same consideration was made also for "type", which showed a huge amount of "earthquakes" types, while the other values had a very low frequency. Considering the future Classification Task, we decided to group together all the values different from "earthquake" and put them into the value "other", leading to the following situation:

Table 4: type values

type	**count**
earthquake	2509558
other	60850

3 Feature Selection and Regression Task

The regression Task consists of the prediction of the missing values on the features "nst", "type" and "magType".

We first transformed the variable "time" (which is a timestamp) into "time_unix", which is an integer and shows the number of seconds starting from 1970, January 1st. We proceeded by checking the amount of missing data, which is the following:

Table 5: Percentage of missing values

nst	**gap**	**magNst**
18.87%	22.83	20.92%

These Nans were temporarly dropped in order to show the correlation matrix between numerical features, which is as follows in figure 3

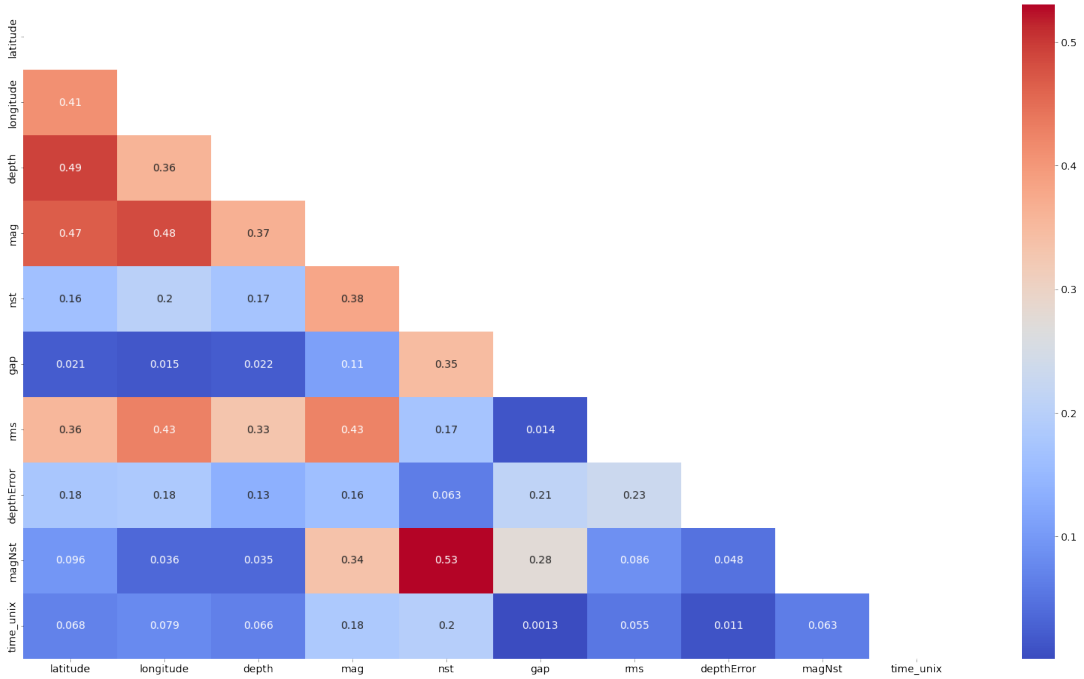


Figure 3: Correlation matrix between numerical attributes

We did not have any particularly highly correlated feature. In fact, the highest Pearson correlation coefficient is between "nst" add "magnst" with a value of (0.53) due to the intrinsic meaning of the features themselves. This kind of analysis has been fundamental in order to decide which features were worth to keep for the regression task, so as to check a "one-to-one" correlation between the variables. The features to keep were different each time, depending on the target variable.

3.1 One Hot Encoding and Hold-Out

We first used the OneHotEncoding technique for the cateorical variables "type" and "magType". The first one was converted to "type_earthquake" which takes value 1 when it comes to an earthquake and 0 otherwise and the following columns were added: "magType_md", "magType_ml", "magType_mc", "magType_mb", "magType_mh", "magType_other" while the columns "type" and "magType" were deleted.

Later on we proceeded by dividing the whole dataset into training set (TR), test set (TS) and blind test set (blind_TS) in which they are conteined the missing values of the target variable. In order to always use the same TS on the regression task and on the three distinct regression tasks, the first division was between TR and TS and the second one between TR and blind_TS. This was done, because since the regression task aims to the prediction of the missing values on three different target columns, the "blind test" changes each time and so does the TR. In addition, in order to perform a model assessment on a TS, it obviously cannot contain any missing value.

3.2 Feature selection and Outliers detection

For the feature selection phase we used a function which takes in input a dataframe and the columns that we want to keep basing on the values of the Correlation Matrix. The function also takes in input the TS, so that we can remove such columns also from it. This function also takes the columns with Nans that we want to temporarly fill with the mean, so as to perform the regression task.

As stated above, we first do a hold hout by using the "TR_TS_division" function, which takes the original dataframe and filters the rows presenting Nans, so that they will not be part of the TS. This division is performed just once at the beginning of the task, before running the code on the various columns. It is common practice to perform a 80 - 20 hold out between TR and TS, but the amount of missing values changes every time and so does the TR. The original dataset without missing values consists of 1.895.280 records and so the TS will consist of 514.081 records which represent the 20%. However, we must also consider the division between TR and blind_TS. In fact, this percentage should be calculated on the dataset without missing values (that go to the blind test) and this leads to a 27% new percentage for the TS. We make a randomSplit on a variable where we stored the amount of data without missing values with this new percentage, take the 27% as TS and make a union with the remaining 73% and the amount of missing data, which will lead us to a new TR. For every of the three iterations of the regression task, this TR will be splitted into TR and blind_TS, where the latter takes the missing values of the target variable.

We further check for outliers. We used the boxplot technique and just drop the outlier records.

We then standardize our data after performing a join with the encoded variables. It is important to point out that the model that we used to perform the standardization on the TS is the same used in the TR.

Finally, we proceeded for checking another phenomena that could have influenced our regression task, namely the multicollinearity. We did this by means of the Variance Inflation Factor and dropped all variables with VIF greater than 5.

3.3 Linear Regression

Finally, we had reached all the steps needed to compute the regression. We performed a linear regression and also exploited the power of lasso, which performs both variable selection and regularization to enhance the prediction accuracy and interpretability of the resulting statistical model. We performed a gridsearch with the following hyper-parameters values and while using a Crossvalidation with 4 folds.

Table 6: parameters

RegParam	**maxIter**	**elasticNetParam**
[1e-5,1e-3]	[0.1, 1e-7]	[0, 1, 0.5]

3.4 Results

At this point, we wanted to check how out model performed and compare the results with the ones obtained by using the mean as technique to fill the Nans. In order to run a model which only computer the mean, it was sufficient to use the DecisionTreeRegressor with maxDepth = 0. This returns the average of the target column and also lets us compute the CrossValidation with folds = 4. This is crucial for the model selection phase.

We run the code three times, every time for a different target. At the end of each prediction, we joined the predicted values with the original TR, so that the dataset for the next iteration of the regression task has the values predicted instead of Nans.

The results are reported in the following table 7, while the last one (8) refers to the best hyperparameters configuration. We can see that after performing the VIF and checking the

correlation matrix, LASSO does not result as the best model (elasticNetParam = 1), while Ridge does instead (elasticNetParam = 0).

	4 - Fold Cross Validation		Test Set
	Mean	**Regression**	**Best**
gap	61.90	52.72	87.86
magNst	8.56	6.12	17.03
nst	10.08	6.45	14.19

Table 7: CVResults

	Best Params		
	elasticNetParam	**regParam**	**maxIter**
gap	0.0	1e-05	100
magNst	0.0	1e-05	100
nst	0.0	1e-05	100

Table 8: Best Params

Eventually, we end up with the final dataset with no missing values. Finally, we join the final dataset with the original categorical column and perform the same operation on the TS and save both the final TR and TS.

3.5 Comments

It is noticeable how the values predicted with the regression are better than the ones with the average. In general, the error on "gap" is higher than the ones on "nst" and "magNst". This is probably due to the fact that the instrinsecal meaning of "gap" is not really related to the rest of the dataset.

4 Classification

For what concerns the classification task, our aim was to develop a model which was able to correctly distinguish between two different types of events, namely "earthquakes" and "other". We remind the fact in "other" we have grouped all the events that were present in the "type" variable, but with a value that was different from "earthquakes". Despite that, our dataset was still very imbalanced, presenting 97.7% of records labeled as "earthquakes". Considering this situation, we opted for creating a new feature, named "earthquakes" which had value "1" when the seismic event was an earthquake indeed, while taking "0" in any other cases. For what it concerned the other features, they were still the same used before. Specifically, we had 11 columns in total (plus "_c0") where just the "magType" column was filled with categorical values. Furthermore we also remind that the split between training set and test set has been done using the 80-20 division, leaving us with a training set with 2056385 records and a test set with 514023 records.

4.1 Outliers

Also in this case we decide to perform an outlier detection on the training set, in order to gain the best model as possible. This, has been achieved through the exploitation of the boxPlot

method. To avoid deleting to many records, the original algorithm has been modified. In fact we agreed on multiply the interquartile range by 3.5 instead of the previous 1.5 (as usually it is done). This changing of performing was deemed correct, since the fact that this algorithm works on the features individually and it is not able to distinguish a record as an outlier as a whole (considering all the feature simultaneously). However, it allows us the elimination of the records which presented values completely out of range with respect to rest (for every single feature). Taking into account these consideration we deemed clever the deployment of this modified algorithm. After performing what have just cited, the number of outliers became 1501833.

4.2 Features Selection

During the next step we computed a feature selection to verify if there were attributes which were completely independent from the target label. In fact, in that case we would have dropped them, avoiding to consider them during the classification task. In order to do that, we performed the Chi Square Test among each feature and "earthquake". All the numeric attributes have been discretized so to allow us to compute the test just cited. The latter, has been realized through the usage of the quantile approach, allowing us to obtain a uniform distribution for each bin. In particular we generated 5 bins for each feature. The result of the Chi Square Test are reported in the table 9:

Features	**pValue**	**statistic**
nst	0.0	883.214168
latitude	0.0	8620.20677
magNst	0.0	4120.30167
longitude	0.0	8552.25538
time_unix	0.0	7757.04160
gap	0.0	1458.14740
rms	0.0	6495.30171
magType	0.0	3047.30333
depth	0.0	82875.9680
mag	0.0	13130.4542
depthError	0.0	8933.29214

Table 9: Chi Square Test result

The test's hypothesis is that the feature is considered independent from the label. Having the p-value zero for each test computed, we could conclude that we has to reject all the hypothesis of indipendence. For these reasons we decided to keep all the features for performing the classification task.

4.3 Random Forest

As a classification model we chose the random forest model, since being an ensemble method, we deemed that it could lead to a better performance with respect to a single classfier. In order to do that, we firstly performed a One Hot Encoder on the categorical feature "magType", so it became useful for the classification. To avoid overfitting and to choose the best hyperparameter tuning, it has been performed a grid_search with a 4-fold cross validation. The tested values and parameters are reported in the table below.

Param	Values	Best
maxDepth	[4,8,11,14,18,22]	18
minInstancesPerNode	[75,150,300]	150
featureSubsetStrategy	[5,7,11,15]	11
impurity	gini	gini
numTrees	20	20
bootstrap	True	True

Table 10: Param Grid

Having an imbalanced situation, we opted to add a parameter called "weightCo". In order to do that a new feature named "weight" has been created. The latter represented the weight of the record with respect of the value on "earthquake". The following formula has been applied: weigth(i) = n / c * ni. With:

- n = number of records.
- c = number of different classes.
- ni = number of records of the ith class considered.

Doing that, allowed the unbalanced class to acquire more importance, leading the model the ability to distinguish in a better way the two classes. We remind that as a metric we used the AUC. The average value of the AUC computed on the best hyperparameters through the 4-fold cross validation was 96.92%. Furthermore, once we gathered the best hyperparameters we also performed once again everything, but utilizing the whole training set with the purpose of obtaining the final model.

4.4 Performance

Once we gained the final model with the best hyperparameters, we perfrmed the prediction using the test set. The confusion matrix is shown in the table 11.

	True Earthquake	True Other
Predicted Earthquake	486727	1284
Predicted Other	13112	12900

Table 11: Confusion Matrix Test Set

Subsequently, we also computed some measures in order to check the quality of the model on doing the predictions. In table 12 can be seen the values for AUC, Precision, Recall and F1. Moreover, below, can also be seen the figure regarding the Roc Curve.

Measure	Value
AUC	0.9892
Weighted Precision	0.9835
Weighted Recall	0.9719
Weighted F1	0.9759

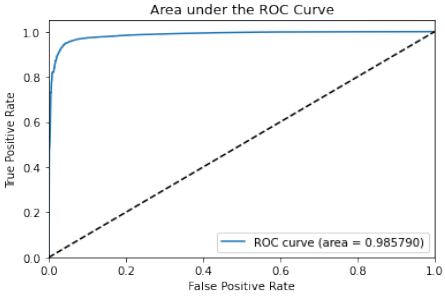


Table 12: Assessment on Test Set and Roc Curve

5 Conclusion

In conclusion we can state that despite a very imbalanced initial situation, we were able to correctly create well performing classification and regression models. We strongly believe that this is just the starting point for a series of further analysis, that could lead to the discovery of new hidden patterns behind this data.