

# Usage of Multiple RTL Features for Earthquakes Prediction

P. Proskura[1(✉)], A. Zaytsev[2], I. Braslavsky[1], E. Egorov[2], and E. Burnaev[2]

[1] Institute of Information Transmission Problems Russian Academy of Science, Moscow, Russia
polina.231.11@gmail.com
[2] Skolkovo Institute of Science and Technology, Moscow, Russia
{a.zaytsev,e.egorov,e.burnaev}@skoltech.ru

**Abstract.** We construct a classification model, that predicts if an earthquake with the magnitude above a threshold will take place at a given location in a time range 30–180 days from now. A common approach is to use expert-generated features like Region-Time-Length (RTL) features as an input to the model. The proposed approach aggregates of multiple generated RTL features to take into account effects at various scales and to improve the quality of a machine learning model. For our data on Japan earthquakes 1992–2005 and predictions at locations given in this database, the best model provides precision as high as 0.95 and recall as high as 0.98.

**Keywords:** Machine learning · RTL features · Earthquakes prediction

## 1 Introduction

Physical modeling now fails to provide an accurate earthquake predictions because of the complex nonlinear behavior of seismicity. Instead researches adopt data-based approaches and construct machine learning or statistics-inspired models based on past data on earthquakes [3].

There are a number of problem statements related to earthquake predictions: consider a target region and predict place and time of the next earthquake, split the region into a grid and predict earthquake at the each of them, consider each earthquake as a separate event with given location and time and predict the value of this magnitude [3]. In this paper, we consider the third problem statement and construct a model that predicts if the magnitude exceeds a given threshold for an earthquake at a given location and time.

A starting point of a model for earthquake prediction is an empirical relationship or a physical modeling that provide a representation of reality. Often this representation is not good enough, and one adopts a machine learning approach on top of the physics-driven description.

Supported by Skoltech.

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11619, pp. 556–565, 2019.
https://doi.org/10.1007/978-3-030-24289-3_41

For the problem of the prediction of earthquakes scientists consider several empirical statistical relationships e.g. Gutenberg–Richter law and Omori–Utsu (O–U) law. Gutenberg–Richter law [11] expresses the relationship between the magnitude and total number of earthquakes in the following way:

$$\log N = a - bM, \tag{1}$$

where $N$ is the number of events with the magnitude greater than $M$, $a$ and $b$ (commonly referred to as the **b-value**) are coefficients fitted using given data. Omori–Utsu (O–U) law [18] represents the decay of aftershock activity with time

$$\dot{N}(t) = \frac{C_1}{(C_2 + t)^p}, \tag{2}$$

where $t$ is time, $N$ is a number of earthquakes, $C_1$, $C_2$ and decay exponent $p$ (commonly referred to as the **p-value**) are coefficients fitted using given data. Both these models provide high-level description on number of earthquakes in the target region, moreover, Omori–Utsu law provides a connection between past and future seismologic activity in a region. Another physics-inspired features are RTL features [17] that provide an aggregation of past seismic activity into a single feature by weighting of past earthquakes that occur near the point, where we want to predict an earthquake. RTL features also have a number of hyperparameters to be fitted using collected data.

On top of these features we can construct a machine learning model [2,4,15,16]. An example of such a work [4] considered the prediction of earthquakes as a binary classification problem. Authors generated 51 meaningful seismic features calculated for a dataset at hand based on well-known seismic heuristics such as "Standard Deviation of b-value" or "Time ($T$) of n events". As models they used various ensemble methods such as Random Forest, Rotation Forest and RotBoost.

In this paper we consider a different problem statement: we want to predict earthquakes at distant time interval. As input features we use normalized RTL features with different parameters and at different time scales. We examine a number of machine learning techniques to make use of generated features.

During evaluation of the models and model construction we take into account peculiarities of the problem: imbalance of classes (there are only a few large earthquakes in the dataset) [5,6]; data are time-series based; there are no external features for these data, and we need to generate features from the data on past earthquakes.

## 2 Problem Statement

In the given dataset each earthquake has 4 parameters: location $(x, y)$, time $t$ and magnitude $M$. We define that the earthquake takes place $c(x, y, t) = 1$ at some location $(x, y, t)$ if there is at least one earthquake of magnitude $M_e \geq M_c$ with coordinates $(x_e, y_e, t_e)$ satisfying the following constraints:

$$||(x, y) - (x_e, y_e)||_2 \leq R_c, \quad \delta_c < t_e - t < T_c. \tag{3}$$

Otherwise we define that $c(x, y, t) = 0$

Our goal is to construct a model that predicts if there is an earthquake in a time cylinder $[T + T_{min}, T + T_{max}]$ using information about all earthquakes up to time $T$.

We aggregate information about earthquakes up to time $T$ in a vector of features of fixed length $\mathbf{x}$. In particular we generate RTL features using procedure described in Subsect. 4.1. Figure 1 demonstrates used space-time cylinder used for generation of input features and the target interval for prediction.

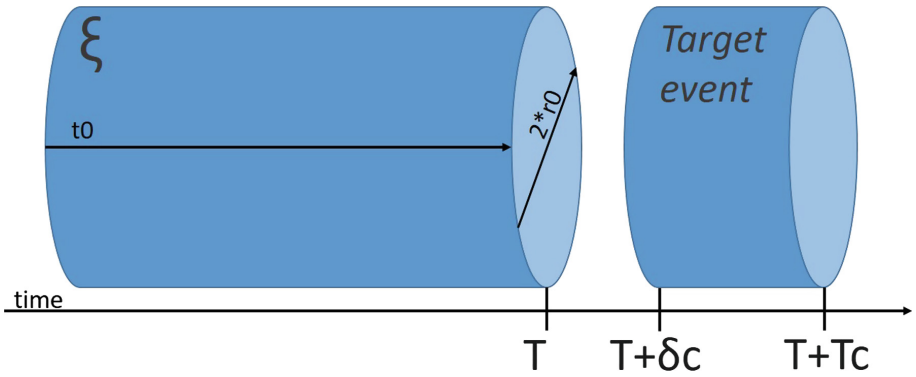


**Fig. 1.** Space-time cylinder used for generation of RTL features and interval considered for prediction

After these procedure for all earthquakes in a database we have a pair $(\mathbf{x}_i, c_i)$ with $c_i$ defined by (3) with the magnitude threshold $M_c = 5$. All these pairs form a sample $D = \{(\mathbf{x}_i, c_i)\}_{i=1}^{n}$ of size $n$. Our goal is to create a model $\hat{c}(x, y, t)$ that resembles true relationship $c(x, y, t)$.

## 3 Data

We study the prediction of strong earthquakes in the middle-term horizon. Strong earthquake is an earthquake with the magnitude higher than $M_c = 5$. Predictions of earthquakes are related with the difficulties:

- The sample is unbalanced. In Japan, from 1990 to 2016, there were 247, 204 earthquakes. Consider a distribution by the magnitudes which is shown in Fig. 2. The most part of classifiers and their metrics are common with balanced samples, where the amount of target class is approximately the half of the total. Thus in our case, the tuning of classifier and choice of metrics are needed to make them more sensitive to the target class.
- The dataset is nonhomogeneous as the network of seismic stations changes with time. So, the process of features generation is harder, because every couple measurements are needed to be evaluate for approximate equality.

– There is a lag in time between available and desired prediction interval, because exact point is unavailable according to our measurements.

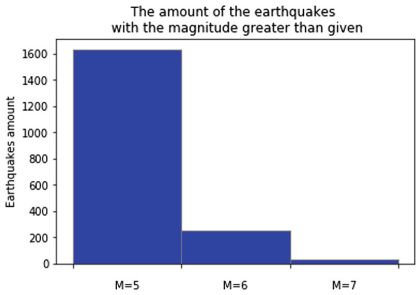


**Fig. 2.** Histogram of magnitudes: earthquakes with greater magnitude consist a small part of the sample, as the total sample size is about 300000 and there are only about 2000 with a magnitude greater or equal to 5 among them.

## 4 Methods

### 4.1 RTL Features

As inputs for Machine Learning model we use RTL features. The basic assumption of the **Region-Time-Length (RTL)** algorithm [17] is that the influence weight of each prior event on the main event under investigation may be quantified in the form of a weight. Weights become larger when an earthquake is larger in magnitude or is closer to the investigated place or time. Thus, **RTL** characterizes the level of seismicity at the point of space in the certain time.

The **RTL** takes into account weighted quantities associated with three parameters (time, place and magnitude) of earthquakes. A **RTL** parameter is defined as the product of the following three functions:

$$\mathsf{RTL}(x, y, t, M) = \mathsf{R}(x, y, t, M) \cdot \mathsf{T}(x, y, t, M) \cdot \mathsf{L}(x, y, t, M),$$

where $\mathsf{R}(x, y, t, M)$ is an epicentral distance, $\mathsf{T}(x, y, t, M)$ is a time distance and $\mathsf{L}(x, y, t, M)$ is a rupture length. They depend on the size of the space-time cylinder $\mathcal{E}_{r_0,t_0}$, defined by radius $r_0$ and time length $t_0$.

$$\mathsf{R}(x, y, t, M) = \sum_{e_i \in \mathcal{E}} \exp\left(-\frac{r_i}{r_0}\right),$$

$$\mathsf{T}(x, y, t, M) = \sum_{e_i \in \mathcal{E}} \exp\left(-\frac{t - t_i}{t_0}\right),$$

$$\mathsf{L}(x, y, t, M) = \sum_{e_i \in \mathcal{E}} \left(\frac{l_i}{r_i}\right).$$

where $e_i$ is full description of an earthquake $(x_i, y_i, t_i, M_i)$, $l_i$ is an empirical relationship specific for Japan $\log l_i = 0.5M_i - 1.8$, $r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}$. We use only earthquakes with magnitude at least $M_i \geq M_0 = 5$.

### 4.2 Normalization of RTL Features

**RTL** is a very unstable statistics. Therefore in the article [13] authors proposed to normalize the parameters on the variances.

Normalization occurs during data preprocessing. For our case we transform the data to make each feature zero-mean and unit-variance [1]:

- Calculate mean and standard variance for each feature.
- Subtract the mean from each feature.
- Divide the values of each feature calculated previous by its standard deviation.

Also subtracting the moving average instead of mean. It helps to take into account time trend.

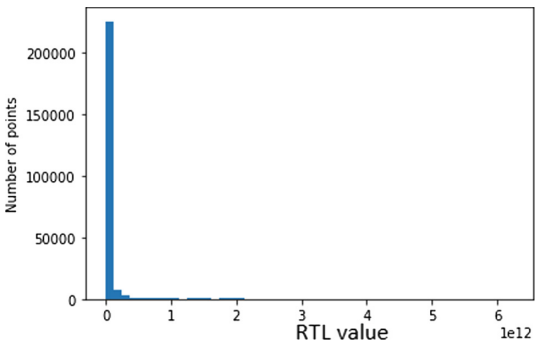


**Fig. 3.** Histogram of RTL values for each point. Small values prevail.

The negative **RTL** means a lower seismicity compared to the background rate around the investigated place, and the positive **RTL** represents a higher seismicity compared to the background Fig. 3. We are interested in both types of anomalies.

### 4.3 Classifiers

As classifiers we used the following machine learning methods:

- **Major RTL** is the method which estimates the threshold for RTL features. We evaluate our model to count optimal threshold from the train sample. If the value less than that threshold than the label is 0, 1 otherwise. Using the estimated threshold we label new data points.

– **Logistic regression** is a statistical model used to predict the probability of occurrence of an event. The model is the following:

$$Pr(c = 1|\mathbf{x}) = f(\boldsymbol{theta}^T \mathbf{x} + b),$$

where $f(z) = \frac{1}{1+e^{-z}}$ is a logistic function, and $\boldsymbol{theta}$ and $b$ are the parameters of the model estimated from data [9].
– **Random Forest** is an ensemble classifier that is developed on the basis of majority voting of decision trees. Various number of decision trees are generated over bootstrap samples of the training dataset. The final decision is based on aggregation of the predictions obtained by all the decision trees. Thus, a Random Forest allows to find complicated relationships in the data, but at the same time more resistant to retraining [12].
– **AdaBoost** is another method that combines classifiers into ensembles. The key feature of this is that it introduces weights for all objects. At each iteration, the weights of each incorrectly classified object increase, so a new classifier ensemble focuses attention on these objects. AdaBoost is sensitive to noise and rejects in data. However, it is less prone to overfitting compared with other algorithms of machine learning [8].
– **Gradient Boosting** is an ensemble method that builds an ensemble of trees one-by-one, then the predictions of the individual trees are summed. The next decision tree tries to cover the discrepancy between the target function $f(x)$ and the current ensemble prediction by reconstructing the residual. Thus, an iteration process is constructed, in each iteration of which the loss function is minimized by gradient descent [10]. Another nice feature of Gradient Boosting is ability to treat imbalanced-classification problems after simple modifications [14].

### 4.4 Resampling Techniques

The problem at hand is imbalanced: number of objects with big enough earthquakes is significantly higher, than number of objects with relatively small earthquakes with magnitude smaller than 5. So, it is natural to apply machine learning heuristics that can deal with the problem. Here we consider two simple, yet efficient techniques for resampling: we modify our initial training sample to make more emphasis on minor class objects:

– Oversampling—increase weights of minor class objects in a random way
– Undersampling—drop some major class objects to balance number of instances of each class in a training sample.

## 5 Results

Selection of used metrics is motivate by imbalanced of the problem at hand. In addition to common Precision, Recall and ROC AUC we look at F1-score and PR AUC. See definitions of the metrics in Appendix A.

### 5.1 Models for RTl Features Generated with a Single Pair of Hyperparameters $(r_0, T_0)$

We use the following grid for **RTL** parameters:

$$\mathbf{r_0} = [10, 25, 50, 100],\ \mathbf{t_0} = [30, 90, 180, 365].$$

The grid to find the best way to predict earthquakes was multidimensional and included $(r_0, t_0)$ array, different values of adjusted magnitude, number of test samples and different classifiers.

Results of the grid with the best test size and best $t0$ for every $r0$ are given in Table 1. The best result is for Gradient Boosting [7], for the greater size of cylinder and for the smaller gap between time of measurements and the target event. Gradient Boosting works better than Logistic Regression because of non-linear behaviour of dependence of the magnitude of earthquakes on RTL features. Both these approaches work better than Major RTL because of difficulties of the RTL model, so more sophisticated machine learning approaches work better than simple threshold rule for the problem of earthquakes prediction.

**Table 1.** Results for different values of hyperparameters for generation of RTL features: the better results are for bigger size of cylinder

| r0 | best t0 | Algorithm | Precision | Recall | F1 | ROC AUC | PR AUC |
|---|---|---|---|---|---|---|---|
| 10 | 180 | Logistic Regression | 0.54 | 0.36 | 0.43 | 0.64 | 0.53 |
| | | Random Forest | 0.62 | 0.51 | 0.56 | 0.76 | 0.69 |
| | | AdaBoost | 0.63 | 0.50 | 0.56 | 0.77 | 0.83 |
| | | Gradient Boosting | 0.62 | 0.52 | 0.56 | 0.80 | 0.69 |
| | | Major_RTL | 0.57 | 0.47 | 0.47 | 0.52 | 0.77 |
| 25 | 90 | Logistic Regression | 0.72 | 0.58 | 0.64 | 0.70 | 0.44 |
| | | Random Forest | 0.79 | 0.51 | 0.62 | 0.74 | 0.68 |
| | | AdaBoost | 0.75 | 0.67 | 0.71 | 0.77 | 0.53 |
| | | Gradient Boosting | 0.82 | 0.6 | 0.69 | 0.79 | 0.77 |
| | | Major_RTL | 0.67 | 0.52 | 0.62 | 0.70 | 0.60 |
| 50 | 180 | Logistic Regression | 0.91 | 0.84 | 0.88 | 0.83 | 0.67 |
| | | Random Forest | 0.91 | 0.84 | 0.87 | 0.80 | 0.70 |
| | | AdaBoost | 0.83 | 0.96 | 0.89 | 0.81 | 0.74 |
| | | Gradient Boosting | 0.89 | 0.92 | 0.91 | 0.87 | 0.73 |
| | | Major_RTL | 0.60 | 0.74 | 0.74 | 0.80 | 0.71 |
| 100 | 180 | Logistic Regression | 0.94 | 0.94 | 0.94 | 0.80 | **0.94** |
| | | Random Forest | **0.97** | 0.90 | 0.94 | 0.89 | 0.90 |
| | | AdaBoost | 0.96 | 0.96 | 0.96 | 0.92 | 0.90 |
| | | Gradient Boosting | 0.95 | **0.98** | **0.97** | **0.93** | **0.94** |
| | | Major_RTL | 0.90 | 0.86 | 0.89 | 0.83 | 0.89 |

### 5.2 Aggregation of a Number of RTL Features

In this section we used 16 calculated RTL features as inputs for a machine learning models. Quality of obtained models are available in Table 2. The improvement compared to the single best RTL feature with $M_c = 50, R_c = 50, \delta_c = 10, T_c = 180$ is insignificant. We conclude, that we can't get more information going this way.

**Table 2.** Result for usage of multiple RTL features with different hyperparameters for generation

| Algorithm | Precision | Recall | F1 | ROC AUC | PR AUC |
|---|---|---|---|---|---|
| Gradient Boosting (best single RTL) | 0.97 | 0.96 | 0.96 | 0.95 | 0.94 |
| Major_RTL | 0.90 | 0.85 | 0.89 | 0.82 | 0.89 |
| Logistic Regression | 0.94 | 0.95 | 0.94 | 0.81 | 0.94 |
| Random Forest | 0.97 | 0.90 | 0.94 | 0.89 | 0.90 |
| AdaBoost | 0.97 | 0.96 | 0.97 | 0.92 | 0.90 |
| Gradient Boosting | 0.95 | 0.98 | 0.97 | 0.93 | 0.93 |

### 5.3 Usage of Resampling Techniques

There are a number of resampling techniques that can deal with imbalanced classification problems [5,6]. Here we consider oversampling, undersampling and balanced approaches. The results are in Table 3. For each approach we used machine learning algorithm that provided the best overall performance. We see that both undersampling and oversampling improve the quality of models.

**Table 3.** Model quality for application of imbalanced classification approaches. In these experiments we used a training sample of smaller size

| Approach | Algorithm | Precision | Recall | F1 |
|---|---|---|---|---|
| Initial | Gradient Boosting | 0.84 | 0.59 | 0.59 |
| Oversampling | Random Forest | 0.94 | 0.99 | 0.97 |
| Undersampling | Random Forest | 0.96 | 0.87 | 0.91 |

## 6 Conclusion

We considered the problem of middle term earthquakes prediction. Usage of Machine learning provide an improvement compared to the state-of-the-art major_RTL method.

In particular the model based on Gradient Boosting with RTL features as inputs provide the best performance. However, for many cases RTL features generated using one set of hyperparameters one suitable RTL feature is not worse than a set of them generated using a set of hyperparameters. Another heuristic that improves quality of the earthquake prediction models is resampling to deal with imbalance of a given data.

**Acknowledgements.** The research was partially supported by the Russian Foundation for Basic Research grant 16-29-09649 ofi m.

## A Quality Metrics for Classification Problem

Introduce necessary definitions: Classification problem can be formulated as whether this object belongs to the target class or not.

- True Positive—if the object belongs to the target class and we predict that it belongs.
- True Negative—if the object doesn't belong to the target class and we predict that it doesn't.
- False Positive—if the object doesn't belong to the target class but we predict that it does.
- False Negative—if the object belongs to the target class but we predict that it doesn't.

The **precision** score quantifies the ability of a classifier to not label a negative example as positive. The is the probability that a positive prediction made by the classifier is positive. The score is in the range [0, 1] with 0 is the worst, and 1 is perfect. The precision score can be defined as:

$$\textbf{Precision} = \frac{TruePositive}{TruePositive + FalsePositive}$$

The **recall** score quantifies the ability of the classifier to find all the positive samples. It defines what part of positive samples have been chosen by classifier as positive. The score is in the range [0, 1] with 0 is the worst, and 1 is perfect.

$$\textbf{Recall} = \frac{TruePositive}{TruePositive + FalseNegative}$$

The **F1-score** is a single metric that combines both precision and recall via their harmonic mean. It measures the test accuracy and reaches its best value at 1 (perfect precision and recall) and worst at 0.

$$\textbf{F1} = 2\frac{PrecisionRecall}{Precision + Recall}$$

**ROC AUC score** counts the curve area under the Roc_curve. Roc_curve is the plot True Positive rate from the False Positive rate, which defines as

$$TruePositiveRate = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$FalsePositiveRate = \frac{FalsePositive}{FalsePositive + TrueNegative}$$

ROC AUC score measures the quality of binary classifier. The best value is 1, value 0.5 is equal to random classification.

**PR AUC score** counts the curve area under the Precision_Recall_curve: Precision from Recall. Precision-Recall is a useful measure of success of prediction when the classes are very imbalanced. The perfect classifier curve ends in $(1.0, 1.0)$ and has area under it that equals 1.

## References


1. Aksoy, S., Haralick, R.M.: Feature normalization and likelihood-based similarity measures for image retrieval. Pattern Recogn. Lett. **22**(5), 563–582 (2001)
2. Asencio-Cortés, G., Martínez-Álvarez, F., Morales-Esteban, A., Reyes, J.: A sensitivity study of seismicity indicators in supervised learning to improve earthquake prediction. Knowl.-Based Syst. **101**, 15–30 (2016)
3. Asim, K.M., Awais, M., Martínez-Álvarez, F., Iqbal, T.: Seismic activity prediction using computational intelligence techniques in northern Pakistan. Acta Geophys. **65**(5), 919–930 (2017)
4. Asim, K., Martínez-Álvarez, F., Basit, A., Iqbal, T.: Earthquake magnitude prediction in Hindukush region using machine learning techniques. Nat. Hazards **85**(1), 471–486 (2017)
5. Burnaev, E., Erofeev, P., Papanov, A.: Influence of resampling on accuracy of imbalanced classification. In: Eighth International Conference on Machine Vision (ICMV 2015), vol. 9875, p. 987521. International Society for Optics and Photonics (2015)
6. Burnaev, E., Erofeev, P., Smolyakov, D.: Model selection for anomaly detection. In: Eighth International Conference on Machine Vision (ICMV 2015), vol. 9875, p. 987525. International Society for Optics and Photonics (2015)
7. 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. ACM (2016)
8. Freund, Y., Schapire, R.: A short introduction to boosting. J. Jpn. Soc. Artif. Intell. **14**, 771–780 (1999)
9. Friedman, J., Hastie, T., Tibshirani, R., et al.: Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann. Stat. **28**(2), 337–407 (2000)
10. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**, 1189–1232 (2001)
11. Gutenberg, B., Richter, C.: Seismicity of the Earth and associated phenomena (1951)
12. Ho, T.K.: Random decision forests. In: Proceedings of 3rd International Conference on Document Analysis and Recognition, vol. 1, pp. 278–282. IEEE (1995)
13. Huang, Q.: Seismicity pattern changes prior to large earthquakes-an approach of the RTL algorithm. Terr. Atmos. Oceanic Sci. **15**(3), 469–492 (2004)
14. Kozlovskaia, N., Zaytsev, A.: Deep ensembles for imbalanced classification. In: 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 908–913. IEEE (2017)
15. Panakkat, A., Adeli, H.: Neural network models for earthquake magnitude prediction using multiple seismicity indicators. Int. J. Neural Syst. **17**(01), 13–33 (2007)
16. Rouet-Leduc, B., Hulbert, C., Lubbers, N., Barros, K., Humphreys, C.J., Johnson, P.A.: Machine learning predicts laboratory earthquakes. Geophys. Res. Lett. **44**(18), 9276–9282 (2017)
17. Sobolev, G., Tyupkin, Y.: Low-seismicity precursors of large earthquakes in Kamchatka. Volcanol. Seismol. **18**, 433–446 (1997)
18. Utsu, T., Ogata, Y., et al.: The centenary of the Omori formula for a decay law of aftershock activity. J. Phys. Earth **43**(1), 1–33 (1995)