



Clustering activity at Mt Etna based on volcanic tremor: A case study

Giuseppe Nunnari¹

Received: 9 November 2020 / Accepted: 11 March 2021
© The Author(s) 2021

Abstract

This paper deals with the classification of volcanic activity into three classes, referred to as Quite, Strombolian and Paroxysm. The main purpose is to give a measure of the reliability with which such a classification, typically carried out by experts, can be performed by Machine Learning algorithms, by using the volcanic tremor as a feature. Both supervised and unsupervised methods are considered. It is experimentally shown that at least the Paroxysm activity can be reliably classified. Performances are rigorously assessed, in comparison with the classification made by expert volcanologists, in terms of popular indices such as the f_1 -score and the Area under the ROC curve (AuC). The work is basically a case study carried out on a dataset recorded in the area of the Mt Etna volcano. However, as volcanic tremor is a geophysical signal widely available, considered methods and strategies can be easily applied to similar volcanic areas.

Keywords Supervised classification · Unsupervised clustering · K-means · Fuzzy-c means · Gaussian mixed models · Volcanic activity

Introduction

Nowadays, classifying volcanic activity is delegated to human experts, as it requires a timely consultation of various kinds of instrumental data. This activity is then all the more demanding as the time interval between samples is smaller. It is therefore interesting to know to what extent Machine Learning (ML) approaches can help to solve this problem by automatically processing instrumental data. In particular, among several kinds of geophysical data, such as seismic, ground deformation, chemical, etc., in this paper we chose to consider the seismic tremor, which is widely available in areas of active volcanism. Although the use of volcanic tremor for classifying volcanic activity is not new in literature, see for instance (Langer et al. 2011) who applied Self-Organized Maps (SOM) and Fuzzy clustering, with the aim of detecting imminent eruptive activity at Mt Etna, we must consider that a clear measure of the reliability of ML algorithms to cluster volcanic activity,

based on the use of volcanic tremor, is still incomplete and could benefit from additional contributions. In this work, we aim to contribute to this field, providing a reliable estimation of the performances of both supervised and unsupervised classifiers, adopting as metric the f_1 -score, which, as is known, represents the harmonic mean between *Precision* and *Recall* and in terms of Area under the ROC (AuC). Furthermore, we will compare the f_1 index obtained by using the unsupervised approaches against a supervised classification algorithm, in order to estimate the performance loss. Results presented in this paper can integrate those recently presented by Spampinato et al. (2019), who measured the reliability of a warning system, based on the use of seismic tremor measured on Etna, in terms of the AuC index, estimating $AuC \simeq 0.8$. The validity of the results shown in this work is based on a dataset, recorded at Mt Etna from January 2011 and December 2015, which contains a quite high number of paroxysm episodes (namely 48) and therefore allowing a reliable estimate of the classifiers performances.

Review papers concerning clustering approaches are provided by Liao (2005), Xu and Tian (2015), and Bano and Khan (2018), while some application of Machine Learning for the specific field of volcanic seismic signals processing has been recently provided by Malfante et al. (2018). Recent papers dealing with ML application in solid Earth geoscience and seismology have been provided by Bergen et al. (2019) and Kong et al. (2019). Other ML methods

Communicated by: H. Babaie

✉ Giuseppe Nunnari
giuseppe.nunnari@dieei.unict.it

¹ Dipartimento di Ingegneria Elettrica Elettronica e Informatica, Università degli Studi di Catania, Viale A. Doria, 6, 95125, Catania, Italy

and applications in geophysics, with specific examples to classify tremor data can be found in the recent book (Langer et al. 2020).

The paper is organized as follows: in Section “[Statement of the problem](#)” the problem is stated. The considered dataset and pre-processing steps are described in Section “[Dataset and preliminary analysis](#)”. Considered supervised classification methods are listed in Section “[Supervised classification of volcanic activity](#)”, together with a description of the classification metrics. Numerical results concerning supervised methods are reported in Section “[Supervised classification](#)”. Unsupervised classification approaches considered and results, in comparison with the supervised approaches are reported in Section “[Numerical results concerning unsupervised classification](#)”. In Section “[Discussion and Conclusions](#)”, conclusions are drawn. A formal description of considered ML approaches and hints about their implementation is reported in [Appendix A](#).

Statement of the problem

The problem of classifying volcanic activity consists in associating a label to each kind of activity observed at the summit crater area. In particular, in this work we will follow the line drawn by Cannavó et al. (2017) and subsequently continued by Hajian et al. (2019), who classify the volcanic activity into three classes based on the following description:

- The class of Quite (**Q**) state, labeled also as 0, indicates quiet, i.e. no relevant activity or unknown activity, characterized by low amplitudes of all physical features that can be related with the volcanic activity. Representing classes with a double symbol, numerical and literal, make easier on one hand the visualization as a timeseries of classes (numeric representation) and the readability (literal representation).
- the class of Strombolian (**S**) activity, labeled also as 1, indicates Strombolian activity, essentially characterized by a mildly explosive activity, with medium amplitude of seismic tremor RMS, shallower source of the seismic tremor, presence of clustered infrasonic events, no eruption column, but possible ash emissions.
- the class of Paroxysm (**P**) activity, labeled also as 2, indicates paroxysm, i.e. an energetic activity with lava fountains, characterized by high amplitude of seismic tremor RMS, presence of clustered infrasonic events, shallower source of the seismic tremor, eruption column formation and ash emissions.

For example, Mt Etna would appear to an observer as represented in [Fig. 1](#), during a Strombolian activity and



Fig. 1 Example of Strombolian activity at Mount Etna. The image was kindly provided by the INGV-Sez. di Catania

as in [Fig. 2](#) during a paroxysmal. Comparing the two images it is evident that the ejected matter reaches even some thousands of meters for Paroxysm and only a few tens or some hundreds of meters for Strombolian activity. This is representative of the fact that the energy levels corresponding to the various kinds of volcanic activity can be quite different.

In a well-monitored volcanic area, such as Mt Etna, the onset of this type of activity can be detected by various instruments such as cameras or Doppler radars. However, in remote volcanoes, very often these kinds of sensors are not present, while seismic monitoring networks are more frequently installed.

The classification of volcanic activity at Mt Etna by using a supervised Bayesian Network (BN) classifier, which discriminates the three kinds of activity mentioned above, based on five features, namely the seismic RMS tremor, the

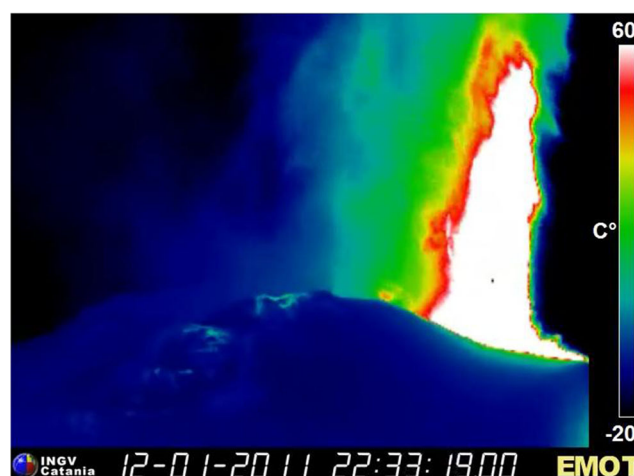


Fig. 2 Example of Paroxysm activity at Mt Etna. The image was kindly provided by the INGV-Sez. di Catania

infrasonic RMS tremor, the depth of the volcanic source, the radar Doppler and the tilt derivative, was first proposed by Cannavó et al. (2017). Other supervised classification approaches, namely Decision Tree (DT) and K-Nearest-Neighbor (KNN), on the same dataset have been applied by Hajian et al. (2019).

Dataset and preliminary analysis

The dataset taken into account in this study was recorded from January 1, 2011 to December 31, 2015 and after processing was sampled every 10 minutes. During this time interval the eruptive activity of Mt Etna was rich in episodes characterized by phases of degassing, alternating with explosive and effusive activities, to periodic eruptions from fissures opened on its flanks. A detailed description of the activity occurred is outside the scope of this work and the interested readers can refer to Cannavó et al. (2017) and references therein. For the purposes of this paper, it is enough to mention that expert volcanologists have labeled the 262944 samples into 213190 samples of **Q** activity, 49048 of **S** activity and 706 samples of **P** activity. Samples labeled as **Q**, **S** and **P** are the 81.08%, 18, 65% and 0.27% of the total, respectively. This time-series of classes, shown in Fig. 3, will be, again, considered in this paper for evaluating to what extent classification of volcanic activity can be performed by unsupervised classifiers (i.e. clustered) and estimate the performance in comparison with supervised ones. Classification will be carried out considering the seismic tremor as the only feature.

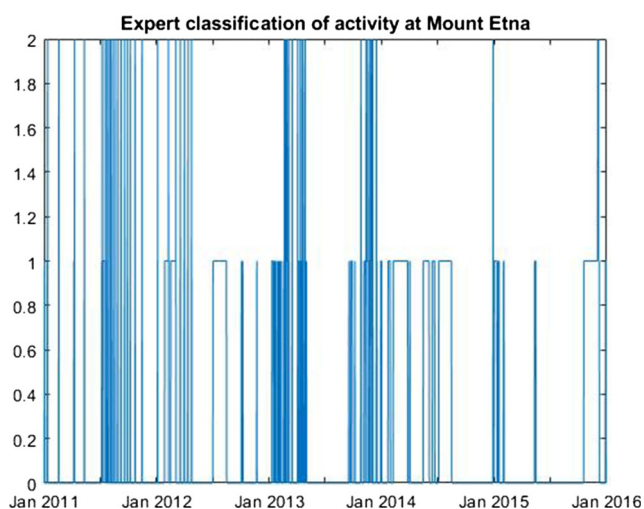


Fig. 3 Time series of classes from January 2011 to December 2015, recorded at Mt Etna with 10 m sampling time: 0, 1 and 2 represents Quiet, Strombolian and Paroxysmal activity, respectively. The dataset was kindly provided by the INGV-Sez. di Catania

Volcanic seismic tremor

Volcanic seismic tremor, also referred to as harmonic tremor, is a sustained release of seismic and infrasonic energy typically associated with the underground movement of magma, the venting of volcanic gases from magma, or both. It is a long-duration release of seismic energy, with distinct spectral lines, that often precedes or accompanies volcanic activity. More generally, volcanic tremor is a long-duration continuous signal generated by a spatially distributed source, which contrasts distinctly with transient sources of seismic radiation, such as earthquakes and explosions. The relation between long-period events and an imminent eruption was first observed by Chouet (1996). As indicator of the level of volcanic seismic tremor, the root mean square (RMS) amplitudes of seismic signals in the tremor band have been routinely recorded at the Istituto Nazionale di Geofisica e Vulcanologia (INGV), Osservatorio Etneo, for real-time volcano monitoring, through the so-called permanent seismic network, equipped with broadband (40 s cutoff period), three-component seismometer, which records data in real time at a sampling rate of 100 Hz. The original signals are then filtered in the band [0.5, 5.5] Hz, since, as suggested by (Chouet 1996), it concentrates the main part of released seismic energy. The geographic coordinates of most of the considered seismic stations are reported in Table 1.

Regression analysis of tremor time series

RMS tremor recorded at different stations of Mt Etna are linear correlated. To illustrate this aspect, we have computed linear regression models of the form:

$$y(t) = mx(t) + b \quad (1)$$

where:

- $x(t)$ is the tremor value measured at time t in a given reference station, assumed as input;
- $y(t)$ is the tremor estimated by the linear model, into another station, assumed as output;
- m and b are the slope and the offset coefficients, respectively.

Figure 4, report, as an example, the scatter plots of tremor recorded at the couples ESLN-ECZM, assuming the tremor recorded at the ESLN station as input and that at ECZM as output; the strong linear correlation appears evident. In order to highlight that this is almost independent of the particular class of activity, samples belonging to different classes were represented by different colors. The regression coefficient R , the slope and the offset, for the whole set of stations are reported in Table 2. It can be

Table 1 StationID and geographic coordinates of the considered seismic stations

StationID	Latitude	Longitude	Elevation	Location
ECNE	37.7653	15.0018	2946	Etna Cratere Nord Est
ECBD	37.7802	15.0865	1465	Etna Case Bada
ECZM	37.7313	14.9041	1391	Etna Case Zampini
ECPN	37.7437	14.9865	3038	Etna Cratere del Piano
EMCN	37.7912	15.0336	1916	Monte Conca
EMFO	37.7357	15.0902	1209	Etna Monte Fontane
EMFS	37.7196	14.9979	2552	Etna Monte Frumento
EMNR	37.8168	15.026	1845	Etna Monte Nero
EMPL	37.679	14.9703	1484	Etna Monte Parmentelli
EMSG	37.8208	4.9498	1435	Etna Monte Spagnolo
EPDN	37.7659	15.0168	2862	Etna Pizzi Deneri
EPIT	37.8113	15.0567	1657	Etna Pozzo Pitarrone,
ESLN	37.6934	14.9744	1787	Etna Serra La Nave
ESPC	37.6925	15.0274	1655	Etna Serra Pizzuta Calvarina
ESVO	37.7731	14.9469	1736	Etna Monte Scavo

seen that almost all stations exhibit R close to 1. The lower R values corresponds to stations, namely ECNE, ECPN, EPDN, located very close to the active craters areas, which are, probably for this reason, characterized by very high tremor levels and/or other kinds of local phenomena. For classification purposes, the high linear correlation of tremor time series, implies that the dataset dimensionality can be significantly reduced, as it will be discussed in the next section. For now, we mention here that this property suggested to us a strategy for filling missing data at several stations. Indeed, in the considered time interval a not negligible quantity of missing data has plagued the network, as shown in the last column of Table 2. It is possible to

see that in the considered time interval, the percentage of missing data ranges between about 19% and 38%, and in particular, only two stations are below 20%.

In order to overcome this shortcoming, which would have involved the loss of many samples useful for classification, among the various candidate strategies, we have chosen to fill missing data by the average value of the tremor calculated using expression (2), learned from (Cannavò et al. 2017),

$$T_{RMS}(t) = \frac{1}{n} \sum_{i=1}^n \frac{RMS_i(t) - \mu_t\{RMS_i(t)\}}{\sigma_t\{RMS_i(t)\}} \quad (2)$$

In this expression, the index, i refers to the n seismic stations, μ_t represents the median operator over two years previous the time t , and σ_t represents the interquartile range over the same period. An example of filled time series at the station EBCN, in comparison with the true one, is reported in Fig. 5.

Supervised classification of volcanic activity

In this paper, to perform the supervised classification, of volcanic activity at Mount Etna, the six classification models listed below were considered:

- A Fisher Discriminant model, here simply referred to as DISC, implemented by using the *fitcdiscr* function of the Matlab Statistical and Machine and Learning Toolbox, choosing a linear kernel. More details about this kind of classifiers are reported in Section A.1.1.
- A Multiclass error-correcting output model. In particular, as ECOC, in this paper we refer to an Ensemble

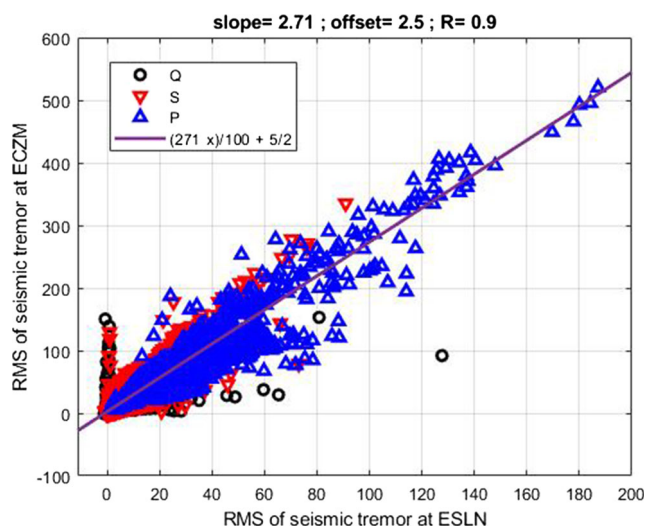


Fig. 4 Scatter plot and corresponding linear regression model at the ECZM - ESLN couple of stations

Table 2 Regression model coefficients assuming the tremor measured at the ESLN station as input. The last column reports the missing data in percent

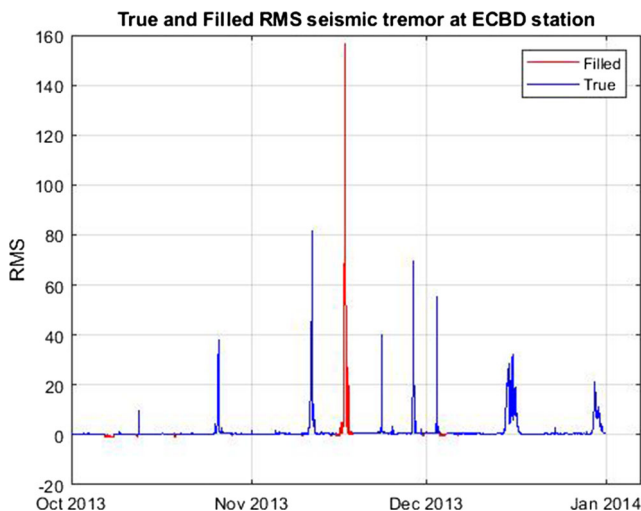
Station	R	slope	offset	Missing %
EBCN	0.91	1.78	2.42	38.86
ECBD	0.78	0.55	0.19	21.96
ECNE	0.07	2.07	3.75	30.03
ECZM	0.90	2.71	2.50	27.15
ECPN	0.33	0.41	0.41	22.81
EMCN	0.68	1.07	0.55	29.78
EMFO	0.89	0.56	0.17	31.83
EMFS	0.89	1.36	0.52	19.27
EMNR	0.52	0.44	0.24	25.26
EMPL	0.6	0.77	0.24	19.14
EMSG	0.69	0.41	0.29	25.29
EPDN	0.14	2.27	1.84	32.91
EPIT	0.32	0.57	0.62	32.72
EPLC	0.77	1.5	1.79	32.16
ESLN	1	1	0	24.90
ESPC	0.99	0.68	0.17	23.48
ESVO	0.94	0.69	0.59	27.51

multiclass classifier implemented by using the *fitcecoc* Matlab function, using a Support Vector Machine (SVM) algorithm, with linear kernel and One-Versus-One (OVO) coding scheme. More details are reported in Section A.1.2.

- An Ensemble model, here referred to as ENSE, implemented through the *fitensemble* function of the Matlab Statistical Toolbox. We have adopted as basic learner a *Adaboost2* algorithm, building a medium coarse tree with a maximum number of splits set to 20, a maximum number of cycles up to 30 and a learning

rate set to 0.1. A short description of Ensemble models is reported in Appendix A.1.3.

- A K-Nearest Neighbor model, here referred to as KNN, implemented through the *fitcknn* function of the Matlab library, setting Euclidean distance metrics and 10 samples as the maximum number of neighbors. More details in Section A.1.4.
- A Naive-Bayes model, here referred to as NBYE, based on the *fitcnb* function of the Matlab library, with normal kernel distribution and unbounded support. More details in Section A.1.5.
- A Decision Tree for multiclass classification, here simply referred to as TREE, implemented through the Matlab function *fitctree*, growing a medium coarse tree, with a maximum number of splits up to 20. The Gini's index was adopted as a split criterion. More details in Section A.1.6.

**Fig. 5** Filled and original tremor time series at the ECBD station

There are no particular reasons behind the choice of these methods, other than they are among the most popular. Obviously, many other supervised classification algorithms could have been chosen, including the popular neural networks. We want to stress here that the main aim of this work is not to compare classification algorithms, but to measure to what extent the described problem can be solved using widely available and easily implementable algorithms. More deep details concerning these approaches can be found in popular textbooks such as Bishop (2006), Hastie et al. (2008), and Goodfellow et al. (2016) or directly from the MATLAB Statistical and Machine Learning Toolbox (Inc 2017).

Reducing the dimensionality of the dataset

As well known, see for instance (Liu and Motoda 2008), performances of classification algorithms significantly depend on the number of features. Therefore, reducing the dimensionality of the dataset is a good practice (Mohamed 2020). We have performed this task by using the Principal Component Analysis (PCA), which consists of projecting each data point onto the main directions in the feature space, in order to obtain lower-dimensional data, while preserving as much of the data variability as possible. The first principal component can be defined as the direction that maximizes the variance of the projected data. Similarly, the i^{th} principal component is the direction orthogonal to the first $(i - 1)^{\text{th}}$ principal ones that maximizes the variance of the projected data.

For our data set, the cumulative variability explained, versus the number of principal components, is shown in Fig. 6. It can be seen that taking the first two components it is possible to account for about 96% of the dataset variability. This percentage rises to 99%, if the first three principal components are considered. Therefore, the PCA analysis points out that the original 17-D dataset can be reduced to 2-D or 3-D. In this paper, in order to be more conservative, we have chosen to consider 3 principal components. However, the effectiveness of reducing the dataset dimensionality by using the PCA must be done with some caution, as a known drawback of the PCA approach for classification problems is that the variance is preserved regardless of the classes (see for instance (Lei et al. 2017)). Thus, a subspace with large variance is not necessarily one in which classes are well separated.

In addition to the reduction of the dataset dimensionality through the PCA, we have explored the possibility of

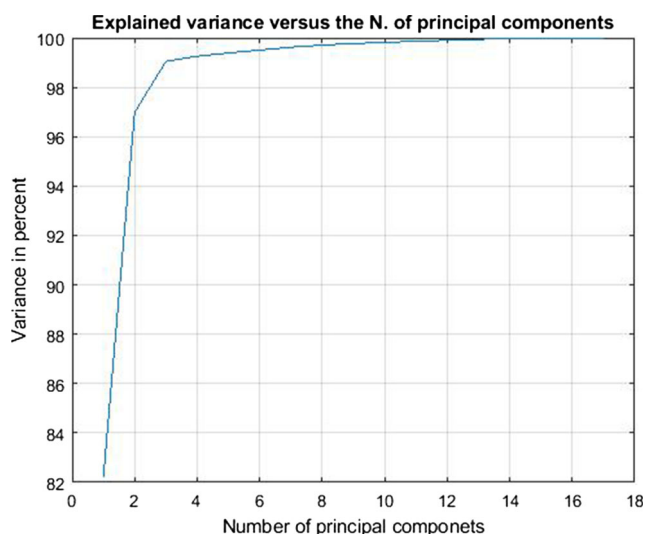


Fig. 6 Cumulative explained variance versus the principal components

performing this task by heuristically choosing two stations of the network and finally by averaging the seismic tremor among all the stations in the network through expression (2). Summarizing, the following three reduced data set were considered to perform the supervised classification:

1. The 3-D dataset represented by the first three PCA components;
2. a 2-D dataset consisting of tremor recorded at two selected stations of the network, namely ESLN and EBCN;
3. the 1-D dataset obtained averaging the RMS tremor over the whole network, by using expression (2).

The stations ESLN and EBCN, were selected according to the following criteria:

- The time series recorded at these stations are not affected by very high tremor levels, as for instance observed at stations ECNE, EPDN and EPIT.
- From trials performed, it seems that this couple of stations allow a good classification of class **P**. However, it is to be stressed, that other choices would be possible, without significantly affecting the classifier performances.

Classification metrics

Given a classification experiment, let us indicate as $P(i)$ and $N(i)$ the number of actual positive and actual negative cases in the i_{th} class, respectively. Moreover, let $TP(i)$, $TN(i)$, $FP(i)$ and $FN(i)$ the number of true positive, true negative, false positive and false negative cases, respectively, recognized by the classifier, for the i_{th} class. Referring to these quantities, the following rates can also be defined:

$$TPR(i) = \frac{TP(i)}{P(i)} = \frac{TP(i)}{TP(i) + FN(i)} = 1 - FNR(i) \quad (3)$$

$$TNR(i) = \frac{TN(i)}{N(i)} = \frac{TN(i)}{TN(i) + FP(i)} = 1 - FPR(i) \quad (4)$$

$$FNR(i) = \frac{FN(i)}{P(i)} = \frac{FN(i)}{FN(i) + TP(i)} = 1 - TPR(i) \quad (5)$$

$$FPR(i) = \frac{FP(i)}{N(i)} = \frac{FP(i)}{FP(i) + TN(i)} = 1 - TNR(i) \quad (6)$$

The meaning of the above indices can be expressed as follows:

- The $TPR(i)$ expresses the proportion of actual positives that are correctly classified by the model as belonging to the i_{th} class. Best values of TPR approach to 1, while in the worst case TPR approach 0. The TPR is referred to also as *specificity* or *Recall* (r).
- The $TNR(i)$ expresses the proportion of actual negatives that are correctly classified as not belonging

to the i_{th} class. As for the TPR , best values of TNR approach 1, while worst values approaches 0. The TNR is referred to also as specificity of selectivity.

- The $FNR(i)$ expresses the proportion of false negatives in the i_{th} class, with respect to all actual positives in the same class. Of course in the best case FNR approaches 0, while in the worst case approaches 1.
- The $FPR(i)$ expresses the proportion of false positives in the i_{th} class with respect to the total number of actual negatives in the same class. Similar to the FNR in the best case FNR approaches 0, while in the worst case approaches 1.

Another useful index, is the Positive Predicted Value (PPV) or simply *Precision*, which, for the generic class i is defined as:

$$PPV(i) = \frac{TP(i)}{TP(i) + FP(i)} = 1 - FDR(i) \tag{7}$$

In expression (7), FDR stands for False Discovery Rate.

For the purposes of this paper we also make use of the f_1 -score, defined as:

$$f_1(i) = \frac{2TP(i)}{2TP(i) + FP(i) + FN(i)} \tag{8}$$

It is easy to verify that indicating as $r(i)$ the TPR for the i_{th} class and as $p(i)$ the PPV , the f_1 can be also written as:

$$f_1(i) = 2 \frac{r(i)p(i)}{r(i) + p(i)} \tag{9}$$

which highlights that it is the harmonic mean between $p(i)$ and $r(i)$. Therefore, f_1 is a more reliable measure of performance than the simple *Precision* and/or *Recall*. For this reason, in this paper we will always show f_1 , instead of the simple *Precision* and/or *Recall*.

A useful way to collect most of the performance indices shown above is the so-called Confusion Matrix (CM). On the CM, see for instance the next Section “[Numerical results concerning unsupervised classification](#)”, the rows correspond to the predicted class (Output Class) and the columns correspond to the true class (Target Class). The diagonal cells correspond to observations that are correctly classified. The off-diagonal cells correspond to incorrectly classified observations. Both the number of observations and the percentage of the total number of observations are shown in each cell. The column on the far right of the plot shows the percentages of all the examples predicted to belong to each class that are correctly and incorrectly classified, i.e the PPV and the FDR . The row at the bottom of the plot shows the percentages of all the examples belonging to each class that are correctly and incorrectly classified, i.e. the TPR and the FNR , respectively. The cell

in the bottom right of the plot shows the overall *Accuracy*. The accuracy can formally be described by using expression (10)

$$Accuracy = 100 \frac{\sum_{n=1}^N I(C(x_n) = y_n)}{N} \tag{10}$$

where

- $I(g)$ is a function that returns 1 if g is true and 0 otherwise,
- $C(x_n)$ the class label assigned by the classifier to the sample x_n
- y_n the true class label of the sample x_n
- N is the number of samples in the testing set.

Another useful index for evaluating the degree of accuracy and reliability of a statistical classification is Cohen’s Kappa, since it compensates for random hits. It can be computed (see for instance (Tallon-Ballesteros and Riquelme 2014)) starting from the CM, by using expression (11)

$$\kappa = \frac{N \sum_{i=1}^m CM_{ii} - \sum_{i=1}^m C_{i\text{true}} C_{i\text{pred}}}{N^2 - \sum_{i=1}^m C_{i\text{true}} C_{i\text{pred}}} \tag{11}$$

where:

- m is the dimension of the confusion matrix CM, i.e. the number of classes.
- N is the number of samples in the testing set.
- $CM_{ii}, i = 1, \dots, m$ the entries of the CM main diagonal.
- $C_{i\text{true}}$ the true number of labels of class i .
- $C_{i\text{pred}}$ the predicted number of labels of class i .

The range of Kappa values extends from $-1 \leq \kappa \leq 1$, with positive one indicating strong agreement, negative one indicating strong disagreement, and zero indicating chance-level agreement.

Another useful tool for evaluating the reliability of supervised classifiers is represented by the Receiver Operating Characteristic (ROC) metric and in particular the so-called area under curve AuC . ROC curves typically feature true positive rate on the Y axis, and false positive rate on the X axis. This means that the top left corner of the plot is the ideal point, since it is characterized by a false positive rate of zero, and a true positive rate of one. Even though this is not very realistic, it does mean that a larger area under the curve (AUC) is usually better. For a specific application of the ROC curves to evaluate the reliability of a warning system in volcanic areas, the readers can see the recent paper (Spampinato et al. 2019). For more general further details about metrics concerning classifiers, see Powers (2011).

Supervised classification

Preliminary considerations

Before starting with the description of supervised classification results, it can be useful to make some preliminary considerations regarding the separability of the three considered classes, using the seismic tremor as the only feature. While in Section “Reducing the dimensionality of the dataset” we have drawn attention to the fact that the tremor recorded in the various stations is linearly correlated, which implies that the dimensionality of the dataset can be significantly reduced, here, we want to point out that classes, as labeled by the experts, are largely overlapping in the feature space of the RMS tremor. To justify this statement, we can look at Fig. 4, from which it is easy to realize that the three classes are largely overlapping. This is more evident between classes **Q** and **S** which therefore, regardless of the classification technique, we may expect that will result poorly separable. Among the three classes, class **P** seems to have more chances to be correctly classified. To further confirm that the three classes are not easily separable, by using the seismic tremor as the only feature, we have applied the Silhouette criterion (Rouseeuw 1987). The test was performed by using the K-means algorithm and the search was extended in the range from 1 up to 6 clusters. Results reported in Fig. 7 show that that the highest Silhouette value occurs at 2 clusters, suggesting that this is the optimal number of clusters, regardless of the dataset, among the considered ones. It is also to be observed that the K-means operating on the 3-D data set was not able to contribute, since it indicates a silhouette value independent of the number of classes.

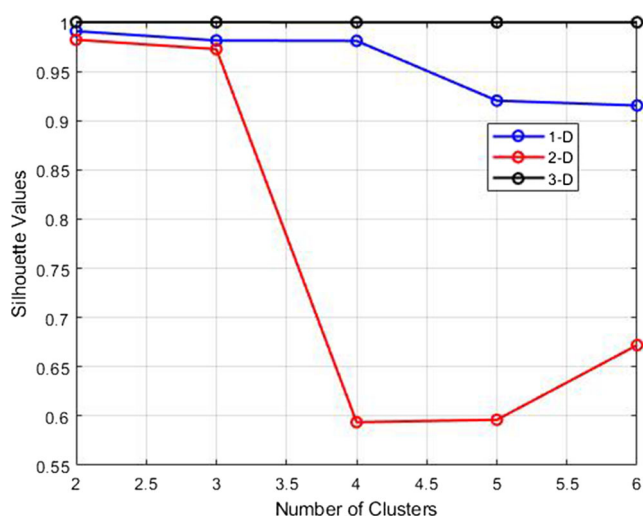


Fig. 7 Silhouette values versus the number of clusters for the three reduced dataset

Assessing the generalization capabilities of classifiers

A popular strategy to estimate the generalization capabilities of a model is to divide the available samples in at least two subsets for *training* and for *testing*, respectively. The former is used for tuning the model parameters and the latter for assessing its performance on fresh data. However, sometimes the testing set is used in some way during the training, for instance to understand when the training must be stopped. In this case, a third set of data, referred to as the *validation* set is introduced, in order to be more confident that performances are evaluated on a really fresh dataset.

Since we have a limited number of samples in the **P** class, a validation set is not realistic. To overcome this drawback, we have adopted the *k-fold cross* validation strategy. This technique consists of splitting the dataset into *k* groups, after shuffling the samples randomly. Then for each unique group the following steps are performed:

- A group is held out and used for testing.
- The remaining groups are used for training a model.
- The fitted model is evaluated on the testing set.
- The obtained score is then retained while the model is discarded.

This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model $k - 1$ times. At the end of the rotation process, the skill of the model is evaluated using the sample of model evaluation scores.

In the next section, we report validation accuracy computed for the three different reduced dataset, as above discussed.

Numerical results

We start with the global indices, i.e. computed averaging over the classes, namely the Accuracy, the f_1 index and Cohen’s K, which are shown in Fig. 8.

Roughly speaking, it is possible to say that in terms of *Accuracy*, there are not significant differences between the various classifiers and to the use of a particular kind of reduced dataset. Instead, the global f_1 index points out a slight advantage using the ENSE and the KNN.

As for the Cohen *K* index, it is always higher than 0.6, thus meaning that the classification is globally enough reliable. Furthermore, also this index contributes to assign an advantage to the use of ENSE and KNN.

However, as stressed above, for our application, non all classes have the same importance. In order to evaluate the reliability in discriminating individual classes, we report in Fig. 9, the single class f_1 indices. The figure highlights

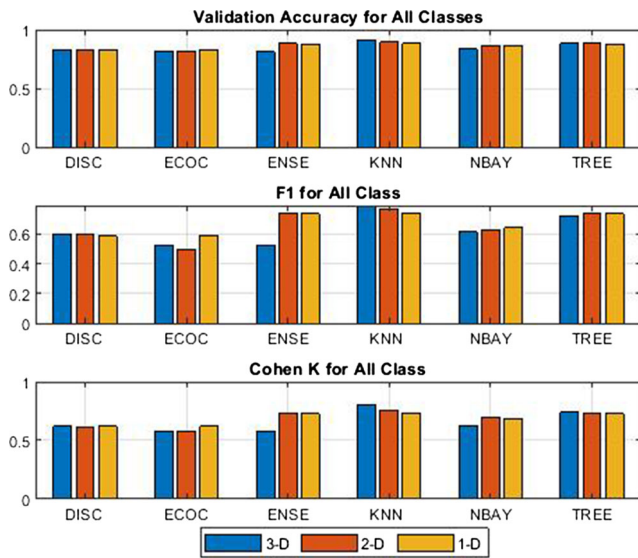


Fig. 8 Global indices for the three options

that this index, regardless of the classification method and the data set, is quite high for the **Q** class, but limited for the **S** class, which is therefore poorly discriminated. Another aspect, probably the most interesting for the purposes of this work, is that the f_1 index for the **P** class is about 0.7 for almost all the considered models, and in particular for the DISC, ENSE and KNN classifiers, regardless of the reduced dataset. However, even if it may seem trivial, with regard to the classification of class **P**, there is no substantial difference even to operate with the 1-D dataset.

The ROC curves for the six classifiers are shown in Fig. 10. It is possible to see that regardless for the considered classifiers the **P** class is always best classified among the

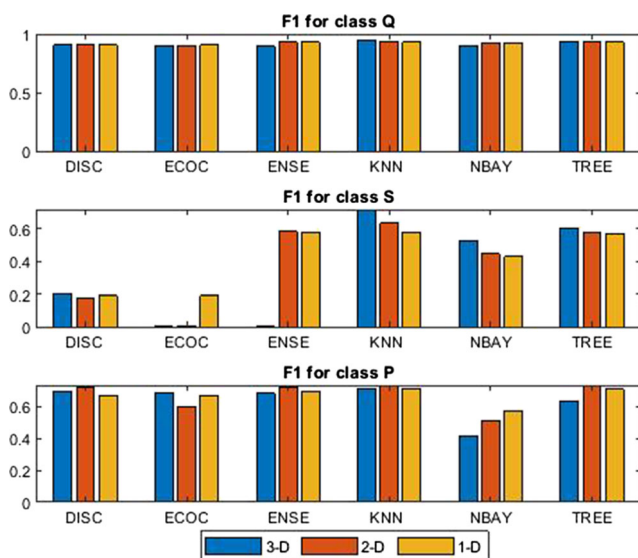


Fig. 9 f_1 evaluated for each class and each classification approach

others. In more detail, the Area under curve (AuC) for the six classifiers and for each class is reported in Fig. 11. It is possible to see that almost all classifiers work well for classifying **P** class samples, but perform poorly for the **Q** and **S** classes. On average, for the **P** class the AuC index is 0.87, while for the **Q** and **S** class are 0.65 and 0.64, respectively.

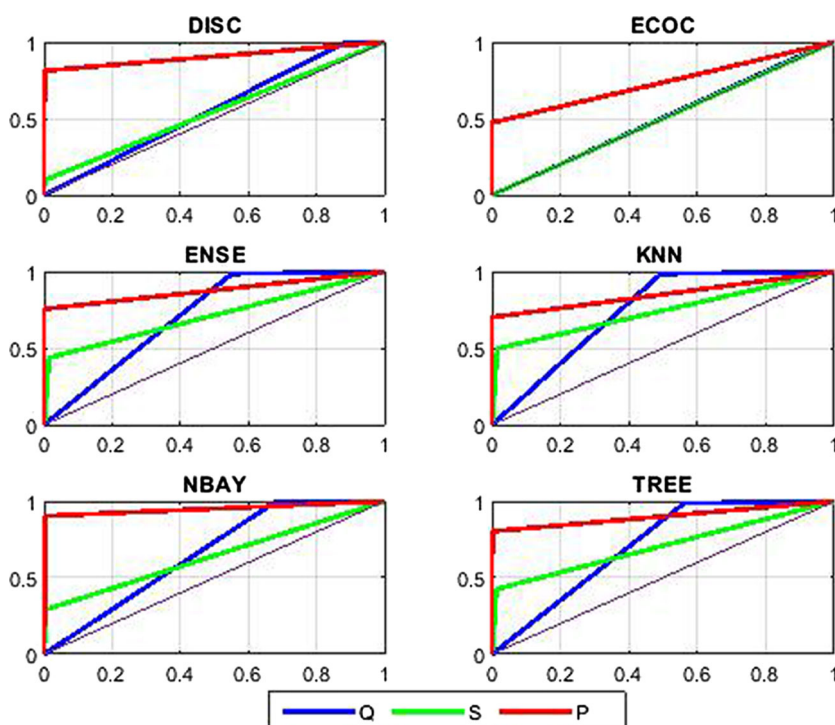
Operating with the 2-D dataset offers the possibility to visualize the boundary regions, as shown in Fig. 12. The regions in red, green and blue colors represent couples of values classified as class **Q**, **S** and **P**, respectively. In most of the cases, the shape of these areas can be easily interpreted. For instance, looking at the DISC, which was trained as a classifier with linear boundaries, it is possible to see that low values of RMS at both the ESLN and EBCN are attributed to the **Q** class, low values at ESLN and high at EBCN are attributed to the **S** class and, finally, high values at both stations are attributed to the **P** class. An easy interpretation can also be found for the NBAY, ECOC, ENSE and TREE classifiers, while the interpretation of the KNN is a little more difficult. Indeed, the classification regions for this classifier exhibit, in the middle-upper part of the plan, an area assigned to class **S**, that the other classifiers definitely assign to class **P**. This is due to the fact that not only classes **Q** and **S** are partially overlapping in the expert classification, but also classes **S** and **P**.

Supervised classification by using a more balanced data set

Among the criticisms that can be made to the results shown in the previous section, there could be that of having carried out the classification on data set in which the number of samples in the various classes are strongly unbalanced, especially comparing class **Q** and class **P**. Indeed, as mentioned in Section “Dataset and preliminary analysis”, the original data set, from now referred as the Full dataset, contains 262944 samples labeled into 213190 samples of class **Q**, 49048 of class **S** and only 706 of class **P**, which in percent are the 81.08%, 18,65% and 0.27%, respectively. Therefore, one might be led to think that reducing the weight of the **Q** class samples, which are overwhelmingly, there could be some advantage in terms of performance. To address this question, we carried out the classification experiment described below.

From the Full dataset we extracted a new dataset, in the following referred as the Subset, which includes all the **P** samples of the Full data set, almost all the **S** samples, but a smaller amount of samples of the **Q** class. In order to extract such a subset we have considered that the Full dataset contains 48 paroxysmal episodes occurred on Mount

Fig. 10 ROC curves for the six classifiers trained using the 2-D dataset



Etna during 2011-2015. With the aim of preserving all **P** samples, we extracted 48 data windows, centered on the starting date of each paroxysmal episode and spanning a prefixed number of samples before and after this date. The time series of classes obtained in this way is shown in Fig. 13.

Such a time series consists of 10278 samples, of which 5834 labeled as **Q**, 3738 as **S** and 706 as **P**. This means that in the Subset the weights of **Q**, **S** and **P** are 56.76%, 36.37% and 6.87% respectively.

With the obtained Subset, the models of six supervised classifiers, were obtained, trained, by using the *k*-folding technique. These new classifiers were subjected to classifying the Full dataset and the performance indices were estimated and compared with those of the corresponding classifiers, trained and tested with the Full dataset. Such a comparison, in terms of the f_1 index, evaluated for the single classes, is shown in Fig. 14.

The following considerations can be made:

- The f_1 indices for the **Q** class are almost equal for the two intercompared cases, except that the ECOC model trained with the subset was not able to classify samples of this class.
- As concerning the **S** class, the f_1 index is lower than 0.6 for all classifiers and in particular DISC and ECOC perform very poorly. It is to be stressed that the ECOC model trained with the Subset was not able to correctly classify any samples of the Full dataset assigned to the **S** class.
- For the **P** class there are not substantial differences for the two inter-compared cases.

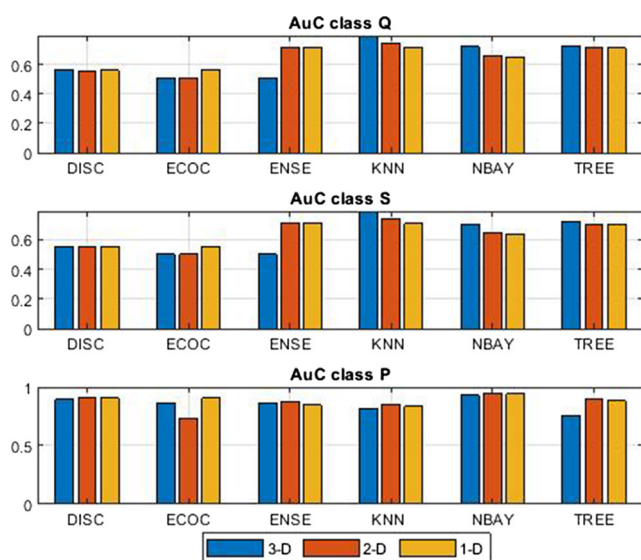
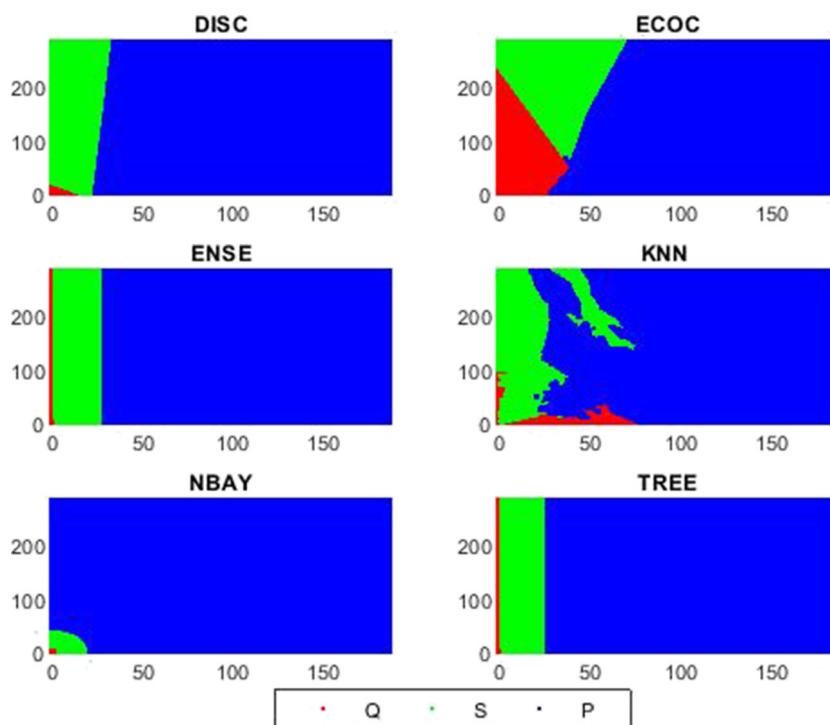


Fig. 11 Area under the ROC for all methods and classes.

In summary, from experimental results, it seems that there are no advantages training the classifiers with the Subset, if it is used for classifying samples of the Full dataset, which of course is what a user would like to do. On the contrary, some classes are worse classified compared to the case in which the training is carried out on the Full dataset.

Fig. 12 Classification regions computed for the six classifiers, trained by using the k-fold approach (axes units represent RMS of seismic tremor). The axes are graduated in RMS of seismic tremor; on the abscissa and ordinate we have the values measured at the ESLN and EBCN stations, respectively



Numerical results concerning unsupervised classification

In this section we report numerical results obtained by using the following three clustering approaches:

- k-means.
- Fuzzy c-means.
- Gaussian mixture models.

A short description of each algorithm is provided in Section A. The *kmeans*, *fcm* and *fitgmdist* functions of the Matlab

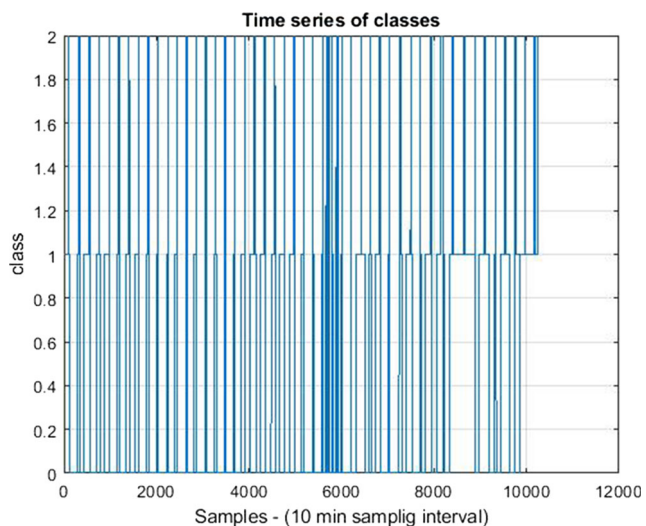


Fig. 13 Time series of the classes for the Subset

Statistical and Machine Learning Toolbox, were considered for implementing the three approaches listed above. The criteria for setting the parameters, left to the user are indicated below. During the model fitting, in this case, dealing with unsupervised classification, the dataset was not split, i.e. the whole dataset was given as input to the classifiers.

Similarly to what was done for the supervised classification, we report results obtained performing the clustering

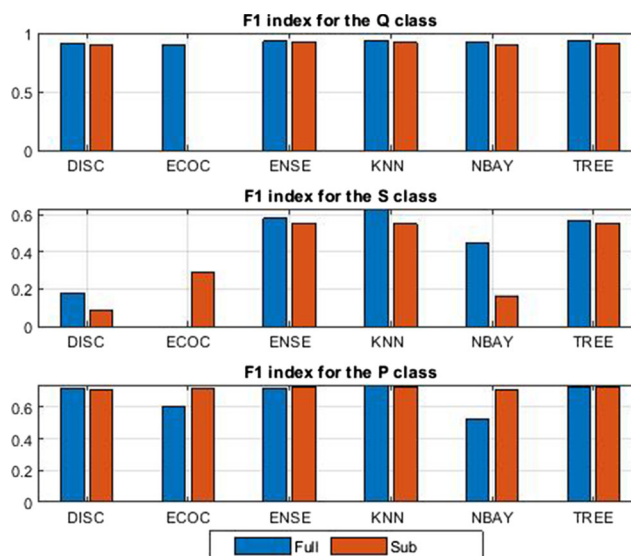


Fig. 14 f_1 index obtained for the model trained and tested with the Full data set and the model trained with the Subset and tested with the Full set

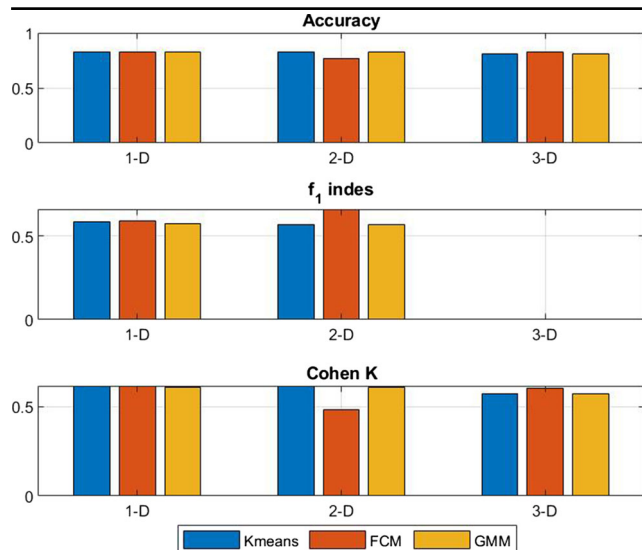


Fig. 15 Global Accuracy, f_1 and Cohen K for the 1-D, 2-D and 3-D dataset and for the three considered clustering approaches

on three dataset with reduced dimension, presented in Section “Supervised classification”. The performances of the unsupervised clustering were evaluated assuming as true the time series of classes shown in Fig. 3. The global indices, in terms of Accuracy, f_1 and Cohen K, for the three dataset are shown in Fig. 15. In the figure, in correspondence with the f_1 index relating to the 3-D dataset, the values should not be considered zero, but *Not a Number*. The reason is that, for all trials carried out, choosing different options for the tunable parameters, the considered clustering algorithms, were not able to correctly recognize any sample of the class **P** class. For this reason, in the following of this section we will report results for the 2-D and 1-D data sets, only.

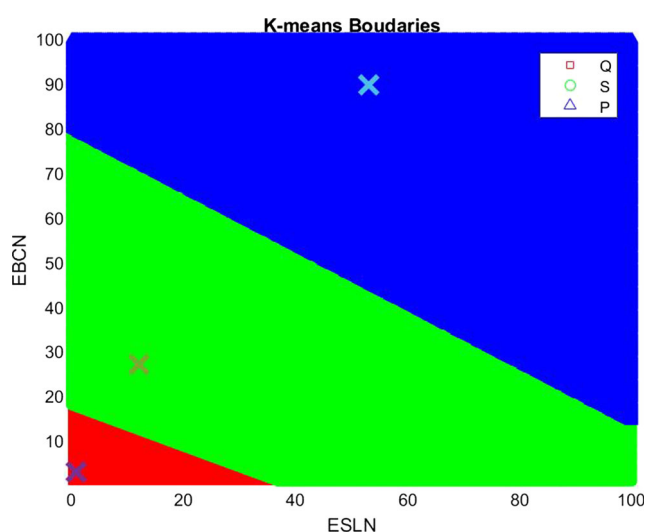


Fig. 16 Classification regions obtained performing the K-means clustering; cluster centers for the three classes are also indicated by the \times symbol

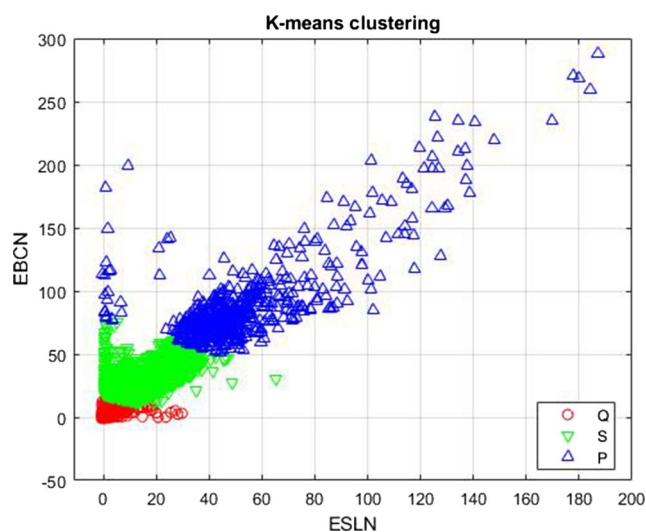


Fig. 17 Samples classified by the K-means (axes units represent RMS of seismic tremor)

Results by using the k-means and the 2-D dataset

As explained in Section A.2.1, using this approach it is necessary to choose the kind of distance. Based on trials performed, among the *Euclidean*, *cityblock*, *cosine* and *correlation*, we have chosen the former one. The classification regions of the K-means performed on 2-D dataset in the ESLN-EBCN feature plane, are shown in Fig. 16. It can be seen that the algorithm determines well-distinct regions: the one in the lower corner in red color, characterized by low values of the tremor at both stations, represents the **Q** class, the intermediate one in green represents the **S** class and, finally, the region in blue color, characterized by high values of tremor at both the stations, represents the **P** class. For each class, the searched cluster centers are reported in Table 3. Applying the classification mask shown in Fig. 16 to the 2-D dataset, we obtained the pattern distribution shown in Fig. 17. Performances of the k-means classifier, in terms of Confusion Matrix, assuming as target the expert classification, are reported in detail in Fig. 18.

The behavior of the K-means algorithm, referring to the **P** class, can be considered relatively satisfactory. Indeed, of the 706 samples labeled by the expert as belonging to **P**, 424 have been correctly attributed, while 255 were recognized

Table 3 Class centers using the K-means in the ESLN-EBCN plane

Class	μ_1	μ_2
Q	0.8697	2.9700
S	11.9562	26.9495
P	52.4986	89.0279

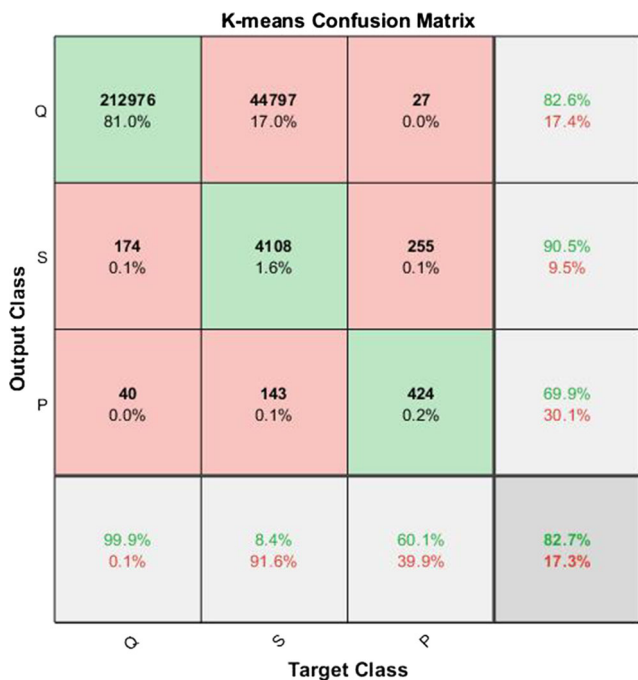


Fig. 18 Confusion Matrix obtained classifying the 2-D dataset by using the K-means algorithm

as S and only 27 as Q, therefore giving $TPR\% = 60.1$ and $PPV\% = 69.9$. A weak point of the K-means classifier, for the considered dataset is the low value of TPR (8.4%) for the class S. Indeed, 44797 samples of the total 49048 assigned by the human expert as belonging to the S class, are instead recognized as Q. Obviously, this contributes to lowering the classifier’s performance also with respect to the Q class.

The ROC curves obtained applying the K-means algorithm to the 2-D dataset are reported in Fig. 19. The figure confirms that in terms of ROC curves, the only class classified with acceptable accuracy by K-means is class P, being $AuC = 0.81$. For both the remaining classes Q and S we have $AuC = 0.54$, therefore indicating a behavior very close to a random classifier.

Results by using the FCM and 2-D dataset

Working with the FCM algorithm, in addition to the choice of the metrics, which also in this case we have chosen as Euclidean, it is possible to tune the m parameter (see Section “A.2.2”), which controls the degree of fuzzy overlap among the clusters. After some trials, we have experimentally found that best results were obtained by setting $m = 2$, which is also the default value assumed by the *fcm* Matlab function. The cluster centers obtained for the FCM model, reported in Table 4, are a bit different from the ones searched by the K-means algorithm.

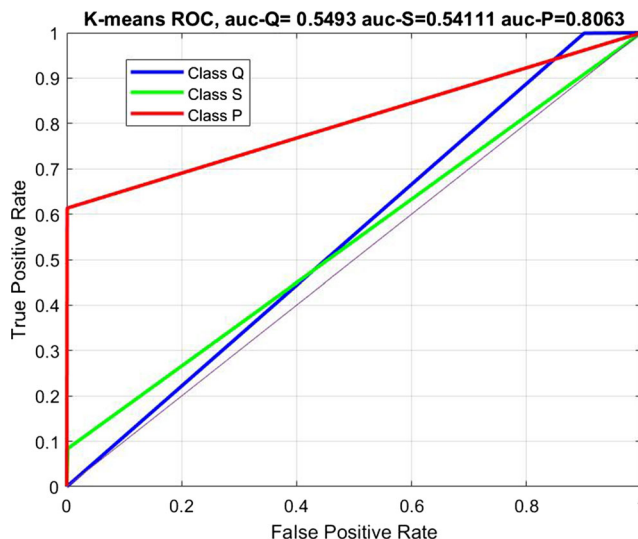


Fig. 19 ROC curves of the K-means classifier for the three classes

The classification regions provided by the FCM in the ESLN-EBCN feature plane are shown in Fig. 20. Compared with the analogous mask obtained by the K-means it is possible to see that the area assigned to the Q-region is smaller, while that assigned to the P one is larger. The performances of this classifier are reported in Fig. 21 in terms of Confusion Matrix. It is possible to see the FCM, compared with the K-means, gives higher TPR for the P and S classes, but with a lower PPV. In particular, for the P class we have a TPR of 82.9%, but with a $PPV = 56\%$, which in terms of f_1 means $f_1 = 0.67$. Therefore, by using the FCM, we get a higher number of samples correctly classified but at the price of a higher level of false positive.

The ROC curves obtained applying the FCM algorithm to the 2-D dataset are reported in Fig. 22 which shows that for this classifier the AuC assumes the values $AuC = 0.91$ for the P class and about 0.67 for both the S and Q classes. Therefore, while the P class is classified with an acceptable accuracy, the Q and S are poorly classified. However, in terms of ROC curves, the FCM performs better than the K-means.

Results by using the GMM

Tuning a GMM model offers to the user the possibility of choosing different settings for the covariance matrices.

Table 4 Class centers using the FCM algorithm in the ESLN-EBCN plane

Class	μ_1	μ_2
Q	0.6598	2.0578
S	1.8531	6.5577
P	39.7895	69.0634

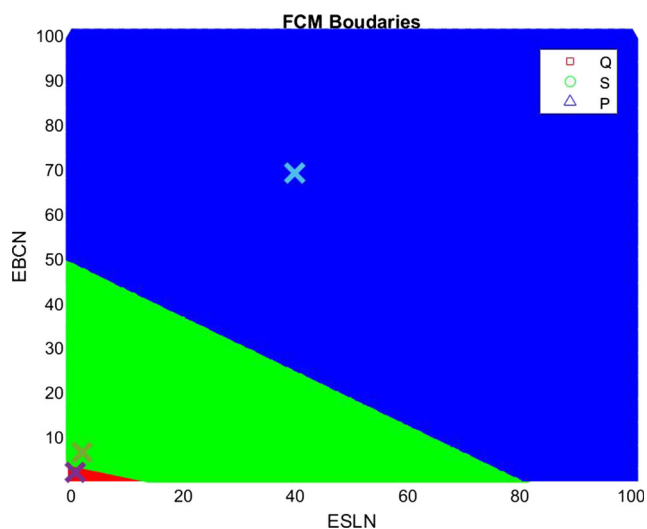


Fig. 20 Classification regions obtained by using the FCM algorithm. The cluster centers are indicated by the × symbol

Indeed, even if ultimately the covariance matrices are automatically computed by the software, the choice of their structure is left to the user. After some trials, with the aim of maximizing the model performances against the **P** class, we have found convenient to assume, diagonal covariance matrices, shared among the classes. The obtained mixing coefficients π_i and mean μ_i parameters of the GMM model are reported in Table 5. The classification mask, in the feature plane ESLN-EBCN, is shown in Fig. 23.

	Q	S	P	
Q	176871 67.3%	23526 8.9%	4 0.0%	88.3% 11.7%
S	36242 13.8%	25140 9.6%	117 0.0%	40.9% 59.1%
P	77 0.0%	382 0.1%	585 0.2%	56.0% 44.0%
	83.0% 17.0%	51.3% 48.7%	82.9% 17.1%	77.0% 23.0%
	Q	S	P	
Target Class				

Fig. 21 Confusion Matrix obtained classifying the 2-D dataset by using the FCM algorithm

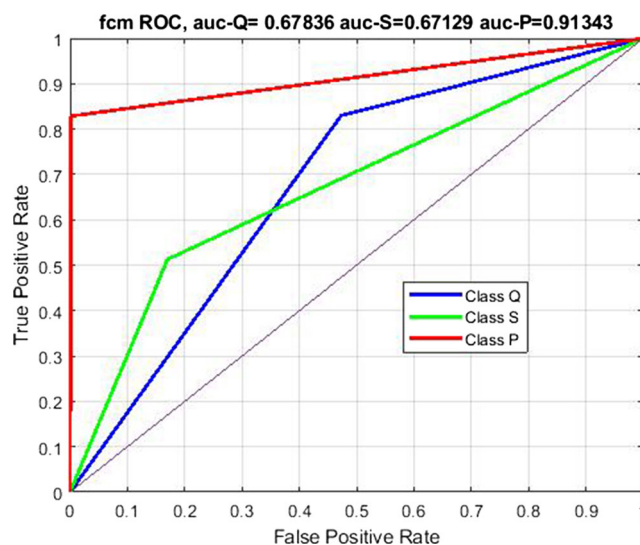


Fig. 22 ROC curves for each class obtained by using the FCM algorithm

It is possible to see that also this model, similar to the k-means and FCM ones, simply classifies as **Q** low RMS values recorded at both ESLN and ECBN, as **S** low RMS at ESLN and high at ECBN and, finally, as **P** high RMS values at both the stations. Obviously, the position of the boundary regions are different with respect to K-means and FCM.

The classifier performances in terms of CM are reported in Fig. 24, while the ROC curves are reported in Fig. 25. This last figure shows that for this classifier the *AuC* is 0.80 for the **P** class and about 0.54 for both the **S** and **Q** classes. Therefore, while the **P** class is classified with an acceptable accuracy, the **Q** and **S** are poorly classified. In terms of *AuC*, the GMM performs worse than the FCM and similar to the Kmeans.

Comparison among the unsupervised classifiers for the 2-D data

A direct comparison among the three unsupervised classifiers, in terms of global *Accuracy*, f_1 and Cohen's *K*, by using the 2-D dataset is reported in Fig. 26. It is possible to see that Kmeans and GMM perform quite similar in terms of all the three considered global indices. In particular, while they outperform the FCM model in terms of *Accuracy* and Cohen's *K*, the FCM performs better in terms of global f_1 .

Table 5 Components of the fitted GMM model

Component	π_i	μ_1	μ_2
1	0.982276	0.8756	3.0013
2	0.015458	12.9308	28.3762
3	0.002266	54.5323	88.9329

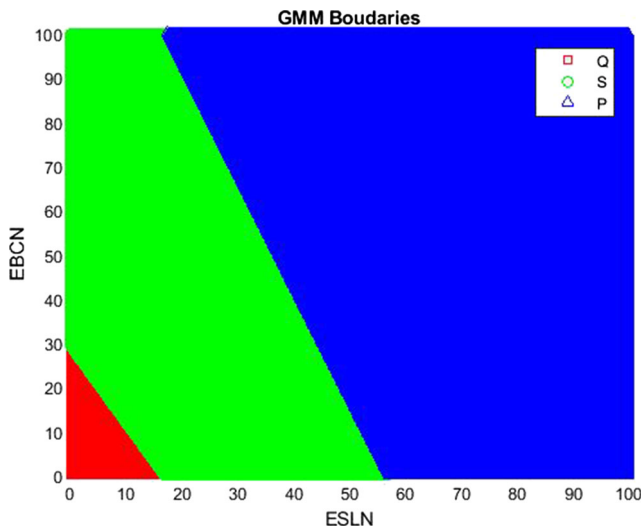


Fig. 23 Classification regions obtained by using the GMM algorithm

The performance indices for each single class are shown in Fig. 27. The figure shows that the f_1 score for class Q is quite high regardless of the clustering algorithm and so is the Accuracy. For class S both the f_1 and Accuracy indices are low. For class P, the f_1 index is about 0.7 regardless of the clustering method. Furthermore, it should be noted that the Cohen K index for class Q is very low, unlike the Accuracy index, thus indicating that the statistic on class Q is not very reliable. On the contrary, the statistic on class P is very reliable.

GMM Confusion Matrix

	Q	S	P	
Q	212997 81.0%	45251 17.2%	30 0.0%	82.5% 17.5%
S	155 0.1%	3669 1.4%	246 0.1%	90.1% 9.9%
P	38 0.0%	128 0.0%	430 0.2%	72.1% 27.9%
	Q	S	P	
	99.9% 0.1%	7.5% 92.5%	60.9% 39.1%	82.6% 17.4%
	Target Class			

Fig. 24 Confusion Matrix obtained by using the GMM model

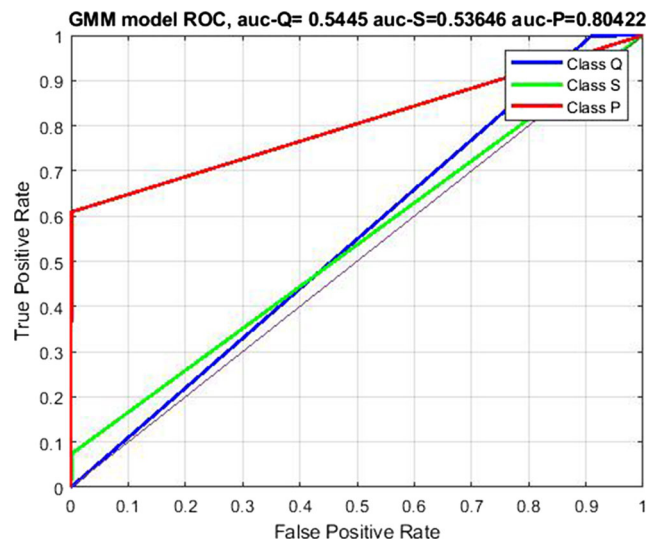


Fig. 25 ROC curves for the three classes obtained by using the GMM algorithm

Models in 1-D and comparison with 2-D

Operating in 1-D, i.e. training the models by using the average seismic tremor, estimated model parameters are reported in Tables 6 and 7 for the K-means and FCM model, respectively. The parameters of the fitted GMM model are reported in Table 8.

A comparison of the performance, in terms of f_1 and AuC, Accuracy and Cohen's K indices, between models fitted by using the 1-D and 2-D dataset, is reported in Figs. 28, 29. As a general comment it is easy to say that there are not meaningful advantage in performing the unsupervised clustering operating on the 2-D with respect to the 1-D dataset. This result which may appear trivial, is

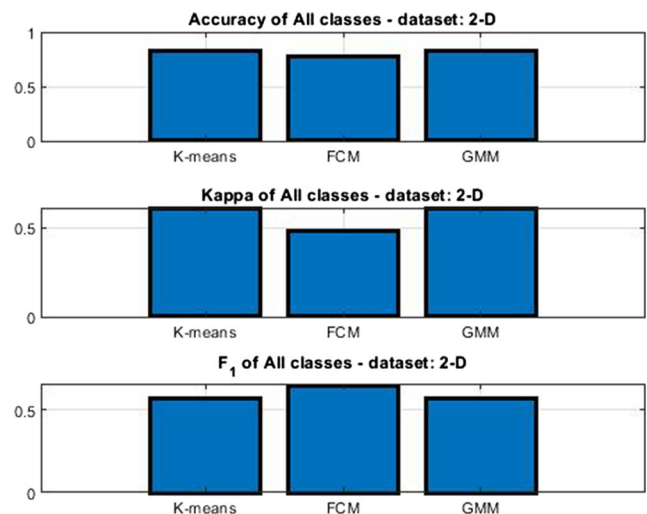


Fig. 26 Comparison among the unsupervised classifiers in terms of global Accuracy, f_1 and Cohen's K, by using the 2-D dataset

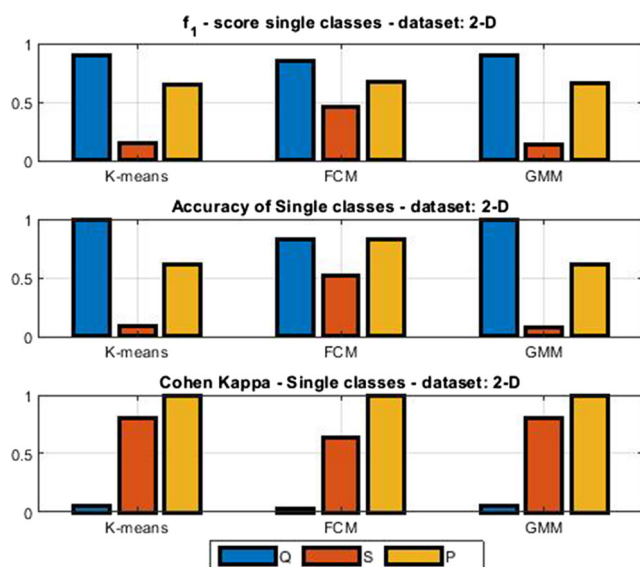


Fig. 27 Comparison among the unsupervised classifiers in terms of f_1 for each individual class by using the 2-D dataset

essentially due to the fact that the seismic tremor timeseries at different stations are strongly correlated, as discussed in the previous Section “Regression analysis of tremor time series”.

In more details, Fig. 28 shows that the f_1 index, averaged among the methods, is about 0.88 for the **Q** class, about 0.25 for the **S** class and about 0.66 for the **P** class. So it might seem that in terms of f_1 , both class **Q** and class **P** are well classified. In reality, if we look at the AuC index reported in Fig. 29, we realize that class **Q** has an AuC index of 0.58 while class **P** has an AuC of 0.84. Bearing in mind that a random classifier has an $AuC = 0.5$, we can conclude that while the value of $f_1 = 0.88$ is not representative of the true performance of the classifier with respect to class **Q**, the value average value of $f_1 = 0.66$ for class **P** is representative of the true capabilities of the classifier. In a few words we can conclude that the unique class reliably discriminated is class **P**.

Another result that could be interesting to observe is that the value of AuC for class **P**, estimated in this paper, could agree with that obtained by Spampinato et al. (2019), operating with different methodologies on the same area, who computed a value $AuC \sim 0.8$.

Table 6 Cluster centers using the Kmeans and the 1-D dataset

Class	μ
Q	0.2272
S	17.5085
P	65.8976

Table 7 Cluster centers using the FCM and the 1-D dataset

Class	μ
Q	0.1791
S	16.2816
P	62.1765

Measuring the performance of the FCM classifier assuming as true the output of the ENSE supervised classifier

In this section we report a comparison between one of the best among the unsupervised classification algorithms, in our case the FCM operating on the 1-D dataset, and one of the best supervised classifiers presented in the previous section. For this comparison we have chosen the ENSE classifier. In order to perform this comparison, we have computed a CM, assuming as target the time series obtained by the ENSE classifier and as output the one provided by the FCM. Such a CM is shown in Fig. 30. It is possible to see that:

- Supervised and unsupervised algorithms almost fully agree in classifying samples of class **Q**, being $TPR\% = 100$ and $PPV\% = 91.6$.
- The two kinds of classifiers almost fully agree also for the **P** class, being $TPR\% = 100$ and $PPV\% = 97.6$.
- Supervised and unsupervised algorithms do not fully agree with samples of **S** class, exhibiting $PPV = 100\%$ but $TPR = 18.5\%$.

This experimentally demonstrates that the unsupervised classification of RMS tremor can effectively replace the supervised one, limited to the clustering of paroxysm activity. In other terms, it is possible to reach the same level of performances, avoiding the laborious commitment of the experts or at least they could take the results provided the unsupervised classifier as a good starting point for further developments. Furthermore, without significant loss of performance, it is possible to carry out the classification on the 1-D dataset, that is, using the average RMS tremor time series, obtained using the expression (2).

Table 8 Parameters of the fitted GMM model and the 1-D dataset

Component	π_i	μ
1	0.981812	0.8756
2	0.015824	18.8321
3	0.002364	54.5323

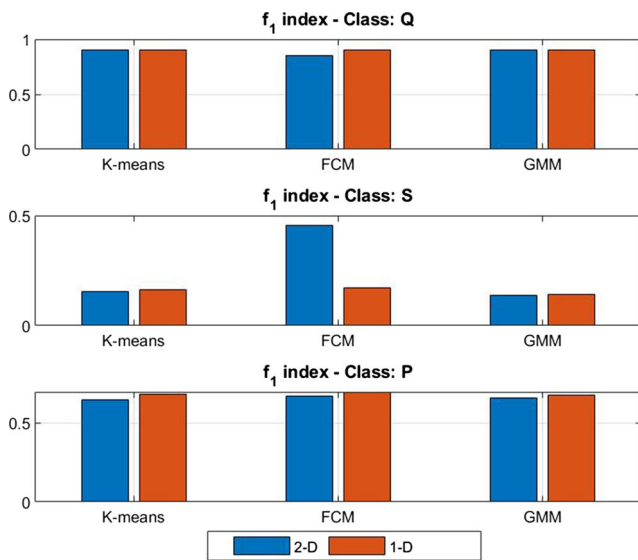


Fig. 28 Performance in terms of the f_1 index by using the 2-D and the 1-D dataset

Comparing the true and the unsupervised time series of class

The following discussion aims to show that the misclassified samples occur during the transition phases from quiet to paroxysm and vice versa. Indeed, the 706 samples classified by the experts as belonging to the **P** class, form actually 48 paroxysmal events (or patterns). A few of these episodes are shown in Fig. 31. The time series of classes provided by the unsupervised classifier is reported in yellow color, the average RMS tremor is reported in red color, and the expert class in blue. The figures also show that paroxysm is

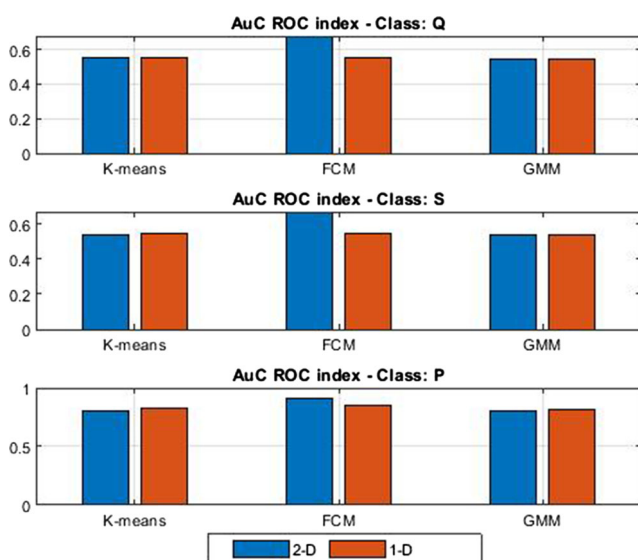


Fig. 29 Comparison in terms of AuC index by using the 2-D and the 1-D dataset



Fig. 30 Confusion Matrix obtained assuming as target the output of the ENSE supervised classifier and as output class the output of the FCM clustering algorithm

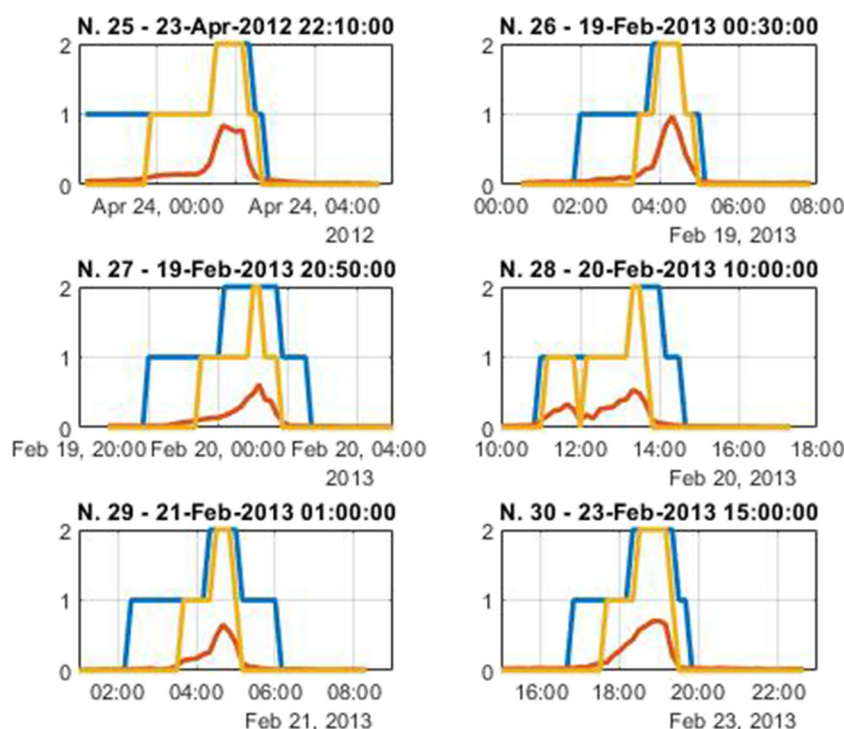
always preceded by Strombolian and usually also followed. It is also possible to see that the yellow and the blue curves usually agree during paroxysm, while the major differences can be observed during the Strombolian activity. It is also interesting to observe the close correlation between the average level of the RMS tremor (red curve) and the time series of class provided by the unsupervised classifier (yellow curve). Therefore, the greatest uncertainties concern the classification of the **S** episodes, for which the average level of the RMS tremor is not particularly different from the level it assumes during the activity classified as **Q**.

Discussion and Conclusions

In this work we have addressed both the problem of supervised and unsupervised clustering of volcanic activity, based on seismic tremor, a kind of geophysical signal widely available in many volcanoes around the world, in order to objectively evaluate to what extent popular clustering algorithms can automate this task, usually performed by human experts. Results achieved can be summarized as follows:

- All the considered classification algorithms, both supervised and non, agree that by using the RMS of tremor as the only feature, only events belonging to class **P** can be classified with a reliability that, measured

Fig. 31 Comparison among the true (blue color) and clustered (yellow) time series of classes for a few paroxysm episodes; the red curve is the average RMS tremor



in terms of the f_1 index, is about 0.7 and in terms of AuC about 0.8.

- Almost all the supervised classifiers considered are able to perform this task of classifying the class **P** events, but a limited superiority can be assigned to ENSE, KNN and TREE models. Similarly, among the considered unsupervised classifiers a limited superiority can be assigned to the FCM one.
- The greatest uncertainties of classifiers concern the attribution of events belonging to class **S**. In this regard, the RMS level of tremor alone is not enough to safely discriminate whether these samples are to be attributed to class **Q** or to class **S**.

As a final remark, it is to stress that the expert classification is not necessarily based exclusively on the evaluation of physical signals and the dating of the events can therefore be imprecise. Therefore, a basic problem arises, concerning the establishment of objective criteria that experts must consider to classify volcanic activity. However, this problem is beyond the scope of this paper and is left for future developments.

Appendix A

The purpose of this section is to provide a formal description of the ML methods considered in this paper and to give some useful insights on how they have been implemented. All the software used in this work has been coded by

the author by using the standard Matlab Statistical and Machine Learning Toolbox library (Inc 2017). In the following we report a brief description of each method and references to papers where these methods were originally introduced.

A.1.1 Supervised classification

In this section, we provide a description of the six supervised approaches considered in the paper, starting with the Discriminant Analysis.

Discriminant analysis

Discriminant analysis, here referred as DISCR, is the evolution of the original Fisher discriminant, named for its inventor (Fisher 2008). The model for discriminant analysis works as follows:

- The class vector Y is generated assuming that the input data X can be reliably fitted assuming a Gaussian mixture distribution.
- For linear discriminant analysis, the model assumes the same covariance matrix for each class, i.e. only the means vary. It could be easily demonstrated that assuming different covariance matrices for each class, the boundary curves become, in general, quadratic.

Predictions are performed according with the following expression:

$$\hat{y} = \arg \min_{y=1, \dots, K} \sum_{k=1}^K \hat{P}(k|x)C(y|k) \tag{12}$$

where

- \hat{y} is the predicted class.
- K is the number of classes.
- $\hat{P}(k|x)$ is the posterior probability of class k for observation x .
- $C(y|k)$ is the cost of classifying an observation as y when its true class is k .

Therefore, the user can choose the type of boundary regions (linear or quadratic) based on the kind of assumption about the structure of covariance matrix (e.g equal or different for the various classes, diagonal or full etc).

Another choice left to the user is the cost function C . By default, the standard form of the cost matrix C , is $C_{i,j} = 1$ if $i \neq j$ and $C_{i,j} = 0$ if $i = j$. Entries for the C matrix can also be negative. In this paper we have assumed the standard form.

In the Matlab Statistical toolbox this method is implemented by the function *fitcdiscr* which allows several options. Among these we have chosen to operate with a linear discriminant type. This implies that we have assumed that all classes have the same covariance matrix. Furthermore the algorithm applies a regularization expression of the form

$$\hat{\Sigma}_{\gamma} = (1 - \gamma)\hat{\Sigma} + \gamma \text{diag}(\hat{\Sigma}) \tag{13}$$

where $\hat{\Sigma}$ is the empirical pooled covariance matrix and γ is the amount of regularization $0 \leq \gamma \leq 1$. In this paper we have set $\gamma = 0$. Moreover, we have adopted a double *logit* score transformation of data.

A.1.2 Error-correcting output codes

Error-Correcting Output Codes (ECOC) (Dietterich and Bakiri 1995) is an ensemble method designed for multi-class classification problem (i.e. $K > 2$). Indeed, while some algorithms, such as Decision Tree, Naive Bayes can handle multi-class problems directly, ECOC is a meta method which combines L binary classifiers in order to solve the multi-class problem. All kinds of classifiers can be used as binary classifiers, such as Discriminant, KNN, Naive Bayes, SVM, Decision Trees etc. The number L depends on the kind of coding considered among several available, such as one-versus-all (OVA), one-versus-one (OVO), etc. For instance if the OVO coding is adopted the number of needed binary classifiers is $L = K(K - 1)/2$, where as usual, K is the number of classes. The basic ingredients of the ECOC method are:

- a coding design matrix M

- a binary loss function $g(\cdot, \cdot)$

The coding design matrix is a matrix whose elements direct which classes are trained by each binary learner, that is, how the multiclass problem is reduced to a series of binary problems. Each row of the coding design matrix corresponds to a distinct class, and each column corresponds to a binary learner. In a ternary coding design, for a particular column (or binary learner):

- a row containing 1 directs the binary learner to group all observations in the corresponding class into a positive class.
- A row containing -1 directs the binary learner to group all observations in the corresponding class into a negative class.
- A row containing 0 directs the binary learner to ignore all observations in the corresponding class.

For instance, suppose that, as is the case in this paper, that the goal is to classify data into 3 classes, namely Q, S and P. Choosing the one-versus-one (OVO) coding, the coding matrix assumes the form shown in Table 9.

In this ECOC model the Learner $L1$ trains on observations in Class Q or Class S, and treats Class Q as the positive class and Class S as the negative class. The other learners are trained similarly.

In general, indicating as:

- M the coding design matrix whose elements are $m_{k,l}$,
- s_l be the predicted classification score for the positive class of learner l
- $g(\cdot, \cdot)$ a binary loss is function
- L the number of learners

the algorithm assigns a new observation to the class \hat{k} that minimizes the aggregation of the losses for the L binary learners. Different expressions can be used to compute the predicted class for the observation. For instance, in the *loss-based decoding*, the class producing the minimum sum of the binary losses over binary classifiers determines the predicted class of an observation, as expressed by Eq. 14

$$\hat{k} = \arg \min_{k=1, \dots, K} \sum_{l=1}^L |m_{k,l}|g(m_{k,l}, s_l) \tag{14}$$

Another possible choice is the *loss-weighted decoding*, where the class producing the minimum average of

Table 9 OVO coding matrix to classify into 3 classes

Class	L1	L2	L3
Q	1	1	0
S	-1	0	1
P	0	-1	-1

the binary losses, over binary classifiers, determines the predicted class of an observation, that is Eq. 15.

$$\hat{k} = \arg \min_{k=1, \dots, K} \frac{\sum_{l=1}^L |m_{k,l}| g(m_{k,l}, s_l)}{\sum_{l=1}^L |m_{k,l}|} \quad (15)$$

As concerning the loss function $g(\cdot, \cdot)$ a number of possible choices are possible, such as the Hamming function, the exponential function, etc. Here, just to give an idea, we report the Hamming Loss function (16).

$$g(y_j, s_j) = 1 - \frac{\text{sign}(y_j s_j)}{2} \quad (16)$$

where, $y_j \in \{-1, 1, 0\}$ is a class label for a particular binary learner and as s_j is the score for observation j . In the Matlab Statistical and Machine Learning Toolbox the ECOC model is implemented by the function *fitcecoc*, which allows several parameters to be chosen. In this application we have considered all default parameters which use $K(K - 1)/2$ binary Support Vector Machine (SVM) models together with the one-versus-one coding system.

A.1.3 Ensemble methods

Under the name Ensemble Methods, here referred as ENSE, a wide range of algorithms can be recognized, based on the idea that the use of multiple learning algorithms allows to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. This idea can be formalized as follows. Consider for example a set of k regression models. Suppose that each model makes an error ϵ_i on each example, with the errors drawn from a zero-mean multivariate normal distribution with variances $E[\epsilon_i^2]$ and covariance $c = E[\epsilon_i \epsilon_j]$. Then it is possible to demonstrate (Goodfellow et al. 2016) that the expected error of the ensemble predictor is

$$E \left[\left(\frac{1}{k} \sum_i \epsilon_i \right)^2 \right] = \frac{1}{k} v + \frac{k-1}{k} c \quad (17)$$

This means that in the case where the errors are perfectly correlated and $c = v$, the mean squared error reduces to v , so the model averaging does not help at all. In the case where the errors are perfectly uncorrelated and $c = 0$, the expected squared error of the ensemble is only $\frac{1}{k}$. This means that the expected squared error of the ensemble decreases linearly with the ensemble size. In other words, on average, the ensemble will perform at least as well as any of its members, and if the members make independent errors, the ensemble will perform significantly better than its members. Different approaches are available to build the ensemble of models. Among the most popular we have the Bagging and the Boosting approaches.

In the framework of the Matlab Statistical and Machine Learning Toolbox, the Ensemble method is implemented as

the function *fitcensemble* that has been used in this paper. In particular we have adopted as basic learner the *Adaboost2* algorithm building a medium coarse tree with a maximum number of splits set to 20, a maximum number of cycles up to 30 and a learning rate set to 0.1.

A.1.4 K-Nearest Neighbor K classifier

In classification problems a key point is that of estimating the posterior probability $\hat{p}(y|x)$, $x \in R^p$ and $y \in 1, \dots, n_c$, i.e. the probability of having the class $k \in [1, \dots, n_c]$, after observing the feature vector x . To perform this estimation, let assume the following.

Suppose that we have a training data set $T = \{(x_i, y_i)\}$, $i = 1, \dots, N$, $y \in \{1, \dots, n_c\}$ and let N_k be the number of points x that belongs to the class k . Obviously, we must have $\sum_{k=1}^{n_c} N_k = N$. If we wish to classify a new point x , we draw an hypersphere centered at x containing precisely K points irrespective of their class. Here, we are indicating for historic reasons as K this number which might be confused with the number of classes we want to discriminate, but the reader will understand the true meaning by the context.

Let us indicate as κ the number of points inside the sphere belonging to the class k . It is trivial to understand that:

- the conditional density associated with each class $y = k$ can be written as

$$p(x|y = k) = \frac{\kappa}{N_k V} \quad (18)$$

- the unconditional density $p(x)$ can be written as

$$p(x) = \frac{K}{NV} \quad (19)$$

- the priority for each class can be written as

$$p(y = k) = \frac{N_k}{N} \quad (20)$$

In order to classify a new observation x , we can introduce the following estimation function

$$\hat{y}(x) = k, k = \arg \max_{y=1, \dots, n_c} p(y = k|x) \quad (21)$$

Now, in order to estimate $p(y = k|x)$ we can use the Bayes' rule

$$p(y = k|x) = \frac{p(x|y = k)p(y = k)}{p(x)} \quad (22)$$

Substituting expressions (18),(19) and (20) into (22), we obtain the following estimation function

$$\hat{y}(x) = k, k = \arg \max_{y=1, \dots, n_c} p(y = k|x) = \kappa \quad (23)$$

The interpretation of Eq. 23 is simple: a new observation vector x will be assigned to the class corresponding to the samples that are in greater number in its neighborhood.

It is trivial to understand that the parameter κ controls the trade-off between the bias and the variance.

The K-Nearest Neighbor classifier is implemented in the Matlab Toolbox under the function *fitcknn* that we have used in this paper, setting the dimension of the neighbor to 10 and the Euclidean distance as metrics.

A.1.5 Naive-Bayes classifier

Naive Bayes classifiers, here referred to as NBYE, are probabilistic classifiers based on the Bayes theorem, with a strong (naive) independence assumption between the features. In more detail, indicating as X the features matrix, and as Y the output class, the generic feature $x \in X$ is assigned to one of K class label $y \in Y$ such that $\hat{y} = \arg \max_y p(y|x)$, where $p(y|x)$ is the class posterior probability density, which is computed by applying the Bayes rule. The assumption of independence among features greatly simplifies the computation of $p(y|x)$. Naive Bayes classifiers are implemented in the Matlab Toolbox under the function *fitcnb* that we have used in this paper choosing a normal kernel distribution and unbounded support.

A.1.6 Decision learning tree

Decision Tree (DTs) algorithms, here referred as TREE, originally introduced by Breiman et al. (1984), belongs to the non-parametric supervised learning methods, used for both classification and regression problems. They are based on the use of a tree-like model of decisions which mimic the human level thinking so, in simple cases, they can be easily interpreted. The key idea underlying tree-based methods is that of partitioning the feature space into a set of cuboids, and then fit a simple model (like a constant) in each one. A DT consists of a tree in which each internal (i.e. non-leaf) node is labeled with an input feature. Therefore, a basic problem during the building of the tree is the choice of the feature to be associated to a splitting node. This problem is basically solved by invoking the concept of *impurity*. Several definition of this quantity are available, such as:

- the Gini's impurity or Gini's Diversity Index (gdi), defined as

$$gdi = 1 - \sum_i p^2(i) \tag{24}$$

where the sum is over the classes i at the node, and $p(i)$ is the observed fraction of classes with class i that reach the node. A node with just one class (a pure node) has a Gini index 0; otherwise the Gini index is positive. So the Gini index is a measure of node impurity.

- The Deviance, defined as

$$D = \sum_i p(i) \log_2 p(i) \tag{25}$$

where $p(i)$ is defined as for the Gini index.

Decision trees are implemented in the Matlab Statistical and Machine Learning Toolbox under the function *fitctree*, that we have considered in this paper by using the Gini's index as split criterion.

A.2. Unsupervised classification

Among the numerous unsupervised classification algorithms, in this work we have taken what are probably the simplest and most popular: K-means, Fuzzy C-Means (FCM) and Gaussian Mixture models (GMM).

A.2.1 K-means

The K-means is the prototype of distance based clustering algorithms. After indicating the number K of classes the user intends to classify, the algorithm searches for the cluster centers which best represent the classes, by optimizing an appropriate cost function. Formally, if we have a data set $\{x_1, \dots, x_N\}$ consisting of N observations of a random D -dimensional Euclidean variable x , let us indicate as $\mu_k, k = 1, \dots, K$ a set of *prototypes*, i.e. candidates as cluster centers, associated with the classes. Then, the following cost function is defined

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2 \tag{26}$$

which represents the sum of the squares of the distances of each data point to its assigned prototype vector μ_k . In expression (26), $r_{nk} \in \{0, 1, \}$ is a binary indicator variable describing which of the K clusters the data point x_n is assigned to, so that if data point x_n is assigned to cluster k then $r_{nk} = 1$, and $r_{nj} = 0$ for $j \neq k$.

With these definitions, the clustering problem is to find values $\{r_{nk}\}$ and $\{\mu_k\}$ which minimize J . This optimization problem can be solved with an iterative procedure in which each iteration involves two successive steps corresponding to successive optimizations with respect to the r_{nk} and the μ_k . In the first step J is minimized with respect to the r_{nk} , keeping the μ_k fixed. In the second phase J is minimized with respect to the μ_k , keeping r_{nk} fixed. These two steps of updating r_{nk} and updating μ_k correspond respectively to the E (expectation) and M (maximization) steps of the known EM algorithm that we encounter also in the Gaussian Mixture algorithm. Here, it must be stressed that the Euclidean distance in expression (26), which is usually adopted, for special purposes can be replaced. A

list of allowed distance definitions is reported in (Xu and Tian 2015). For high dimensional dataset others definitions may be adopted, such as the L_1 distance (also known as *cityblock*), the *cosine* distance, i.e. one minus the cosine of the included angle between points (treated as vectors), the *correlation*, i.e. one minus the sample correlation between points, and so on. In the Matlab toolbox the K-means is implemented under the function *kmeans* which was considered in this paper with the default parameters, in particular by using the squared Euclidean distance metric.

A.2.2 Fuzzy c-means

The Fuzzy c-means (FCM) (Bezdec 1981) can be considered an extension of the K-means, but while in this latter algorithm each point belongs exclusively to one class, the FCM allows each data point to belong to multiple clusters, with varying degrees of membership. In more detail the FCM algorithm is based on the minimization of the following cost function:

$$J = \sum_{i=1}^D \sum_{j=1}^N \mu_{ij}^m \|x_i - c_j\|^2 \quad (27)$$

where:

- D is the number of data points.
- N is the number of clusters.
- m is the fuzzy partition matrix exponent for controlling the degree of fuzzy overlap, with $m > 1$. Fuzzy overlap refers to how fuzzy the boundaries between clusters are, that is the number of data points that have significant membership in more than one cluster.
- x_i is the i th data point.
- c_j is the center of the j th cluster.
- μ_{ij} is the degree of membership of x_i in the j th cluster. For a given data point, x_i , the sum of the membership values for all clusters is one.

The FCM algorithm performs the following steps:

1. Randomly initialize the cluster membership values, μ_{ij} .
2. Calculate the cluster centers:

$$c_j = \frac{\sum_{i=1}^D \mu_{ij}^m x_i}{\sum_{i=1}^D \mu_{ij}^m} \quad (28)$$

3. update μ_{ij} according with the following expression

$$\mu_{ij} = \frac{1}{\sum_{k=1}^N \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \quad (29)$$

4. Calculate the objective function, J_m .
5. Repeat steps 2 to 4 until J_m improves by less than a specified minimum threshold or until after a specified maximum number of iterations.

The user can choose a fuzzy partition matrix exponent, indicated as m , being $m > 1$, for controlling the degree of fuzzy overlap.

In the Matlab toolbox the FCM algorithm is implemented under the function *fcm* and considered in this paper assuming the default parameter, which means that the fuzzy partition matrix exponent is set to $m = 2$ and the maximum number of iterations set to 100.

A.2.3 Gaussian mixture models

Gaussian mixture models are probabilistic clustering algorithms which can be useful for fitting multidimensional dataset by a superposition of K Gaussian distributions, as shown in expression (30)

$$p(x) = \sum_{k=1}^K \pi_k N(x|\mu_k, \Sigma_k) \quad (30)$$

where π_k are the so-called mixing coefficients, K the number of clusters the user would like to separate, $N(x|\mu_k, \Sigma_k)$, normal gaussian distribution with mean μ_k and covariance Σ_k . Obviously, in a multidimensional space μ_k are vectors and Σ_k matrices. The mixing coefficients are themselves probabilities and must meet the condition: $\sum_{k=1}^K \pi_k = 1$.

One of the main advantages of using Gaussian mixtures is that it is possible to determine the shape of the distribution depending on the data covariance. Various options for the covariance matrix structures, represent the degree of freedom for the user. A Gaussian Mixture clustering model makes use of the iterative Expectation-Maximization (EM) algorithm (Xu and Jordan 1996). In the Matlab Toolbox, the training of a GMM model can be performed by using the *fitgmdist* function by which allows several parameters to be set, including other than the class number K , also the kind of assumptions concerning the covariance matrices (full or diagonal), the maximum number of iterations, the regularization value. In this paper we have set the Covariance Type to full, the Shared Covariance option to True and the Regularization Value to 0.01.

Acknowledgements We are indebted to the Authors of the paper (Cannavó et al. 2017) for enabling data used in this paper. This work was supported by the Università degli Studi di Catania, under the project Piano della Ricerca 2020-2022.

Funding Open access funding provided by Università degli Studi di Catania within the CRUI-CARE Agreement.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate

if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Bano S, Khan MNA (2018) A survey of data clustering methods. *Int J Adv Sc Technol* 113:133–142. <https://doi.org/10.14257/ijast.2018.113.14>
- Bergen KJ, Johnson PA, de Hoop MV, Beroza GC (2019) Machine learning for datadriven discovery in solid Earth geoscience. *Science* 363:1–10. <https://doi.org/10.1126/science.aau0323>
- Bezdec JC (1981) Algorithms, pattern recognition with fuzzy objective function. Plenum Press, New York, pp 1–272. <https://doi.org/10.1007/978-1-4757-0450-1>
- Bishop CM (2006) Pattern recognition and machine learning. Springer, Berlin, pp 1–758
- Breiman L, Friedman J, Olshen R, Stone C (1984) 11 Classification and regression trees. CRC Press, Boca Raton, pp 246–280
- Cannavó F, Cannata A, Cassisi C, Di Grazia G, Montalto P, Prestifilippo M et al (2017) A multivariate probabilistic graphical model for real-time volcano monitoring on Mount Etna. *J Geophys Res Solid Earth* 122:3480–3496. <https://doi.org/10.1002/2016JB013512>
- Chouet B (1996) Long-period volcano seismicity: its source and use in eruption forecasting. *Nature* 380(6572):309–316. <https://doi.org/10.1038/380309a0>
- Dietterich TG, Bakiri G (1995) Solving multiclass learning problems via error-correcting output codes. *J Artif Intell Res* 2:263–286
- Fisher RA (2008) The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, Chapman - Hall/CRC. Taylor - Francis Group, LLC vol. 7, pp 179–188
- Goodfellow I, Bengio Y, Courville A (2016) Deep Learning the Mit Press
- Hajian A, Cannavó F, Greco F, Nunnari G (2019) Classification of mount etna (italy) volcanic activity by machine learning approaches. *Annals of Geophysics* 62(2, VO231):1–11. <https://doi.org/10.4401/ag-8049>
- Hastie T, Tibshirani R, Friedman J (2008) The Elements of statistical learnings - data mining, inference, and prediction. Springer, Berlin, pp 1–764
- Inc TheMathWorks (2017) Statistical and machine learning toolbox user guide, R2017b, The MathWorks Inc 1–2070
- Kong Q, Trugman TD, Ross ZE, Bianco MJ, Meade J, Gerstoft P (2019) Machine Learning in seismology: turning data into insights. *Seismology Res Lett* 90(1):3–14
- Langer H, Falsaperla S, Hammer C (2020) Advantages and pitfalls of pattern recognition - selected cases in geophysics. Elsevier, Amsterdam, pp 1–331
- Langer H, Falsaperla S, Messina A, Spampinato S, Behncke B (2011) Detecting imminent eruptive activity at Mt Etna, Italy, in 2007–2008 through pattern classification of volcanic tremor data. *J Volcanol Geotherm Res* 200:1–17
- Lei B, Xu G, Feng M, Zou Y, Van der Heiden F, De Ridder D, Tax DMJ (2017) Classification, parameter estimation and state estimation - an engineering approach using MATLAB Wiley
- Liao TW (2005) Clustering of time series data - a survey. *Pattern Recogn* 38:1857–1874
- Liu H, Motoda H (2008) Less is more, (in Computational Methods of Feature selection, Liu and Motoda Eds, chap 1), Chapman-Hall, CRC Press, 3–15
- Malfante M, Dalla Mura M, Mtaxian JP, Mars JI, Macedo O, Inza A (2018) Machine learning for volcano-seismic signals. *IEEE Signal Process Mag* 35(2):20–30. <https://doi.org/10.1109/MSP.2017.2779166>
- Mohamed AA (2020) An effective dimension reduction algorithm for clustering Arabic text. *Egyptian Inf J* 21:1–5
- Powers DMW (2011) Evaluation: from precision, recall and f-measure to ROC, informedness, markedness and correlatio. *J Mach Learn Technol* 2(1):37–63
- Rouseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J Comput Appl Math* 20(1):53–65
- Spampinato S, Langer H, Messina A, Falsaperla S (2019) Short-term detection of volcanic unrest at Mt. Etna by means of a multi-station warning system, *Scientific Reports*
- Tallon-Ballesteros AJ, Riquelme JC (2014) Data mining methods applied to a digital forensics task for supervised machine learning. In: Computational intelligence in digital forensics-forensic investigation and applications, pp 413–451. <https://doi.org/10.1007/978-3-319-05885-6-17>
- Xu L, Jordan MI (1996) On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Comput* 8(1):129–151
- Xu D, Tian D (2015) A comprehensive survey of clustering algorithms. *Ann Data Sci* 2(2):165–193. <https://doi.org/10.1007/s40745-015-0040-1>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.