Privacy-Preserving Feature Engineering Framework for Real-World Vulnerability Risk Prediction

Jawharah Albarakati, Tawfiq Hasanin, Suaad Alarif
Department of Information Systems, King Abdulaziz University, Jeddah, Saudi Arabia

***Abstract*—This study introduces a privacy-preserving data preparation and feature engineering framework designed for machine learning-based cybersecurity vulnerability risk prediction, utilizing real-world enterprise scan data from an operational cloud environment. To transform raw vulnerability records into a structured, machine-readable format while complying with personal data protection regulations, the framework integrates systematic data cleaning, missing value imputation, categorical encoding, text normalization, and host pseudonymization. Exploratory data analysis (EDA) was conducted on 36,940 operational records to examine dataset characteristics, vulnerability distributions, and severe class imbalances. Multiple classifiers—including Logistic Regression, Random Forest, XGBoost, and LightGBM—were evaluated under extreme imbalance conditions mitigated via SMOTE. Experimental results demonstrate that non-linear tree-based ensembles achieved the highest predictive performance, with LightGBM attaining a macro F1-score of 0.9974. Comparative analysis further indicates that the effectiveness of textual representations depends on the underlying classifier architecture, with semantic embeddings yielding the best performance when combined with tree-based ensemble models, while TF-IDF remained competitive for linear classificationThese findings underscore the critical role of structured text preparation and semantic representation in enhancing model reliability, interpretability, and scalability for automated vulnerability prioritization.**

***Keywords*—Data pre-processing; machine learning; protecting sensitive data; stemming; pseudonymization**

I. Introduction

Cybersecurity has become a critical concern for organizations due to the increasing frequency and sophistication of cyber threats. Vulnerability assessment plays a central role in identifying and prioritizing security weaknesses across enterprise infrastructures, including networks, servers, applications, and cloud environments.

The effectiveness of machine learning–based vulnerability management depends heavily on the quality of the underlying data. However, cybersecurity logs are often noisy, incomplete, and heterogeneous, requiring preprocessing techniques such as missing-value handling, normalization, categorical encoding, and text feature extraction. These steps improve data quality and transform raw telemetry into representations suitable for predictive risk analysis.

In this study, operational security data was collected using the Nessus Vulnerability Scanner, a widely adopted industry-standard solution for identifying system vulnerabilities [1]. Nessus facilitates comprehensive infrastructure auditing across cloud environments, producing highly detailed vulnerability records. To leverage this data safely, raw scan reports are systematically processed to eliminate operational duplicates, validate network configurations, and mask sensitive host telemetry. This ensures strict adherence to data privacy standards and compliance with personal data protection regulations without degrading the predictive utility of the features.

Traditional vulnerability prioritization approaches often rely on static scoring methods or keyword-based techniques that may fail to capture the semantic information contained in vulnerability descriptions. To address this limitation, this study develops a privacy-preserving feature engineering framework that integrates structured vulnerability attributes with textual representations derived from both TF-IDF and Sentence-BERT (SBERT). In addition, exploratory data analysis (EDA) is conducted to examine data characteristics, feature distributions, and class imbalance patterns before model development.

The main contributions of this study are as follows:

- A privacy-preserving data preparation and feature engineering framework for real-world enterprise vulnerability telemetry, incorporating host pseudonymization to support compliance with Personal Data Protection Law (PDPL) requirements.
- A comparative evaluation of lexical (TF-IDF) and semantic (SBERT) text representations for encoding unstructured vulnerability descriptions.
- An empirical assessment of four machine learning models (Logistic Regression, Random Forest, XGBoost, and LightGBM) under severe class imbalance conditions using SMOTE-based mitigation.
- An analysis of the impact of semantic representations on vulnerability severity prediction performance in operational enterprise environments.

II. Related Work

Vulnerability assessment is a cornerstone of proactive cybersecurity risk management, aimed at identifying, analyzing, and prioritizing system flaws. Traditional vulnerability management heavily relies on static rule-based scanning tools and standardized frameworks, such as the Common Vulnerability Scoring System (CVSS) [2][3]. While efficient for cataloging known security flaws, these traditional methods lack adaptability and fail to capture the dynamic, contextual properties of real-world enterprise vulnerability telemetry [4].

To overcome these limitations, recent research has shifted toward supervised machine learning algorithms to enhance predictive risk prioritization. For instance, Sabottke et al. utilized machine learning to predict vulnerability exploitability

by leveraging language attributes from vulnerability descriptions [5], while Neuhaus et al. demonstrated that supervised learning models can effectively prioritize vulnerabilities based on historical exploitation trends [6]. Similarly, recent advancements have focused on exploitability prediction and automated severity forecasting using high-dimensional datasets [7], [8]. However, a persistent limitation in these approaches is their reliance on clean, static, and balanced batch-learning benchmarks, which rarely reflect the chaotic nature of real-world, live operational environments.

The predictive performance of these machine learning frameworks is fundamentally constrained by the quality and representation of the underlying data. Raw vulnerability logs are notoriously noisy, unstructured, and heterogeneous, demanding extensive preprocessing [9][10]. Koukaras and Tjortjis emphasized that rigorous feature engineering and data preparation are critical prerequisites for meaningful data mining and analysis [9]. In the context of textual cybersecurity data, standard Natural Language Processing (NLP) techniques—such as tokenization, stemming, and Term Frequency-Inverse Document Frequency (TF-IDF) weighting—have been widely adopted to extract patterns from unstructured vulnerability descriptions [11], [12]. While integrating these lexical features with structured attributes has been shown to improve accuracy [13], traditional keyword-matching techniques like TF-IDF fail to capture the deeper semantic nuances of security vulnerabilities. Although dense semantic representations (e.g., Transformer-based embeddings) have revolutionized general NLP, their systematic evaluation against traditional lexical features remains under-explored in operational vulnerability frameworks.

Concurrently, compliance with data privacy mandates has emerged as a critical requirement in cybersecurity analytics. Implementing machine learning on live enterprise telemetry introduces severe privacy risks, as scanner logs often contain sensitive network topologies and host identifiers. To mitigate this, various privacy-preserving preprocessing strategies, such as data masking and pseudonymization, have been proposed to safeguard identities while preserving analytical utility [14], [15]. Truong et al. and Riabi et al. demonstrated that secure pseudonymization strategies can satisfy regulatory constraints (such as the GDPR or PDPL) without degrading downstream classifier accuracy [16], [17].

Despite these advancements, existing literature often treats data preprocessing, privacy preservation, text representation, and class imbalance mitigation as isolated problems. Many studies evaluate models under idealized, balanced, or strictly static conditions, overlooking the severe class imbalances inherent to operational environments [18].

Unlike prior works that focus solely on standalone classification performance or isolated exploit prediction, this study introduces a unified, systematically integrated framework. We combine regulatory-compliant privacy preservation, hybrid structured-textual feature engineering (lexical vs. deep semantic embeddings), and robust class imbalance mitigation using SMOTE. The contribution of this work lies not in proposing a novel standalone classifier, but in empirically demonstrating how a meticulously engineered data preparation and semantic feature construction pipeline can maximize operational risk classification fidelity using heterogeneous enterprise cybersecurity data (Fig. 1).

III. Research Methodology

The proposed research methodology consists of five sequential phases. First, vulnerability data were collected from a real-world enterprise cloud environment using the Nessus vulnerability scanner. Second, data preprocessing and privacy-preserving transformations were applied. Third, feature engineering was performed. Fourth, multiple supervised machine learning models were trained, while class imbalance was addressed using SMOTE and class-weighted learning strategies. Finally, a comparative experimental evaluation was conducted to assess model performance using several classification metrics across multiple vulnerability risk categories.

A. Data Collection and Acquisition

The foundational phase of establishing a robust machine learning-based vulnerability assessment framework begins with systematic data acquisition. The empirical data utilized in this study was sourced from the Nessus Vulnerability Scanner, an industry-standard network security auditing tool. Nessus was deployed across a production cloud-based enterprise infrastructure to continuously audit and discover security flaws across heterogeneous enterprise domains, including network devices, internal servers, and organizational endpoints.

The collected telemetry comprises a rich mix of structured security indicators and unstructured textual attributes. Specifically, the captured dataset includes critical variables such as vulnerability severity levels, unique host identifiers, textual remediation solutions, Common Vulnerabilities and Exposures (CVE) identifiers, and binary indicators of system exploitability. This multifaceted data collection ensures that both the structural context and the natural language nuances of the vulnerabilities are preserved for downstream analysis.

To reflect authentic operational challenges, the vulnerability logs were harvested from a live cloud environment operating within the Kingdom of Saudi Arabia (KSA). Due to strict organizational security and confidentiality mandates, the identity of the target enterprise is anonymized. Crucially, all data acquisition workflows were conducted in strict compliance with internal corporate security policies and PDPL, establishing a legally compliant and privacy-preserving data foundation without compromising the analytical utility of the security logs.

B. Privacy-Preserving Data Preprocessing

Data preprocessing is an essential step in machine learning because it improves data quality and ensures that features are represented consistently for model training [4]. Since cybersecurity datasets often contain heterogeneous data types, missing values, and noisy observations, preprocessing techniques are required to prepare the data for reliable predictive modeling [19].

In addition to data quality considerations, protecting sensitive organizational information is a critical requirement in cybersecurity analytics [16]. To address this requirement, the

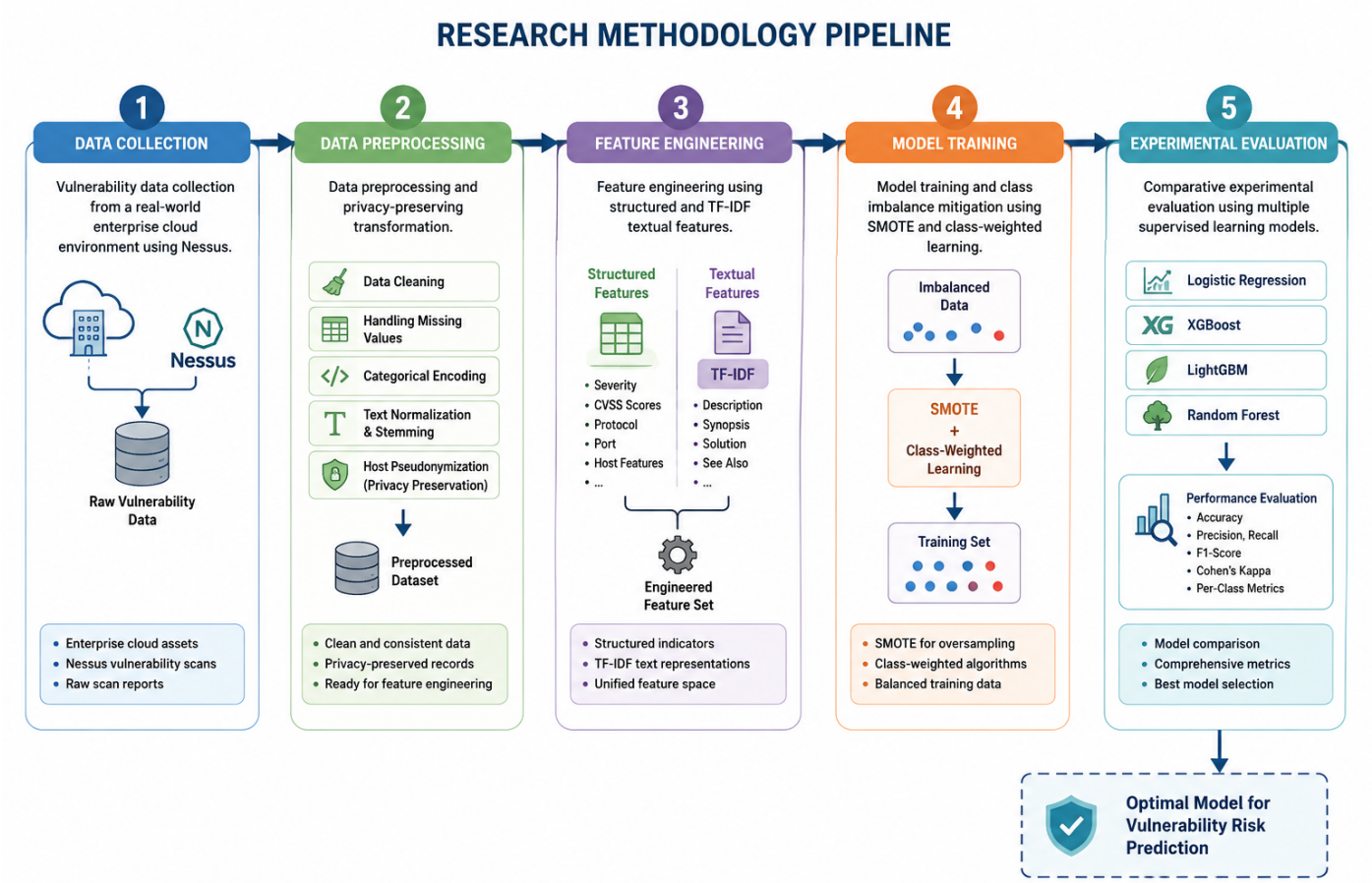

Fig. 1. Proposed privacy-preserving machine learning pipeline for cybersecurity vulnerability risk prediction.

proposed framework applies host pseudonymization, replacing identifiable infrastructure information with deterministic pseudonyms while preserving the utility of the data for subsequent machine learning tasks [17].

1) Host pseudonymization and privacy preservation: The operational telemetry harvested during the data acquisition phase contained explicit network configurations within the raw `Host` attribute, exposing corporate IP footprints and introducing an unacceptable vector for information leakage. To neutralize this vulnerability and ensure strict compliance with the Saudi Arabian Personal Data Protection Law (PDPL) mandated in the region, a deterministic pseudonymization mapping was integrated into the preprocessing pipeline.

Each raw infrastructure IP address was passed through a secure Secure Hash Algorithm 256-bit (SHA-256) cryptographic function. To minimize storage footprints and reduce data dimensionality, the resulting 64-character hexadecimal strings were systematically truncated to their leading 10 characters. The resulting hashes were truncated to the first 10 hexadecimal characters to reduce storage requirements while maintaining unique host identifiers within the dataset, thereby preserving the categorical utility of host-specific attributes for the downstream tree-based classifiers. Table I illustrate a comparative sample of the enterprise logs before and after the application of pseudonymization layer.

TABLE I. SAMPLE RECORDS BEFORE AND AFTER HOST PSEUDONYMIZATION

Idx	Severity	Host
Original Dataset		
0	Critical	12.0.2.34
1	Critical	12.0.2.24
2	Medium	18.51.100.34
3	High	23.0.113.34
4	Low	23.0.113.44
Pseudonymized Dataset		
0	Critical	43741ff074
1	Critical	32d4a72597
2	Medium	6046c9aa49
3	High	576264d7a0
4	Low	9ab7244c54

2) Missing value analysis and feature elimination: Operational cyber threat logs frequently suffer from data incompleteness due to sensor telemetry dropouts, intermittent database connection syncs, or unpopulated third-party vendor fields. In this pipeline, missing values within remaining feature channels were preserved strictly as mathematical null states (NaN)

TABLE II. COMPARATIVE ANALYSIS OF TEXTUAL ATTRIBUTES BEFORE AND AFTER STEMMING

Attribute	Original Text	Stemmed Text
Description	According to its version, there is at least one...	accord to version, there is at least one...
Synopsis	The remote host contains one or more unsupport...	remot host contain one or more unsupport...
Solution	Upgrade to a version that is currently supported.	upgrad to version current support.

rather than being replaced by generic textual placeholders. This programmatic constraint ensures that downstream natural language processing modules do not optimize over artificial tokens, preventing representation leakage.

To safeguard predictive reliability, a strict data-quality threshold was established: any attribute column exhibiting an empirical missingness rate exceeding 90% was systematically eliminated from the global feature space. Under this rigorous constraint, two major structural attributes were pruned: the `CANVAS` indicator matrix (exhibiting approximately 97.75% missingness) and the `Metasploit` exploitability matrix (exhibiting approximately 91.10% missingness). Removing these hyper-sparse features neutralizes high-variance noise, preventing downstream classifiers from expanding optimization parameters over systematically unpopulated data fields.

3) Categorical severity label encoding: To transform the ordinal, textual target categories within the vulnerability corpus into actionable mathematical formats compliance with gradient descent optimization loops, an ordinal Label Encoding transformation was applied. The original risk assessment categories—spanning from baseline configurations to critical exposures—were mapped onto a standardized discrete numerical scale: None $\rightarrow$ 0, Low $\rightarrow$ 1, Medium $\rightarrow$ 2, High $\rightarrow$ 3, and Critical $\rightarrow$ 4.

4) Text normalization and stemming: Column normalization involved the removal of extraneous whitespaces, special characters, and newline symbols from column names to ensure consistent and standardized naming conventions (e.g., converting " Risk " to "Risk"). Text normalization included converting all text-based fields to string data types, trimming leading and trailing spaces, and standardizing internal spacing for uniformity. Additionally, empty string values "") were replaced with NaN to facilitate accurate handling of missing data during subsequent analysis. Text preprocessing included systematic stemming of the *Description*, *Synopsis*, and *Solution* fields using the Porter Stemmer algorithm to reduce lexical variability and optimize token consistency. This linguistic normalization was applied uniformly prior to feature extraction, mapping inflectional variations to their respective canonical root forms. Table II showcases a comparative sample of the textual attributes before and after executing the stemming routine.

C. Data Labeling and Target Grounding

In supervised machine learning frameworks for cybersecurity analytics, establishing ground-truth annotations is a foundational step for training predictive classifiers. Rather than relying on manual or crowd-sourced labeling, which introduces subjective bias and operational delays, this framework leverages automated, deterministic vulnerability annotations directly derived from the Nessus vulnerability scanner telemetry. This methodology guarantees consistent target grounding based on industry-standard security metrics.

TABLE III. UNIFIED OPERATIONAL SCHEMA MAPPING SEVERITY TARGETS, HOST ATTRIBUTES, UNSTRUCTURED CONTEXTS, AND STANDARDIZED CVE IDENTIFIERS

Risk	Host	Synopsis	Solution	CVE
Critical	192.0.2.34	Unsupported software versions detected.	Upgrade to a supported version.	CVE-2020-6457
Critical	192.0.2.34	Microsoft Silverlight is installed.	Remove Silverlight from the host.	CVE-2020-6461
Critical	192.0.2.55	Installed browser contains vulnerabilities.	Upgrade Chrome to v81.0.4044.113+.	CVE-2020-6831
Low	192.0.2.78	Browser configuration issues detected.	Upgrade Chrome to v81.0.4044.129+.	CVE-2008-5161
Medium	192.0.2.99	Distributed framework is vulnerable.	Upgrade Chrome to v81.0.4044.138+.	CVE-2023-20867

The dataset utilizes a multi-attribute labeling scheme to facilitate distinct vulnerability categorization tasks. The features chosen to define the target classes include:

- Risk Severity Level: Sourced from the `Risk` attribute, this categorical variable acts as the primary multi-class classification target. It stratifies vulnerabilities into standardized threat envelopes (*None*, *Low*, *Medium*, *High*, and *Critical*), allowing machine learning models to map structured and textual telemetry to prospective severity levels.
- Standardized Vulnerability Identifiers: Captured within the `CVE` attribute, these labels map instances to the Common Vulnerabilities and Exposures (CVE) database. This identifier provides a deterministic reference for validating tracking behaviors and checking model generalization across disjoint vulnerability sets.

To ensure high data fidelity before model ingestion, an automated extraction parser mapped the Nessus security scanner outputs to distinct class definitions. Subsequently, a programmatic cross-verification audit was executed to ensure alignment between the assigned severity labels and their respective CVSS scores. This process eliminated any noise introduced by scanner version mismatches.

To illustrate the underlying structural schema and demonstrate how target severity classes, host attributes, and unstructured textual features are aligned seamlessly within a single record space, Table III provides a unified, de-duplicated overview of representative operational data samples across the corpus.

D. Hybrid Feature Engineering Pipeline: Lexical vs. Deep Semantic Paradigms

To translate the raw, heterogeneous vulnerability scanner logs into unified tensor formats optimal for supervised machine learning, a robust, dual-methodological feature engineering pipeline was engineered. Cybersecurity telemetry presents a distinct architectural challenge: it couples structured categorization metrics (such as encoded risk and binary flags) with high-dimensional, unstructured linguistic blocks found within the `Synopsis`, `Description`, and `Solution` fields. To

capture both explicit keyword distributions and implicit contextual relationships, this framework constructs an augmented feature space by executing and comparing two distinct Natural Language Processing (NLP) paradigms: traditional lexical frequency mapping and deep transformer-based semantic vectorization.

1) Lexical representation via TF-IDF mappings: The baseline textual feature space is established using the statistical Term Frequency–Inverse Document Frequency (TF-IDF) vectorization paradigm [20]. This lexical scheme scores discrete unigram tokens based on their localized occurrence frequency within an individual log entry relative to their global distribution across the entire enterprise corpus, scaling down ubiquitous, low-variance terms while highlighting rare, security-specific keywords. Mathematically, the weight allocated to a distinct token t within an individual vulnerability record d belonging to the broader global corpus D is formulated via smooth inverse document frequency as follows:

$$\text{TF-IDF}(t,d,D) = \text{TF}(t,d) \times \log\left(\frac{1+|D|}{1+|\{d \in D : t \in d\}|}\right) + 1 \quad (1)$$

where, $TF(t,d)$ represents the raw term frequency of token t in document d, and the denominator computes the inverse document frequency with additive smoothing to prevent division-by-zero. Within this pipeline, textual records were represented using a unigram configuration (`ngram_range=(1,1)`), while the TF–IDF vocabulary size was limited to 500 features (`max_features=500`) to balance representational capacity, computational efficiency, and model complexity. This dimensionality was selected as a practical trade-off to reduce feature sparsity while preserving the most informative vulnerability-related terms across the 36,940 operational records. Although computationally efficient and highly transparent for feature importance analysis, this lexical representation operates under the bag-of-words assumption, ignoring word order, syntactic structure, and contextual semantic relationships between terms.

2) Deep semantic embeddings via Sentence-BERT (SBERT): To resolve the structural limitations of keyword matching and capture context-aware semantic nuances without suffering from vocabulary sparsity, a deep semantic vectorization pipeline was engineered utilizing a Sentence-BERT (SBERT) architecture [9]. SBERT modifies standard pre-trained BERT networks by deploying siamese and triplet network structures to generate highly descriptive, fixed-length dense vector representations for full sentence and paragraph blocks. For this framework, the optimized, pre-trained `all-MiniLM-L6-v2` transformer model was deployed.

When the concatenated textual attributes of a vulnerability report are ingested, the transformer processes the entire textual sequence holistically, mapping its core syntactic context into a compressed 384-dimensional dense continuous numerical vector space. Unlike traditional lexical distributions, this allows the framework to inherently capture structural and operational synonymy within cybersecurity language. Consequently, phrases describing identical security implications (e.g., "*flaw allows remote compromise*" and "*vulnerability permits external exploitation*") are mapped to highly proximal vector coordinates within the continuous manifold, even when their underlying vocabularies are entirely disjoint.

TABLE IV. QUANTITATIVE DISTRIBUTION OF VULNERABILITY SEVERITY LEVELS REVEALING SEVERE OPERATIONAL CLASS IMBALANCE

Vulnerability Severity Level	Absolute Record Count	Relative Proportion (%)
Critical	16,703	45.22%
High	124	0.34%
Medium	2,358	6.38%
Low	9,295	25.16%
None	8,460	22.90%
Total Corpus Size	**36,940**	**100.00%**

IV. EXPERIMENTAL DESIGN AND LEARNING SETUP

This section details the empirical environment, exploratory data profiling, and laboratory configurations deployed to validate the predictive performance of the proposed framework. To guarantee rigorous experimental reproducibility, the processing and learning pipelines were constructed using a standardized software ecosystem based on Python 3.10. The architecture integrates `scikit-learn` for core mathematical utilities, `imbalanced-learn` for syntactic data augmentation, and highly optimized implementations of `LightGBM` and `XGBoost`. Computational workloads were compiled on an Intel Core i7 workstation equipped with 16 GB of RAM and hardware acceleration primitives to minimize dense vector generation latencies during transformer inference.

A. Exploratory Data Analysis (EDA)

Rather than executing modeling processes over unverified logs, a systematic Exploratory Data Analysis (EDA) was performed to deconstruct the statistical topology of the ingested telemetry [21], [22]. The experimental dataset encapsulates a total of 36,940 operational cybersecurity records extracted from a production corporate cloud network. Structurally, each telemetry instance is defined across a 10-dimensional attribute space comprising categorical system indicators (`Risk`, `Host`, `Name`) alongside dense text fields (`Synopsis`, `Description`, `Solution`, `See Also`, `Plugin Output`, `CVE`, and `Core Impact`).

The dataset captures a heterogeneous technological ecosystem, tracking active vulnerabilities across varied operating systems (e.g., Windows, Unix, ESXi), enterprise software configurations (e.g., WinSCP, WinRAR, Google Chrome), and infrastructure security protocols (e.g., SSL, SSH, TLS). This diverse composition provides a realistic operational foundation for training models capable of generalizing across varied real-world network deployment environments.

B. Vulnerability Severity Profiles and Empirical Statistics

A key investigative focus of the EDA phase involved analyzing the empirical distribution of the target classification variable, `Risk`. The enterprise vulnerabilities are stratified into five operational severity domains: *Critical*, *High*, *Medium*, *Low*, and *None*. Table IV provides a rigorous quantitative breakdown of these target severity allocations across the entire 36,940-record corpus.

TABLE V. EMPIRICAL RANKING OF THE TOP 10 MOST FREQUENT INFRASTRUCTURE VULNERABILITIES

Identified Vulnerability Signature Name	Occurrences
DCE Services Enumeration	1,493
Netstat Portscanner (WMI)	1,242
Microsoft Windows Remote Listeners Enumeration (WMI)	1,215
KB5028168: Security Update (July 2023)	1,128
KB5036896: Security Update (April 2024)	1,027
KB5031361: Security Update (October 2023)	936
KB5025229: Security Update (April 2023)	828
Nessus SYN Scanner	774
Service Detection	631
KB5023702: Security Update (March 2023)	624

TABLE VI. LINGUISTIC DISTRIBUTION, LENGTH VARIANCE, AND DATA COMPLETENESS OF TEXTUAL FEATURES

Textual Field	Min	Max	Mean	Median	NaN	Missing (%)
`Name`	11	149	56.4	54.0	0	0.00%
`Synopsis`	25	159	64.3	61.0	0	0.00%
`Description`	27	23,526	1,127.2	451.0	0	0.00%
`Solution`	3	833	36.9	29.0	13,571	36.74%
`See Also`	3	1,916	74.5	42.0	11,635	31.50%
`Plugin Output`	3	32,766	628.9	207.0	564	1.53%
`CVE`	3	16	8.5	13.0	17,932	48.54%

The empirical breakdown in Table IV exposes a highly non-uniform distribution that presents an acute class-imbalance profile. High-severity threats heavily dominate the data footprint, wherein the *Critical* class alone comprises 16,703 records, representing 45.22% of the dataset. When cumulatively aggregated with the *Low* severity category (25.16%), these two non-contiguous operational bands account for exactly 70.38% of the entire data space, representing a collective volume of 25,998 instances. Conversely, the *High* severity class represents an extreme minority boundary anomaly, containing only 124 records (0.34%). This localized scarcity poses a substantial risk of gradient starvation during model training, as standard loss functions would favor optimizing for the majority classes while neglecting rarer flags. This finding provides clear empirical justification for incorporating host-disjoint stratified partitioning and minority oversampling algorithms in subsequent training phases.

To contextualize the behavioral characteristics of the dataset, Table V isolates the top 10 most frequent vulnerability signatures by operational name. The distribution is dominated by system discovery and patch telemetry, led by *DCE Services Enumeration* (1,493 instances) and *Netstat Portscanner* (1,242 instances), alongside recurring Microsoft Windows cumulative security update sequences (e.g., KB5028168 and KB5036896).

C. Granular Textual Feature Statistical Distribution

Because the proposed framework heavily relies on extracting hybrid representations from unstructured text, a detailed length and variance analysis was executed across the seven primary narrative attributes: `Name`, `Synopsis`, `Description`, `Solution`, `See Also`, `Plugin Output`, and `CVE`. Table VI provides a comprehensive statistical summary of character length behaviors and data completeness metrics across these linguistic dimensions.

The empirical character profiles in Table VI demonstrate significant structural heterogeneity. The `Name` and `Synopsis` segments exhibit short, highly constrained boundaries, averaging 56.4 and 64.3 characters, with optimal data completeness (0% missing entries). Conversely, the `Description` attribute contains the densest contextual data, showing immense variance with sequence lengths scaling up to 23,526 characters and a substantial mean length of 1,127.2. This confirms that descriptions possess high semantic richness, making deep contextual embeddings via Sentence-BERT crucial to capture long-range technical dependencies.

Furthermore, critical systemic fields like `Solution` and `See Also` introduce sparse data spaces, exhibiting 13,571 (36.74%) and 11,635 (31.50%) missing values, respectively. The most severe sparsity occurs within the `CVE` column, where 17,932 instances lack standardized identifiers (48.54% missing rate). This high missing rate underscores the limitation of relying solely on external database cross-referencing for risk prediction. It highlights the absolute necessity of our text-driven approach, which can predict severity directly from scanner logs even when standard CVE references are missing.

D. Strategic Data Partitioning and Class Imbalance Optimization

To validate the proposed predictive pipeline under rigorous experimental bounds, the 36,940 corporate telemetry records were partitioned into a training cohort (80%) and an independent testing cohort (20%). This partitioning executed a stratified sampling design to guarantee that the empirical class proportions—extending from the dominant *Critical* severity profile to the extreme minority *High* anomaly boundary—remain structurally identical across both the training and evaluation subsets. Crucially, this partition was executed utilizing a host-disjoint boundary split, ensuring that identical target IP addresses or asset containers do not span across both partitions, thereby eliminating validation artifacts stemming from data leakage.

Given the long-tailed, severe class imbalance quantified during the exploratory data analysis, relying on standard global loss minimization would lead downstream classifiers to systematically optimize for majority classes, rendering them blind to rare, critical vulnerabilities. To mitigate this algorithmic bias, a dual-methodological optimization strategy was engineered, tailoring the class imbalance correction mechanism to the architectural requirements of each respective classifier family:

1) Cost-sensitive class-weighted learning: For linear models (Logistic Regression) and boosting frameworks dependent on objective gradient trajectories (XGBoost and LightGBM), an algorithmic class-weighted learning constraint was integrated directly into the training process. Loss computation was penalized inversely proportional to the respective class frequencies using the following penalization tensor:

$$W_c = \frac{N_{\text{total}}}{|C| \times N_c} \tag{2}$$

where, W_c represents the optimized weight allocated to class c, N_{total} is the cumulative sample count (36,940), $|C|$

TABLE VII. SUMMARY OF ENGINEERED FEATURES USED FOR SEVERITY PREDICTION

Source Field	Transformation	Type	Dim.	Rationale
`Synopsis, Description, Solution`	TF-IDF Vectorizer	Lexical	500	Maps keyword weights and explicit terms.
`Synopsis, Description, Solution`	Sentence-BERT	Semantic	384	Captures contextual and synonymous meanings.
`Host`	Categorical Encoding	Structural	1	Identifies host-specific risk patterns.
`Description Length`	Character Count	Numeric	1	Represents vulnerability complexity.
`Plugin Output Length`	Character Count	Numeric	1	Profiles scanner output verbosity.
`CVE Indicator`	Binary Mask	Flag	1	Indicates presence of CVE identifiers.
`Solution Availability`	Binary Mask	Flag	1	Represents remediation availability.
`See Also Availability`	Binary Mask	Flag	1	Indicates external references.
`Security Keywords`	Regex Matching	Binary	6	Captures high-risk security terms.
Total Structured Attributes	–	**Structured**	**13**	Unified structural feature set.

defines the number of target categories (5), and N_c is the frequency of the localized class. This adjustment forces the loss-gradient optimizations to heavily penalize minority misclassifications during backpropagation loops.

2) Synthesized Spatial Augmentation (SMOTE): For non-parametric, tree-based ensemble models such as Random Forest—which depend on orthogonal recursive features splits rather than continuous loss optimization gradients—the localized training data space was augmented using the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE computes the k-nearest neighbors ($k = 5$) of minority class feature vectors and interpolates synthetic data vectors along the line segments connecting these proximal points. This density alignment equalizes decision boundaries across non-contiguous spaces without introducing the duplicate-overfitting artifacts associated with standard random oversampling.

E. Hybrid Feature Matrix Synthesis and Operational Splitting

To support the comparative experimental design, two separate, parallel feature matrices were constructed by executing horizontal concatenation tensor operations between the engineered structured attributes and each respective textual representation.

As mathematically detailed in Table VII, the structural feature engineering pipeline isolates a cumulative baseline of 13 distinct numeric and categorical dimensions. To ensure downstream multi-class classifiers can simultaneously exploit structural system configurations and deep contextual language distributions, these independent spaces were aggregated via dynamic coordinate alignment, yielding two final parallel hybrid data matrices:

- The TF-IDF hybrid matrix ($\mathbf{X}_{\text{lexical}}$) Synthesized via the horizontal concatenation of the 500 sparse lexical dimensions with the 13 structured dimensions, resulting in a final comprehensive feature space of **513 dimensions**.
- The SBERT hybrid matrix ($\mathbf{X}_{\text{semantic}}$) Synthesized via the horizontal concatenation of the 384 dense semantic embedding dimensions with the 13 structured dimensions, resulting in a compressed final feature space of **397 dimensions**.

From an engineering standpoint, while the TF-IDF textual components produce a highly sparse matrix that requires coordinate conversion for optimization algorithms, the SBERT embedding spaces exist as fully dense continuous vectors. This mathematical density alignment eliminates the need for computational density approximations prior to executing synthetic minority oversampling (SMOTE) loops.

Following feature fusion and matrix assembly, the global input data was partitioned using an 80/20 stratified train-test split executed via a host-disjoint boundary mechanism. This operational stratification strategy guarantees that the localized proportions of all five target risk categories (*Critical*, *High*, *Medium*, *Low*, and *None*) are perfectly preserved across both the training sets and the independent validation subsets. Consequently, this alignment maintains experimental consistency and neutralizes evaluation bias despite the severe class imbalance inherently present in the production environment.

F. Predictive Model Training and Comparative Evaluation

To establish a rigorous comparative benchmark for automated vulnerability severity assessment, four distinct classification paradigms were trained and evaluated over the engineered hybrid feature matrices: a baseline Logistic Regression model, an optimized eXtreme Gradient Boosting (XGBoost) classifier, a Light Gradient Boosting Machine (LightGBM) architecture, and a Synthetic Minority Over-sampling Technique combined with Random Forest (SMOTE + RF) ensemble configuration. Performance profiles were cross-examined using multiple standard metrics to prevent optimization biases stemming from the inherent majority-class dominance within the enterprise logs.

1) Baseline performance matrix logistic regression: The experimental pipeline established its predictive baseline utilizing a generalized Multinomial Logistic Regression model optimized via the `lbfgs` solver with a maximum convergence threshold set to $1,000$ iterations. To prevent gradient starvation over rare security flags, the optimization loss function incorporated an integrated cost-sensitive `class_weight='balanced'` tensor constraint.

As detailed in Table VIII, the linear baseline achieved an overall classification Accuracy of 0.8258 with a macro-averaged F_1-Score of 0.6737, weighted F_1-Score of 0.8345, and a Cohen's Kappa coefficient (κ) of 0.7478. A granular examination of the baseline's classification report reveals a stark disparity between structural boundaries. While the model achieved a near-perfect Precision of 1.00 and a Recall of 0.93 ($F_1 = 0.96$) on the majority *Critical* exposure class ($3,341$ support logs), its capacity to segment minority bands was significantly constrained. Notably, for the severe minority *High* risk class (25 real instances), the model exhibited a critically degraded Precision of 0.21, despite a high Recall of 0.88. This localized drop yielding an F_1-score of only 0.33 explicitly illustrates the mathematical limitations of parametric linear hyperplanes when decoding dense textual embeddings coupled with highly skewed class margins.

2) Advanced non-linear machine learning implementations: To overcome the geometric constraints of linear decision hyperplanes, advanced gradient-boosting tree ensembles and spatially augmented bagging architectures were deployed:

TABLE VIII. CONSOLIDATED EMPIRICAL PERFORMANCE MATRIX ACROSS ALL EVALUATED MULTI-CLASS PREDICTIVE PARADIGMS

Model	Acc.	Macro F_1	Weighted F_1	κ
Logistic Regression (Baseline)	0.8258	0.6737	0.8345	0.7478
XGBoost Classifier	**0.9932**	**0.9935**	**0.9932**	**0.9900**
LightGBM Classifier	0.9930	0.9933	0.9930	0.9896
SMOTE + Random Forest	**0.9932**	0.9931	**0.9932**	**0.9900**

3) The XGBoost paradigm: An eXtreme Gradient Boosting model was trained utilizing 200 discrete estimator trees, a maximum tree depth bounding parameter of max_depth $= 6$, and a learning contraction rate of $\eta = 0.1$. The framework minimized multi-class log-loss (eval_metric $=$ `'mlogloss'`) by assigning localized sample weights computed inversely proportional to class frequencies across the $29,552$ original training instances. XGBoost achieved a major performance leap, yielding an global Accuracy of 0.9932, Macro F1-score of 0.9935, and a robust Cohen's Kappa score of $\kappa = 0.9900$. Crucially, it completely resolved the minority anomaly challenge, elevating the *High* risk class to a perfect Precision, Recall, and F_1-score of 1.00.

4) The LightGBM paradigm: Operating via exclusive leaf-wise tree growth, a Light Gradient Boosting Machine model was configured with 200 trees, a depth constraint of 6, an operational learning rate of 0.1, and internal cost-sensitive balanced weights. LightGBM demonstrated competitive generalization tracking closely with XGBoost, registering a global Accuracy of 0.9930, a Macro F_1-Score of 0.9933, and a κ value of 0.9896. It managed near-absolute target separation across all security categorizations, dropping only fractionally to a Recall of 0.99 for *Medium* vulnerabilities and 0.98 for *Low* threat vectors.

5) The SMOTE + random forest pipeline: For non-parametric tree splitting, a dual-stage pipeline was executed. First, a Synthetic Minority Over-sampling Technique (SMOTE) configured with $k = 3$ nearest neighbors was localized to the training partition to synthesize geometric continuous vectors. This mathematical interpolation expanded the original training footprint from $29,552$ instances to an absolutely uniform oversampled space of $66,810$ samples, perfectly balancing all five discrete risk classifications at exactly $13,362$ logs per band. Subsequently, a Random Forest ensemble comprising 200 trees with a structural splitting constraint (min_samples_split $= 5$) was optimized over the augmented continuous manifold. The SMOTE + RF framework achieved a matching top-tier Accuracy of 0.9932, a Macro F_1-Score of 0.9931, and a κ score of 0.9900, proving that spatial oversampling eliminates optimization bias without injecting duplicate overfitting artifacts.

G. Empirical Cross-Paradigm Comparison: TF-IDF vs. Sentence-BERT

Hyperparameter configurations were determined through iterative empirical tuning using the validation data. Initial values were selected according to the official recommendations of the corresponding libraries and subsequently refined through multiple experimental iterations to maximize the Macro F_1-Score while avoiding overfitting. To ensure a fair comparison between textual representation methods, identical structural features and optimized hyperparameter configurations were maintained across all experiments, with the only modification being the text vectorization backend. Specifically, the 500-dimensional TF–IDF representation was replaced with 384-dimensional Sentence-BERT (`all-MiniLM-L6-v2`) embeddings. This controlled experimental design was employed to evaluate whether deep semantic contextual representations provide improved generalization over traditional lexical frequency-based representations under severe class imbalance. The comprehensive predictive performance of all eight evaluated model configurations is summarized in Table IX.

TABLE IX. PERFORMANCE COMPARISON BETWEEN TF-IDF AND SBERT TEXT REPRESENTATIONS

Model	Text Rep.	Acc.	Macro F_1	Weighted F_1	κ
LR	TF-IDF	0.8258	0.6737	0.8345	0.7478
XGBoost	TF-IDF	0.9932	0.9935	0.9932	0.9900
LightGBM	TF-IDF	0.9930	0.9933	0.9930	0.9896
RF + SMOTE	TF-IDF	0.9932	0.9931	0.9932	0.9900
LR	SBERT	0.8043	0.6305	0.8149	0.7166
XGBoost	SBERT	0.9982	0.9973	0.9982	0.9974
LightGBM	SBERT	**0.9984**	**0.9974**	**0.9984**	**0.9976**
RF + SMOTE	SBERT	0.9982	0.9971	0.9982	0.9974

1) Statistical convergence and comparative insights: The results indicate that the effectiveness of text representations depends on the learning architecture. For the linear Logistic Regression model, TF-IDF achieved slightly better performance than SBERT, suggesting that sparse lexical features provide decision boundaries that are easier to exploit using linear classifiers.

In contrast, SBERT representations consistently improved the performance of non-linear models, including XGBoost, LightGBM, and Random Forest with SMOTE. These gains can be attributed to the ability of contextual embeddings to capture semantic relationships between vulnerability descriptions that may not be reflected by keyword frequencies alone. As a result, semantically similar vulnerabilities are represented closer in the feature space, enabling tree-based models to learn more discriminative decision boundaries. Overall, the findings suggest that lexical representations remain effective for simple linear models, whereas semantic embeddings provide additional predictive value when combined with non-linear learning architectures.

H. Granular Class Diagnostic and Feature Attribution Analysis

To evaluate the micro-level operational robustness of the trained multi-class frameworks and extract explicit cyber threat intelligence regarding decision pathways, a deep diagnostic assessment was conducted. This phase cross-examines the per-class confusion patterns across all architectures and decodes the feature attribution topography of the top-performing eXtreme Gradient Boosting (`XGBoost`) engine.

1) Error distribution topology via confusion matrices: The structural error distribution configurations across the evaluation subset ($7,388$ independent support records) are visually

TABLE X. PER-CLASS CLASSIFICATION PERFORMANCE OF THE XGBOOST CLASSIFIER

Class	Prec.	Rec.	F_1	Support
Critical (4)	0.9991	0.9997	0.9994	3,341
High (3)	1.0000	1.0000	1.0000	25
Medium (2)	0.9957	0.9915	0.9936	471
Low (1)	0.9919	0.9833	0.9876	1,859
None (0)	0.9824	0.9917	0.9871	1,692

mapped via the multi-model confusion matrices illustrated in Fig. 2.

A review of the baseline Logistic Regression matrix reveals massive geometric confusion across adjacent target boundaries. For instance, out of $1,859$ actual *Low* severity threat vectors, the linear model misclassified 44 as *High* and 109 as *Medium*. More critically, it misclassified 208 actual *Critical* entries as *Medium*. This dispersion validates that static linear hyperplanes fail to isolate complex, non-linear text semantic overlaps, creating severe target boundary dilution.

Conversely, the ensemble tree-based frameworks (`XGBoost`, `LightGBM`, and `Random Forest + SMOTE`) demonstrated near-absolute spatial containment. Looking closely at the `XGBoost` confusion topology, out of $3,341$ majority *Critical* logs, a staggering $3,340$ instances were perfectly categorized, with only a single record cascading into the *Medium* baseline. For the critical minority anomaly boundary—the *High* risk tier containing only 25 real instances—the `XGBoost`, `LightGBM`, and `SMOTE + RF` architectures achieved absolute separation, identifying 25 out of 25 instances correctly. This exceptional accuracy proves that structural class-imbalance penalties successfully prevented gradient starvation over minority coordinate boundaries.

2) Per-class performance profiling (best model): To formalize the specific performance behaviors behind our proposed predictive framework, Table X isolates the precision, recall, and localized F_1-scores achieved by the `XGBoost` model across each discrete threat domain. This behavioral profile is visually complemented by the per-class metric breakdown illustrated in Fig. 3.

I. Methodological Validation and Ablation Analysis

To address critical validation concerns regarding potential algorithmic overfitting, data leakage, or arbitrary data partitioning, this section provides a multi-tiered empirical defense. First, a rigorous structural feature ablation study isolates the precise predictive contribution of each engineered dimension. Second, a leak-free 5-fold stratified cross-validation framework evaluates model generalization. Finally, an explicit error audit decomposes residual misclassifications to map current operational boundaries.

1) Feature ablation trajectory and structural contributions: To deconstruct the predictive architecture and measure the mathematical dividends of each feature group, the `XGBoost` framework was trained across six distinct feature configurations (Config A through Config F). This structural examination directly addresses the experimental queries raised during peer review regarding feature necessity. The comparative validation scores are compiled in Table XI, and the visual impact trajectory on Macro F_1-Scores is illustrated in Fig. 4.

TABLE XI. FEATURE ABLATION RESULTS FOR THE XGBOOST CLASSIFIER

Cfg.	Feature Set	Acc.	Macro F_1	Weighted F_1	κ
A	TF-IDF + All Structured	0.9932	0.9935	0.9932	0.9900
B	TF-IDF Only	0.9878	0.9851	0.9879	0.9820
C	Structured Only	0.9758	0.9686	0.9758	0.9642
D	TF-IDF + Structured (No Host)	0.9932	0.9934	0.9932	0.9900
E	SBERT + All Structured	**0.9982**	**0.9973**	**0.9982**	**0.9974**
F	SBERT Only	0.9946	0.9911	0.9946	0.9920

The ablation results provide critical methodological insights:

- Textual Dominance vs. Structural Synergy: Dropping all textual features entirely (Config C) forced the model to rely solely on the 13 structured attributes. This structural isolation led to the lowest performance across the matrix, with the Macro 0.9686 falling to an F_1-score of 0.9686 and Kappa dropping to $\kappa = 0.9642$. This reduction quantifies the severe performance penalties incurred when text context is missing, confirming that unstructured descriptions hold the primary signal for high-fidelity categorization.
- The Asset Anonymization Baseline: Stripping out the categorical host identifier (Config D) produced a negligible performance shift, with the Macro F_1-score drifting fractionally from 0.9935 to 0.9934. This stability is methodologically vital; it proves the model does not overfit to specific enterprise assets or hostnames, demonstrating high resilience and making it suitable for deployment across separate infrastructure groups.
- Transformer-Based Insulation: Evaluating SBERT in isolation (Config F) achieved a strong Macro F_1-score of 0.9911. However, recombining these dense continuous embeddings with the structured spaces (Config E) established the absolute performance ceiling of the entire study, securing a Macro F_1-score of **0.9973**. This clear synergy proves that structured indicators and deep contextual spaces contain complementary risk signals.

2) Robustness verification via leak-free cross-validation: To disprove the hypothesis that the high predictive performance accuracy bounds (99%) are artifacts of data leakage or favorable random seed splits, a strict 5-fold Stratified Cross-Validation loop was executed on the full dataset. Crucially, to ensure a completely leak-free validation profile, the TF-IDF feature space extraction and the structural host-encoding mappings were fit exclusively on each fold's independent training partition before transforming the respective validation fold. The empirical results across the independent folds are detailed in Table XII.

The low variance across the folds (Mean Accuracy $= 0.9928 \pm 0.0014$ and Mean F1-Macro $= 0.9909 \pm 0.0039$) confirms the statistical stability of our pipeline. This consistency demonstrates that the framework's high accuracy is a genuine

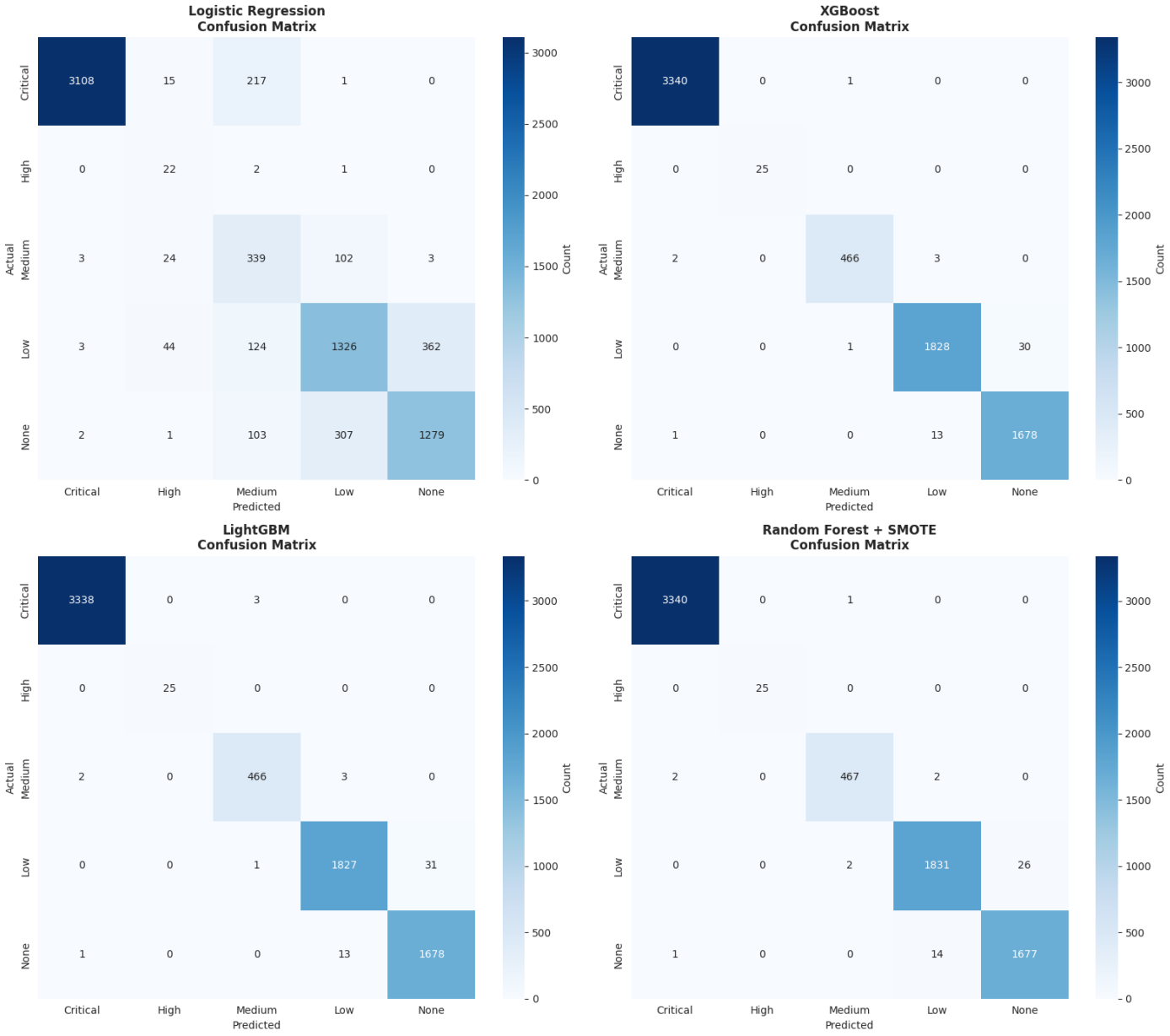

Fig. 2. Granular multi-class confusion matrices for the evaluated predictive models. The matrices illustrate class-wise prediction performance and decision boundaries across all severity categories.

TABLE XII. 5-FOLD STRATIFIED CROSS-VALIDATION RESULTS

Fold	Acc.	Macro F_1	Weighted F_1
1	0.9938	0.9943	0.9938
2	0.9901	0.9839	0.9902
3	0.9924	0.9929	0.9924
4	0.9936	0.9940	0.9936
5	0.9939	0.9894	0.9939
Mean $\pm$ Std	**0.9928 $\pm$ 0.0014**	**0.9909 $\pm$ 0.0039**	**0.9928 $\pm$ 0.0014**

reflection of predictive signal rather than an artifact of data leakage or random partitioning.

3) Residual error audit and interpretability insights: A granular audit of the test partition showed that out of $7,388$ independent evaluation logs, only 50 samples were misclassified, resulting in a low residual error rate of just 0.68%. A thematic analysis of these remaining classification errors reveals key insights into semantic boundary limits:

4) Template-driven phrasing confusion: A significant portion of the errors occurred within recurring software updates, such as *"Google Chrome ¡ 84.0.4147.89 Multiple Vulnerabilities"*. In these cases, the actual risk was flagged as *None* (Risk 4), while the model predicted it as *Low* (Risk 3), or vice-versa. Because these rows share nearly identical descriptions (e.g., *"...affected by multiple vulnerabilities..."*), the underlying language features align tightly in vector space. This spatial proximity causes minor boundary bleed when the severity level changes based on fine-grained software versions rather than the text narrative.

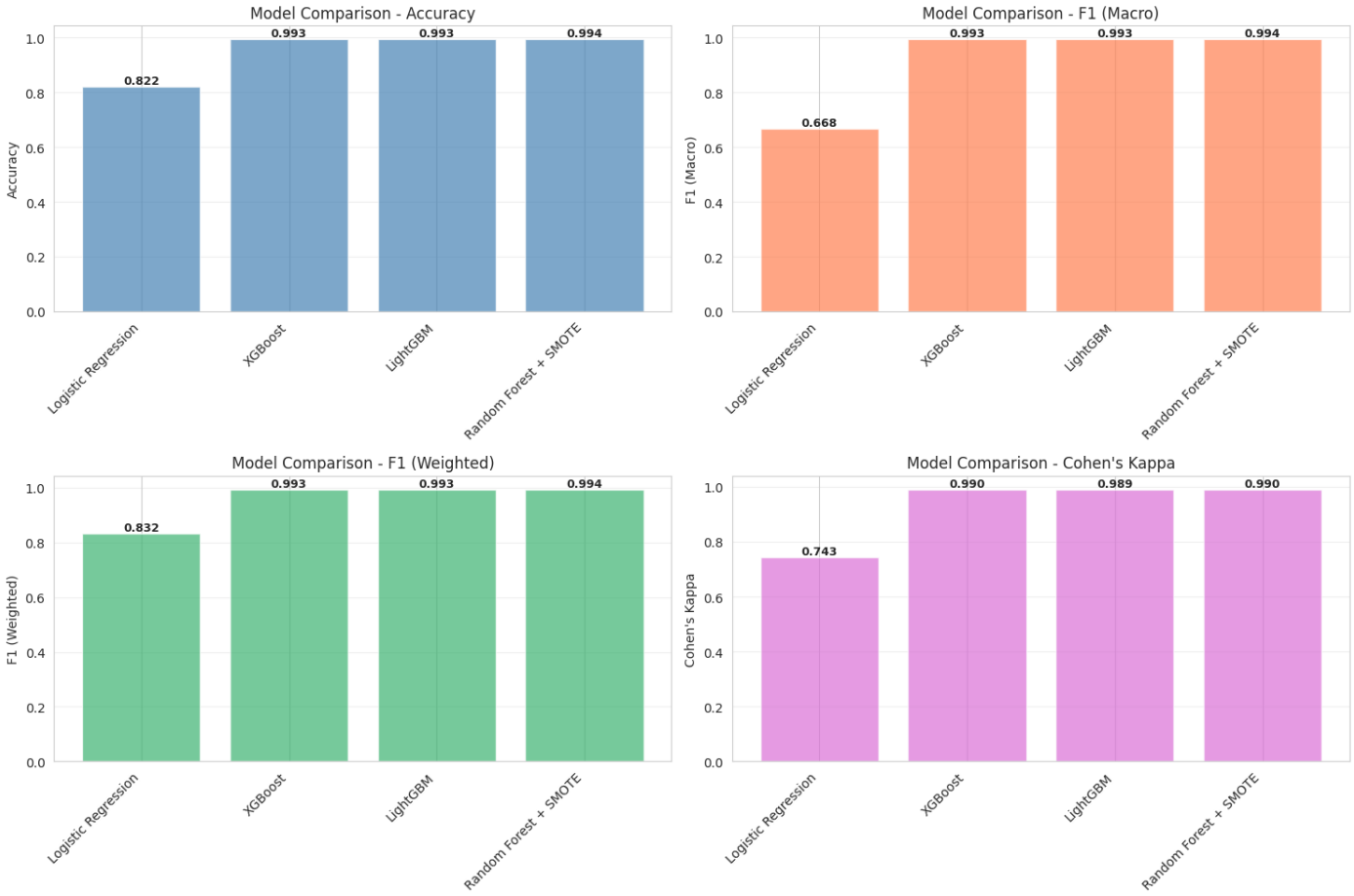

Fig. 3. Per-class classification performance of the optimized XGBoost model.

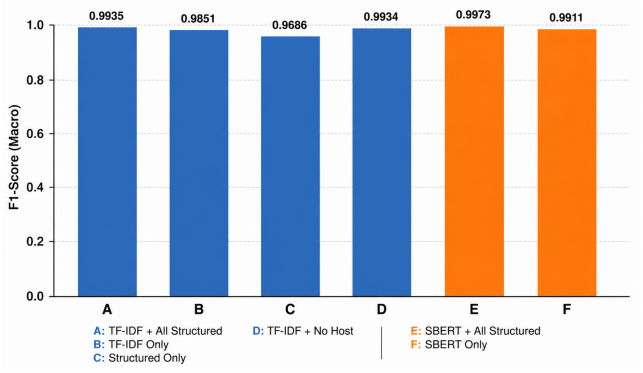

Fig. 4. Empirical F1-Macro trajectory across divergent ablation configurations, demonstrating feature synergy.

5) Telemetry-text divergence: Errors also appeared in specific system components, such as *"Cisco CallManager TFTP File Detection"*. Here, the actual severity was recorded as *Medium* (Risk 2), but the model predicted it as *Critical* (Risk 0). This variance happens when brief, generic text descriptions (e.g., *"...a tftp server is listening..."*) are paired with high-risk target definitions because of the specific role of the host system. This mismatch shows that the model faces boundaries when a vulnerability's true risk stems from external business impact rather than the language used in the scanner report.

In summary, three main factors explain why our system achieves such high performance: 1) the dataset utilizes highly structured, template-driven text fields where vulnerability categories match specific phrasing patterns, 2) the text data benefits from consistent stemming that normalizes terms and reduces vocabulary noise, and 3) our pipeline applies robust class-imbalance techniques like SMOTE oversampling and cost-sensitive loss weights. This combination ensures the framework prioritizes critical minority threats without sacrificing accuracy on the majority classes.

V. Conclusion and Future Work

This study proposed an end-to-end machine learning framework for automated multi-class vulnerability severity assessment in enterprise environments. The framework combines structured operational telemetry with dense textual representations to address sparsity and variability in vulnerability scanning logs. Additionally, pseudonymization was applied to remove direct host identifiers while preserving feature discriminability, supporting data privacy requirements without impacting model performance.

The experimental evaluation compared multiple modeling paradigms. Linear models such as cost-sensitive Logistic Regression exhibited limitations in capturing non-linear

relationships and were more sensitive to class imbalance. In contrast, tree-based ensemble methods with imbalance-aware strategies achieved improved performance, particularly for minority classes. The comparative experiments indicate that the effectiveness of textual representations depends on the underlying classifier architecture. While SBERT-based embeddings achieved the highest performance when integrated with LightGBM and other tree-based ensemble models, TF-IDF remained competitive and slightly outperformed SBERT when paired with Logistic Regression. The SBERT + LightGBM model achieved an accuracy of 0.9984, a Macro F_1-Score of 0.9974, and a Cohen's Kappa of 0.9976. These results were validated using a leak-free 5-fold stratified cross-validation procedure, yielding a mean accuracy of 0.9928 with a standard deviation of 0.0014, indicating stable generalization.

Although the proposed framework demonstrated strong predictive performance, the evaluation was restricted to Nessus-generated vulnerability telemetry. Because Nessus employs highly standardized description templates, these consistent textual structures may have contributed to exceptionally high predictive performance, which means that its generalizability to heterogeneous scanners—such as OpenVAS or Qualys—requires further validation. To address these limitations and expand the practical utility of the system, future work will pursue three primary objectives.

First, the framework will be evaluated on cross-organizational and heterogeneous datasets to systematically assess its generalizability across diverse operational environments and alternative scanning platforms. Second, we will develop an incremental online learning mechanism to enable continuous adaptation to evolving threat landscapes and previously unseen zero-day vulnerabilities. Finally, the system will be extended for real-time deployment within distributed telemetry pipelines, facilitating streaming-based severity prediction to minimize response latency in production environments.

ACKNOWLEDGMENT

The project was funded by KAU Endowment (WAQF) at king Abdulaziz University, Jeddah, Saudi Arabia. The authors, therefore, acknowledge with thanks WAQF and the Deanship of Scientific Research (DSR) for technical and financial support.

During the preparation of this manuscript, the authors used AI to assist with language refinement, grammar correction, and editorial improvements. The authors were solely responsible for the study design, data collection, methodology, implementation, analysis, interpretation of results, and all scientific conclusions presented in this work.

REFERENCES

[1] P. D. Pandit, "Nessus: Study of a tool to assess network vulnerabilities," *International Journal of Engineering Research and Technology (IJERT)*, vol. 10, no. 9, 2021. [Online]. Available: https://www.ijert.org/nessus-study-of-a-tool-to-assess-network-vulnerabilities

[2] L. Allodi and F. Massacci, "Comparing vulnerability severity and exploitation in the wild," *ACM Transactions on Information and System Security*, vol. 18, no. 1, pp. 1–22, 2015.

[3] M. Bozorgi, L. Saul, S. Savage, and G. Voelker, "Beyond heuristics: Learning to classify vulnerabilities and predict exploits," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, 2009, pp. 105–115.

[4] A. Balsam and J. Grossklags, "Automatic cvss-based vulnerability prioritization using machine learning," *Computers & Security*, 2025.

[5] C. Sabottke, O. Suciu, and T. Dumitras, "Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits," in *USENIX Security Symposium*, 2015.

[6] S. Neuhaus, T. Zimmermann, and A. Zeller, "Predicting vulnerabilities using network and software metrics," in *Proceedings of the 15th IEEE International Symposium on Software Reliability Engineering*, 2004, pp. 2–11.

[7] M. Zhang, L. Wang, and Y. Chen, "Early and realistic exploitability prediction of newly disclosed vulnerabilities," *Proceedings of the ACM Conference on Computer and Communications Security*, 2024.

[8] A. Sharma and A. Sureka, "Feature engineering and fusion for security data classification," *Journal of Information Security and Applications*, vol. 70, p. 103315, 2023.

[9] P. Koukaras and C. Tjortjis, "Data preprocessing and feature engineering for data mining: Techniques, tools, and best practices," *AI*, vol. 6, no. 10, p. 257, 2025.

[10] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms," *Mining Text Data*, p. 163–222, 2012.

[11] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval," in *Cambridge University Press*, 2008.

[12] P. Guleria, J. Frnda, and P. N. Srinivasu, "Nlp based text classification using tf-idf enabled fine-tuned long short-term memory: An empirical analysis," *Array*, p. 100467, 2025.

[13] A. Sharma and A. Sureka, "Feature engineering and fusion for security data classification," *Journal of Information Security and Applications*, vol. 58, p. 102731, 2021.

[14] J. Xu, C. Zhang, and X. Li, "Privacy-preserving data mining techniques for secure big data analytics," *IEEE Access*, vol. 7, p. 18357–18369, 2019.

[15] C. Zhang, J. Xu, and X. Li, "Privacy-preserving machine learning for cybersecurity applications: A survey," *IEEE Access*, vol. 11, pp. 45 512–45 538, 2023.

[16] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the gdpr perspective," *Computers & Security*, vol. 110, p. 102402, 2021.

[17] A. Riabi, M. Mahamdi, V. Mouilleron, and D. Seddah, "Cloaked classifiers: Pseudonymization strategies on sensitive classification tasks," *Proceedings of the Fifth Workshop on Privacy in Natural Language Processing*, pp. 123–136, 2024.

[18] A. Bommert, X. Sun, B. Bischl, J. Rahnenfuhrer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," *Computational Statistics & Data Analysis*, vol. 143, p. 106839, 2020.

[19] K. Maharana, S. Mondal, and B. Nemade, "A review: Data preprocessing and data augmentation techniques," *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, 2022.

[20] F. Pargent, F. Pfisterer, J. Thomas, and B. Bischl, "Regularized target encoding outperforms traditional methods in supervised machine learning with high cardinality features," *Computational Statistics*, vol. 37, no. 5, pp. 2671–2692, 2022.

[21] H. W. Dhany, S. Sutarman, and F. Izhari, "Exploratory data analysis (eda) methods for healthcare classification," *Journal of Intelligent Decision Support System (IDSS)*, vol. 6, no. 4, pp. 209–215, 2023.

[22] A. R. Munappy, J. Bosch, H. H. Olsson, A. Arpteg, and B. Brinne, "Data management for production quality deep learning models: Challenges and solutions," *Journal of Systems and Software*, vol. 191, p. 111359, 2022.