UNIVERSITÀ DEGLI STUDI DI CATANIA

Master's Degree in Data Science
Department of Mathematics and Computer Science (LM-Data)

Multiparametric Analysis in Volcano's Environment

Elaborato finale

Author:
Gaetano De Angelis

Supervisor:
Prof. Sebastiano Battiato
INGV Co-Supervisor:
Dr. Salerno Giuseppe
Dr. Reitano Danilo

Academic Year 2024–2025

Contents

Chapter 1

Introduction

The importance of the data nowadays is becoming more relevant day by day. Today, the impact of data and technology in general is fundamental for most of activities of the world. This aspect is truer in a crucial,delicate and fundamental sector as the vulcanologist one; using data of a previous event to try to predict new ones and to alert the population in advance could save a huge quantity of lives. In the Earth domain Science, Volcanic systems represent some of the most complex and dynamic natural environments on Earth; in particular Mt.Etna is the biggest Europe's Volcano and one of the most active of the world.

Until recent times, Etna was considered a predominantly effusive volcano, that is, mainly characterized by the emission of lava flows. These can cause material damage but do not represent a direct threat to the lives of the 900,000 people living in potentially at-risk areas. However, recent studies have revealed that this volcano is also capable of producing highly explosive activity, such as the Plinian eruption of 122 B.C.

The original dataset on which the thesis relies on, is a dataset containing several data collected by INGV using sensors on different position (during the study we've implemented a method to localize the sensor from which comes the value) above the volcano Etna. The data are related on different aspects of a volcano's behavior, such as heat flux, Clinometry, SO2-Flux (later we'll see the meaning of these variables) and so on. The original dataset is a csv.file, containing several dataset each of which describes and contains value about the variable (date, and value of the variable for example Depth and magnitude for volcanic tremor).

The aim of the research is to manage, collect and clean the data in a first

moment, just to see the distribution of the data, the pattern and the typical behaviors. I've plotted several graphical representations to show the trend of the variables, such as heat flux and so on, and see if there are any outliers or anomalous behaviors. After several graphical visualizations, according with the INGV's experts, we've decided to implement some techniques to capture some correlation between variables (clustering to divide variables base on magnitude and depth of earthquake, and correlation matrix to analyze and visualize variables relationship). Later on, I have trained several models pointing at different aims; first of all I have tried with random forest algorithm to analyze again and better, correlation within features and then I have train LSTM-models in order to predict earthquakes based on volcanic tremor data.

The thesis is an elaborated work done, from one side, using INGV'S expert data and expertise.INGV analyzes and manages data collected on Mt. Etna in order to understand its geological and geophysical behavior.The aim of the thesis is to develop a case study, able to predict future events such as eruptions or earthquake. The thesis represents an innovative study even for INGV, for the innovative models and algorithm used during the thesis, trying to visualize, under a new and different point of view, some of the trillions of data present in their domain.

Chapter 2

Introduction of Machine and Deep Learning techniques

2.1 Short story of Machine Learning AI

The first study approach or citation to the automated learning in general was implemented from Alan Turing on his work "On computable numbers". But in this case, he only starts to reflect about the need to focus on machines and algorithms which can learn. For this reason, there is not only one or a specific one father of this science; Arthur Samuel was another important pioneer who has for the first time implemented a program able to learn (in this case was able to learn how to play checkers). And later on, there are other important researchers who could be considered as pioneer of deep learning and IA, later one, pointing the discussion on a highest plane, such as Geffrey Hinton (pioneer of deep neural network) or Yann Le Cunn(pioneer of CNN).

2.2 Concept of Machine Learning?

Let's take a moment to let our imagination run free and consider what a world would look like if every process related to productivity and beyond were performed by automated artificial brains or machines. This is essentially what we are talking about: Most of the most powerful techniques that are currently at the center of discussion are based on concepts such as machine learning and related technologies.

To understand easily the idea under machine learning we must think

about special algorithm or systems which does not necessarily need a fix structure of rules to implement their tasks. The concept is basically this; machine learning is a branch of AI composed of algorithms able to use the data to improve automatically their capabilities and performance without relying on any instruction or rules. And from this we can understand the importance, powerful and meaning of the data; just using and reading the data in a correct way machine learning algorithms are able to take companies and life efficiency to another level.

First of all, I have to point out an important distinction on three different approaches of machine learning:

1. **Supervised Learning** This is a learning approach using labeled data. So, the algorithm learns from these labels, some target given to the "machine" to let it learn which is the final goal. The performance of the algorithm is evaluated by comparing the prediction with the target one. It is, to make it simpler, such as the algorithm could try and try again in a certain way to maximize it results and let it be nearest to the target. One of the most popular supervised techniques is the classification, in which the algorithm assigns the input data to a target class/label (for example algorithm designed to classify whether event is an earthquake or not).

2. **Unsupervised Learning** Now it should be easier to understand unsupervised learning; this approach does not rely on any labels or target class to the algorithm. Instead, the algorithm analyzes the data and identifies relevant patterns or structures on its own. One common technique of this type is the clustering, where the algorithm divides data in different "clusters" (group), grouping together data points that share common features (for example, one cluster could contain deep earthquakes,and the second one shallow earthquakes, and so on).

3. **Reinforcement Learning** Now let's briefly introduce Reinforcement Learning. In this approach, the algorithm, often called an agent, learns by interacting with an environment rather than relying on labeled data. The agent takes actions and receives feedback in the form of rewards or penalties, which guide it toward achieving a certain goal. Over time, by trying different strategies and learning from the outcomes, the agent improves its behavior to maximize cumulative rewards. An example in our case could be a robot that needs to predict whether an eruption

will occur or not. We provide it with rewards or feedback each time it correctly identifies a potential eruption.

2.3 Process of Machine Learning techniques

Let's try now to understand something else about the typical structure of machine learning process techniques and algorithms, how they work and how works the managing, training and building of these algorithms. The typical process through which a machine learning model works, or more in general a data scientist works, is based on several and distinct phases. In a machine learning process, the dataset is typically split into training, validation, and test sets. The training set is used to teach the model, the validation set to fine-tune and optimize it, and the test set to evaluate its final performance on unseen data.

- Collecting data and pre-processing: the first thing every data scientist should do or from which every machine learning process start is the collection of the data. Sadly, machine learning is not such as a magic box that let be possibly create something amazing from nowhere. The first and most important thing to let be these algorithm useful is to understand the data; is fundamental to know which are talking about the data, in which domain and how they're organized. For this reason, usually, the figure of data scientist is paired and cooperated by the figure of expert's domain, who covers a primary role to lead and support data scientists. In our case the role played by INGV experts was fundamental to understand the meaning of the data, the patterns of them and the behavior of Etna in general, in a way to know how manage data and which technique or model to implement.

  In every case data need to be processed and cleaned, to be prepared for the training of the model. In most of the cases there are some issues on the data distribution, such as missing values, different scales of values or not well organized. The second phase, after has understood the real meaning of the data, is to manage them to remove issues and let be the data prepared for the training. Some examples of pre-processing process are: data cleaning (remove duplicate rows, remove missing values), feature engineering

(selecting most relevant features or creating new ones), managing unbalanced data (e.g. SMOTE technique consists to oversample minority classes), dimensionality reduction (PCA reduces the number of features while retaining the most relevant ones).

- Training of the model: Once we've prepared and managed our data, is time to train the model. The model in this phase understand pattern and typical behaviors of the data, and adjust its own parameters or weights(for neural network) to improve its performance.There is a clear distinction between parameters and Hyper-parameters. Parameters are values learned automatically from the data during training, such as the weights in a neural network or coefficients in a regression model. Hyper-parameters, on the other hand, are set by the data scientist or algorithm designer before training begins, such as the learning rate, number of layers, or regularization strength. In this sense, Hyper-parameters are aspects of the model that the designer can adjust to optimize performance. The division of which quantity of data relies to each set depends on the dataset, but usually is 80 percent on training set, 10 percent on validation set and 10 percent on test set.
- Validation set: The validation set allows monitoring intermediate results and guiding the optimization of model performance. We've talked before about the division of the data, but if we've too few data? In this cases it's usually applied cross-validation; dataset is divided in k (chooses by algorithm designer) folds, the model is trained k times using k-1 folds for the training and one fold for the validation set; at the end the final result is given by the average of the result of each fold.

  Which are the most popular technique to make fine-tuning using validation set?

  - Hyperparameter tuning: It's the process of selecting the optimal values for a model's hyper-parameters, which are parameters set before the training process and cannot be learned directly from the data. The validation set is typically used to evaluate different hyperparameter configurations, employing techniques such as grid search (we fix a set of parameters, try all the available combinations and choose the best one), random search (method choose randomly a set of parameters), or

Bayesian optimization (a probabilistic model to estimate the parameters with more probability to improve performances) to identify the combination that maximizes performance while avoiding overfitting.

- Early Stopping: Is a popular technique used to stop training of the model whenever results stop to improve; in a certain way is an automatic system which interrupts, after some epoch (a complete pass over the training data), training of the model if it doesn't improve the performance. This is used to avoid overfitting, the situation in which model fit too good train data and isn't able to generalize on unseen data, test set.
- Model selection: Based on the results obtained in the validation set, is necessary to choose the best model which had obtained best results. The goal is to choose the best model in terms of generalization, accuracy and complexity
- Regularization techniques: Usually overfitting is caused by complexity of the model; a solution to this kind of issue is named regularization. These methods aim to introduce a penalty to reduce impact of large weights or parameters. Three regularization techniques:
  (a) L1-regularization: Also called Manhattan distance, it pushes several parameters to became zero, permitting the model focus only on relevant ones.
  (b) L2-regularization: Or Euclidean distance, pushes to became smaller large parameters without loosing them or cancel them
  (c) Dropout: Technique used for deep neural networks deactivate randomly some neurons, forcing the model to learn more robust representations.

- Test set and visualization of results: Final phase in which is showed the ability of the model to work and unseen data, here is evaluated the generalization capacity of the model.It permit to understand strenght and weakness points of the model and give a full and completed visualization of the model results. There are several metrics used to evaluate performance of the models:

(a) Classification metrics:

- Precision: It counts how much observation predicted as positives from the model are real positive. There are:
  - True positive: cases that are positives and are correctly predicted as positives by the model (e.g. earthquake really happened that are predicted as happened)
  - False positive: cases that are negatives but that the system has predicted as positives (earthquake not happened but predicted as happened by the model)
  - True negative: negative cases predicted correctly as negatives by the system (earthquake not happened predicted as not happened by the model)
  - False negative: positive cases predicted as negative by the model (earthquake happened predicted as not happened by the model)

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.1}$$

- Recall: Number of positive observations the model correctly individuated by the model. The goal is to minimizes false positives.
- Accuracy: Number of total correct prediction of the model, When the original dataset is a lot unbalanced, accuracy could provides issues and errors, in these case is preferred to use F1-score or recall or precision. To better understand if we have to predict when if a tweet is about earthquake or not, and we have 10000 not about earthquake and 100 about it, the precision could be higher and the algorithm could predict all the tweet as "not about earthquake", but we want to focus our attention on the "about earthquake" cases and so could be more useful other metrics such as precision or recall.
- F1-Score: It combines precision and recall; variate between 0 and 1 where 1 is the best possible combination between accuracy and precision.

- Confusion Matrix: Is a graphical representation, classification model, which show how the model predicts each class. Is a table which the rows represents real classes and columns represents predicted classes.

- ROC and AUC curve: The ROC curve is a model able to evaluate the capacity of a model to distinguish between classes, it put on the x-axis (FPR) false positives rate and on y-axis true-positive rate (TPR). AUC (Area under the curve) instead represents the area under the ROC curve and represents a valid index to evaluate performance of the model to distinguish between class, if the value is 0.5 the model classification casually, if the value is 1 the model classify perfectly.

(b) Regression metrics

- MSE(mean square error): It measures the difference between the real values and the predicted values, this difference is squared to penalize big errors and to be more sensitive to the outliers.

- RMSE(root mean square error): to show the error on the same scale of the original date, as the MSE it penalizes big errors.

- MAE(mean absolute error): A linear measure of the mean error, without any square. So it provides the difference between the prediction and the real values

- R^2 score: It's a statistic measure used to explain how well a model explain variability of the data; if it is 1 it means that the model explain perfectly data, instead a 0 value it means that this index is not better of a simple mean.

- Final prediction: Conclusive phase of every classical machine learning process and not only, is the prediction. If we're talking about a classification problem the output will be the assignment of a value to a class, instead if is a regression problem it will provide a continuous numeric value

2.4 Definition and explanation of Deep Learning basics

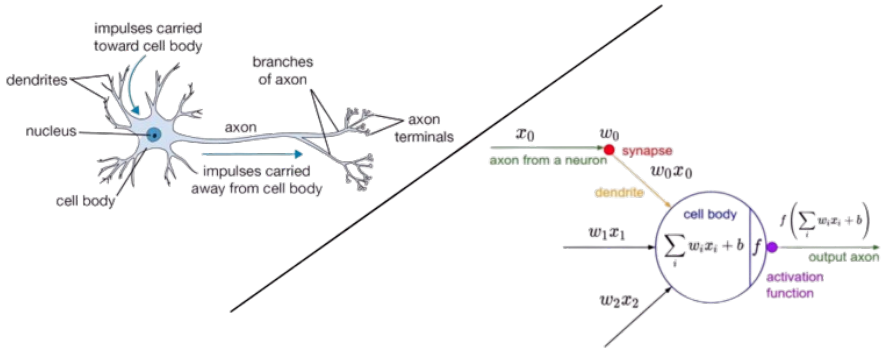

Figure 2.1: Parallelism between neural networks and human neurons

The necessity and the capacity to extract relevant aspect by the data is bring to another level; deep learning is a branch of ML (machine learning) which focus the speech on the concept of deep neural networks. These are complex structures composed by several level, inspired by the structure of the human neurons. The structure of a neural network is composed by input, hidden and output layers; each of them are full of neurons artificial neurons which remember, as we say before, the structure of the real human neurons. In an artificial neuron, the equivalent of the human neuron's dendrites are the input variables multiplied by their corresponding weights, i.e.

$$z = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n. \tag{2.2}$$

The weighted inputs are then summed together and adjusted by a bias term b, representing the role of the cell body:

$$z = \sum_{i=1}^{n} x_i w_i + b. \tag{2.3}$$

Finally, the activation function $\varphi(z)$ is applied, which corresponds to the axon transmitting the signal:

$$y = \varphi(z).$$

There are two phase to explain how the neural network works k:

- Feedforward pass: Consists in propagating the inputs through the network to obtain the output.Basically is the initial step in which the input signals are propagated forward through all layers of the network to compute the final outputs.

  To train the network, we use a "loss function" $\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y})$, which measures the difference between the predicted output $\hat{\mathbf{y}}$ and the true output $\mathbf{y}$. Common choices are the "mean squared error" (MSE) for regression:

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{m} \sum_{j=1}^{m} (\hat{y}_j - y_j)^2, \tag{2.4}$$

  or the "cross-entropy loss" for classification:

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{m} \sum_{j=1}^{m} \sum_{k=1}^{K} y_{jk} \log(\hat{y}_{jk}), \tag{2.5}$$

  where m is the number of examples in the batch and K is the number of classes.

- Backward pass: The network parameters are updated using *backpropagation*, which applies the chain rule to compute the gradient of the loss with respect to each weight.

  For a single weight w_i, the derivative of the loss is obtained as:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial z} \cdot \frac{\partial z}{\partial w_i}, \tag{2.6}$$

  where: - $\frac{\partial L}{\partial y}$ measures how the loss changes with respect to the output of the network, - $\frac{\partial y}{\partial z}$ is the derivative of the activation function, - $\frac{\partial z}{\partial w_i}$ is the derivative of the weighted sum with respect to the weight w_i (which corresponds to the input x_i).

  Finally, the weights are updated using a learning rate η:

$$w_i \leftarrow w_i - \eta \frac{\partial L}{\partial w_i}. \tag{2.7}$$

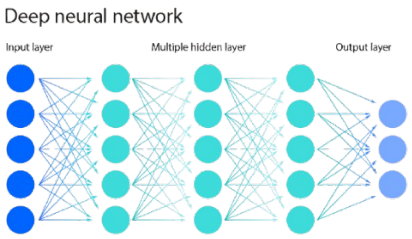

Figure 2.2: A graphical representation of a typical deep neural network structure, composed of input, hidden, and output layers.

These are basically the steps with whom the neural networks unrolls and complete their processes. Basically, to let it be more understandable, the neural networks works starts going forward to reach the output. At the end of the process a loss function is computed to underline the difference between the target output and the predicted one; then is back-propagated the process in reverse mode to adjust the weight to improve the performance and reduce the loss. Is just like trying to reproduce the reverse process to improve the results obtained via feed-forward step.

Chapter 3

Experiment on Etna's dataset

3.1 Some information about INGV

Volcanoes are one of the most fascinating natural features of the Earth. But we know that, as every natural events, volcanic eruptions could be extraordinarily beautiful and extremely dangerous at the same time. For this reason, monitoring and surveillance of active volcanoes is of primary importance, to mitigate the impact of society and to allow for timely and accurate information and alerts. The Istituto Nazionale di Geofisica e Vulcanologia (INGV) is a public research Institution that carried out research and surveillance/monitoring in the fields of earthquakes, volcanoes, tsunamis, and space weather. It manages national monitoring networks, coordinates activities to reduce risk from natural phenomena, provides data and scientific expertise to the National Civil Protection Service, and disseminates information to the public. Specifically, the Osservatorio Etneo of INGV Catania, carried out the monitoring and surveillance of the Sicilian active volcanoes by means networks of multi-parametric stations which transmit collected data in real-time to the observatory (e.g., geodetic, geochemical, seismic). Data from the instrumental infrastructure are also coupled with discrete observation from ground and space. Data are then procecessed enveloped in model for early warning and available for monitoring and surveillance purpose in the 24/7 operational room.

3.2 More on data and Volcano's activities

Records of data used in this thesis spans from June of 2017 to 28 February 2018 and is collected from the multi-parametric networks of INGV Catania on Mt Etna. Below, I report an overall description of the volcano dynamics and detail about the used datasets, and a specific description of the approach carried out to localize the position of the seismic events recorded in this period
.

- Volcano description: Description of the volcano's structure and activities to understand better the nature of the data used in this study. Generally, Volcano consist of: Deep-plumbing system: From here magma is generated; pressure and temperature here are high and so the rocks could melt, this magma contains deep volatile such as e.g., He, C, O2, S, H

- Intermediate plumbing system: magma goes up through earth's crust, during the ascent it could accumulate on magmatic reservoirs; during this process the magma could be contaminated by crustal rocks. As the magma begins to rise toward the surface, the pressure drops, inducing dissolved gases to undergo a phase change, causing them to exsolve and form gas.

- Shallow plumbing system: Here volatiles start to end it's process of decoupling from magma to be independent gas phases. Magma ascent and depending on melt/gas ratio and flow dynamics generate degassing, earthquake, seismic volcanic tremor caused by the turbulent passage of fluids. Vigorous degassing and magma approaching at the top of the crater may be reflected by increase in temperature.

The dataset used in this study, as already mentioned, is in a csv file format and consist of 10 papers, each of them containing seismic, and chemical, thermal and Clinometry data collected at Mt Etna between dd/mm/yy and dd/mm/yy. . Let's see them:

- Data description: The first paper is only categorical variables; it's a simple and descriptive dataset in which the first column is the names of variables, in second is instead a short description explaining of the data itself (e.g. SO2 flux in the first column and sulfur dioxide emissions in second column with measurements unit).

- Earthquake: target dataset of this study is the earthquake catalog of seismic events occurred on Mt. Etna in which most of the models and tries have been tested on. First column of earthquake paper contains the date during which the earthquake has been detected from June of 2017 to 28 February 2018 daily). In the second column there are times of the events, when was detected the origin of the earthquake. then we've the latitude and longitude of earthquakes to get its position; then there are depth of the earthquake and ML which identifies magnitude or in other words power of the earthquake (useful to understand impact of the earthquake).

- Heat flux: It measures flow of thermal energy released by the volcanic summit craters ; this is interesting to be continuously monitored cause in presence of the eruption there is an increasing of the heat flux in the volcano's higher levels, this cause magma goes up. In this paper there are, as usual, in the first column information about date and time of the heat flux's value detection; in the second one there is the value of this variable.

- ^{3}He-^{4}He Ratio: Quantity of Helium into the Volcano's building; ^{3}He comes from the coat, so from the first levels, lowest ones, of the Volcano's building, instead ^{4}He comes from the highest levels. This ratio is useful to understand how much gas comes from deep of the volcano edifice and how much is contaminated from the Earth's crust. An high ratio means that it comes a sort of "fresh" magma instead a low ratio is associated to "old" magmatic systems cause gas are melded with the Earth's cluster .

- CO_2 flux: gas phase which decouple form magma in intermediate plumbing system (7-12 km below sea level); when the value of this gas increases it could reflect deep magmatic recharge. As the previous file, this paper is composed by two columns, first dataset, second one containing gas values; this is the same for all the gas variables.

- CO_2/SO_2: In volcanic monitoring, the ratio CO_2/SO_2 is a key geochemical indicator of magma dynamics. Because CO_2 degasses at greater depths than SO_2, an elevated CO_2/SO_2 ratio commonly signals deep magma recharge or ascent, whereas a declining ratio often reflects shallow degassing and imminent surface eruption. Continuous

measurements of this ratio, combined with seismic and deformation data, therefore provide valuable early-warning information on changes in the volcanic system.

- SO_2 flux: The SO_2 flux is an important sign of volcanic activity at Mount Etna. Since SO_2 tends to come out of the magma when it gets closer to the surface, changes in its flux often give us a clue about what is happening in the shallow plumbing system of the volcano (2 below sea level). For example, a sudden increase in SO_2 flux can mean that magma is rising quickly and an eruption might be near. Looking at SO_2 flux together with CO_2 flux and their ratio helps scientists better understand and follow the changes in Etna's behavior.

- HCI flux: The HCI flux is another gas signal measured at Mount Etna, even if it is usually less abundant than CO_2 or SO_2. Hydrochloric acid gas tends to be released when magma is already very close to the surface, so its presence and variations can give useful hints about ongoing or imminent eruptions (0 km b.s.l). Even small changes in HCl flux, when compared with other gases, can help scientists get a clearer picture of the degassing process and the state of activity of the volcano.

- Clinometry: Clinometry is the measurement of ground tilt of Mount Etna. When magma rises inside the volcano, it can push and deform the volcano edifice producing small but measurable changes in the slope of the ground. Through Instruments, such as tiltmeters, scientists, can detect the deformations caused by the magma. This dataset is composed by several columns:

  - Time: As usual even here first column is about time information, in a single string there are date and time when was detected the signal
  - Ten-batt: This is about the energetic state of the station, to take track of this energetic state; this specifically indicates tension in volt of the battery
  - temp-CR10: A measure useful to be sure that the instruments are stable or could show if there are some thermal effects.

- Tilt-xAvg: Tilt, inclination, of the ground on the X axis (X-axis it means a reference direction, Ovest). Unit is expressed in decimal grades, low values.

- Tilt-yAvg: Inclination on the Y axis (so Ovest in our case)

- Temp-tilt: Temperature of the environment where is the inclinometer (instrument used to evaluate the inclination of the ground). It's useful, cause if the temperature change a lot it could cause a false tilt.

- Nord-tilt: Inclination projected northwards, indicates shift of the Etna's ground considering Nord as reference's point.

- Barometer: Measures atmospheric pressure registered by the sensor; this could filter some atmospheric effects which could affects on the Clinometric measure.

- Volcanic tremor: Important aspect which measure the movement or tremor of the Etna's building; as we say before, this measure is correlated within the other ones such as SO2 flux and so on, so an increasing or a change in this variable detected by the sensor positioned in the volcano's top, could means an ascent of the magma and correlated gas ans so it means imminent eruption. The first column contains date of signal's detection, second one average width of volcanic tremor.

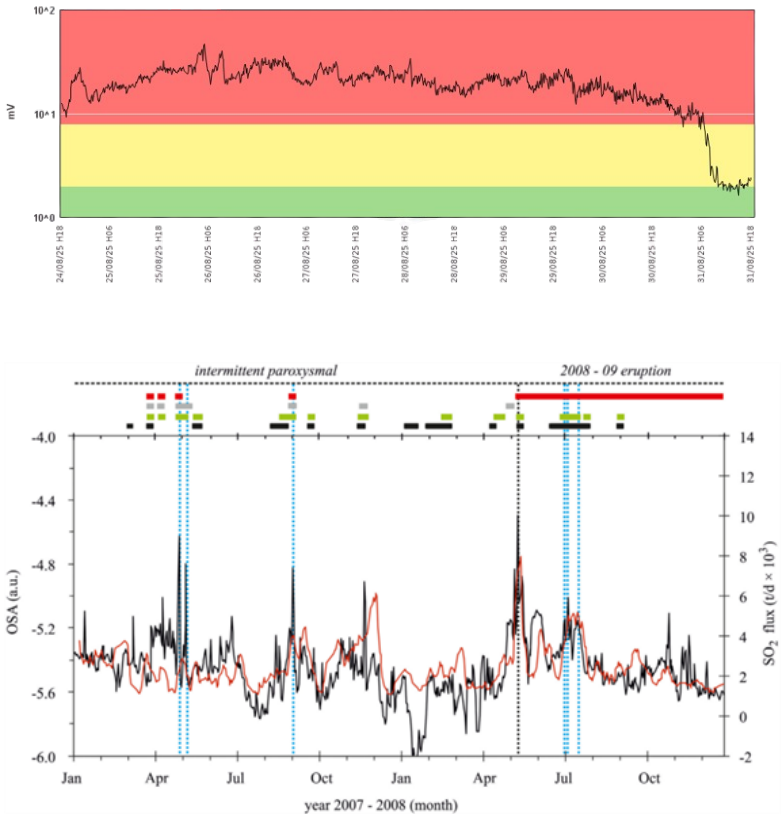

Figure 3.1: Figure 3.1: Weekly averaged SO2 flux (in red) and daily averaged overall spectral amplitude (OSA) of the volcanic tremor (in black) in 2007 and 2008; tremor is represented on a logarithmic. (By Salerno GG, Burton M, Di Grazia G, Caltabiano T and Oppenheimer C. 2018, Coupling Between Magmatic Degassing and Volcanic Tremor in Basaltic Volcanism. Front. Earth Sci. 6:157. doi: 10.3389/feart.2018.00157).

Figure 3.1 shows an example of the multi-parametric approach to explore mechanism of eruptive activity. Specifically, the graph displays the weekly averaged SO2 flux (in red) and daily averaged overall spectral amplitude (OSA) of the volcanic tremor (in black) in 2007 and 2008; tremor is represented on a logarithmic. The black-dashed vertical line between April and July 2008 indicates the start of the 2008-09 eruptive activity on 13 May while the blue-dashed lines indicate the selected case studies explored at intraday scale. The upper graph shows the summary of the eruptive activity of Mt. Etna in the studied period. The 2-year long time window can be split

into two phases according to the eruptive activity observed at the surface. The first phase relates to an intermittent paroxysmal phase consisting of ash emission (black square), Strombolian activity (green-square), lava fountain (gray-square), and lava emission (red-square), the second phase refer to a period characterized by persistent lava emission and irregular moderate Strombolian explosions and ash emissions along the 2008-09 eruptive fissure. Uncertainty in SO2 flux by stationary array arise from several sources, such as wind-plume transport speed, plume height and radiative transfer, these parameters maybe highly variable being the flux affected by and uncertainty that may ranges between –22 and 36, up to 100% in challenging condition. Uncertainty in daily RMS depends on the intraday variability of the seismic averaged signal and it ranges between 10% for the intraday study and mean of 25% for the daily long-term investigation.

Chapter 4

Pre-processing: techniques used on our case

In the original dataset there were several issues such as for example date-time format, outliers, missing values and so on. For this reason i have chosen to clean each variable separately one after other and then to create a merged dataset composed by the new adjusted variables. Cleaning operations for each variable:

- Heat-flux: This variable was the most "relevant" under the cleaning phase point of view; Heat-Flux had several issues and for this reason i have implemented several function and solution to solve them. The main problem of this (and the majority of variables we'll see) is about the separator in the date-time info; the standard Europe format to represent date uses the point, instead in the original dataset each date's number was separated by the comma. So i have replaced the comma with the points in date strings, then i have eliminated every time's duplicate rows and i have managed the outliers, using IQR (inter-quartile range, difference between quartiles) removing extremes outliers out of the IQR. At the end i have saved new results on a new clean file.

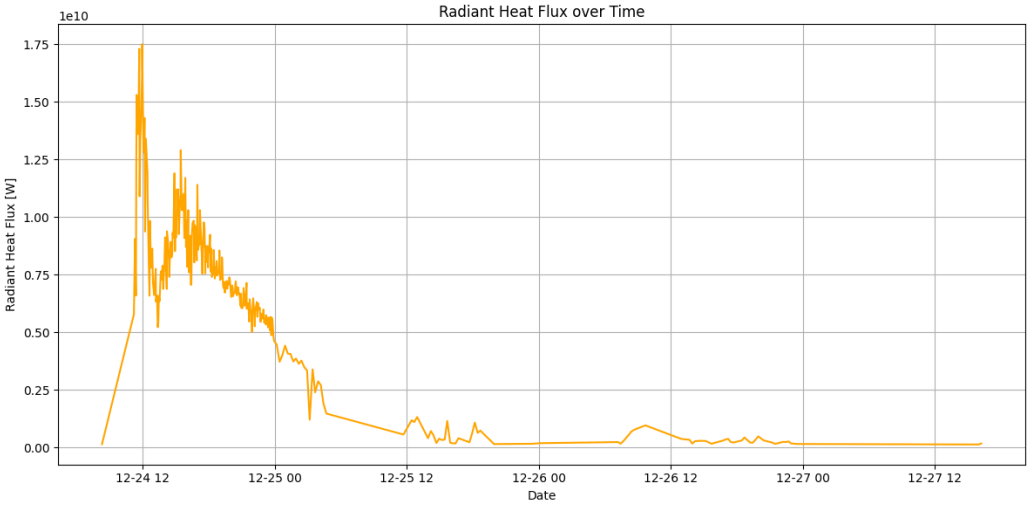

Figure 4.1: Graphical representation of heat-flux trends after cleaning operations.

- SO_2 Flux: The cleaning operations are more or less similar for each variables cause the issues are quite shared. Here the operations for SO_2 flux are similar to the first ones; the date-time correction was made more manual, mapping on the month and not days, so the range was on daily frequencies within day as interval and not on minutes as in the previous case. Cleaning of missing values and outlier was easier without IQR.

- HCI-flux: Here the problem was more consistent, cause there are several outliers and more missing values. I have implemented a more robust cleaning process in order to manage better outliers: identifying them, saving them on a isolated output, then i have cleaned the outliers and plotted a graphical representation to show the result of the cleaned file.

	Date	daily HCl flux (t/d)	outlier
0	2017-01-19	483.00	False
1	2017-01-25	321.00	False
2	2017-01-31	533.00	False
3	2017-02-03	681.00	False
4	2017-02-08	568.00	False

Figure 4.2: Table showing cleaned HCI-flux file, after managing outliers operations

- ^{3}He-^{4}He Ratio : The ^{3}He/^{4}He ratio file was first cleaned to remove inconsistent or missing entries and to prepare it for analysis.As in the previous cases, the cleaning process included skipping unnecessary header rows, removing fully empty lines, replacing missing values with NAN markers, converting dates into a standard date-time format, and ensuring that numerical values used a consistent decimal separator. Duplicate entries were also checked and removed to avoid bias in the analysis. After this step, the data was properly structured and ready for visualization.

- To better understand the variability of the measurements from different stations, a box-plot was used. A box-plot is a statistical visualization that summarizes the distribution of a dataset by showing its median, the range of values within the first and third quartiles (the "box"), and potential outliers (values that are unusually high or low). This type of plot is useful because it highlights not only the central tendency of the data but also the spread and the presence of anomalies, which can be important when monitoring gas emissions and identifying abnormal patterns in the ^{3}He/^{4}He ratio.

- CO_2-SO_2 flux: First of all, trying to open the file, i had skipped the first row because probably useless header, and then it clean the columns name from weird spaces. The main variable here is the “CO2/SO2”, and

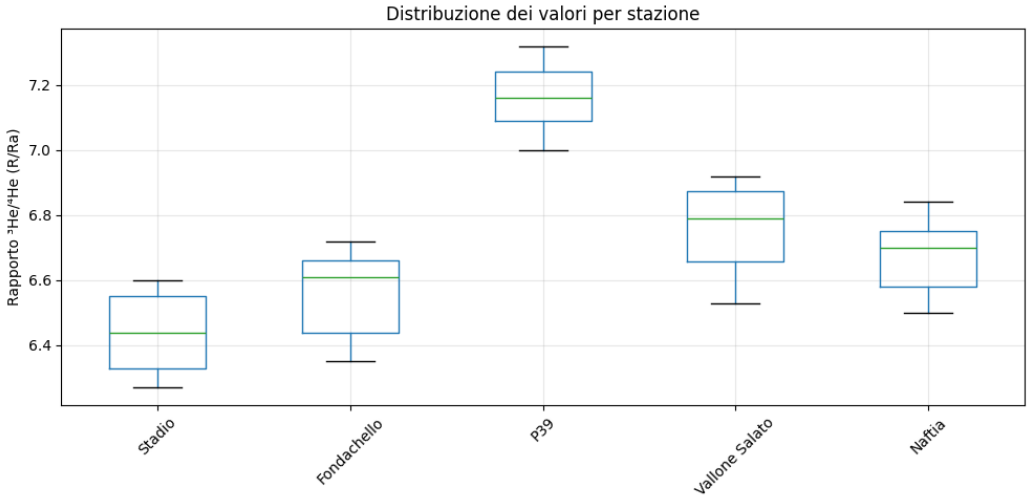

Figure 4.3: Box-plot representation for each value (column of $^3He/^4He$ paper)

it comes in strange format with commas and text "NAN", so the code convert it to real floating numbers. Then the Date is turned into proper date-time so we can sort and filter by it, and rows with no date get kicked out. It also remove duplicates and mark as outlier if ratio is more than 20, because probably that is unrealistic high. The script extract Year, Month, Day from the date for later groupings, sort all by time and save it again as cleaned CSV. In the end it print some basic statistics, like mean value around 5.19 with deviation 5.43, total 498 rows and only 14 are outliers.

- CO_2 flux: The main variable here is "f CO2_norm", that is the normalized CO2 flux values. These were in text with comma as decimal, so we change it to dot and convert to number. Dates were also converted to proper date-time, and any row with missing values was removed. After that the clean data was saved again as CSV. The final dataset have 761 daily measurements, mean flux is around 0.42 with standard deviation 0.13, most values are between 0.35 and 0.48, minimum is 0.00 and maximum reach 0.94.

- Clinometry: As always i have split the original single-column data into proper columns using the comma separator, then it set the first line as header and removed it from the data body. The time column was

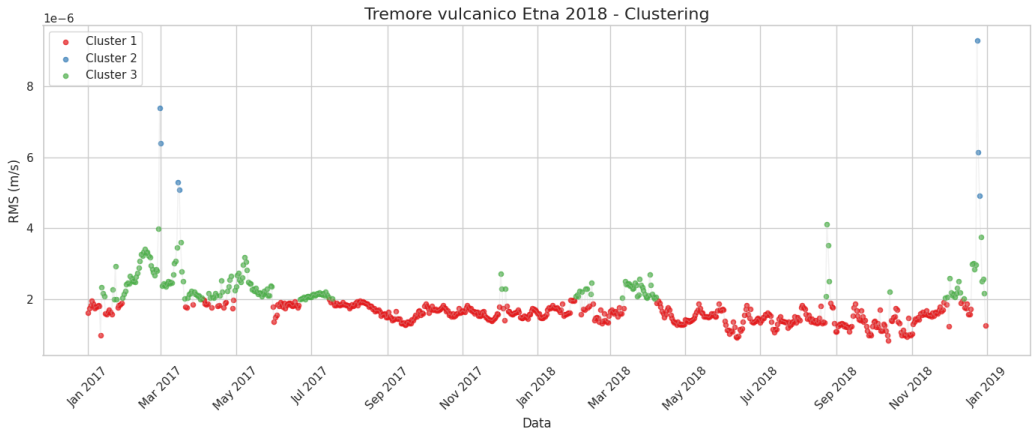

Figure 4.4: Clusters graphical representation based on volcanic tremor values

converted to a valid date-time format, while all the other columns with numbers (battery, temperature, tilt components, barometer, etc.) were transformed into numeric type. Rows without a valid time were discarded, and empty lines removed. The cleaned dataset now contains the measurements with proper structure, starting from 21 May 2008, including battery voltage, temperatures, tilt readings on X and Y axis, and barometric pressure.

- Volcanic tremor: The script renamed the columns to "Data" and "RMS", converted the numerical values to proper float format (comma to dot), and changed the date column to date-time using day-first format. The data were sorted by time and rows with missing values removed. Outliers were filtered using the inter-quartile range (IQR) method, and duplicated dates were discarded. The time series was then resampled to daily frequency to ensure continuity. The final dataset contains about 698 measurements, with RMS values ranging mostly between 1.5×10^{-6} and 2.0×10^{-6}, average around 1.74×10^{-6}, and no negative or missing RMS values after cleaning. This indicates a relatively stable tremor signal in the considered period, with limited variability and no extreme peaks.

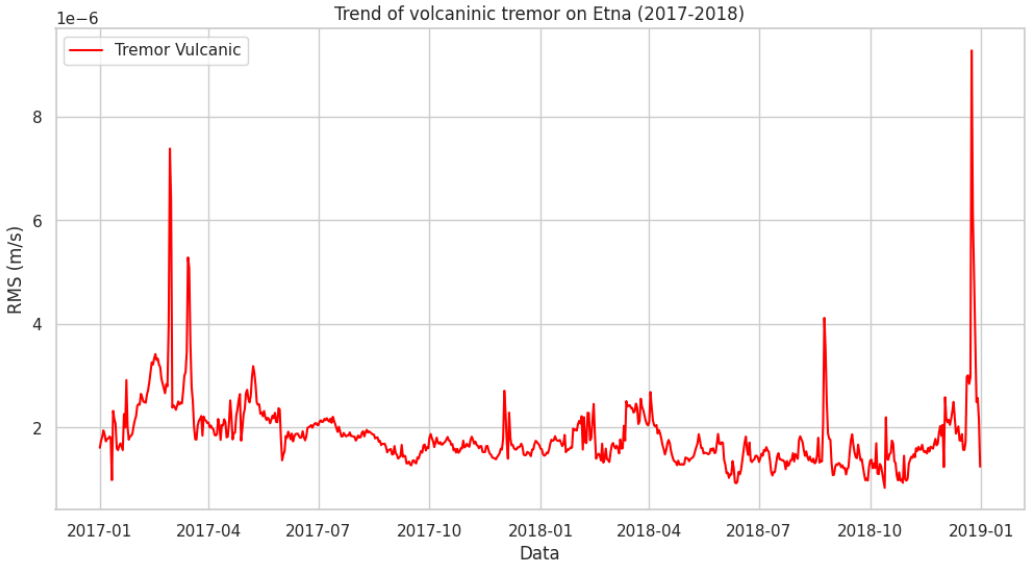

Figure 4.5: Here the trend of the volcanic tremor detected on Etna during 2017-2018

- The figure shows the daily RMS values of volcanic tremor recorded at Etna during 2017–2018, classified into three clusters. Cluster 1 (red) represents the background activity with relatively stable and lower RMS values around 1.5–2.0 $\times 10^{-6}$ m/s, which dominates most of the time series. Cluster 3 (green) indicates intermediate fluctuations with several moderate increases spread over both years. Cluster 2 (blue) marks the highest peaks, mostly concentrated in early 2017 and at the very end of 2018, where the RMS values rise above 6×10^{-6} m/s. This clustering highlights periods of enhanced volcanic tremor and separates them from the quieter background phases.

- As to remember definition of RMS: The RMS (Root Mean Square) represents the average energy of the seismic signal over a specific time window. It is commonly used to quantify volcanic tremor intensity, where higher RMS values indicate stronger ground vibrations and potentially higher volcanic activity.

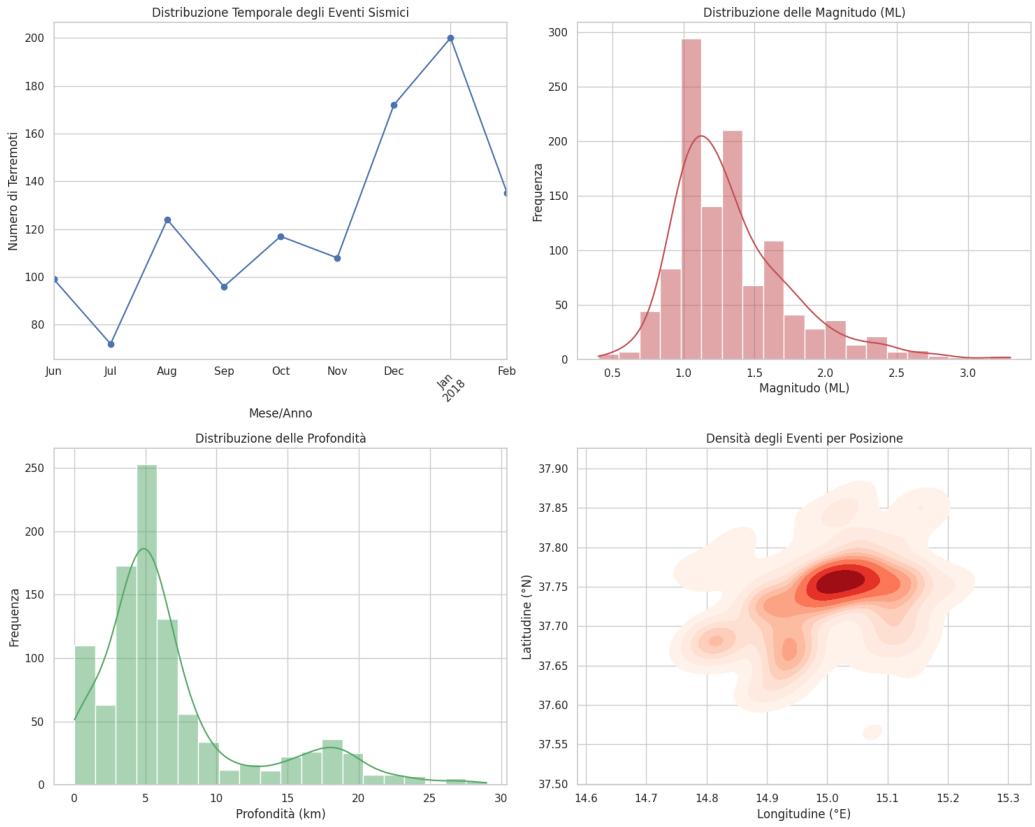

Figure 4.6: Graphical representation of earthquake data showing distribution, number of events, and their density.

- I have added this useful plot cause shows the daily RMS values of volcanic tremor at Etna during 2017–2018. For most of the period, the tremor remains in a low to moderate range around 1.5–2.0×10^{-6} m/s, with some notable peaks. The most significant increases occur in early 2017 and late 2018, where the RMS exceeds 7×10^{-6} m/s, indicating short episodes of stronger volcanic tremor possibly linked to higher volcanic activity. RMS, as you can easily understand, is one of the most used index to show graphical representation in Volcano's data analysis.

- Earthquake: The most crucial and interesting file/paper, one about earthquake. After the same cleaning operations applied for the previous variables, i have create a few different graphs to understand better what

is happening. There is a timeline to see how many earthquakes happen each month, histograms to show the distribution of magnitude and depth, and a density map to see where most of the events are located on the volcano. Finally, I add a simple function that lets you put latitude and longitude, and it will show on a map where the signal was recorded. This gives a clear view of the earthquakes and helps to prepare for other models or analysis.

4.1 Merging Dataset and analyzing relevant aspect about features

4.1.1 Let's merge all the cleaned files obtained

Here I take many different files that was already cleaned before, like heat flux, SO2 flux, HCl flux, He ratio, CO2/SO2 ratio, clinometer, volcanic tremor and earthquakes. I make a script that load each file one by one and check if they have a column with date or time, because is important to put them all in the same timeline. Later on i have converted it to real date-time and I have removed duplicated rows that can make problems later. After this, I start to merge them, beginning from first file and adding all the others step by step. I use outer join, so even if one file miss some dates, we not lose the data of the other files. At the end I have a very big data-frame, with more than 480 thousand rows and 27 columns, covering a long period from 2008 to 2022. Many columns have missing values because not all data was recorded at same time, but this is normal for this kind of volcano monitoring data. Then I also make a filter to select only a period until end of 2019, and I take only the columns that are numeric and have enough coverage of data, like temperature, tilt, gas flux and earthquakes information. This final file is good for next steps, because it is easy to use for advanced analysis and solve the problem of missing values.

4.1.2 Correlation Matrix

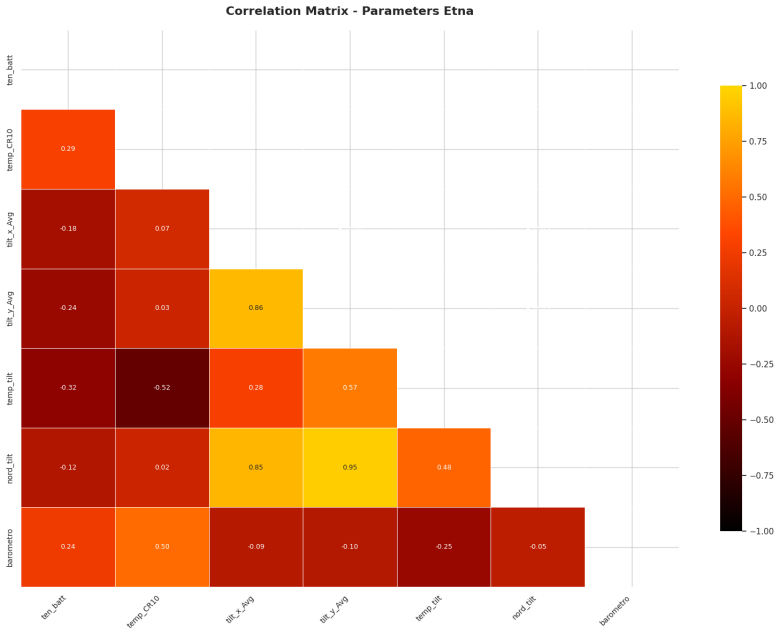

Figure 4.7: Plotted the correlation matrix in order to analyze feature correlation.

For see how the parameters talk each other, I make a correlation matrix using Spearman method. I've tried several correlation matrix types to see which one could be more adapt to our case, at the end i choose Spearman because the data not always go in straight line, and this method is more strong when we have strange values or numbers that jump a lot. I also put a mask on top triangle, to handle the issue of missing values and we can read better. In this graph, the color show how much two parameters go together (positive is red/yellow, negative is black/dark), and I mark with a star when the correlation is very strong, like more than 0.7. This help to find fast which sensors or variables are moving same way or opposite, and can give first idea if there is some connection between tilt, gas, or other signals.

4.1.3 Additional data cleanup operations on merged dataset

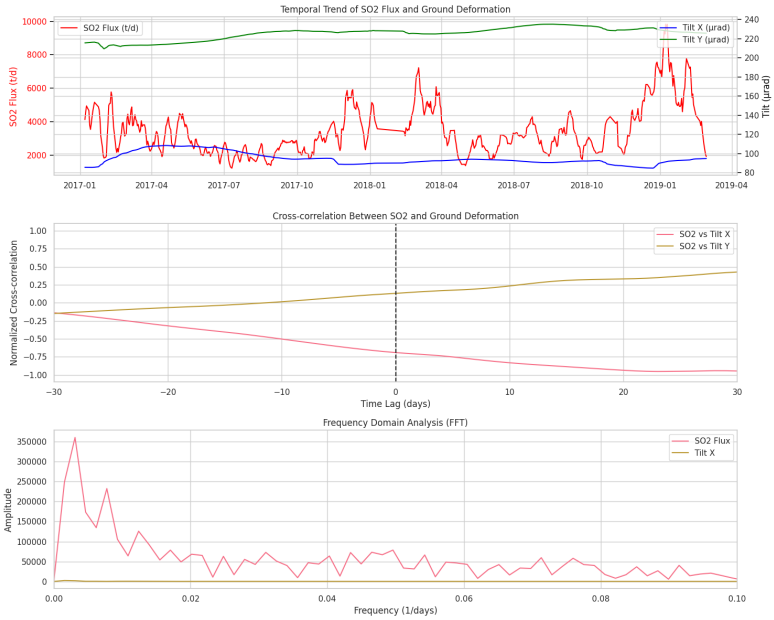

Figure 4.8: Added cleaning operations implemented on the merged dataset to solve other incoming issues.

Using the correlation matrix i have noticed that some issues persists, there was several missing values and outliers underlined on the confusion matrix results; so in order to handle these issues i tried to make a better look to the data of SO_2 emission and ground deformation,tilt (most challenging variables). First I loaded the two datasets and I put them in the same timeline, making average per day to remove too much noise. Then I have smoothed the values with a rolling window [1] to see more clear trend. After that, I check how the SO_2 flux and the tilt are moving together in time, using cross correlation to see if one is coming before or after the other. I also make a frequency analysis (FFT) to see if there is some common periodicity in the signals. At the end, I

[1]Since most of the problems comes by looking at the whole timestamp of the variable cause the imbalance increase on long-term, i have restricted the focus on a restricted window of time to get more robust and unbalanced variable's data

create some plots that show the two time series, their correlation with lag, and also the main frequencies. This help to understand better the dynamic between gas emission and ground movement, and give more clean data for other analysis.

4.1.4 More useful aspects and visualizations

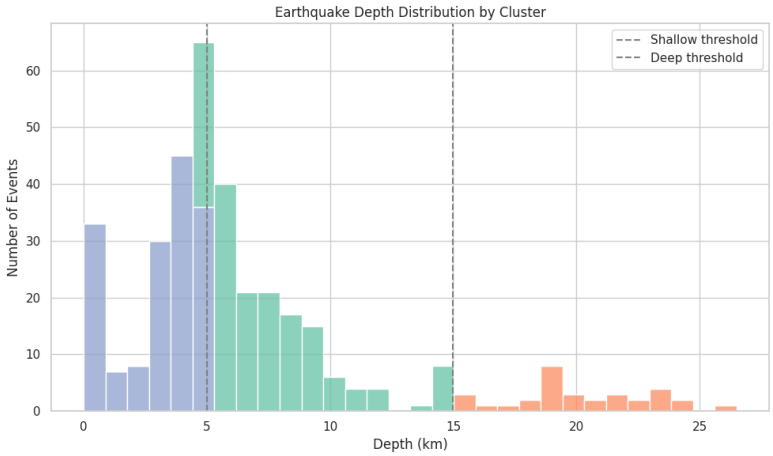

Figure 4.9: First manual clustering including more cleaning process and plotting of distribution of the data.

Now, before the final prediction, it's useful to see how we have classified the earthquake based on the depth of each of them and it's useful to connect earthquake's depth with other parameters (such as SO_2 and clinometry) .In order to do this We fix the depth column because sometime it have comma instead of dot, and we also create a real date-time if is missing. After that we remove wrong or strange data, such as negative depth or missing time. Then we make a simple manual cluster of earthquakes by depth: events less than 5 km we call Cluster 1 (shallow), between 5 and 15 km we call Cluster 2 (normal), and events greater than 15 km are Cluster 3 (deep). This allows us to see the distribution of the seismicity in different depth zones. The histogram shows that most of the events are shallow or intermediate, and only a few are deep. For example in date 07 July 2017 we see many events in Cluster 1 and Cluster 2, especially between 6 and 10 km, and only some very shallow near 3 km. This gives a first impression of how the volcano was active in that period.

4.1.5 Clustering:dividing earthquakes in several clusters based on their deep

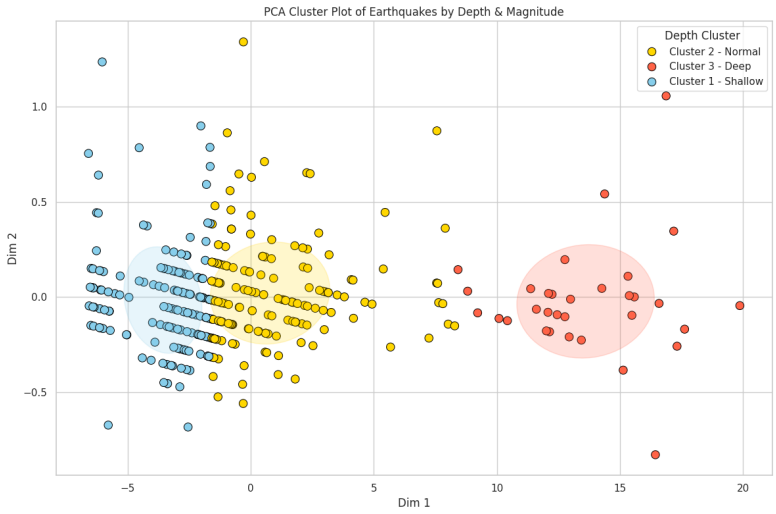

Figure 4.10: More complete clustering visualization taking into account magnitude's variable.

Just before the prediction, we need more deep analysis of the earthquakes because I not only look at the depth like before, but I also take the magnitude and I try to see if they together give some different information. First I clean the data and I fix the depth and the magnitude because sometimes they was with comma or strange value. Then I make a manual cluster based on depth (shallow, normal, deep) like before, but I also use PCA technique to reduce the information of depth and magnitude into two dimension easy to see. With this I can make a scatter plot where each point is one earthquake and the color show the cluster of depth, and I draw some ellipse to see how the group of points stay together. From this plot you can see if the shallow or deep event also have more or less magnitude, and if they make some clear pattern. This is important because it show relation between how deep the earthquake happen and how strong it is, not only one variable but two together.

Chapter 5

Training of the model

5.1 Features standardization

Just before the training we pre-process the dataset to prepare it for machine learning analysis. We first load the numerical time series data and remove any non-numeric columns, such as timestamps, and rows with missing values to ensure data consistency. Subsequently, we apply feature scaling using two approaches: Z-score standardization and Min-Max normalization. The Z-score standardization transforms each feature to have zero mean and unit variance, which is particularly beneficial for algorithms sensitive to the scale of the data, such as Support Vector Machines or Principal Component Analysis.Then i have implemented the creation of sliding windows: each window represents a fixed number of past time steps (30 days in this case), while the target corresponds to the selected feature at a future horizon of one day. This method allows the model to learn temporal dependencies and patterns in the data, which is essential for time series forecasting tasks. Furthermore, the feature selection is driven by the relevance of seismic activity, identified as the most critical and scientifically meaningful variable based on expert knowledge from the Istituto Nazionale di Geofisica e Vulcanologia (INGV).

5.2 Random Forrest

5.2.1 First try: just to see earthquake's path

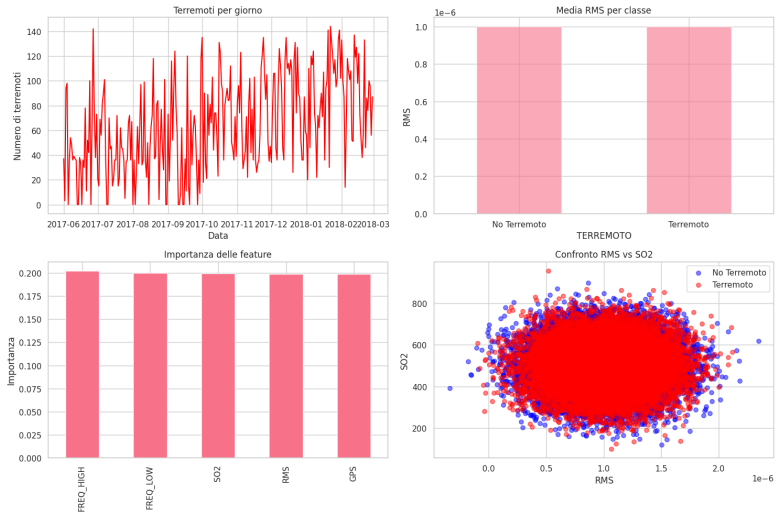

Figure 5.1: Feature importance and number of earthquake registered daily.

Just to give an idea about earthquake frequency, a Random Forest classifier was developed to predict the occurrence of seismic events (`TERREMOTO`) based on five instantaneous geophysical measurements: RMS, low- and high-frequency components, SO_2, and GPS. The model treats each observation as an independent sample without applying temporal aggregation or window-based feature engineering, aiming to directly classify each time point as either "no earthquake" or "earthquake". Random Forest is an ensemble method that builds multiple decision trees on bootstrapped subsets of the data and combines their outputs through majority voting, offering robustness to noise and non-linear relationships. The dataset was split into 70% for training and 30% for testing, and the trained model achieved an overall accuracy of approximately 52%. The recall for the earthquake class was 31% and the precision was 44%, indicating that a substantial number of seismic events were missed (high false negatives) while many non-seismic points were incorrectly flagged as events (false positives). Feature importance analysis revealed that all five variables contributed nearly equally, with no single dominant predictor emerging. This is caused first of all by the fragility of the model, without

any rolling window or robust cleaning process ore features engineering one to prepare feature and dataset to the model; the second reason is the fact that, and we'll see even later on, earthquake are natural complex and unpredictable events, so the performance of the models decreases drastically taking into account the "real world" aspect. In addition it's more an exploratory analysis to see if the raw signals contains useful information to predict earthquake, but it had bad results.

5.2.2 Improving performance and trying center our scope

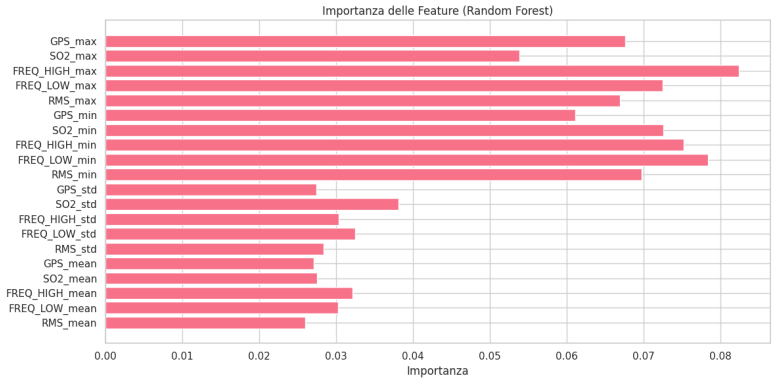

Figure 5.2: Visualization of most relevant features based on alarm system.

Now is time to go deeper trying to create a predictive system-like to know if there will an earthquake in the next step. The model leverages five key geophysical features (RMS, FREQ_LOW, FREQ_HIGH, SO_2, GPS), which were normalized using Min-Max scaling and transformed through a sliding window of 30 time steps. For each window, four statistical descriptors (mean, standard deviation, minimum, and maximum) were extracted, resulting in a total of 20 derived features. The dataset was split into training (80%) and testing (20%) sets, maintaining class balance via stratification and using class weights during training. The model achieved an accuracy of 87%, with a recall of 92% for the earthquake class, indicating a strong ability to detect seismic events while accepting a moderate level of false alarms (precision = 81%). The confusion matrix shows few missed events (FN = 287) compared to correctly predicted ones (TP = 3254), which is suitable for early-warning purposes. Feature importance analysis revealed that extreme values (maximum and minimum) of high- and low-frequency seismic signals, RMS, and

gas emissions were the most influential predictors, highlighting the relevance of peak variations in anticipating seismic activity. So the model increases a lot the performances thanks to the more robust pre-processing phase and give a real idea of the features importance to know the happening or not of an earthquake; this model focused only on the happening probability of an event. It doesn't get temporal sequences and the scope of seismic precursors;Furthermore the model is not robust and it risks to generate overfitting and several unnecessary alarms.

5.3 Training of LSTM model

5.3.1 Starting feature engineering

Before the training is necessary to have a target variable on which the model will be based, so we constructed a binary target variable named `quake_next24h`. This variable indicates whether a significant seismic event will occur within the next 24 hours from each recorded observation in the dataset. Specifically, for each earthquake timestamp extracted from the earthquake's dataset, we defined a temporal window of 24 hours preceding the event. All samples falling within this window were labeled as `1`, indicating a positive class ("earthquake within 24 hours"), while all other samples were assigned `0`. This procedure transformed the continuous volcanic monitoring time series into a supervised dataset suitable for training predictive model, Long Short-Term Memory (LSTM) one.

5.4 Training our model

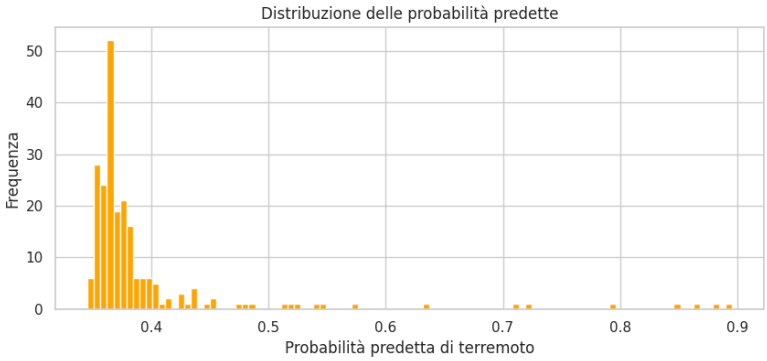

Figure 5.3: Trend of LSTM predictive model base on target label earthquake24.

To forecast the occurrence of significant seismic events, we developed a Long Short-Term Memory (LSTM) neural network using historical volcanic and seismic data. The model was trained to perform binary classification, predicting whether at least one earthquake with a magnitude $M_L \geq 2.0$ would occur within the next 24 hours. Input features consisted of time-ordered numerical measurements (e.g., seismic parameters, radiative flux, geophysical indicators), organized into sliding windows of 24 hours to capture temporal dependencies. Class imbalance, due to the relative rarity of earthquakes compared to non-event periods, was mitigated by applying class weighting during training. The network architecture comprised a single LSTM layer with 64 units, followed by a dropout layer (0.3) to reduce overfitting, a fully connected hidden layer with 32 neurons (ReLU activation), and a final sigmoid output for probability estimation. Despite achieving moderate accuracy ($\sim$77%), the model showed low recall for the positive class, meaning that many actual earthquake events were missed. This limitation arises from both the scarcity of positive samples and the intrinsic complexity of earthquake triggering mechanisms, which cannot be fully captured by the available features. Furthermore, the model tends to produce conservative probability estimates (mostly in the range 0.3–0.4), reflecting uncertainty in the predictions. For this reason we'll focus our attention on other metrics such as recall and F1-Score.

5.5 Improving performance and trying center our scope

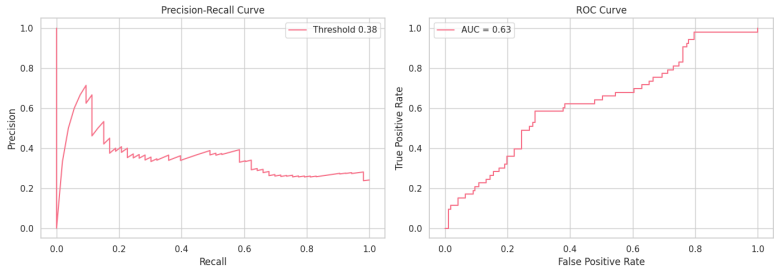

Figure 5.4: Focus on "earthquake events" adjusting F1-score and recall.

As we say we're focus now on other metrics such as Recall, but why? The precision in the first development of our model has reached high level; this is cause most of the prediction are corrected, but in the majority of cases the prediction is "no earthquake" and this is generally the limit of the precision which focused its attention only on the number of correct prediction, instead we want to find the "earthquake" cases which are more rare.The model focuses on two key metrics: recall and the F1-score. The decision to prioritize recall stems from the practical need to minimize false negatives, i.e., cases where an actual seismic event is not detected by the system. Missing an event can have severe consequences in terms of public safety and early warning systems, especially when we're talking about earthquakes. However, optimizing exclusively for recall may lead to an excessive number of false positives (false alarms), which can reduce trust in the predictive system. For this reason, the F1-score is adopted as a complementary metric, as it represents the harmonic mean between precision and recall, offering a balanced assessment of the model's predictive capabilities.

To achieve this balance, a dynamic threshold selection process was implemented, where the decision boundary was adjusted to guarantee a recall of at least 0.90 while maximizing precision. Furthermore, a temporal smoothing mechanism was applied to reduce spurious isolated alerts, and class weights were incorporated to address the natural imbalance between earthquake and non-earthquake periods. From a theoretical perspective, this approach shifts the problem from a standard classification task to a risk-aware predictive

modeling problem, where the cost of misclassification is asymmetric and explicitly encoded in the optimization strategy. Practically, this means that the system is designed to err on the side of caution, accepting a moderate increase in false alarms in exchange for a significant reduction in missed events, which is crucial in early warning contexts.

5.6 LSTM multivariate

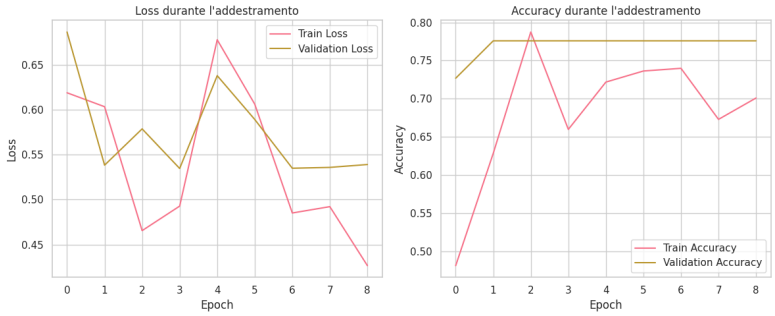

Figure 5.5: Multivariate LSTM fitting graphical results.

Building upon the previous approach, which primarily relied on precision as its main indicator of performance, this final implementation introduces a multivariate Long Short-Term Memory (LSTM) model that better addresses the specific needs of seismic early warning systems. In the first development phase, the model achieved high precision mainly because the majority of predictions were "no earthquake", which, although correct in most cases, led to a significant number of missed events (false negatives). This highlighted a critical limitation: precision alone does not adequately reflect the ability to detect rare but high-impact events such as earthquakes.

To overcome this, the LSTM-based model was designed to prioritize recall and improve the F1-score, ensuring a more balanced trade-off between detecting true events and controlling false alarms. This shift reflects a risk-averse strategy, where minimizing false negatives is considered more critical than maintaining a very high overall accuracy. Practically, this means that the system tends to classify more sequences as potential precursors, thus capturing a higher proportion of real events even at the cost of a moderate increase in false positives.

From a theoretical standpoint, the model operates on multivariate time series of volcanic and geophysical signals, using sequences of 96 time steps as inputs and predicting the binary likelihood of an earthquake occurring within the next 24 hours. Class imbalance was explicitly addressed using weighted loss functions, and the architecture consists of two stacked LSTM layers with dropout regularization and a final sigmoid unit. The optimization process employed binary cross-entropy loss with the Adam optimizer and incorporated early stopping to prevent overfitting. Compared to the initial model, this configuration resulted in a significant improvement in the F1-score, providing a more reliable and operationally meaningful assessment of predictive performance for early warning purposes.

5.7 Reviewing the results

The first important observation is that, although algorithms are often perceived as "magic wands," they are not. In real-world contexts, data typically present several challenges due to the inherent difficulty of collecting them and their complex interpretation by predictive models. Once pre-processing issues are addressed, Machine Learning and Deep Learning models may achieve theoretically high performance; however, when applied to natural events, their accuracy tends to decrease due to the unpredictable nature of such phenomena.

In this work, the implemented models can be considered as simulations based on real data recorded during the 2018 eruption of Mount Etna. These results may serve as a starting point for analyzing more recent datasets and for developing predictive frameworks aimed at forecasting future events. Although the performance of the final LSTM model decreased compared to the initial ones, it can be regarded as more realistic, as it better reflects the complexity of actual seismic phenomena.

Chapter 6

Conclusion

All of us are becoming increasingly aware of the relevance of data in every field and of how it is transforming our daily lives. This is particularly true in sensitive areas such as earthquake monitoring, where the accurate collection, processing, and transmission of data can play a crucial role in safeguarding human lives.

This thesis aimed to provide an extensive overview of the role of the data scientist and of the fundamental principles of data science, introducing the basic concepts of machine learning and deep learning. The study then focused on a practical case involving data collected from Mount Etna, following the eruption and subsequent seismic events that occurred in 2018.

Several pre-processing techniques were applied to clean and prepare the data for training, addressing the common challenges of incomplete and irregular datasets in this field. Subsequently, multiple machine learning and deep learning models were implemented to predict the occurrence of an earthquake within a 24-hour time window.

The results highlight two key insights: first, that real-world events inevitably contain an unpredictable component, meaning that even theoretically strong models may experience a substantial decrease in performance when applied in practice; and second, that for this type of data, multivariate LSTM models demonstrate greater robustness and a higher capacity to capture actual earthquake occurrences, reducing the risk of missing critical events.

Bibliography

[1] Geopop, *Cos'è il Machine Learning: esempi e applicazioni*, 2023. Disponibile su: https://www.geopop.it/cose-il-machine-learning-esempi/ [Accessed: 7 September 2025].

[2] Istituto Nazionale di Geofisica e Vulcanologia (INGV), *Monitoraggio dell'Etna*, 2024. Disponibile su: https://www.ingv.it/ [Accessed: 7 September 2025].

[3] Paonita, A., Liuzzo, M., Salerno, G., Federico, C., Bonfanti, P., Caracausi, A., Giuffrida, G., La Spina, A., Caltabiano, T., Gurrieri, S., Giudice, G., *Intense overpressurization at basaltic open-conduit volcanoes as inferred by geochemical signals: The case of the Mt. Etna December 2018 eruption*, Science Advances, 2021. Disponibile su: https://www.science.org/doi/10.1126/sciadv.abg6297 [Accessed: 7 September 2025].

[4] De Gori, P., Giampiccolo, E., Cocina, O., Branca, S., Doglioni, C., Chiarabba, C., *Re-pressurized magma at Mt. Etna, Italy, may feed eruptions for years*, Communications Earth & Environment, 2021. Disponibile su: https://doi.org/10.1038/s43247-021-00282-9 [Accessed: 7 September 2025].

[5] Büyükakpınar, P., Cannata, A., Cannavò, F., Carbone, D., De Plaen, R. S. M., Di Grazia, G., King, T., Lecocq, T., Liuzzo, M., Salerno, G., *Chronicle of Processes Leading to the 2018 Eruption at Mt. Etna As Inferred by Seismic Ambient Noise Along With Geophysical and Geochemical Observables*, Journal of Geophysical Research: Solid Earth, 2022. Disponibile su: https://doi.org/10.1029/2022JB025024 [Accessed: 7 September 2025].

[6] Bonforte, A., Guglielmino, F., *Very shallow dyke intrusion and potential slope failure imaged by ground deformation: The 28 December 2014 eruption on Mount Etna*, Geophysical Research Letters, 2015. Disponibile su: `https://doi.org/10.1002/2015GL063462` [Accessed: 7 September 2025].

Appendix A

Appendix: Reports Code

In this appendix, we report the most relevant portions of the code used during the analysis. The full scripts are not included; instead, we provide the key sections as representative examples to illustrate the implemented procedures. All code is fully reproducible.

A.1 Data Preprocessing

```
import pandas as pd
import numpy as np

# ======================================
# 1. FUNZIONI UTILI PER IL PREPROCESSING
# ======================================

def clean_datetime(df, col="Data", dayfirst=True):
    """Converte colonna data in formato datetime standard."""
    df[col] = df[col].astype(str).str.replace(",", ".")
    df[col] = pd.to_datetime(df[col], dayfirst=dayfirst,
    ↪  errors="coerce")
    return df.dropna(subset=[col])

def remove_duplicates_missing(df):
```

```
    """Rimuove duplicati e righe completamente vuote."""
    return df.drop_duplicates().dropna(how='all')

def handle_outliers_iqr(df, col):
    """Gestione outlier tramite Interquartile Range (IQR)."""
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    return df[~((df[col] < (Q1 - 1.5 * IQR)) | (df[col] > (Q3 + 1.5
    ↪  * IQR)))]

# ======================================
# 2. PULIZIA DELLE SINGOLE VARIABILI
# ======================================

# --- Heat Flux ---
heat = pd.read_csv("heatflux_raw.csv")
heat = clean_datetime(heat, "Data")
heat = remove_duplicates_missing(heat)
heat = handle_outliers_iqr(heat, "HeatFlux")

# --- SO2 Flux ---
so2 = pd.read_csv("so2_flux_raw.csv")
so2 = clean_datetime(so2, "Date")
so2 = remove_duplicates_missing(so2)

# --- HCl Flux ---
hcl = pd.read_csv("hcl_flux_raw.csv")
hcl = clean_datetime(hcl, "Date")
hcl = remove_duplicates_missing(hcl)
hcl = handle_outliers_iqr(hcl, "HCl_flux")

# --- CO2 Flux ---
```

```
co2 = pd.read_csv("co2_flux_raw.csv")
co2 = clean_datetime(co2, "Date")
co2 = remove_duplicates_missing(co2)
co2["f_CO2_norm"] = co2["f_CO2_norm"].str.replace(",", ".").astype(float)

# --- CO2/SO2 Ratio ---
co2so2 = pd.read_csv("co2_so2_raw.csv", skiprows=1)
co2so2 = clean_datetime(co2so2, "Date")
co2so2 = remove_duplicates_missing(co2so2)
co2so2["CO2/SO2"] = co2so2["CO2/SO2"].str.replace(",", ".").astype(float)
co2so2 = co2so2[co2so2["CO2/SO2"] < 20]  # filtro valori anomali

# --- 3He/4He Ratio ---
he_ratio = pd.read_csv("he_ratio_raw.csv")
he_ratio = clean_datetime(he_ratio, "Date")
he_ratio = remove_duplicates_missing(he_ratio)

# --- Clinometry ---
clin = pd.read_csv("clinometry_raw.csv", sep=",")
clin = clean_datetime(clin, "Time")
clin = remove_duplicates_missing(clin)
numeric_cols = clin.columns.difference(["Time"])
clin[numeric_cols] = clin[numeric_cols].apply(pd.to_numeric, errors="coerce")

# --- Volcanic Tremor ---
tremor = pd.read_csv("tremor_raw.csv")
tremor = clean_datetime(tremor, "Data")
tremor = remove_duplicates_missing(tremor)
tremor = handle_outliers_iqr(tremor, "RMS")
tremor = tremor.resample("D", on="Data").mean().interpolate()
```

```python

# --- Earthquakes ---
eq = pd.read_csv("earthquakes_raw.csv")
eq = clean_datetime(eq, "Date")
eq = remove_duplicates_missing(eq)
eq["Depth"] = eq["Depth"].astype(str).str.replace(",", ".").astype(float)
eq["ML"] = eq["ML"].astype(str).str.replace(",", ".").astype(float)
eq = eq[(eq["Depth"] >= 0) & (eq["ML"] >= 0)]

# ======================================
# 3. MERGE FINALE DEI DATASET PULITI
# ======================================

dfs = [heat, so2, hcl, co2, co2so2, he_ratio, clin, tremor, eq]
merged = dfs[0]
for df_next in dfs[1:]:
    merged = pd.merge(merged, df_next, on="Data", how="outer")

merged = merged.sort_values("Data").reset_index(drop=True)
merged = merged.loc[:, merged.columns.notna()]  # rimuove colonne vuote

# ======================================
# 4. SALVATAGGIO RISULTATI
# ======================================
merged.to_csv("merged_cleaned_dataset.csv", index=False)
print(f"Dataset finale: {merged.shape[0]} righe, {merged.shape[1]} colonne")
```

Appendix B

Correlation Analysis and Clustering and other useful function

B.1 Correlation Matrix Computation

The following script shows the creation of a Spearman correlation matrix between the main variables of the merged dataset, including masking of the upper triangle for better visualization.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# Load merged dataset (already cleaned as described in Section 4)
df = pd.read_csv("merged_cleaned_dataset.csv")
df = df.select_dtypes(include=[np.number]).dropna()

# Compute Spearman correlation
corr = df.corr(method='spearman')

# Mask upper triangle for readability
```

```
mask = np.triu(np.ones_like(corr, dtype=bool))

# Plot
plt.figure(figsize=(12, 10))
sns.heatmap(corr, mask=mask, cmap="coolwarm", annot=False, cbar=True)
plt.title("Spearman Correlation Matrix of Volcano Monitoring Variables")
plt.tight_layout()
plt.show()
```

B.2 Earthquake Clustering by Depth and Magnitude

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Load and clean earthquake dataset
eq = pd.read_csv("earthquakes_raw.csv")
eq['Date'] = pd.to_datetime(eq['Date'], dayfirst=True, errors='coerce')
eq['Depth'] = eq['Depth'].astype(str).str.replace(",", ".").astype(float)
eq['ML'] = eq['ML'].astype(str).str.replace(",", ".").astype(float)
eq = eq.dropna(subset=['Date','Depth','ML'])

# Manual clustering by depth
def classify_depth(d):
    if d < 5: return "shallow"
    elif d <= 15: return "intermediate"
    else: return "deep"

```

```
eq['Cluster'] = eq['Depth'].apply(classify_depth)

# PCA for depth-magnitude visualization
features = eq[['Depth','ML']]
pca = PCA(n_components=2)
principal_components = pca.fit_transform(features)
eq['PC1'], eq['PC2'] = principal_components[:,0],
↪  principal_components[:,1]

# Plot clusters
plt.figure(figsize=(8,6))
for cluster, color in zip(['shallow','intermediate','deep'],
                          ['red','green','blue']):
    subset = eq[eq['Cluster']==cluster]
    plt.scatter(subset['PC1'], subset['PC2'], c=color,
    ↪  label=cluster, alpha=0.6)

plt.legend()
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.title("Earthquake Clustering by Depth and Magnitude")
plt.tight_layout()
plt.show()
```

B.2.1 Merging of Cleaned Files

Creation of the merged dataset on which analysis are based on.

```
files = ["heatflux_clean.csv","so2_clean.csv","hcl_clean.csv",
         "co2_clean.csv","co2so2_clean.csv","tremor_clean.csv",
         "earthquakes_clean.csv","clinometry_clean.csv"]

dfs = [pd.read_csv(f, parse_dates=['Date']) for f in files]
```

```
6
7  merged = dfs[0]
8  for df in dfs[1:]:
9      merged = pd.merge(merged, df, on='Date', how='outer')
10
11 merged = merged.sort_values('Date').reset_index(drop=True)
12 merged.to_csv("merged_cleaned_dataset.csv", index=False)
```

Appendix C

Training of the models

C.1 LSTM multivariate with clinometry

```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping

# === CONFIG ===
SEQUENCE_LENGTH = 96
FEATURES = ['ten_batt', 'temp_CR10', 'tilt_x_Avg', 'tilt_y_Avg',
            'temp_tilt', 'nord_tilt', 'barometro']
TARGET = 'quake_next24h'

# === 1. Caricamento ===
df = pd.read_csv("merged_etna_with_quake_target.csv")
print("Colonne trovate:", df.columns)

# === 2. Pulizia ===
df = df[FEATURES + [TARGET]].dropna()
print("Dopo dropna:", df.shape)

# === 3. Normalizzazione ===
scaler = MinMaxScaler()
df[FEATURES] = scaler.fit_transform(df[FEATURES])

# === 4. Creazione sequenze ===
```

```python
def create_sequences(df, features, target, seq_length):
    X, y = [], []
    for i in range(len(df) - seq_length):
        seq = df[features].iloc[i:i+seq_length].values
        label = df[target].iloc[i + seq_length]
        X.append(seq)
        y.append(label)
    return np.array(X), np.array(y)

X, y = create_sequences(df, FEATURES, TARGET, SEQUENCE_LENGTH)
print("X shape:", X.shape, "y shape:", y.shape)

# === 5. Train/Test split ===
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, shuffle=False)

# === 6. Class weights ===
neg, pos = np.bincount(y_train.astype(int))
total = neg + pos
class_weight = {
    0: (1 / neg) * (total / 2.0),
    1: (1 / pos) * (total / 2.0)
}
print("Class weights:", class_weight)

# === 7. Modello LSTM ===
model = Sequential([
    LSTM(64, return_sequences=True, input_shape=(SEQUENCE_LENGTH, len(FEATURES))),
    Dropout(0.3),
    LSTM(32),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# === 8. Training ===
early_stop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    epochs=30,
    batch_size=64,
    class_weight=class_weight,
    callbacks=[early_stop],
    verbose=2
)

# === 9. Valutazione ===
```

```
y_pred = (model.predict(X_val) > 0.5).astype(int)
print("\n=== CLASSIFICATION REPORT ===")
print(classification_report(y_val, y_pred, digits=4))
```