

Tunisian Republic



Ministry of Higher Education and Scientific
Research
University of Carthage



National Engineering School of Carthage

GRADUATION PROJECT DISSERTATION

Presented with a view to obtaining the

National Engineering Diploma in Infotronics Systems

Classification of MILCOs and Non-threat objects with LF-SAS

by

Mrs. MAYSSA BOUHACHEM

Conducted at

Royal Military Academy of Belguim



Head of Jury: : Mr.Mohamed Nabil Maghraoui

Reviewer : Mr.Faouzi Jaidi

Industrial Supervisor : Mr.Camilo Andres Hurtado Erasso

Academic Supervisor: Mr.Yassine Hachaichi

Academic Year: 2025 - 2026

Acknowledgments

I would first and foremost like to express my sincere gratitude to my professional supervisor, Mr. Camilo Hurtado, for giving me the opportunity to carry out this internship at the Royal Military Academy and for his guidance, availability, and valuable advice throughout this experience. I would also like to thank the Maritime Applications team and the CISS Department for their warm welcome, support, and collaboration, as well as my fellow intern and friend for the mutual support and shared experiences throughout the internship. My sincere thanks also go to my academic supervisor, Mr. Yassine Hchaichi, for his guidance and follow-up, and to all the professors at the National Engineering School of Carthage for the knowledge and skills they have provided throughout my studies. I am grateful to the president of the jury and the reviewer for their time and valuable feedback. Finally, I would like to express my deepest and most heartfelt thanks to my family, whose constant support, patience, encouragement, and belief in me have accompanied me throughout every step of this journey. Their presence and support have been a source of strength throughout my studies, and I am profoundly grateful to have had them by my side.

Mrs. MAYSSA BOUHACHEM

Contents

General Introduction	1
1 General context	3
1.1 Presentation of the Host Institution	4
1.1.1 Presentation of the Royal Military Academy (RMA)	4
1.1.2 The Communication, Information, Systems and Sensors (CISS) Department	5
1.2 Project specifications	6
1.2.1 Project Presentation	6
1.2.2 Methodology and Tasks	7
1.2.3 Problem Statement	7
1.2.4 Functional Requirements	8
1.2.5 Non-Functional Requirements	8
1.2.6 Project Implementation Steps	9
1.3 General Concepts	10
1.3.1 Mines	10
1.3.2 Mine countermeasure	11
1.3.3 Acoustic systems	13
1.3.4 Synthetic aperture sonar	16
1.3.5 Low-Frequency vs High-Frequency Sonar	19
1.3.6 The LFM Chirp Signal	21
1.4 State of the Art	21
1.4.1 Image-Based Approaches	21
1.4.2 Classification from Acoustic Time Series	22
1.4.3 Positioning of This Work	24
2 Data Processing and Classification Methodology	25
2.1 Dataset Description	26
2.1.1 Experiment Configuration	27
2.1.2 Data Structure	28
2.2 Signal Processing Pipeline	29

2.2.1	The ω -k Focusing Algorithm	31
2.2.2	Notation and Wavenumber Variables	31
2.2.3	Analytic signal formation:	33
2.2.4	Transformation to the wavenumber domain	34
2.2.5	Range compression to scene center	35
2.2.6	Stolt interpolation	36
2.2.7	Image formation	37
2.3	Data Preparation for the Classifiers	38
2.3.1	Active-Zone Detection and Stride-Based Subsampling	38
2.3.2	Dataset Export and Metadata	39
2.3.3	Feature Extraction with tsFresh	41
2.3.4	Feature Cleaning and Selection	43
2.3.5	Group-Based Nested Cross-Validation	45
2.4	Machine Learning Classifiers and Evaluation Protocol	48
2.4.1	Logistic Regression	49
2.4.2	Support Vector Machine (RBF Kernel)	50
2.4.3	Random Forest	52
2.4.4	Gradient Boosting: XGBoost and LightGBM	55
2.4.5	H2O AutoML: Automatic machine learning	58
2.4.6	Hyperparameter Optimization with Optuna	59
2.4.7	Evaluation Metrics and Confidence Intervals	60
3	Results and Interpretation	64
3.1	Model Results	65
3.1.1	Logistic Regression	65
3.1.2	Support Vector Machine (RBF Kernel)	68
3.1.3	Random Forest:	71
3.1.4	XGBoost	74
3.1.5	Light GBM	77
3.1.6	Comparison Across Classifiers	80
3.1.7	H2O AutoML	81
3.2	Target-Level Difficulty Analysis	82

3.2.1	Per-Target OOF Performance	82
3.2.2	Fold Composition and Cross-Classifer Consistency	83
3.2.3	Relating Difficulty to Target Metadata	84
3.3	Model Interpretability	85
3.3.1	Global Interpretability	85
3.3.2	Local Interpretability	88
	General Conclusion	93
	Annexes	99

List of Figures

1.1	Logo of the Royal Military Academy (RMA) [4].	4
1.2	Geographical distribution of the CISS Department's	6
1.3	Classification of sea mines [8].	11
1.4	Signature reduction through active compensation	12
1.5	Principle of sonar operation[14].	14
1.6	Comparison of the main sonar mapping geometries	16
1.7	Geometry of data collection from a SAS AUV	17
1.8	Side-scan sonar versus synthetic aperture sonar	18
1.9	HF-SAS Image versus LF-SAS Image	20
1.10	A plot showing a linear chirp signal [22].	21
2.1	Setup of TREX13 trial area off Panama City, Florida.[32]	27
2.2	Selection of targets deployed during TREX13	28
2.3	Successive pings recorded as the sonar moves past a target	29
2.4	Raw SAS pulse-compressed image vs focused SAS image	30
2.5	Sequence of operations in the ω -k focusing algorithm.	31
2.6	Effect of the Hilbert transform on a single range line	33
2.7	Effect of fftshift on the two-dimensional wavenumber spectrum	34
2.8	Comparison of a single ping's raw pulse-compressed signal and its SAS-focused signle	37
2.9	Energy is computed for each row	39
2.10	Hierarchical structure of the exported HDF5 dataset.	40
2.11	One outer-fold iteration	46
2.12	Train/test role of one target group across the 5 outer folds	47
2.13	OOF decision	48
2.14	SVM margin and slack variables	51
2.15	Random Forest ensemble structure.	54
2.16	Representation of a Gradient Boosting classification process [41].	56
3.1	Logistic Regression ROC curve (out-of-fold predictions).	67

3.2	Logistic Regression confusion matrix (out-of-fold predictions).	68
3.3	SVM (RBF) ROC curve (out-of-fold predictions).	70
3.4	SVM (RBF) confusion matrix (out-of-fold predictions).	71
3.5	Random Forest ROC curve (out-of-fold predictions).	73
3.6	Random Forest confusion matrix obtained from pooled out-of-fold predictions.	74
3.7	XGBoost ROC curve (out-of-fold predictions).	76
3.8	XGBoost confusion matrix obtained from pooled out-of-fold predictions.	77
3.9	LightGBM ROC curve (out-of-fold predictions).	79
3.10	LightGBM confusion matrix obtained from pooled out-of-fold predictions.	79
3.11	Out-of-fold performance of all five classifiers	80
3.12	Impurity-based versus permutation importance	88
3.13	SHAP beeswarm plot, five false-positive hard targets	89

List of Tables

1.1	Evolution of sonar technologies used for mine detection [15], [16]	15
1.2	Comparison between low- and high-frequency SAS systems	20
1.3	Image-based methods for SAS mine classification.	22
1.5	Feature- and model-based methods for time-series (signal-level) target classification.	23
2.1	Main categories of tsFresh features present in the extracted feature set.	43
2.3	Feature cleaning and selection steps	44
3.1	Optuna search space for the Logistic Regression classifier	66
3.2	Logistic Regression per-fold performance	66
3.3	Optuna search space for the SVM (RBF) classifier	69
3.4	SVM (RBF) per-fold performance	69
3.5	Optuna search space for the Random Forest classifier	71
3.6	Random Forest per-fold performance	72
3.7	Optuna search space for the XGBoost classifier	75
3.8	XGBoost per-fold performance	75
3.9	Optuna search space for the LightGBM classifier	78
3.10	LightGBM per-fold performance	78
3.11	Per-fold performance of H2O AutoML.	81
3.12	The six targets with the lowest OOF classification accuracy.	82
3.13	Target composition of the five outer folds.	83
3.14	OOF accuracy for the six hardest targets across classifiers.	83
3.15	Metadata for the six hardest-to-classify targets.	84
3.16	Top 20 features by impurity-based importance	86
3.17	Top 20 features by permutation importance (roc_auc)	87
3.18	Class-conditional typical ranges and hard-target raw values	90
4.1	APL-UW tower-rail SAS system parameters used during TREX13 [31].	99
4.2	Parameters used for ω -k SAS image formation in this work.	99
4.3	TREX-13 target inventory	100

4.4	Basic statistical features.	103
4.5	Nonlinear dynamics and complexity features.	105
4.6	Variability and threshold-based features.	105
4.7	Autocorrelation-based features.	106
4.8	Peak-related features.	106
4.9	Continuous wavelet transform (CWT) coefficient features.	107
4.10	Autoregressive model features.	107
4.11	Change quantile features.	108
4.12	Frequency-domain (FFT) features.	108
4.13	Trend and regression-based features.	109
4.14	Stationarity test features.	110
4.15	Energy distribution features.	110
4.16	Entropy-based features.	111
4.17	Symmetry-based features.	111

List of Abbreviations

AP: Average Precision

AUC: Area Under the Curve

AUV: Autonomous Underwater Vehicle

CI: Confidence Interval

CISS: Communication, Information, Systems and Sensors

DFT: Discrete Fourier Transform

DIDSON: Dual-Frequency Identification Sonar

HDF5: Hierarchical Data Format version 5

HF: High Frequency

LF: Low Frequency

LFM: Linear Frequency Modulated

MBES: Multibeam Echo Sounder

MCM: Mine Countermeasures

MILCO: Mine-Like Contact

OOF: Out-Of-Fold

PCB: Pulse-Compressed Baseband

RF: Random Forest

ROC: Receiver Operating Characteristic

ROV: Remotely Operated Vehicle

SAR: Synthetic Aperture Radar

SAS: Synthetic Aperture Sonar

SHAP: SHapley Additive exPlanations

SSS: Sidescan Sonar

SVM: Support Vector Machine

UXO: Unexploded Ordnance

General Introduction

The world’s oceans, seas, rivers, and coastal waters lie a quiet yet deadly legacy from decades of armed conflict: unexploded ordnance (UXO) and sea mines resting on or buried beneath the seafloor. According to the Geneva International Centre for Humanitarian Demining (GICHD), sea mines were responsible for about 40% of vessel losses in past maritime conflicts, with many of these munitions never recovered after the fighting ceased [1]. Today, the legacy of this pollution remains a threat to maritime navigation, offshore infrastructure development, fisheries and marine ecosystems. 1.6 million metric tons of conventional munitions sits on the seabed at German coastal waters alone [2].

Acoustic sensing technologies, led by Synthetic Aperture Sonar (SAS), represent a critical advancement in underwater threat detection. Unlike turbidity-limited optical sensors or high-frequency sonars that lack the penetration depth for sub-bottom imaging, Low-Frequency SAS (LF-SAS) facilitates simultaneous long-range detection and sediment penetration, including the detection of partially or fully buried mines. This technology is particularly adept at MILCO (Mine-Like Object) classification because metallic structures exhibit resonant scattering at low frequencies—a physical signature absent in natural seafloor clutter [3].

Despite these physical advantages, automatically distinguishing Mine-Like Objects (MILCO) from Non-Threat Objects in SAS data remains a challenging classification problem. The difficulty arises from the high variability in target appearance depending on orientation, burial depth, and sediment type, combined with the fact that some man-made clutter objects share similar shapes or materials with actual threats, making their acoustic signatures hard to separate reliably. This difficulty is compounded, in a defence context, by the fact that classification errors carry highly asymmetric operational costs, and that the adoption of machine learning models in such critical applications is often limited by the difficulty of interpreting their decisions.

This project, carried out within the Research Department of the Royal Military Academy of Belgium, tackles this challenge using the real-world LF-SAS dataset TREX13, collected by researchers from the Applied Physics Laboratory of the University of Washington and the Naval Surface Warfare Center [3]. To address this problem, the work is structured

around three main objectives: first, developing a reproducible signal processing and feature extraction pipeline for LF-SAS data; second, training and comparing several classical machine learning classifiers, namely Logistic Regression, Support Vector Machine (SVM), Random Forest, XGBoost, and LightGBM; and third, applying interpretability methods to understand the classification decisions and relate them to the physical characteristics of the underlying targets. These objectives aim to assess the ability of the models to distinguish MILCO from Non-Threat Objects while considering the challenges associated with different target types, orientations, and burial conditions.

The report is structured as follows. The first chapter presents the operational context and theoretical background of the project, including underwater acoustics, SAS imaging principles, and existing MILCO classification approaches. The second chapter describes the TREX13 dataset, signal processing and feature extraction procedures, and the machine learning methodology. The third chapter presents and discusses the classification results and the interpretability analysis. The report concludes with a general conclusion summarising the main findings, limitations, and directions for future work.

GENERAL CONTEXT

Plan

1.1	Presentation of the Host Institution	4
1.2	Project specifications	6
1.3	General Concepts	10
1.4	State of the Art	21

Introduction

This chapter presents the operational and technical context of the project. After introducing the host institution and the project specifications, the fundamental concepts of naval mines, mine countermeasures, and sonar-based detection are covered, with particular focus on synthetic aperture sonar and its operating principles. The chapter concludes with a state of the art review of existing MILCO classification approaches, positioning the present work within the existing literature.

1.1 Presentation of the Host Institution

This section presents the host institution where the internship was carried out. It first introduces the Royal Military Academy (RMA), followed by an overview of the Communication, Information, Systems and Sensors (CISS) Department, in which the project was conducted.

1.1.1 Presentation of the Royal Military Academy (RMA)

The Royal Military Academy (RMA) is Belgium's military university, responsible for the academic, military, physical, and leadership training of future officers, as well as for the continuing education of officers throughout their careers within the Belgian Defence.



Figure 1.1: Logo of the Royal Military Academy (RMA) [4].

Founded in 1834 and located in Brussels, the RMA is the only military university in Belgium and trains officers for all components of the Belgian Defence, including the Land Component, Air Component, Naval Component, Medical Component, and Cyber Command. Courses are offered in French, Dutch, and English.

The institution comprises two faculties: the Faculty of Applied Sciences (Polytechnic), which awards a Master of Science in Engineering Sciences, and the Faculty of Social and Military Sciences. In addition to undergraduate and graduate education, the RMA provides a wide range

of professional development programmes for officers, including Staff Officer Training, Senior Officer Candidate Training, and the Higher Staff Course through its Defence College.

Beyond its educational mission, the RMA is also actively involved in scientific research. Its research activities focus primarily on addressing military challenges and, more broadly, on issues related to security and safety, while contributing to numerous national and international research programmes [4].

1.1.2 The Communication, Information, Systems and Sensors (CISS) Department

The Communication, Information, Systems and Sensors (CISS) Department is a research unit within the Faculty of Applied Sciences (Polytechnic) at the Royal Military Academy, where this internship was conducted. CISS is an active research department comprising 64 researchers and managing more than 140 projects in fields such as radar systems, sonar, antenna design, signal processing, and cognitive radio.

The department conducts research on electromagnetic and acoustic sensing systems, including radar, sonar, antennas, wireless communications, and electronic warfare technologies. A major focus of its activities is the development of advanced signal processing and artificial intelligence techniques for the extraction, analysis, and interpretation of information acquired by sensors. These technologies support applications such as target detection and classification, underwater surveillance, autonomous systems, spectrum monitoring, and maritime situational awareness.

In addition to its research in sensing and communication systems, CISS contributes to the design and evaluation of innovative defence and security technologies through its participation in numerous national, European, and NATO-funded projects. These collaborations involve academic institutions, research centres, governmental organisations, and industrial partners. The geographical distribution of these collaborations, spanning Europe, North America, and other regions of the world, is illustrated in Figure 1.2.



Figure 1.2: Geographical distribution of the CISS Department's external research collaborations over the past five years, as shown on the RMA Research Portal [5].

Within this research environment, the present final-year project focuses on the classification of mine-like objects using low-frequency synthetic aperture sonar signals, under the supervision of Mr. Camilo Andres Hurtado Erasso.

1.2 Project specifications

1.2.1 Project Presentation

This final-year project is carried out within an ongoing research effort at the CISS department of the Royal Military Academy, focused on the detection and classification of underwater mines and Unexploded Ordnance (UXO) using low-frequency sonar. Submarine mines remain a critical and persistent threat to navies worldwide, making reliable automatic classification of Mine-Like Contact (MILCO) a key operational priority for naval mine countermeasures (MCM).

The internship was originally defined with the objective of moving beyond "black-box" machine learning models and AI by implementing an interpretability analysis on a Random Forest classifier applied to low-frequency Synthetic Aperture Sonar (SAS) data in order to understand how the model distinguishes threats from the seabed environment and to ensure the reliability of its classification in defence contexts.

As the project progressed, its scope naturally extended beyond the analysis of a single classifier. A complete machine learning pipeline was developed and validated, comparing several families of classifiers, including logistic regression, support vector machines (SVMs), Random Forest, and gradient boosting methods, using features extracted with `tsfresh` from the TREX13 datasets. The analysis was subsequently narrowed to model interpretability and the operational implications of classification thresholds.

1.2.2 Methodology and Tasks

The work was structured around the following tasks, as defined in the internship agreement:

- **Data Processing:** analysis and processing of low-frequency sonar data, including beamforming and synthetic aperture sonar (SAS) processing, provided by the RMA.
- **Model Implementation:** application of machine learning classifiers, with the Random Forest classifier as the primary reference model, to the processed sonar datasets.
- **Interpretability Analysis:** implementation of explainable artificial intelligence (XAI) methods to decompose and understand the classifier’s decision-making process.
- **Validation:** evaluation of the model’s transparency and its alignment with maritime defence requirements.

1.2.3 Problem Statement

In mine countermeasure operations, classification errors carry highly asymmetric costs, as a missed mine (false negative) can result in loss of life or vessel damage, whereas a false alarm (false positive) generally leads only to an additional, time-consuming verification procedure. This asymmetry highlights the need for reliable classification methods that can support decision-making in operational environments.

Moreover, the adoption of machine learning models in applications that are critical to defence is often limited by the difficulty of interpreting their predictions processes. Although these models may achieve strong classification performance, they often provide limited insight into the reasons behind their predictions. In military contexts where classification decisions can have serious consequences, this lack of transparency raises significant concerns about trust, accountability, and operational validation.

This leads to the central question addressed in this project: How can a machine learning classifier for underwater mine detection be designed, evaluated, and interpreted in a way that is both operationally effective and sufficiently transparent to be trusted in a defence context?

1.2.4 Functional Requirements

The functional requirements defining the core capabilities expected from the system are as follows:

- **Process raw sonar signals:** Extract and prepare 1D acoustic signals from SAS imagery (TREX13 datasets) for feature-based classification.
- **Extract relevant features:** Compute a comprehensive set of statistical and signal-based features (via tsFresh) capable of capturing the relevant characteristics of mine-like versus non-mine-like objects.
- **Train and compare classifiers:** Implement and evaluate multiple classifier families to identify the most suitable model(s) for MILCO classification.
- **Apply interpretability methods:** Implement SHAP-based analysis on the selected classifier to explain individual predictions and overall feature importance.
- **Support operational decision-making:** Analyse classification thresholds while considering the different consequences of false negative and false positive predictions in the context of mine countermeasures.

1.2.5 Non-Functional Requirements

In addition to the core functionalities, the system must satisfy the following quality constraints:

- **Reliability:** Ensure that classification results are robust and reproducible, notably through cross-validation strategies that prevent data leakage between physical targets.
- **Transparency:** Guarantee that model decisions can be explained and justified, in line with the interpretability requirements of defense applications.
- **Computational efficiency:** Ensure that feature extraction and model training remain tractable given the available computational resources (including GPU-based training where applicable).
- **Reproducibility:** Document the methodology and parameters thoroughly enough to allow the pipeline to be reproduced or extended in future work.

1.2.6 Project Implementation Steps

To meet the requirements defined above, the work was carried out through the following successive steps:

1. **Theoretical research:** Study of the principles of SAS imaging, the physics of acoustic scattering from underwater objects, and the fundamentals of machine learning classification and interpretability methods.
2. **Data preparation:** Processing of the TREX13 datasets, including signal cropping, feature extraction using tsfresh, and feature selection through variance and correlation based pruning.
3. **Progressive model development:** Iterative implementation and tuning of multiple classifiers, ranging from baseline models such as Logistic Regression and SVM to more advanced ensemble and boosting methods, including Random Forest, LightGBM, and XGBoost approaches.
4. **Interpretability analysis:** Application of SHAP to the selected reference classifier to identify the most influential features and assess the consistency of the model's decision making process.
5. **Final integration and documentation:** Consolidation of the results into a coherent pipeline and preparation of three required deliverables. The first deliverable is a comprehensive technical report covering data processing, classifier performance, and the results of the interpretability analysis. The second is a scientific contribution consisting of a structured draft and an organised set of results and visualisations, intended to provide the main foundation for a scientific paper. The third is a collaborative paper development effort, in which the technical data, figures, and analysis are provided as a basis for a conference paper to be finalised and co-authored with the RMA supervisor.

1.3 General Concepts

1.3.1 Mines

Sea mines have been used in naval warfare since the nineteenth century, with some of the earliest recorded applications dating back to the Crimean War (1853-1856). Since then, they have evolved from relatively simple explosive devices into more sophisticated systems designed to threaten both surface ships and submarines. In general, a sea mine consists of three main elements, which are an explosive charge, a sensing system, and a triggering mechanism [6]. Naval mines can vary considerably in both size and destructive capacity. Their dimensions may range from about 50 cm to nearly 3 m, while their explosive payload can range from roughly 50 kg to more than 1,000 kg, depending on their design and intended operational use. They may be employed for defensive purposes, such as protecting strategic or territorial waters, or for offensive purposes, such as restricting enemy movement, channelling vessels into predefined routes, or disrupting maritime trade and military operations [7].

1.3.1.1 Classification by Actuation Mechanism

Naval mines may first be categorised according to the mechanism that triggers their detonation. Two principal groups can be distinguished, namely contact mines and influence mines. Contact mines are activated when a vessel comes into physical contact with the mine and may incorporate mechanisms designed to make mine countermeasure operations more difficult. Influence mines, in contrast, are triggered without direct physical contact and respond to characteristic signatures produced by nearby vessels. They can rely on magnetic, acoustic, or pressure effects and may combine several sensing principles to improve their selectivity and resistance to countermeasures [7]. Some modern systems may also incorporate intelligent processing techniques to distinguish potential targets from false alarms [6].

1.3.1.2 Classification by Position

Naval mines can also be classified according to their position in the water column. As illustrated in Figure 1.3, three main categories are commonly identified, consisting of drifting mines, bottom mines, and moored mines.

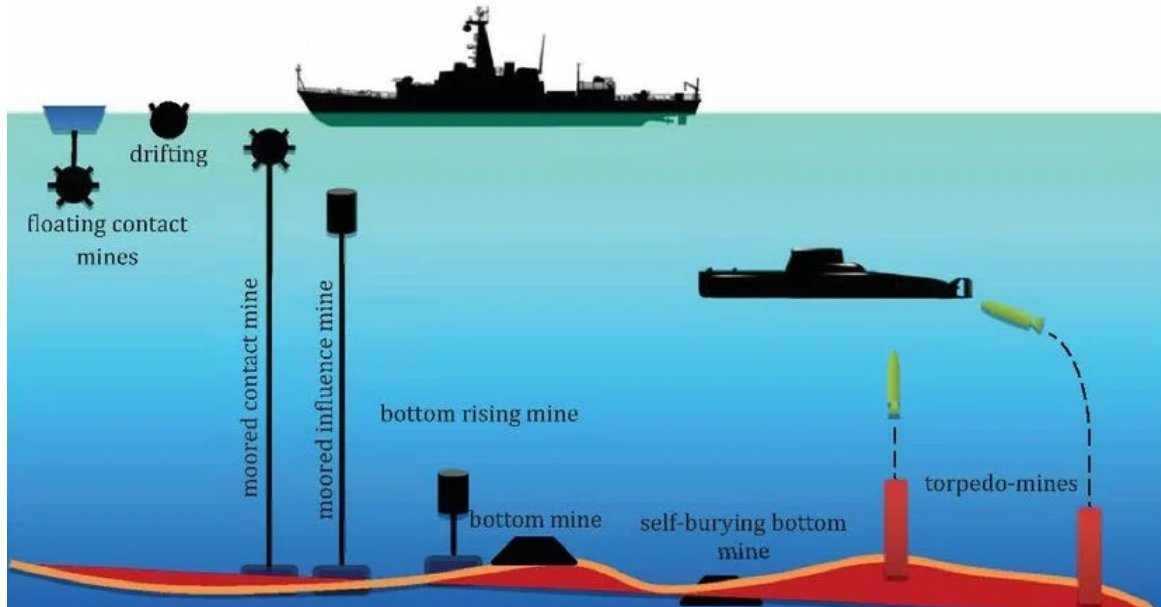


Figure 1.3: Classification of sea mines [8].

Drifting mines are free floating devices that move with wind and water currents and may be activated through contact or remote mechanisms. Bottom mines are placed directly on the seabed, where they may remain exposed or become partially or fully buried in sediment, which can affect their detection by sonar. Moored mines are anchored to the seabed while remaining suspended at a specified depth within the water column. Unlike bottom mines, they do not rest directly on the seabed and may use contact, influence, or remote activation mechanisms [7].

1.3.2 Mine countermeasure

Mine countermeasures (MCM) refer to the set of strategies, technologies, and operations developed to reduce the threat posed by naval mines. Because mines can remain hidden for long periods and endanger both military and civilian vessels, MCM plays a critical role in ensuring safe navigation and maintaining access to maritime areas [6]. These defensive measures are divided into passive measures, which aim to reduce a vessel's vulnerability to mine activation, and active measures, which seek to detect and neutralise mines before they can cause harm [7].

1.3.2.1 Passive Countermeasures

Passive countermeasures aim to reduce the risk of triggering influence mines by limiting the detectable signatures produced by a vessel while transiting through a potentially mined area. Since influence mines can rely on magnetic, acoustic, or pressure signatures to detect nearby

vessels, reducing these signatures decreases the likelihood of mine activation [9].

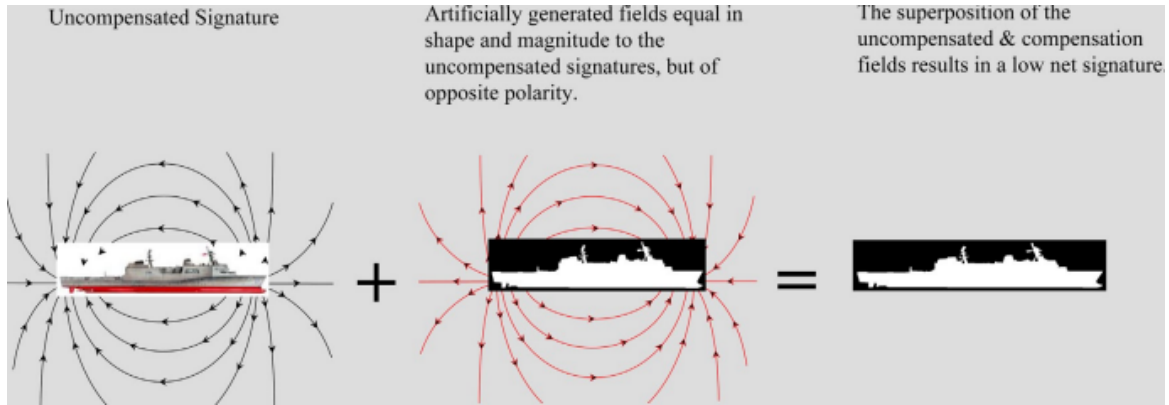


Figure 1.4: Signature reduction through active compensation. (a) Uncompensated signature; (b) artificially generated fields with the same shape and magnitude but opposite polarity; and (c) superposition of the two fields, resulting in a reduced net signature [10].

Among these signatures, the magnetic signature is particularly important because it is strongly influenced by the ferromagnetic properties of a vessel's hull. Naval vessels can reduce this signature using degaussing systems, which generate a compensating magnetic field through onboard coils, as illustrated in Figure 1.4. In addition, some mine countermeasure vessels are constructed from non-magnetic materials, such as fibreglass or wood, to further reduce their magnetic signature [7], [9]. Acoustic signatures can be reduced through quieter propulsion systems and noise reduction techniques, while pressure signatures are mainly mitigated through operational measures such as controlling vessel speed and selecting suitable water depths [6].

Although passive countermeasures can reduce the probability of mine activation, they cannot completely eliminate the threat. They are therefore generally complemented by active countermeasures, which aim to detect and neutralise mines before a vessel enters the affected area.

1.3.2.2 Active Countermeasures

Active countermeasures aim to directly address the mine threat by detecting and neutralising mines within a suspected area. Unlike passive countermeasures, which focus on reducing the probability of mine activation, active measures seek to locate and remove the threat.

One traditional approach is minesweeping, which aims to clear a predefined area without necessarily locating each mine individually. Mechanical sweeping uses towed equipment to remove or expose mines, while influence sweeping generates magnetic, acoustic, or pressure signatures designed to trigger influence mines at a safe distance [7]. However, modern mines

may incorporate mechanisms that reduce the effectiveness of sweeping techniques [6].

An alternative approach is mine hunting, which consists of systematically searching for individual mines on the seabed, followed by their detection, classification, identification, and neutralisation. Various sensing technologies can be used during this process, including electromagnetic, magnetic, chemical, and acoustic systems. Among these, sonar systems are particularly important because they can provide detailed information about the location and characteristics of potential targets. Mine hunting systems are commonly deployed from dedicated mine countermeasure vessels or unmanned platforms such as autonomous underwater vehicles (AUVs) and remotely operated vehicles (ROVs). Compared with minesweeping, mine hunting provides a more precise assessment of the suspected area but is more dependent on sensor performance and environmental conditions [7], [11].

1.3.3 Acoustic systems

Acoustic systems, commonly referred to as **SONAR** (Sound Navigation and Ranging), constitute the third category of underwater munition detection technologies and are the primary focus of this work.

◆ Historical background:

The origins of sonar trace back to the early twentieth century, when concerns over navigational safety and submarine warfare drove the development of the first acoustic detection systems, notably Fessenden’s oscillator, which by 1914 could echo-locate an iceberg several kilometres away. The development of sonar accelerated during the First and Second World Wars, as the need to detect and locate submarines became increasingly important. Post-war advances in transducer design, propagation modelling, and computing power gradually extended the capabilities of sonar beyond the detection of underwater objects to include their classification and identification, with applications in both military and civilian fields [12].

◆ Functioning principle:

A sonar system transmits acoustic pulses into the water using a transducer . When these waves hit a target or the seabed, part of the energy reflects back as an echo, which is picked up by hydrophones (underwater acoustic receivers) as illustrated in the figure 1.5. By analyzing the echo’s time delay, amplitude, and frequency, the system estimates the target’s range, bearing, and physical characteristics. In active sonar, the system both emits and receives

signals; in passive sonar, it only listens for sounds produced by external sources[13].

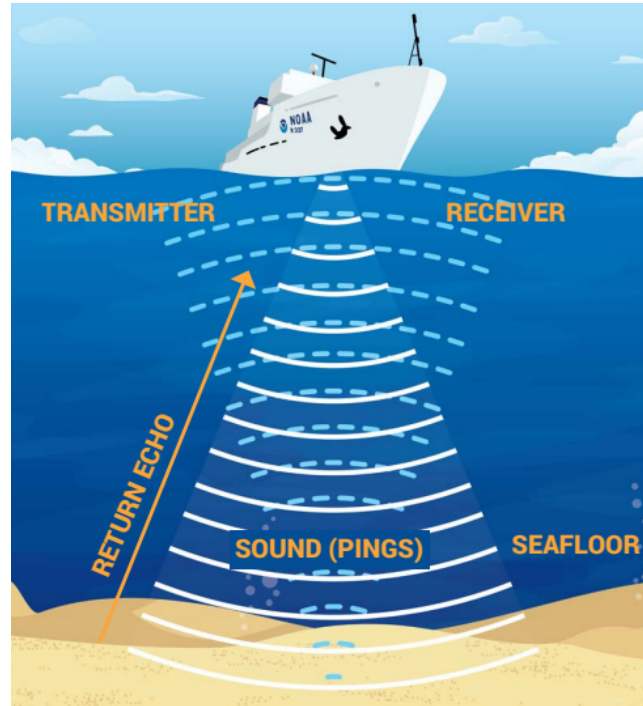


Figure 1.5: Principle of sonar operation[14].

◆ Evolution and types of sonar systems:

The continuous evolution of sonar technology has resulted in a variety of specialized systems designed to address different mine-hunting requirements. These include side-scan sonar, multibeam echo sounders, dual-frequency identification sonar, and synthetic aperture sonar. Each technology offers distinct advantages in terms of coverage area, imaging resolution, target burial detection, and classification capability. Table 1.1 summarizes the historical evolution of these sonar modalities and their respective contributions to mine detection.

Table 1.1: Evolution of sonar technologies used for mine detection [15], [16]

Period	Sonar Type	Contribution to Mine Detection
1940s	Passive sonar	Enabled the detection of underwater acoustic sources but provided no imaging capability.
1950s	Sectorscan sonar	Introduced forward-looking acoustic mapping using directive beams, enabling the detection and approximate localization of objects ahead of the vessel, though with limited resolution and seabed coverage.
1960s	Sidescan sonar (SSS)	Enabled high-resolution, wide-swath seabed imaging, allowing effective detection, classification, and mapping of mines over large areas.
1980s	Multibeam echo sounder (MBES)	Provided dense bathymetric mapping over wide swaths using multiple narrow beams and improving target localization on the seafloor.
1990s	Dual-Frequency Identification Sonar (DIDSON)	Introduced near-video-quality acoustic imagery for close-range target inspection and identification.
2000s	Synthetic Aperture Sonar (SAS)	Combined wide-area coverage with very high image resolution, significantly enhancing mine detection and classification.

Following the historical evolution presented in Table 1.1, sonar mapping systems are commonly classified into three main categories: single-beam echo sounders, multibeam echo sounders, and side-scan sonars, as illustrated in Figure 1.6. Single-beam echo sounders transmit a single acoustic pulse vertically beneath the vessel to measure seabed depth, providing limited coverage directly below the survey platform. To overcome this limitation, multibeam echo sounders employ arrays of narrow acoustic beams, allowing wide-swath bathymetric mapping

and, in many systems, the acquisition of acoustic backscatter information. For high-resolution seabed imaging, side-scan sonar is the preferred technology, transmitting fan-shaped beams on both sides of the platform to produce detailed acoustic images over a much larger area, making it particularly suitable for mine detection and seabed object classification.

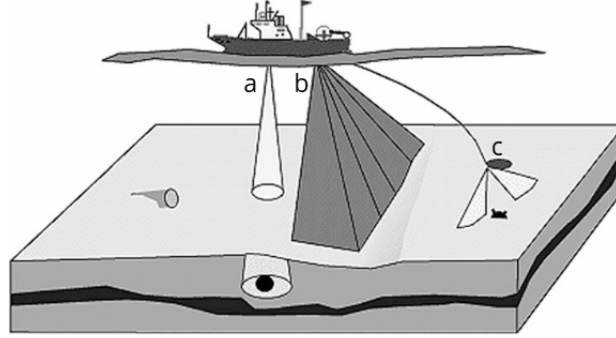


Figure 1.6: Comparison of the main sonar mapping geometries: single-beam echo-sounder (a), multibeam echo-sounder (b), and sidescan sonar (c), adapted from [17].

Building upon the principles of side-scan sonar, Synthetic Aperture Sonar (SAS) further enhances imaging performance by combining successive acoustic measurements through coherent signal processing, enabling significantly higher resolution.

1.3.4 Synthetic aperture sonar

Synthetic Aperture Sonar (SAS) is an advanced sonar technique that produces high-resolution images of the seafloor. Its principle originates from Synthetic Aperture Radar (SAR), a technology widely used in satellites to generate 2D, 3D, and topographic images of the Earth's surface [17].

1.3.4.1 Operating principle of SAS

SAS addresses the resolution limitations of conventional sonar, for which along-track resolution generally decreases as the range increases. Instead of increasing the physical length of the sonar array, which is constrained by the size of platforms such as AUVs [18], SAS uses the motion of the platform to create a synthetic aperture.

As the sonar platform moves along the survey track, it repeatedly transmits acoustic pulses, called *pings*, and records the echoes reflected from the same area from different along-track positions. These measurements are then coherently combined through signal processing to form a single image. By combining the returns from successive pings, SAS

effectively creates a much longer synthetic array, which allows it to achieve high along-track resolution that is largely independent of range [17].

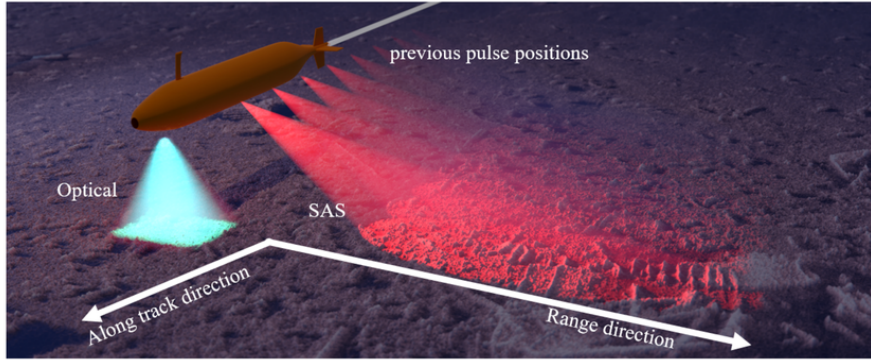


Figure 1.7: Geometry of data collection from a SAS AUV. The platform moves along the along-track direction while repeatedly insonifying the seafloor with successive pings; the range direction, perpendicular to the platform’s motion, indicates the distance from the sonar to a given point on the seabed. The SAS footprint (red) overlaps between successive pings, in contrast to a narrower single-shot optical footprint [19].

Two directions define the imaging geometry of an AUV-mounted sonar, as illustrated in Fig. 1.7. The *along-track direction* (cross-range) corresponds to the direction of motion of the platform, while the *range direction* is measured perpendicular to the platform’s motion, from the sonar to a given point on the seafloor. In conventional real-aperture sonar, resolution in the along-track direction degrades as range increases, since the physical beamwidth spreads out with distance. SAS addresses this limitation by combining the returns from successive pings, taken at known along-track positions, to synthesize a much longer virtual array. This allows SAS to achieve along-track resolution that is largely independent of range, unlike single-ping sonar systems.

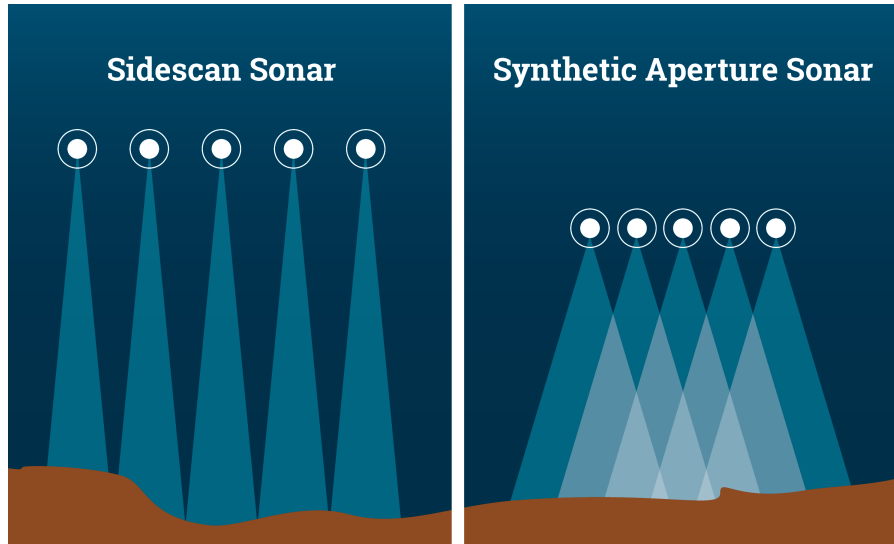


Figure 1.8: Side-scan sonar (left) versus synthetic aperture sonar (right). Side-scan pulses image separate, non-overlapping strips of the seafloor. SAS pulses instead overlap along the track, so that the same seafloor patch is observed from multiple along-track positions. [18].

This principle is illustrated further in Figure 1.8, which compares side scan sonar with SAS. In conventional side scan sonar, successive acoustic pulses produce separate sections of the seafloor, and the along-track resolution is mainly determined by the physical characteristics of the sonar. SAS, in contrast, deliberately overlaps the footprints of successive pings, allowing the same area to be observed from multiple along-track positions. The coherent combination of these observations enables the platform’s motion to act as a much longer synthetic array, resulting in substantially improved along-track resolution [17].

1.3.4.2 Signal Processing of SAS Data

Accurate SAS image formation depends on precise knowledge of the platform’s trajectory, sensor orientation, environmental conditions, particularly the sound speed in water, and seabed characteristics. These factors must be accurately estimated or compensated for before image reconstruction.

Data Segmentation The initial stage consists of dividing the recorded sonar data into smaller segments suitable for synthetic aperture formation. This determines how successive pings are grouped to construct each synthetic aperture. When required, high precision motion compensation, commonly referred to as micronavigation, is applied before image reconstruction to account for deviations in the platform’s trajectory.

Image Formation Techniques SAS image reconstruction can be performed using either frequency domain or time domain approaches. Frequency domain methods are generally computationally efficient and are well suited to stable and approximately linear platform trajectories. Examples include the wavenumber ($-k$) algorithm and chirp scaling methods.

In contrast, time domain methods provide greater flexibility and can accommodate irregular platform motion and non uniform imaging grids. This flexibility generally comes at the cost of increased computational complexity.

Beamforming and Image Construction At this stage, the selected pings are coherently combined to reconstruct the acoustic reflectivity of the seafloor. The beamforming process aligns the received signals according to their expected propagation paths and combines them to produce a focused image. The accuracy of the resulting image depends on the quality of the motion estimates, the acoustic propagation model, and the selected image formation method.

Autofocus and Residual Error Compensation Following image formation, additional processing can be applied to reduce residual phase errors caused by inaccuracies in motion estimation or uncertainties in the acoustic environment. Autofocus techniques use information contained in the received data to improve image sharpness and compensate for remaining phase errors. However, their effectiveness may be limited when these errors vary significantly across a large scene.

1.3.5 Low-Frequency vs High-Frequency Sonar

The operating frequency of a sonar system is a key parameter that directly influences both spatial resolution and propagation capability. In general, sonar systems can be broadly categorized into low-frequency (typically below 100 kHz)[20] and high-frequency systems (typically above 300 kHz)[20], although these limits may vary depending on the system design and application [21].

High-frequency SAS systems provide higher spatial resolution and clearer acoustic shadows, which significantly improves the detection of small objects. In contrast, low-frequency systems offer lower resolution but enable wider coverage and longer detection ranges. This difference in image quality is illustrated in Fig. 1.9, where the same object appears with strong shadow contrast in high frequency, while it is less distinct in the low-frequency image.

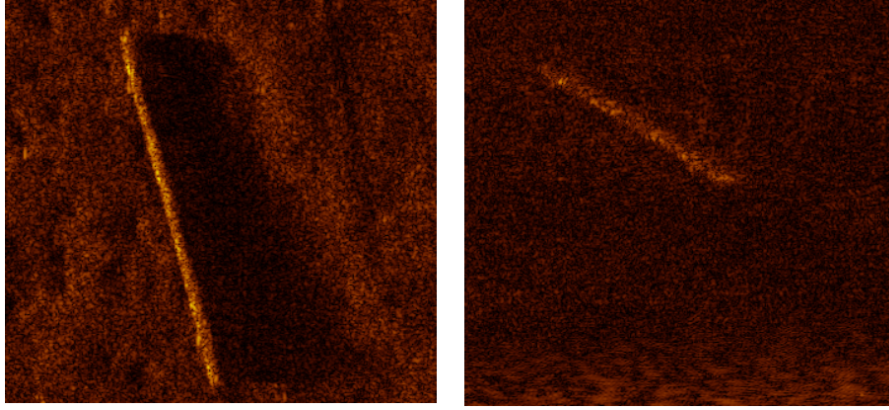


Figure 1.9: The image on the left is a high-frequency close-up view of a 3 meter pipe. On the right is a low-frequency close-up view of the same object[21]

To summarize these differences, Table 1.2 presents a comparison between low- and high-frequency SAS systems in terms of performance and typical applications.

Table 1.2: Comparison between low- and high-frequency SAS systems

Parameter	Low-Frequency SAS	High-Frequency SAS
Typical Frequency Range	Few kHz – ~ 100 kHz	~ 100 kHz – 1 MHz
Propagation Range	Long (low attenuation)	Short (high attenuation)
Spatial Resolution	Low	High (cm-scale)
Swath Coverage	Wide	Narrower
Shadow Visibility	Weak or absent for small objects	Clear and well-defined
Detection Capability	Large / buried objects	Small, exposed objects
Typical Use	Reconnaissance, wide-area mapping	Mine detection, detailed seabed imaging

Overall, high-frequency SAS is better suited for high-resolution imaging and precise target detection, whereas low-frequency SAS is preferred for wide-area coverage and potential subsurface investigation. Consequently, many modern systems employ dual-frequency configurations to exploit the complementary advantages of both regimes.

1.3.6 The LFM Chirp Signal

The transmitted waveform used for acoustic sensing is a Linear Frequency Modulated (LFM) chirp signal. An LFM chirp is a type of frequency-modulated signal in which the instantaneous frequency varies linearly with time over a defined duration [22], as illustrated in Fig. 1.10. Typically, the signal sweeps from a starting frequency (f_0) to a final frequency (f_1), covering a bandwidth (B) within a time interval (T).

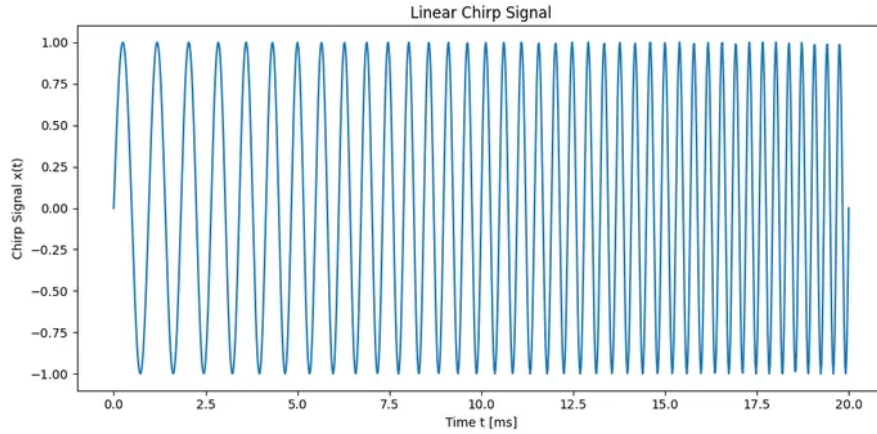


Figure 1.10: A plot showing a linear chirp signal [22].

This type of signal is widely employed in sonar systems due to its ability to achieve high range resolution while maintaining a high signal-to-noise ratio. By transmitting a long-duration, wideband signal and applying pulse compression techniques at the receiver, the LFM chirp enables precise target detection and improved imaging performance [17].

1.4 State of the Art

The classification of mine-like objects from sonar data has been extensively studied, with approaches primarily differing in the type of data representation used. Two dominant paradigms can be identified: image-based methods, which operate on sonar imagery, and time-series-based methods, which exploit the raw or processed acoustic echo signals prior to image formation. These paradigms involve different assumptions, feature representations, and learning strategies.

1.4.1 Image-Based Approaches

Historically, most SAS-based mine classification work has operated on the sonar *image* rather than the raw acoustic signal, following two main strategies: hand-crafted geometric features

combined with classical classifiers, and deep learning applied directly to image tiles. In the feature-based strand, Del Rio Vera et al. [23] extracted highlight/shadow geometric descriptors from segmented SAS images and matched them via a nearest-neighbour classifier, reaching 95% accuracy on simulated data, while Tucker et al. [24] applied canonical correlation analysis (CCA) to highlight/shadow regions, achieving 88% accuracy on real HF-SAS imagery. More recently, deep learning has become dominant: Williams [25] showed that a small CNN applied to HF-SAS image tiles could match human expert performance (98.7% accuracy), and Gebhardt et al. [26] confirmed this trend with a DNN using global average pooling, reaching 93–98% accuracy while consistently outperforming SVM baselines. Table 1.3 summarises these four representative approaches. However, all of them are validated exclusively on high-frequency imagery, where acoustic shadows are well-defined; their applicability to the low-frequency regime used in this work, where shadow visibility is limited and signal-level information is comparatively more informative, remains largely unexplored.

Table 1.3: Image-based methods for SAS mine classification.

Reference	Dataset	Features	Classifier	Acc.	Limitation
Del Rio Vera et al. [23]	Simulated SAS (NURC)	Geometric highlight/shadow	Nearest-neighbour matching	95%	no real SAS validation
Tucker et al. [24]	HF-SAS	CCA highlight/shadow	CCA-based classifier	88%	Requires clear highlight/shadow contrast
Williams [25]	HF-SAS	Image tiles	CNN	98.7%	Requires large labelled dataset
Gebhardt et al. [26]	Sidescan/SAS	ROI tiles	DNN + GAP	93%	Large model size

1.4.2 Classification from Acoustic Time Series

A parallel and comparatively less explored line of research focuses on the classification of underwater targets directly from time-domain or spectral representations of acoustic backscatter, thereby bypassing the image formation stage. These approaches typically rely on physics-informed modelling or statistical feature extraction, coupled with classical or shallow learning algorithms. Table 1.5 summarises three representative studies along this line,

discussed in detail below.

Bucaro et al. [27] employed a finite-element structural acoustics model to simulate the scattering response of a buried unexploded ordnance (UXO) target, using the resulting signatures to train a Relevance Vector Machine (RVM). While the model demonstrated successful generalisation to real measurements acquired in a controlled sediment environment, the study is limited by its focus on a single target geometry and controlled experimental conditions.

Seo et al. [28] proposed a classification framework based on the discrete Fourier transform (DFT) magnitude spectrum of active sonar returns, combined with a logistic regression model to distinguish cylindrical objects from clutter. Although validated on both simulated and experimental data, the approach is restricted to a binary classification task and depends on simplified statistical models of clutter, which may not fully capture real environmental variability.

Xu, Li, and Jia [29] investigated the classification of underwater spherical shells using features derived from the acoustic backscattering morphological function, comparing a deep convolutional neural network (CNN) with a traditional feature-engineering-based classifier. Their results demonstrate the superiority of learned representations in noisy conditions; however, the evaluation is conducted exclusively on synthetic data, limiting the assessment of real-world generalisation.

These limitations underscore the need for approaches capable of handling real, heterogeneous acoustic data, as addressed in the present work.

Table 1.5: Feature- and model-based methods for time-series (signal-level) target classification.

Reference	Target	Features	Classifier	Data
Bucaro et al. [27]	UXO vs. clutter (buried)	Simulated backscatter	RVM	Synth. + Real
Seo et al. [28]	Cylinder vs. clutter	DFT spectrum	Logistic Reg.	Synth. + Real
Xu et al. [29]	Shell material	Form function	CNN vs. feat. eng.	Synthetic

1.4.3 Positioning of This Work

As summarised in Table 1.3, existing SAS mine classification methods predominantly operate on high-frequency (HF) imagery, exploiting fine spatial resolution and well-defined acoustic shadows as the primary source of discriminative information. The time-series-based methods reviewed in Table 1.5, while bypassing image formation, remain confined to narrow, single-target-type settings and rely predominantly on simulated data, with real-world validation, when present, restricted to controlled experimental conditions.

In contrast, this work focuses on the low-frequency (LF) regime using the TREX13 dataset, where image quality is inherently lower and acoustic shadows are often weak or absent, particularly for small targets. Rather than relying on image-based representations, this study extracts time-series features directly from focused LF-SAS signals, thereby bypassing the image formation stage entirely. Furthermore, unlike the reviewed time-series approaches, which are trained largely on synthetic data, the present pipeline is trained and evaluated exclusively on real, multi-target acoustic measurements.

Model evaluation is performed using a *GroupKFold* cross-validation strategy, in which samples are grouped according to physical target identity. This protocol prevents data leakage between training and testing sets and provides a more realistic assessment of generalisation performance, representing a level of methodological rigor that is rarely adopted across either paradigm reviewed above.

Conclusion

Throughout this chapter, the operational context of underwater mine countermeasures was established, along with the key technical concepts underlying the project. The principles of synthetic aperture sonar, the trade-offs between low- and high-frequency operation, and the existing classification approaches for mine-like objects were examined. The state of the art review identified a clear gap in the literature regarding signal-level classification on low-frequency SAS data, which directly motivates the methodology developed in the following chapter.

DATA PROCESSING AND CLASSIFICATION METHODOLOGY

Plan

2.1	Dataset Description	26
2.2	Signal Processing Pipeline	29
2.3	Data Preparation for the Classifiers	38
2.4	Machine Learning Classifiers and Evaluation Protocol	48

Introduction

This chapter presents the data and methodology used in this work for the classification of mine-like and non-mine-like underwater targets. It first describes the dataset and the experimental conditions, followed by the signal processing steps used to obtain the target signals. The chapter then presents the feature extraction and preprocessing procedures, and finally introduces the machine learning models used for classification.

2.1 Dataset Description

The data used in this work originates from the Target and Reverberation Experiment 2013 (TREX13), a large-scale ocean acoustics measurement campaign conducted off the coast of Panama City, Florida, between April and June 2013[30]. The experiment was jointly sponsored by the U.S. Office of Naval Research (ONR) and the Strategic Environmental Research and Development Program (SERDP), and involved several research vessels, including the R/V Sharp, the R/V Walton Smith, and the Canadian Forces Auxiliary Vessel *Quest*[31]. The primary objective of TREX13 was to jointly measure acoustic reverberation, transmission loss, and bottom backscatter alongside detailed environmental parameters (bathymetry, sediment properties, sea-surface conditions), enabling rigorous model-to-data comparison and the identification of environmental factors governing sonar performance in shallow water[31]. As part of the broader campaign, a dedicated synthetic aperture sonar (SAS) target and clutter identification experiment was conducted using a rail-based acquisition system, following an experimental protocol inherited from the earlier GULFEX12 campaign, in which each target configuration was surveyed over multiple passes to compensate for dropouts caused by environmental artifacts such as fish[31]



Figure 2.1: Setup of TREX13 trial area off Panama City, Florida.[32]

Beyond its immediate acoustic measurements, TREX13 was designed to serve as ground truth for the validation of finite-element (FE) scattering models, which, once validated, could be exercised to generate free-field scattering amplitudes for use in fast ray-based propagation models; the resulting combined FE-and-ray modelling framework offers substantial cost savings by reducing the number and duration of future field campaigns, while supporting SERDP in making informed decisions on sonar system performance for underwater UXO management[31]. In addition to this model-validation role, the target and clutter scattering data collected during TREX13 provide a real, multi-target acoustic dataset suitable for the direct development and evaluation of classification algorithms, offering a benefit that extends beyond model calibration: the ability to build and test classifiers against genuine field-measured signals rather than purely synthetic data, which is the benefit exploited in the present work.

2.1.1 Experiment Configuration

The APL-UW tower-rail SAS system used for TREX13 consisted of a 42-m long rail. A sonar tower carrying the transmit/receive electronics traversed the rail, acquiring data over two frequency bands, 1–30 kHz and 30–50 kHz, transmitted as linear frequency-modulated (LFM) chirps[31]. It is the low-frequency (1–30 kHz) band that constitutes the LF-SAS regime exploited in the present work.

Targets were rotated through a series of aspect angles with respect to the rail, nominally from -80° to 80° in 20° increments for symmetric targets, extended to a full 360° rotation

for non-symmetric objects[31]. Beyond aspect-angle rotation, targets were also deployed under varying burial states, ranging from *proud* (resting on the water-sediment interface) to *partially* and *fully buried*, as illustrated in Fig. 2.2[31]. This variability in burial condition is directly relevant to the LF-SAS regime used in this work, as low frequency offers improved sediment penetration and sensitivity to buried or partially buried objects compared to higher-frequency SAS.



Figure 2.2: Selection of targets deployed during TREX13. (left) Munitions targets, including bullet-shaped and finned UXO, a TP-T round, and an aluminum cylinder/pipe. (middle) Clutter items, including a telephone pole section, a scuba tank, and a 55-gallon drum. (right) Illustration of target burial state on the seafloor[31].

2.1.2 Data Structure

The data used in this work consist of calibrated cross-range/time scattering signals, extracted from the pulse-compressed, baseband SAS returns collected in the 1–30 kHz LF band described above, prior to any image formation. Each signal corresponds to the isolated scattering response of a single physical target at a given aspect angle and burial condition.

The dataset comprises 855 files, acquired over 27 distinct physical targets, of which 12 are classified as Mine-Like Objects (MILCO) and the remaining 15 as Non-MILCO (Clutter). Among the 27 targets, 8 were partially buried; Fig. 2.2 (right) illustrates the target burial,

while the remaining targets were surface-laid. Burial condition is documented per target in Table 4.3 in Appendix A. The complete list of target names and corresponding file counts is also provided in the same table. Each signal is labelled according to this binary scheme, assigned at the target level rather than per-signal. Because each physical target contributes multiple signals, corresponding to its different aspect angles and range, the dataset exhibits a natural grouped structure, in which samples from the same target are strongly correlated; this structure is what later motivates the target-grouped cross-validation strategy adopted in this work.

2.2 Signal Processing Pipeline

The formation of SAS data begins with the emission of successive linear frequency-modulated (LFM) pulses, or *pings*, transmitted and received by a source and a multi-element receiving array mounted on the sonar platform. At each along-track position, the backscattered return, or *echo*, is acquired and digitized for further processing, as illustrated in Fig. 2.3[31].

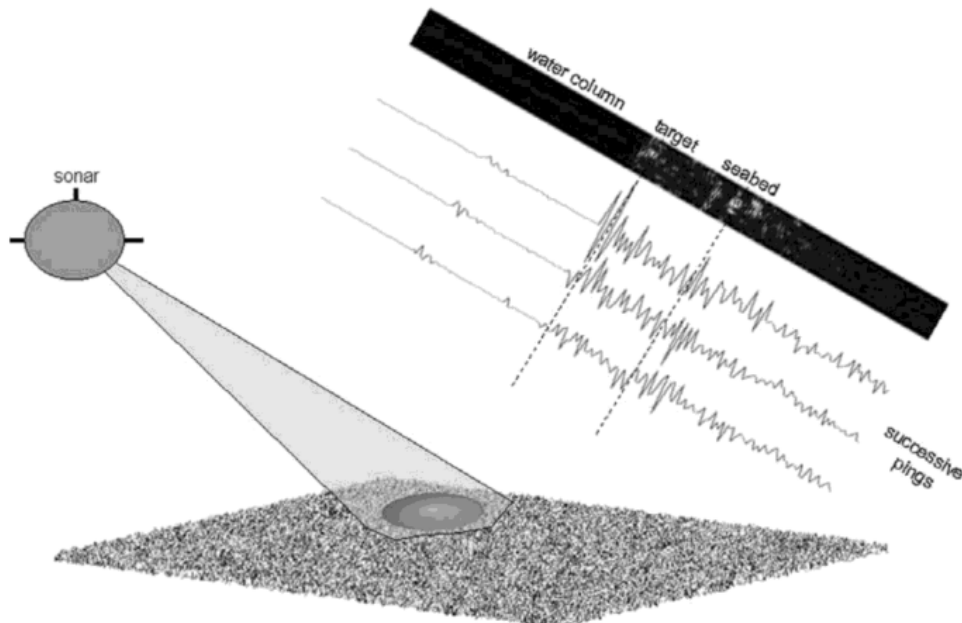


Figure 2.3: Successive pings recorded as the sonar moves past a target. Each ping’s backscattered return contains contributions from the water column, the target, and the seabed, arriving at different propagation delays.[17].

Each recorded echo encodes the propagation delay, and thus the slant range, between the sonar and every reflecting element in the scene. To improve range resolution and signal-to-noise ratio, each raw echo is first *pulse-compressed* by cross-correlating it with a reference replica of the transmitted LFM waveform, yielding a pulse-compressed baseband

(PCB) time signal[31]. Successive PCB signals, one per along-track position, are then stacked to form a two-dimensional raw data matrix, with one dimension corresponding to *slow time* (along-track, or cross-range) and the other to *fast time* (propagation delay, or range). At this stage, the data remains *unfocused*, because the slant range between the sonar and a given point target varies continuously as the platform approaches and recedes from it; the pulse-compressed energy associated with that target does not concentrate at a single location in the raw data matrix, but instead traces a curved, hyperbolic locus spanning multiple range and along-track samples. This behavior is illustrated in Fig. 2.4 (top), where the target response appears as a distinct arc rather than a localized point.

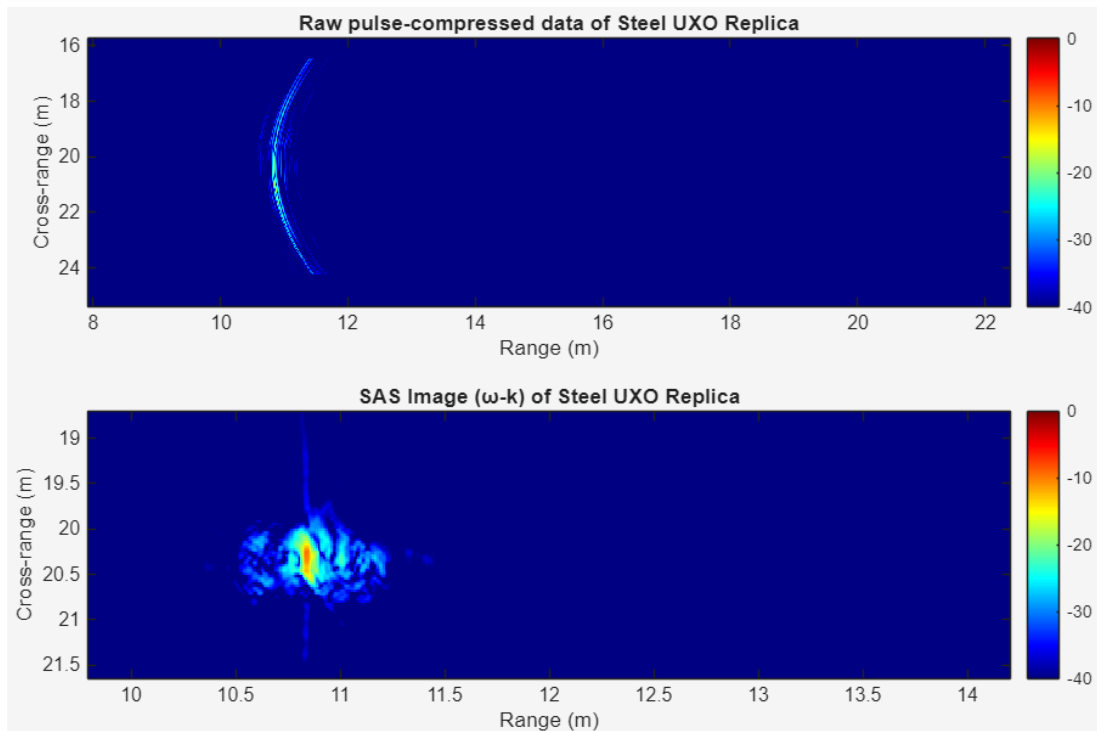


Figure 2.4: Raw pulse-compressed data (top) and the corresponding focused SAS image obtained via the ω -k algorithm (bottom), for a steel UXO replica from the TREX13 dataset. In the raw data, the target response traces a hyperbolic arc; after focusing, the energy is concentrated at the true target location.

Recovering a spatially accurate SAS image from this raw, unfocused representation therefore requires a coherent focusing step, capable of compensating for the range- and aperture-dependent propagation delay and of concentrating the scattered contributions of the target back into a single, well-localized response like the one in the bottom image of Fig. 2.4. This is the role of the ω -k algorithm, described in the following section.

2.2.1 The ω -k Focusing Algorithm

Among the available SAS image-formation methods, the ω -k algorithm was selected because it is computationally more efficient than time-domain backprojection for regularly sampled linear apertures[33]. Originally developed for seismic migration¹[34], the algorithm was later adapted to synthetic aperture radar and sonar imaging, where the focusing operation is expressed as a coordinate transformation, or Stolt mapping, performed in the two-dimensional wavenumber domain, followed by a single inverse Fourier transform[35]. The implementation used in this work follows the sequence of five operations summarized in Fig. 2.5, and detailed in the following subsections.

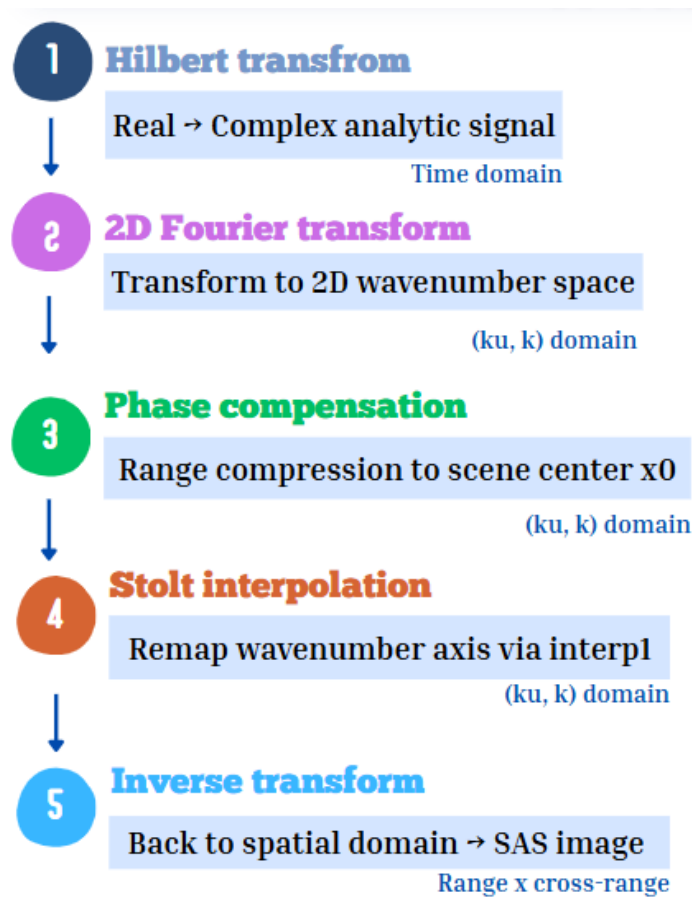


Figure 2.5: Sequence of operations in the ω -k focusing algorithm.

2.2.2 Notation and Wavenumber Variables

Before describing the five processing steps, the wavenumber variables used in the ω -k algorithm are defined here. Let u denote the along-track coordinate and x the range coordinate, with

¹Seismic migration is a wavefield imaging process that reconstructs the true spatial locations of subsurface reflectors from recorded seismic data.

corresponding spatial wavenumbers k_u and k_x . The acoustic wavenumber k is related to the fast-time frequency f by

$$k = \frac{2\pi f}{c}, \quad (2.1)$$

where c is the assumed sound speed. Thus, k is derived from the temporal frequency of the received signal, whereas k_u is obtained from the spatial variation along the synthetic aperture. Although k has units of rad/m, it is distinguished from the geometric spatial wavenumbers k_u and k_x .

In the numerical implementation, k and k_u are sampled over discrete sets of values. The spectrum is therefore represented as a two-dimensional matrix $S(k_u, k)$, whose axes correspond to the sampled along-track and acoustic wavenumbers. The range wavenumber k_x is introduced below through the wave-propagation relation.

The range of k_u is determined by the along-track sampling interval d_u . The corresponding spatial sampling frequency is $1/d_u$, so the Nyquist criterion gives

$$f_{u,\max} = \frac{1}{2d_u}. \quad (2.2)$$

Using $k_u = 2\pi f_u$, the maximum unaliased along-track wavenumber is

$$k_{u,\max} = 2\pi f_{u,\max} = \frac{\pi}{d_u}. \quad (2.3)$$

Consequently,

$$-\frac{\pi}{d_u} \leq k_u < \frac{\pi}{d_u}. \quad (2.4)$$

For two-way sonar propagation, the effective wavenumber magnitude is $2k$. Its decomposition into the along-track and range directions gives

$$(2k)^2 = k_u^2 + k_x^2, \quad (2.5)$$

from which

$$k_x = \sqrt{4k^2 - k_u^2}. \quad (2.6)$$

At broadside ($k_u = 0$), this relation reduces to $k_x = 2k$. For a propagating wave, k_x must be

real-valued, which requires

$$|k_u| \leq 2k. \quad (2.7)$$

Values satisfying $|k_u| > 2k$ lead to imaginary k_x and correspond to non-propagating components; these are excluded from the subsequent processing.

2.2.3 Analytic signal formation:

Each range line of the raw pulse-compressed data matrix is first converted to its analytic representation by combining the original signal with its Hilbert transform. The resulting analytic signal has no negative-frequency components, enabling the frequency-domain processing described below to operate on a one-sided spectrum:

$$s_a(t) = s(t) + j \mathcal{H}\{s(t)\}, \quad (2.8)$$

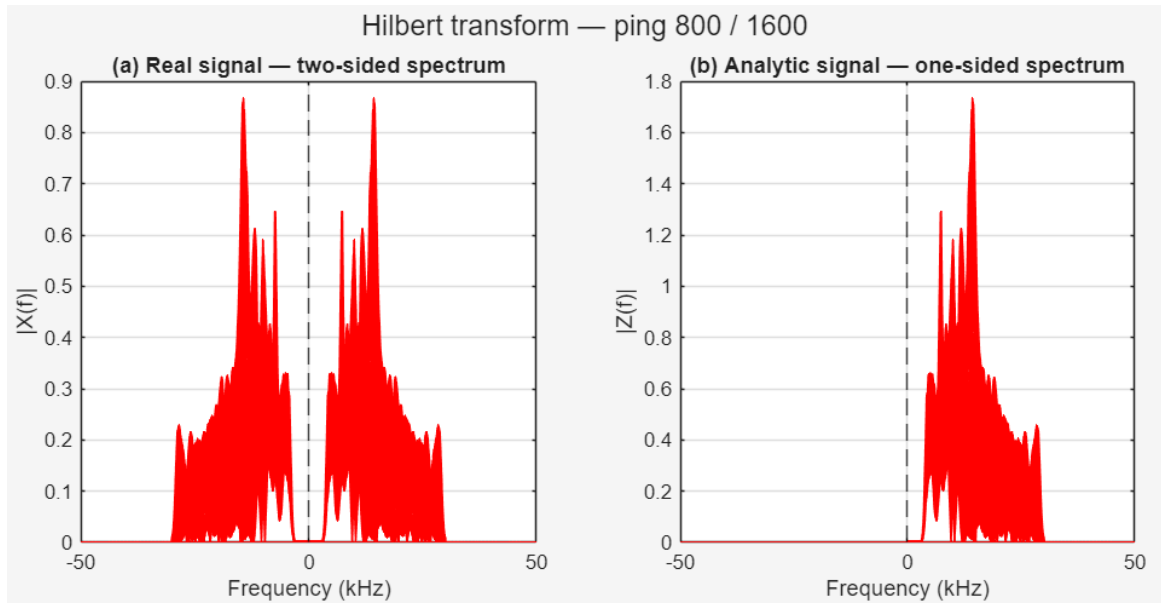


Figure 2.6: Effect of the Hilbert transform on a single range line (ping 800/1600). (a) Spectrum of the raw real-valued signal, exhibiting the expected symmetric, two-sided structure around 0 Hz. (b) Spectrum of the resulting analytic signal, in which the negative-frequency content has been suppressed, yielding a one-sided spectrum suitable for the wavenumber processing described below.

where $s(t)$ is the raw pulse-compressed range line and $\mathcal{H}\{\cdot\}$ denotes the Hilbert transform operator. This step is required prior to the wavenumber processing; the phase compensation and Stolt interpolation operations are formulated under the assumption of a single-valued, unambiguous acoustic wavenumber $k = 2\pi f/c$; without first removing the redundant negative-frequency content, both operations would act inconsistently on the mirrored

spectral components, producing spurious energy, or ghost targets, in the focused image.

2.2.4 Transformation to the wavenumber domain

The analytic data matrix is transformed into the two-dimensional wavenumber domain by a 2-D Fast Fourier Transform (FFT), followed by an `fftshift` along the along-track (cross-range) dimension, as shown in Fig. 2.7. This yields a representation of the data as a function of the acoustic wavenumber k and the along-track wavenumber k_u :

$$S(k_u, k) = \text{fftshift}(\text{FFT}_{2D}\{s_a(u, t)\}), \quad (2.9)$$

with

$$k = \frac{2\pi f}{c}, \quad k_u \in \left[-\frac{\pi}{du}, \frac{\pi}{du}\right], \quad (2.10)$$

where f is the fast-time frequency, c the assumed sound speed, and du the along-track sample spacing.

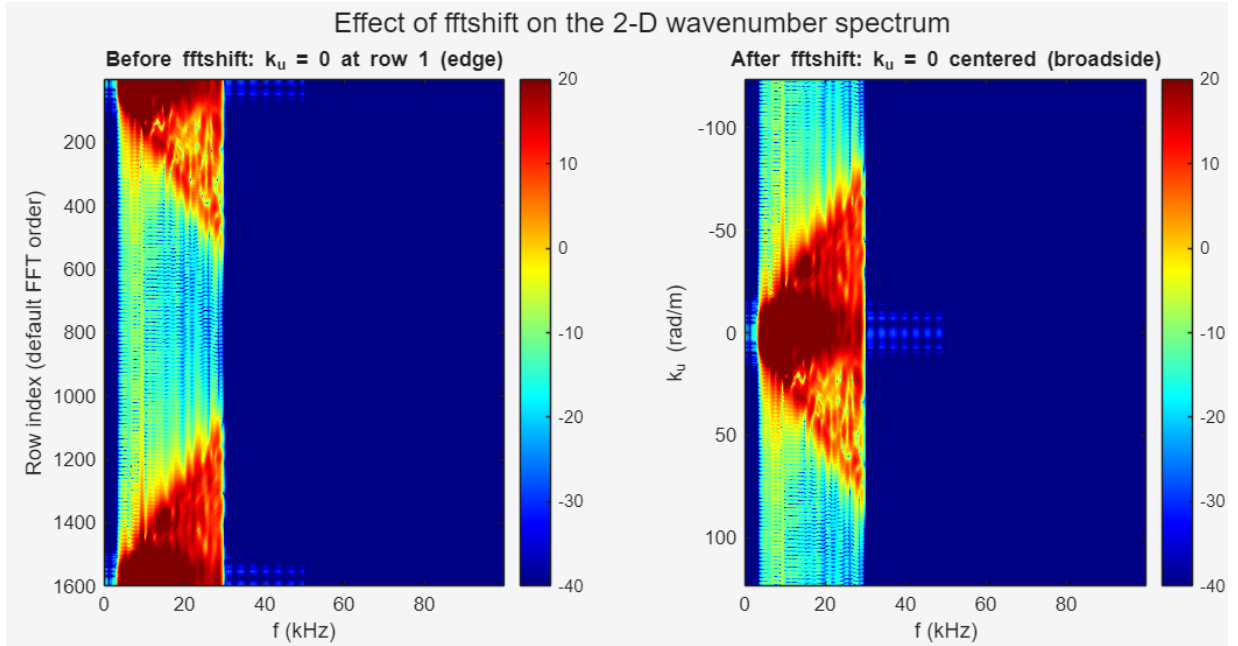


Figure 2.7: Effect of `fftshift` on the two-dimensional wavenumber spectrum DK (Domain in K wavenumber). (left) Before shifting, $k_u = 0$ lies at row 1, and broadside energy is split between the top and bottom edges of the matrix due to the default FFT frequency ordering. (right) After shifting, $k_u = 0$ is centered, and the broadside energy forms a single continuous, symmetric feature.

This step is necessary because both the phase compensation and the Stolt interpolation described below are formulated directly in the (k_u, k) domain, rather than in the original space-time domain of the raw data; the 2-D FFT is therefore what converts the analytic signal

matrix, indexed by along-track position and fast time, into its wavenumber-domain equivalent, indexed by k_u and k . The subsequent `fftshift` operation reorders the rows of this matrix so that $k_u = 0$, corresponding broadside (the direction perpendicular to the sonar's track) falls at the center of the array rather than at its edge. As illustrated in Fig. 2.7 (left), without this reordering the energy concentrated near broadside appears split between the top and bottom edges of the matrix, a direct consequence of the default FFT frequency ordering; after shifting (right), this energy forms a single, continuous, symmetric feature centered on $k_u = 0$, matching the physical structure assumed by the subsequent processing steps.

2.2.5 Range compression to scene center

Before the Stolt interpolation step can be applied, the two-dimensional wavenumber spectrum must first be corrected for the fact that every point scatterer in the raw data was recorded relative to its own actual range, rather than a common reference range. This is achieved by applying a phase-only matched filter, which effectively re-centers, or refocuses, the data to a chosen reference range x_0 at the middle of the imaged scene so that only the deviation from this reference range needs to be corrected in the subsequent interpolation step:

$$A(k_u, k) = \begin{cases} x_0 \sqrt{4k^2 - k_u^2}, & \text{if } \sqrt{4k^2 - k_u^2} \in \mathbb{R}, \\ 0, & \text{otherwise,} \end{cases} \quad (2.11)$$

$$S_c(k_u, k) = S(k_u, k) \exp(jA(k_u, k)), \quad (2.12)$$

where $A(k_u, k)$ is the compensation phase and $S_c(k_u, k)$ the resulting range-compressed spectrum.

The filter in Eq. 2.12 is applied purely as a phase term, i.e. it multiplies the spectrum by a complex exponential of unit magnitude, leaving the amplitude of $S(k_u, k)$ unchanged and modifying only its phase. This is what makes the operation a *focusing* step rather than a filtering-out of any part of the signal: no energy is removed, only the relative phase relationships between frequency components are adjusted, so as to align the data as if it had all been recorded at the common reference range x_0 .

The condition in Eq. 2.11 restricts this compensation to the region of the (k_u, k) domain where $\sqrt{4k^2 - k_u^2}$ is real-valued. This quantity becomes imaginary whenever

$k_u > 2k$, a condition corresponding to spatial frequencies that do not correspond to any physically propagating wave (evanescent components); setting the phase term to zero in this region, rather than allowing it to take a complex, non-physical value, avoids introducing spurious amplification or artifacts into a part of the spectrum that carries no meaningful propagating-wave information.

2.2.6 Stolt interpolation

Before Stolt interpolation, the spectrum is represented in the mixed coordinates (k_u, k) (Section 2.2.2), whereas image formation requires a spectrum defined on the spatial-frequency grid (k_u, k_x) . Since

$$k_x = \sqrt{4k^2 - k_u^2} \quad (2.13)$$

(Eq. 2.6) is a nonlinear function of k_u , uniformly sampled values of k do not correspond to uniformly spaced values of k_x . Stolt interpolation therefore resamples the spectrum for each fixed k_u , mapping the samples from k to their corresponding k_x values and interpolating them onto a uniform k_x grid. Only the samples for which k_x remains real-valued are retained. The resulting transformation can be expressed as

$$S_c(k_u, k) \xrightarrow{\text{Stolt interpolation}} \hat{S}(k_u, k_x). \quad (2.14)$$

Geometrically, this resampling transforms the hyperbolic phase-history locus associated with a point scatterer into a form compatible with a rectangular wavenumber grid, allowing the focused image to be obtained through an inverse Fourier transform[33], [34].

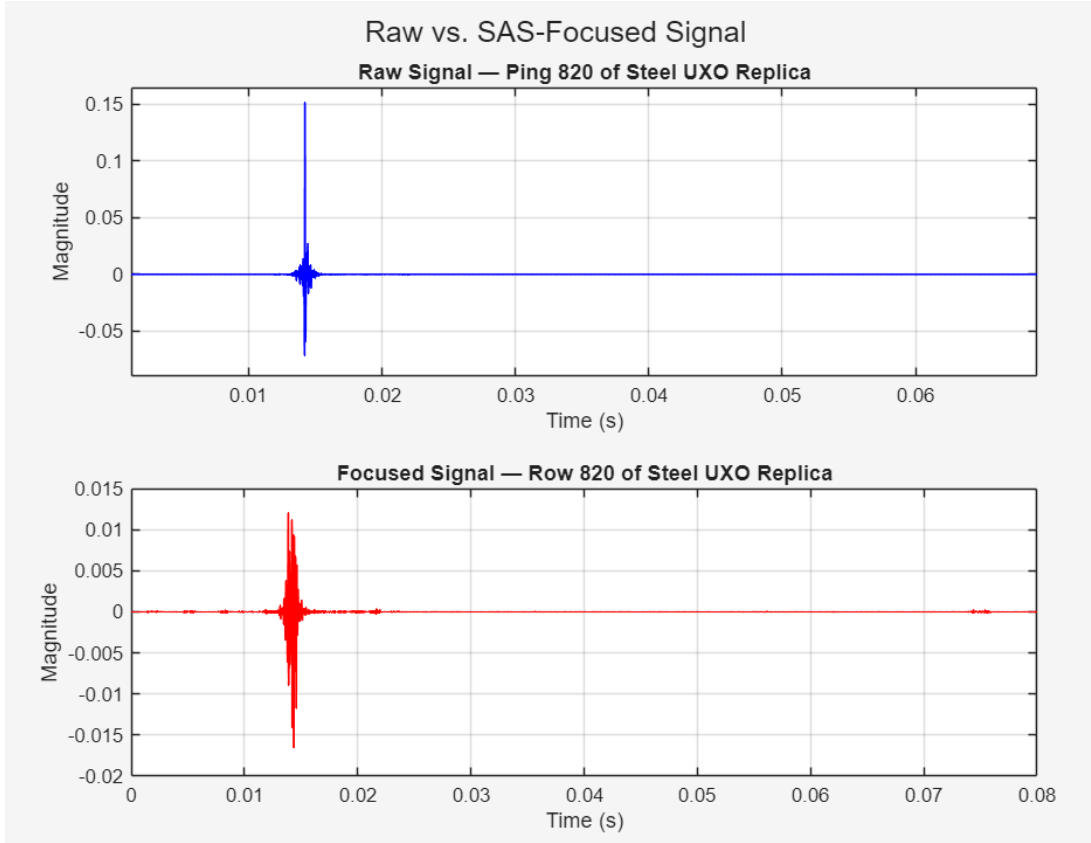


Figure 2.8: Comparison of a single ping’s raw pulse-compressed signal (top) and its SAS-focused counterpart (bottom) at the same along-track position. The focused signal exhibits a sharper temporal concentration of the target response.

The focusing process produces a more localized target response, as illustrated in Fig. 2.8. In this work, the time-domain signals used for feature extraction are obtained from the SAS-focused representation rather than directly from the raw pulse-compressed data. The sharper concentration of the focused response results from the coherent combination of returns acquired over the synthetic aperture, providing a more localized representation of the target echo for subsequent feature extraction.

2.2.7 Image formation

The focused SAS image is obtained by applying a two-dimensional inverse FFT to the Stolt-interpolated spectrum, followed by an `ifftshift` along the range dimension to restore the correct spatial ordering:

$$I(x, u) = \text{ifftshift}(\text{IFFT}_{2D}\{\hat{S}(k_u, k_x)\}). \quad (2.15)$$

As with the earlier `fftshift` step, this final `ifftshift` is a pure reordering operation, undoing the wavenumber-axis centering introduced earlier so that the inverse-transformed data is correctly indexed in the spatial (range) domain rather than in the shifted wavenumber convention. The resulting image, shown in Fig. 2.4 (top), concentrates the target’s scattered energy at its true spatial location.

2.3 Data Preparation for the Classifiers

The ω -k algorithm described above produces a focused SAS image for each target, aspect angle, and burial condition. However, each image covers the full scene, and there is no label telling us which rows actually contain the target versus background or noise. Since classification is performed on individual one-dimensional signals, not full images, we first need a way to pick out the rows that genuinely correspond to the target.

2.3.1 Active-Zone Detection and Stride-Based Subsampling

For each focused image, we compute the energy of every along-track row:

$$E[n] = \sum_m |\text{img}[n, m]|^2, \quad n = 1, \dots, N, \quad (2.16)$$

where $\text{img}[n, m]$ is the focused image, n is the row (along-track position), and m is the range sample within that row. This gives an energy profile along the track where the target shows up as a clear peak against a low, flat background.

A threshold on this energy profile identifies the *active zone*, the contiguous block of rows where energy is clearly above background. The threshold is set as a fixed fraction of the peak row energy,

$$E_{\text{th}} = \alpha \cdot \max_n E[n], \quad (2.17)$$

with $\alpha = 0.05$, so that any row whose energy exceeds 5% of the maximum is included in the active zone. In the example in Fig. 2.9, this gives an active zone of 85 rows.

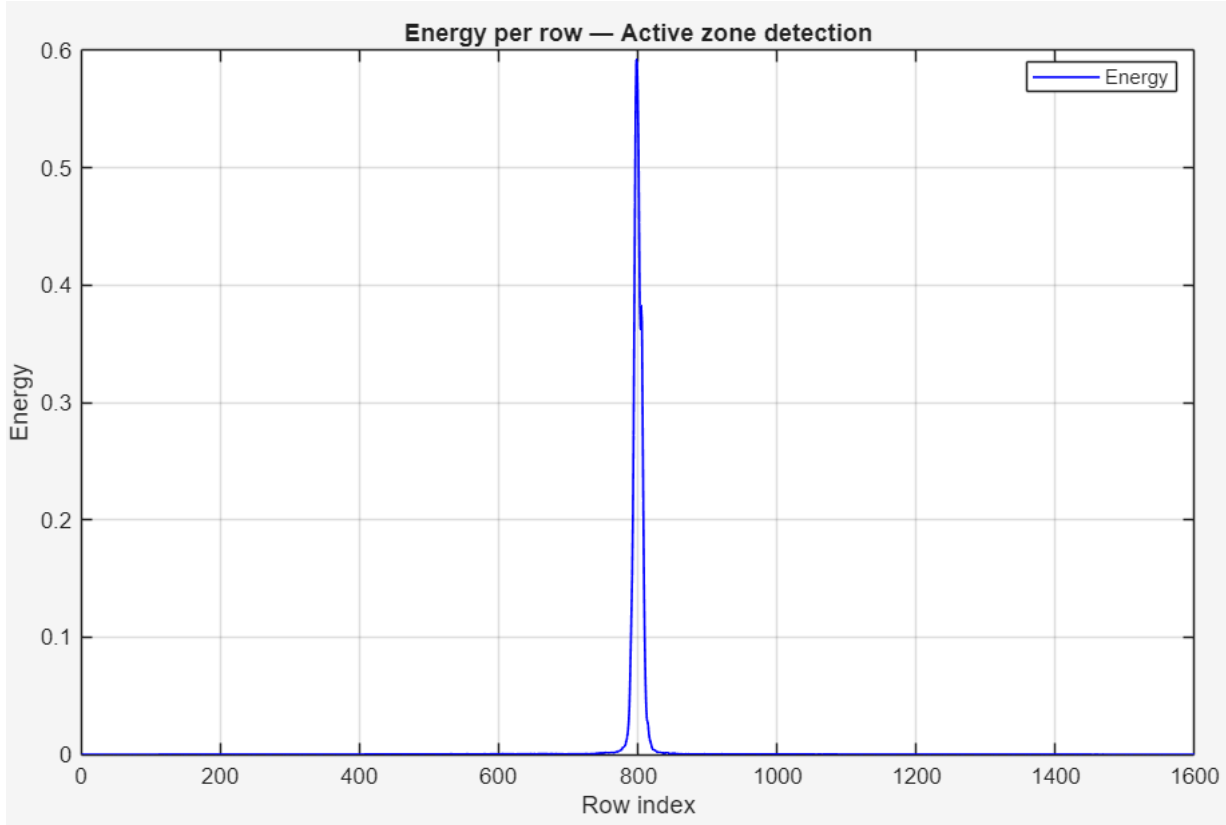


Figure 2.9: Energy is computed for each row, showing the peak used to detect the target’s active zone.

Since neighboring rows in the active zone are very similar (they come from almost the same along-track position), keeping all of them would just add redundant, highly correlated signals. To avoid this, we keep only every S row within the active zone, using a fixed stride of $S = 50$ applied uniformly across all 855 files. In the example shown in Fig. 2.9, this stride applied to the 85-row active zone selects 2 signals (rows 756 and 806).

Applying this active-zone detection and stride-based subsampling procedure across the full dataset yields a total of 1158 signals extracted from the 855 files, of which 624 correspond to MILCO targets and 519 to Non-MILCO (Clutter) targets.

2.3.2 Dataset Export and Metadata

Hierarchical Data Format version 5 (HDF5) is a file format designed for storing and organizing large, heterogeneous scientific datasets. Unlike a flat file or a simple array-based format, HDF5 organizes data in a structure analogous to a file system: *groups*, which behave like directories and can be nested arbitrarily deep, contain *datasets*, which store the actual numerical arrays, and both groups and datasets can carry *attributes*, small pieces of metadata (strings, numbers, or short arrays) attached directly to the object they describe[36]. This format was chosen for

exporting the signals extracted in this work because it allows the natural hierarchical structure of the TREX13 acquisition, target, range, and aspect angle, to be mirrored directly in the file's internal organization, while allowing the class label, target identity, and acquisition conditions to be stored as attributes directly alongside each corresponding set of signals, rather than in a separate index or lookup table.

The signals extracted using the active-zone detection and stride-based subsampling procedure described above are exported to a structured HDF5 file, organized hierarchically to reflect the physical structure of the acquisition.

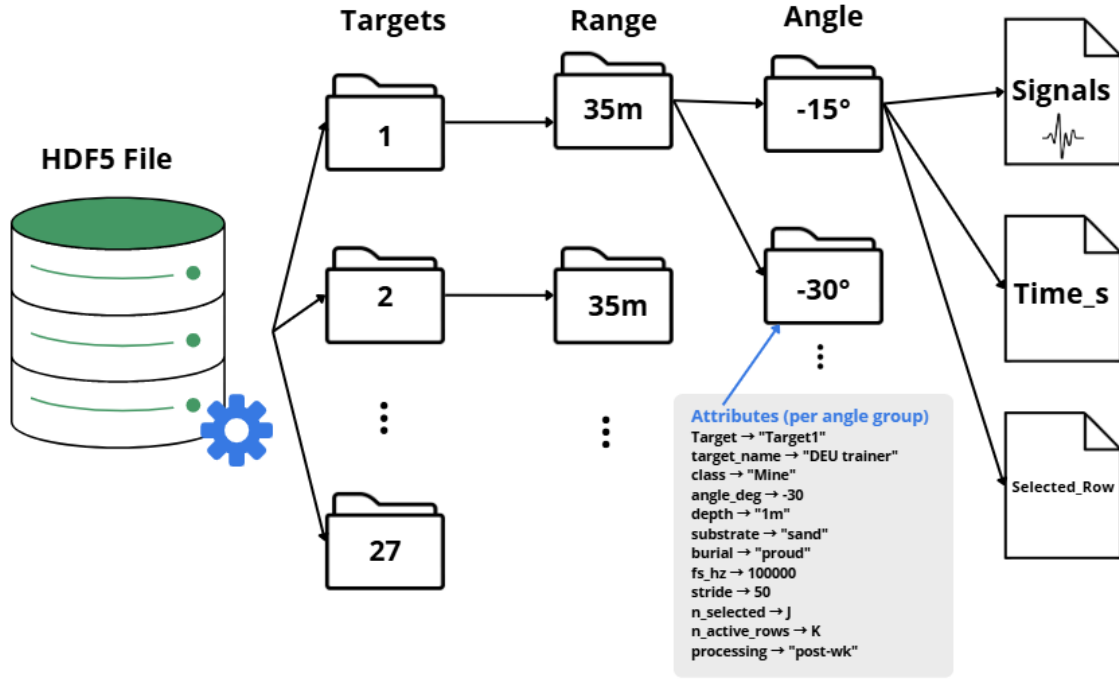


Figure 2.10: Hierarchical structure of the exported HDF5 dataset.

At the top level, the file contains one group per physical target (e.g. /Target1, ..., /Target27). Within each target group, data is further organized by acquisition range (e.g. 35m) and, within each range, by aspect angle relative to the rail (e.g. /-15Deg, /-30Deg). Each such angle group contains three datasets:

- **Signals**, a JM array containing the J extracted time-domain signals for this target/range/angle combination, each of length M samples.
- **Time_s**, a $1S$ array giving the corresponding time axis.
- **selected_rows**, recording the specific along-track row indices retained by the stride-based subsampling step, allowing each extracted signal to be traced back to its original position in the focused SAS image.

In addition to these datasets, each angle group carries a set of HDF5 attributes recording the metadata associated with that acquisition, including: the target identifier and target name (e.g. "DEU trainer"), the class label ("Mine" or "Clutter"), the aspect angle in degrees, the burial depth and condition (e.g. "proud"), the sediment substrate (e.g. "sand"), the sampling frequency, the stride value used for subsampling, the number of signals selected and the number of active rows detected, and a processing tag identifying the data as post- k ("post-wk"). Storing this metadata directly alongside each signal, rather than in a separate lookup table, ensures that every extracted signal remains self-describing and traceable to its full acquisition context, which is essential for the target-grouped cross-validation strategy adopted later in this work.

2.3.3 Feature Extraction with tsFresh

Following signal extraction and export, each raw time-domain signal must be converted into a fixed-length numerical representation suitable for classical machine learning classifiers, which, unlike deep learning architectures operating directly on raw sequences, require a finite set of descriptive input features rather than a variable-length time series. This is achieved using **tsFresh** (*Time Series Feature extraction based on Scalable Hypothesis tests*), an open-source Python library that automatically computes a large, standardized collection of statistical, spectral, and information-theoretic descriptors from time series data[37].

tsFresh's feature set is predefined rather than learned: the library ships with approximately 100 built-in *feature calculators*, each implementing a specific, well-established time-series characteristic (e.g. mean, skewness, autocorrelation, FFT coefficients, sample entropy). Since many of these calculators are parameterized, computing their output over a predefined grid of parameter values (e.g. autocorrelation at multiple lags, FFT coefficients at multiple frequency bins, quantiles at multiple levels), a single calculator can expand into several dozen individual output features. This is what allows tsFresh to generate several hundred features per signal from a comparatively small set of underlying calculators, with the calculators and their parameter grids fixed by the library itself, independent of the specific dataset to which they are applied.

tsFresh was chosen for this work over manual feature engineering for two main reasons. First, it systematizes and scales the kind of hand-crafted feature extraction traditionally used

in signal-level target classification, computing hundreds of established time-series descriptors, ranging from simple statistics (mean, variance, skewness) to more elaborate characterizations (autocorrelation, Fourier and wavelet coefficients, entropy measures, and nonlinear dynamics indicators), without requiring each one to be manually implemented and validated. Second, `tsFresh` provides a built-in, statistically principled feature selection procedure, based on hypothesis testing against the target label, which can be used to prune the extracted feature set down to those exhibiting significant predictive relevance, reducing the risk of overfitting introduced by a very high-dimensional feature space[37].

In this work, feature extraction was performed using `tsFresh`'s `extract_features` function, applied to the dataset reshaped into "long" format (one row per signal per time step), with the `EfficientFCParameters` setting selecting the reduced, computationally efficient subset of feature calculators described above. Applied to each extracted signal in this work, this configuration yields 777 features per signal. This initial feature set is subsequently cleaned and reduced, as described in the next section.

Table 2.1 summarises the main families of features present in the extracted set, grouped by the type of characteristic they capture.

Table 2.1: Main categories of tsFresh features present in the extracted feature set.

Category	Description / example calculators
Basic statistics	Sum of values, absolute energy, median, variation coefficient, skewness, minimum, count above mean, longest strike above/below mean, last location of minimum, sum of reoccurring values, Benford correlation
Distribution shape	Quantile (10th percentile), symmetry-looking, large standard deviation
Change & variability	Mean change, mean second derivative, change quantiles (mean aggregation)
Autocorrelation	Autocorrelation (lag 0), partial autocorrelation at lags 2–9, aggregated autocorrelation (mean and variance)
Frequency domain (FFT)	Imaginary part (2 coefficients) and phase angle (nearly full spectrum, coefficients 1, 4–99) of FFT coefficients
Wavelet-based	Continuous wavelet transform (CWT) coefficients, limited to a small subset of coefficients and widths
Peaks & extrema	Number of peaks (standard and CWT-based), longest strike above/below mean, location of last minimum, index mass quantile
Entropy & complexity	Fourier entropy, Lempel-Ziv complexity, CID (complexity-invariant distance)
Nonlinear dynamics	Friedrich coefficients, max Langevin fixed point
Trend & regression	Linear trend p -value; aggregated linear trend slope (chunk-length 5, variance aggregation)
Autoregressive model	Autoregressive (AR) model coefficients (order 10, coefficients 6–8)
Stationarity test	Augmented Dickey-Fuller test statistic and number of lags used
Crossings & thresholds	Number of mean crossings, count above mean, ratio beyond $r\sigma$
Energy distribution	Energy ratio by signal chunk (segment-wise energy fraction)

2.3.4 Feature Cleaning and Selection

In machine learning, the process of selecting, modifying, or removing variables in a dataset in order to improve the performance and reliability of downstream models is known as *feature engineering*. Feature selection, in particular, involves removing variables that are redundant, uninformative, or otherwise unsuitable for the learning task at hand: input data may contain many variables, some of which do not improve predictive performance and only add unnecessary complexity to the model [38]. In this thesis, feature engineering is applied

exclusively as a feature *selection* step, since the input variables themselves, the tsFresh-derived signal descriptors described in the previous section, are already generated in a comprehensive, automated manner; the goal here is therefore to identify and remove those features that are invalid, non-discriminative, or redundant with others, rather than to construct new variables.

The raw feature set produced by tsFresh, while comprehensive, contains a substantial number of features that are uninformative, ill-defined, or redundant for the classification task at hand. Before use in classification, this raw set is cleaned and reduced through the sequence of filtering steps summarised in Table 2.3. To avoid data leakage, this cleaning pipeline is not applied once to the full dataset; instead, it is fit independently on the training folds of each outer cross-validation iteration (Section 2.3.5) and then applied, unchanged, to that iteration’s test fold. As a result, the exact number of features removed at each step varies slightly across the 5 outer folds; the counts reported below are the average across folds.

Table 2.3: Feature cleaning and selection steps applied to the raw tsFresh feature set (averaged across the 5 outer folds).

Filtering step	Features removed	Criterion
Invalid values (NaN/Inf)	1	Feature entirely undefined (NaN or Inf for every signal)
Zero-variance features	52	Feature takes a constant value across all training signals
Highly correlated features	529	Pairwise correlation ≥ 0.85 with another retained feature
Total removed	582	

Three filtering steps are applied, in order, to the training data of each outer fold. First, values that are Inf are converted to NaN; a feature that is NaN for every signal in the training data (arising when a tsFresh calculator is undefined for that entire fold, e.g. statistics requiring a minimum signal length, or divisions by zero) is removed entirely, since imputing it would introduce a value with no information content. Features that are only partially undefined are instead imputed with the median of the training data, preserving them rather than discarding potentially useful information. Second, zero-variance features, which take a constant value across all training signals, are discarded, as they carry no discriminative information. Third, since tsFresh’s calculators are not mutually independent and several tend to produce highly

correlated outputs, one feature from each pair with a pairwise correlation above 0.85 is removed, reducing redundancy without discarding genuinely distinct information. The fitted imputer, variance filter, and correlation pruner are then applied, unchanged, to the corresponding test fold.

Applying this sequence of cleaning steps reduces the initial 777 tsFresh features to a final set of, on average, 195 features per fold (195 across all folds), used as input to the classification models described in the following sections.

2.3.5 Group-Based Nested Cross-Validation

Each physical target was insonified from multiple aspect angles, producing several highly correlated signals per target. Because of this, a standard k-fold cross-validation is not suitable here: splitting samples randomly across folds could place signals from the same target in both the training and test sets, letting the model implicitly learn target-specific characteristics rather than generalizable mine/clutter features, leading to overly optimistic, unrealistic performance estimates.

To prevent this, `GroupKFold` cross-validation was used, with physical target identity as the grouping variable. All signals from a given target are assigned entirely to either the training set or the test set, never split between the two.

The evaluation uses two levels of cross-validation, both grouped by target (Figure 2.11). In the outer loop, the dataset is split into 5 folds: in each iteration, 4 folds (4/5) serve as training data and 1 fold (1/5) is held out as the test fold, repeated 5 times so that every target is used as test data exactly once. Inside each outer iteration, the training data (4/5) is further split into 3 inner folds to search for the best hyperparameters with Optuna, using a number of trials scaled to the dimensionality of each classifier’s search space (detailed per classifier in Chapter 3); this search never touches the outer test fold.

Within each outer iteration, features are cleaned (imputation, zero-variance removal, correlation pruning at 0.85, scaling), fit on the training folds only and applied to the test fold by transform.

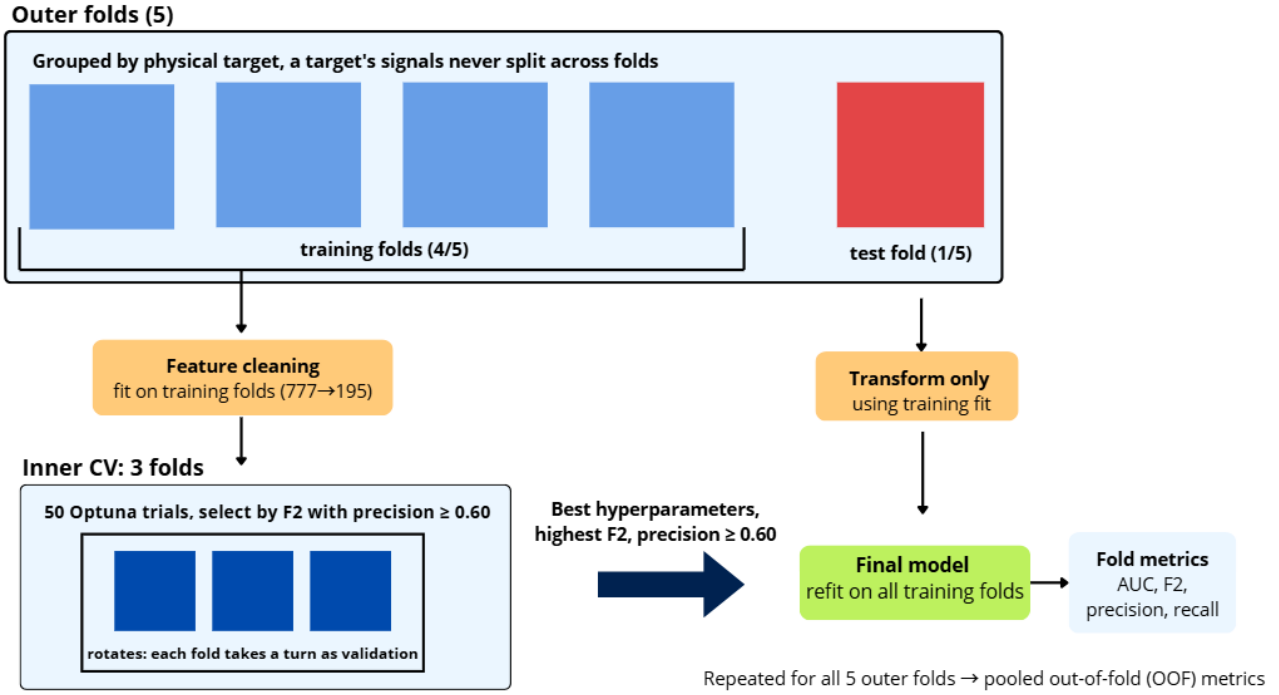
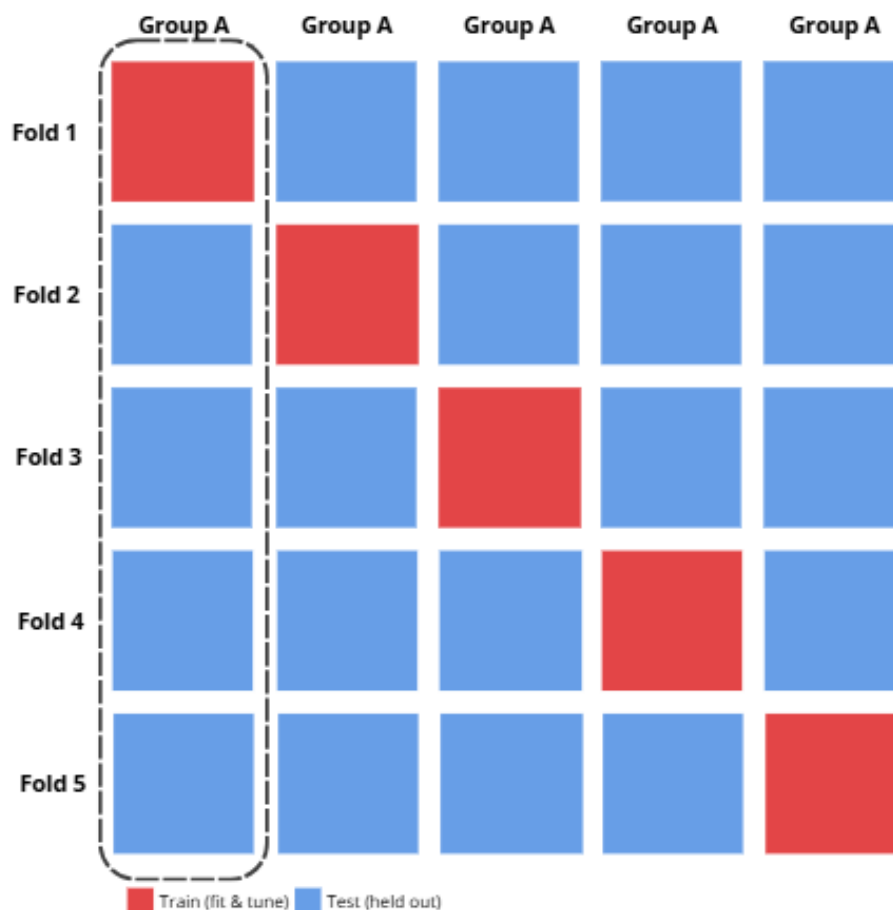


Figure 2.11: One outer-fold iteration: feature cleaning, inner 3-fold hyperparameter search, and final model evaluation on the held-out test fold.

The inner loop selects the best hyperparameters using the F_2 score with a precision floor of 0.60. F_2 was chosen over a threshold-independent metric such as AUC, or a balanced metric such as F_1 , because it weights recall roughly four times more than precision, reflecting the operational cost of this task: missing a real mine (a false negative) is far more costly than raising a false alarm on seabed clutter. However, on this near-balanced dataset ($\sim 51\text{--}52\%$ positive), a degenerate model that predicts the positive class for almost every sample still achieves a precision close to the base rate, so its inflated recall can still produce a competitive F_2 score. To rule out this failure mode, a precision floor of 0.60 is enforced: only hyperparameter configurations reaching at least this precision are considered, and the highest- F_2 configuration among them is selected; if no configuration clears the floor in a given fold, the highest-precision configuration is selected instead. A final model is then refit on the full training data (4/5) with those hyperparameters and evaluated once on the test fold.

Each outer iteration trains a completely separate, independent model. As illustrated in Figure 2.12 for one target group, a given target is used as test data in exactly one outer iteration (by a model that never trained on it) and as training data in the other four (for models that are never used to test it).



**Group A is tested only in Fold 1, by a model that never trained on it.
It is trained on in Folds 2-5, but never tested there.**

Figure 2.12: Train/test role of one target group across the 5 outer folds. Leakage is judged within a single fold's model, not across folds.

Concretely, each outer iteration writes its test-fold predictions into a single array indexed by signal, so that after all five iterations every signal holds exactly one predicted probability, produced by the one model that never trained on it (Figure 2.13). All metrics reported for a given classifier, per-fold values aside, are computed once on this pooled array rather than on any individual fold. predicted probabilities are thresholded at 0.5 to obtain the confusion matrix category for each signal, true positive, true negative, false positive, or false negative, from which precision, recall, and F_2 follow; AUC and the ROC curve are instead computed directly from the probabilities by scanning across all possible thresholds, rather than a single fixed one.

Folds as test	Fold 1					...	Fold 5	
Signal	0	1	2	...	238	...	1157	1158
True label	1	0	0	...	1	...	0	1
Predicted proba	0.45	0.71	0.08	...	0.93	...	0.22	0.82
Decision (≥ 0.5)	0	1	0	...	1	...	0	1
Category	FN	FP	TN	...	TP	...	TN	TP

TP, TN = correct (green); FP, FN = incorrect (orange/rouge)

Figure 2.13: Worked example: for each signal, the true label and the predicted probability from its own fold’s model are compared at the 0.5 threshold to obtain its confusion matrix category.

Reporting the fold with the best individual score would instead be a form of cherry-picking, since any classifier is likely to have at least one favorably split fold among five just by chance; the pooled out-of-fold metrics use every signal exactly once and therefore give the most representative estimate of performance on unseen targets.

With only 27 targets, a single fixed train/test split would depend heavily on which targets happened to fall into the test set, giving an unreliable estimate. Rotating the test fold across all targets instead yields an out-of-fold prediction for every signal, each produced by a model that never saw its target during training. Pooling these out-of-fold predictions gives a much more stable performance estimate than any single split, and the per-fold spread (mean $\pm$ std) additionally shows how sensitive the model is to which targets it is trained or tested on.

2.4 Machine Learning Classifiers and Evaluation Protocol

After the feature extraction and preprocessing steps, the resulting feature set was used to train several machine learning models for the binary classification of Mine and Clutter signals. Six classifiers were considered: Logistic Regression, Support Vector Machine (SVM), Random Forest, LightGBM, XGBoost, and H2O AutoML. These models were selected to represent different machine learning approaches, including linear, kernel-based, and tree-based methods. The following subsections briefly describe the principle behind each model, the role of its main

hyperparameters, and how it is used for the classification task; the corresponding search spaces and the configurations selected during cross-validation are reported in Chapter 3.

2.4.1 Logistic Regression

Logistic Regression is a widely used baseline method for binary classification because of its simplicity, interpretability, and computational efficiency. In this thesis, it was selected as the simplest classifier in the benchmark, providing a linear reference against which the more complex kernel-based and tree-based methods can be compared. The formulation presented below follows the treatment of Logistic Regression in Chapter 4 of Hastie et al.’s *The Elements of Statistical Learning* [39].

For a sample represented by a feature vector $\mathbf{x} \in \mathbb{R}^p$ ($p = 195$ in this study), Logistic Regression first computes a weighted linear combination of the input features,

$$z = \mathbf{w}^\top \mathbf{x} + b, \quad (2.18)$$

where $\mathbf{w} \in \mathbb{R}^p$ denotes the feature-weight vector and b is the bias term. The linear score z is then transformed into a probability using the sigmoid function:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}. \quad (2.19)$$

The resulting probability represents the likelihood that the sample belongs to the positive class, defined as *Mine* in this study. A fixed decision threshold of 0.5 is used to obtain the final prediction: a sample is classified as a mine when $P(y = 1 \mid \mathbf{x}) \geq 0.5$, and as non-mine otherwise. Since the probability is obtained from a linear combination of the input features, the corresponding decision boundary is a linear hyperplane in the feature space.

The model parameters are estimated by minimizing the binary cross-entropy, or log-loss,

$$\mathcal{L}(\mathbf{w}, b) = - \sum_{i=1}^n [y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i)], \quad (2.20)$$

where $\hat{p}_i = \sigma(\mathbf{w}^\top \mathbf{x}_i + b)$ is the predicted probability for sample i . To limit the magnitude of the model coefficients and reduce overfitting, a regularization penalty can be added to the loss

function. For the ℓ_2 penalty, for example, the objective becomes

$$\mathcal{L}_{\text{reg}}(\mathbf{w}, b) = \mathcal{L}(\mathbf{w}, b) + \frac{1}{2C} \|\mathbf{w}\|_2^2, \quad (2.21)$$

where C controls the regularization strength inversely: smaller values of C impose stronger regularization, whereas larger values allow the model to fit the training data more freely.

Two regularization penalties were considered in the experiments. The ℓ_1 penalty,

$$\|\mathbf{w}\|_1 = \sum_j |w_j|,$$

encourages sparse coefficient vectors by driving some weights to exactly zero. This property can implicitly perform feature selection. In contrast, the ℓ_2 penalty,

$$\|\mathbf{w}\|_2^2 = \sum_j w_j^2,$$

shrinks the coefficients toward zero without generally removing features completely.

Logistic Regression therefore serves as a direct linear baseline in the benchmark. Its performance indicates how much of the classification problem can be explained by a linear relationship between the extracted features and the target class, providing a reference for evaluating the additional flexibility offered by the kernel-based and tree-based classifiers.

2.4.2 Support Vector Machine (RBF Kernel)

Support Vector Machines (SVMs) are supervised learning methods commonly used for binary classification. Given a set of labeled training samples, an SVM looks for the decision boundary that separates the two classes while staying as far as possible from the closest training points on either side, a property known as maximizing the *margin*. This formulation is described in Chapter 12 of Hastie et al.'s *The Elements of Statistical Learning* [39].

For a sample represented by a feature vector $\mathbf{x} \in \mathbb{R}^p$, a linear SVM defines its decision boundary as the hyperplane

$$\mathbf{w}^\top \mathbf{x} + b = 0, \quad (2.22)$$

where $\mathbf{w}$ is the weight vector and b is the bias term. A wider margin around this boundary generally yields a more robust classifier, since the decision is kept as far as possible from the

training samples of both classes, leaving more room for error on unseen data.

In practice, the two classes are rarely perfectly separable. SVMs therefore use a *soft-margin* formulation, which tolerates some samples falling on the wrong side of the margin by introducing a slack variable $\xi_i \geq 0$ for each sample, measuring how far it violates its margin. This leads to the optimization problem

$$\min_{\mathbf{w}, b, \xi} \quad \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^n \xi_i, \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad (2.23)$$

where $y_i \in \{-1, +1\}$ is the class label of sample i . The hyperparameter C controls the trade-off between these two goals: a large C penalizes margin violations heavily, pushing the classifier to fit the training samples more strictly, while a small C tolerates more violations in exchange for a wider, more general margin.

Once solved, only a subset of the training samples — those lying on or inside the margin, called *support vectors* — end up influencing the position of the decision boundary; all other samples can be ignored once training is complete.

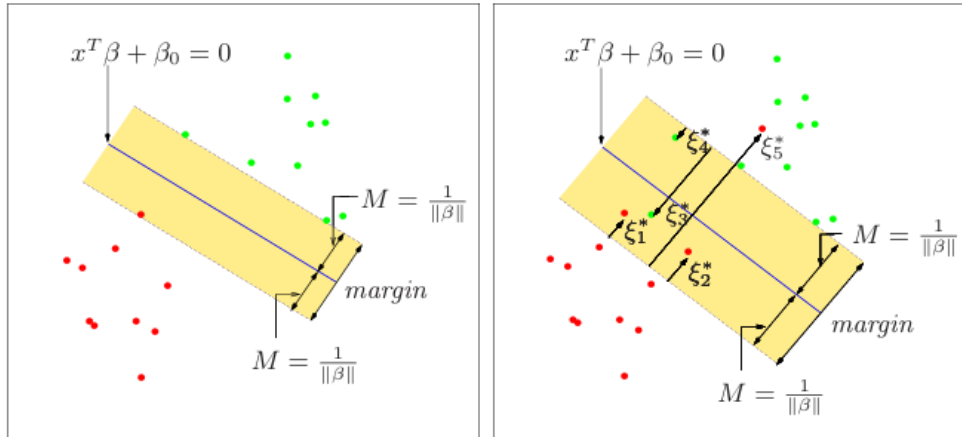


Figure 2.14: Support vector classifiers: the separable case (left), where the decision boundary maximises the margin $M = 1/\|\beta\|$; and the non-separable case (right), where points violating the margin are penalised through slack variables. Adapted from [39], Figure 12.1.

The linear formulation above assumes the two classes can be reasonably separated by a straight hyperplane in the original feature space. When this is not the case, SVMs can instead learn a nonlinear boundary through the *kernel trick*: rather than explicitly mapping the data into a higher-dimensional space where it would become separable, a kernel function directly computes the similarity between pairs of samples in that space. The decision function can then

be written purely in terms of these pairwise similarities,

$$f(\mathbf{x}) = \sum_{i \in \mathcal{S}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b, \quad (2.24)$$

where $\mathcal{S}$ is the set of support vectors, α_i their associated coefficients, and $K(\mathbf{x}_i, \mathbf{x})$ the kernel function; a new sample is classified according to the sign of $f(\mathbf{x})$.

Several kernel functions exist (linear, polynomial, sigmoid, and radial basis function), and this work adopts the *Radial Basis Function (RBF) kernel*, a common default choice for nonlinear SVM classification:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2), \quad (2.25)$$

where $\gamma > 0$ controls how far the influence of each support vector reaches: a large γ restricts this influence to nearby points, producing a more flexible but overfitting-prone boundary, while a small γ produces a smoother, more global boundary.

The RBF kernel therefore allows the SVM to model nonlinear relationships between the extracted features and the target class, while keeping the same margin-based learning principle as the linear formulation. In addition to C and γ , class-balanced weighting which reweights the margin-violation penalty for each class to compensate for any imbalance between Mine and non-Mine samples was also tuned as a hyperparameter. The search space and per-fold selected values for all three are given in Chapter 3.

2.4.3 Random Forest

Random Forest is an ensemble learning method that builds a large collection of decision trees and combines their predictions, reducing the variance and overfitting risk of any single tree **breiman2001random**. Each individual decision tree recursively partitions the feature space into regions by selecting, at each node, the feature and threshold that best separate the classes, typically by minimizing an impurity measure such as the Gini index:

$$G(t) = 1 - \sum_{c \in \{\text{Mine}, \text{Clutter}\}} p_c(t)^2 \quad (2.26)$$

where $p_c(t)$ is the proportion of samples belonging to class c at node t . A split is chosen to minimize the weighted impurity of the resulting child nodes, and this process is repeated recursively until a stopping criterion is reached (e.g., minimum samples per leaf, or maximum tree depth).

A single decision tree, grown to full depth, tends to overfit the training data. Random Forest addresses this through two sources of randomness introduced when building each tree $b = 1, \dots, B$:

- **Bootstrap aggregating (bagging):** each tree is trained on a bootstrap sample, obtained by sampling n training examples with replacement from the original training set.
- **Random feature selection:** at each split, only a random subset of $m < p$ features (with $p = 195$ in this study) is considered, rather than all available features. This decorrelates the individual trees, since strong predictors are prevented from dominating every split.

Once all B trees are grown, the forest's prediction for a new sample $\mathbf{x}$ is obtained by aggregating the individual tree predictions, typically via majority voting for classification:

$$\hat{y}(\mathbf{x}) = \text{majority vote } \{T_b(\mathbf{x})\}_{b=1}^B \quad (2.27)$$

where $T_b(\mathbf{x})$ denotes the class predicted by tree b . Averaging over many decorrelated trees reduces the variance of the ensemble relative to a single tree, while keeping bias low, since each individual tree remains largely unpruned and flexible [39].

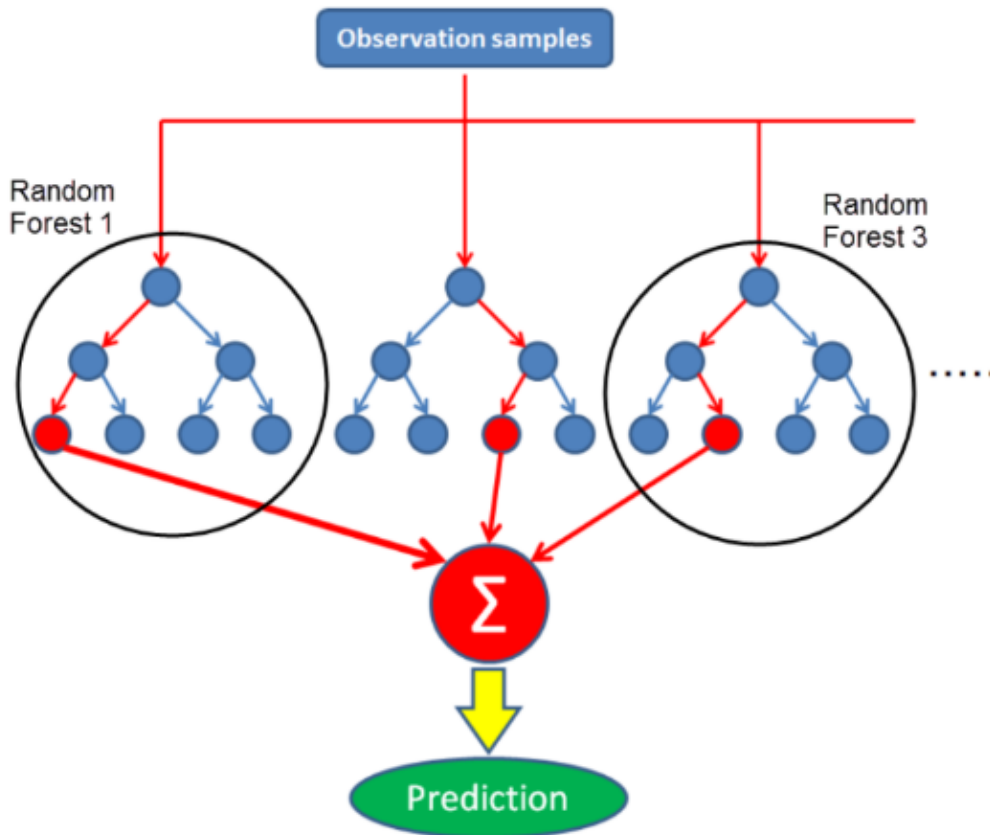


Figure 2.15: Random Forest ensemble structure.

The main hyperparameters of Random Forest control the size of the ensemble, the degree of randomness, and the complexity of the individual trees. The parameter `n_estimators` determines the number of trees B in the forest. Increasing the number of trees generally improves the stability of the predictions by reducing variance, although the improvement eventually becomes marginal while computational cost continues to increase.

The parameter `max_features` controls the number of features considered at each split. A smaller subset increases the diversity among trees, whereas considering a larger number of features can make individual trees more similar. The parameters `max_depth`, `min_samples_split`, and `min_samples_leaf` control the complexity of each tree. Limiting the tree depth or requiring more samples for splits and leaves generally produces simpler trees and can reduce overfitting.

Finally, `class_weight` can be used to assign different importance to the classes when the training data are imbalanced. This allows the learning process to account for differences in class frequency rather than treating every training sample equally.

2.4.4 Gradient Boosting: XGBoost and LightGBM

Gradient Boosting is an ensemble learning framework in which weak learners, typically decision trees, are constructed sequentially. Unlike Random Forest, where the individual trees are trained independently, each new tree in Gradient Boosting is trained to improve the predictions of the previously constructed ensemble. The general framework was introduced by Friedman as a stage-wise optimization procedure based on the gradient of a chosen loss function [40].

The prediction of a Gradient Boosting model is built progressively according to

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \eta h_m(\mathbf{x}), \quad (2.28)$$

where $h_m(\mathbf{x})$ denotes the learner added at iteration m , and η is the learning rate controlling its contribution to the ensemble. A smaller learning rate results in smaller updates and generally requires a larger number of boosting iterations, allowing the model to build its prediction more gradually.

At each iteration, the new learner is trained to reduce the loss of the current model. This is achieved by using the negative gradient of the loss function with respect to the current prediction:

$$r_{im} = -\frac{\partial \mathcal{L}(y_i, F_{m-1}(\mathbf{x}_i))}{\partial F_{m-1}(\mathbf{x}_i)}. \quad (2.29)$$

The new learner is fitted to these gradient-based values and then added to the existing ensemble. For binary classification, a classification loss such as the logarithmic loss can be used [40].

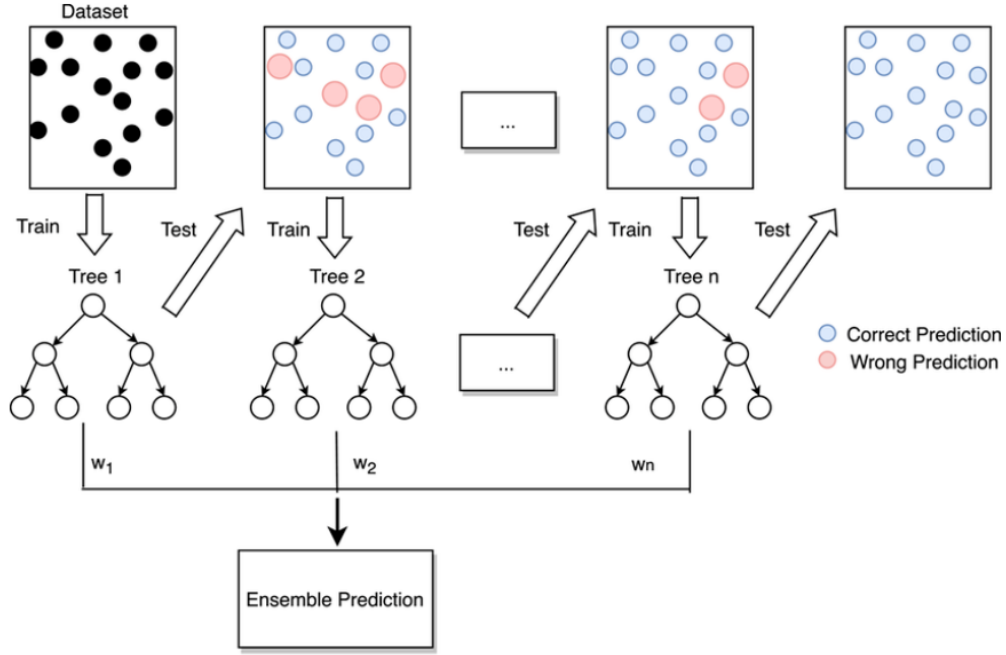


Figure 2.16: Representation of a Gradient Boosting classification process [41].

XGBoost and LightGBM are optimized implementations of the Gradient Boosting framework that use decision trees as their base learners. Although both follow the same sequential boosting principle, they introduce different strategies for tree construction, regularization, and computational efficiency. The following subsection focuses on XGBoost, while LightGBM is discussed separately.

2.4.4.1 XGBoost

XGBoost (*Extreme Gradient Boosting*) extends the conventional Gradient Boosting framework by incorporating explicit regularization and second-order optimization to improve both predictive performance and control overfitting [42].

At boosting iteration m , XGBoost adds a new tree h_m to the existing model. Its objective can be expressed as

$$\text{Obj}^{(m)} = \sum_{i=1}^n \mathcal{L}(y_i, F_{m-1}(\mathbf{x}_i) + h_m(\mathbf{x}_i)) + \Omega(h_m), \quad (2.30)$$

where $\mathcal{L}$ denotes the training loss and $\Omega(h_m)$ is a regularization term controlling the complexity of the new tree.

For a tree containing T leaves, the regularization term is given by

$$\Omega(h) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2, \quad (2.31)$$

where w_j is the weight assigned to leaf j , while γ and λ control the complexity of the tree and the magnitude of its leaf weights, respectively [42]. This explicit regularization distinguishes XGBoost from the basic Gradient Boosting formulation and helps limit unnecessary tree complexity.

XGBoost also uses a second-order Taylor approximation of the loss function. Instead of considering only the first-order gradient, it incorporates both the gradient and the second derivative of the loss when determining the contribution of the new tree. This provides a more informative approximation of the objective and allows the tree structure and leaf weights to be optimized efficiently [42].

Several XGBoost hyperparameters control the balance between model flexibility, regularization, and computational cost. The number of boosting trees, `n_estimators`, determines how many sequential learners are added to the ensemble, while `learning_rate` controls the contribution of each tree. The parameter `max_depth` limits the depth of individual trees, thereby controlling their complexity. The parameter `min_child_weight` imposes a minimum amount of instance weight required for further splitting, providing an additional mechanism for controlling tree growth.

XGBoost can also introduce randomness through `subsample`, which controls the fraction of training samples used to construct each tree, and `colsample_bytree`, which controls the fraction of features considered for each tree. These parameters can reduce correlations between successive learners and help limit overfitting.

Finally, `reg_alpha` and `reg_lambda` control ℓ_1 and ℓ_2 regularization, respectively, while `scale_pos_weight` can be used to compensate for class imbalance by assigning greater weight to the positive class when necessary.

2.4.4.2 LightGBM

LightGBM is another tree-based gradient boosting framework designed to improve training efficiency, particularly for large-scale and high-dimensional datasets [43]. One of its main characteristics is histogram-based learning, in which continuous feature values are grouped

into discrete bins before searching for the best tree splits. This reduces the computational cost and memory requirements during training.

LightGBM also introduces two important techniques to improve the efficiency of gradient boosting. The first is *Gradient-based One-Side Sampling (GOSS)*, which retains observations with large gradient values while sampling a portion of observations with smaller gradients. This allows the algorithm to reduce the number of training samples considered during split finding while preserving observations that are more informative for the current model. The second is *leaf-wise tree growth*. Instead of growing the tree level by level, LightGBM selects the leaf that provides the largest reduction in the loss and splits it. This strategy can achieve a lower loss with a given number of leaves, although it may increase the risk of overfitting when the dataset is relatively small [43].

For both XGBoost and LightGBM, hyperparameters such as the learning rate, number of trees, and parameters controlling tree complexity can be adjusted during model training. In this work, these hyperparameters were optimized using Optuna based on cross-validated performance.

2.4.5 H2O AutoML: Automatic machine learning

H2O AutoML is an automated machine learning framework that trains and tunes a diverse set of models with minimal manual intervention, then combines the best-performing ones into a final ensemble [44]. Rather than relying on a single algorithm and hand-tuned hyperparameters, AutoML systematically searches across multiple model families, including Generalized Linear Models, Random Forests, Gradient Boosting Machines (GBM), XGBoost, and deep neural networks training each with a range of hyperparameter configurations under a fixed time or model-count budget.

Once the individual models have been trained, H2O AutoML constructs a *Stacked Ensemble*, following the stacking methodology introduced by Wolpert [45]. Stacking combines the predictions of several *base learners* $h_1, h_2, \dots, h_K$ (here, the individual models trained during the AutoML search) using a second, higher-level model, called the *meta-learner*, which learns how to best combine their outputs rather than combining them by a fixed rule such as averaging or majority vote:

$$\hat{y}(\mathbf{x}) = g(h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_K(\mathbf{x})) \quad (2.32)$$

where $g(\cdot)$ is the meta-learner, typically a regularized linear model (e.g., logistic regression with non-negative weight constraints), trained on the out-of-fold predictions of the base learners obtained via cross-validation. Using out-of-fold predictions, rather than each base learner’s predictions on its own training data, is essential to prevent the meta-learner from overfitting to base learners that memorized their training set rather than generalizing.

In this study, all base learners and the Stacked Ensemble were evaluated under the same GroupKFold cross-validation used for the other classifiers, ensuring that the out-of-fold predictions used to train the meta-learner respect the target-identity grouping and do not leak information between folds.

2.4.6 Hyperparameter Optimization with Optuna

Model performance depends strongly on hyperparameter choice. For the Logistic Regression, SVM, Random Forest, LightGBM, and XGBoost classifiers, hyperparameters were tuned using Optuna [46], an optimization framework that searches sequentially rather than exhaustively (grid search) or randomly (random search). Its default *Tree-structured Parzen Estimator (TPE)* sampler uses the results of previous trials to guide the next configuration, progressively focusing the search on promising regions of the hyperparameter space [46].

Each trial evaluates one candidate configuration $\boldsymbol{\theta}$ using the same inner GroupKFold procedure described in Section 2.3.5, preventing data leakage from influencing the selected hyperparameters. Rather than a threshold-independent metric such as AUC, the objective maximized is the F_2 -score at a fixed decision threshold of 0.5, which weights recall more heavily than precision, reflecting the higher cost of missing a true mine relative to a false alarm. Since maximizing F_2 alone can favor degenerate, over-predicting classifiers on a near-balanced dataset, only trials whose mean inner-fold precision exceeds a fixed floor (0.60) are eligible for selection:

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta} \in \Theta_{\text{qualified}}} F_2(\boldsymbol{\theta}), \quad \Theta_{\text{qualified}} = \{\boldsymbol{\theta} \in \Theta \mid \text{precision}(\boldsymbol{\theta}) \geq 0.60\}. \quad (2.33)$$

If no trial clears the floor, the highest-precision trial is selected instead. AUC and AP are still reported in Chapter 3 as discriminative summaries, but were not used as the optimization

objective. Trial counts were scaled to each model’s search-space dimensionality and are detailed in Chapter 3.

2.4.7 Evaluation Metrics and Confidence Intervals

Model performance is evaluated using metrics derived from the confusion matrix, a standard tool for summarizing the outcomes of a binary classifier. In this work, the positive class corresponds to MILCO (mine-like object), and the negative class to non-MILCO. Given a decision threshold applied to the predicted probability score, each prediction falls into one of four categories:

- **True Positive (TP)**: a MILCO signal correctly classified as MILCO,
- **False Negative (FN)**: a MILCO signal misclassified as non-MILCO,
- **False Positive (FP)**: a non-MILCO signal misclassified as MILCO,
- **True Negative (TN)**: a non-MILCO signal correctly classified as non-MILCO.

These four quantities form the basis of the metrics used throughout this chapter.

◆ Confusion Matrix

is a 2x2 table that cross-tabulates the predicted class against the true class, allowing all classification outcomes to be inspected at once rather than being collapsed into a single score. Given the decision threshold, each prediction falls into one of four categories: a *true positive (TP)* is a MILCO signal correctly classified as MILCO, a *false negative (FN)* is a MILCO signal misclassified as non-MILCO, a *false positive (FP)* is a non-MILCO signal misclassified as MILCO, and a *true negative (TN)* is a non-MILCO signal correctly classified as non-MILCO. These four quantities form the basis of the metrics defined below.

◆ Recall

quantifies how many of the actual MILCO signals are successfully detected:

$$\text{Recall} = \frac{TP}{TP + FN}. \quad (2.34)$$

A low recall means that mine-like objects are being missed, which is the most critical type of error in an MCM context.

◆ Precision

quantifies how many of the signals flagged as MILCO are actually MILCO:

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (2.35)$$

Low precision corresponds to a high false alarm rate.

◆ F₂-score

is a weighted harmonic mean of precision and recall that places more emphasis on recall than on precision, controlled by the parameter β :

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}}. \quad (2.36)$$

Setting $\beta = 2$ gives the F₂-score:

$$F_2 = 5 \cdot \frac{\text{Precision} \cdot \text{Recall}}{(4 \cdot \text{Precision}) + \text{Recall}}, \quad (2.37)$$

which weights recall approximately four times more heavily than precision. This reflects the operational cost asymmetry of the task at hand: missing a real mine (a false negative) is far more costly than raising a false alarm on seabed clutter (a false positive). The F₂-score was therefore used, in place of a symmetric metric such as accuracy or AUC, as the objective for hyperparameter selection during cross-validation (Section 2.3.5).

◆ Accuracy

gives the overall fraction of correctly classified signals:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \quad (2.38)$$

Because the dataset is not perfectly balanced between classes, accuracy alone can be misleading and is reported only as a complementary indicator.

◆ ROC-AUC

evaluates the classifier's ability to rank MILCO signals above non-MILCO signals across all possible thresholds, independent of any single operating point. A value of 1.0 indicates perfect separation between the two classes, 0.5 corresponds to random guessing, and values

below 0.5 indicate systematically inverted predictions.

◆ Average Precision (AP)

summarizes the precision–recall trade-off as a single value by computing the area under the Precision–Recall curve, and is particularly informative when the positive class is less frequent than the negative class.

Among these metrics, **recall is treated as the primary criterion** for model selection in this work. This choice reflects the asymmetric cost of errors in mine countermeasures: failing to detect an actual mine (a false negative) carries far more severe consequences than flagging a harmless object as suspicious (a false positive), which only results in an additional, avoidable inspection.

◆ Bootstrap Confidence Intervals

Because the dataset used in this work is relatively small, a single point estimate of a performance metric can be sensitive to which signals happen to fall into the test folds. To quantify this uncertainty, confidence intervals for the reported metrics are computed using the bootstrap method, a resampling technique that estimates the variability of a statistic without requiring assumptions about the underlying data distribution.

For a given classifier, the out-of-fold predictions obtained across the GroupKFold procedure are pooled and resampled with replacement 1,000 times. For each of the 1,000 bootstrap samples, the metric of interest (e.g., recall, precision, AUC) is recomputed, producing an empirical distribution of the metric. The 95% confidence interval is then obtained by taking the 2.5th and 97.5th percentiles of this distribution:

$$\text{CI}_{95\%} = [Q_{2.5}(\hat{m}^*), Q_{97.5}(\hat{m}^*)], \quad (2.39)$$

where $\hat{m}^*$ denotes the bootstrap distribution of the metric m .

Reporting confidence intervals alongside point estimates allows the performance of different classifiers to be compared while accounting for the uncertainty introduced by the limited size of the dataset, rather than relying on single-value comparisons that may not be statistically meaningful.

Conclusion

The methodology presented in this chapter establishes the complete processing chain used to prepare the data for target classification. The acquired sonar signals were processed and transformed into focused target responses, from which relevant time-series features were extracted and prepared for machine learning. The resulting dataset was then used as input for the different classification models introduced in this chapter. Their performance will be evaluated and compared in the following chapter.

RESULTS AND INTERPRETATION

Plan

3.1	Model Results	65
3.2	Target-Level Difficulty Analysis	82
3.3	Model Interpretability	85

Introduction

This chapter presents and analyzes the results obtained from the classification pipeline described in Chapter 2. All results reported here were obtained using the GroupKFold cross-validation procedure ($k = 5$), which groups folds by physical target identity to prevent data leakage. While the AUC is reported throughout as a threshold-independent measure of overall discriminative performance, recall is treated as the primary evaluation metric, since in a mine countermeasures (MCM) context missing an actual mine (false negative) is far more costly than misclassifying a piece of clutter as a mine (false positive). Unless stated otherwise, all classifiers were trained on the same set of 195 tsFresh features obtained after cleaning and correlation-based pruning. The baseline performance of the individual classifiers is first reported, followed by the effect of hyperparameter optimization and the results obtained with H2O AutoML. Finally, the interpretability of the models is addressed using feature importance, SHAP, and LIME, in order to relate the statistical behavior of the classifiers to the physical characteristics of the LF-SAS signals.

3.1 Model Results

All seven classifiers were evaluated using the nested, target-grouped cross-validation procedure described in Section 2.3.5. For each classifier, the results below report the selected hyperparameters, per-fold performance, ROC analysis, and pooled out-of-fold performance. Unless otherwise stated, the reported classification metrics use a decision threshold of 0.5.

3.1.1 Logistic Regression

◆ Hyperparameters:

Hyperparameters were searched with 50 Optuna trials per outer fold, given the relatively small dimensionality of the Logistic Regression search space. Table 3.1 shows the search space used, with the `liblinear` solver, chosen because it supports both ℓ_1 and ℓ_2 penalties and is well suited to a dataset of this size.

Table 3.1: Optuna search space for the Logistic Regression classifier

Hyperparameter	Range / choices
C	$[10^{-4}, 10^2]$ (log-uniform)
penalty	$\{\ell_1, \ell_2\}$
class_weight	{balanced, None}

◆ **Per-fold results:**

Performance varies substantially between folds, with Fold 3 achieving the strongest performance ($\text{AUC} = 0.888$), while folds 1 and 2 remain close to chance level ($\text{AUC} \approx 0.5$), suggesting the targets held out in those two folds are considerably harder to separate from non-mine clutter given the available features. Table 3.2 reports outer-fold performance.

Table 3.2: Logistic Regression per-fold performance

Fold	AUC	AP	F_2	Precision	Recall
1	0.481	0.498	0.522	0.512	0.524
2	0.520	0.624	0.451	0.520	0.437
3	0.888	0.881	0.828	0.786	0.839
4	0.755	0.713	0.657	0.691	0.650
5	0.633	0.686	0.772	0.615	0.825
Mean $\pm$ SD	0.655 ± 0.168	0.680 ± 0.139	0.646 ± 0.160	0.625 ± 0.116	0.655 ± 0.179

◆ **ROC curve:**

Figure 3.1 shows the ROC curve computed on the pooled out-of-fold predictions.

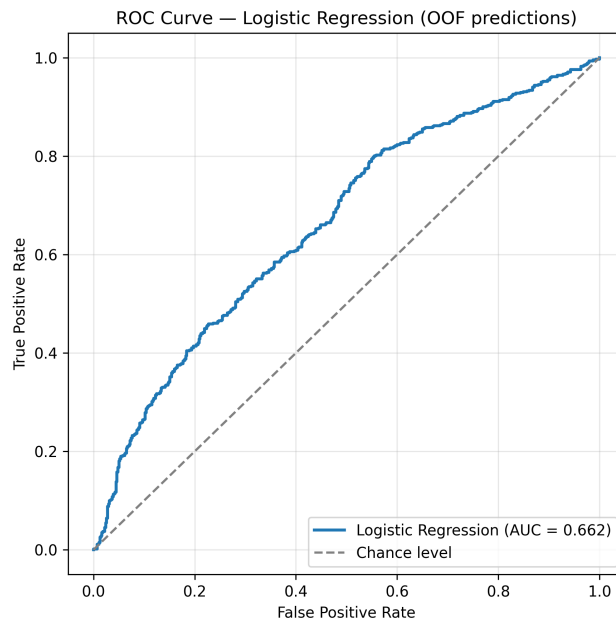


Figure 3.1: Logistic Regression ROC curve (out-of-fold predictions).

The curve remains consistently above the chance diagonal across the full range of thresholds, with an out-of-fold AUC of 0.662, confirming that the model carries real discriminative signal beyond a random guess, although the moderate separation from the diagonal reflects the limited performance already visible in the per-fold results.

◆ Out-of-fold aggregate performance

Pooling out-of-fold predictions across all five outer folds yields $\text{AUC} = 0.662$, $\text{AP} = 0.625$, and $F_2 = 0.653$, with precision 0.626 and recall 0.660 at the same 0.5 threshold. Figure 3.2 shows the corresponding confusion matrix.

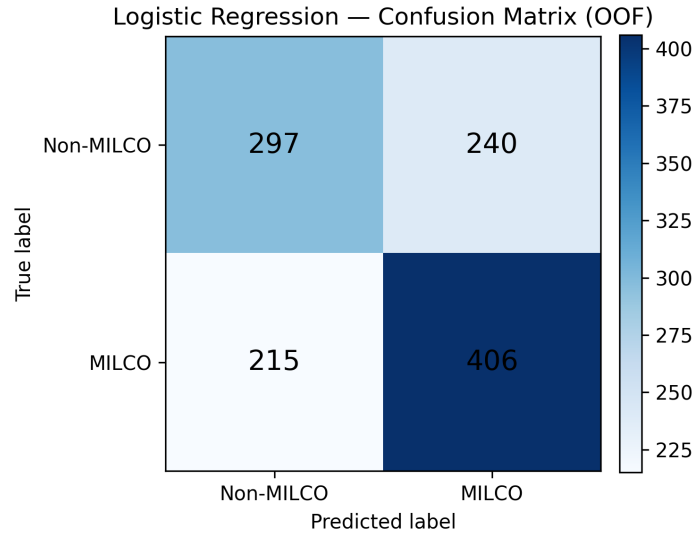


Figure 3.2: Logistic Regression confusion matrix (out-of-fold predictions).

Of the 621 true MILCO samples, the classifier correctly identifies 406 (recall ≈ 0.654); among the 646 samples predicted as MILCO, 406 are correct (precision ≈ 0.629). These pooled values are close to, but slightly below, the mean per-fold precision and recall in Table 3.2, reflecting the fact that the pooled metrics weight every signal equally, whereas the per-fold mean weights each outer fold equally regardless of its size; the overall picture is consistent with a modest linear baseline that captures some, but not most, of the separation between the two classes.

3.1.2 Support Vector Machine (RBF Kernel)

◆ Hyperparameters:

The SVM classifier uses an RBF kernel to capture nonlinear relationships between the extracted features and the target classes. The two main parameters controlling the model are the regularization parameter C and the kernel coefficient γ . Because their suitable values can vary over several orders of magnitude, both were explored on logarithmic scales. Hyperparameters were searched with the same number of Optuna trials per outer fold as the Logistic Regression model. The complete search space is given in Table 3.3

◆ Per-fold results:

The outer-fold results in Table 3.4 show a marked variation in SVM performance across the target groups. Table 3.4 reports outer-fold performance. The strongest results are obtained

Table 3.3: Optuna search space for the SVM (RBF) classifier

Hyperparameter	Range / choices
C	$[10^{-3}, 10^2]$ (log-uniform)
γ	$[10^{-4}, 10^1]$ (log-uniform)
<code>class_weight</code>	{balanced, None}

in folds 3 and 4, with AUC values of 0.902 and 0.937, respectively. Fold 4 is particularly notable, reaching a recall of 0.949 and an F_2 score of 0.916. Fold 5 also achieves high recall (0.902), although its AUC is considerably lower (0.634) and its precision the lowest among all folds (0.629), suggesting that this fold’s model classifies more samples as MILCO overall, which raises recall at the cost of more false alarms, rather than genuinely separating the two classes better. In contrast, folds 1 and 2 provide little discrimination, with AUC values of 0.514 and 0.519.

Table 3.4: SVM (RBF) per-fold performance

Fold	AUC	AP	F_2	Precision	Recall
1	0.514	0.523	0.582	0.557	0.589
2	0.519	0.578	0.473	0.568	0.454
3	0.902	0.910	0.836	0.794	0.847
4	0.937	0.945	0.916	0.804	0.949
5	0.634	0.682	0.830	0.629	0.902
Mean $\pm$ SD	0.701 ± 0.205	0.727 ± 0.192	0.727 ± 0.190	0.671 ± 0.120	0.748 ± 0.216

The large variation between folds indicates that the classification difficulty depends strongly on the physical targets included in the test set. Nevertheless, the mean recall of 0.748 shows that the SVM maintains good sensitivity to the MILCO class across the outer folds, which is particularly relevant for the mine-detection objective.

◆ ROC curve:

Figure 3.3 presents the ROC curve obtained from the pooled out-of-fold predictions. The curve remains predominantly above the chance level, with an AUC of 0.674. This indicates that the SVM provides discriminative information beyond random classification, although the moderate AUC also reflects substantial overlap between the two classes.

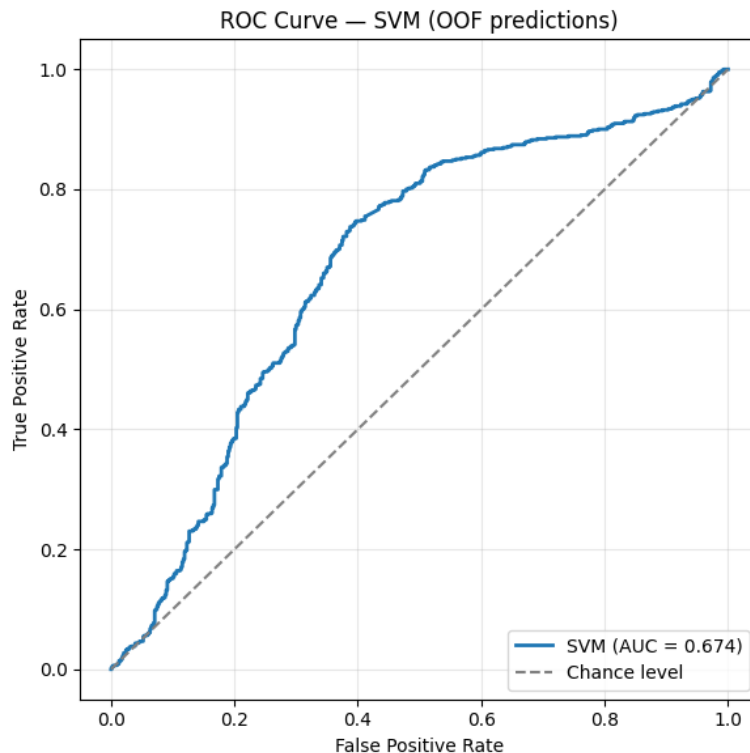


Figure 3.3: SVM (RBF) ROC curve (out-of-fold predictions).

Rather than following a smooth, uniformly bowed shape, the curve rises only gradually at low false positive rates, then climbs more steeply between roughly $\text{FPR} = 0.25$ and 0.4 , before flattening again as it approaches the top-right corner. This uneven profile is consistent with the pooled nature of the out-of-fold predictions: since each fold’s predictions come from a separately trained model of markedly different quality (Table 3.4), the pooled curve mixes a subset of signals that are well separated by the stronger fold-3/fold-4 models with a larger subset that remains close to chance under the weaker fold-1/fold-2 models, rather than reflecting a single, uniformly discriminative classifier.

◆ Out-of-fold aggregate performance:

The confusion matrix in Fig. 3.4 summarizes the classification decisions obtained from the pooled out-of-fold predictions. The classifier correctly identifies 476 of 621 MILCO samples (recall ≈ 0.767), and correctly rejects 304 of 537 non-MILCO samples (precision ≈ 0.671).

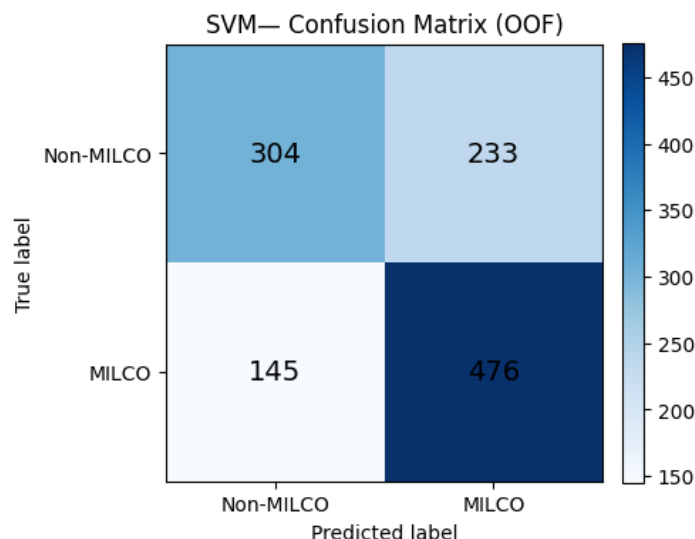


Figure 3.4: SVM (RBF) confusion matrix (out-of-fold predictions).

This favors sensitivity to the MILCO class, at the cost of more false positives consistent with F_2 weighting recall over precision.

3.1.3 Random Forest:

◆ Hyperparameters:

The Random Forest classifier combines an ensemble of decorrelated decision trees, with hyperparameters controlling both the size of the ensemble (`n_estimators`) and the complexity of each individual tree (`max_depth`, `min_samples_split`, `min_samples_leaf`, `max_features`). As with the other classifiers, hyperparameters are searched independently within each outer fold (Section 2.3.5), so the selected configuration varies from fold to fold rather than converging to a single global choice. Given its larger, six-dimensional search space compared to Logistic Regression and SVM, hyperparameters were searched with 100 Optuna trials per outer fold. The complete search space is given in Table 3.5.

Table 3.5: Optuna search space for the Random Forest classifier

Hyperparameter	Range / choices
<code>n_estimators</code>	[100, 800] (step 50)
<code>max_depth</code>	[2, 30]
<code>min_samples_split</code>	[2, 20]
<code>min_samples_leaf</code>	[1, 10]
<code>max_features</code>	$\{\sqrt{\cdot}, \log_2, \text{None}\}$
<code>class_weight</code>	$\{\text{balanced}, \text{balanced_subsample}, \text{None}\}$

◆ Per-fold results:

Table 3.6 reports outer-fold performance, with precision, recall, and F_2 computed at the fixed decision threshold of 0.5. As with the other classifiers, folds 3 and 4 achieve the strongest results (AUC of 0.919 and 0.956), while folds 1 and 2 remain close to chance level (AUC of 0.558 and 0.542); fold 5 again pairs a comparatively high recall (0.874) with low precision (0.638) and AUC (0.678). This same fold-to-fold pattern across classifiers reflects the fixed, target-based outer splits rather than model-specific behavior.

Table 3.6: Random Forest per-fold performance

Fold	AUC	AP	Precision	Recall	Accuracy
1	0.558	0.584	0.577	0.661	0.551
2	0.542	0.634	0.535	0.445	0.507
3	0.919	0.929	0.790	0.924	0.834
4	0.956	0.959	0.863	0.915	0.881
5	0.678	0.707	0.638	0.874	0.641
Mean $\pm$ SD	0.731 ± 0.197	0.763 ± 0.171	0.681 ± 0.140	0.764 ± 0.208	0.683 ± 0.168

With a mean AUC of 0.731 and mean recall of 0.764, Random Forest performs slightly better on average than both Logistic Regression and SVM across the outer folds.

◆ ROC curve:

Figure 3.5 shows the ROC curve computed on the pooled out-of-fold predictions. The curve rises sharply at low false positive rates before flattening between $FPR \approx 0.2$ and 0.4, then climbs steadily toward the upper-right corner, reaching an out-of-fold AUC of 0.702, the highest among the classifiers evaluated so far.

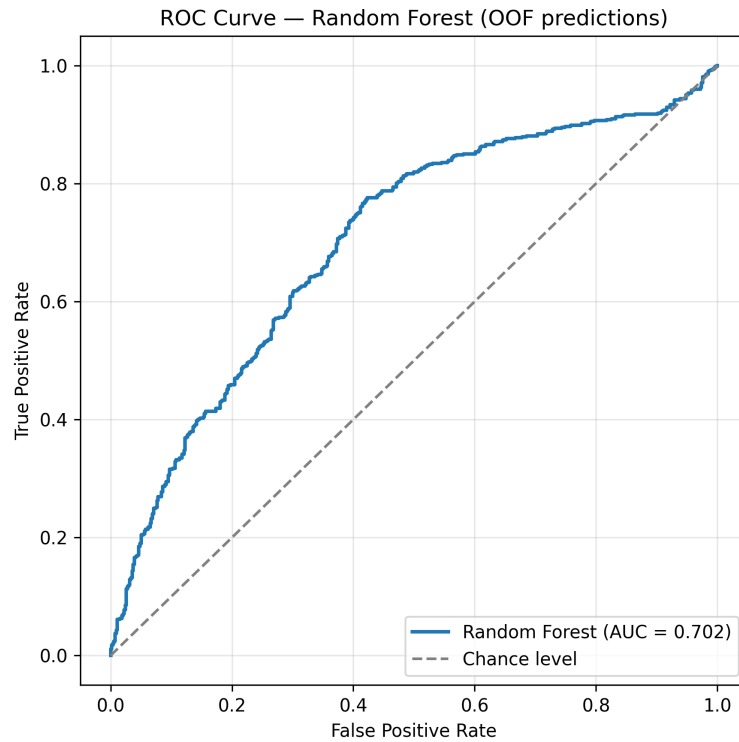


Figure 3.5: Random Forest ROC curve (out-of-fold predictions).

◆ Out-of-fold classification performance.

The classifier correctly identifies 476 of 621 MILCO samples (recall ≈ 0.767), with a precision of ≈ 0.681 among the 699 samples predicted as MILCO. Of the 537 true non-MILCO samples, 314 are correctly rejected, corresponding to a specificity of ≈ 0.585 , reflecting the same trade-off toward sensitivity already observed for Logistic Regression and SVM.

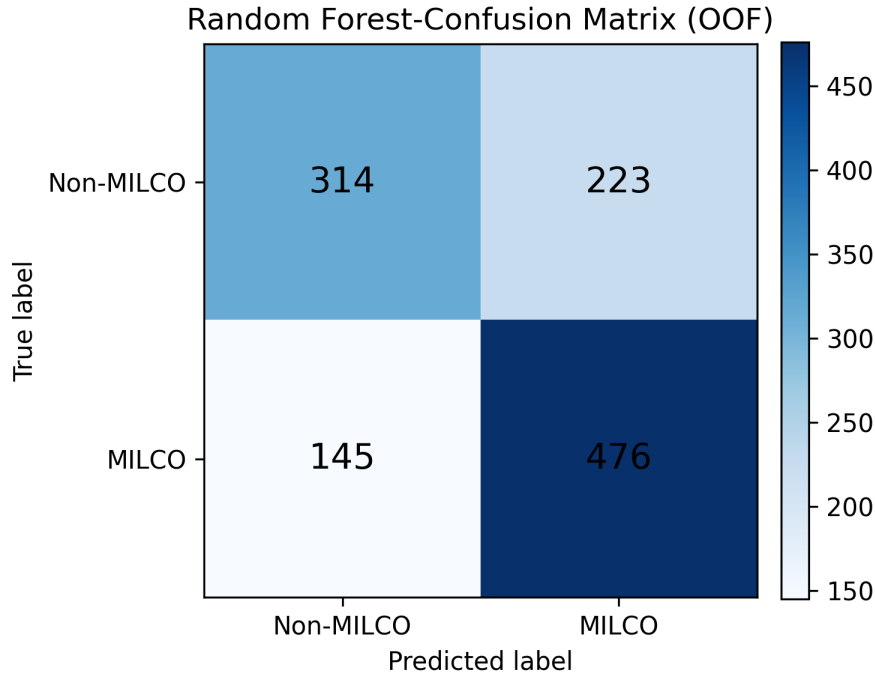


Figure 3.6: Random Forest confusion matrix obtained from pooled out-of-fold predictions.

The classifier correctly identifies 476 of 621 MILCO samples (recall ≈ 0.767), with a precision of ≈ 0.681 among the 699 samples predicted as MILCO. Of the 537 true non-MILCO samples, 314 are correctly rejected, corresponding to a specificity of ≈ 0.585 , reflecting the same trade-off toward sensitivity already observed for Logistic Regression and SVM.

3.1.4 XGBoost

XGBoost was included to evaluate a boosting-based tree ensemble against the previously considered linear and bagging-based classifiers. Unlike Random Forest, which builds trees independently and combines their predictions, XGBoost constructs the trees sequentially, with each new tree aiming to correct the errors made by the preceding ensemble. This allows the model to capture nonlinear relationships and interactions between the extracted signal features.

◆ Hyperparameters.

The model contains several parameters controlling both the boosting process and the complexity of the individual trees. These were optimized using Optuna within each outer cross-validation fold, with 150 trials per fold given the larger, nine-dimensional search space. The search covered the number of boosting rounds, the learning rate, tree depth, row and

feature subsampling, and several regularization parameters, which are particularly important given the limited number of distinct physical targets available for training.

Class imbalance was handled through `scale_pos_weight`: rather than using a fixed value, Optuna selected between no reweighting and a value computed from the negative-to-positive class ratio of the corresponding training fold. The complete search space is given in Table 3.7.

Table 3.7: Optuna search space for the XGBoost classifier

Hyperparameter	Range / choices
<code>n_estimators</code>	[100, 800] (step 50)
<code>max_depth</code>	[2, 10]
<code>learning_rate</code>	$[10^{-3}, 3 \cdot 10^{-1}]$ (log-uniform)
<code>subsample</code>	[0.5, 1.0]
<code>colsample_bytree</code>	[0.5, 1.0]
<code>min_child_weight</code>	[1, 10]
<code>reg_alpha</code> (ℓ_1)	$[10^{-4}, 10^1]$ (log-uniform)
<code>reg_lambda</code> (ℓ_2)	$[10^{-4}, 10^1]$ (log-uniform)
<code>scale_pos_weight</code>	$\{1.0, n_{\text{neg}}/n_{\text{pos}}\}$

◆ Per-fold results:

Table 3.8 reports outer-fold performance, with precision, recall, and F_2 computed at the fixed decision threshold of 0.5. The same fold pattern seen for the other classifiers recurs here: folds 3 and 4 achieve the strongest results (AUC of 0.901 and 0.946), with fold 4 also reaching the highest precision (0.908) among all folds, while folds 1 and 2 remain close to chance level (AUC of 0.570 and 0.552). With a mean AUC of 0.728, XGBoost performs similarly to Random

Table 3.8: XGBoost per-fold performance

Fold	AUC	AP	Precision	Recall	Accuracy
1	0.570	0.605	0.559	0.613	0.524
2	0.552	0.642	0.552	0.445	0.520
3	0.901	0.918	0.727	0.924	0.782
4	0.946	0.950	0.908	0.846	0.877
5	0.673	0.748	0.646	0.818	0.637
Mean $\pm$ SD	0.728 ± 0.185	0.772 ± 0.157	0.678 ± 0.147	0.729 ± 0.196	0.668 ± 0.159

Forest and slightly better than Logistic Regression and SVM on average, though with a lower mean recall (0.729) than either Random Forest (0.764) or SVM (0.748).

◆ ROC curve:

Figure 3.7 shows the ROC curve computed on the pooled out-of-fold predictions, reaching an out-of-fold AUC of 0.662. The curve rises steadily above the diagonal through most of the false positive range, most noticeably up to $\text{FPR} \approx 0.4$, before flattening closer to the diagonal between $\text{FPR} \approx 0.55$ and 0.65 and finishing with a final gradual climb toward the upper-right corner.

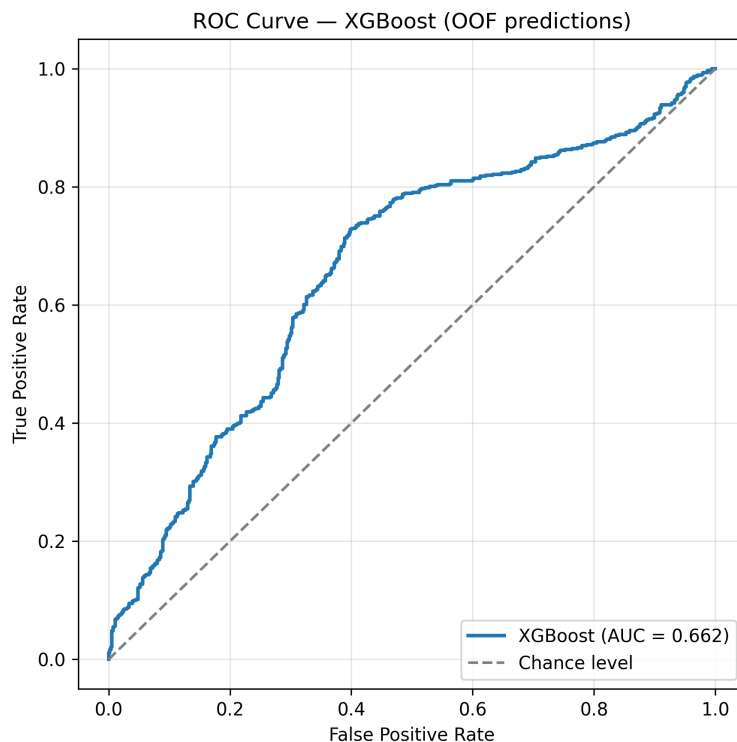


Figure 3.7: XGBoost ROC curve (out-of-fold predictions).

◆ Out-of-fold classification performance:

Fig. 3.8 shows the confusion matrix from the pooled out-of-fold predictions at the fixed threshold of 0.5.

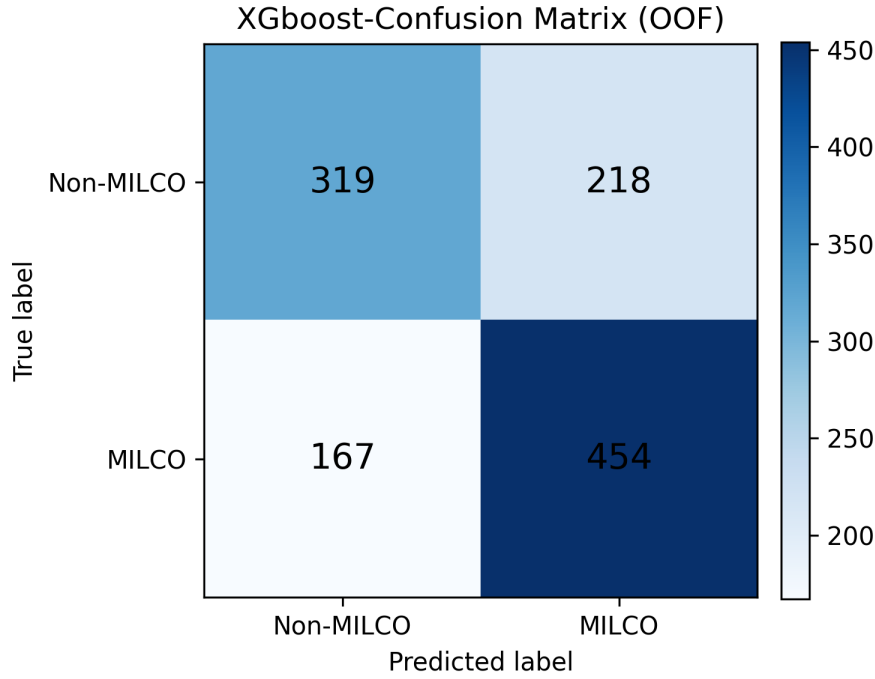


Figure 3.8: XGBoost confusion matrix obtained from pooled out-of-fold predictions.

Of the 621 MILCO signals, 454 are correctly flagged and 167 are missed (recall ≈ 0.731); of the 672 signals predicted as MILCO, 454 are correct (precision ≈ 0.676). Of the 537 non-MILCO signals, 319 are correctly cleared, corresponding to a specificity of ≈ 0.594 .

3.1.5 Light GBM

◆ Hyperparameters:

LightGBM introduces a similar set of hyperparameters to XGBoost, with a few adjustments reflecting its leaf-wise growth strategy. In addition to the number of boosting rounds, learning rate, row and feature subsampling, and ℓ_1/ℓ_2 regularization already described for XGBoost, the search also included `num_leaves`, which directly bounds the number of leaves a tree can grow via leaf-wise splitting, and `min_child_samples`, the minimum number of samples required in a leaf. As with the other classifiers, hyperparameters were optimized using Optuna within each outer fold, with 150 trials per fold given the similarly large, nine-dimensional search space. Class imbalance was again handled through `scale_pos_weight`, chosen between no reweighting and the negative-to-positive class ratio of the corresponding training fold. The complete search space is given in Table 3.9.

Table 3.9: Optuna search space for the LightGBM classifier

Hyperparameter	Range / choices
<code>n_estimators</code>	[100, 800] (step 50)
<code>max_depth</code>	[2, 10]
<code>num_leaves</code>	[7, 127]
<code>learning_rate</code>	$[10^{-3}, 310^{-1}]$ (log-uniform)
<code>subsample</code>	[0.5, 1.0]
<code>colsample_bytree</code>	[0.5, 1.0]
<code>min_child_samples</code>	[5, 50]
<code>reg_alpha</code> (ℓ_1)	$[10^{-4}, 10^1]$ (log-uniform)
<code>reg_lambda</code> (ℓ_2)	$[10^{-4}, 10^1]$ (log-uniform)
<code>scale_pos_weight</code>	$\{1.0, n_{\text{neg}}/n_{\text{pos}}\}$

◆ Per-fold results:

Table 3.10 reports outer-fold performance, with precision, recall, and F_2 computed at the fixed decision threshold of 0.5. The same fold pattern recurs once again: folds 3 and 4 achieve the strongest results (AUC of 0.895 and 0.836), while folds 1 and 2 remain close to chance level (AUC of 0.586 and 0.544). With a mean AUC of 0.707, LightGBM performs slightly below

Table 3.10: LightGBM per-fold performance

Fold	AUC	AP	Precision	Recall	Accuracy
1	0.586	0.606	0.583	0.677	0.559
2	0.544	0.639	0.540	0.454	0.511
3	0.895	0.912	0.673	0.924	0.729
4	0.836	0.853	0.764	0.803	0.771
5	0.674	0.743	0.648	0.797	0.633
Mean $\pm$ SD	0.707 ± 0.154	0.751 ± 0.132	0.642 ± 0.086	0.731 ± 0.178	0.641 ± 0.110

both Random Forest (0.731) and XGBoost (0.728), but ahead of SVM (0.701) and Logistic Regression, on average across the outer folds.

◆ ROC curve:

Figure 3.9 shows the ROC curve computed on the pooled out-of-fold predictions, reaching an out-of-fold AUC of 0.693. The curve climbs steadily above the diagonal from low false positive rates onward, reaching a true positive rate of about 0.8 by $\text{FPR} \approx 0.5$, then flattens somewhat between $\text{FPR} \approx 0.5$ and 0.65 before a further rise toward the upper-right corner.

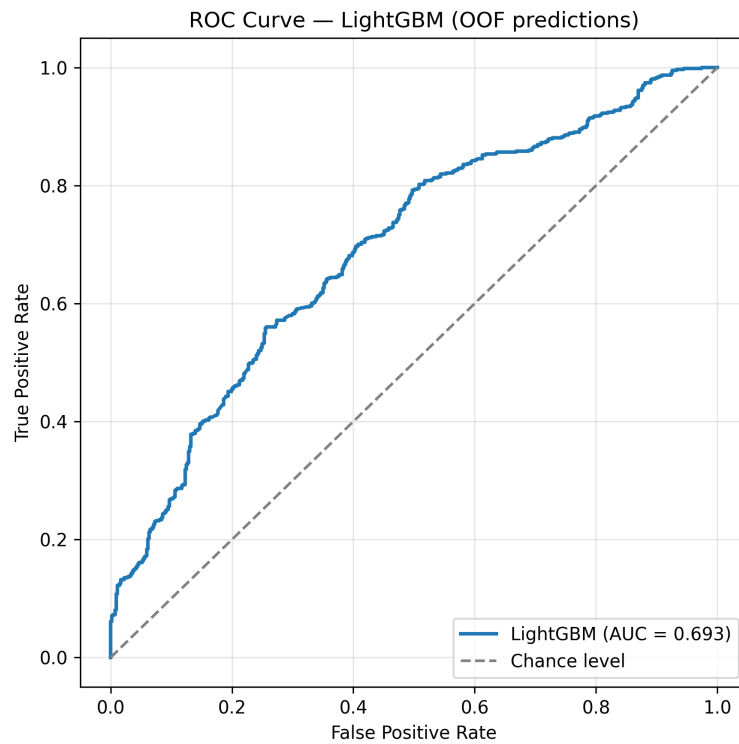


Figure 3.9: LightGBM ROC curve (out-of-fold predictions).

◆ Out-of-fold classification performance.

Fig. 3.10 shows the confusion matrix from the pooled out-of-fold predictions at the fixed threshold of 0.5.

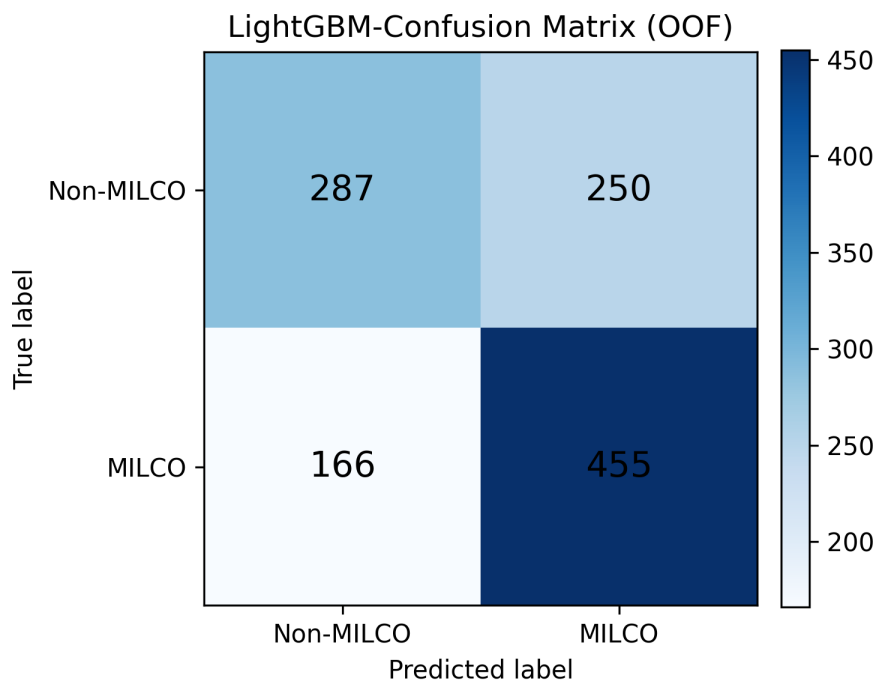


Figure 3.10: LightGBM confusion matrix obtained from pooled out-of-fold predictions.

Of the 621 MILCO signals, 455 are correctly flagged and 166 are missed (recall ≈ 0.733); of the 705 signals predicted as MILCO, 455 are correct (precision ≈ 0.645). Of the 537 non-MILCO signals, 287 are correctly cleared, corresponding to a specificity of ≈ 0.535 .

3.1.6 Comparison Across Classifiers

Figure 3.11 compares the five classifiers across AUC, precision, and recall using pooled out-of-fold predictions and 95% bootstrap confidence intervals.

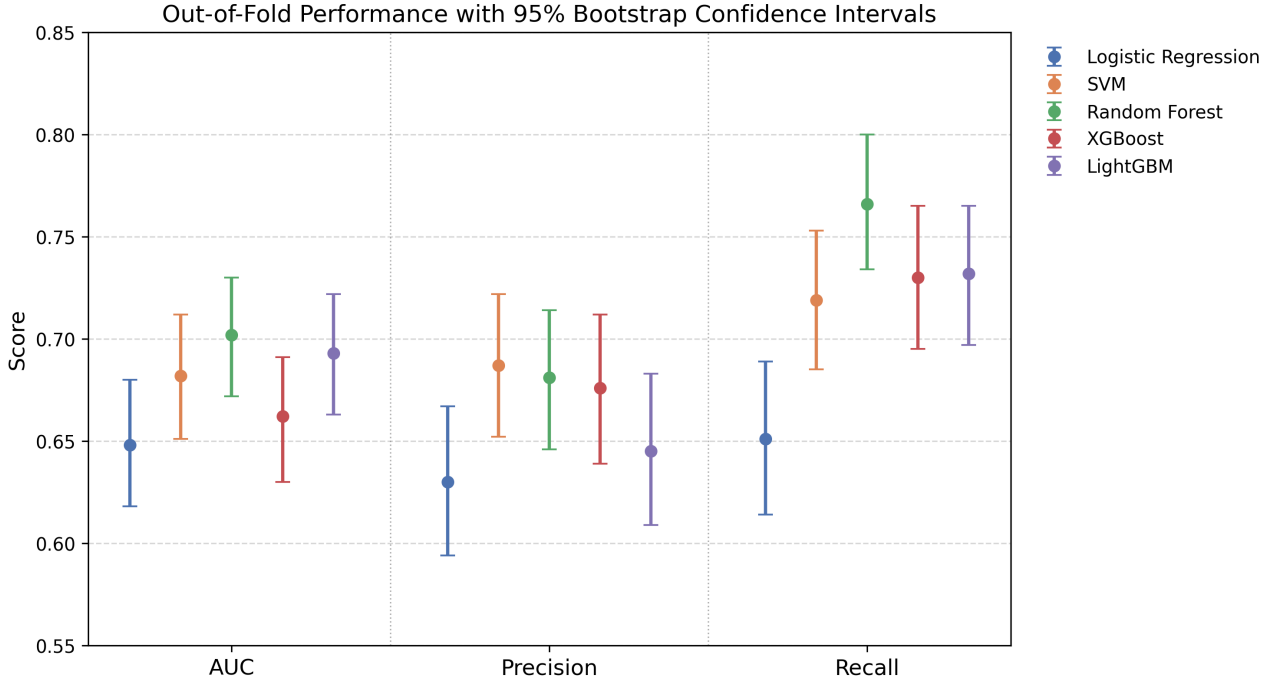


Figure 3.11: Out-of-fold performance of all five classifiers, with 95% bootstrap confidence intervals, grouped by metric.

Random Forest achieves the highest recall (0.766), and its confidence interval does not overlap with that of Logistic Regression (0.651), the weakest classifier on this metric – a meaningful gap given the recall-priority objective of this task. It does, however, overlap with SVM, XGBoost, and LightGBM, so Random Forest cannot be confidently distinguished from the other nonlinear classifiers on recall alone. Differences on AUC and precision are similarly inconclusive, with overlapping intervals across all four nonlinear classifiers. Logistic Regression trails on every metric, suggesting a linear boundary alone does not fully separate MILCO from non-MILCO signals. The wide intervals overall reflect the limited number of distinct physical targets (27) in the dataset.

Taken together, these results point to a single robust conclusion: nonlinear classifiers consistently outperform the linear baseline, but no nonlinear classifier can be confidently

distinguished from another. Combined with the observation that the same folds are easy or difficult across every classifier (Section 2.3.5), this suggests that classifier choice, beyond the linear/nonlinear distinction, is not the main factor limiting performance on this dataset; the feature representation and the intrinsic difficulty of certain physical targets are more likely candidates. Rather than selecting a single best classifier by hand, the next section evaluates H2O AutoML, which automatically searches over and combines multiple model families, to test whether a broader, automated search can close this gap.

3.1.7 H2O AutoML

H2O AutoML performs an automated search across multiple model families – gradient boosting machines (GBM), extremely randomized trees (XRT), and stacked ensembles – selecting, for each fold, the model that maximizes the F_2 score among trials clearing a minimum precision floor of 0.60, consistent with the selection criterion applied to every classifier in this chapter (Section 2.3.5). H2O’s XGBoost implementation was unavailable in this environment and was excluded from the search.

Table 3.11: Per-fold performance of H2O AutoML.

Fold	AUC	AP	F_2	Precision	Recall	Best model
1	0.540	0.574	0.609	0.566	0.621	GBM (grid)
2	0.539	0.622	0.463	0.552	0.445	GBM (grid)
3	0.868	0.884	0.810	0.737	0.831	XRT
4	0.940	0.945	0.917	0.833	0.940	GBM (grid)
5	0.682	0.741	0.797	0.632	0.853	GBM (grid)

Out-of-fold aggregate performance was $AUC = 0.695$ (95% CI: 0.665–0.724), average precision = 0.719 (95% CI: 0.679–0.758), precision = 0.667 (95% CI: 0.631–0.703), and recall = 0.740 (95% CI: 0.706–0.773), computed over 1,000 bootstrap resamples. The selected classifiers were predominantly gradient boosting models, with a single fold favoring an extremely randomized trees model; no stacked ensemble was selected under the F_2 + precision-floor criterion in any fold. As with the individual classifiers compared in the previous section, per-fold performance was highly heterogeneous, ranging from near-chance discrimination ($AUC = 0.54$, folds 1–2) to strong separation ($AUC = 0.94$, fold 4). Given that

GroupKFold partitions the data by physical target identity rather than by individual signal, this heterogeneity is consistent with the limited number of distinct targets (27) rather than instability in the AutoML search itself. Which specific targets drive this difficulty, and how they relate to the extracted features, is examined in the following section.

3.2 Target-Level Difficulty Analysis

The classifier comparison in Chapter 3 showed that performance varies substantially across outer folds, with the same folds performing strongly or weakly regardless of which classifier was used. This section investigates that pattern at the level of individual physical targets, first identifying which targets are hardest to classify (Section 3.2.1), then showing that this difficulty is consistent across classifiers and explains the fold-level pattern observed earlier (Section 3.2.2), and finally relating it to the targets’ physical metadata (Section 3.2.3).

3.2.1 Per-Target OOF Performance

Out-of-fold predictions were aggregated by physical target to assess whether classification difficulty is concentrated in specific targets rather than spread evenly across the dataset. Table 3.12 lists the six targets with the lowest Random Forest OOF accuracy.

Table 3.12: The six targets with the lowest OOF classification accuracy.

Target	Name	Class	Signals	Accuracy
1	Driver Evaluation Unit (DEU)	Mine	46	0.000
24	Hollow alum cylinder w/ notch	Non-Mine	36	0.056
17	2 ft alum cylinder	Non-Mine	14	0.071
23	Solid alum cylinder w/ notch	Non-Mine	40	0.075
2	Rock	Non-Mine	25	0.080
15	2:1 aspect telephone pole	Non-Mine	37	0.108

These six targets account for the majority of misclassifications: target 1 is a true Mine that is never correctly detected, while the remaining five are Non-Mine targets confidently misclassified as Mine. This pattern is consistent across all classifiers evaluated in Chapter 3, not specific to Random Forest, as shown next.

3.2.2 Fold Composition and Cross-Classifier Consistency

Since the outer cross-validation split is identical across all classifiers, each outer fold contains the same set of physical targets regardless of which model is trained. Table 3.13 lists the target composition of each of the five outer folds.

Table 3.13: Target composition of the five outer folds.

Fold	Targets	Contains hard-six targets
1	2, 3, 8, 12, 23, 28	2, 23
2	1, 7, 9, 15, 18, 19	1, 15
3	10, 11, 13, 16, 21	-
4	6, 14, 20, 25	-
5	5, 17, 22, 24, 29, 30	17, 24

Folds 1, 2, and 5 each contain at least two of the six hardest targets identified in Section 3.2.1, while folds 3 and 4 contain none. This distribution matches the per-fold performance reported in Chapter 3 for every classifier: folds 3 and 4 consistently achieved the strongest results, while folds 1 and 2 consistently performed close to chance level.

Table 3.14 compares OOF accuracy for the six hardest targets across four classifiers (Random Forest, SVM, XGBoost, and LightGBM).

Table 3.14: OOF accuracy for the six hardest targets across classifiers.

Target	Name	RF	SVM	XGBoost	LightGBM
1	Driver Evaluation Unit (DEU)	0.000	0.000	0.000	0.000
2	Rock	0.080	0.280	0.080	0.120
15	2:1 aspect telephone pole	0.108	0.189	0.135	0.135
17	2 ft alum cylinder	0.071	0.214	0.143	0.143
23	Solid alum cylinder w/ notch	0.075	0.100	0.000	0.000
24	Hollow alum cylinder w/ notch	0.056	0.083	0.139	0.222

Target 1 is misclassified with 0% accuracy by every classifier, and the remaining five targets rank among the worst-performing targets for all four classifiers, despite these models spanning linear, kernel, and tree-based approaches. This indicates that classification difficulty

on this dataset is primarily driven by which physical targets are being classified, rather than by the choice of classifier.

3.2.3 Relating Difficulty to Target Metadata

To investigate the factors associated with the observed classification errors, the six targets with the lowest classification performance are examined in terms of their physical characteristics and acquisition conditions. Their metadata are presented in Table 3.15.

Table 3.15: Metadata for the six hardest-to-classify targets.

Target	Name	Class	Burial	Ranges (m)	Signals
1	Driver Evaluation Unit (DEU)	Mine	Proud	35	46
2	Rock	Non-Mine	Proud	35	25
15	2:1 aspect telephone pole	Non-Mine	Proud	15, 25, 30	37
17	2 ft alum cylinder	Non-Mine	Proud	10	14
23	Solid alum cylinder w/ notch	Non-Mine	Proud	5, 15, 20	40
24	Hollow alum cylinder w/ notch	Non-Mine	Proud	5, 15	36

Four of the six targets – the telephone pole (15) and the three aluminum cylinders (17, 23, 24) – share a slender, cylindrical shape and are frequently classified as Mine with high confidence (Section 3.2.1). Their acoustic responses may resemble those of cylindrical ordnance in the Mine class, such as howitzer rounds and mortars, causing the classifiers to associate shape with class.

Burial condition does not explain the pattern: all six targets were measured in the Proud condition, but other Proud targets are classified with high accuracy.

Aspect angle also does not explain the pattern. Target 1 (DEU) has 0% accuracy across all 18 surveyed angles, while targets 23 and 24 show low accuracy at most angles. An aggregate check suggested lower accuracy beyond $\pm 80^\circ$ (28% vs. 71%), but this result is affected by the survey design: only targets 1, 2, 3, 23, and 24 were surveyed beyond $\pm 80^\circ$, four of which are already among the six targets with the lowest performance. Excluding these targets leaves only target 3, so no reliable angle effect can be established.

Range shows a similar pattern. Excluding the six targets, accuracy is highest at close range (100% at 5 m and 10 m) and lower at 15–25 m (64–68%), consistent with signal

degradation over distance. The six targets, however, remain difficult to classify even at close range (0% at 5 m and 14% at 10 m), so proximity does not improve their classification.

Overall, burial condition, aspect angle, and range do not explain these errors. The shared cylindrical geometry appears to be a more likely factor, as the acoustic responses of these targets may overlap with those of mine targets. This observation motivates the feature-level analysis that follows.

3.3 Model Interpretability

Predictive performance indicates how well the classifier distinguishes MILCO from non-MILCO targets, whereas interpretability provides insight into the basis of these decisions. This section therefore examines which features contribute most to the Random Forest predictions at the global level and how they influence specific difficult predictions.

The analysis is conducted at two levels. *Global interpretability* uses impurity based and permutation feature importance to identify the most influential features. *Local interpretability* uses SHAP (SHapley Additive exPlanations) [47] to explain individual predictions, with a focus on the most confidently misclassified signals among the six hardest targets. Section 3.3.1 presents the global analysis, while Section 3.3.2 presents the local SHAP analysis.

3.3.1 Global Interpretability

The Random Forest was fitted to the full cleaned dataset using the median or mode of the hyperparameters selected by Optuna across the five outer folds (500 trees, a maximum depth of 2, a minimum of 10 samples required to split a node, a minimum of 5 samples per leaf, and the logarithm of the number of features considered at each split). Two complementary feature importance measures were computed for this model. The impurity based importance, obtained from the reduction in impurity across the trees, is computationally efficient but can be biased towards continuous features [39]. Permutation importance was also computed using 10 repeats and ROC AUC as the scoring metric. This measure shows how much the model performance decreases when each feature is randomly changed, but it requires more computation. Tables 3.16 and 3.17 report the top 20 features under each method.

Table 3.16: Top 20 features by impurity-based importance

Feature	Importance
10th percentile	0.058100
Ratio beyond 0.5σ	0.054700
Sum of values	0.047300
Median	0.043200
Coefficient of variation	0.038500
Maximum Langevin fixed point ($m = 3$, $r = 30$)	0.038000
Count above mean	0.034500
Absolute energy	0.033100
Minimum	0.028800
CWT coefficient (c_1 , $w = 2$)	0.028300
Mean change	0.026800
Energy ratio by chunks (segment 6)	0.025800
CWT coefficient (c_1 , $w = 5$)	0.024100
Aggregated linear trend slope (chunk length 5, variance)	0.023900
Skewness	0.023700
Linear trend p -value	0.022800
FFT coefficient (imaginary part, coefficient 1)	0.021800
Fourier entropy (100 bins)	0.018800
Index mass quantile ($q = 0.1$)	0.018700
Energy ratio by chunks (segment 7)	0.018700

Overall, the top features are mainly basic statistical and distributional descriptors, such as percentiles, median, sum, variability, and amplitude threshold measures. Energy, trend, and frequency or time-frequency features are also represented, but to a lesser extent.

Table 3.17: Top 20 features by permutation importance (roc_auc)

Feature	Importance
Mean change	0.003000
Ratio beyond 0.5σ	0.002400
Median	0.002200
Coefficient of variation	0.002100
Aggregated linear trend slope (chunk length 5, variance)	0.001800
10th percentile	0.001800
Sum of values	0.001700
CWT coefficient ($c_1, w = 2$)	0.001500
CWT coefficient ($c_1, w = 5$)	0.001500
Absolute energy	0.001400
Ratio beyond 5σ	0.001100
Energy ratio by chunks (segment 0)	0.001100
Energy ratio by chunks (segment 9)	0.001100
Energy ratio by chunks (segment 1)	0.001100
Skewness	0.001000
Linear trend p -value	0.001000
Friedrich coefficient ($c_0, m = 3, r = 30$)	0.001000
Change quantiles (mean, $q_h = 1.0, q_l = 0.2$)	0.000900
Energy ratio by chunks (segment 7)	0.000900
CWT coefficient ($c_7, w = 2$)	0.000900

◆ Agreement between the two methods:

As shown in Figure 3.12, the two rankings show substantial agreement, with 7 of the top 10 features (70%) and 13 of the top 20 (65%) shared between the two methods. The main difference is *mean change*, which ranks 11th by impurity based importance but first by permutation importance.

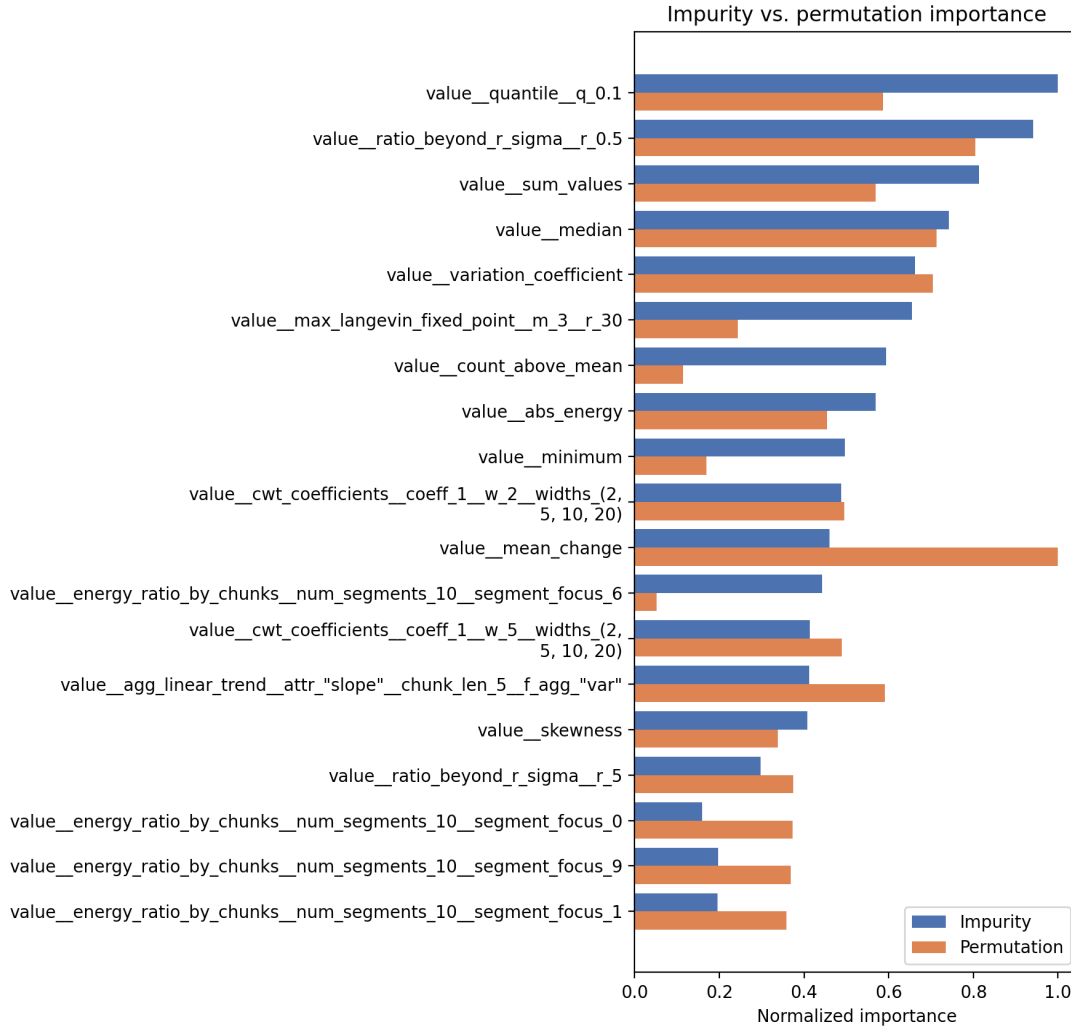


Figure 3.12: Impurity-based versus permutation importance, normalized to $[0, 1]$ within each method.

◆ Feature type interpretation:

The most influential features are mainly statistical and distributional descriptors, including percentiles, median, sum, variability, and amplitude threshold measures. Energy, trend, and frequency or time-frequency features are also represented, but to a lesser extent. This indicates that the shallow Random Forest relies primarily on basic signal characteristics for class discrimination.

3.3.2 Local Interpretability

To complement the target-level difficulty findings of Section 3.2 with the global feature analysis, SHAP explanations were generated for the five Non-Mine targets most confidently misclassified as Mine (targets 2, 15, 17, 23, and 24). Target 1, the sole Mine target among the six hardest cases, was misclassified in the opposite direction and is examined separately below via Table 3.18

rather than included in the beeswarm, since combining both error directions in a single plot would mix two opposing SHAP patterns and obscure the interpretation of either. Figure 3.13 shows the SHAP beeswarm plot restricted to these five false-positive targets.

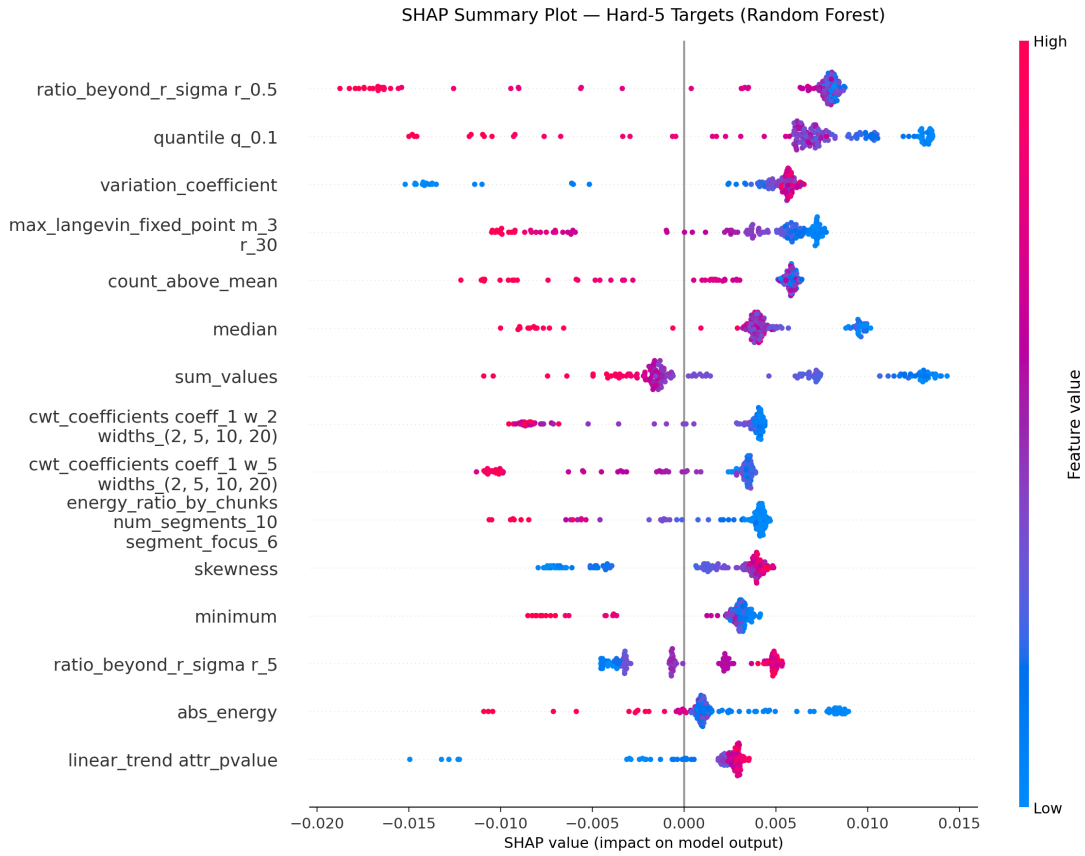


Figure 3.13: SHAP beeswarm plot restricted to the five Non-Mine targets most confidently misclassified as Mine (Random Forest). Color indicates the underlying raw feature value (pink = high, blue = low); horizontal position indicates the SHAP contribution (right = pushes toward Mine).

Four features – `quantile_q_0.1`, `ratio_beyond_r_sigma_0.5`, `max_langevin_fixed_point` ($m = 3$, $r = 30$), and `count_above_mean` – are consistently among the most influential features across these five targets, and all four also rank among the top global features (Section 3.3.1), indicating that the model’s core statistical descriptors, not a distinct rarely-used feature, drive these misclassifications. Although all five signals are true Non-Mine targets, most of their SHAP contributions on these features are positive, i.e. push the prediction towards Mine – the feature-level mechanism behind their misclassification. For `quantile_q_0.1`, `max_langevin_fixed_point`, and `count_above_mean`, this positive push corresponds to low raw feature values; `ratio_beyond_r_sigma_0.5` is less clearly monotonic, with high-value points appearing on both sides of the axis.

To test whether these patterns reflect genuinely atypical signals, each feature value is compared against the typical interquartile range of its true class, computed from the non-hard-6

signals (Table 3.18).

Table 3.18: Class-conditional typical ranges (25th–75th percentile, non-hard-6 targets) and hard-target raw values for the four common features.

	quantile_q_0.1	ratio_beyond_r_sigma_0.5	max_langevin	count_above_mean
Typical Mine (IQR)	0.000008–0.000038	0.104–0.136	0.011–0.052	162–194
Typical Non-Mine (IQR)	0.000047–0.000108	0.134–0.342	0.043–0.137	195–246
Target 1 (true: Mine)	0.000104	0.283	0.083	225
Target 2 (true: Non-Mine)	0.000037	0.071	0.022	148
Target 15 (true: Non-Mine)	0.000009	0.117	0.010	153
Target 17 (true: Non-Mine)	0.000009	0.091	0.012	188
Target 23 (true: Non-Mine)	0.000004	0.096	0.006	169
Target 24 (true: Non-Mine)	0.000003	0.089	0.005	174

The table confirms the pattern seen in the figure. For the five Non-Mine targets, every raw value falls below the lower bound of the Non-Mine range, several landing directly within the typical Mine range – consistent with the positive SHAP push observed above. Target 1 shows the mirror pattern: all four raw values fall outside the typical Mine range and inside the typical Non-Mine range, consistent with its misclassification in the opposite direction.

Combined with the shape-clustering observation from Section 3.2.3, this gives a concrete, feature-level explanation for the earlier target-level difficulty pattern: the slender aluminium cylinders and telephone pole (targets 17, 23, 24, 15) produce acoustic returns whose basic amplitude and energy statistics genuinely resemble typical Mine-class values, while target 1’s return genuinely resembles a typical Non-Mine signal. In both cases, the misclassification traces to a real property of the underlying signal rather than an artifact of the explanation method.

Conclusion

This chapter presented the complete classification and interpretability analysis applied to the LF-SAS dataset. Several classifiers were evaluated using target-grouped cross-validation, with Random Forest achieving the strongest overall performance, although no clear superiority over the other nonlinear models was established. The results showed that classification performance was strongly affected by the physical targets included in each test fold, with six targets identified as particularly difficult to classify. The limited number of distinct targets in the dataset is an important limitation, as it restricts the ability to assess model generalisation

to new targets and increases the influence of individual target characteristics on the results. Feature importance and SHAP analyses further showed that statistical and distributional signal characteristics play an important role in the classification decisions. Overall, the results highlight the need for a larger and more diverse dataset, together with better feature representation and target independent validation, to improve the reliability and generalisation of LF-SAS target classification.

General Conclusion

This work addressed the central question raised in Chapter 1, namely how a machine learning classifier for underwater mine detection can be designed, evaluated, and interpreted in a way that is both operationally effective and sufficiently transparent to be trusted in a defence context. The project was structured as a complete pipeline, from raw low-frequency SAS signals to interpretable classification decisions, built and validated exclusively on real, multi-target acoustic data from the TREX13 campaign.

Chapter 2 established the technical foundation of this pipeline: focused target signals were recovered through the ω -k algorithm and converted into 195 cleaned tsFresh features per signal, with feature cleaning and model selection performed under a target-grouped, nested cross-validation protocol to prevent leakage between correlated signals from the same target, and a cost-sensitive F_2 objective adopted to reflect the asymmetric cost of missed mines relative to false alarms.

Chapter 3 showed that non-linear classifiers, namely Random Forest, SVM, XGBoost, LightGBM, and H2O AutoML, consistently outperformed the linear baseline, though none could be confidently distinguished from another once confidence intervals were accounted for. The true bottleneck was instead found at the target level: six physical targets, predominantly slender cylindrical objects, were misclassified across every classifier and alone accounted for most of the observed performance variation. Global feature importance, computed through both impurity-based and permutation methods, showed that the classifier relies primarily on simple statistical and distributional descriptors of the signal, such as low-order quantiles, amplitude thresholds, and basic variability measures, rather than on complex frequency- or wavelet-based features. Local SHAP explanations applied to the misclassified targets confirmed that this reliance directly explains the observed errors, as the raw values of these same features, for the five confidently misclassified non-mine targets, fall outside their expected class range and inside the range typical of mine signals, and vice versa for the single misclassified mine target. These errors therefore reflect genuine acoustic ambiguity between the misclassified objects and the opposite class, rather than artefacts of the explanation method, grounding the model’s decisions, both correct and incorrect, in physically interpretable signal characteristics.

Taken together, these results show that non-linear classifiers trained on tsFresh-derived low-frequency SAS features can reach operationally meaningful recall (0.73–0.77 on average) while remaining transparent, since both their global behaviour and their specific failure modes can be traced back to real signal properties. At the same time, with only 27 distinct physical targets, a small number of ambiguous objects can dominate the observed performance, a constraint that classifier tuning alone cannot overcome. These results must also be qualified, as the strongest performance was obtained under favourable conditions, namely surface-laid targets at close range, whereas real mine countermeasures scenarios often involve targets buried or obscured by sediment and vegetation, conditions only partly represented here. Feature-based time-series classification alone may therefore not suffice once this full environmental diversity is considered. Future work should enlarge and diversify the dataset, where needed with simulated scattering data, and enrich the feature set with highlight-shadow, shape-based, physics-informed, and multi-aspect descriptors. Combining SAS image-based detection with feature-based deep learning classification is a further promising direction, though one requiring substantially more data than is currently available. Overall, this project shows that model transparency and operational performance are not competing objectives, though further progress will depend as much on the diversity of the underlying data and features as on the classifiers themselves.

References

- [1] Geneva International Centre for Humanitarian Demining, “Underwater Explosive Ordnance,” Geneva International Centre for Humanitarian Demining (GICHD), Geneva, Switzerland, Tech. Rep., 2023. [Online]. Available: https://www.gichd.org/fileadmin/user_upload/GICHD_Underwater_EO_23052025_WEB.pdf.
- [2] M. Seidel, T. Frey, and J. Greinert, “Underwater UXO detection using magnetometry on hovering AUVs,” *Journal of Field Robotics*, vol. 40, no. 4, pp. 848–861, 2023. DOI: 10.1002/rob.22159.
- [3] S. G. Kargl and K. L. Williams, “Full Scale Measurement and Modeling of the Acoustic Response of Proud and Buried Munitions at Frequencies from 1–30 kHz,” Applied Physics Laboratory, University of Washington, Second Interim Report SERDP Project MR-1665, Aug. 2010. [Online]. Available: https://s3.us-gov-west-1.amazonaws.com/sepub-demo-0001-124733793621-us-gov-west-1/s3fs-public/project_documents/MR-1665-IR.pdf.
- [4] Royal Military Academy, *Welcome to the Royal Military Academy*, Accessed June 2026, 2026. [Online]. Available: <https://www.rma.ac.be/en>.
- [5] Royal Military Academy Research Portal, *Communication, Information, Systems and Sensors*, Accessed June 2026, 2026. [Online]. Available: <https://researchportal.rma.ac.be/en/organisations/communication-information-systems-and-sensors/>.
- [6] S. Hożyń, “A Review of Underwater Mine Detection and Classification in Sonar Imagery,” *Electronics*, vol. 10, no. 23, p. 2943, 2021. DOI: 10.3390/electronics10232943.
- [7] O. L. Lopera Tellez, A. Borghgraef, and E. Mersch, “The Special Case of Sea Mines,” in *Mine Action: The Research Experience of the Royal Military Academy of Belgium*, IntechOpen, 2017. DOI: 10.5772/66994.
- [8] B. Szturomski, *The Effect of an Underwater Explosion on a Ship*, 2015.
- [9] H. Nain, M. C. Isa, M. M. Muhammad, N. H. N. Yusoff, M. S. D. Yati, and I. M. Nor, “Management of Naval Vessels’ Electromagnetic Signatures: A Review of Sources and Countermeasures,” *Defence S&T Technical Bulletin*, vol. 6, no. 2, p. 93, 2013.

- [10] J. J. Holmes, “Past, Present, and Future of Underwater Sensor Arrays to Measure the Electromagnetic Field Signatures of Naval Vessels,” *Marine Technology Society Journal*, vol. 49, no. 6, pp. 123–133, 2015, ISSN: 0025-3324. DOI: 10.4031/MTSJ.49.6.1.
- [11] A. Schwartz and E. Brandenburg, “An Overview of Underwater Technologies for Operations Involving Underwater Munitions,” *Marine Technology Society Journal*, vol. 43, no. 4, pp. 62–75, 2009.
- [12] L. Bjørnø and I. Bjørnø, “Underwater Acoustics and its Applications: A Historical Review,” pp. 3–8, 1999.
- [13] NOAA Ocean Exploration, *Sonar*, <https://oceanexplorer.noaa.gov/technology/sonar/>, Accessed: 26 June 2026, 2023.
- [14] NOAA Ocean Exploration, *Sonar: Seeing with Sound*, Fact Sheet, Accessed August 2026, 2025. [Online]. Available: <https://oceanexplorer.noaa.gov/wp-content/uploads/2025/04/sonar-fact-sheet.pdf>.
- [15] A. Schwartz and E. Brandenburg, “An Overview of Underwater Technologies for Operations Involving Underwater Munitions,” *Marine Technology Society Journal*, vol. 43, no. 4, pp. 62–75, 2009. [Online]. Available: <https://www.ingentaconnect.com/content/mts/mtsj/2009/00000043/00000004/art00009>.
- [16] L. Bjørnø and I. Bjørnø, “Underwater Acoustics and its Applications: A Historical Review,” pp. 3–8, May 1999, 24–27 May 1999.
- [17] P. Blondel, *The Handbook of Sidescan Sonar*. Berlin: Springer, 2009.
- [18] NOAA Ocean Exploration, *Synthetic Aperture Sonar (SAS)*, <https://oceanexplorer.noaa.gov/technology/sonar-sas/>, Last updated August 29, 2024. Accessed: June 30, 2026, 2024.
- [19] O. Bryan, R. E. Hansen, T. S. F. Haines, N. Warakagoda, and A. Hunter, “Challenges of Labelling Unknown Seabed Munition Dumpsites from Acoustic and Optical Surveys: A Case Study at Skagerrak,” *Remote Sensing*, vol. 14, no. 11, p. 2619, 2022. DOI: 10.3390/rs14112619.
- [20] E. Chang, *Side Scan Sonar Frequency & Resolution Guide (High vs Low Frequency)*, <https://www.uniquegroup.com/media-centre/blog-articles/side-scan-sonar-frequency/>, Accessed: July 3, 2026, 2026.

- [21] J. R. Dubberley and M. L. Gendron, “Synthetic Aperture Sonar Low Frequency vs. High Frequency Automatic Contact Generation,” in *Proceedings of OCEANS 2009*, Biloxi, MS, USA: IEEE, 2009. DOI: 10.1109/OCEANS.2009.5422091.
- [22] G. Hunter, *Chirps*, <https://blog.mbedded.ninja/programming/signal-processing/chirps/>, Last updated January 10, 2025. Accessed: July 4, 2026, 2025.
- [23] J. Del Rio Vera, E. Coiras, J. Groen, and B. Evans, “Automatic Target Recognition in Synthetic Aperture Sonar Images Based on Geometrical Feature Extraction,” *EURASIP Journal on Advances in Signal Processing*, vol. 2009, p. 109 438, 2009. DOI: 10.1155/2009/109438.
- [24] J. D. Tucker, M. R. Azimi-Sadjadi, and G. J. Dobeck, “Canonical Coordinates for Detection and Classification of Underwater Objects From Sonar Imagery,” in *Proceedings of OCEANS 2007 – Europe*, Aberdeen, UK, 2007. DOI: 10.1109/OCEANSE.2007.4302329.
- [25] D. P. Williams, “Underwater Target Classification in Synthetic Aperture Sonar Imagery Using Deep Convolutional Neural Networks,” in *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, Cancún, Mexico, 2016, pp. 2497–2502. DOI: 10.1109/ICPR.2016.7900012.
- [26] D. Gebhardt, K. Parikh, I. Dzieciuch, M. Walton, and N. A. Vo Hoang, “Hunting for Naval Mines with Deep Neural Networks,” in *Proceedings of OCEANS 2017*, Anchorage, AK, USA, 2017, pp. 1–5. DOI: 10.1109/OCEANS.2017.8232051.
- [27] J. A. Bucaro et al., “Acoustic identification of buried underwater unexploded ordnance using a numerically trained classifier,” *The Journal of the Acoustical Society of America*, vol. 132, no. 6, pp. 3614–3617, 2012. DOI: 10.1121/1.4763987.
- [28] Y. Seo, B. On, S. Im, T. Shim, and I. Seo, “Underwater Cylindrical Object Detection Using the Spectral Features of Active Sonar Signals with Logistic Regression Models,” *Applied Sciences*, vol. 8, no. 1, p. 116, 2018. DOI: 10.3390/app8010116.
- [29] T. Xu, X. Li, and H. Jia, “Underwater Spherical Shell Classification and Parameter Estimation Based on Acoustic Backscattering Characteristics,” *IEEE Access*, vol. 9, pp. 162 756–162 764, 2021. DOI: 10.1109/ACCESS.2021.3133055.

- [30] Applied Physics Laboratory, University of Washington, *TREX13: Target and Reverberation Experiment 2013*, <https://www.apl.uw.edu/project/project.php?id=trex13>, Project overview page, accessed 2026, APL-UW.
- [31] S. G. Kargl, K. L. Williams, T. M. Marston, D. S. Plotnick, and A. L. Espana, “Acoustic Response of Underwater Munitions near a Sediment Interface: GULFEX12, TREX13, and BAYEX14 Field Measurements,” Applied Physics Laboratory, University of Washington, Final Report ADA619420, 2015, Prepared for SERDP/ONR, available via Defense Technical Information Center (DTIC).
- [32] S. M. Murphy, P. C. Hines, and K. Dunphy, “Classifying Continuous Active Sonar Echoes for Target Recognition,” in *Proceedings of the SPIE (or relevant conference, if known)*, Contact: stefan.murphy@drdc-rddc.gc.ca, Dartmouth, Nova Scotia, Canada, 200X.
- [33] H. J. Callow, “Signal Processing for Synthetic Aperture Sonar Image Enhancement,” Ph.D. dissertation, University of Canterbury, 2003.
- [34] R. H. Stolt, “Migration by Fourier Transform,” *Geophysics*, vol. 43, no. 1, pp. 23–48, 1978.
- [35] M. P. Hayes and P. T. Gough, “Synthetic Aperture Sonar: A Review of Current Status,” *IEEE Journal of Oceanic Engineering*, vol. 34, no. 3, pp. 207–224, 2009. DOI: 10.1109/JOE.2009.2020853.
- [36] The HDF Group. “HDF5: The HDF5 Library and File Format,” Accessed: Aug. 12, 2026. [Online]. Available: <https://www.hdfgroup.org/solutions/hdf5/>.
- [37] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, “Time Series Feature Extraction on Basis of Scalable Hypothesis Tests (tsfresh – A Python Package),” *Neurocomputing*, vol. 307, pp. 72–77, 2018. DOI: 10.1016/j.neucom.2018.03.067.
- [38] P. Domingos, “A Few Useful Things to Know about Machine Learning,” *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012. DOI: 10.1145/2347736.2347755.
- [39] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd. New York, NY: Springer, 2009.
- [40] J. H. Friedman, “Greedy function approximation: a gradient boosting machine,” *Annals of Statistics*, pp. 1189–1232, 2001.

- [41] T. Zhang et al., “Improving Convection Trigger Functions in Deep Convective Parameterization Schemes Using Machine Learning,” *Journal of Advances in Modeling Earth Systems*, vol. 13, 2021. DOI: 10.1029/2020MS002365.
- [42] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
- [43] G. Ke et al., “LightGBM: A highly efficient gradient boosting decision tree,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [44] H2O.ai, *H2O AutoML: Automatic Machine Learning*, <https://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html>, H2O documentation, accessed 2026.
- [45] D. H. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.
- [46] Optuna Development Team, *Optuna: A hyperparameter optimization framework – Documentation*, <https://optuna.readthedocs.io/en/stable/>, Accessed 2026.
- [47] S. M. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 4765–4774.
- [48] F. R. Kschischang, *The Hilbert Transform*, Lecture notes, The Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto, Oct. 2006.
- [49] V. S. Tutatchikov, “Two-Dimensional Fast Fourier Transform: Butterfly in Analog of Cooley-Tukey Algorithm,” in *2016 IEEE 11th International Forum on Strategic Technology (IFOST)*, 2016, pp. 495–498. DOI: 10.1109/IFOST.2016.7884162.
- [50] MathWorks, *fftshift – Shift zero-frequency component to center of spectrum*, <https://www.mathworks.com/help/matlab/ref/fftshift.html>, MATLAB documentation, accessed 2026.
- [51] M. Christ et al. “tsfresh Documentation,” Accessed: Aug. 11, 2026. [Online]. Available: <https://tsfresh.readthedocs.io/en/latest/index.html>.

Annexes

Annexe A

This Annexe summarises the parameters of the SAS system and processing pipeline used in this thesis. Table 4.1 presents the acquisition parameters of the APL-UW tower-rail system used during TREX13, and Table 4.2 summarises the parameters used in the ω -k focusing implementation described in Chapter 2. Table 4.3 lists the full TREX13 target inventory, including target names, class labels, burial condition, and the number of files and signals extracted per target.

Table 4.1: APL-UW tower-rail SAS system parameters used during TREX13 [31].

Parameter	Value	Units
Carrier frequency band (LF)	1–30	kHz
Carrier frequency band (HF)	30–50	kHz
Number of receive elements	6	–
Pulse repetition rate	2	Hz
Rail length	42	m
Target range grid	5–40	m
Transducer depression angle	10	°
Mast pivot height above seafloor	3.3	m

Table 4.2: Parameters used for ω -k SAS image formation in this work.

Parameter	Value	Units
Sampling frequency f_s	100	kHz
Assumed sound speed c	1524	m/s
Along-track sample spacing du	0.0254 (1 inch)	m

Table 4.3: TREX-13 target inventory: class label, ranges used, and number of signals extracted per target after active-zone detection and stride-based subsampling.

#	Target name	Class	Ranges available	Burial	Files	Signals
1	driver Evaluation Unit (DEU)	Mine	35m	Proud	18	46
2	Rock	Non-Mine	35m	Proud	18	25
3	55-gal drum (water, w/ fixture)	Non-Mine	35m	Proud	18	38
5	5:1 aspect telephone pole	Non-Mine	25m, 30m, 40m	Proud	27	55
6	55-gal drum (water-filled)	Non-Mine	25m, 35m, 40m	Proud	27	76
7	3 ft alum cylinder	Non-Mine	30m, 35m, 40m	Proud	27	48
8	155mm Howitzer w/o collar	Mine	5m, 10m, 25m, 30m, 40m	Proud / Partial	54	78
9	155mm Howitzer w/ collar	Mine	5m, 10m, 15m, 20m, 35m	Proud / Partial	54	73
10	Panel target	Non-Mine	20m, 25m, 30m	Proud	27	38
11	152mm TP-T	Mine	5m, 10m, 20m, 25m, 35m	Proud / Partial	54	60
12	81mm mortar	Mine	15m	Proud	9	9
13	Scuba tank (water, w/o stem)	Non-Mine	15m, 25m, 35m	Proud	27	39
14	Scuba tank (water, w/ stem)	Non-Mine	15m, 20m, 25m	Proud	27	34
15	2:1 aspect telephone pole	Non-Mine	15m, 25m, 30m	Proud	27	37
16	2 ft alum pipe	Non-Mine	15m, 25m, 30m	Proud	27	34
17	2 ft alum cylinder	Non-Mine	10m	Proud	9	14
18	Cement block	Non-Mine	15m	Proud	9	14
19	Tire	Non-Mine	20m	Proud	9	9
20	Alum UXO replica	Mine	5m, 10m, 15m, 30m, 40m	Proud / Partial	54	58
21	Steel UXO replica	Mine	5m, 10m, 15m, 25m, 30m	Proud / Partial	54	58
22	Original material UXO	Mine	5m, 10m, 20m, 30m, 40m	Proud / Partial	54	60
23	Solid alum cylinder w/ notch	Non-Mine	5m, 15m, 20m	Proud	36	40
24	Hollow alum cylinder w/ notch	Non-Mine	5m, 15m, 20m	Proud	36	36
25	105-mm bullet-shaped UXO#1	Mine	5m, 10m, 15m, 20m, 25m	Proud / Partial	54	59
28	155mm Howitzer w/ collar (2)	Mine	15m, 25m, 30m	Proud	27	37
29	105-mm bullet-shaped UXO#2	Mine	15m, 20m, 35m	Proud	27	32
30	105-mm finned shell #1	Mine	5m, 10m, 15m, 20m	Proud / Partial	45	51
Total					855	1158

Mine class: 624 signals, Non-Mine class: 519 signals, (Targets 4, 26, 27 were not used and are omitted.)

Eight targets (8, 9, 11, 20, 21, 22, 25, 30) were surveyed under two burial conditions, proud (surface-laid) and partially buried, with 18 files acquired at the partial-buried condition for each; all other targets were surveyed proud only.

Annexe B

This Annexes summarises the mathematical definitions and properties underlying the signal processing techniques used in this thesis, in particular the ω - k focusing algorithm described in Chapter 2.

The Hilbert Transform

The Hilbert transform $\mathcal{H}[g(t)]$ of a signal $g(t)$ is defined as the convolution of $g(t)$ with the signal $1/\pi t$:

$$\mathcal{H}[g(t)] = g(t) * \frac{1}{\pi t} = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{g(\tau)}{t - \tau} d\tau, \quad (4.1)$$

understood as the Cauchy principal value of the (improper) integral. Equivalently, the Hilbert transform is the output of a linear time-invariant filter, the *Hilbert transformer*, with impulse response $1/\pi t$ [48]. The transform of a real-valued signal $g(t)$ is itself real-valued, and is commonly denoted $\hat{g}(t)$.

The Two-Dimensional Fast Fourier Transform

For a two-dimensional discrete signal $f(k, t)$ of size NM , the two-dimensional discrete Fourier transform (2-D DFT) is defined as

$$F(p, m) = \sum_{k=0}^{N-1} \sum_{t=0}^{M-1} f(k, t) e^{-j2\pi\left(\frac{kp}{N} + \frac{tm}{M}\right)}, \quad (4.2)$$

where $p = 0, \dots, N-1$ and $m = 0, \dots, M-1$ index the transformed (frequency) domain[49]. The 2-D DFT can be computed as a separable operation: a one-dimensional DFT is first applied along one dimension (e.g. each row), followed by a one-dimensional DFT along the other dimension (e.g. each column) of the resulting matrix[49]. In practice, this is computed efficiently using the Fast Fourier Transform (FFT) algorithm, which reduces the computational complexity of each one-dimensional transform from $O(N^2)$ to $O(N \log N)$.

In the context of this work, the 2-D FFT is applied to the analytic signal matrix $s_a(u, t)$, indexed by along-track position u (rows) and fast time t (columns), transforming it into the two-dimensional wavenumber domain $S(k_u, k)$, indexed by the along-track wavenumber k_u and the range wavenumber $k = 2\pi f/c$. This transformation is the necessary first step of the ω - k

algorithm , since both the subsequent phase compensation and Stolt interpolation operations are formulated directly in this wavenumber domain rather than in the original space–time domain.

The `fftshift` Operation

The Fast Fourier Transform (FFT) produces its output in a native order, with the zero-frequency (DC) component located at the first index, followed by increasing positive frequencies, and finally the negative frequencies wrapped around at the end of the array [50]:

$$\text{FFT order : } [0, \Delta f, 2\Delta f, \dots, f_{\max}, -f_{\max}, \dots, -\Delta f] \quad (4.3)$$

While mathematically correct, this ordering is inconvenient for visualization and for operations requiring a monotonically increasing frequency (or wavenumber) axis, such as interpolation in the wavenumber domain. The `fftshift` function rearranges the array by swapping its left and right halves (or, for a 2-D matrix, swapping diagonally opposite quadrants), so that the zero-frequency component is relocated to the center of the array [50]:

$$\text{Shifted order : } [-f_{\max}, \dots, -\Delta f, 0, \Delta f, \dots, f_{\max} - \Delta f] \quad (4.4)$$

This operation is a circular permutation and involves no loss or alteration of the underlying data, only a reindexing. Its inverse, `ifftshift`, restores the native FFT ordering and must be applied before computing an inverse FFT if the spectrum has been previously shifted.

Annexe C

This appendix describes the tsFresh [51] features retained after the cleaning and selection procedure described in Section 2.3.4, grouped by category. Unless otherwise noted, $x = (x_1, \dots, x_n)$ denotes the signal, μ its mean, and σ its standard deviation.

Basic Statistics

These features summarize the overall amplitude and shape of the signal, such as its energy, spread, and symmetry. Since mine-like and clutter targets tend to scatter energy differently, these simple statistics can already help distinguish between them.

Table 4.4: Basic statistical features.

Feature & Definition	Description
sum_values $\sum_{i=1}^n x_i$	Calculates the sum over the time series values
abs_energy $\sum_{i=1}^n x_i^2$	Returns the absolute energy of the time series which is the sum over the squared values
mean_change $\frac{1}{n-1} \sum_{i=1}^{n-1} (x_{i+1} - x_i)$	Returns the mean over the differences between subsequent time series values
mean_second_derivative_central $\frac{1}{n-2} \sum_{i=1}^{n-2} \frac{1}{2} (x_{i+2} - 2x_{i+1} + x_i)$	Returns the mean value of a central approximation of the second derivative
median $\text{median}(x) = \begin{cases} x_{(\frac{n+1}{2})} & \text{if } n \text{ is odd} \\ \frac{x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)}}{2} & \text{if } n \text{ is even} \end{cases}$	Median value of the signal
variation_coefficient σ/μ	Standard deviation divided by mean (relative variability)

Continued on next page

Table 4.4 – continued from previous page

Feature & Definition	Description
skewness $\frac{n}{(n-1)(n-2)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s} \right)^3,$ where $s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$	Asymmetry of the value distribution
longest_strike_below_mean $\max\{\ell : x_i, \dots, x_{i+\ell-1} < \mu\}$	Returns the length of the longest consecutive subsequence in x that is smaller than the mean of x
longest_strike_above_mean $\max\{\ell : x_i, \dots, x_{i+\ell-1} > \mu\}$	Returns the length of the longest consecutive subsequence in x that is bigger than the mean of x
count_above_mean $ \{i \in \{1, \dots, n\} \mid x_i > \bar{x}\} $	Number of points above the mean
last_location_of_minimum $\frac{\max\{i : x_i = \min(x)\}}{n}$	Relative position of the last minimum
benford_correlation $\text{corr}(P_{\text{obs}}(d), \log_{10}(1 + \frac{1}{d})),$ where $d \in \{1, 2, \dots, 9\}$ and $P_{\text{obs}}(d) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{\{\text{lead}(x_i)=d\}}$	Measures how closely the leading-digit distribution of a time series matches the theoretical Newcomb-Benford distribution, used mainly to flag anomalous or manipulated data.

Nonlinear and Complexity Measures

These features capture how irregular or predictable the signal is, beyond what simple statistics can describe. Since the scattering response of a target depends on its internal structure and material, differences in signal complexity can help separate mine-like objects from clutter.

Table 4.5: Nonlinear dynamics and complexity features.

Feature & Definition	Description
cid_ce__normalize_True $\sqrt{\sum_{i=1}^{n-1} (x_{i+1} - x_i)^2}$, on z -normalized x	This function calculator is an estimate for a time series complexity
friedrich_coefficients__coeff_0/2/3__m_3__r_30 Coefficients c_k of $\Delta x_i \approx \sum_{k=0}^3 c_k x_i^k$, least-squares fit over $r=30$ bins	Coefficients of a fitted Langevin (drift) model
max_langevin_fixed_point__m_3__r_30 $\max \{x^* \in \mathbb{R} \mid h(x^*) = 0\}$, where $h(x) \approx \sum_{k=0}^m c_k x^k$	Largest fixed point of dynamics
lempel_ziv_complexity__bins_2 $\text{lempel_ziv_complexity}(x, b) = \frac{c}{n}$, where $c = \mathcal{D} $ and $\mathcal{D} = \{w_1, \dots, w_c\}$ is the set of distinct sub-words from LZ parsing of the binned sequence s	Calculate a complexity estimate based on the Lempel-Ziv compression algorithm.

Variability and Threshold-Based Features

These features describe how much the signal fluctuates and how often it crosses certain amplitude levels. Since different targets produce echoes with different levels of variability.

Table 4.6: Variability and threshold-based features.

Feature & Definition	Description
large_standard_deviation__r_0.02/0.1 $\sigma > r \cdot (\max(x) - \min(x))$, $r = 0.05$	Whether standard deviation exceeds r of the value range
ratio_beyond_r_sigma__r_0.5/5/6/7/10 $\frac{1}{n} \{i \mid x_i - \bar{x} > r \cdot \sigma(x)\} $	Ratio of values that are more than $r \cdot \text{std}(x)$ (so r times sigma) away from the mean of x .
number_crossing_m_m_1 $\sum_{i=1}^{n-1} \mathbb{1}_{\{(x_i > m) \neq (x_{i+1} > m)\}}$	Number of times the signal crosses the level -1

Autocorrelation

These features measure how similar the signal is to a delayed version of itself, revealing repeating patterns or periodic structure in the echo. Since different targets can produce echoes with different internal repetition patterns.

Table 4.7: Autocorrelation-based features.

Feature & Definition	Description
autocorrelation__lag_0 $\rho(\ell) = \frac{1}{(n-\ell)\sigma^2} \sum_{i=1}^{n-\ell} (x_i - \mu)(x_{i+\ell} - \mu)$	Correlation of the signal with itself, shifted by the given lag
agg_autocorrelation__f_agg_mean/var__maxlag_40 $\text{agg_autocorrelation}(x) = f_{\text{agg}}(R(1), \dots, R(m)),$ where $m = \max(n, \text{maxlag})$ and $R(l) = \frac{1}{(n-l)\sigma^2} \sum_{t=1}^{n-l} (x_t - \mu)(x_{t+l} - \mu)$	Calculates the value of an aggregation function f_{agg} (e.g. the variance or the mean) over the autocorrelation $R(l)$ for different lags
partial_autocorrelation__lag_2-9 $\alpha_k = \frac{\text{Cov}(x_t, x_{t-k} \mid x_{t-1}, \dots, x_{t-k+1})}{\sqrt{\text{Var}(x_t \mid x_{t-1}, \dots, x_{t-k+1}) \text{Var}(x_{t-k} \mid x_{t-1}, \dots, x_{t-k+1})}}$	Partial autocorrelation (direct correlation at each lag, controlling for shorter lags)

Peaks

These features describe the number and position of prominent peaks in the signal, reflecting how the target's energy is distributed over time.

Table 4.8: Peak-related features.

Feature & Definition	Description
number_cwt_peaks__n_1/5 $\text{number_cwt_peaks}(x, n) = \{p \mid p \text{ is a peak detected in } (x * \psi_a)_{a=1, \dots, n} \text{ across sufficient widths and SNR}\} $	Number of peaks detected via continuous wavelet transform

Continued on next page

Table 4.8 – continued from previous page

Feature & Definition	Description
<code>number_peaks__n_1/3/10/50</code> $\sum_{i=n+1}^{N-n} \prod_{j=1}^n \mathbb{I}_{\{x_i > x_{i-j}\}} \cdot \mathbb{I}_{\{x_i > x_{i+j}\}}$	Number of peaks with a given minimum support width
<code>index_mass_quantile__q_0.1</code> $\frac{1}{n} \min \left\{ i : \sum_{k \leq i} x_k \geq q \sum_k x_k \right\}$	Calculates the relative index i of time series x where q% of the mass of x lies left of i.

Wavelet Coefficients

These features capture how the signal’s frequency content changes over time, at different scales. Since different targets can produce echoes with different time-frequency patterns.

Table 4.9: Continuous wavelet transform (CWT) coefficient features.

Feature & Definition	Description
<code>cwt_coefficients__coeff_0..13__w_2/5/10__widths_(2,5,10,20)</code> $\text{cwt_coefficients}(x, w, \text{coeff}) = (x * \psi_w)[\text{coeff}] = \sum_{i=1}^n x_i \cdot \psi(\text{coeff} - i, w),$ $\text{where } \psi(t, a) = \frac{2}{\sqrt{3a} \pi^{1/4}} \left(1 - \frac{t^2}{a^2} \right) \exp \left(-\frac{t^2}{2a^2} \right)$	Calculates a Continuous wavelet transform for the Ricker wavelet, also known as the “Mexican hat wavelet”, used for Wavelet coefficient 0 at scales 2, 5, 10 and for Wavelet coefficients 3–13 at scale 2

Autoregressive Model

Table 4.10: Autoregressive model features.

Feature & Definition	Description
<code>ar_coefficient__coeff_6/7/8__k_10</code> $\phi_j \text{ from } x_i = \sum_{j=1}^{10} \phi_j x_{i-j} + \varepsilon_i$	This feature calculator fits the unconditional maximum likelihood of an autoregressive AR(k) process.

Change Quantiles

These features describe how much the signal changes between consecutive points, focused on specific amplitude ranges.

Table 4.11: Change quantile features.

Feature & Definition	Description
<code>change_quantiles__f_agg_mean__isabs_False__qh_*__ql_*</code> $\text{mean}\{x_{i+1} - x_i \mid x_i, x_{i+1} \in [Q_{q_l}(x), Q_{q_h}(x)]\}$	Average change between consecutive points, restricted to a given value-quantile band

Frequency Domain — FFT Coefficients

These features break the signal down into its individual frequency components, showing how much energy is present at each frequency and how the phases align.

Table 4.12: Frequency-domain (FFT) features.

Feature & Definition	Description
<code>fft_coefficient__attr_imag__coeff_1-32</code> $\text{Im}(X_k)$	Imaginary part of FFT coefficients 1–5
<code>fft_coefficient__attr_angle__coeff_0-99</code> $\text{arg}(X_k)$	Phase angle of FFT coefficients 1/4–99 (nearly full spectrum)

Trend and Regression

All features in this section are based on an **Ordinary Least Squares (OLS)** linear fit, which finds the line $y = a + bt$ that minimizes the sum of squared residuals between the fitted line and the data.

For `agg_linear_trend`, the series is first split into non-overlapping chunks of length L (`chunk_len`), and each chunk is reduced to a single aggregated value c_j using the aggregation function f_{agg} (mean, max, min, or median):

$$c_j = f_{\text{agg}}(x_{(j-1)L+1}, \dots, x_{jL}), \quad j = 1, \dots, \lceil n/L \rceil$$

An OLS line $c_j = a + bj + \varepsilon_j$ is then fit on these aggregated chunk values versus the chunk index j , and one attribute of that single fit (a , b , r , p -value, or standard error) is returned as the feature value.

Table 4.13: Trend and regression-based features.

Feature & Definition	Description
<code>linear_trend__attr_pvalue</code> p -value of the OLS fit $x_i = a + bi + \varepsilon_i$	Significance of the linear trend over the whole series
<code>linear_trend__attr_rvalue</code> r -value of the same OLS fit	Correlation coefficient of the linear trend over the whole series
<code>agg_linear_trend__attr_rvalue__chunk_len_5/10/50__f_agg_mean</code> r -value of the OLS fit on c_j , with $c_j = \text{mean}(\text{chunk}_j)$	Correlation coefficient of the trend fit on chunk-mean values
<code>agg_linear_trend__attr_intercept__chunk_len_5/10/50__f_agg_mean</code> Intercept a of the OLS fit on c_j , with $c_j = \text{mean}(\text{chunk}_j)$	Intercept of the trend fit on chunk-mean values
<code>agg_linear_trend__attr_slope__chunk_len_5__f_agg_max</code> Slope b of the OLS fit on c_j , with $c_j = \text{max}(\text{chunk}_j)$	Slope of the trend fit on chunk-maximum values
<code>agg_linear_trend__attr_slope__chunk_len_5__f_agg_var</code> Slope b of the OLS fit on c_j , with $c_j = \text{var}(\text{chunk}_j)$	Slope of the trend fit on chunk-variance values

Stationarity Test

All features in this section are derived from the Augmented Dickey-Fuller (ADF) regression:

$$\Delta x_i = \alpha + \gamma x_{i-1} + \sum_{j=1}^p \delta_j \Delta x_{i-j} + \varepsilon_i$$

fitted by OLS, where $\Delta x_i = x_i - x_{i-1}$, p is the lag order (selected automatically via the `autolag` criterion, e.g. AIC), and γ is the coefficient tested against the null hypothesis $H_0 : \gamma = 0$ (unit root / non-stationary series).

Table 4.14: Stationarity test features.

Feature & Definition	Description
augmented_dickey_fuller__attr_teststat__autolag_AIC $\text{teststat} = \frac{\hat{\gamma}}{\text{SE}(\hat{\gamma})}$	ADF test statistic for $H_0 : \gamma = 0$, indicating presence of a unit root
augmented_dickey_fuller__attr_usedlag__autolag_AIC p , the lag order selected by the autolag criterion	Number of lagged difference terms used in the ADF regression

Energy Distribution

The `energy_ratio_by_chunks` feature splits the series $x = (x_1, \dots, x_n)$ into N contiguous chunks (`num_segments`) using `numpy.array_split`: if n is not divisible by N , the first $n \bmod N$ chunks receive one extra point each, so chunk sizes differ by at most 1. Let chunk_j denote the j -th chunk (0-indexed, `segment_focus`), for $j = 0, \dots, N - 1$.

Table 4.15: Energy distribution features.

Feature & Definition	Description
energy_ratio_by_chunks__num_segments_N__ segment_focus_j $\frac{\sum_{i \in \text{chunk}_j} x_i^2}{\sum_{i=1}^n x_i^2}$	Fraction of total signal energy contained in chunk j out of N (possibly unequal) contiguous segments

Entropy

This feature measures how spread out or concentrated the signal's frequency content is. A signal with energy focused in a few frequencies has low entropy, while one with energy spread across many frequencies has high entropy, which can help distinguish targets with different scattering behavior.

Table 4.16: Entropy-based features.

Feature & Definition	Description
<code>fourier_entropy__bins_2</code> $-\sum_{k=1}^{\text{bins}} p_k \log(p_k) \cdot \mathbb{1}_{\{p_k > 0\}}, \quad p_k \text{ from binning } \tilde{P}_{xx} = \frac{P_{xx}}{\max(P_{xx})}$	Calculate the binned entropy of the power spectral density of the time series (using the welch method).

Symmetry

This feature indicates whether the distribution of signal values is symmetric around its mean, by comparing the mean and median relative to the overall value range. Since mine-like and clutter targets can produce echoes with different degrees of amplitude symmetry, this can serve as a simple discriminative cue.

Table 4.17: Symmetry-based features.

Feature & Definition	Description
<code>symmetry_looking__r_0.05/0.1</code> $ \mu - \text{median}(x) < r \cdot (\max(x) - \min(x)), \quad r \in \{0.05, 0.1\}$	Boolean indicator of whether the distribution of x looks symmetric, i.e. whether the absolute difference between the mean and the median is smaller than r times the range of the values

Abstract

Naval mines and unexploded ordnance remain a persistent threat to maritime operations, making reliable automatic classification a key priority for mine countermeasures. This work develops and evaluates a complete, interpretable machine learning pipeline for the classification of mine-like objects from low-frequency synthetic aperture sonar (SAS) signals, using real, multi-target acoustic data from the TREX13 campaign. Focused target signals are recovered via the $-k$ algorithm and converted into tsFresh-derived time-series features, cleaned and selected under a target-grouped, nested cross-validation protocol designed to prevent data leakage between correlated signals. Six classifiers, spanning linear, kernel-based, and tree-based approaches, are compared using a cost-sensitive F_2 objective reflecting the operational asymmetry between missed detections and false alarms. Non-linear classifiers consistently outperform the linear baseline, achieving recall values between 0.73 and 0.77, although performance is shown to depend more strongly on which physical targets are evaluated than on classifier choice. SHAP-based interpretability analysis reveals that misclassifications stem from genuine acoustic similarities between certain slender clutter objects and mine-like shapes, grounding the model's decisions in physically meaningful signal properties. These findings demonstrate that transparency and operational performance can be achieved jointly in this domain, while highlighting data diversity and feature representation as key directions for further improvement.

Keywords: Synthetic Aperture Sonar, Mine-Like Object Classification, Low-Frequency Sonar, Machine Learning, Interpretability, SHAP, tsFresh, TREX13, Mine Countermeasures

