FINAL PROJECT - TF 234801

DESIGNING AN INFRASOUND-BASED REGIONAL VOLCANO MONITORING, CLASSSIFICATION AND FORECASTING FOR ANAK KRAKATAU

MUHAMAD HAYKAL HANIF GIFARI ADI
NRP. 02311940000154

Advisor
Dr. Eng. Dhany Arifianto, ST. M.Eng
NIP 197310071998021001

Study Program Bachelor of Engineering Physics
Department of Engineering Physics
Faculty of Industrial Technology and Systems Engineering
Institut Teknologi Sepuluh Nopember
Surabaya
2024

FINAL PROJECT - TF 234801

DESIGNING AN INFRASOUND-BASED REGIONAL VOLCANO MONITORING, CLASSIFICATION AND FORECASTING FOR ANAK KRAKATAU

MUHAMAD HAYKAL HANIF GIFARI ADI
NRP 02311940000154

Advisor

Dr. Eng. Dhany Arifianto, ST. M.Eng

NIP 197310071998021001

Study Program Bachelor of Engineering Physics
Department of Engineering Physics
Faculty of Industrial Technology and System Engineering
Institut Teknologi Sepuluh Nopember
Surabaya
2024

Halaman ini sengaja dikosongkan

STATEMENT OF ORIGINALITY

The undersigned below:

Name of student / NRP : Muhamad Haykal Hanif Gifari Adi
Department : Engineering Physics
Advisor / NIP : Dr. Eng. Dhany Arifianto, ST., M.Eng.

hereby declare that the Final Project with the title of **"Designing an Infrasound-Based Regional Volcano Monitoring, Classification and Forecasting for Anak Krakatau"** is the result of my own work, is original, and is written by following the rules of scientific writing. If in the future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with the provisions that apply at Institut Teknologi Sepuluh Nopember.

Surabaya, 15 January 2024
Student

(Muhamad Haykal Hanif G. A.)
NRP. 02311940000154

Halaman ini sengaja dikosongkan

APPROVAL SHEET

FINAL PROJECT

DESIGNING AN INFRASOUND-BASED REGIONAL VOLCANO MONITORING, CLASSIFICATION AND FORECASTING FOR ANAK KRAKATAU

By:

Muhamad Haykal Hanif Gifari Adi

NRP. 02311940000154

Approved:

Advisor

Dr. Eng. Dhany Arifianto, ST. M.Eng

NIP. 197310071998021001

Acknowledged,

Kepala Departemen

Teknik Fisika FTIRS, ITS

Dr. Suyanto, S.T, M.T

NIP. 197111131995121002

SURABAYA

February, 2024

Halaman ini sengaja dikosongkan

APPROVAL SHEET

DESIGNING AN INFRASOUND-BASED REGIONAL VOLCANO MONITORING, CLASSIFICATION AND FORECASTING FOR ANAK KRAKATAU

FINAL PROJECT

Submitted to fulfill one of the requirements
for obtaining a degree Bachelor of Engineering at
Undergraduate Study Program of Engineering Physics
Department of Engineering Physics
Faculty of Industrial Technology and Systems Engineering
Institut Teknologi Sepuluh Nopember

By:

MUHAMAD HAYKAL HANIF GIFARI ADI

NRP. 02311940000154

Approved by the Final Project Examiner Commitee:

1.	Dr. Eng. Dhany Arifianto, ST. M.Eng	Advisor
2.	Agus Muhamad Hatta, S.T., M.Si., Ph.D.	Chair Examiner
3.	Dr. Ir. Purwadi Agus Darwito, M.Sc.	Examiner
4.	Dr.-Ing. Doty Dewi Risanti, S.T., M.T.	Examiner

SURABAYA

February, 2024

Halaman ini sengaja dikosongkan

DESIGNING AN INFRASOUND-BASED REGIONAL VOLCANO MONITORING ,CLASSIFICATION AND FORECASTING FOR ANAK KRAKATAU

Student Name /NRP	**:**	**Muhamad Haykal Hanif Gifari Adi / 02311940000154**
Department	**:**	**Engineering Physics FTIRS – ITS**
Advisor	**:**	**Dr. Eng. Dhany Arifianto, ST., M.Eng.**

Abstract

This study investigates the application of machine learning to the detection and clustering of seismo-acoustic signals for volcanic activity monitoring, focusing on the Anak Krakatau eruption. Utilizing the Adaptive F-Detector and Bartlett beamforming, we analyzed infrasound signals from the I06AU station to identify characteristics corresponding to different volcanic states. A Convolutional Autoencoder was employed for feature extraction, proving effective in encoding crucial seismic signatures. Deep Embedded Clustering (DEC) was applied to the reduced-dimensionality data, revealing distinct patterns significant for understanding volcanic activities. Our findings demonstrated that, upon iterative fine-tuning, the machine learning model adeptly captured both temporal and frequential features. These features are then utilized to classify different types of volcanic activity, showcasing the model's capability to discern between various states of volcanic unrest. Additionally, the incorporation of Time-to-event forecasting using Gradient Boosted Trees further enhanced our ability to predict eruptive activities, underscoring its potential as a cornerstone for early detection systems. Despite the inherent unpredictability of volcanic systems, the model achieved an appreciable level of accuracy, with a mean squared error (MSE) value of 22.167 for unsupervised pattern discovery, and in eruption forecasting, it delivered a mean absolute error (MAE) of 73.41 minutes and a root mean square error (RMSE) of 106.44, alongside an R^2 of 75.4%. Although these results indicate the model's efficacy, they also point to the necessity for ongoing refinement to enhance its predictive precision.

Keywords: ***infrasound, seismo-acoustic, volcanic activity***

Halaman ini sengaja dikosongkan

ACKNOWLEDGEMENT

Alhamdulillah, all praises to Allah the Almighty of God the most Gracious and the most Merciful for his blessings and mercy for He allows the completion of this research as part of the requirements to graduate from bachelor's degree in Engineering Physics from the department of Engineering Physics, Insitut Teknologi Sepuluh Nopember. This was achieved by completing the Final Project entitled **"Designing an Infrasound-based Regional Volcano Monitoring, Classification and Forecasting for Anak Krakatau**". This report is the culmination of a long journey that I have undertaken to complete this study. Many prayers, help, and support from various parties have been instrumental in the completion of this final assignment.. A thousand thanks to:

1. Dr. Eng. Dhany Arifianto, ST. M.Eng as this research's advisor for his guidance and expertise in the subject
2. Mr. Hatta, Mr. Darwinto, and Mrs. Risanti as the examiner for this research's proposal and final thesis defense who have given critical criticisms and suggestions for the final project's improvement
3. Dr. Zahra Zali, for publicly opening the source code for her PhD thesis and answering author's questions regarding its underlying processes
4. The author's parents, Cahyono Adi and Rimadani Nurhasmy, for their prayers, advice, trust and financial support to sustain the research
5. The author's brother, Adrian Gifariadi, for his lessons, his outlook over politics, and constant repetitive dark humours
6. The people from *Himpunan Mahasiswa Islam Koordinator Komisariat Sepuluh Nopember*, for being my source of stress relieve watching the idiotic things being done by the members

As well as other people who cannot be mentioned one by one. Hopefully this final assignment report can be used as well as possible

Surabaya, 8 February 2023

Haykal Hanif

Halaman ini sengaja dikosongkan

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I
INTRODUCTION

1.1 Background

Volcanic eruptions have killed more than 300,000 people worldwide since the 16th century, with Southeast Asia accounting for more than half of this total (Brown *et al.*, 2017). The ensuing economic devastation has also been disastrous, costing the world economy upwards of a few trillion USD over the previous century (Daniell *et al.*, 2015). Sadly, 9 to 14% of the world's population still reside within 100 km of an active volcano, with Southeast Asia once again making up the majority, despite the significant hazards associated with doing so (Freire *et al.*, 2019). This condition also applies to Indonesia where there are 36 cities in Indonesia that have a high risk of volcanic eruptions.

Anak Krakatau, an active volcanic island located in the Sunda Strait between Java and Sumatra in Indonesia, has a complex and dynamic geological history. It emerged from the caldera of Krakatau, which famously erupted in 1883. The region's tectonic setting, part of the Pacific Ring of Fire, makes it prone to frequent volcanic activity. Anak Krakatau has experienced several significant eruptions, the most recent major one occurring in 2018, leading to a devastating tsunami (Walter *et al.*, 2019; Cutler *et al.*, 2022). This event highlights the volcano's potential to trigger not only eruptive hazards but also secondary disasters like tsunamis, especially considering its geological instability and the densely populated areas surrounding the Sunda Strait.

Geologically, Anak Krakatau has been characterized by a series of eruptive phases, with varying intensities and styles. These phases have shaped the island's current morphology, which continues to evolve. Studies have shown that the volcano's activity is influenced by magmatic intrusions and flank instabilities, which can lead to sudden and unpredictable changes in eruptive behavior (Zorn et al., 2023). The island's growth and collapse cycles pose significant risks, as evidenced by the 2018 event, where a significant portion of the volcano collapsed, generating a deadly tsunami (Williams, Rowley, & Garthwaite, 2019). The full eruptive history of Krakatau Volcano can be seen on the **Table 1.1**

As a result, volcanologists throughout the world have been working to get a better understanding of these dynamic but unpredictable processes to reduce the risks associated with volcanic eruption. The utilization of infrasound signals emanating from volcanic activities has

garnered significant attention. Infrasound, a type of sound wave that is below the frequency range of human hearing (<20 Hz) waves which can travel over large distances from a volcanic center (Caudron *et al.*, 2015) has proven effective in monitoring and analyzing volcanic activities. Using infrasound, volcano monitoring efficiently delivers on all fronts, expanding our understanding of volcanoes while supplying crucial information needed for hazard assessment and prompt mitigation activities.

Table 1.1 Eruptive History of Anak Krakatau Volcano

No	Eruption Period	Status	Max VEI
1	2021 May 25 – 2023 Oct 11 (Continuing)	Confirmed Eruption	2
2	2018 Jun 18 – 2020 Apr 17	Confirmed Eruption	3
3	2017 Feb 17 – 2017 Feb 19	Confirmed Eruption	1
4	2014 Mar 31 – 2014 Mar 31	Confirmed Eruption	1
5	2013 Oct 31 – 2013 Oct 31	Confirmed Eruption	1
6	2013 Mar 11 – 2013 Mar 29	Confirmed Eruption	1
7	2010 Oct 25 (?) – 2012 Sep 9 ± 1 days	Confirmed Eruption	2
8	2009 Mar 25(?) – 2009 Sep 16(?) ±15 days	Uncertain Eruption	2
9	2007 Oct 23 ± 2008 Aug 30 (?)	Confirmed Eruption	2
10	2001 Jul 21 – 2001 Sep 17	Confirmed Eruption	1
11	2000 May 29 – 2000 Oct 30 (?)	Confirmed Eruption	1

Volcano infrasound has changed during the past two decades from a subject of academic research to a practical monitoring instrument as it can quantify eruption parameters including volumetric and mass flow rates directionality of eruptive blasts and plume height. It is currently used for volcano monitoring by volcano observatories across the world to identify, locate, and characterize volcanic activity (Watson *et al.*, 2022). It has proven effective to employ infrasound for near-real-time eruptions at volcanoes, such as Tungurahua (Fee, Garces and Steffke, 2010), Etna (Ripepe *et al.*, 2018), Stromboli (Le Pichon *et al.*, 2021), Kilauea (Garcés *et al.*, 2003), and Sakarajima (Albert, 2015). In the coming years, infrasound will go beyond simple detection and source localization into providing estimation of eruption mass and mass flow rate even for big eruptions in near real-time (Fee *et al.*, 2017).

Beyond monitoring, infrasound signals can also be used to generate and disseminate timely and meaningful warning information to enable individuals, communities and organizations threatened by a hazard to prepare and to act appropriately and in sufficient time

to reduce the possibility of harm or loss; in short: an Early Warning System. This type of system has been commonly used in various places such as Mt Etna (Ripepe *et al.*, 2018), Peru (Machacca Puma et al., 2021), United States (John, Guffanti and Thomas, 2005) and Mt Ruapehu (Christophersen, Behr and Miller, 2022) which uses local and remote multiparametric parameters (seismicity, infrasound signals, ground deformation, gas geochemistry and video cameras) through a specific algorithm to enable early warning notifications. Currently there are various forecasting techniques to employ for early warning systems: (1) Expert interpretation (opinion), (2) Event trees, (3) Belief networks, (4) Failure forecasting, (5) Process / Source models, and (6) Machine-learning algorithms (Carniel *et al.*, 2020).

Considering Anak Krakatau's active status and its potential to cause widespread damage, continuous monitoring and advanced forecasting systems are crucial. The use of infrasound signals, coupled with remote sensing and seismic data, can provide valuable insights into the volcano's behavior. This can aid in early warning systems, which are vital for the safety of the communities in the vicinity of Anak Krakatau and for maritime activities in the Sunda Strait (Perttu, Caudron, et al., 2020; Triyangoro and Ontowirjo, 2021). This function enables the likelihood of event occurrence to be forecasted in advance through understanding of the natural processes that generate hazards, together with past experience and monitoring of current conditions (Jacks, Davidson and H. G., 2010). Since a comprehensive infrasound-based volcanic monitoring, classification, and forecasting system has not been fully implemented before, there is no existing alarm system that can provide reliable predictions.

Zali (2023) has provided a strong framework for a comprehensive monitoring and classification system for underground seismic signals, but it still lacks a forecasting function. Accordingly, our research focuses on the development of an infrasound-based regional monitoring, classification, and forecasting system for Anak Krakatau. The objectives are threefold: firstly, to refine and enhance the methods for extracting infrasound signals linked to volcanic activities, an approach that holds promise for more nuanced and detailed data than traditional methods. Secondly, this study aims to delve deeper into the interpretation of these signals, striving to uncover the complex and intricate processes underlying volcanic systems. Finally, and most crucially, the research seeks to rigorously test and validate the reliability and accuracy of eruption forecasting models. Through this research, we aspire to contribute significantly to the evolving field of volcano monitoring, offering new perspectives and tools to predict and understand one of nature's most formidable phenomena.

1.2 Research Questions

By conducting the research, several problems were identified in the paper. The problems are as follows:

a. How can infrasound signals related to volcanic activity be effectively extracted using current methodologies?
b. What patterns or trends in volcanic activity are revealed through Deep Embedded Clustering analysis?
c. How can the extracted infrasound signals be integrated into a forecasting system for volcanic eruptions?

1.3 Research Objectives

Based on the above background, the problem formulation in this study are:

a. To develop and refine methodologies for the effective extraction of infrasound signals associated with volcanic activity.
b. To interpret patterns to gain insights into the underlying processes and dynamics of volcanic systems.
c. To test and validate eruption forecasting to ensure its accuracy in forecasting volcanic eruptions.

1.4 Research Scope

The scope of the research may be limited to the understanding of eruption dynamics and development of the early warning system for Anak Krakatau. Real-time monitoring is not available due to the availability of the dataset

1.5 Research Relevance

Anak Krakatau, located in Indonesia, is an active stratovolcano that has a history of frequent and destructive eruptions. Developing a predictive early warning system based on infrasound signals could potentially help to reduce the impact of these eruptions by providing advance warning of an impending eruption.

1.6 Report Systematics

This final assignment report consists of four chapters. Chapter 1 is entitled introduction with background sub-chapters, problem formulation, objectives, problem limitations. Chapter

2 contains a literature review surrounding the research topic. Chapter 3 contains the research design, methods used and steps for completing the research. Chapter 4 contains results and discussion, and Chapter v contains conclusions and suggestions

Halaman ini sengaja dikosongkan

CHAPTER II
LITERATURE REVIEW & THEORITICAL FRAMEWORK

2.1 Volcano Acoustics

Acoustics is known as the study of the production, control, transmission, reception, and effects of sound. Infrasound, defined as low-frequency acoustic waves typically below 20 Hz, is a subset of acoustic waves that, due to their low frequency, exhibit unique propagation characteristics and interactions with the environment. Infrasound acoustics has a significant history in both natural and man-made event detection. Initially discovered after the Krakatoa eruption in 1883, infrasound's capability to travel immense distances (over 100 kilometers) and around the globe multiple times was recognized (Le Pichon, Blanc and Hauchecorne, 2019).

During World War II and the era of atmospheric nuclear testing. The infrasound technology was extensively used for explosion detection, gaining substantial attention. However, this interest diminshed when nuclear tests moved underground following the Limited Test Ban Treaty (CTBT) of 1963. In 1996, another CTBT pact was formed that required the establishment of a verification framework, of which the IMS infrasound network is a part of the IMS infrasound network was intended to detect air explosions everywhere on the earth, is equal to 1 kiloton of TNT. However, studies demonstrate that long-range stratospheric infrasound ducting considerably improves the detection capabilities, with explosions equivalent to a few hundred tons of TNT being detectable worldwide at two or more stations across the whole network (Le Pichon *et al.,* 2009). Since the introduction IMS infrasound network, infrasound technology has advanced quickly, which has enhanced our capacity to utilize infrasound not only to detect nuclear explosions, but to comprehend and monitor volcanic processes.

A volcano can produce seismo-acoustic waves by a variety of processes, from discrete rock impacts to the harmonic echoes of lava tubes to impulsive punctuations of strombolian and volcanic explosions, and culminating in roaring jets of subplinian and plinian eruptions. (A. Garces, 2013). The propagation and radiation of infrasound signals from volcanic explosions are adeptly modeled using the Navier-Stokes equations, offering insight into the movement of density and pressure disturbances in fluids at rest. The foundational work by Lighthill (1997), along with significant advancements made by Woulff and McGetchin (1976), set forth a sophisticated solution to the acoustic wave equation applicable to free and half-spaces. This

formulation categorizes the sources of sound radiation into monopoles, associated with mass injections like those seen in volcanic eruptions; dipoles, resulting from force fields without introducing new mass but causing fluid movement; and quadrupoles, representing turbulence within the gas jet from two opposing dipoles. The scaling laws introduced by Woulff and McGetchin, which build upon Lighthill's initial research, link the acoustic power radiated by monopole (Π_m), dipole (Π_d) and quadrupole (Π_q) sources to gas velocity during fluid flow from volcanic vents:

$$monopole: \Pi_m = K_m \frac{\pi R^2 \rho_{air}\, v^4}{c} \tag{2.1}$$

$$dipole: \Pi_d = K_d \frac{\pi R^2 \rho_{air}\, v^6}{c} \tag{2.2}$$

$$quadropole: \Pi_m = K_q \frac{\pi R^2 \rho_{air}\, v^8}{c} \tag{2.3}$$

For a monopole source, which corresponds to the injection of mass such as in a volcanic explosion, the radiated acoustic power (Π_m) is proportional to the vent radius (R) squared, the density of air (ρ_{air}) and the fourth power of the exit velocity at the vent (v), divided by the speed of sound (c). The dipole source, which results from forces applied at the source and characterized by a pair of monopoles oscillating out of phase, has its acoustic power (Π_d) also dependent on the square of the vent radius and the density of air, but the exit velocity is raised to the sixth power. Lastly, the quadrupole source, related to the turbulence within the gas jet, has its radiated acoustic power (Π_q) expressed similarly but with the exit velocity raised to the eighth power. The constants K_m, K_d, and K_q are empirical constants for the monopole, dipole, and quadrupole distributions, respectively. These constants are determined empirically and are used to scale the respective equations to fit observational data or theoretical models.

Kim et al. (2012) recently provided an extensive overview on how the wave equation's integral solution is formulated for monopole and dipole sources in a half-space, a concept based on the Green's Function approach to the Helmholtz wave equation. This review takes into account the context of compact acoustic sources, defined as sources whose sizes are significantly smaller than the acoustic wavelength. Additionally, it considers that infrasound from volcanoes in the far-field is transmitted through a half-space environment, which is essentially an atmosphere with a flat, solid boundary beneath it.

$$p_m(r,t) = \frac{1}{2\pi r}\dot{S}\left(t - \frac{r}{c}\right) \tag{2.4}$$

$$p_h(r,t) = \frac{1}{2\pi r c^2}\left[\frac{x}{r}\ddot{D}_x(t - r/c) + \frac{y}{r}\ddot{D}_y(t - r/c)\right] \tag{2.5}$$

$$p_v(r,t) = \frac{1}{2\pi r c^2}\left[\frac{z}{r}\frac{z_0\partial^3 D_z(t - r/c)}{\partial t^3}\cos\theta\right] \tag{2.6}$$

where p_m, p_h, and p_v are monopole, horizontal and vertical dipole pressure fields, respectively; $\dot{S}$ is the time history of mass acceleration at the source; (x, y, z) are coordinates in a Cartesian system centred at the source location; $D_{x,y,z}$ are dipole momenta (in units of kg m s^{-1}) in the (x, y, z) directions; θ is the angle between the axis of the dipole and the vertical direction; z_0 is source elevation; and c is the sound speed. Any infrasound time series can be written as the radiation from a multipole source, that is the linear superposition of a monopole, with horizontal and vertical dipoles (and terms of higher order):

$$p(r,t) = p_m(r,t) + p_h(r,t) + p_v(r,t) \tag{2.7}$$

Volcanic activity often generates low-frequency (0.01-20 Hz) acoustic waves in the atmosphere, known as infrasound. These sound waves are inaudible and have a low attenuation across extended distances; it maintains its properties intact during transmission. The infrasonic signal travels across short distances in a very uniform environment with no structures that may scatter, attenuate, or reflect acoustic waves, exposing more information about the source (Montalto, 2010). Compared to audible sound waves, this sound constitutes a sizable portion of the acoustic energy released by a volcano.

Infrasound signals from active volcanoes may be produced by a variety of occurrences. When explosive sources cause fluids in a conduit to reverberate acoustically, linear models such as this one implies that sound waves can travel through both magma and the atmosphere. Nonlinear models on the other hand allow harmonic frequencies to not follow a geometric length scale; this may help to explain, for instance, why tremor frequencies are comparable at volcanoes of wildly different sizes (Hagerty and Benites, 2003).

2.2 Anak Krakatau geological setting and eruption history

Anak Krakatau, is a notably active volcano situated along the edge of the caldera created by the monumental 1883 eruption of Krakatau, located in Indonesia's Sunda Strait. The formation of the Krakatau volcanic complex is attributed to arc volcanism, a process initiated by the subduction of the Australian plate under the Sunda plate right beneath Krakatau. As **Figure 2.1** suggested, the thinning of the Earth's crust in this region is a consequence of the extensional faulting and rifting occurring within the Sunda Strait (Dahren *et al.*, 2012). The 1883 eruption of Krakatau was marked by a series of significant explosions driven by magma, leading to devastating pyroclastic flows and a tsunami in the Sunda Strait, as detailed by Self and Rampino in 1981. This event obliterated much of the volcanic structure, resulting in the creation of a caldera, part of which lies underwater. Anak Krakatau started to take shape on the northeastern rim of this caldera during an eruption in 1927 and rose above sea level for the first time in 1928, a phenomenon recorded by Stehn in 1929.

Following a period of 15 months of inactivity, Anak Krakatau re-entered an eruptive phase on June 18, 2018, as evidenced by a surge in seismic events (Global Volcanism Program and Venzke, 2023). The first ash emissions were observed on June 21, 2018, with the first glow of lava visible on July 1, 2018. By September 22, 2018, lava had reached the ocean, and through December 2018, there were continuous reports of explosions, ash emissions, and glowing material (Global Volcanism Program and Venzke, 2023) . A notable escalation in eruption intensity occurred on December 22, 2018, with 423 explosions recorded in a six-hour span from 12:00 to 18:00 local time. That evening, at 21:03 local time, a significant part of the volcano's southwest side and summit collapsed into the sea due to eruptive forces, triggering a tsunami first noticed at 21:27 local time. This tsunami struck the shores of Bantan and Lampung, penetrating as far as 330 meters inland, with the highest surveyed runup reaching 13.5 meters(Muhari *et al.*, 2019).

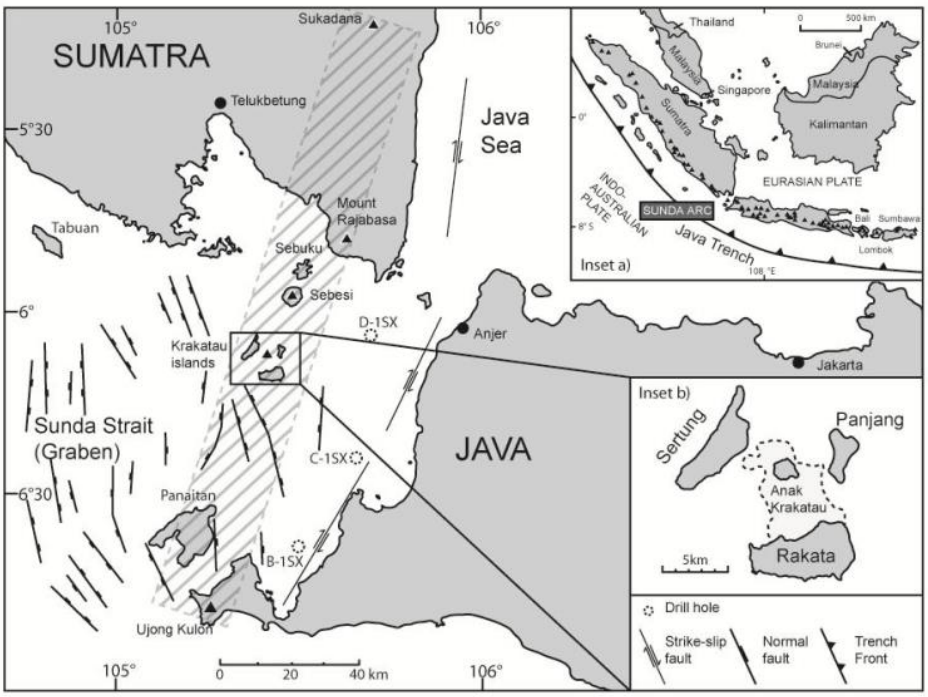


Figure 2.1 Structural map of the Sunda Strait region, showing location of Krakatau complex in the Sunda Strait between Java and Sumatra, and the location of Anak Krakatau taken from Dahren et.al, (2012)

As expected, infrasound signal has provided valuable insights into the eruption's dynamics and the subsequent tsunami of Anak Krakatau's 2018 eruption. According to Rose (2020), The climactic eruption phase of Anak Krakatau volcano that began on December 22, 2018, is distinctly captured in various acoustic data, including waveforms, spectrograms, and PMCC detections at IM I06AU. This phase is marked by a high-amplitude, broadband frequency signal lasting around 7.5 hours on December 22, indicative of intense volcanic activity. Following this, a lower amplitude, low-frequency signal was also recorded, representing the volcano's flank collapse as can be seen on **Figure 2.2 b)**

Rose (2020) has suggested that the continued broadband pulses until December 28 can be identified as Surtseyan activity, likely due to the volcanic vent being submerged under seawater, leading to interactions between water and lava. This period also saw base surges and ash plumes. Additionally, broadband noise visible in the spectrograms is attributed to local surf activity, particularly given the proximity of the recording station I06AU to the Cocos Islands.

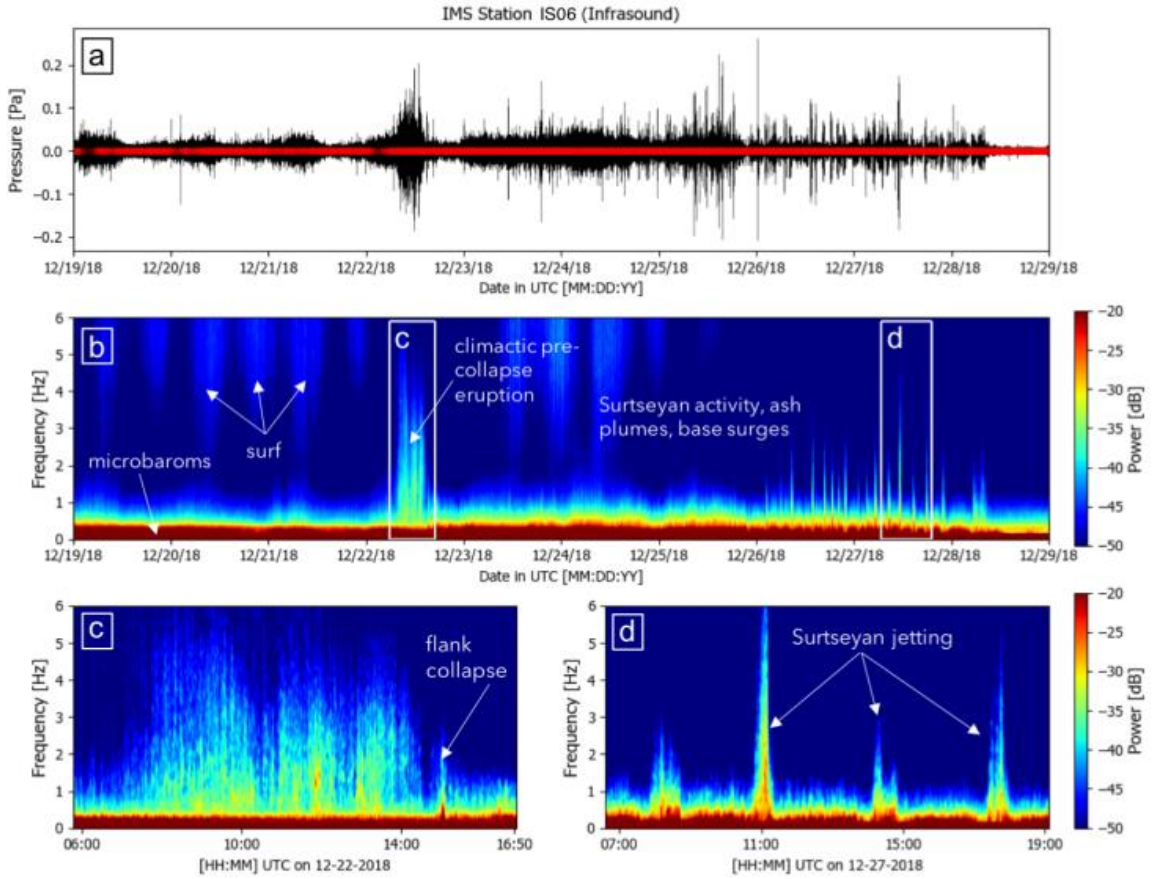


Figure 2.2 Infrasound Station I06AU waveforms for a 10-day period of Anak Krakatau 2018 unrest (Rose, 2020)

On the other hand, Perttu (2020) has observed an infrasound signal begins about eight hours before the main collapse. During this period, two seismic signals indicating minor mass movements and a brief period of calm were detected before the major collapse. The collapse of Anak Krakatau resulted in a tsunami with several waves, with the final wave being the most significant. The collapse led to multiple infrasound signals detected at regional stations. Analysis results of Perttu's (2020) observations can be seen on **Table 2.1**

Table 2.1 Descriptions of the phases during 2019 anak krakatau unrest observed by Perttu et al., (2020)

Phase	*Date/Time UTC (2018-2020)*	*Description of Volcanic Activity*
Phase 0	June 18 - December 21	Initial activity, characterized by Strombolian explosions with a range of intensities, ash plumes less than 5 km and lava flows
Phase 1	December 22 - December 30	Collapse and transition to explosive and Surtseyan style eruptions with high level plumes over 10 km
Phase 1A	December 22 - 00:00-13:50	Intense Strombolian activity, possible lava flow on SW flank, ejected incandescent material reaching the sea
Phase 1B	December 22, 13:50-14:00	Collapse of Anak Krakatau cone and generation of tsunami
Phase 1C	December 22, 14:00-16.55	Propagation and impact of tsunami and three phreatomagmatic eruptions, directly after collapse at 13:55, 15:25, and 15:55
Phase 1D	December 22, 16.55 - December 28, 05:05	Sustained high level plume with intermittent pulsing and SO2 detected
Phase 1E	December 28 - December 30	No sustained plume from satellite and ramp down in geophysical observations
Phase 2	January 2019	After a pause another period of high level plumes and the rebuilding of the island
Phase 3	February 2019 - time of writing	Sporadic eruptions

2.3 Infrasonic signals associated with volcanic eruptions

Each type of volcanic eruption, has its unique infrasonic signature, providing key information for hazard mitigation and understanding volcanic source processes (Fee and Matoza, 2013). Based on their intensities and consequently the height of the ash column, volcanic eruptions can be classified as Strombolian, Vulcanian, Subplinian, and Plinian.

Hawaiian eruptions, characterized by lava fountaining and fissure eruptions, are associated with distinctive infrasonic signatures. The infrasound from these low-level fissure eruptions typically produces a tremor-like signal with broadband spectra, which is somewhat rare. This type of signal is valuable for tracking the movement of eruptive activity along fissures, including the opening of new fissure segments, especially in situations of limited visibility.

Strombolian eruptions, such as those observed at Stromboli Volcano in Italy, are known for producing infrasonic signals of short duration. These explosions produce infrasound signals with high amplitudes, typically ranging from 20 to 80 Pa at a distance of 350 meters from the source (Fee and Matoza, 2013). These signals have short durations of approximately 3 to 5 seconds . The spectral content of these signals is expected to be broad, reflecting the abrupt and intense nature of the explosive events.

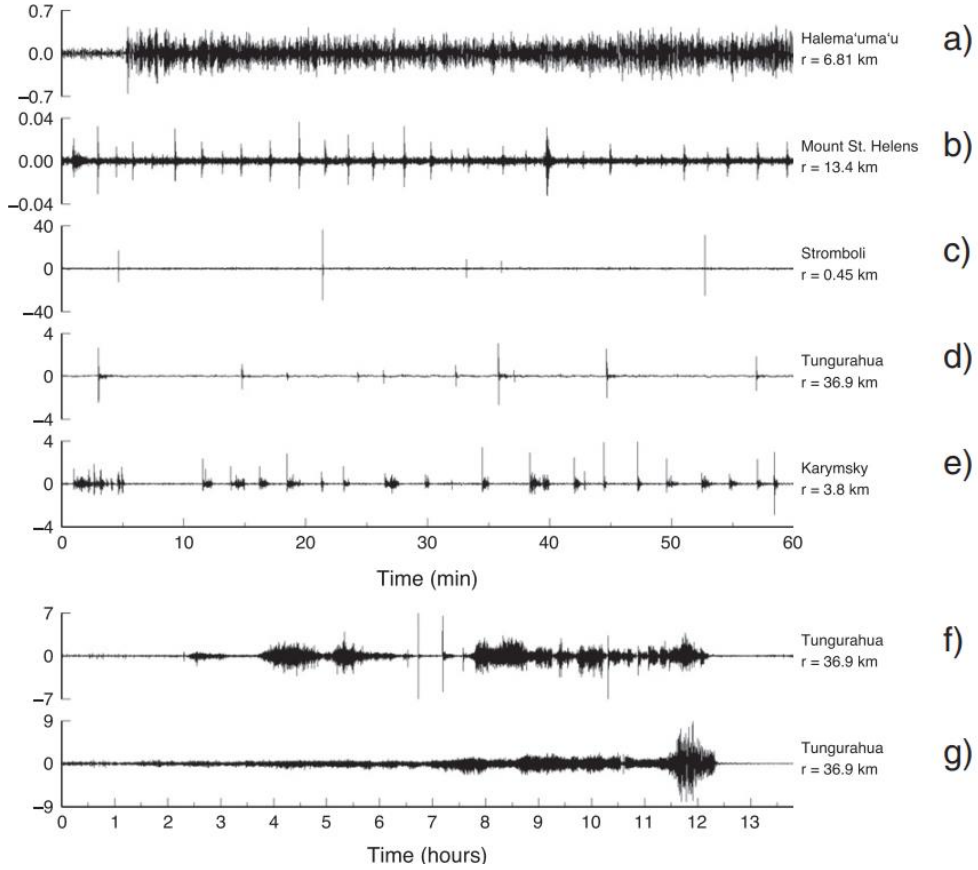


Figure 2.3 Infrasound waveforms from hawaiian to plinian activity a) Harmonic Tremor from Halema'uma'u Vent. b) LP seismic "drumbeat" events at Mount St. Helens. c) Strombolian activity. d) Harmonic tremor. f) Subplinian Eruption from Tunrguruhua Volcano g) Subplinian-Plinian eruption of Tunrguruhua Volcano

2.4 Signal station-level processing

The subject of array processing has generated a vast literature. Several recent books have summarized the principal features of the subject and contain extensive bibliographies (Evers, 2008; Arrowsmith *et al.*, 2009; Dugick, Blom and Webster, 2020). In this section we attempt to review a few of the basic procedures that are in current use within the infrasound community.

2.4.1 Bartlett Beamforming

Bartlett Beamforming is instrumental in pinpointing the directionality of the infrasound signals, As eruption's date and position were known, these details were utilized to determine each station's projected arrival time and back azimuth as well as to locate and isolate the event signal, thereby enhancing the focus on signals originating from the volcano while diminishing background noise. Bartlett beamforming is completed in the frequency domain using a model in which the data on the m^{th} sensor of an array is a combation of a time-shifted coherent signal and local noise:

$$x_m(t) = g(t - \tau_m) + \eta_m(t) \tag{2.8}$$

In the signal model from **Equation 2.8**, $x_m(t)$ represents the signal received at the m^{th} sensor of an array at time t. The function $g(t - \tau_m)$ stands for the coherent signal that has been time-shifted by τ_m, which is the time delay for the signal to reach the m^{th} sensor from the source. The term $\eta_m(t)$ denotes local noise at the m^{th}, which is independent of the coherent signal. The model illustrates how a signal at each sensor consists of both the time-shifted signal from the source and some amount of noise. This vector form represents the same concept but for all sensors simultaneously:

$$\vec{x}(t) = g(t - \vec{\tau}) + \vec{\eta}(t) \tag{2.9}$$

Taking a Fourier transform $\vec{\mathcal{X}}(f)$ of this relation gives the signal model form in the frequency domain,

$$\vec{\mathcal{X}}(f) = \mathcal{G}(f)\vec{\Phi} + \vec{\mathcal{N}}(f), \quad \vec{\Phi}(f, \vec{\tau}) = e^{-2i\pi f\vec{\tau}} \tag{2.10}$$

Where $\mathcal{G}(f)$ is the fourier transform of the coherent signal, $\vec{\mathcal{N}}(f)$ is the transform of the noise, and $\vec{\Phi}$ is a vector of phasors often termed the "steering vector" related to time delays at frequency f.

For a plane wave, the time delays in the steering vector are determined by considering two key aspects: the azimuth of the direction-of-arrival, symbolized as φ, and the trace velocity, denoted as v_{tr}. This trace velocity, which is the speed at which sound moves across the ground, is calculated based on the ambient sound speed, c_0, and the inclination angle of the plane wave, represented by ϑ. The formula for trace velocity is $v_{\text{tr}} = \frac{c_0}{\cos\vartheta}$. These factors together establish the

slowness, represented as $\vec{s}$, of the plane wave. By integrating this slowness with the positions of the array elements, denoted as $\vec{z_j}$, the time delays for the arrival of the wave at these elements can be accurately defined.

$$\tau_j = \vec{s} \cdot \vec{z_j}, \quad \vec{s} = \left(\frac{\sin\varphi}{v_{\text{tr}}}, \frac{\cos\varphi}{v_{\text{tr}}}\right) \tag{2.11}$$

The ordinary least squares estimate for the coherent signal is more commonly termed the Bartlett beam and can be defined as,

$$\hat{\mathcal{G}}_{\text{Bartlett}}(f, \vec{s}) = \frac{\overrightarrow{\mathcal{X}^{\dagger}}\vec{\Phi}}{\overrightarrow{\Phi^{\dagger}}\vec{\Phi}} \tag{2.12}$$

Where $\overrightarrow{\mathcal{X}^{\dagger}}$ and $\overrightarrow{\Phi^{\dagger}}$ signify the conjugate transpose (Hermitian transpose) of $\vec{\mathcal{X}}$ and $\vec{\Phi}$ respectively. The estimated beam power is often expressed as the square of the estimated signal amplitude $\mathcal{P}_{\text{Brlt}}(f, \vec{s}) = \hat{\mathcal{G}}^2_{\text{Bartlett}}(f, \vec{s})$. For Bartlett method, a signal covariance matrix is computed from the outer product of the signal vector, $\boldsymbol{S_X} = \frac{1}{J}\sum_j \vec{\mathcal{X}_j}\,\vec{\mathcal{X}_j^{\dagger}}$ and that matrix is used in analysis. In order to utilize these methods, multiple sub-windows are needed to ensure that the estimated covariance matrix is full rank for inversion or eigen-decomposition; however, a whitening step is included to ensure these options are possible. From the above definition, it's clear that the Bartlett beam can be defined from the signal covariance as,

$$\hat{\mathcal{P}}_{\text{Bartlett}}(f, \vec{s}) = \left|\frac{\overrightarrow{\Phi^{\dagger}}\boldsymbol{S_X}\vec{\Phi}}{\overrightarrow{\Phi^{\dagger}}\vec{\Phi}}\right|^{\frac{1}{2}} \tag{2.13}$$

2.4.2 Adaptive F-Detector

The basis for the detection algorithm used in this study is the F-statistic (e.g. Shumway 1971; Smart 1971; Smart & Flinn 1971; Blandford 1974; Evers & Haak 2001). Following Blandford (1974), the F-statistic is defined as the ratio of the power on the beam versus the residual power, as follows:

$$F = \left(\frac{J-1}{J}\right)\frac{\sum_{n=n_0}^{n_0+(N-1)}\left[\sum_{j=1}^{J} x_j(n+l_j)\right]^2}{\sum_{n=n_0}^{n_0+(N-1)}\left(\sum_{j=1}^{J}\left\{x_j(n+l_j) - \left[\frac{1}{J}\sum_{m=1}^{J} x_m(n+l_m)\right]\right\}^2\right)} \tag{2.14}$$

The variable J represents the total count of sensors, and $x_j(n)$ signifies the amplitude of the nth sample in the zero-mean time series from the j sensor. The term l_j refers to the time-alignment delay determined through beamforming. The processing interval begins at the sample index n_0, and the processing window encompasses N samples.

As demonstrated by Shumway et al. (1999), AFD accounts for temporal changes in noise by applying an adaptive window to update the detection distribution, which makes a distinction between the signal and correlated noise, the distribution of the F-statistic aligns with $cF_{2BT,2BT(N-1)}$. Here, B indicates the bandwidth of the filtered data within the processing window denoted by T, and N is the count of elements in the array. The constant c is defined as follows

$$c = \left(1 + N\frac{P_s}{P_n}\right) \tag{2.15}$$

with $\frac{P_s}{P_n}$ denoting the correlated-noise power to uncorrelated noise power ratio (Shumway et al., 1999).

2.5 Network Level Processing

2.5.1 Association

A pair-based, joint-likelihood association approach is developed that identifies events by computing the probability that individual detection pairs are attributable to a hypothetical common source and applying hierarchical clustering to identify events from the pair-based analysis. The framework is based on a Bayesian formulation introduced for infrasonic source localization and utilizes the propagation models developed for that application with modifications to improve the numerical efficiency of the analysis (Blom *et al.*, 2020).

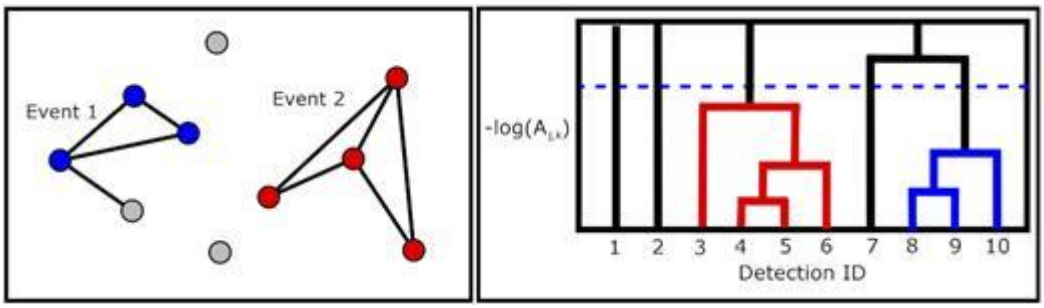


Figure 2.4 Cluster-based analysis of binary linkages (left-hand panel) and hierarchical analysis of linkages (right-hand panel) for pair-based association

2.5.2 Localization

Event analysis using the Bayesian Infrasonic Source Localization (BISL) methodology to estimate both the spatial location of the event as well as the origin time with quantified

uncertainty (Modrak, Arrowsmith and Anderson, 2010). Preliminary analysis of the back projections can be used to define the spatial region of interest or it can be specified by the analysis. Analysis identifies the maximum posteriori solution

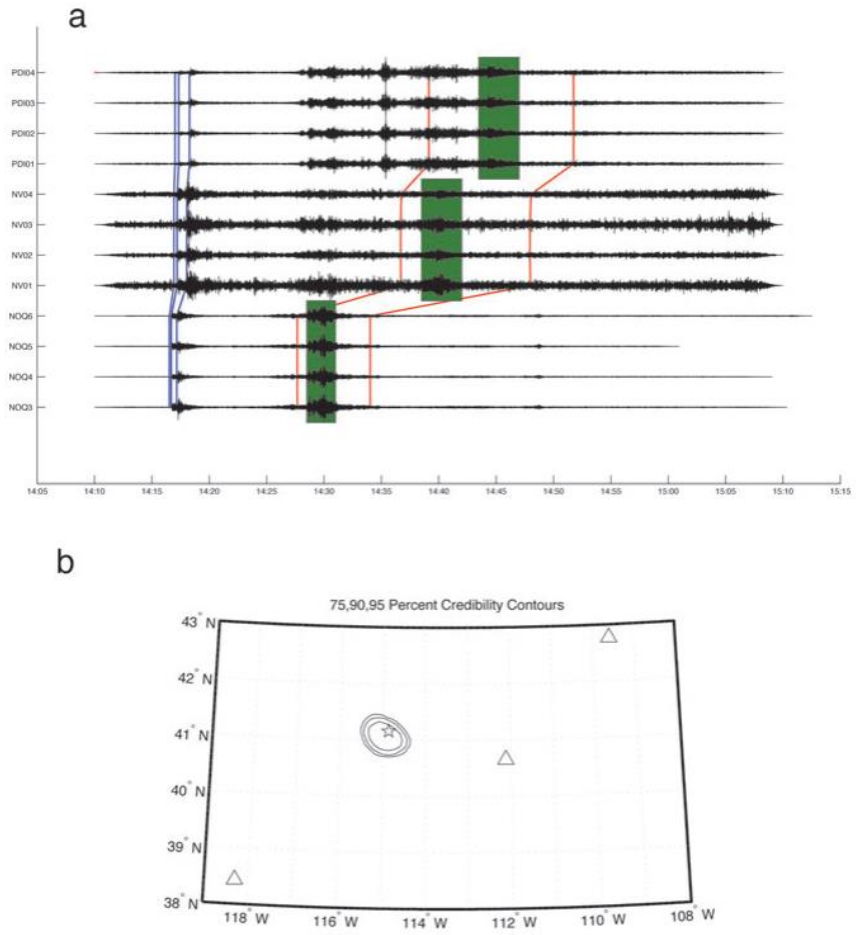


Figure 2.5 Results from applying the algorithm to infrasonic observations of the 2008 Wells, Nevada earthquake from (Modrak, Arrowsmith and Anderson, 2010) **(a)** : Infrasonic observations at each array, where the green boxes highlight the arrivals used in estimating source location, blue lines represent predicted arrival times of seismic phases (P_n, P_g and L_g), and red lines denote the earliest and latest possible arrival times of epicentral infrasound arrivals. **(b)**: 75, 90 and 95 per cent credibility contours obtained from the algorithm. The seismically-derived location solution is shown by a star (associated seismic localization uncertainties and the BISL posterior mode are not depicted)

2.6 Laslo's Best Beam

Smart and Flinn (1971) identified that the frequency-wavenumber spectra can be computed using a singular summation. This involves a relationship where $k = \omega\vec{p}$, linking the slowness vector $\vec{p}$ to the wavenumber $\vec{k}$ at a given frequency ω. Through this relation, the frequency-wavenumber spectrum is transformed into a frequency-slowness power spectrum:

$$P(\vec{\omega},\vec{p}) = \left|\sum_{n=1}^{N} G(\omega,\overrightarrow{r_n}) \cdot e^{-i\,\omega\vec{p}\cdot\overrightarrow{r_n}}\right|^2 \tag{2.16}$$

In equation, $G(\omega,\overrightarrow{r_n})$ represents the Fourier transforms of the recordings from N instruments, each located at position vectors denoted by $\overrightarrow{r_n}$. The power spectrum $G(\omega,\overrightarrow{r_n})$ is essentially a collection of beams that sample the atmospheric conditions above the array, as noted by Haak [1996]. The orientation and characteristics of these beams are governed by the slowness vector $\vec{p}$. When these beams detect coherent energy, indicative of energy emanating from a specific direction in the atmosphere, there is an increase in the power of the spectrum. The most effective beam in this scenario is identified by its distinctive values of apparent sound speed, $\overrightarrow{r_n}$, and back azimuth φ:

$$|\vec{p}| = \sqrt{p_x^2 + p_y^2} = \sqrt{\left(\frac{cos\ \phi}{c_{app}}\right)^2 + \left(\frac{sin\ \phi}{c_{app}}\right)^2} = \frac{1}{c_{app}} \tag{2.17}$$

The Fisher ratio (F) is defined as

$$P_s(\omega,\vec{p}) = \frac{P_S(\omega,\vec{p})}{P_T(\omega) - P_S(\omega,\vec{p})} \cdot (N-1) \tag{2.18}$$

Where

$$P_s(\omega,\vec{p}) = \left|\frac{1}{N}\sum_{n=1}^{N} G(\omega,\overrightarrow{r_n}) \cdot e^{-i\omega\vec{p}\overrightarrow{r_n}}\right|^2 \tag{2.19}$$

$$P_{T(\omega)} = \frac{1}{N}\sum_{n=1}^{N} |G(\omega,\overrightarrow{r_n})|^2 \tag{2.20}$$

$P_s(\omega,\vec{p})$ signifies the portion of the spectrum's energy attributable to the signal, which is derived from the power spectrum $P_s(\omega,\vec{p})$. $P_s(\omega,\vec{p})$ encompasses the entire spectral power, accounting for both the signal and noise components. Therefore, in the absence of the $(N-1)$ weighting factor, F essentially acts as an indicator of the signal-to-noise power ratio. The presence of a coherent infrasonic wave passing over the array will cause an increase in F. On the other hand, incoherent noises or minor scale coherent wind disturbances tend to yield lower F values, usually ranging between one and five. For our use in this project, we are employing eight traces from arrays of a station to compute the best beam waveform.

2.7 Deep Embedded Clustering (DEC)

Deep Embedded Clustering (DEC) is an advanced clustering technique that effectively combines deep learning, specifically autoencoders, with traditional clustering methods. The

process begins with training a deep autoencoder on the dataset. An autoencoder is a neural network designed to learn a compressed, lower-dimensional representation of the input data in its latent space. DEC employs autoencoders for nonlinear dimensionality reduction, enabling improved representation capabilities. This combination facilitates the extraction of useful features from high-dimensional data, which is essential for effective clustering (Huang, Hu and Lin, 2023).

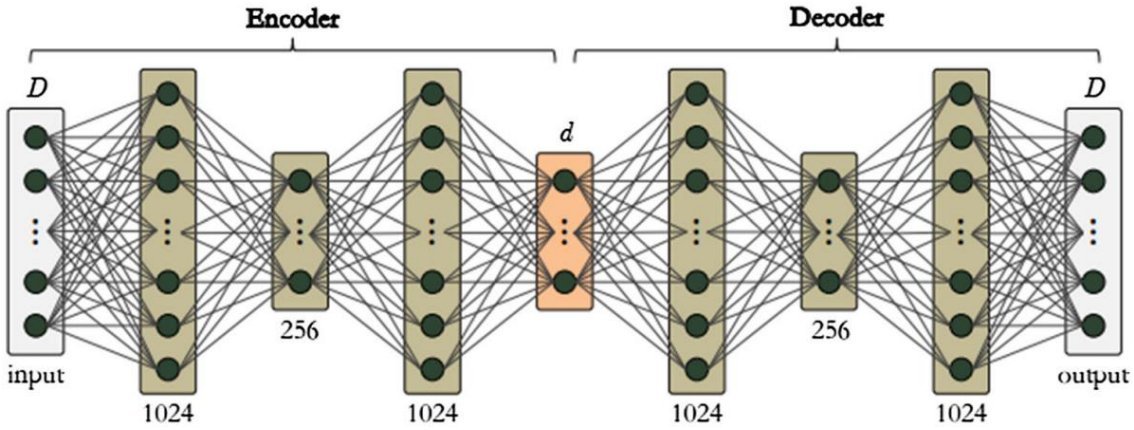


Figure 2.6 The architecture of embedded autoencoder taken from Huang et. al.,(2023)

DEC has shown significant potential in enhancing the understanding of volcanic activity. For instance, DEC could analyze the time-dependent structure of Earth's volcanic eruptions, identifying patterns like episodic material discharge and order clustering of events (Gusev, 2014). Additionally, Mezyk and Malinowski (2021) used deep embedded K-means clustering, a variant of DEC, for interpreting complex geological structures in seismic profiles, which is vital for volcanic studies. Its application extends to analyzing seismic data for identifying impulsive seismicity types and their source mechanisms, as shown by Jenkins et al. (2021).

After training the autoencoder to capture the reduced representations of the data, the next step is to initialize cluster centers. This is typically done using the K-means algorithm to partition the latent space into distinct regions. Deep Embedded Clustering (DEC) then optimizes the autoencoder's parameters and cluster assignments in tandem. This joint optimization, which involves minimizing a loss function that measures the proximity of data points to cluster centers, refines both data representation and cluster quality. The DEC approach leverages the latent representations to cluster complex, high-dimensional data effectively, as detailed by Mousavi et al. in (2020) and Xie et al. in (2016). By doing so, it reveals more meaningful patterns in the data, such as the STFT features from seismic signals, thereby

providing a deeper understanding of volcanic activity. The fine-tuning phase employs Student's t-distribution to measure the similarity between each data point and the cluster centroids, enhancing the model's clustering performance

In the initial phase of training, the autoencoder is taught to accurately reproduce its input from the latent space representations. Following this, the features obtained from the autoencoder's bottleneck are utilized for clustering through the k-means method. This process divides the latent space into 'k' distinct clusters, each identified by a central point, referred to as the cluster centroid (μ_j). Subsequently, in the fine-tuning phase, the model is concurrently trained to both represent features effectively and assign them to specific clusters. This involves measuring the resemblance between each point in the embedded space z and the cluster centroids μ_j, using a method known as Student's t-distribution.

$$q_{ij} = \frac{\left(1 + |z_i||\mu_j|^2\right)^{-1}}{\sum_j \left(1 + |z_i||\mu_j|^2\right)^{-1}} \tag{2.21}$$

q_{ij} represents the likelihood of associating a given sample 'i' with a cluster 'j', leading to a collection of probabilistic class assignments. To further refine this, an additional target distribution, denoted as p_{ij}, is computed. This is achieved by leveraging the membership probabilities represented by q_{ij}, as described in the following manner:

$$p_{ij} = \frac{\frac{q_{ij}^2}{\sum_i q_{ij}}}{\sum_j \left(\frac{q_{ij}^2}{\sum_i q_{ij}}\right)} \tag{2.22}$$

In the fine-tuning stage, the clustering layer aims to reduce the Kullback–Leibler (KL) divergence. This process is focused on aligning the soft assignments, represented by q.2, more closely with the target distribution, denoted as p_{ij}.

$$L = KL(P||Q) = \sum_i \sum_i p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right) \tag{2.23}$$

q_{ij} indicating the probability that each embedded point belongs to a particular cluster, establishes the certainty of these cluster assignments. The auxiliary target distribution p_{ij} adjusts the impact of each centroid in the loss function, giving more weight to samples where cluster assignment confidence is higher. Consequently, the network predominantly learns from assignments made with high confidence, enhancing the precision of cluster centroids by

reducing the discrepancy between q_{ij} and p_{ij}. During the iterative fine-tuning process, not only are the cluster centroids refined, but the autoencoder's weights are also updated. This leads to the development of more effective clustering features, thereby improving overall clustering performance.

2.8 Time-To-Event eruption forecasting

Eruption forecasting is a vital aspect of volcanology, aiming to predict hazardous volcanic activity to inform risk mitigation strategies. Recent advancements have shown promise in improving the utility and accuracy of eruption forecasts, notably through the increased volume and quality of multidisciplinary monitoring data, advancements in computing power, and machine learning algorithms. These innovations, when utilized within integrative and fully probabilistic forecasting frameworks, could substantially enhance the capability of volcanologists to issue timely warnings (Poland and Anderson, 2020).

The Time-To-Event (TTE) model for volcanic eruption forecasting is a probabilistic approach that aims to estimate the timing and likelihood of future volcanic events. This model integrates geophysical data and statistical methods to assess the probability of an eruption within a given time frame. For instance, the Bayesian Event Tree model for eruption forecasting (BET-EF) is a versatile tool that combines theoretical models, prior beliefs, monitoring measures, and historical data to provide probabilities of specific volcanic events (Marzocchi, Sandri and Selva, 2008)

Gradient boosted trees (GBT) regression, a powerful machine learning technique, can be adapted for TTE modeling in volcanic activity. GBT develops predictive models through iterative refinement, where each new model is trained to improve upon the errors of the previous ones. This method is known for its ability to handle complex nonlinear relationships and provide high predictive accuracy. It's particularly useful when dealing with the multifaceted and intricate nature of volcanic data, where various factors can influence the timing and magnitude of an eruption.

LightGBM optimizes traditional gradient boosting methods by constructing trees leaf-wise rather than level-wise, enhancing efficiency and accuracy. In gradient tree boosting, the model improves iteratively by adding trees that learn from the errors of preceding ones. At each stage, the model adds a new tree, which we'll call Tree t, that aims to correct the residuals—differences between observed and predicted values—left by the previous t-1 trees. This process

is depicted in **Figure 2.4**, where the data is initially fitted by Tree 1, followed by the residual fitting in Tree 2, and so forth, until the t-th tree.

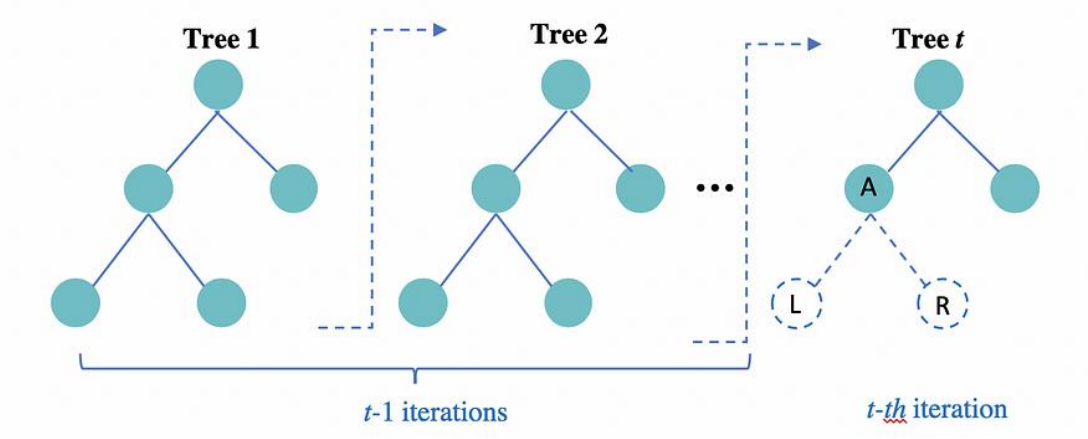


Figure 2.7 Gradient Boosted Trees process

Once the autoencoder is trained, the encoded features h are used as inputs for the LightGBM model. The LightGBM then follows these mathematical steps

$$F_0(h) = arg\,\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma) \tag{2.24}$$

equation represents the initialization of the model. $F_0(h)$ is the initial prediction for all instances in the dataset, often taken as the mean of the target variable. It minimizes the loss function L over all data points i from 1 to N, where y_i is the actual "time to eruption" and γ is a constant value.

$$r_{im} = -\left[\frac{\partial L\big(y_i, F(h_i)\big)}{\partial F(h_i)}\right]_{F(h)=F_{m-1}(h)} \tag{2.25}$$

This is the computation of the pseudo-residuals or the negative gradient of the loss function. $\frac{\partial L}{\partial F}$ is the partial derivative of the loss function with respect to the model's prediction at a particular data point. It's calculated for each instance i at each iteration m. $F_{m-1}(h)$ is the model's prediction at the previous iteration, and h_i is the encoded feature vector for instance i.

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{N} L\big(y_i, F_{m-1}(h_i) + \gamma h_m(h_i)\big) \tag{2.26}$$

After computing the pseudo-residuals, a new tree $h_m(h)$ is fitted to these residuals. Then, the best multiplier γ_m for the tree's predictions is determined by minimizing the loss function.

$$F_{m(h)} = F_{\{m-1\}(h)} + \nu\gamma_m h_{m(h)} \tag{2.27}$$

This updates the model by adding the new tree weighted by the step size v and the best multiplier γ_m . v is the learning rate that controls the contribution of each tree to the final model, preventing overfitting by shrinking the tree's impact. The final prediction $\hat{y}$ for "time to eruption" is the sum of the initial model and the contributions from all M trees, each weighted by its respective γ_m and scaled by the learning rate ν.

$$\hat{y}(h) = F_M(h) = F_0(h) + \nu \sum_{m=1}^{M} \gamma_m h_m(h) \tag{2.28}$$

These equations collectively represent the mathematical foundation of the LightGBM regression for forecasting the "time to eruption". The iterative nature of boosting allows each new tree to correct the errors made by the ensemble of all previous trees, thus improving the model's predictions with each iteration

CHAPTER III
RESEARCH METHODS

This chapter will further explain the resources needed to perform this research as well as the methodology required to conduct the research. The methodology will be explained in depth from the, data collection, data pre-processing and cleaning, model preparation, model training, model testing and validation, and finally data visualization.

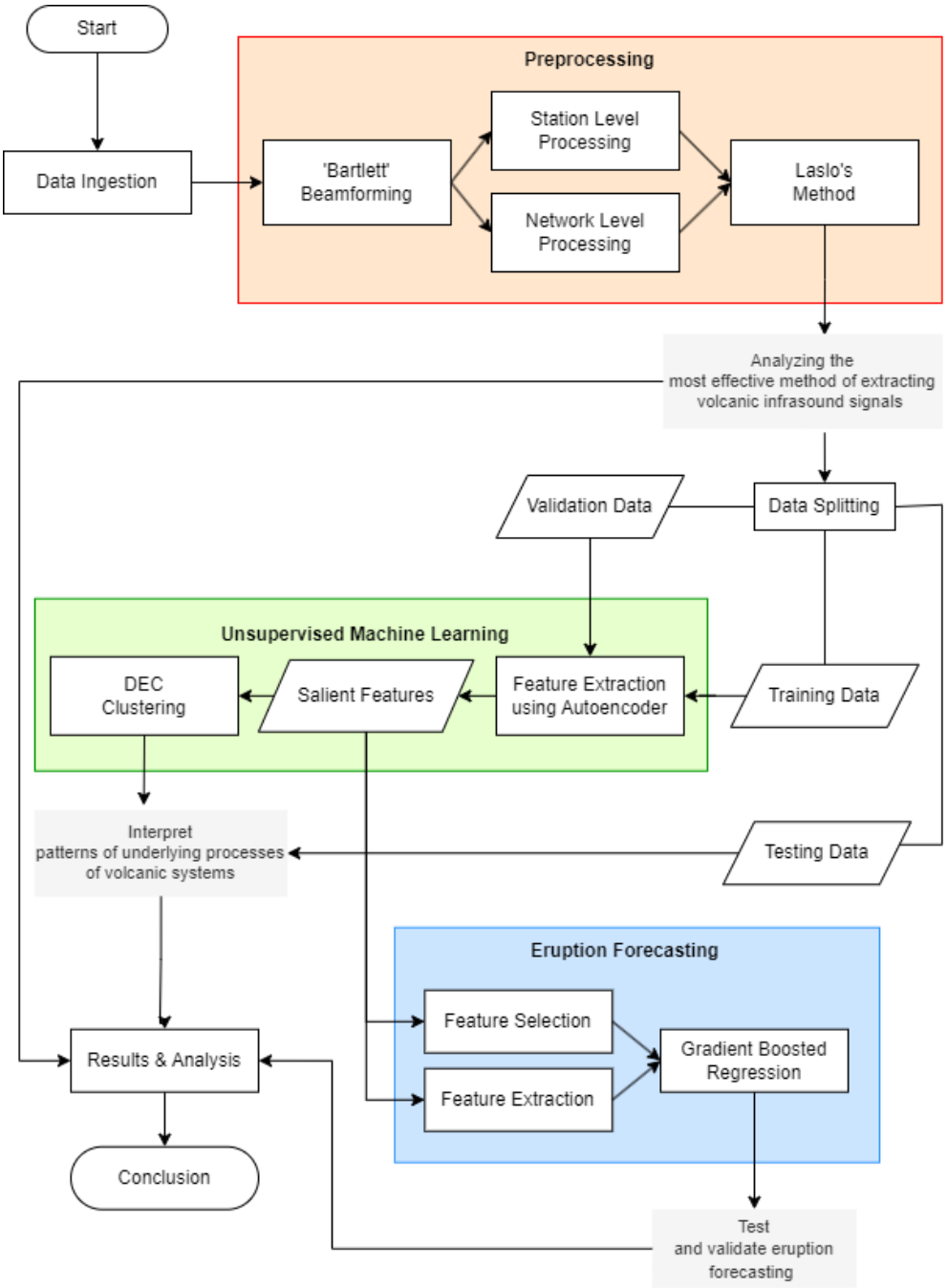


Figure 3.1 Research Flowchart

3.1 Data Ingestion

Data collection refers to the process of acquiring and importing data for further cleaning, analysis, or storage in a database. We will gather Infrasound signals from Anak Karakatau Volcano from 106AU, 152GB, 107AU International Monitoring System (IMS) Infrasound Arrays through IRIS Data Services (http://ds.iris.edu/gmap/) as seen on **Table 3.1.** The data gathered will range from 24th of July, 2018 until the 1st of September, 2019; covering unrest phases of 2018 eruption as explained on **Table 2.1.**

Table 3.1 Infrasound station placement relative to Anak Krakatau Volcano

Array	**Sensors**	**Anak Krakatau**	
		Distance (km)	*Back-azimuth (°)*
106AU	8	1158	55
107AU	8	3479	-67
152GB	7	3639	94

Figure 3.2 Plot of Stations

We harnessed the robust capabilities of Obspy, a powerful Python library, to retrieve and process seismic data from the comprehensive IRIS Data Services. The data, originating from an array of monitoring stations, was meticulously selected to ensure the highest fidelity in our analysis. The example depicted in **Figure 3.3** showcases the raw daily data of 20Hz sampling rate filtered within a frequency range of 0.7 to 4 Hz, which is a critical band for capturing the low-frequency signals often associated with volcanic activity (Rose, 2020). By applying precise bandpass filters after acquiring the raw waveforms, we isolated the essential frequencies that are most indicative of seismo-acoustic phenomena. This preprocessing step is

crucial for enhancing the signal-to-noise ratio and thus preparing the data for subsequent machine learning tasks aimed at recognizing patterns indicative of volcanic unrest.

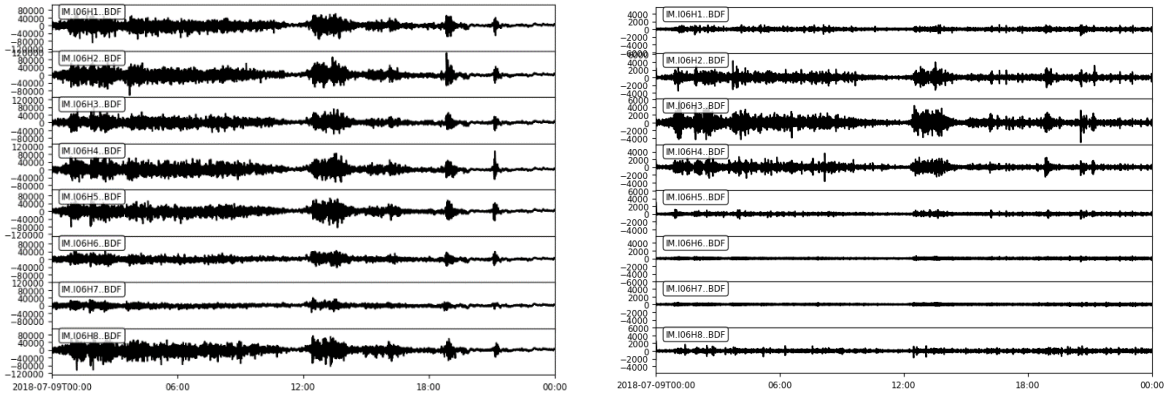


Figure 3.3 Waveform infestion results for I06AU station from 2018-07-09 (right) Raw 20 Hz sampling waveform sample (left) Bandpass-filtered waveform sample

3.2 Preprocessing

The preprocessing component of the methodology serves as a critical preliminary step in the monitoring seismo-acoustic activity in Anak Krakatau volcano. Using Infrapy python based infrasound processing toolkit, which includes an array based beamformer with location and association capabilities, along with a spectral single-sensor detector (Dugick, Blom and Webster, 2020), we employed station-level processing techniques, specifically Bartlett Beamforming, Adaptive F-Detector, to identify and isolate potential infrasonic events indicative of volcanic activity.

Table 3.2 Parameters for Bartlett Beamforming

Parameters	*Value*
Minimum Frequency	0.7 Hz
Maximum Frequency	4 Hz
Minimum Backazimuth	50˚
Maximum backazimuth	60˚
Window length	600 s
Window step	30 s

Conventional beamforming methods (Bartlett, Capon) separate coherent and incoherent parts of a signal through the assumption of planar waves arriving at the array. A signal

backazimuth and slowness can be estimated as signals are shifted to account for travel. time differentials across array elements, bringing the signal into phase across as the noise deconstructively cancels out. In the classical, or Bartlett methodology , data records on each array element are time-shifted versions of the other with local noise (Dugick, Blom and Webster, 2020), The usage in our project we will be employing Bartlett beamforming with specifications as revealed on **Table 3.2.** The parameters our beamforming are predetermined from previous observation by Rose (2020) and Gheri et al. (2023) with the results of a daily recording shown in **Figure 3.4**

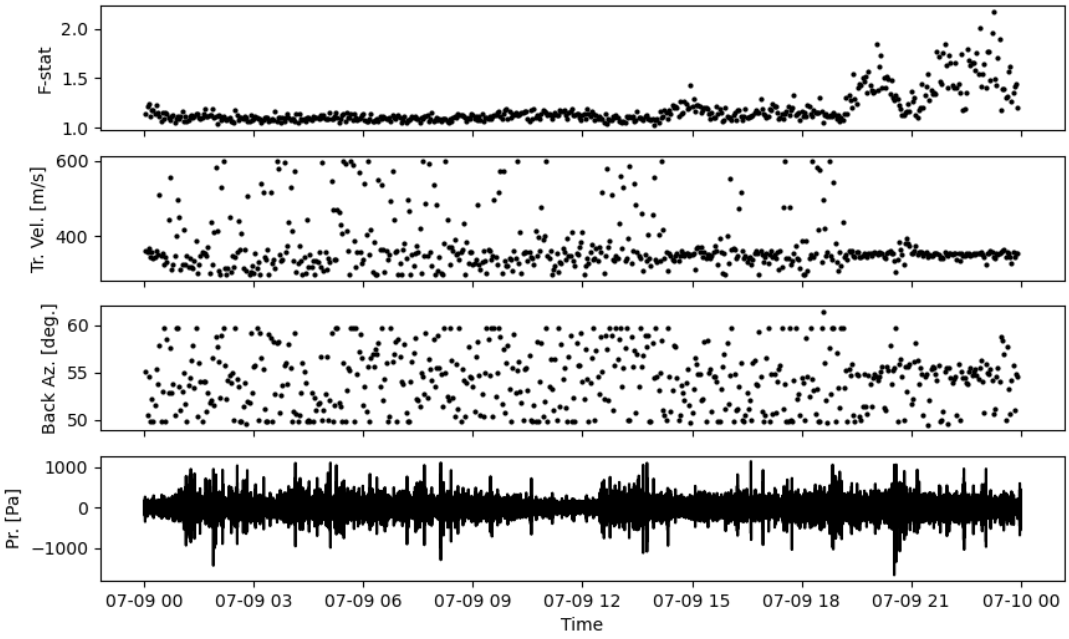


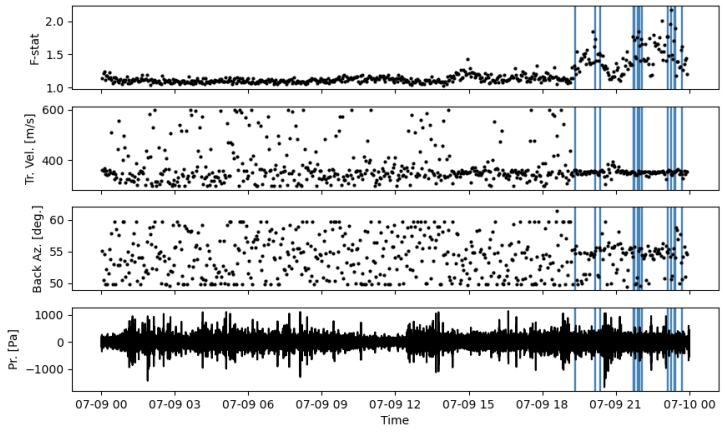
Figure 3.4 Example of beamforming results for I06AU station from 2018-07-09

InfraPy uses an Adaptive Fisher-Detector (AFD) for identifying detections in the beamforming results. The AFD accounts for both correlated and uncorrelated noise through an increase in the value of the F-statistic required to declare a detection based upon background F-values being elevated from coherent or persistent noise sources (Arrowsmith *et al.*, 2009). Using the same parameters used by Evers (2008) on Mt.Etna, the summary of parameters used for our adaptive F-detector can be seen on **Table 3.3**

Table 3.3 Parameters for Adaptive F Detector

Parameters	*Value*
Minimum Frequency	0.7 Hz
Maximum Frequency	4 Hz
P-Value	0.05
Backazimuth Width	5
Minimum Duration	25 s
Merge Detection	False

The output generated this process, results in a set of unlabeled infrasound data segments. These segments are the product of meticulous signal processing and are selected based on their potential relevance to volcanic activity, yet without predefined labels. The absence of labels reflects the unsupervised nature of the subsequent analysis—there are no predetermined categories or outcomes associated with each segment. The subsequent VONA notification (Global Volcanism Program and Venzke, 2023) doesn't provide enough information to label the whole segment results.



No	F-Stat	Trace Vel. (m/s)	Back Azimuth
1	1.3258	361.25	55.31
2	1.7313	356.91	55.08
3	1.5221	357.23	55.64
4	1.7678	354.42	54.2
5	1.8481	358.55	55.06
6	1.7308	354.53	55.47
7	1.7746	354.79	54.97
8	2.1751	345.58	50.63
9	1.893	355.23	53.97
10	1.6124	345.38	50.65

Figure 3.5 Example of Adaptive F Detector results for I06AU station from 2018-07-09

Best beam is later utilized to each of the isolated segments from the adaptive F-detector. This applies an adaptive delay-and-sum analysis within each analysis window using the beartlett beamforming results and uses an envelope at window edges to smoothly transition

between (Evers, 2008). For our use in this project, we are employing eight traces from arrays of a station to compute the best beam waveform as seen on the example **Figure 3.4**

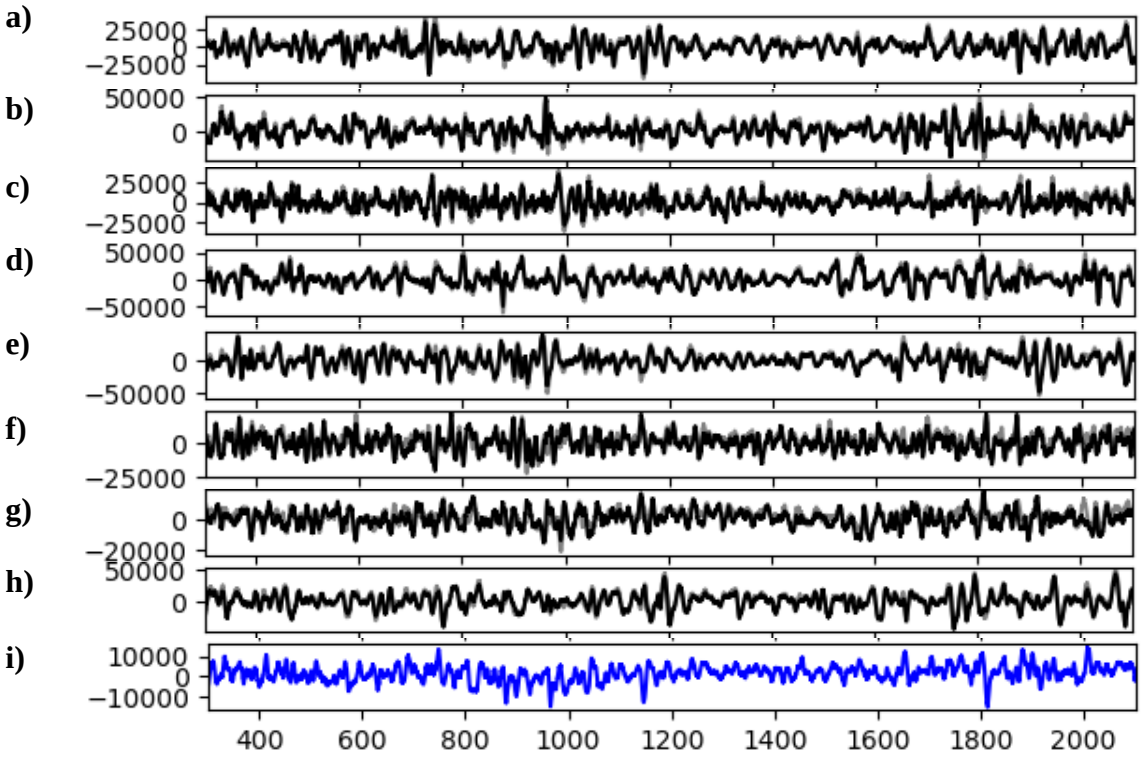


Figure 3.6 Best-beam results using Laslo's method for I06AU station arrays from 2018-07-09. a) b) c) d) e) f) g) h) represents raw waveforms results I06H1, I06H2, I06H3, I06H4, I06H5, I06H6, I06H7, and I06H8 with grey lines indicating best-beam result for each array. Row i) represents isolated Laslo's best beam results.

3.3 Data Splitting

In the field of machine learning and data analysis, data splitting is a crucial step to evaluate the performance of predictive models. Observing the provided image detailing the phases of volcanic activity as on **Table 2.1**, it's clear that the dataset encompasses various stages of the Anak Krakatau eruption from initial activity to sporadic eruptions, each with distinct characteristics. The dataset for the Mount Krakatau is inititally temporally split based on the specific date ranges that correspond to distinct volcanic phases as represented by **Figure 3.5.**

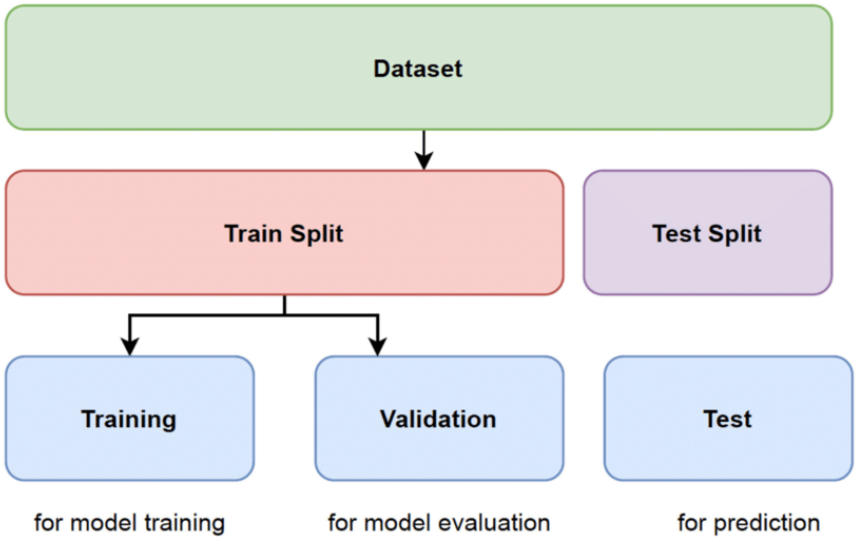


Figure 3.7 Data Splitting visualization

The training split, which ranges from Phase 0 to Phase 2 (June 2019 - January 2019) of the unrest represents the training split, comprising 2329 data points and encompasses the initial stages of volcanic activity, including Strombolian explosions and lava flows, all the way through to more intense activities such as collapse transitions, high-level plumes, and pauses in activity. The training split is then further randomly divided where 80% are used for the actual training of the model, allowing the model to identify and learn the patterns in the data and the remaining 20% are set aside for validation, which serves to evaluate the model's learning and adjust its parameters without exposing it to the testing split. The testing split, which represented Phase 3 (February – August 2019) of the unrest is consisted of 211 data points. It covers sporadic eruptions that occurred after the training period. This phase has similarities with phase 0 where there is an interplay of low, moderate and higher activities of the volcano, therefore a good choice to test the performance of our model. This phase is crucial as it tests the model's ability to generalize from the patterns it learned during training to make accurate predictions on new, unseen data.

3.4 Unsupervised Machine Learning

With the absence of predetermined categories or outcomes associated with each segment, the data is prepared for unsupervised machine learning techniques, such as clustering, which will attempt to uncover inherent structures or patterns within the data. These patterns might correspond to different states of volcanic activity, but without the explicit labels that supervised learning would require. The goal is to discover natural groupings within the data that could

signify meaningful distinctions, like the presence of volcanic tremors or other signals of interest.

3.4.1 Feature Extraction using Convolutional Autoencoder

Feature extraction stands as a cornerstone within the unsupervised machine learning framework, serving as the conduit through which raw infrasound signals are translated into a comprehensive set of descriptors. To understand the process of seismo-acoustic signals from volcanic activity, it's important to start with the data preparation stage, where Short-Time Fourier Transform (STFT) features are extracted. STFT is a powerful technique for analyzing the frequency content of signals over time, it is previously used by Zali (2023) to extract complex, time-varying signals for seismic data from 2021 Mount Geldingadalir eruption. We adapted the usage for our seismo-acoustic dataset.

Clustering emerges as a powerful solution for interpreting this unlabeled data. As a branch of unsupervised learning, clustering algorithms like k-means, introduced by MacQueen et al. (1967), organize data into groups based on similarity. However, the high-dimensional nature of STFT features poses a challenge for traditional clustering methods, which tend to become less effective as data dimensionality increases (Steinbach, Ertöz and Kumar, 2004). To overcome this, an autoencoder, a form of deep neural network, is utilized for dimensionality reduction. The autoencoder learns to condense the high-dimensional data into a more manageable, lower-dimensional representation. This process not only simplifies the data but also helps in retaining the most significant features for clustering.

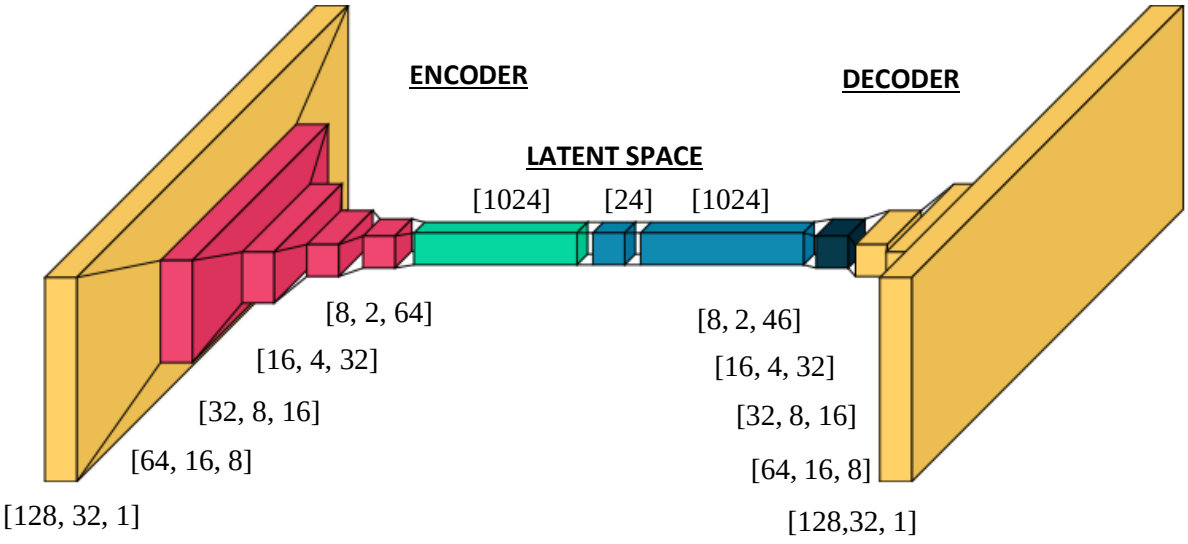


Figure 3.8 VisualKeras representation of the Autoencoder architecture

Table 3.4 Autoencoder Architecture

Layer Name	Type	Filters	Kernel Size	Stride	Activation	Output Shape	Trainable Parameters
Input	-		-	-	-	[128,32, 1]	0
Conv2D_1	Convolution	8	[7, 5]	[2, 2]	ELU	[64, 16, 8]	288
Conv2D_2	Convolution	16	[5, 3]	[2, 2]	ELU	[32, 8, 16]	1936
Conv2D_3	Convolution	32	[5, 3]	[2, 2]	ELU	[16, 4, 32]	7712
Conv2D_4	Convolution	64	[5, 3]	[2, 2]	ELU	[8, 2, 64]	30784
Flat						[1024]	0
Encoded					ELU	[24]	24600
FC					ELU	[1024]	25600
Reshape						[8, 2, 46]	0
ConvT_1	Transposed Convolution	32	[5, 3]	[2, 2]	ELU	[16, 4, 32]	30752
ConvT_2	Transposed Convolution		[5, 3]	[2, 2]	ELU	[32, 8, 16]	7696
ConvT_3	Transposed Convolution		[5, 3]	[2, 2]	ELU	[64, 16, 8]	1928
Decoded	Transposed Convolution		[7, 5]	[2, 2]	ELU	[128,32, 1]	281

As displayed on **Figure 3.5** and **Table 3.4,** the input to the autoencoder is the result of applying Short-Time Fourier Transform (STFT) to the waveform, represented as a 3D array with dimensions 128x32 and a single channel depth [128, 32, 1]. The data first passes through a series of convolutional layers. These layers apply filters to the input to create feature maps that represent different aspects of the data. The encoder has four convolutional layers with an increasing number of filters: 8, 16, 32, and 64. The kernel size remains consistent at [5, 3] across these layers, and the stride is [2, 2]. As the data moves through these layers, the spatial dimensions reduce while the depth increases due to the growing number of filters, which is reflected in the output shapes: [64, 16, 8], [32, 8, 16], [16, 4, 32], and [8, 2, 64], respectively. After the convolutional layers, the data is flattened into a one-dimensional array and passes through a fully connected layer, sometimes referred to as a dense layer. Here, the data is compressed into the latent space, a lower-dimensional representation of the input data, which is [24] in this case. This is the core of the autoencoder where the data is the most compressed, capturing the essence of the input data.

The process in the decoder is a mirror image of the encoder. It starts with the latent space and aims to reconstruct the original input from this compressed data. Transposed convolutional layers, also known as deconvolutional layers, are used to upsample the data back to its original dimensions. These layers have filters, kernel sizes, and strides similar to those in the encoder, but they work in reverse to increase the dimensions of the data. The decoder has three transposed convolutional layers with 32, 16, and 8 filters respectively, progressively upsampling the data and increasing its spatial dimensions while reducing the depth. Finally, the last layer of the decoder is another transposed convolutional layer that reconstructs the data back to the original input shape, which is [128, 32, 1]. The reconstructed output is the autoencoder's best attempt at recreating the original input data from the compressed representation in the latent space.

3.4.2 Deep Embedded Clustering

Once the autoencoder is trained, the latent representations are used to initialize cluster centers, commonly through algorithms such as K-means. DEC then performs a joint optimization process that refines the parameters of the autoencoder while also enhancing the cluster assignments. This is achieved by minimizing a clustering loss function, which typically aims to decrease the distance between data points and their respective cluster centers in the latent space, thereby improving the representational accuracy and the purity of the clusters.

The Calinski-Harabasz (CH) index and the silhouette method are two popular approaches for determining the optimal number of clusters in K-means clustering. The CH index measures the ratio of between-cluster sum of squares (SSB) to the within-cluster sum of squares (SSW) for each number of clusters, while the silhouette method measures the average silhouette width for each data point. The CH index focuses on the separation between clusters, while the silhouette method focuses on the cohesion within clusters. It is common to use multiple methods and visualizations to make an informed decision about the number of clusters, as there is no definitive answer to the optimal number of clusters, and the choice depends on the specific data and the context of the problem.

The loss function for the clustering layer from **Equation 2.1.7** is denoted as $L_c = \lambda L$, where λ is a crucial hyper-parameter that influences the balance of the clustering layer. If λ is excessively large, it can warp the latent space, obscuring the data's key features. Conversely, a very small λ might render the clustering layer's impact negligible. The value of 0.1 for λ is chosen based on its usage in other studies such as Zali (2023), Mousavi et al., (2020) and Xie

et al., (2016). For optimizing the clustering layer, stochastic gradient descent is employed, combined with a momentum of 0.9 and a learning rate of 0.01. Momentum here refers to the gradient's moving average, assisting in the network weight updates. The autoencoder's weights are modified every 200 iterations, and the training ceases when the number of samples with changed cluster assignments falls below 0.01% of the total input data.

3.5 Time-To-Eruption forecasting

Time-to-event (TTE) eruption forecasting using Gradient Boosting Trees, specifically through an implementation like LightGBM, is a sophisticated machine learning approach that predicts the likelihood and timing of volcanic eruptions. The procedure begins by training the LightGBM model on training split of the data using features from the latent space. Each tree built during the learning process focuses on correcting the errors of the previous ensemble, which improves the model's ability to discern patterns leading up to an eruption.

In machine learning, high-dimensional data refers to data with a large number of features or variables. The curse of dimensionality is a common problem in machine learning, where the performance of the model deteriorates as the number of features increases. This is because the complexity of the model increases with the number of features, and it becomes more difficult to find a good solution. Dimensionality reduction can help to mitigate these problems by reducing the complexity of the model and improving its generalization performance. Our procedure will compare two main approaches to dimensionality reduction: feature selection and feature extraction.

To optimize the performance of the LightGBM model, hyperparameter tuning is essential. Optuna is a hyperparameter optimization framework that automates the search for the best hyperparameters, such as the number of leaves, the minimum data in leaves, the learning rate, and the number of trees. It employs a Bayesian optimization technique to efficiently explore the hyperparameter space and identify the most effective combination for the predictive model (Akiba *et al.*, 2019).

During the tuning process, Optuna evaluates the LightGBM model's performance with various hyperparameter sets, guiding the search towards hyperparameters that minimize prediction error. This iterative tuning not only refines the model's accuracy but also ensures that it generalizes well to new data, which is vital for reliable forecasting of volcanic eruptions. The result is a robust predictive model that estimates the time until the next volcanic event with a quantified level of certainty, aiding in early-warning efforts and risk management.

Halaman ini sengaja dikosongkan

CHAPTER IV
RESULTS AND ANALYSIS

This chapter will explain the methods on the implementation of the model architecture as mentioned in the previous chapter. This entails the step-by-step process results including the features procedures during the research

4.1 Identifying seismo-acoustic signals related to volcanic activity

In our preliminary analysis, we used station-level processing techniques that involves Bartlett beamforming and the Adaptive F-Detector (AFD) to detect potential volcanic activity through stations, including I06AU, I52GB, and I07AU. We will also compare the results of station-level processing with network-level processing and chose the most effective method to detect volcanic activity.

4.1.1 Station-Level Processing

The station-level processing results on a set of backazimuth, trace velocity, and F-statistic values as depicted on **Figure 4.1.** The backazimuth values from the detections, is used to determine the directionality to the source location of the detected signals, which is vital for associating them with Mount Krakatau. Consequently, we filtered the backazimuth values of the station-level processing with values from **Table 3.2.** The results show that for I06AU station, there is more detection after filtering with 50̊-60̊, while in I52GB, filtering would result in less detection. The results the adaptive F-detector can be seen on **APPENDIX C.**

Using the same workflow to the year-long dataset, the results in **Table 4.1** reveals a high event detection rate on I06AU station relative to I52GB and I07AU. This is further complimented on **Figure 4.2** where I06AU is most consistently available and reliable source for infrasound detections relevant to our project's scope; mostly linked to the active periods of the volcano, with signal triggers aligned with specific volcanic events. While other stations like I52GB also contributed valuable data, I06AU's comprehensive coverage and data quality made it the primary focus for our subsequent analysis, as has been explored by Gheri, at, al (2023).

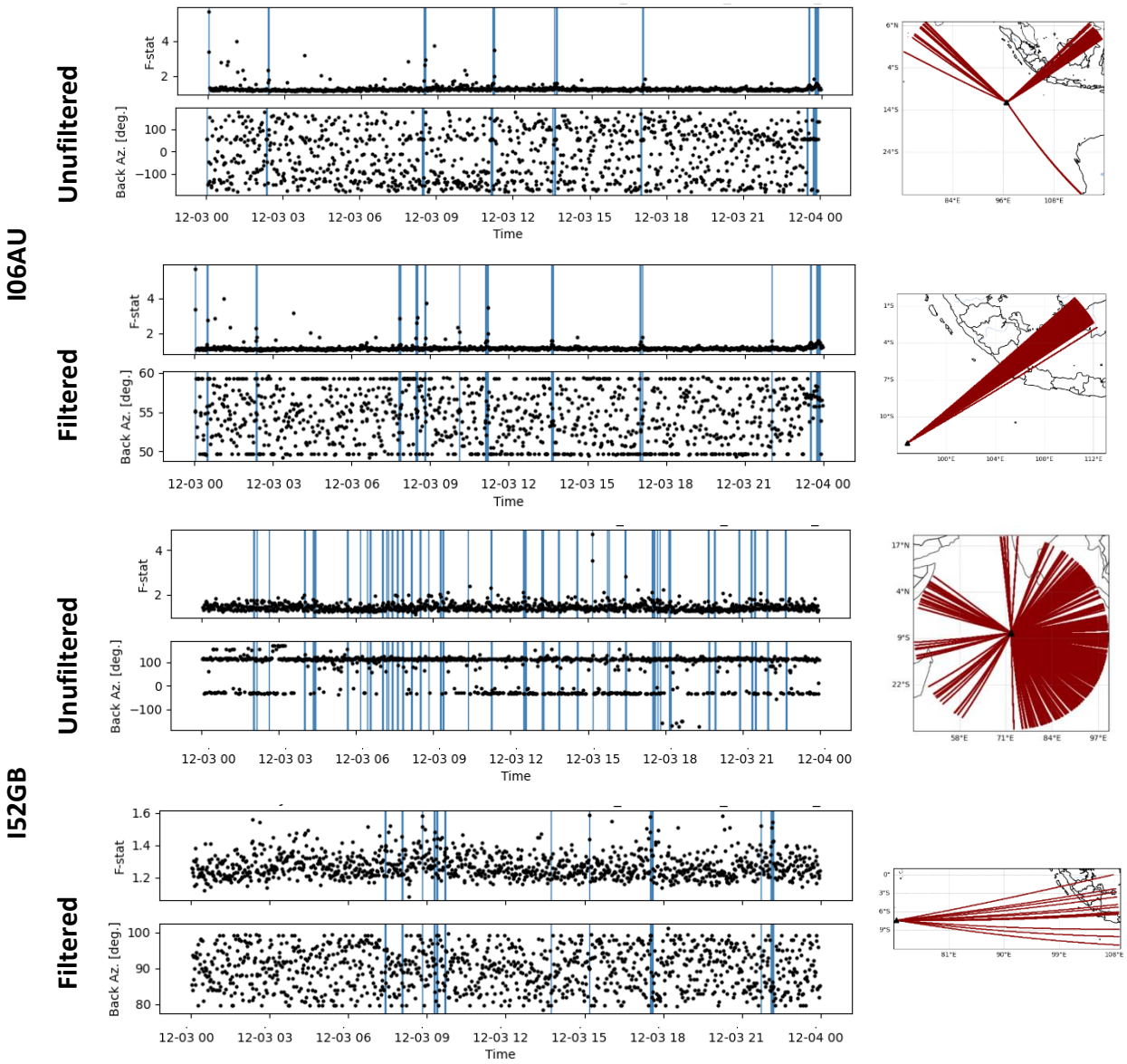


Figure 4.1 Results of Bartlett beamforming with vertical blue bars signify periods where the F-statistic exceeded a threshold and captured by the Adaptive F-Detector (AFD). The "unfiltered" column represents the results of full backazimuth value (360) during the process, while "filtered" column represents the results of partial backazimuth value explained on Table 3.2 .The right panel signifies the spatial plotting of the AFD results

Table 4.1 AFD Detection Results from Various Stations

Stations	Detection Triggers
I06AU	4658
I52GB	563
I07AU	-

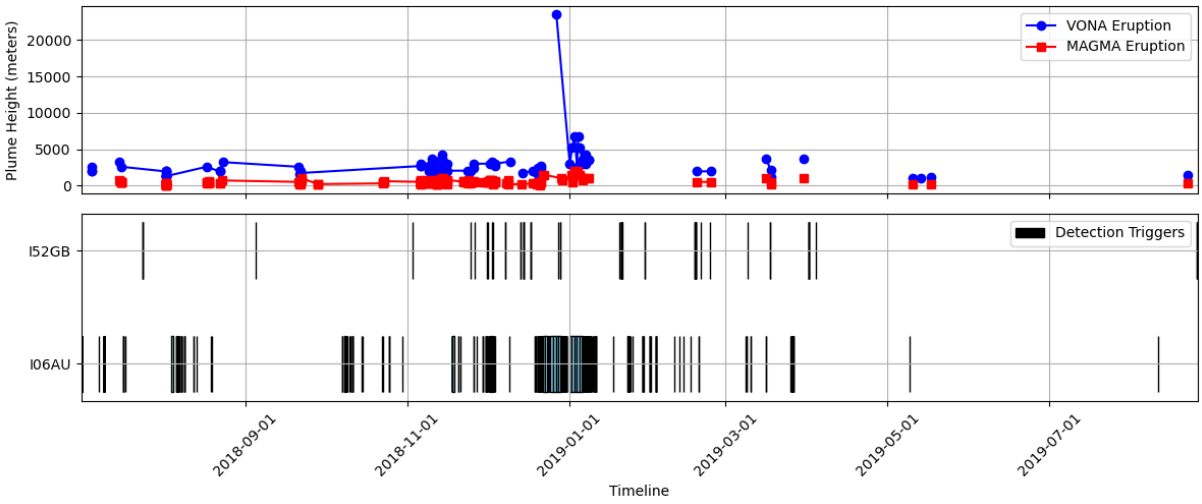


Figure 4.2 Comparison of the availability of station-level processing results with VONA and MAGMA notifications

4.1.2 Comparison with Network-Level Processing method

A multi-station approach or network-lebel processing can be utilized to accurately determine the location and timing of a source event. By employing a pair-based joint-likelihood method, it's possible to cluster detections that are highly likely to have originated from the same source, aiding in event identification. This method integrates event analysis to estimate both the spatial positioning and the origin time of events, enhancing the precision and reliability of the localization process.

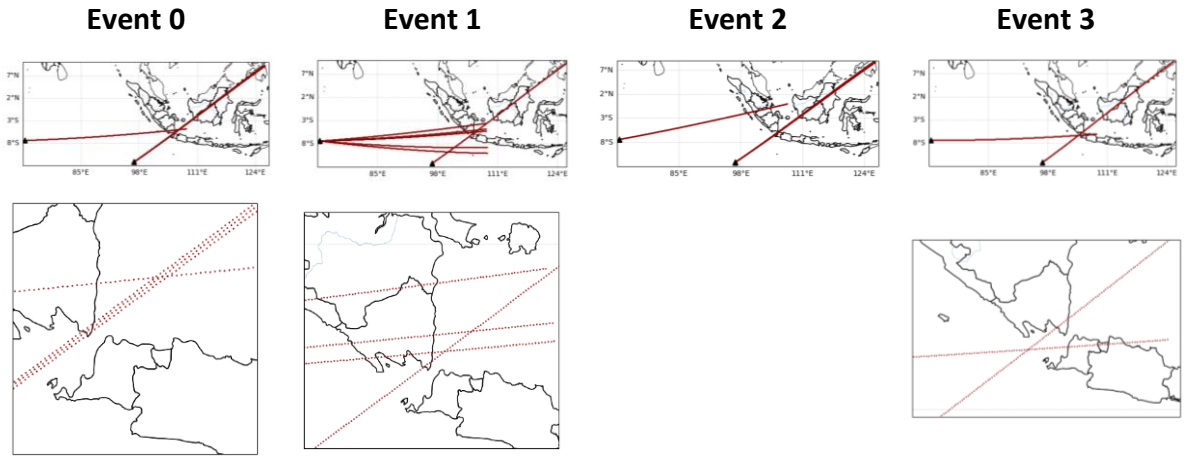


Figure 4.3 Association results of I06AU and I52GB. The red lines indicates the alignment of backazimuth

The results in **Figure 4.3** yielded several events of interest, with Events 1 and 3 standing out due to their intersection proximity to Anak Krakatau. By selecting Events 1 and 3 for

localization process, we focus on occurrences with the highest potential relevance to our intended observation location. The proximity of these events to Anak Krakatau suggests that they could provide significant insights into the characteristic infrasound signals associated with the volcano's different eruptive phases.

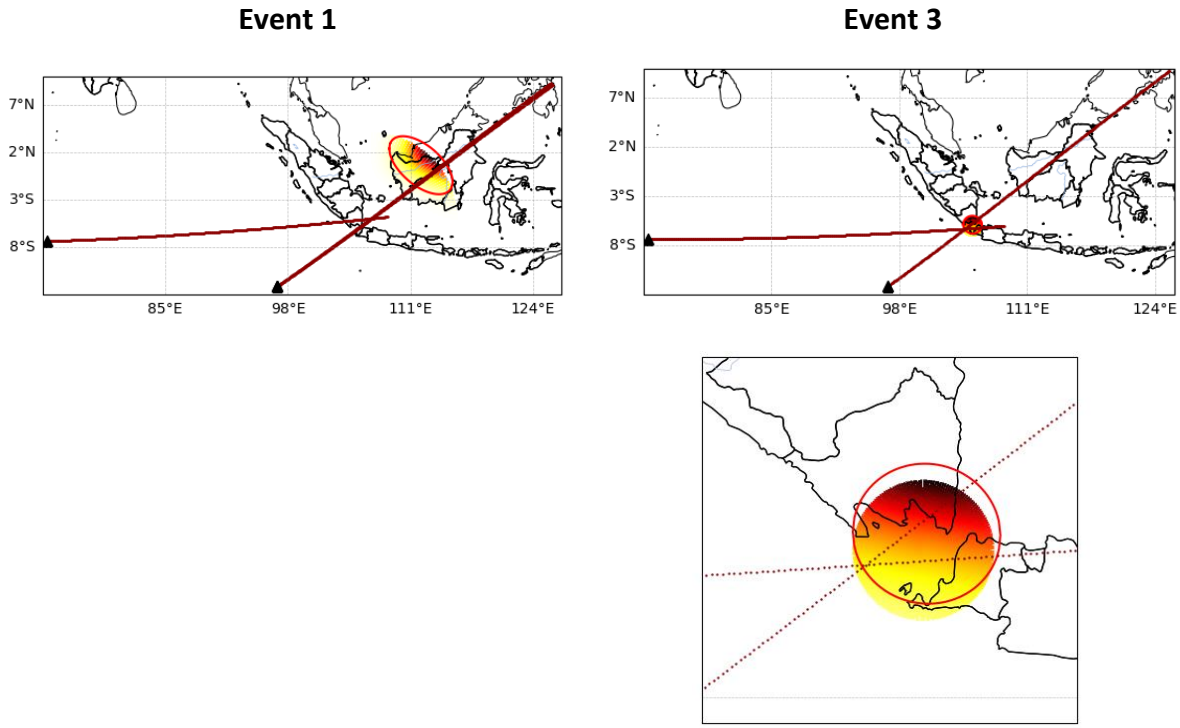


Figure 4.4 Localization results of I06AU and I52GB

Localization results on **Figure 4.4** reveal that Event 3 is closely aligned with Anak Krakatau, suggesting it may be the source of volcanic infrasound activity This clear association strengthens the case for its relevance to volcanic monitoring efforts. In contrast, Event 0 has been disregarded as its associated region is outside the scope of our observation area. When applying the same workflow on the whole observation period, the Event 3 localization results in **Table 4.2** reveals a decrease in detection trigger in comparison with **Table 4.1**. This suggests that while Association and Localization methods may pinpoint specific source locations, they may reduce the overall count of detection triggers.

Table 4.2 Event 3 trigger counts on the whole observation period

Stations	Detection Triggers
I06AU	362
I52GB	30

4.1.3 Extracting relevant seismo-acoustic signals

Due to lack of data shown in **Table 4.2** from a multi-station approach, we adopted the previous single-station approach of I06AU station. Despite the data being from a single station, it encapsulates a rich and representative dataset adequate for our analysis purposes. The results of a year long dataset for our observation period can be examined at **Figure 4.5** where it mantains the same consistent F-statistic threshold pattern as the daily results on **Figure 4.1.**

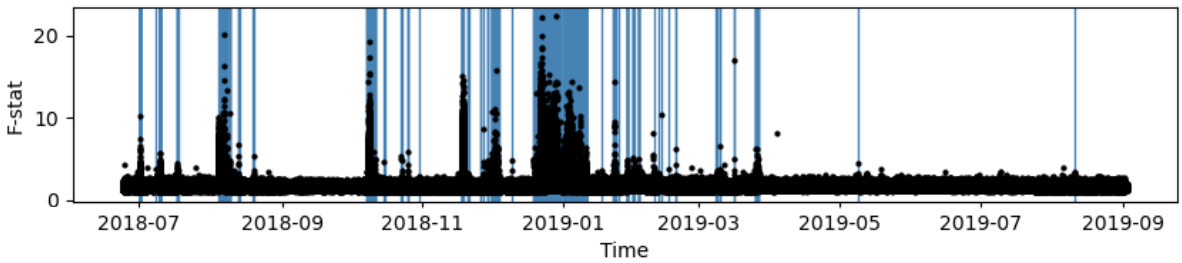


Figure 4.5 F-Statistic Results of beamforminig and AFD of I06AU Station for the whole observation period (July 2018 – September 2019)

Using the results acquired from station level processing such as backazimuth and trace velocity, we directly implement Laslo's method as explained on **Chapter 3.2.3** to isolate relevant signals between arrays inside in a station. As revealed in **Figure 4.4,** we took the best-beam results from each detection segment, where it calculates the summation of the signals of the array in a detected segment. This framework of signal summation distinguishes the noise and relevant signal for further processing, giving us a more accurate representation of ground-truth volcanic activity.

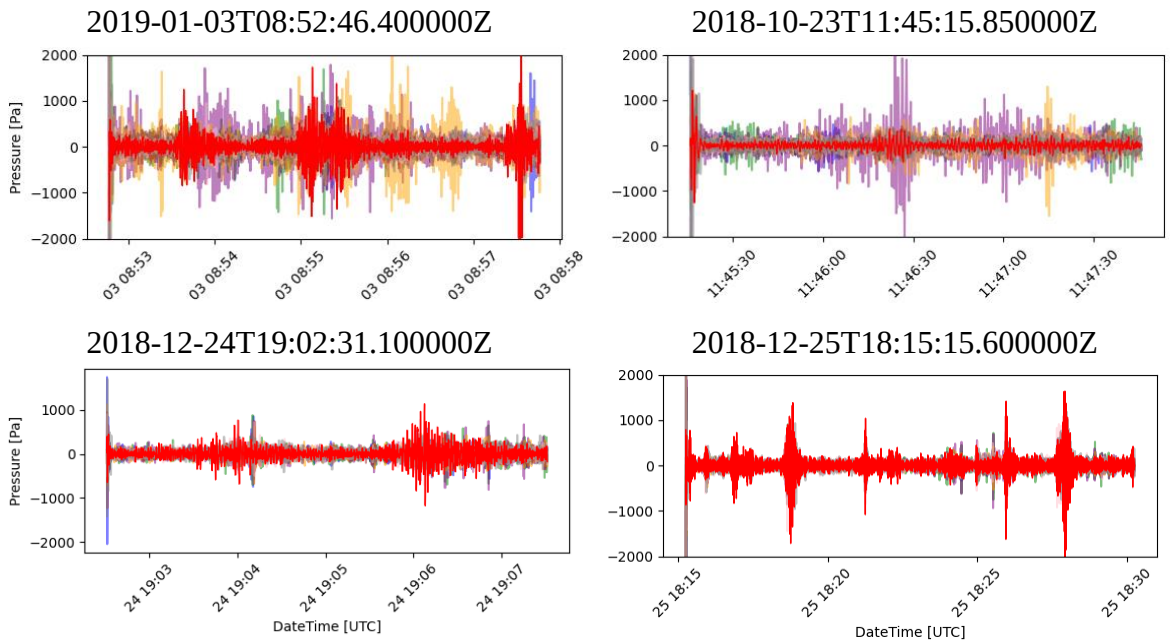


Figure 4.6 Samples of Laslo's best beam visualization towards individual AFD detection results. The bold red line represents the best-beam results, while the semi-transparent lines represent the residuals from eight different arrays.

By correlating these detection results with eruption timelines in the next subchapter, we can begin to map specific signal characteristics to volcanic behaviors. This detailed analysis not only enhances our understanding of the acoustic phenomena associated with volcanic eruptions but also contributes to the development of more sophisticated and accurate early warning systems. The intricate interplay of the temporal and frequential features captured through this analysis forms the basis of our feature extraction methodology, aimed at isolating the most informative attributes for subsequent machine learning applications.

4.2 Deep Embedded Clustering of seismo-acoustic detections

Following **Figure 2.3**, we focus on a well-documented eruptive period of the Anak Krakatau volcano, starting from 18th of June, 2018 until February 2019. The selected time frame encapsulates the sequence of events leading up to, during, and following the significant eruption. The detailed examination of these signals provides invaluable insights into the eruption dynamics and aids in refining our understanding of the processes driving these explosive events.

Using short-time fourier transform (STFT) features from Laslo's best beam waveform as the input to our autoencoder, both the training and validation losses show a consistent exponential decline. The effectiveness of the autoencoder in recreating the original input from the latent spectrogram is demonstrated in **Figure 4.7**, where mean squared error (MSE) metric is used to compute the average squared of the absolute differences between the input data and the reconstructed output. Due to our choice to not normalize the data, the results show an training MSE value of 22.167, which; relatively to testing losses are very close in value throughout the training process.

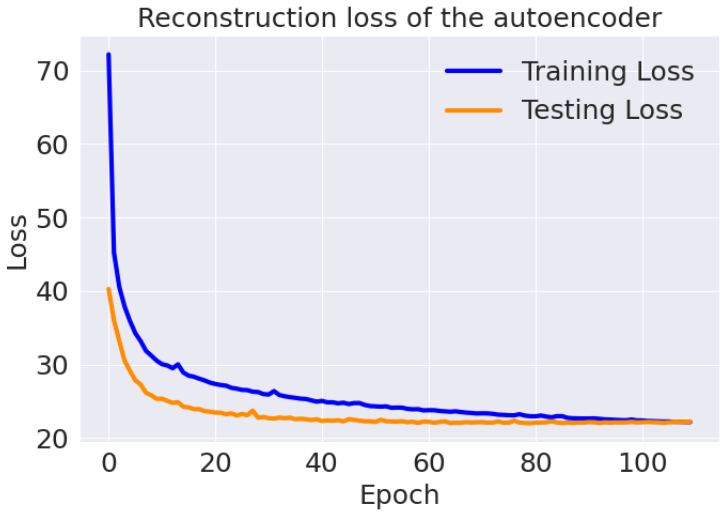


Figure 4.7 Training and validation loss of the autoencoder

The MAE results suggests, the spectrogram retains most of its original structure post-reconstruction; indicating that the autoencoder has successfully captured the features of the latent space during its pre-training phase. This means that the network has effectively learned to identify and extract key features that represent the crucial aspects of the targeted signal. Following this, the k-means clustering algorithm is applied to these features, specifically those at the autoencoder's bottleneck.

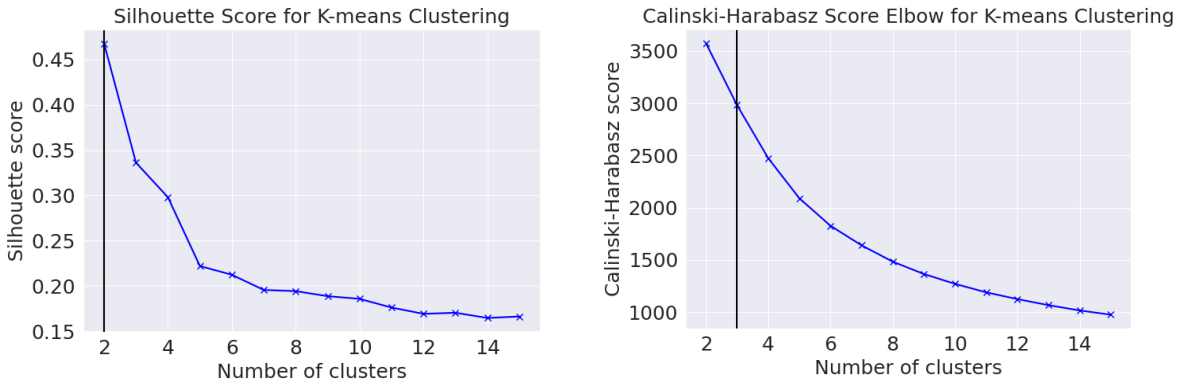


Figure 4.8 Choosing the optimal number of clusters. The number of cluster k=3 is chosen to minimize the loss of Calinski Harbasz score

After training the autoencoder, results show a scatter plot of data points from the latent space as dislayed on **Figure 4.9 a)**. The initial pre-training visualization doesn't show distinct clusters yet, which suggests that the pre-training alone was not sufficient to separate the different types of volcanic activity clearly. Using Calisinzki Harbasz and silhouette score, we chose the optimal number of cluster and apply k-means clustering to the latent space, as shown on **Figure 4.9 b)**. The positions of the centroids in **Figure 4.9 b)** are not shown directly because t-SNE is a non-linear dimensionality reduction method that preserves local structures and relationships at the cost of global ones.

After the fine-tuning process of the latent space depicted in **Figure 4.9 c)** left panel, the raw data and clusters derived from it are now arranged in a serpentine-like formation, reflecting a non-linear and continuous transition between the activity states. Unlike the typical spherical clusters produced by k-means, the continuous and curvilinear arrangement of data points in this t-SNE visualization suggests a gradual progression between clusters rather than discrete, independent groups. This pattern could point to a spectrum of volcanic activity where each subsequent state may develop from the previous one in a sequential manner.

The globally scaled data, as shown in **Figure 4.9 c)** right panel, reveals more discernible clusters compared to the Raw data. This improved clarity indicates that global scaling has enhanced the model's ability to differentiate between states by adjusting the scale of each feature to a common range. As a result, the latent space reflects a more organized distribution of data points, where each cluster is now more cohesive and better separated, with less overlap between different clusters. These results show that fine-tuning make clusters became more cohesive and better separated; regardless of feeding it with unscaled or globally scaled data. The fine-tuned

model is more capable of distinguishing between the various types of volcanic activity represented in the dataset.

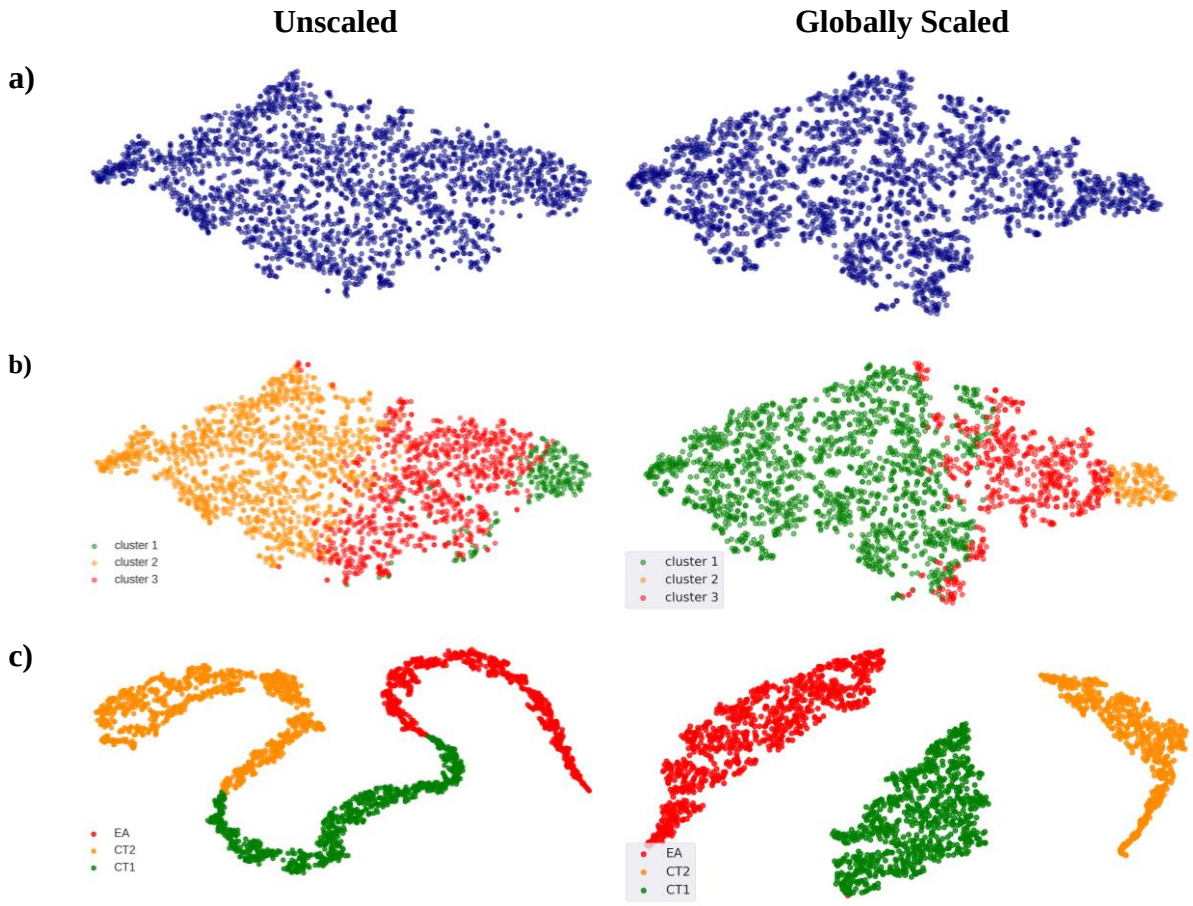


Figure 4.9 t-SNE visualizations of the salient features of Infrasound segments **a)** after pre-training **b)** Clustering results with k=3 after pre-training **c)** after fine-tuning with k=3

Results in **Table 4.2** showed that the number of occurrences for Continuous Tremor 1 (CT1) saw a decrease, indicating a more precise categorization of lower intensity events. Similarly, Continuous Tremor 2 (CT2) also experienced a decrease in occurrences, suggesting an improved distinction of moderate tremor events by the algorithm. In contrast, the occurrences of Explosive Activity (EA) increased, reflecting a significant enhancement in the model's ability to identify and classify the more severe and distinct explosive volcanic events. These changes highlight the fine-tuning stage's effectiveness in optimizing the clustering accuracy and the model's overall discriminative performance.

Table 4.3 Clustering result occurrences

No	Cluster	Occurences	
		Pre-Training	Fine-Tuning
1	Continuous Tremor 1 (CT1)	1273	916
2	Continuous Tremor 2 (CT2)	856	771
3	Explosive Activity (EA)	200	642

From these results, we can infer that fine-tuning process has refined the model's ability to distinguish between different types of volcanic activity. This change suggests that the fine-tuning process may have helped to correct an initial imbalance by reassigning instances that were possibly misclassified during the pre-training phase, leading to a more balanced and accurate representation of the true volcanic activity patterns within the data.

Table 4.4 Summary of clustering results

Event	***CT1***			*CT2*			*EA*		
Feature	Median	Mean	Max	Median	Mean	Max	Median	Mean	Max
enc_0	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_1	130,28	147,30	871,23	268,15	287,28	1080,16	476,25	523,67	2091,18
enc_10	171,66	189,22	1047,83	345,78	369,05	1713,92	730,13	831,44	3922,60
enc_11	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_12	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_13	110,59	152,81	925,65	206,94	276,40	2364,95	345,11	409,30	3020,90
enc_14	187,62	223,42	1062,67	390,25	426,49	1739,00	716,53	805,29	4239,28
enc_15	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_16	90,29	117,17	563,79	255,70	276,26	1207,82	470,05	528,92	3603,54
enc_17	170,32	206,97	1369,54	364,73	413,64	1589,12	592,09	655,49	2531,87
enc_18	143,52	196,78	1029,15	308,28	370,71	1870,83	549,76	605,07	3978,97
enc_19	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_2	172,83	213,23	1296,61	398,53	428,88	1471,12	630,32	697,78	2614,13
enc_20	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_21	165,99	219,13	1481,87	363,41	446,29	2241,56	658,82	770,05	6123,23
enc_22	189,64	218,27	1046,67	408,39	447,11	1715,98	702,98	756,19	2866,45
enc_23	135,01	168,75	928,90	342,75	363,21	1462,65	664,82	778,98	5647,10
enc_3	85,40	120,38	1094,48	200,89	245,89	1441,13	386,20	464,92	4160,81
enc_4	80,47	105,01	541,07	247,84	264,15	929,17	570,24	663,25	3605,34
enc_5	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_6	123,76	153,08	935,92	283,32	310,67	1920,29	599,92	662,40	4288,16
enc_7	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00
enc_8	162,78	182,58	705,44	342,02	353,49	1536,96	619,02	680,20	3881,73
enc_9	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00

Results in clustering; displayed in **Table 4.3** shows Continuous Tremor 1(CT1) with median and mean values are generally on the lower end, except for a few features which show higher values indicating spikes in activity. This suggests a baseline level of activity with occasional increases. As the activity shifts to CT2, there is an upward trend in both median and mean values across the features, indicating an escalation in the frequency and intensity of the volcanic activity. The maximum values in CT2 also show a marked increase, reflecting more significant signal variations which align with moderate volcanic activity levels. When examining the EA cluster, there is a noticeable elevation in both median and mean values across most features, denoting a substantial rise in the intensity of the activity. The maximum values in this cluster show a considerable increase, pointing to very high-intensity, explosive volcanic events. Overall, the data demonstrates a clear progression in the strength and dynamics of volcanic activity, with the statistical measures increasing from CT1 to EA. This indicates that as the volcanic activity intensifies from continuous tremors to explosive events, the infrasound signal characteristics become more pronounced and variable.

4.2.1 Continuous Tremor 1 (CT1)

Figure 4.10 are segment samples that are categorized as "Continuous Tremor 1." The spectrograms appear to exhibit consistent energy within a certain frequency range, likely indicating the presence of a stable process or source that is continuously generating seismic waves. The energy amplitude seems moderate and consistent across the spectrograms, which is in line with the expected behavior of a continuous tremor. There are no high amplitude peaks that would indicate explosive or sudden activity. The energy is distributed evenly over time, which would be expected in a continuous tremor. This suggests a constant, possibly rhythmic source of seismic energy, such as the movement of fluids within the volcanic system.

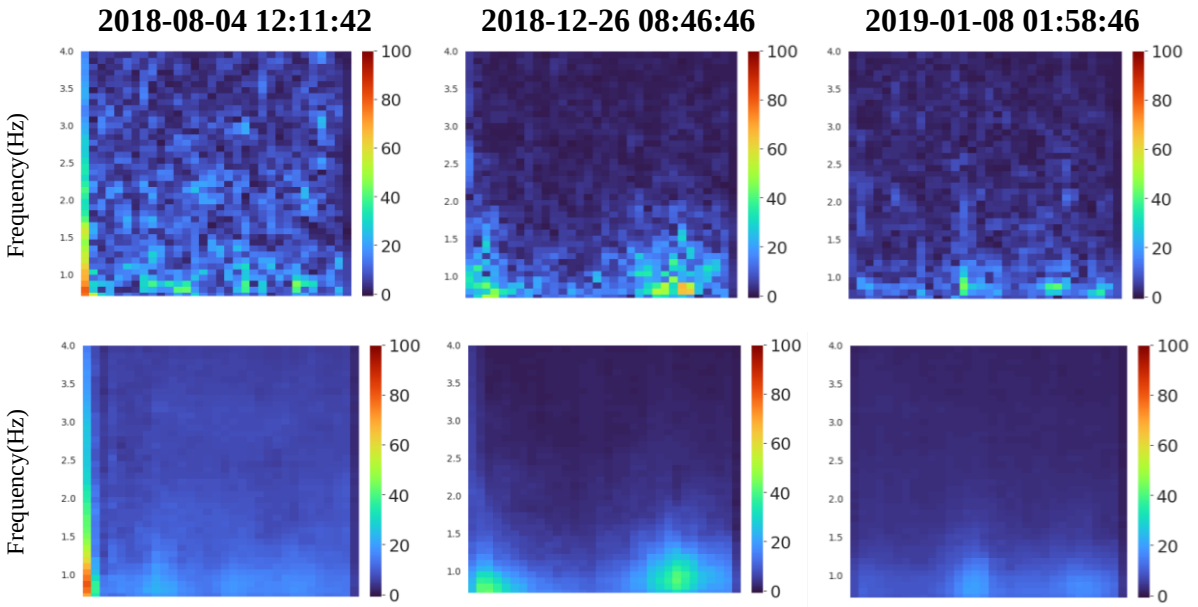


Figure 4.10 Continous Tremor 1 cluster samples. In each subfigure the top row is the is the input frequency-domain features of the autoencoder and the bottom row is the output of the autoencoder. This template of visualization will be later used on Figure 4.11 and Figure 4.12

There seems to be a particular emphasis on the lower frequency bands across the spectrograms. Continuous tremors are often dominated by such low-frequency content, indicating large-scale movement as opposed to localized, high-frequency events. The lack of distinct "hotspots" or high energy peaks throughout the spectrograms reinforces the classification of these signals as a continuous tremor rather than sporadic or explosive activity. There is a visible presence of seismic noise across all frequencies, which is typical for volcanic tremors. The noise level appears to be relatively uniform and not overpowering the tremor signal. The smooth transitions in energy levels without abrupt changes support the continuous nature of the tremor activity.

Upon comparing the raw and encoded data, it reveal key aspects of seismic signatures. The raw data exhibits a more textured distribution of energy across the frequency spectrum, with noticeable energy concentrations in certain frequency bands that suggest consistent seismic activity, possibly related to the movement of magma or volcanic gases. In contrast, the encoded spectrograms present a smoother and more homogenized view of this energy distribution, indicating that the encoding process successfully abstracts the raw data while retaining its essential features. Although the encoded images appear less detailed, they still reflect the overall energy patterns, including the locations of the frequency bands and the amplitude variations over time. This suggests that the encoder effectively captures the

fundamental characteristics of the continuous tremor, such as its sustained nature and frequency content. The lower resolution of the encoded images could potentially enhance the interpretability of the data by focusing on broader seismic trends and reducing the impact of transient noise or non-seismic artifacts. Overall, the encoding process appears to distill the raw seismic data into a form that emphasizes the most critical elements for monitoring volcanic tremors, which could be especially useful for automated detection systems in volcanic monitoring.

4.2.2 Continuous Tremor 2 (CT2)

The spectrograms exhibit a spread of energy across the frequency spectrum but with a particular emphasis on specific bands. If these bands are at slightly higher frequencies or show more variability compared to "Continuous Tremor 1," it might suggest a different type of continuous volcanic process, possibly indicating variations in the magma conduit or different levels of volcanic gas emissions. There seems to be a higher amplitude variation within this cluster compared to "Continuous Tremor 1." If these variations are more pronounced, it might indicate a more dynamic system, while still continuous in nature.

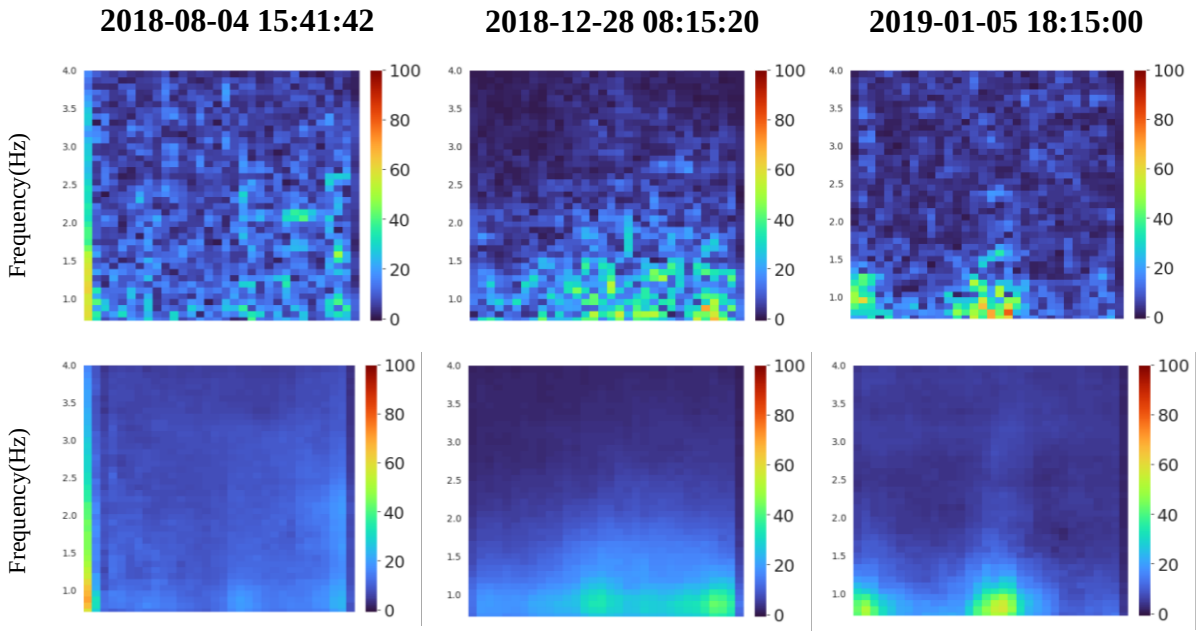


Figure 4.11 Continous Tremor 2 (CT2) cluster samples

Like "Continuous Tremor 1," the energy distribution over time is relatively steady, indicating a persistent process. However, if there are subtle differences in the temporal patterns, such as more frequent energy bursts, it could reflect changes in the volcanic activity's intensity

or frequency. The consistent appearance of certain features across the spectrograms suggests a common source or process. If these features differ from those in "Continuous Tremor 1," maintaining the distinction between the two clusters may be justified.

The side-by-side comparison of the raw data and encoder results for the "Continuous Tremor 2" cluster reveals a consistent capture of seismic activity by the encoding process. In both sets, specific frequency bands are highlighted, indicating the encoder's ability to retain the signature frequencies characteristic of the continuous tremor. The amplitude patterns, while smoothed in the encoded results, mirror the intensity distribution seen in the raw data, ensuring the primary amplitude features are preserved. The temporal patterns, indicative of the persistent nature of the tremor, are clearly retained in the encoded images, demonstrating the encoder’s effectiveness in extracting temporal continuity from the raw seismic signals. Moreover, the encoded spectrograms exhibit a more uniform background, which may enhance the clarity of the seismic signals by reducing visual noise. Harmonic content, a feature indicative of volcanic resonant processes, appears to be generalized in the encoded results, yet still representative of the original data. Overall, the encoder adeptly simplifies the raw spectrograms, extracting and emphasizing the most significant seismic patterns, which can be advantageous for monitoring and interpreting complex seismic data within volcanic regions.

4.2.3 Explosive Activity (EA)

Explosive volcanic activity often produces a broad range of frequencies due to the rapid release of energy. The spectrograms show concentrated areas of higher energy at lower frequencies, which may suggest a release of energy typical of explosive volcanic events. These events often produce strong low-frequency signals due to the large-scale movement of the earth's surface and materials. The presence of bright spots (areas of higher intensity) in the spectrograms indicates moments of higher energy release, which could correspond to explosive eruptions or large gas expulsions.

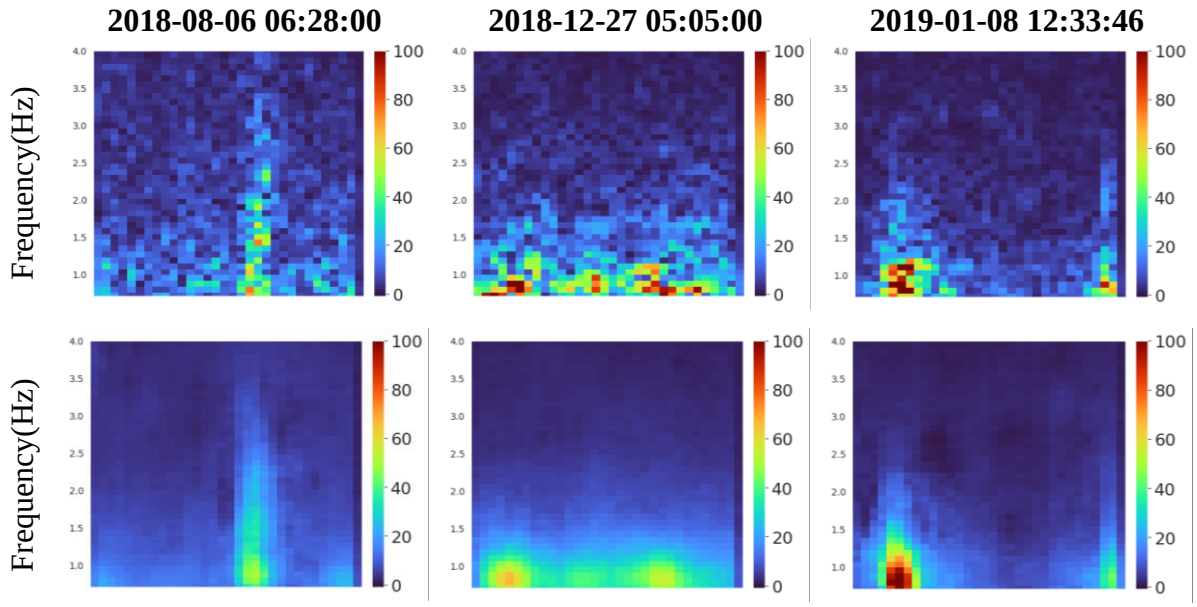


Figure 4.12 Explosive Activity (EA) cluster samples

These high-energy events are short-lived and interspersed with periods of lower energy, this could reflect the sporadic nature of explosive volcanic activity, as opposed to the sustained energy output of continuous tremors. If any harmonic bands are present, they may be related to resonance in magma chambers or conduits. Spasmodic bursts of energy could be indicative of magma fragmentation or gas release during an explosion.

In the raw spectrograms, specific frequency bands stand out with varying intensity, indicative of the seismic activity's energy distribution. These distinct frequency features are mirrored in the encoded results, albeit with a smoother representation, suggesting that the encoder retains the primary frequency characteristics of the original data. High-energy areas or 'hotspots' present in the raw data are identifiable in the encoded images, though they appear less pronounced, which could be attributed to the encoder abstracting away some of the finer details.

Across both sets of spectrograms, the temporal evolution of energy—indicative of the seismic event's duration and progression is consistently captured, demonstrating the encoder's capability to retain the temporal patterns essential for interpreting seismic activity. Furthermore, the encoder appears to streamline the background seismic noise, resulting in a cleaner visualization that may enhance the interpretability of seismic signals. This process of encoding not only simplifies the complex information contained within the raw spectrograms but also emphasizes the most significant patterns, potentially aiding in the categorization and monitoring of seismic events for volcanic activity analysis.

4.3 Revealing the evolution of the 2018 Anak Krakatau unrest

As explained on **Table 2.1,** there are three eruption phases of Anak Krakatau volcano. Understanding the gradual increase in activity of eruption in these phases would help us understand how an eruption works and also to reveal and validate our signal extraction and clustering results. Due to the lack of comparative data regarding the actual activity of the volcano, the discussion in this section is laid qualitatively with references from previous observations.

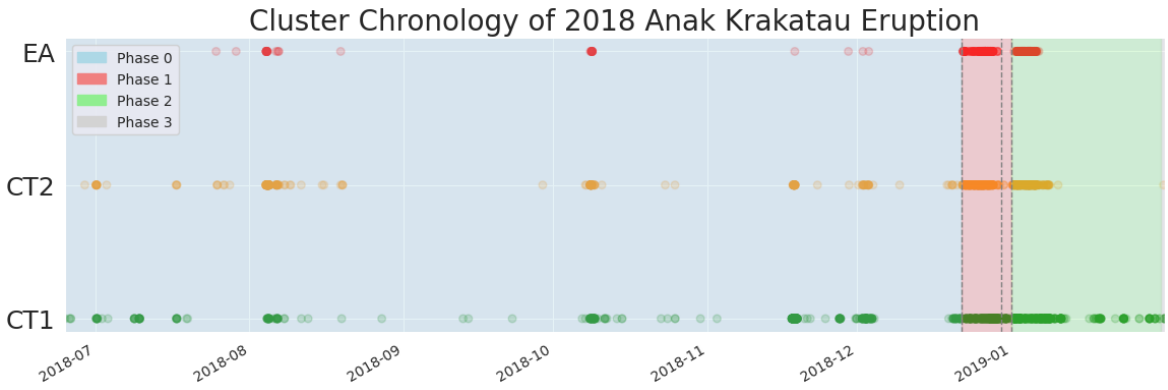


Figure 4.13 Cluster Chronology of 2018 Anak Krakatau Eruption

4.3.1 Phase 0: Initial activity, characterized by Strombolian explosions

The chronology of seismo-acoustic activity during Phase 0 at Anak Krakatau, as shown in **Figure 4.15**, reflects a pattern of ongoing volcanic unrest, starting with a baseline of steady activity marked by Continuous Tremor 1 (CT1). These green dots, which appear regularly across the timeline, suggest a constant, underlying movement of magma or gas (Kurokawa and Ichihara, 2020). Interspersed within this constant activity are instances of Continuous Tremor 2 (CT2), represented by orange dots. CT2's presence, though less frequent, indicates periods of heightened or more dynamic tremor activity, potentially signaling increased pressure or intensified subterranean movement. The sporadic red dots denoting Explosive Activity (EA) correspond with significant eruptive events, characterized by more vigorous releases of energy, as confirmed by the alignment with VONA notifications, which are issued for aviation safety during such explosive events. This clustering of EA dots around VONA notifications implies that these were notable explosive episodes.

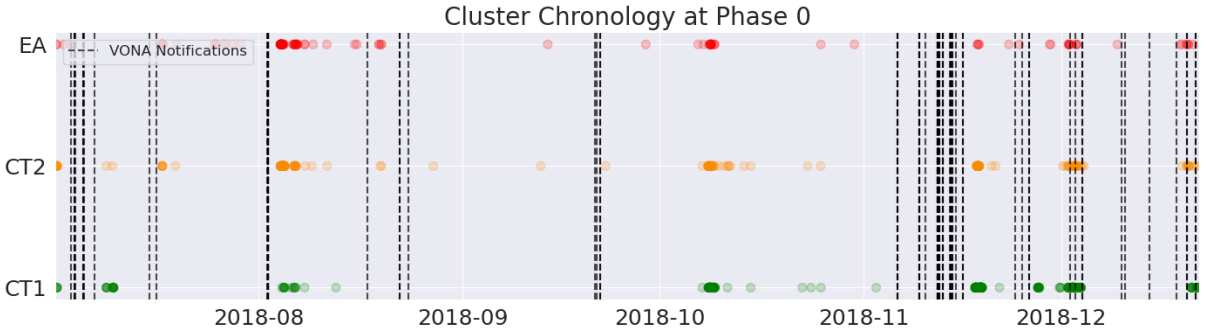


Figure 4.14 Cluster Chronology at Phase 0 with corresponding VONA notification

The graph also reveals a trend of escalating activity leading up to these surtesyan eruptions, with both CT1 and CT2 becoming more frequent, possibly indicative of a build-up to more significant eruptions. This phase of the eruption cycle is thus marked by a complex interplay between persistent, low-level tremor activity and sporadic, high-energy explosive events, underscoring the volatile nature of the volcano and the importance of continuous monitoring for eruption forecasting and aviation safety.

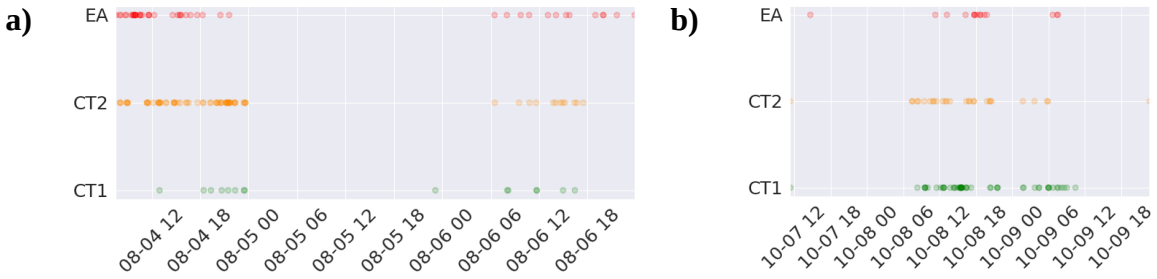


Figure 4.IV.15 "False Detection" clustering results. Figure a) reveals activity during 1st to 15th of August period, Figure b) reveals activity during 3rd-11th of October period

Not all VONA notifications validate our AFD detection and clustering results. As seen on **Figure 4.16 a)**, a series of at least nine explosions took place on 2 August 2018 between 1333 and 1757 local time. Darwin VAAC also reported continuous ash emissions rising to 1.8 km altitude and drifting east on 5 August, with clearly visible in satellite imagery, along with a strong hotspot. (Global Volcanism Program and Venzke, 2023), this activity is not reported at VONA but detected and clustered on our algorithm. This also happens during the period of October 8-9, 2018, where **Figure 4.16 b)** discovered clusters of EA detections that reveals two sustained bursts of activity over 2 days (from October 8-9, 2018) (Global Volcanism Program and Venzke, 2023). Sentinel-2 thermal satellite images also show incandescence in the crater

and emission of ash throughout the month of October, 2018, including October 7 and October 10, 2018 (Rose, 2020). These “false-detection” sequences demonstrate the value of regional infrasound data, particularly in situations in which visual observations are not possible.

4.3.2 Phase 1: Collapse and transition to explosive and Surtseyan style eruptions

The seismic activity represented in **Figure 4.16** for Phase 1 depicts a critical transition period for Anak Krakatau, where the volcano underwent a collapse that led to explosive and Surtseyan-style eruptions, generating high-level plumes over 10 km. The continuous presence of CT1 events, marked by green dots, suggests that a steady volcanic process was underway, possibly related to the destabilization of the volcanic structure that could contribute to a collapse.

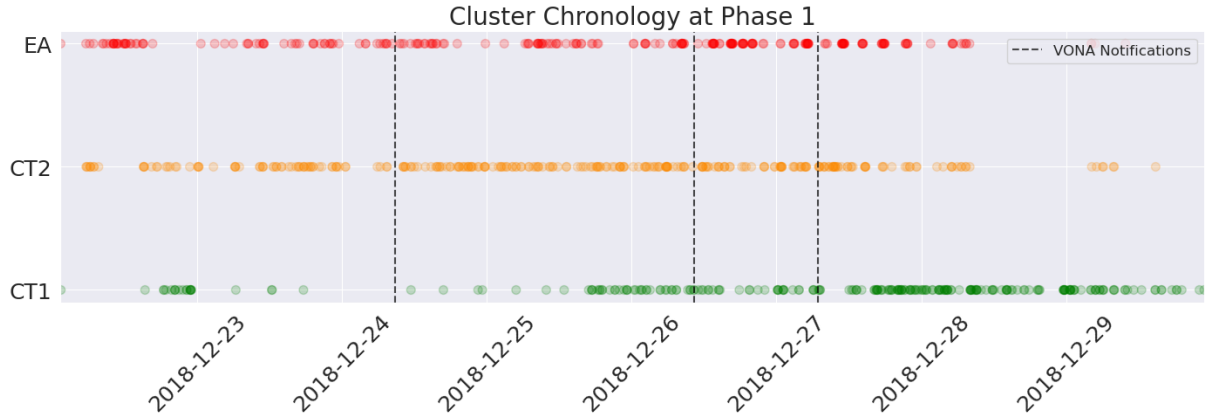


Figure 4.16 Cluster Chronology at Phase 1

The significant increase in CT2 events, as shown by the orange dots, indicates heightened seismic activity. This increase could correspond to intensified subterranean movements, such as the shifting of large volumes of magma, which are often precursors to major structural changes and more violent eruptions. The EA occurrences, highlighted by the red dots, are notably scattered throughout the phase. Some of these events align with VONA notifications, indicating that they were explosive eruptions significant enough to produce substantial ash plumes, posing a threat to aviation. The EA events likely reflect the actual moments of volcanic eruption, where the energy buildup from continuous tremors was released explosively, characteristic of Surtseyan eruptions where magma interacts explosively with water.

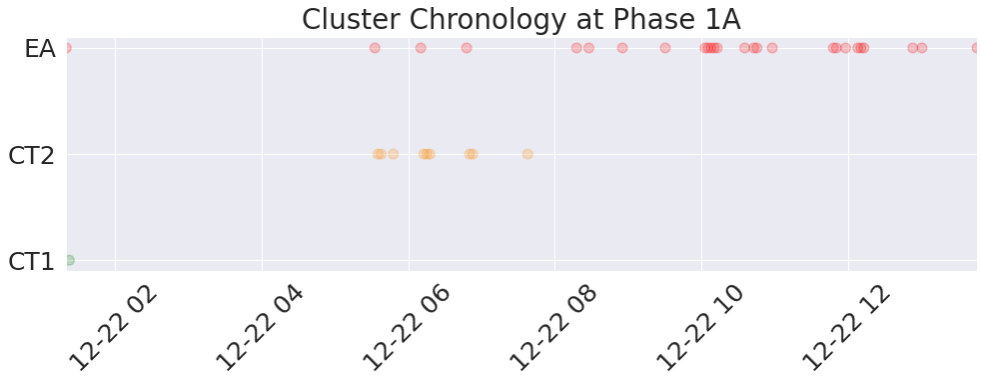


Figure 4.17 Cluster Chronology at Phase 1A

Phase 1A at Anak Krakatau is characterized by intense Strombolian activity and the possible lava flow on the southwest flank (Perttu *et al.*, 2020). CT1 events, depicted in green, are not present. The CT2 events, shown in orange, suggest more dynamic and intense seismic activity, potentially indicating the movement of lava flows and the projection of incandescent material reaching the sea. EA, denoted in red, corresponds with significant eruptive events. These are interspersed throughout the period but are less frequent than tremor events, indicating sporadic explosive eruptions typical of Strombolian activity, where bursts of gas and magma are ejected from the vent (Matoza *et al.*, 2017).

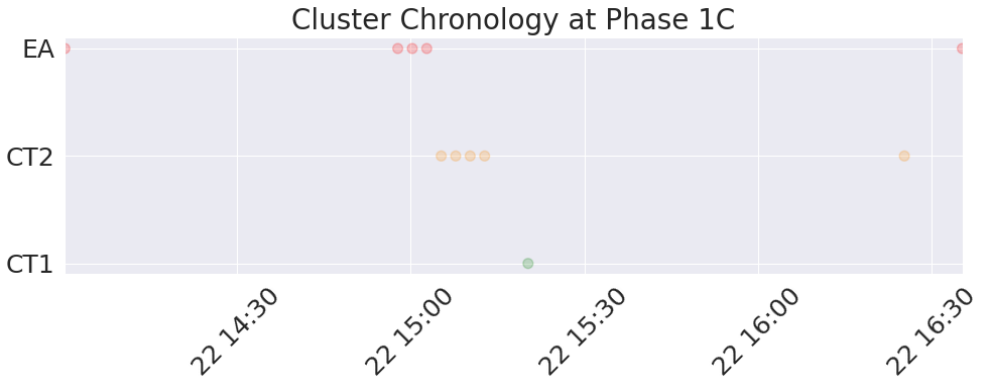


Figure 4.18 Cluster Chronology at Phase 1C

Phase 1C of the Anak Krakatau eruption sequence, as depicted in **Figure 4.18**, spans from 2018-12-22 14:00:00 to 2018-12-22 16:55:00 and is characterized by the propagation and impact of a tsunami and three phreatomagmatic eruptions occurring directly after a significant collapse event (Perttu *et al.*, 2020). The recorded seismic activity includes CT1 events that persist, though less frequently. CT2 events also occur but are not as prevalent as in previous phases, suggesting a reduction in the continuous seismic tremor. Notably, the events are interspersed throughout the timeline, signifying the occurrence of significant explosive eruptions.

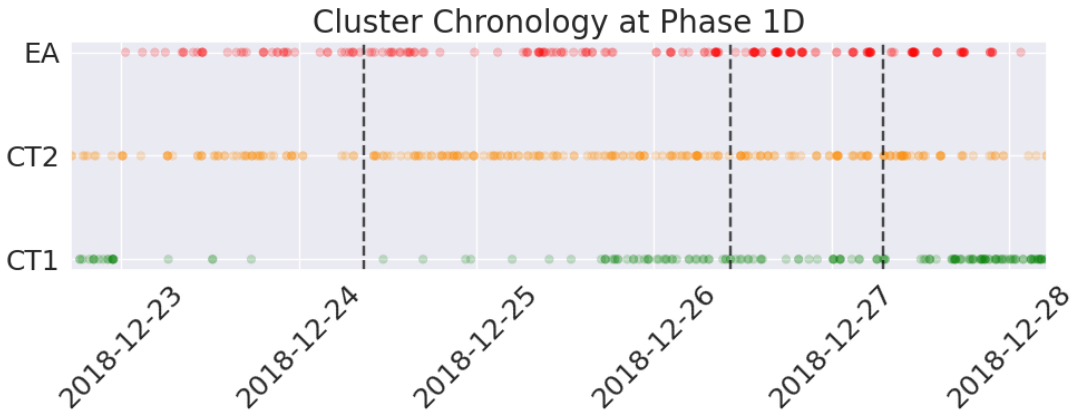


Figure 4.19 Cluster Chronology at Phase 1D

Phase 1D of Anak Krakatau's eruption, as shown in **Figure 4.19** is characterized by sustained high-level plume activity with intermittent pulsing and the detection of SO2 (Perttu, Caudron, *et al.*, 2020), a common byproduct of volcanic eruptions. The seismo-acoustic activity reflects this with CT1 and CT2 events, indicative of ongoing subvolcanic activity. EA events, are more uniformly distributed across the timeline compared to previous phases, suggesting a continuous release of energy and materials from the volcanic system (Fee and Matoza, 2013). The uniformity and frequency of EA events, in conjunction with VONA notifications, suggest that the plume activity was not only sustained but also punctuated by significant bursts of energy, likely corresponding to the intermittent pulsing observed. This phase's seismic profile, featuring the interplay between steady tremor signals and intermittent explosive events, is consistent with the seismic signatures of volcanoes exhibiting sustained eruptive activity.

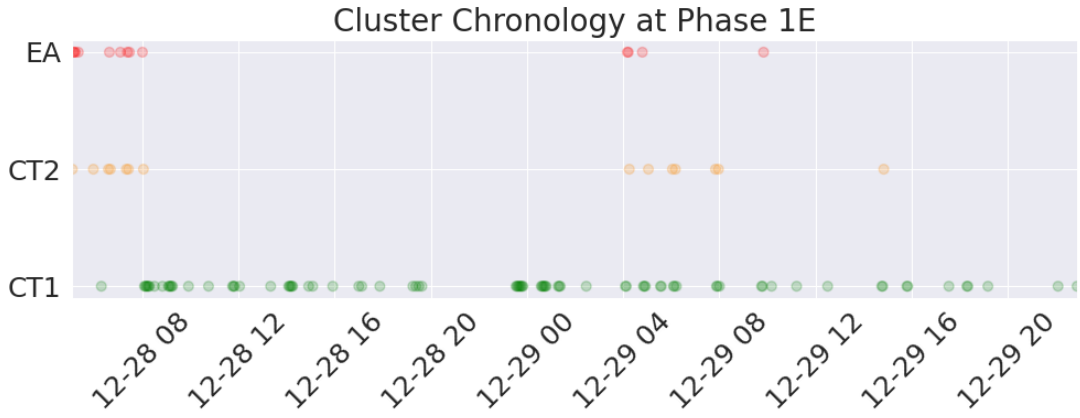


Figure 4.20 Cluster Chronology at Phase 1E

Phase 1E, illustrated in **Figure 4.20**, is a period described by the absence of a sustained plume detectable by satellite and a winding down of geophysical observations. This suggests a decrease in overall volcanic activity, with less frequent CT1 events, indicating a reduction in the continuous seismic energy release from the volcanic system. CT2 events are also reduced and sporadic, which corresponds with a decline in the intensity of the seismic tremor activity. EA events have notably decreased in frequency compared to previous phases. This phase likely reflects a period of relative quiescence following the more vigorous activity of earlier phases. Even though it is described as an absence of a sustained plume, our detection results reveals three distinct explosive activity before a transition to a continuous eruptive plume in Phase 2, similar to the observation by A. Perttu (2020).

The overall pattern in Phase 1 shows a persistent and increasing level of seismic activity, which aligns with the geological interpretation of the volcano undergoing a collapse and transitioning into a more explosive phase. The interplay of continuous tremors with explosive events suggests a volatile system with both ongoing magma dynamics and sudden explosive disruptions. This pattern underscores the complexity of monitoring such phases, where both the continuous movement of magma and the potential for rapid escalation to explosive activity must be closely observed for timely hazard assessment and public safety measures.

4.3.3 Phase 2: Period of high-level plumes and the rebuilding of the island

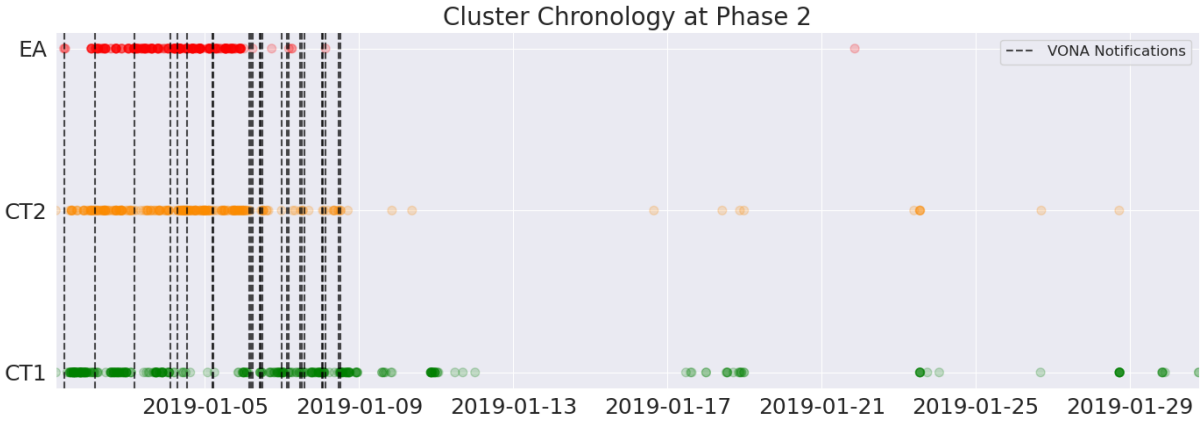


Figure 4.21 Cluster Chronology at Phase 2

Phase 2 of Anak Krakatau's eruption, depicted in **Figure 4.21**, spans from 1st until 31th of January, 2019 and is characterized by another period of high-level plumes and the rebuilding of the island (Perttu, Caudron, *et al.*, 2020). The seismo-acoustic activity during this phase shows a persistence of CT1 (Continuous Tremor 1) events and a continuous presence of CT2 (Continuous Tremor 2) and EA (Explosive Activity) events. The EA events, in particular,

appear to be less frequent but still periodically significant, as indicated by their alignment with VONA notifications.

The rebuilding of a volcanic island and the generation of high-level plumes are typically accompanied by seismic and infrasonic signals. These signals are indicative of the dynamic processes occurring as the volcano re-establish its structure. During such periods, there may be a combination of long-period (LP) events associated with magma movement, harmonic tremor related to the pressurization within the volcanic conduits, and transient explosive signals resulting from gas and magma interactions (Chouet, 2003; McNutt *et al.*, 2015)

4.3.4 Phase 3: Predicting cluster assignment on new unseen detections

Following the results from **Chapter 4.2,** I have decided to use the fine-tuned autoencoder to predict new unseen detections on Phase 3, which is characterized by Sporadic eruptions (Perttu, Caudron, *et al.*, 2020). During the clustering process, it was revealed on **Figure 4.22** that the relative positioning of CT1, CT2, and EA clusters in relation to the already fine-tuned model is maintained. The clusters in the training data appear to have a continuous flow, and this characteristic seems to be present in the new data clustering as well. This suggests that the model is capturing temporal or sequential aspects of seismo-acoustic events. The performance of clustering results can also be seen on **Figure 4.23** as samples from each cluster is visibily similar with clustering results on the previous chapter.

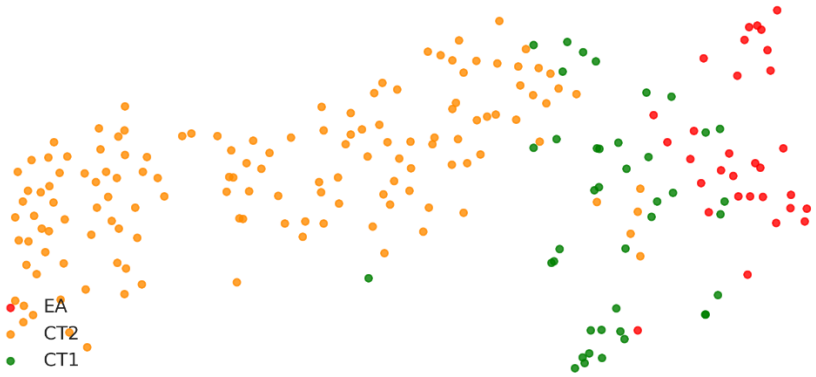


Figure 4.22 t-SNE visualization of salient features of unseen infrasound segments on Phase 3

Phase 3 of Anak Krakatau's eruption, depicted in **Figure 4.24**, spans from 1st of February 2019. The seismo-acoustic activity during this phase shows a persistence of CT1 (Continuous Tremor 1). These events are less frequent than in previous phases, which aligns with the characteristic sporadic nature of the eruptions during this phase. CT2 events are also

sporadic but fewer in number compared to CT1 events. The presence of CT2 suggests that there are periods of heightened seismic activity, potentially indicative of more significant magmatic or hydrothermal processes occurring intermittently. EA events, shown by orange dots, are notably sporadic with clusters of activity followed by periods of quiescence. This pattern is characteristic of sporadic eruptions, where explosive events occur irregularly rather than as part of a continuous sequence.

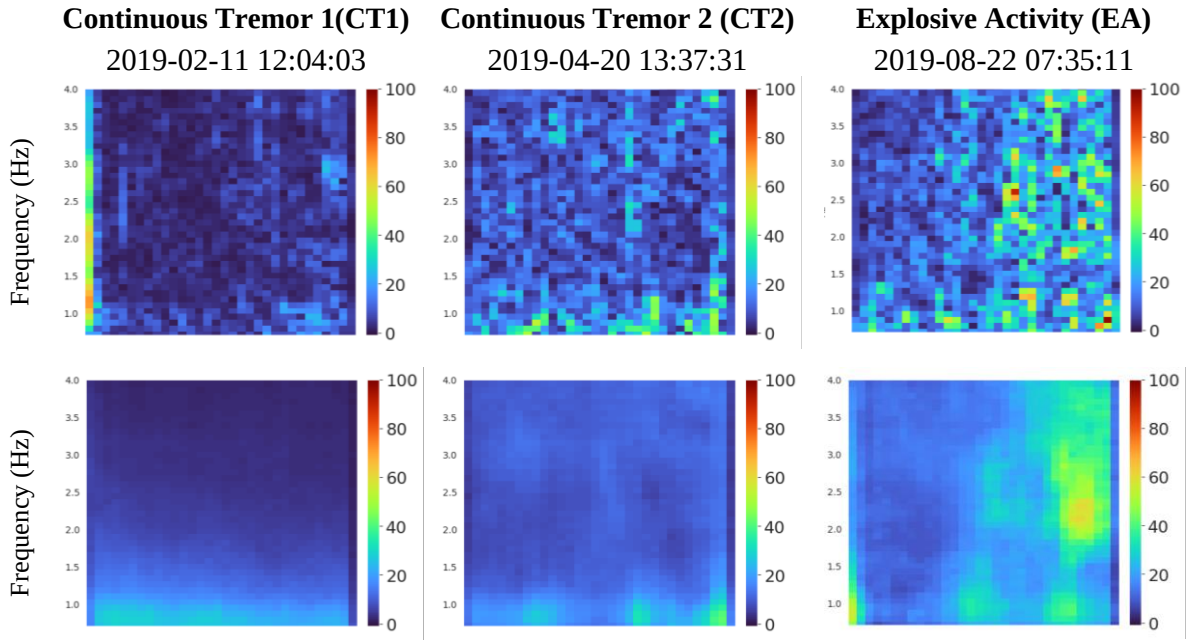


Figure 4.23 Samples of CT1, CT2 and CT3 activites on Phase 3

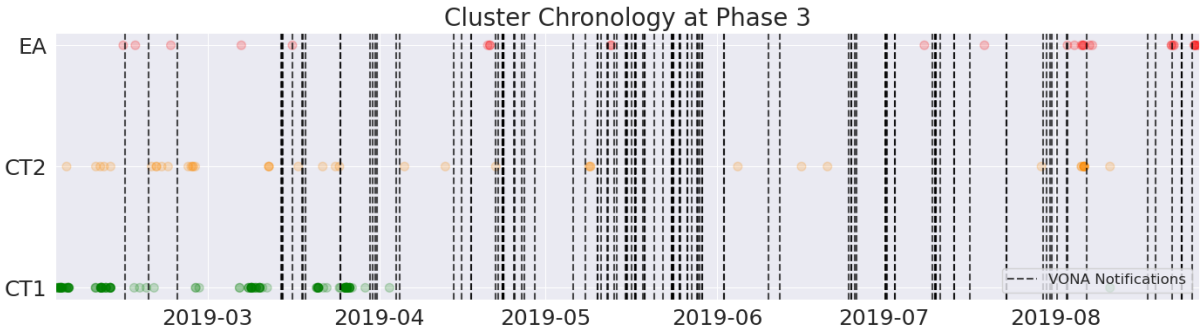


Figure 4.24 Cluster Chronology at Phase 3

The use of a fine-tuned autoencoder for predicting new unseen detections in Phase 3 indicates that the model has been adjusted to account for the less frequent and more unpredictable nature of the seismic events associated with sporadic eruptions, with few caveats. Overall, the timeline reflects a shift from a more continuous pattern of seismicity to one that is

indicative of a less active or dormant phase, interspersed with sporadic bursts of volcanic activity. The use of advanced machine learning models like autoencoders for prediction can enhance the ability of real-time monitoring for such unpredictable volcanic phases.

4.4 Forecasting volcanic eruptions based on insights gained from DEC

We will concentrate our predictive efforts on Phase 0 of the Anak Krakatau volcanic unrest due to its consistent temporal pattern across the whole period. Building upon our understanding of Strombolian eruption dynamics from **Chapter 4.3.1**, we can assume that the precursors to each eruption exhibit homogeneous characteristics, with variations primarily in their detection times relative to the eruption. As shown in **Figure 4.25 a)**, the frequency of data points increases as we approach an eruption, leading to a severe imbalance in our dataset. To avoid amplifiying this imbalance, we have established an 18-hour window prior to each eruptive event for data collection.

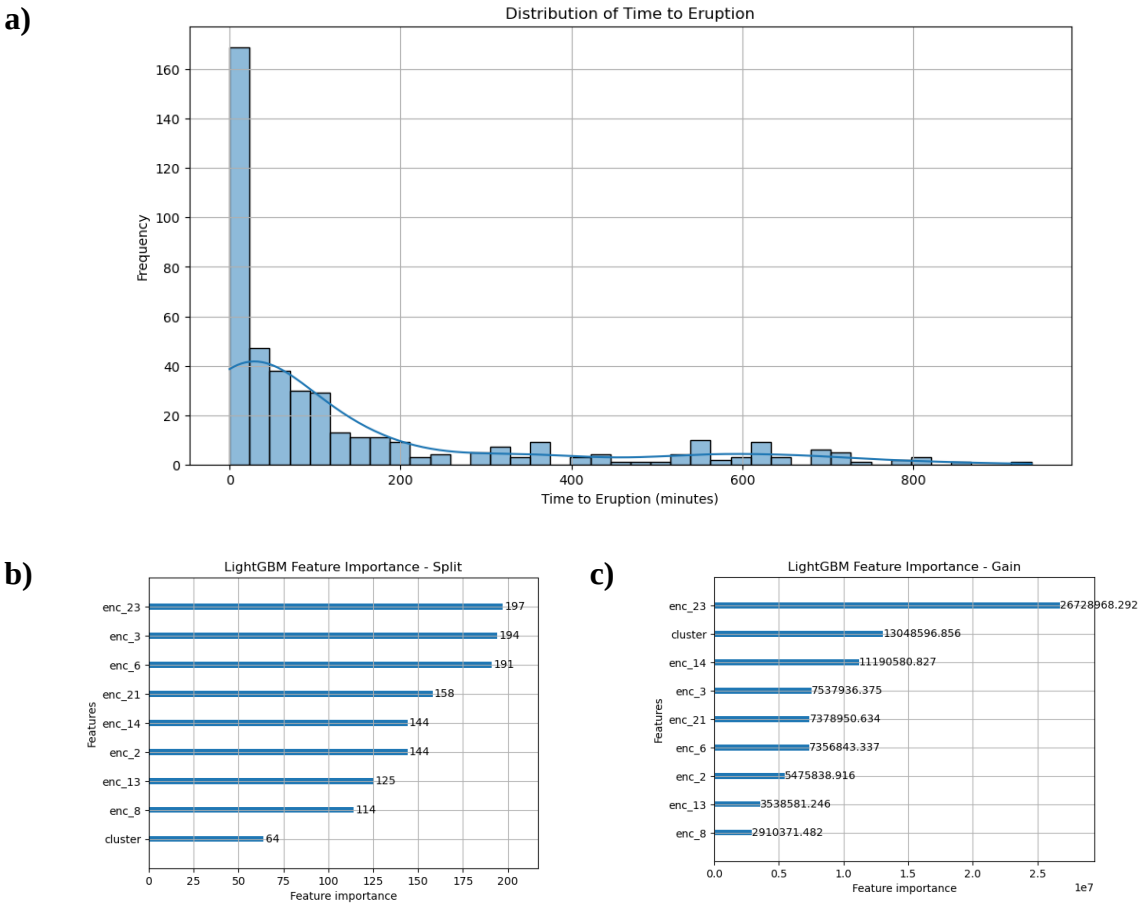


Figure 4.25 Exploratory data analysis during Phase 0 of Anak Krakatau unrest

During our EDA process, **Figure 4.26 b)** shows the number of times a feature is used in a model split. The scale represents the count of splits that include the feature across all

boosting rounds (trees) in the model. The “enc_23” feature in the plot has the highest value of importance, suggested it is frequently used in making splits and a significant predictor for the target variable. **Figure 4.26 c)** plot illustrates the cumulative gain of the splits which use the feature. This is a measure of the total reduction in loss that results from splits on a feature across all trees. “cluster” shows a high value, the opposite of the split feature importance which implies that when this feature is used in a split, it significantly contributes to the model's predictive accuracy. This doesn't necessarily mean that "cluster" is used often to make splits (as seen in the split-based importance plot where it has a lower count), but when it is used, it's very effective in reducing the model's loss.

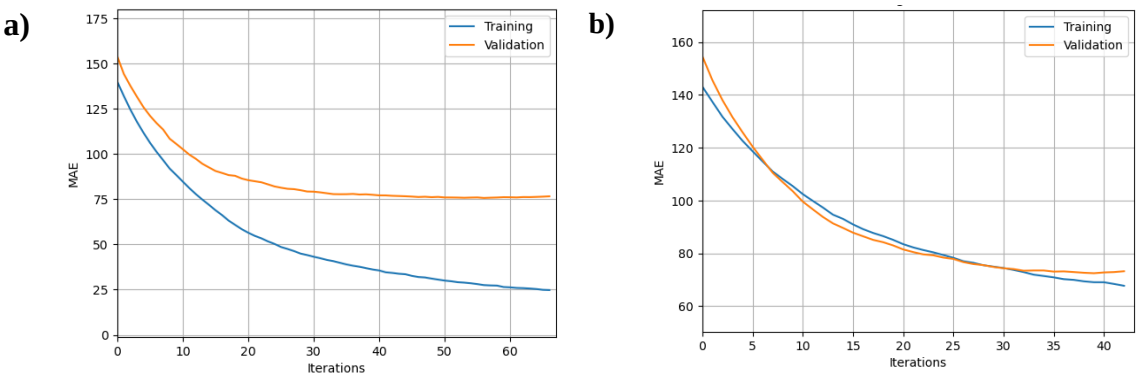


Figure 4.26 MAE results during training **a)** Training on feature selection method **b)** Feature extraction (PCA) method

The gap between the training and validation MAE on feature selection method, as shown on **Figure 4.27 a)** indicates that the model that uses feature selection method is fitting the training data better than it is fitting unseen data. However, the gap is not widening significantly as iterations increase, which is a good sign that the model isn't overfitting severely. The feature extraction method, as seen on **Figure 4.27 b)** shows a smaller gap between training and validation MAE, suggesting that the model with PCA components may generalize better on unseen data, possibly because PCA reduces the dimensionality in a way that mitigates overfitting by discarding some noise.

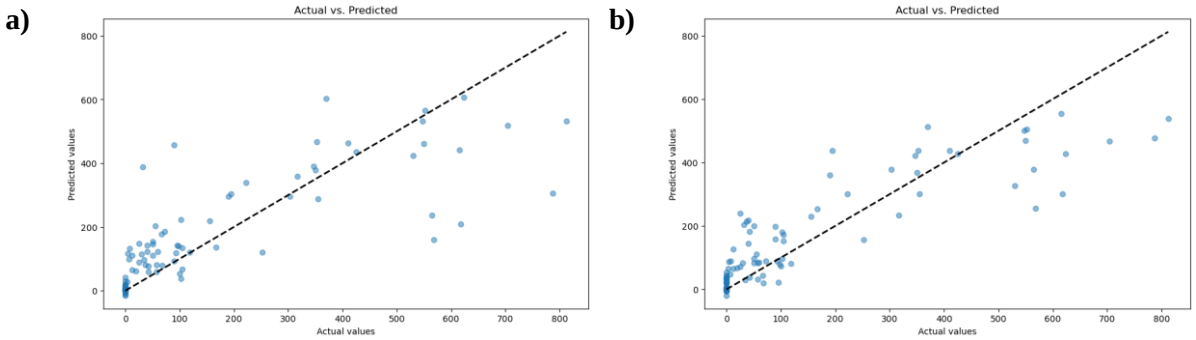


Figure 4.27 Actual vs Predicted results of time-to-eruption values a) Feature selection method b) Feature extraction (PCA) method

Table 4.5 Model results metrics

Metric	*Feature Selection Score*	*Feature Extraction Score*
Mean Absolute Error (MAE)	75.711	73.41
Root-mean-square deviation (RMSE)	127.9062	106.44
Coefficient of Determination (R^2)	0.644	0.7541

As suggested on **Table 4.5,** the model using feature extraction has a slightly lower MAE and RMSE compared to the model using feature selection. The feature extraction model has a higher R^2 value; the closer to 1, the better the model explains the variability of the response data around its mean. The overall results shows that our model can identify a segment of the data with "time-to-eruption" values that is off by about 73.41 minutes off from the actual values. In this context of 18 hours of data range, it is still acceptable with implications of an early warning system. A higher RMSE suggests there may be quite a few large errors, or outliers, where the predictions are particularly poor. An R^2 of 0.754 means that approximately 75.4% of the variance in our target variable (time to event) is explained by the model. This is a moderate score, indicating that while the model has learned a significant portion of the data's structure, there's still room for improvement.

CHAPTER V
CONCLUSION & FURTHER RESEARCH

5.1 Conclusion

The conclusion that can be drawn from the analysis done in this final project are shown in the following:

1. A combination of bartlett beamforming, adaptive F-detector and Laslo's best beam method is effective in extracting relevant signals across an array for volcano monitoring. The effectiveness of these methods is underscored by our detection of 4658 triggers from the I06AU station, which aligns closely with periods of heightened volcanic activity, suggesting a refined approach to signal extraction.
2. The deployment of an unsupervised deep learning approach addresses the challenge of limited labeled datasets in volcanology. With an MSE value of 22.167, our utilization of STFT features and autoencoder not only underscores the potential for automated pattern discovery in limited volcanic signal data but also emphasizes the need for continuous data-driven model refinement to capture the complex temporal dynamics of volcanic systems.
3. Testing and validating eruption forecasting models—has been approached through gradient-boosted regression techniques, achieving an MAE of 73.41 minutes and an R2 of 75.4%. While these results are promising, the inherent unpredictability of volcanic systems necessitates further enhancement of our models.

5.2 Further Research

Our research has conluded a few suggestions:

1. Our deep embedded clustering techniques still relies heavily on segmented detection results of F-statistics, to further understand the nature of volcanic activity, we suggest the use of continues time-series without gaps
2. Current methodologies, including the Bayesian Event Tree and survival analysis for volcanic hazard assessment, show promise in accounting for the stochastic nature and limited knowledge inherent in volcanic activity forecasting

Halaman ini sengaja dikosongkan

REFERENCES

A. Garces, M. (2013) ‘On Infrasound Standards, Part 1 Time, Frequency, and Energy Scaling’, *InfraMatics*, 02(02), pp. 13–35. Available at: https://doi.org/10.4236/inframatics.2013.22002.

Akiba, T. *et al.* (2019) *Optuna: A Next-generation Hyperparameter Optimization Framework*, *arXiv.org*. Available at: https://arxiv.org/abs/1907.10902v1 (Accessed: 12 February 2024).

Albert, S. (2015) *Using infrasound to characterize volcanic emissions at Tolbachik, Karymsky, and Sakurajima volcanoes*. Agu.

Arrowsmith, S. *et al.* (2009) ‘The F-Detector Revisited: An Improved Strategy for Signal Detection at Seismic and Infrasound Arrays’, *Bulletin of The Seismological Society of America - BULL SEISMOL SOC AMER*, 99, pp. 449–453. Available at: https://doi.org/10.1785/0120080180.

Blom, P. *et al.* (2020) ‘Evaluation of a pair-based, joint-likelihood association approach for regional infrasound event identification’, *Geophysical Journal International*, 221(3), pp. 1750–1764. Available at: https://doi.org/10.1093/gji/ggaa105.

Brown, S.K. *et al.* (2017) ‘Volcanic fatalities database: analysis of volcanic threat with distance and victim classification’, *Journal of Applied Volcanology*, 6(1), p. 15. Available at: https://doi.org/10.1186/s13617-017-0067-4.

Carniel, R. *et al.* (2020) *Machine Learning in Volcanology: A Review*, *Updates in Volcanology - Transdisciplinary Nature of Volcano Science*. IntechOpen. Available at: https://doi.org/10.5772/intechopen.94217.

Caudron, C. *et al.* (2015) ‘On the use of remote infrasound and seismic stations to constrain the eruptive sequence and intensity for the 2014 Kelud eruption’, *Geophysical Research Letters*, 42(16), pp. 6614–6621. Available at: https://doi.org/10.1002/2015GL064885.

Chouet, B. (2003) ‘Volcano Seismology’, *pure and applied geophysics 2003 160:3*, 160(3), pp. 739–788. Available at: https://doi.org/10.1007/PL00012556.

Christophersen, A., Behr, Y. and Miller, C. (2022) ‘Automated Eruption Forecasting at Frequently Active Volcanoes Using Bayesian Networks Learned From Monitoring Data and Expert Elicitation: Application to Mt Ruapehu, Aotearoa, New Zealand’, *Frontiers in Earth Science*, 10. Available at: https://www.frontiersin.org/articles/10.3389/feart.2022.905965 (Accessed: 2 January 2023).

Cutler, K.S. *et al.* (2022) ‘Downward-propagating eruption following vent unloading implies no direct magmatic trigger for the 2018 lateral collapse of Anak Krakatau’, *Earth and Planetary Science Letters*, 578, p. 117332. Available at: https://doi.org/10.1016/j.epsl.2021.117332.

Dahren, B. *et al.* (2012) ‘Magma plumbing beneath Anak Krakatau volcano, Indonesia: Evidence for multiple magma storage regions’, *Contributions to Mineralogy and Petrology*, 163, pp. 631–651. Available at: https://doi.org/10.1007/s00410-011-0690-8.

Daniell, J. *et al.* (2015) ‘Global Earthquake and Volcanic Eruption Economic losses and costs from 1900-2014: 115 years of the CATDAT database - Trends, Normalisation and Visualisation’, p. 8119.

Dugick, F.D., Blom, P. and Webster, J. (2020) ‘InfraPy Documentation’.

Evers, L.G. (2008) *The inaudible symphony: on the detection and source identification of atmospheric infrasound*. Delft: Delf Univerity of Technology.

Fee, D. *et al.* (2017) ‘Eruption mass estimation using infrasound waveform inversion and ash and gas measurements: Evaluation at Sakurajima Volcano, Japan’, *Earth and Planetary Science Letters*, 480, pp. 42–52. Available at: https://doi.org/10.1016/j.epsl.2017.09.043.

Fee, D., Garces, M. and Steffke, A. (2010) ‘Infrasound from Tungurahua Volcano 2006–2008: Strombolian to Plinian eruptive activity’, *Journal of Volcanology and Geothermal Research*, 193(1), pp. 67–81. Available at: https://doi.org/10.1016/j.jvolgeores.2010.03.006.

Fee, D. and Matoza, R.S. (2013) ‘An overview of volcano infrasound: From hawaiian to plinian, local to global’, *Journal of Volcanology and Geothermal Research*, 249, pp. 123–139. Available at: https://doi.org/10.1016/J.JVOLGEORES.2012.09.002.

Freire, S. *et al.* (2019) ‘An Improved Global Analysis of Population Distribution in Proximity to Active Volcanoes, 1975–2015’, *ISPRS International Journal of Geo-Information*, 8(8), p. 341. Available at: https://doi.org/10.3390/ijgi8080341.

Garcés, M. *et al.* (2003) ‘Infrasonic tremor observed at Kilauea Volcano, Hawai’i’, *Geophysical Research Letters*, 30(20). Available at: https://doi.org/10.1029/2003gl018038.

Gheri, D. *et al.* (2023) ‘Monitoring of Indonesian volcanoes with the IS06 infrasound array’, *Journal of Volcanology and Geothermal Research*, 434, p. 107753. Available at: https://doi.org/10.1016/j.jvolgeores.2023.107753.

Global Volcanism Program and Venzke, E. (2023) ‘Volcanoes of the World, v.5.1.0’. Global Volcanism Program.

Gusev, A.A. (2014) ‘The fractal structure of the sequence of volcanic eruptions worldwide: Order clustering of events and episodic discharge of material’, *Journal of Volcanology and Seismology*, 8(1), pp. 34–53. Available at: https://doi.org/10.1134/S0742046314010023.

Hagerty, M. and Benites, R. (2003) ‘Tornillos beneath Tongariro Volcano, New Zealand’, *Journal of Volcanology and Geothermal Research*, 125, pp. 151–169. Available at: https://doi.org/10.1016/S0377-0273(03)00094-5.

Huang, X., Hu, Z. and Lin, L. (2023) ‘Deep clustering based on embedded auto-encoder’, *Soft Computing*, 27(2), pp. 1075–1090. Available at: https://doi.org/10.1007/s00500-021-05934-8.

Jacks, E., Davidson, J. and H. G., W. (2010) *Guidelines of Early Warning Systems and Application of Nowcasting and Warning Operations*. World Meteorological Organization.

Jenkins, W.F. *et al.* (2021) ‘Unsupervised Deep Clustering of Seismic Data: Monitoring the Ross Ice Shelf, Antarctica’, *Journal of Geophysical Research: Solid Earth*, 126(9), p. e2021JB021716. Available at: https://doi.org/10.1029/2021JB021716.

John, W.E., Guffanti, M. and Thomas, L.M. (2005) *An Assessment of Volcanic Threat and Monitoring Capabilities in the United States: Framework for a National Volcano Early Warning System*. Open-File Report.

Kim, K., Lees, J.M. and Ruiz, M. (2012) ‘Acoustic multipole source model for volcanic explosions and inversion for source parameters’, *Geophysical Journal International*, 191(3), pp. 1192–1204. Available at: https://doi.org/10.1111/j.1365-246X.2012.05696.x.

Kurokawa, A.K. and Ichihara, M. (2020) ‘Identification of infrasonic and seismic components of tremors in single-station records: application to the 2013 and 2018 events at Ioto Island, Japan’, *Earth, Planets and Space*, 72(1), p. 171. Available at: https://doi.org/10.1186/s40623-020-01302-2.

Le Pichon, A. *et al.* (2009) ‘Assessing the performance of the International Monitoring System’s infrasound network: Geographical coverage and temporal variabilities’, *Journal of Geophysical Research: Atmospheres*, 114(D8). Available at: https://doi.org/10.1029/2008JD010907.

Le Pichon, A. *et al.* (2021) ‘Using dense seismo-acoustic network to provide timely warning of the 2019 paroxysmal Stromboli eruptions’, *Scientific Reports 2021 11:1*, 11(1), pp. 1–12. Available at: https://doi.org/10.1038/s41598-021-93942-x.

Le Pichon, A., Blanc, E. and Hauchecorne, A. (eds) (2019) *Infrasound Monitoring for Atmospheric Studies: Challenges in Middle Atmosphere Dynamics and Societal Benefits*. Cham: Springer International Publishing. Available at: https://doi.org/10.1007/978-3-319-75140-5.

Lighthill, M.J. and Newman, M.H.A. (1997) ‘On sound generated aerodynamically I. General theory’, *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 211(1107), pp. 564–587. Available at: https://doi.org/10.1098/rspa.1952.0060.

Machacca Puma, R. *et al.* (2021) ‘Monitoring of active volcanoes in Peru by the Instituto Geofísico del Perú’, *Volcanica*, 4(S1), pp. 49–71. Available at: https://doi.org/10.30909/vol.04.S1.4971.

MacQueen, J. (1967) ‘Some methods for classification and analysis of multivariate observations’, in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. University of California Press, pp. 281–298. Available at: https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-Fifth-Berkeley-Symposium-on-Mathematical-Statistics-and/chapter/Some-methods-for-classification-and-analysis-of-multivariate-observations/bsmsp/1200512992 (Accessed: 12 February 2024).

Marzocchi, W., Sandri, L. and Selva, J. (2008) ‘BET_EF: a probabilistic tool for long- and short-term eruption forecasting’, *Bulletin of Volcanology*, 70(5), pp. 623–632. Available at: https://doi.org/10.1007/s00445-007-0157-y.

Matoza, R.S. *et al.* (2017) ‘Seismo-acoustic wavefield of strombolian explosions at Yasur volcano, Vanuatu, using a broadband seismo-acoustic network, infrasound arrays, and infrasonic sensors on tethered balloons’, *The Journal of the Acoustical Society of America*, 141(5_Supplement), p. 3566. Available at: https://doi.org/10.1121/1.4987573.

McNutt, S.R. *et al.* (2015) ‘Seismic and Infrasonic Monitoring’, in *The Encyclopedia of Volcanoes*. Elsevier, pp. 1071–1099. Available at: https://doi.org/10.1016/B978-0-12-385938-9.00063-8.

Mezyk, M. and Malinowski, M. (2021) ‘Deep Embedded Clustering As a Seismic Attribute: A Case Study of 2D Crustal-Scale Interpretation’, in. *82nd EAGE Annual Conference & Exhibition*, European Association of Geoscientists & Engineers, pp. 1–5. Available at: https://doi.org/10.3997/2214-4609.202010681.

Modrak, R.T., Arrowsmith, S.J. and Anderson, D.N. (2010) ‘A Bayesian framework for infrasound location’, *Geophysical Journal International*, 181(1), pp. 399–405. Available at: https://doi.org/10.1111/j.1365-246X.2010.04499.x.

Montalto, P. (2010) *Insights into Mt. Etna volcano dynamics by seismic and infrasonic signals*. Universita’ degli Studi di Catania.

Mousavi, S.M. and Beroza, G.C. (2020) ‘A Machine-Learning Approach for Earthquake Magnitude Estimation’, *Geophysical Research Letters*, 47(1), p. e2019GL085976. Available at: https://doi.org/10.1029/2019GL085976.

Muhari, A. *et al.* (2019) ‘The December 2018 Anak Krakatau Volcano Tsunami as Inferred from Post-Tsunami Field Surveys and Spectral Analysis’, *Pure and Applied Geophysics*, 176(12), pp. 5219–5233. Available at: https://doi.org/10.1007/s00024-019-02358-2.

Perttu, A. *et al.* (2020) ‘Reconstruction of the 2018 tsunamigenic flank collapse and eruptive activity at Anak Krakatau based on eyewitness reports, seismo-acoustic and satellite observations’, *Earth and Planetary Science Letters*, 541. Available at: https://doi.org/10.1016/J.EPSL.2020.116268.

Poland, M.P. and Anderson, K.R. (2020) ‘Partly Cloudy With a Chance of Lava Flows: Forecasting Volcanic Eruptions in the Twenty-First Century’, *Journal of Geophysical Research: Solid Earth*, 125(1), p. e2018JB016974. Available at: https://doi.org/10.1029/2018JB016974.

Ripepe, M. *et al.* (2018) ‘Infrasonic Early Warning System for Explosive Eruptions’, *Journal of Geophysical Research: Solid Earth*, 123(11), pp. 9570–9585. Available at: https://doi.org/10.1029/2018JB015561.

Rose, K.M. (2020) *Remote Hydroacoustic and Infrasonic Detection and Characterization of Anak Krakatau Eruptive Activity Leading To, During, and Following the December 2018 Major Flank Collapse and Tsunami*. University of California Santa Barbara.

Smart, E. and Flinn, E.A. (1971) ‘Fast Frequency-Wavenumber Analysis and Fisher Signal Detection in Real-Time Infrasonic Array Data Processing’, *Geophysical Journal International*, 26(1–4), pp. 279–284. Available at: https://doi.org/10.1111/j.1365-246X.1971.tb03401.x.

Steinbach, M., Ertöz, L. and Kumar, V. (2004) ‘The Challenges of Clustering High Dimensional Data’, in L.T. Wille (ed.) *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition*. Berlin, Heidelberg: Springer, pp. 273–309. Available at: https://doi.org/10.1007/978-3-662-08968-2_16.

Triyangoro, A.P. and Ontowirjo, B. (2021) ‘Anak Krakatau Landslide Tsunami Relapse Potential Hazard’, *IOP Conference Series: Earth and Environmental Science*, 698(1), p. 012025. Available at: https://doi.org/10.1088/1755-1315/698/1/012025.

Walter, T.R. *et al.* (2019) ‘Complex hazard cascade culminating in the Anak Krakatau sector collapse’, *Nature Communications*, 10(1), p. 4339. Available at: https://doi.org/10.1038/s41467-019-12284-5.

Watson, L.M. *et al.* (2022) ‘Volcano infrasound: progress and future directions’, *Bulletin of Volcanology 2022 84:5*, 84(5), pp. 1–13. Available at: https://doi.org/10.1007/S00445-022-01544-W.

Woulff, G. and McGetchin, T.R. (1976) ‘Acoustic Noise from Volcanoes: Theory and Experiment’, *Geophysical Journal International*, 45(3), pp. 601–616. Available at: https://doi.org/10.1111/j.1365-246X.1976.tb06913.x.

Xie, J., Girshick, R. and Farhadi, A. (2016) ‘Unsupervised Deep Embedding for Clustering Analysis’. arXiv. Available at: https://doi.org/10.48550/arXiv.1511.06335.

Zali, Z. (2023) *Volcanic tremor analysis based on advanced signal processing concepts including music information retrieval (MIR) strategies* [application/pdf]. Universität Potsdam. Available at: https://doi.org/10.25932/PUBLISHUP-61086.

APPENDIX

A. Waveform Collection, Downloader & Converter code

Downloads I06AU and I52GB waveforms from IRIS FSDN services using Obpsy Mass Downloader and then converts them the .SAC file with it's stats for further processing.

This notebook consists of 3 parts:

1. Dowloading Data using Obspy Mass Downloader
2. Conversion of MSEED Files to .SAC
3. Conversion of .SAC files to CSV

```
import sys
sys.path.append('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/')

import obspy
from obspy import UTCDateTime, read, read_inventory
from obspy.clients.fdsn.mass_downloader import GlobalDomain, \
    Restrictions, MassDownloader
from obspy.core.util.attribdict import AttribDict
import os
import subprocess
import glob
from tqdm.notebook import tqdm
from tqdm import tqdm
from src.utils.converter import *
import concurrent.futures
from concurrent.futures import ThreadPoolExecutor
```

Downloading Data Using Obspy Mass Downloader

Download the data ranging from 2018-06-24T00:00:00 until 2019-09-03T00:00:00

```
# 1.1 I06AU Waveform Download
domain = GlobalDomain()

restrictions = Restrictions(
    # Get data for a whole year.
    starttime=obspy.UTCDateTime(2018, 6, 24),
    endtime=obspy.UTCDateTime(2019, 9, 3),
    # Chunk it to have one file per day.
    chunklength_in_sec=86400,
    network="IM", station="I06H*", location="", channel="BDF",
    # The typical use case for such a data set are noise correlations where
    # gaps are dealt with at a later stage.
    reject_channels_with_gaps=False,
    # Same is true with the minimum length. All data might be useful.
    minimum_length=0.0,
    # Guard against the same station having different names.
    minimum_interstation_distance_in_m=100.0)

mdl = MassDownloader(providers=["IRIS"])
mdl.download(domain, restrictions, mseed_storage="waveform_collection/I06AU/WAVEFORM_I06AU_
MSEED",
             stationxml_storage="waveform_collection/I06AU/I06AU_STATIONS")
```

```
# 1.2 I52GB Waveform Download
domain = GlobalDomain()

restrictions = Restrictions(
    starttime=obspy.UTCDateTime(2018, 6, 24),
    endtime=obspy.UTCDateTime(2019, 9, 3),
    chunklength_in_sec=86400,
    network="IM", station="I52H*", location="", channel="BDF",
    reject_channels_with_gaps=False,
    minimum_length=0.0,
    minimum_interstation_distance_in_m=100.0)


mdl = MassDownloader(providers=["IRIS"])
mdl.download(domain, restrictions, mseed_storage="/run/media/viblab/Markov2/Haykal/AnakKrak
atauEWS/data/raw/I52GB/I52GB_MSEED",
             stationxml_storage="/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/
I52GB/I52GB_STATIONS")
```

Conversion of MSEED Files to .SAC

Converting MSEED Files to SAC Complete with Important Headers

```
# 2.1 I06AU Waveform Conversion
input_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I06AU/I06AU_MS
EED'
output_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I06AU/I06AU_S
AC'
stationxml_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I06AU/I06
AU_STATIONS'

# Run the Function
mseed_to_sac(input_directory, output_directory, stationxml_directory)



# 2.2 I52GB Waveform Conversion
input_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I52GB/I52GB_MS
EED'
output_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I52GB/I52GB_S
AC'
stationxml_directory = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I52GB/I52
GB_STATIONS'

# Run the Function
mseed_to_sac(input_directory, output_directory, stationxml_directory)
```

B. Waveform Processing code

The .SAC files will be then further processed using "Bartlett" beamforming method and Adaptive F-Detector to identify signals most related to Anak Krakatau Volcanic Activity. These specific volcanic activity signals will be later ingested into the algorithm for the Deep Learning processing. This notebook will do as follows:

4. Implementing FK Beamforming to each station group
5. Running Adaptive F-Detector to Isolate relevant Waveform
6.

```
import sys
sys.path.append('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/')

import obspy
from obspy import UTCDateTime, read, read_inventory
from obspy.clients.fdsn.mass_downloader import GlobalDomain, \
    Restrictions, MassDownloader
from obspy.core.util.attribdict import AttribDict
import os
import subprocess
import glob
from tqdm.notebook import tqdm
from tqdm import tqdm
from src.utils.converter import *
import concurrent.futures
from concurrent.futures import ThreadPoolExecutor
```

FK Beamforming using Bartlett method

```
# 4.1 Running Beamforming on I06AU Waveform
base_folder_path = '/run/media/viblab/Markov11/Haykal/AnakKrakatauEWS/waveform_collection/I
06AU/WAVEFORM_I06AU_SAC'
df_grouped_file_paths = sac_path_grouping(base_folder_path)

# Wrap the iterrows with tqdm to create a progress bar
for index, row in tqdm(df_grouped_file_paths.iterrows(), total=df_grouped_file_paths.shape[
0], desc="Processing"):
    # Construct the full path to the grouped files
    full_grouped_path = f"{base_folder_path}/{row['GroupedFilePath']}"

    # Prepare the CLI command
    cli_command = f"infrapy run_fk --local-wvfrms \"{full_grouped_path}\" --freq-min 0.7 --
freq-max 4 --back-az-min 50 --back-az-max 60 --window-len 600 --sub-window-len 30 --window-
step 300"

    try:
        # Execute the CLI command
        subprocess.run(cli_command, shell=True, check=True)
        print(f"Command executed for: {full_grouped_path}")
    except subprocess.CalledProcessError as e:
        print(f"An error occurred while running the command for {full_grouped_path}: {e}")

# 4.1 Running Beamforming on I52GB Waveform
base_folder_path = '/run/media/viblab/Markov11/Haykal/AnakKrakatauEWS/waveform_collection/I
52GB/WAVEFORM_I52GB_SAC'
df_grouped_file_paths = sac_path_grouping(base_folder_path)

# Wrap the iterrows with tqdm to create a progress bar
for index, row in tqdm(df_grouped_file_paths.iterrows(), total=df_grouped_file_paths.shape[
```

```
0], desc="Processing"):
    # Construct the full path to the grouped files
    full_grouped_path = f"{base_folder_path}/{row['GroupedFilePath']}"

    # Prepare the CLI command
    cli_command = f"infrapy run_fk --local-wvfrms \"{full_grouped_path}\" --"

    try:
        # Execute the CLI command
        subprocess.run(cli_command, shell=True, check=True)
        print(f"Command executed for: {full_grouped_path}")
    except subprocess.CalledProcessError as e:
        print(f"An error occurred while running the command for {full_grouped_path}: {e}")
```

Running Adaptive F-Detector

```
# 5.2 Running AFD for I06AU Station
folder_path = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I06AU/I06AU_FK_RES
ULTS'

# List all files in the folder
files = os.listdir(folder_path)
full_paths = [os.path.join(folder_path, file) for file in files]

# Using ThreadPoolExecutor to process files concurrently
with ThreadPoolExecutor(8) as executor:
    # Map the executor to the process_file function and the list of file paths
    results = list(tqdm(executor.map(lambda file_path: subprocess.run(f"infrapy run_fd --lo
cal-fk-label {file_path} --p-value 0.05 --back-az-width 5 --min-duration 25 --merge-dets 'F
alse'", shell=True), full_paths), total=len(full_paths), desc="Processing files"))

# Path to the folder containing the files
folder_path = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/data/raw/I52GB/I52GB_FK'

# List all files in the folder
files = os.listdir(folder_path)
full_paths = [os.path.join(folder_path, file) for file in files]

# Using ThreadPoolExecutor to process files concurrently
with ThreadPoolExecutor(8) as executor:
    # Map the executor to the process_file function and the list of file paths
    results = list(tqdm(executor.map(lambda file_path: subprocess.run(f"infrapy run_fd --lo
cal-fk-label {file_path} --p-value 0.05 --back-az-width 5", shell=True), full_paths), total
=len(full_paths), desc="Processing files"))
```

Laslo Best Beam Calculation

```
import subprocess
from concurrent.futures import ThreadPoolExecutor
import pandas as pd
from tqdm.notebook import tqdm

# Load the CSV file
file_path = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWS/notebook/Best-Beam_ITeration/
BEST-BEAM_DATABASE_Unique_6.csv'  # Replace with your actual CSV file path
df = pd.read_csv(file_path)

# Function to generate CLI command for each row
def generate_command(row):
    return (
        f"infrapy utils best-beam --local-wvfrms '{row['SACFile']}' "
```

```
        f"--local-fk-label '{row['FKResults']}' --signal-start '{row['Start']}' "
        f"--signal-end '{row['End']}' --trace-vel {row['Trace Velocity']} "
        f"--back-az {row['Back Azimuth']} --freq-min {row['Freq Min']} "
        f"--freq-max {row['Freq Max']} --hold-figure False"
    )

# Generating CLI commands for each row
cli_commands = df.apply(generate_command, axis=1).tolist()

# Function to execute a command
def run_command(cmd):
    try:
        subprocess.run(cmd, shell=True, check=True, stdout=subprocess.PIPE, stderr=subproce
ss.PIPE)
        return "Success"
    except subprocess.CalledProcessError as e:
        return f"Failed: {e}"
```

C. Adaptive F-Detector Results

Time (UTC)	F Stat.	Trace Vel. (m/s)	Back Azimuth	Latitude	Longitude	Start	End	Station
02/12/2018 00:53	44395	35527	54,27	-1214645	9682032	-300	150	I06H
02/12/2018 01:10	14221	35426	55,5	-1214645	9682032	0	150	I06H
02/12/2018 01:25	1599	35607	54,3	-1214645	9682032	0	150	I06H
02/12/2018 01:35	20105	35408	54,17	-1214645	9682032	-150	0	I06H
02/12/2018 02:10	26067	35441	54,01	-1214645	9682032	-300	150	I06H
02/12/2018 02:40	2761	35719	53,51	-1214645	9682032	0	450	I06H
02/12/2018 04:20	42118	35488	54,4	-1214645	9682032	-150	150	I06H
02/12/2018 05:43	18404	35538	55,61	-1214645	9682032	0	150	I06H
02/12/2018 06:05	16786	3597	54,44	-1214645	9682032	-150	0	I06H
02/12/2018 08:03	13493	36608	54,7	-1214645	9682032	-150	0	I06H
02/12/2018 08:13	27379	35514	55,21	-1214645	9682032	-150	0	I06H
02/12/2018 09:05	20693	35438	54,57	-1214645	9682032	0	300	I06H
02/12/2018 09:40	15282	35285	54,5	-1214645	9682032	-150	0	I06H
02/12/2018 10:03	1809	35615	55,27	-1214645	9682032	0	150	I06H
02/12/2018 10:55	18133	35355	55,07	-1214645	9682032	-150	150	I06H
02/12/2018 17:00	29685	35885	53,56	-1214645	9682032	-300	150	I06H
02/12/2018 17:25	26947	35338	54,38	-1214645	9682032	-150	150	I06H
02/12/2018 18:25	15409	35114	53,45	-1214645	9682032	-150	0	I06H
02/12/2018 18:53	26009	35379	55,7	-1214645	9682032	-600	450	I06H
02/12/2018 19:08	1926	35331	54,28	-1214645	9682032	-150	150	I06H
02/12/2018 19:40	22295	35538	55,1	-1214645	9682032	-150	150	I06H
02/12/2018 20:03	20812	35748	54,1	-1214645	9682032	-150	150	I06H
02/12/2018 20:13	18073	35477	54,09	-1214645	9682032	0	150	I06H
02/12/2018 20:23	15829	36029	54,09	-1214645	9682032	0	150	I06H
02/12/2018 21:30	21814	35579	54,51	-1214645	9682032	-600	150	I06H
02/12/2018 21:43	14478	35326	54,45	-1214645	9682032	-150	0	I06H
02/12/2018 22:13	14268	35013	55,39	-1214645	9682032	-150	0	I06H
02/12/2018 23:15	16903	35465	54,25	-1214645	9682032	-150	0	I06H
02/12/2018 23:45	16522	3541	55,15	-1214645	9682032	-150	0	I06H

D. Autoencoder Python Code

```
import numpy as np
import h5py
import os, re, glob
import math
from scipy import signal
from scipy.signal import butter, lfilter
from keras.layers import Input, Dense, Conv2D, MaxPooling2D, UpSampling2D, Conv1D, MaxPooli
ng1D, UpSampling1D, Flatten, Dropout, Reshape
from keras.layers import Bidirectional, BatchNormalization, ZeroPadding1D, Conv2DTranspose
from keras.models import Model
from keras import backend as K
from tensorflow.keras.optimizers import SGD, Adam, schedules
from keras import regularizers
from tensorflow.keras.layers import Layer, InputSpec
```

```
from keras.callbacks import ModelCheckpoint, LearningRateScheduler, ReduceLROnPlateau, Earl
yStopping
from keras.initializers import VarianceScaling
from keras.callbacks import CSVLogger
from scipy.optimize import linear_sum_assignment as linear_assignment
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from matplotlib.pyplot import savefig
import seaborn as sns
from keras.utils import plot_model
from IPython.display import Image, display
import tensorflow as tf

sns.set_style('darkgrid')
sns.set_palette('muted')

import tensorflow as tf
gpus = tf.config.list_physical_devices('GPU')
if gpus:
  try:
    # Currently, memory growth needs to be the same across GPUs
    for gpu in gpus:
      tf.config.experimental.set_memory_growth(gpu, True)
    logical_gpus = tf.config.list_logical_devices('GPU')
    print(len(gpus), "Physical GPUs,", len(logical_gpus), "Logical GPUs")
  except RuntimeError as e:
    # Memory growth must be set before GPUs have been initialized
    print(e)
```

Autoencoder Architecture

```
from numpy.random import seed
seed(46)

import tensorflow as tf
tf.random.set_seed(46)

from keras.layers import Input, Conv2D, MaxPooling2D, Flatten, Reshape, UpSampling2D
from keras.models import Model, Sequential
import keras.backend as K

initializer = tf.keras.initializers.GlorotUniform(seed=46)

# Change input shape
inp = Input(shape=(128, 32, 1))

# Adjust encoder layers
e = Conv2D(8, (7, 5),strides=[2,2], activation='elu', kernel_initializer=initializer, paddi
ng='same')(inp)
e = Conv2D(16, (5, 3),strides=[2,2], activation='elu', kernel_initializer=initializer, padd
ing='same')(e)
e = Conv2D(32, (5, 3),strides=[2,2], activation='elu', kernel_initializer=initializer, padd
ing='same')(e)
e = Conv2D(64, (5, 3),strides=[2,2], activation='elu' , kernel_initializer=initializer, pad
ding='same')(e)

# Get shape before flattening
shape_before_flattening = K.int_shape(e)

# Encode to a dense layer
encoded1 = Flatten()(e)
encoded2 = Dense(24, activation="elu")(encoded1)
# Remove redundant dense layer
fc = Dense(np.prod(shape_before_flattening[1:]), activation="elu")(encoded2)
```

```
# Decoder Layers
d = Reshape(shape_before_flattening[1:])(fc)

d = Conv2DTranspose(32, (5, 3), strides=[2,2], activation='elu', kernel_initializer=initial
izer, padding='same')(d)
d = Conv2DTranspose(16, (5, 3), strides=[2,2], activation='elu', kernel_initializer=initial
izer, padding='same')(d)
d = Conv2DTranspose(8, (5, 3),strides=[2,2] , activation='elu', kernel_initializer=initiali
zer, padding='same')(d)
decoded = Conv2DTranspose(1, (7, 5),strides=[2,2] , activation='linear', kernel_initializer
=initializer, padding='same')(d)

autoencoder = Model(inputs=inp, outputs=decoded, name="autoencoder")
encoder = Model(inputs=inp, outputs=encoded2, name="encoder")

autoencoder.summary()
```

Data Loading

```
from sklearn.model_selection import train_test_split

# Access your file using the path
data = np.load('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Preprocessed/Tra
in_Test_Scipy.npy')
# Split the data into training and testing sets
train_data, test_data = train_test_split(data, test_size=0.2, random_state=46, shuffle=True
)
print ('size-data='+str (data.shape), 'size-train='+str (train_data.shape),'size-test='+str
(test_data.shape) )
```

Training

```
lr_schedule = schedules.ExponentialDecay(
    initial_learning_rate= 0.001,
    decay_steps=1000,
    decay_rate=0.8)

### Adapting the learning rate of the optimizer using an exponential decay schedule

optimizer = Adam(learning_rate=lr_schedule)

es = EarlyStopping( monitor='val_loss', mode='min', verbose=1, patience=30), CSVLogger('Pre
train_log.csv')

autoencoder.compile(optimizer=optimizer, loss='mse')
autoencoder.fit(train_data, train_data, batch_size=32, epochs=1000 ,validation_data=(test_d
ata, test_data), callbacks=[es])

import pandas as pd
df = pd.read_csv('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/notebook/Pretrain_l
og.csv')
fig= plt.figure(figsize=(7, 5))
plt.plot(df['epoch'],df['loss'], color='b',label='Training Loss', linewidth=3.0)
plt.plot(df['epoch'],df['val_loss'], color='darkorange',label='Testing Loss', linewidth=3.0
)

plt.ylabel('Loss', fontsize= 18)
plt.xlabel('Epoch', fontsize= 18)
plt.title('Reconstruction loss of the autoencoder', fontsize= 18)
plt.yticks (fontsize= 18)
plt.xticks (fontsize= 18)

plt.legend(loc= 1, frameon= False, fontsize= 18)
```

```
plt.tight_layout()
plt.show ()

### Save the model
from tensorflow.keras.models import save_model
autoencoder.save ('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/Models/autoencoder
-model2')
```

Kmeans clustering based on extracted features from the autoencoder

```
from tensorflow.keras.models import load_model

autoencoder = load_model('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/Models/auto
encoder-model2')
autoencoder_input = autoencoder.input
encoder_output = autoencoder.get_layer('dense_1').output
encoder = Model(inputs=autoencoder_input, outputs=encoder_output)

kmeans = KMeans(n_clusters=1, random_state=46, n_init=20).fit(encoder.predict(data))
y = kmeans.predict(encoder.predict(data))

def plotter(S, y):
    '''
    function to visualize the outputs of t-SNE
    '''

    lw = 2
    # create a scatter plot.
    f = plt.figure(figsize=(22, 10))
    ax = f.add_subplot(111)
    plt.scatter(S[y == 0, 0], S[y == 0, 1],color='navy', alpha=.5, lw=lw, s=100)
    ax.axis('off')
    ax.axis('tight')
    plt.show()

    return f, ax

enc = encoder.predict(data)
from sklearn.manifold import TSNE
redu = TSNE(random_state=123).fit_transform(enc)
plotter(redu, y)
```

Determining optimal number of clusters

```
from sklearn.metrics import calinski_harabasz_score
cal = []
K = range(2,16)
for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=20).fit(encoder.predict(data))
    labelskm = kmeans.predict(encoder.predict(data))
    cal.append(calinski_harabasz_score(encoder.predict(data), labelskm))
fig= plt.figure(figsize=(7, 5))
plt.plot(K, cal, 'bx-')
plt.xlabel('Number of clusters',fontsize= 18)
plt.ylabel('Calinski-Harabasz score',fontsize= 18)
plt.title('Calinski-Harabasz Score Elbow for K-means Clustering',fontsize= 18)
plt.yticks (fontsize= 18)
plt.xticks (fontsize= 18)
plt.axvline(x = 3, color = 'black')
plt.tight_layout()
plt.show()

from sklearn.metrics import silhouette_score
from sklearn.cluster import KMeans
```

```
silhouette_scores = []
K = range(2, 16)

for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=20).fit(encoder.predict(data))
    labels = kmeans.labels_
    silhouette_scores.append(silhouette_score(encoder.predict(data), labels))

# Plotting the silhouette scores
fig = plt.figure(figsize=(7, 5))
plt.plot(K, silhouette_scores, 'bx-')
plt.xlabel('Number of clusters', fontsize=18)
plt.ylabel('Silhouette score', fontsize=18)
plt.title('Silhouette Score for K-means Clustering', fontsize=18)
plt.xticks(fontsize=18)
plt.yticks(fontsize=18)
plt.axvline(x=silhouette_scores.index(max(silhouette_scores)) + 2, color='black')  # +2 bec
ause range starts at 2
plt.tight_layout()
plt.show()

from tensorflow.keras.models import load_model
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from yellowbrick.cluster import SilhouetteVisualizer
import matplotlib.pyplot as plt

# Calculate the encoded features only once
encoded_data = encoder.predict(data)

# Generate encoded data
encoded_data = encoder.predict(data)

# Range of clusters to evaluate
K = range(2, 6)
silhouette_avg_scores = []  # To store the average silhouette scores

print("Silhouette scores for different numbers of clusters:")
for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=20).fit(encoded_data)
    labels = kmeans.labels_
    score = silhouette_score(encoded_data, labels)
    silhouette_scores.append(score)
    print(f"Clusters: {k}, Silhouette Score: {score:.4f}")

# Create subplots for silhouette visualizers
fig, axes = plt.subplots(len(K), 1, figsize=(7, 5*len(K)))

# Loop over the range of cluster numbers to fit KMeans and calculate silhouette scores
for idx, k in enumerate(K):
    # Fit KMeans and predict the cluster labels
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=20)
    cluster_labels = kmeans.fit_predict(encoded_data)

    # Calculate the average silhouette score and append to the list
    silhouette_avg = silhouette_score(encoded_data, cluster_labels)
    silhouette_avg_scores.append(silhouette_avg)

    # Initialize the SilhouetteVisualizer with the KMeans model
    visualizer = SilhouetteVisualizer(kmeans, colors='yellowbrick', ax=axes[idx])
    visualizer.fit(encoded_data)
    axes[idx].set_title(f'Silhouette Plot for k={k}', fontsize=12)

# Adjust the layout
```

```
plt.tight_layout()

# Plot the average silhouette scores
fig, ax = plt.subplots(figsize=(7, 5))
ax.plot(K, silhouette_avg_scores, 'bx-')
ax.set_xlabel('Number of clusters', fontsize=18)
ax.set_ylabel('Average silhouette score', fontsize=18)
ax.set_title('Silhouette Score Elbow for K-means Clustering', fontsize=18)
ax.axvline(x=K[silhouette_avg_scores.index(max(silhouette_avg_scores))], color='black', lin
estyle='--')
ax.tick_params(axis='x', labelsize=18)  # Corrected line for setting x-tick label sizes
ax.tick_params(axis='y', labelsize=18)  # Corrected line for setting y-tick label sizes
plt.tight_layout()
plt.show()
```

T-sne visualizations of seismic event clusters in feature domain after pretraining using K-Means

```
kmeans = KMeans(n_clusters=3, random_state=46, n_init=20).fit(encoder.predict(data))
y = kmeans.predict(encoder.predict(data))

def plotter(S, y, target_names):
    '''
    function to visualize the outputs of t-SNE
    '''
    # choose a color palette with seaborn.
    colors = ['green','darkorange','red']

    lw = 2
    # create a scatter plot.
    f = plt.figure(figsize=(22, 10))
    ax = f.add_subplot(111)
    for color, i, target_name in zip(colors, [0, 1, 2], target_names):
      plt.scatter(S[y == i, 0], S[y == i, 1], color=color, alpha=.5, lw=lw, s=100, label=ta
rget_name)
    plt.legend(loc='lower left', shadow=False, scatterpoints=1, prop={'size': 26})
    ax.axis('off')
    ax.axis('tight')
    plt.show()

    return f, ax

enc = encoder.predict(data)
from sklearn.manifold import TSNE
redu = TSNE(random_state=123).fit_transform(enc)
target_names = ['cluster 1', 'cluster 2', 'cluster 3']

# Print the cluster counts
(unique, counts) = np.unique(y, return_counts=True)
cluster_counts = dict(zip(unique, counts))
print("Cluster counts:", cluster_counts)

plotter(redu, y, target_names)
```

Loading the Pre-trained Model

```
from tensorflow.keras.models import load_model
autoencoder = load_model("/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/Models/auto
encoder-model2")

import keras.backend as K
get_all_layer_outputs = K.function([autoencoder.layers[0].input],
                                   [l.output for l in autoencoder.layers[1:]])

layer_output = get_all_layer_outputs([data]) # return the same thing

n_clusters=3
```

Integrating clustering layer into autoencoder bottelneck

```
from numpy.random import seed
sd=46
seed(sd)
import tensorflow
import tensorflow as tf
tensorflow.random.set_seed(sd)

initializer = tf.keras.initializers.GlorotUniform(seed=sd)

#### clustering layers
class ClusteringLayer(Layer):
    def __init__(self, n_clusters, weights=None, alpha=1, **kwargs):
        if 'input_shape' not in kwargs and 'input_dim' in kwargs:
            kwargs['input_shape'] = (kwargs.pop('input_dim'),)
        super(ClusteringLayer, self).__init__(**kwargs)
        self.n_clusters = n_clusters
        self.alpha = alpha
        self.initial_weights = weights
        self.input_spec = InputSpec(ndim=2)

    def build(self, input_shape):
        assert len(input_shape) == 2
        input_dim = input_shape[1]
        self.input_spec = InputSpec(dtype=K.floatx(), shape=(None, input_dim))
        self.clusters = self.add_weight(shape=(self.n_clusters, input_dim), initializer= in
itializer , name='clusters')
        if self.initial_weights is not None:
            self.set_weights(self.initial_weights)
            del self.initial_weights

    def call(self, inputs, **kwargs):
        q = 1.0 / (1.0 + (K.sum(K.square(K.expand_dims(inputs, axis=1) - self.clusters), ax
is=2) / self.alpha))
        q **= (self.alpha + 1.0) / 2.0
        q = K.transpose(K.transpose(q) / K.sum(q, axis=1))
        return q

    def compute_output_shape(self, input_shape):
        assert input_shape and len(input_shape) == 2
        return input_shape[0], self.n_clusters

    def get_config(self):
        config = {'n_clusters': self.n_clusters}
        base_config = super(ClusteringLayer, self).get_config()
        return dict(list(base_config.items()) + list(config.items()))


print('...Finetuning...')
clustering_layer = ClusteringLayer(n_clusters, name='clustering')(autoencoder.layers[6].out
put)
model = Model(inputs=autoencoder.layers[0].output, outputs=clustering_layer)
model.compile(loss='kld', loss_weights=0.1,  optimizer=SGD(learning_rate=0.01,  momentum=0.
9))

...Finetuning...

### Initializing the weights using Kmean and assigning them to the model

kmeans = KMeans(n_clusters=n_clusters,  random_state=46, n_init=20)
y_pred = kmeans.fit_predict(layer_output[5])
y_pred_last = np.copy(y_pred)
model.get_layer(name='clustering').set_weights([kmeans.cluster_centers_])
```

Finetuning Pre-Trained Model Parameters

```
## parameters for the finetuning
x = data
batch_size=32
tol = 0.0001 # tolerance threshold to stop training
loss = 0
index = 0
maxiter = 100000
update_interval = 500
index_array = np.arange(x.shape[0])

################################################################################
### simultaneous optimization and clustering
def target_distribution(q):
    weight = q ** 2 / q.sum(0)
    return (weight.T / weight.sum(1)).T

for ite in range(int(maxiter)):
    if ite % update_interval == 0:
        q = model.predict(data, verbose=0)
        p = target_distribution(q)  # update the auxiliary target distribution p
        y_pred = q.argmax(1) # evaluate the clustering performance

        loss = np.round(loss, 5)
        print('Iter %d: ' % (ite), ' ; loss=', loss)

        # check stop criterion
        delta_label = np.sum(y_pred != y_pred_last).astype(np.float32) / y_pred.shape[0]
        y_pred_last = np.copy(y_pred)

        if ite > 0 and delta_label < tol:
            print('delta_label ', delta_label, '< tol ', tol)
            break

        IN = layer_output[5]

    idx = index_array[index * batch_size: min((index+1) * batch_size, data.shape[0])]
    loss = model.train_on_batch(x=x[idx], y=p[idx])
    index = index + 1 if (index + 1) * batch_size <= x.shape[0] else 0

import keras.backend as K
get_all_layer_outputs = K.function([autoencoder.layers[0].input],
                                   [l.output for l in autoencoder.layers[1:]])

layer_output = get_all_layer_outputs([x]) # return the same thing

autoencoder.save('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/Models/autoencoder-
model_finetuned')
```

T-sne visualizations of seismic event clusters in feature domain after finetuning

```
y=y_pred

def plotter(S, y, target_names):
    '''
    function to visualize the outputs of t-SNE
    '''
    # choose a color palette with seaborn.
    colors = ['red','darkorange','green']

    lw = 2
    # create a scatter plot.
    f = plt.figure(figsize=(22, 10))
    ax = f.add_subplot(111)
```

```
    for color, i, target_name in zip(colors, range(len(target_names)), target_names):
        plt.scatter(S[y == i, 0], S[y == i, 1], color=color, alpha=.8, lw=lw, s=100, label=
target_name)
    plt.legend(loc='lower left', shadow=False, scatterpoints=1, prop={'size': 26})
    ax.axis('off')
    ax.axis('tight')
    plt.show()

    return f, ax

enc = layer_output[5]
from sklearn.manifold import TSNE
redu = TSNE(random_state=123).fit_transform(enc)
target_names = ['EA', 'CT2', 'CT1']

# Print the cluster counts
(unique, counts) = np.unique(y, return_counts=True)
cluster_counts = dict(zip(unique, counts))
print("Cluster counts:", cluster_counts)

plotter(redu, y, target_names)
```

Rewriting Cluser Result for Temporal Analysis

```
### Save the labels
np.savetxt('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Km-n3-ft.txt', y, fm
t='%i', delimiter=',')

# Change the order of the cluster numbers (just for a nice representation)

with open('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Km-n3-ft.txt', 'r') a
s file :
  filedata = file.read()

# Replace the target string
filedata = filedata.replace('2', 'x')
filedata = filedata.replace('0', '2')
filedata = filedata.replace('1', '0')
filedata = filedata.replace('x', '1')

# Re-write the output
with open('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Km-n3-ft_Rewritten.tx
t', 'w') as file:
  file.write(filedata)
```

Adding new unforseen data

Validation Data Loading

```
new_data = np.load('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Preprocessed
/valid_Scipy.npy')
print ('size-data='+str (new_data.shape))

from tensorflow.keras.models import load_model

# Load the finetuned model
autoencoder = load_model('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/Models/auto
encoder-model_finetuned')

from joblib import load
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt

# Predict encoded features for new_data
```

```
encoded_features = encoder.predict(new_data)

# Apply t-SNE
tsne_output = TSNE(random_state=123).fit_transform(encoded_features)

# Predict cluster assignments for new_data using the full model
cluster_assignments = model.predict(new_data)
y = np.argmax(cluster_assignments, axis=1)  # convert soft assignments to hard

# Define the plotter function
def plotter(S, y, target_names):
    '''
    function to visualize the outputs of t-SNE
    '''
    # choose a color palette with seaborn.
    colors = ['red', 'darkorange', 'green']

    lw = 2
    # create a scatter plot.
    f = plt.figure(figsize=(22, 10))
    ax = f.add_subplot(111)
    for color, i, target_name in zip(colors, range(len(target_names)), target_names):
        plt.scatter(S[y == i, 0], S[y == i, 1], color=color, alpha=.8, lw=lw, s=100, label=
target_name)
    plt.legend(loc='lower left', shadow=False, scatterpoints=1, prop={'size': 26})
    ax.axis('off')
    ax.axis('tight')
    plt.show()
    f.savefig ('Tsne-km-n3-ft.png', dpi=100, bbox_inches="tight")

    return f, ax

# Target names for clusters
target_names = ['EA', 'CT2', 'CT1']  # Adjust as per your actual cluster names

# Visualize
plotter(tsne_output, y, target_names)

### Save the labels
np.savetxt('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Valid-n3-ft.txt', y,
fmt='%i', delimiter=',')

# Path to the file
file_path = '/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Valid-n3-ft.txt'

# Initialize a dictionary to count cluster occurrences
cluster_counts = {}

# Read the file line by line
with open(file_path, 'r') as file:
    for line in file:
        cluster = line.strip()
        if cluster in cluster_counts:
            cluster_counts[cluster] += 1
        else:
            cluster_counts[cluster] = 1

# Print the counts for each cluster
for cluster, count in cluster_counts.items():
    print(f"Cluster {cluster}: {count} occurrences")

Cluster 1: 211 occurrences

# Change the order of the cluster numbers (just for a nice representation)
```

```
with open('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Valid-n3-ft.txt', 'r'
) as file :
  filedata = file.read()

# Replace the target string
filedata = filedata.replace('2', 'x')
filedata = filedata.replace('0', '2')
filedata = filedata.replace('1', '0')
filedata = filedata.replace('x', '1')

# Re-write the output
with open('/run/media/viblab/Markov2/Haykal/AnakKrakatauEWSFinal/data/Valid-n3-ft_Rewritten
.txt', 'w') as file:
  file.write(filedata)
```

E. Gradient Boosted decision tree

Data Loading

```
# Filter the dataset for rows where time_to_eruption is between 10 and 1440 minutes
filtered_data = data_new[(data_new['time_to_eruption'] < 1080)]

start_datetime = pd.Timestamp('2018-06-24', tz='UTC')
end_datetime = pd.Timestamp('2018-12-22 01:23:00+00:00', tz='UTC')

# Filter the dataset based on the datetime range
filtered_by_datetime = filtered_data[(filtered_data['timestamp'] >= start_datetime) & (filt
ered_data['timestamp'] <= end_datetime)]

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

# Load the dataset
data = filtered_by_datetime

# If 'cluster' is not of type 'category', convert it
if data['cluster'].dtype != 'category':
    data['cluster'] = data['cluster'].astype('category')

# Selecting features
selected_features = ['enc_2','enc_3','enc_6','enc_8','enc_13','enc_14','enc_21','enc_23', '
cluster']  # Add/remove features as needed

X = data[selected_features]

# Target variable
y = data['time_to_eruption']

# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

LGBM Regressor

Data Loading

```
import lightgbm as lgb
from sklearn.metrics import mean_absolute_error, r2_score, mean_squared_error

# Creating the LightGBM dataset
train_data = lgb.Dataset(X_train, label=y_train, categorical_feature=['cluster'], free_raw_
data=False)
```

```
test_data = lgb.Dataset(X_test, label=y_test, categorical_feature=['cluster'], reference=tr
ain_data, free_raw_data=False)
```

Optuna Hyperparameter Tuning

```
import optuna
import lightgbm as lgb
from sklearn.metrics import mean_absolute_error

def objective(trial):
    # Define the hyperparameters to be tuned
    param = {
        'objective': 'regression',
        'metric': 'mae',
        'verbosity': -1,
        'boosting_type': 'gbdt',
        'num_leaves': trial.suggest_int('num_leaves', 20, 40),
        'learning_rate': trial.suggest_float('learning_rate', 0.01, 0.1),
        'n_estimators': trial.suggest_int('n_estimators', 20, 100),
        'max_depth': trial.suggest_int('max_depth', 3, 9),
        'min_child_samples': trial.suggest_int('min_child_samples', 5, 30),
        'subsample': trial.suggest_float('subsample', 0.7, 1.0),
        'colsample_bytree': trial.suggest_float('colsample_bytree', 0.7, 1.0),
        'reg_alpha': trial.suggest_float('reg_alpha', 0.0, 0.5),
        'reg_lambda': trial.suggest_float('reg_lambda', 0.0, 0.5)
    }

    # Create a callback list with early stopping
    callbacks = [
        lgb.early_stopping(stopping_rounds=10)
    ]

    # Train a model with the current set of hyperparameters
    evals_result = {}
    model = lgb.train(param, train_data, valid_sets=[test_data],
                      callbacks=callbacks, evals_result=evals_result, verbose_eval=False)

    # Predict on validation data
    y_pred = model.predict(X_test, num_iteration=model.best_iteration)

    # Compute the Mean Absolute Error
    mae = mean_absolute_error(y_test, y_pred)

    return mae

# Create a study object and optimize the objective function
study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=100)

# Best hyperparameters
best_params = study.best_params
print("Best parameters:", best_params)
```

Training

```
# Train the final model using the best parameters

best_params['metric'] = 'mae'

evals_result = {}  # Dictionary to store evaluation results

# Create a callback list with early stopping
callbacks = [
    lgb.early_stopping(stopping_rounds=10),
    lgb.log_evaluation(period=1)
```

```
]

best_model = lgb.train(best_params, train_data, valid_sets=[train_data, test_data],
                       valid_names=['training', 'validation'],
                       early_stopping_rounds=10, evals_result=evals_result)

# Predict with the best model
y_pred_best = best_model.predict(X_test, num_iteration=best_model.best_iteration)

# Calculate MAE
mae = mean_absolute_error(y_test, y_pred_best)
print(f"Best MAE: {mae}")

# Calculate RMSE
rmse = mean_squared_error(y_test, y_pred_best, squared=False)
print(f"Best RMSE: {rmse}")

# Calculate R^2
r2 = r2_score(y_test, y_pred_best)
print(f"Best R^2: {r2}")

import matplotlib.pyplot as plt
import lightgbm as lgb

# Assuming best_model is your trained LightGBM model
lgb.plot_importance(best_model, importance_type='split')
plt.title("LightGBM Feature Importance - Split")
plt.show()

lgb.plot_importance(best_model, importance_type='gain')
plt.title("LightGBM Feature Importance - Gain")
plt.show()

lgb.plot_metric(evals_result, metric='l1')
plt.title('MAE for Training and Validation Sets')
plt.xlabel('Iterations')
plt.ylabel('MAE')
plt.legend(['Training', 'Validation'])
plt.show()



import matplotlib.pyplot as plt

# Assuming y_test are the actual values and y_pred_best are the predicted values from the m
odel
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred_best, alpha=0.5)
plt.title('Actual vs. Predicted')
plt.xlabel('Actual values')
plt.ylabel('Predicted values')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)  # Diagon
al line
plt.show()
```

ABOUT THE AUTHOR

Muhamad Haykal Hanif Gifari Adi is passionate student in Engineering Physics with a robust background in engineering, technology, and organizational leadership. Haykal have honed his skills in complex problem-solving, technical analysis, and project management through hands-on experience in various projects, academic research, and leadership roles in student organizations with a huge interest in solving real-world problems using computational and numerical methods. Understanding his background as an activist and a changemaker, Haykal sought to change the world, step by step at a time.