Applied Artificial Intelligence Research

INNOVIAIR

Reassessing CVSS Severity Classification with LightGBM and Operating System Metadata

Zhixiang Xu[a,*], Weihao Li[a], Zhiyong Zhou[a], Wenpei Zhang[a], Zi'an Yu[a], Mingming Li[a], Yunfei Yin[a]

[a]*Department of Mathematics and Statistics, Kashi University, Kashi, Xinjiang 844000, China*

ARTICLEINFO

Keywords:
LightGBM
CVSS Severity
Operating System Metadata
Data Leakage
TreeSHAP
Vulnerability Classification

ABSTRACT

CVSS severity and observed exploitation are different prediction targets. This study reassesses a LightGBM pipeline using an available 1,314-record vulnerability dataset and explicitly defines its label as a CVSS score greater than 9. The original 98.48% accuracy is reproducible, but the pipeline encodes complete score-defining vectors and fits text features before splitting. We therefore remove identifiers and complete vectors, fit preprocessing only on training records, and compare five classifiers using parsed attack vector, TF-IDF text and two operating-system metadata features. After description deduplication, 882 records with CVE identifier years up to 2023 are used for training and 390 records from identifier year 2024 for testing. This is an identifier-year robustness check, not a disclosure-time evaluation. LightGBM achieves 95.64% accuracy, 41.67% recall, 0.8412 ROC-AUC and 0.4821 average precision; the majority baseline already achieves 93.85% accuracy. Adding operating-system metadata changes average precision by −0.0303 (paired bootstrap 95% interval −0.1077 to 0.0409), providing no consistent improvement. TreeSHAP dependence plots and error cases reveal strong reliance on attack vector and lexical associations. The results support a limited severity-classification case study and demonstrate why these data cannot establish real-world exploitability prediction.

1.Introduction

Vulnerability management requires a clear separation between severity, exploit development and observed exploitation. CVSS describes technical severity, whereas an exploitation predictor requires an independently observed outcome and an explicit observation window[1,2]. A model trained to reproduce a CVSS threshold may be useful as a restricted classification benchmark, but its accuracy cannot establish the probability of an attack. This distinction is especially important when a dataset contains both a score-derived target and attributes used to calculate that score.

Research on exploit prediction already combines heterogeneous information. Jacobs et al. developed EPSS as a data-driven approach to vulnerability threat assessment[3]. A later EPSS study extends this community-driven prediction framework[4]. Suciu et al. model the development of functional exploits using time-varying evidence and explicitly examine label noise[5]. These studies define outcomes related to exploitation or functional exploit development. Our available dataset contains neither outcome. Consequently, we do not present severity classification as a replacement for these approaches or compare their reported scores with ours across incompatible targets.

LightGBM is an established gradient-boosting method[6]. Combining a tree ensemble with structured vulnerability fields and text is not, by itself, an algorithmic innovation. The relevant empirical question here is narrower: does the available operating-system metadata add useful discrimination once target definition, feature provenance and evaluation design are made explicit? A second question is whether a high overall accuracy masks weak identification of the minority class.

The emphasis on global and individual explanations also connects this study to work in Applied Artificial Intelligence Research. Liu combines model comparison with global and individual SHAP analysis in customer churn prediction[7]. Wang studies heterogeneous traffic-prediction features and SHAP attribution[8]. These papers provide methodological context for interpretable applied learning, rather than cybersecurity evidence. We extend this discussion through a vulnerability-data case study in which explanation is used to expose fragile associations and test an environmental-feature claim.

* Corresponding author.
E-mail address: xzxlaplace@163.com.
https://doi.org/10.65455/451jd869
Received 9 August 2026; Received in revised form 20 September 2026; Accepted 20 September 2026; Available 24 September 2026

The contribution is an auditable reassessment with three components: (1) a precise score-derived label and a documented audit of the original high-accuracy pipeline; (2) training-only preprocessing, description deduplication, an identifier-year holdout and controlled operating-system ablations; and (3) uncertainty estimates, dependence analysis and concrete correct and incorrect classifications. No new learning algorithm, operating-system security mechanism or validated exploitation forecast is claimed.

2.Methodology

2.1.Target definition and scope

For record i with supplied CVSS score s_i, the binary outcome is $y_i = 1$ if $s_i > 9$, and $y_i = 0$ otherwise. The threshold is retained from the supplied analysis code so that its results can be audited. It is not the official CVSS critical-category boundary, which includes 9.0[1]; the dataset contains four records scoring exactly 9.0, and these remain negative under our stated rule. We refer to the positive class as the above-9 severity class. Negative means a score at most 9, not safe, unexploitable or low risk.

The label is not derived from Exploit Code Maturity, an Exploit-DB match, a CVSS exploitability sub-score or an observed attack. The CSV contains no exploitation-status field, disclosure timestamp, label-observation date or extraction timestamp. The model output is therefore a score for membership in this severity class; it is not a calibrated exploitation probability. Since the target is deterministic when the CVSS score is known, applying the threshold directly is sufficient. Learning is studied here only as a diagnostic of restricted feature representations, not as a necessary substitute for that rule.

2.2.Audit of the original pipeline

The supplied script defines the target using CVSS Score > 9, excludes the numeric score from the feature matrix, and uses four inputs: label-encoded CVE ID, label-encoded Attack Vector, a comma-separated Affected OS count and the row sum of a TF-IDF vector. Inspection shows that Attack Vector actually contains complete CVSS v3.1 base vectors, not just the AV metric. Encoding these vectors allows the classifier to recognize combinations that define severity. This is target-definition contamination for an exploitation claim, even without the numeric score as an input.

The original vectorizer is fitted to all descriptions before the stratified 80/20 split. Thus, vocabulary and inverse-document-frequency estimates use test text. CVE identifier codes also introduce arbitrary ordinal structure. We reproduce this pipeline only as a historical diagnostic and replace it for the main comparison. Its feature-importance plots do not establish operating-system effects, and the supplied code does not compute SHAP values. The revised explanation analysis is newly calculated from the corrected model.

2.3.Feature construction

The main representation uses three groups. First, only the AV token is parsed from the complete vector and one-hot encoded into Network, Adjacent, Local and Physical categories. The encoder is fitted on training records and ignores unseen categories. The complete vector, numeric CVSS score, CVE identifier, identifier year and other score-defining CVSS components are excluded from model inputs. AV is still a component of the severity formula, so this restricted task is not independent of CVSS construction; we also report a text-only ablation.

Second, descriptions are represented by individual unigram TF-IDF features rather than a single row sum. The vocabulary is fitted only on training descriptions, with max_features = 500, min_df = 2, sublinear_tf = True and L2 normalization. Default lowercasing and tokenization are used; stop words are not removed. Keeping 500 dimensions provides a compact, fixed baseline. The retained common words also make it possible to identify undesirable lexical dependencies in the explanation analysis. We do not equate TF-IDF with deep semantic understanding.

Third, operating-system metadata is represented by OS_count, the number of nonempty comma-separated entries, and OS_missing, an indicator that the source field is missing. Missing fields are assigned a count of zero and an explicit missingness flag. This zero means no recorded entries, not that no systems are affected. Entries are not verified as distinct systems or normalized product versions; the largest count is 356. The two derived variables are related and their SHAP attribution may be shared. We make no claims about kernel parameters, protection mechanisms or host-specific configurations, none of which are observed.

2.4.Deduplication and evaluation design

Descriptions are lowercased, internal whitespace is collapsed and leading/trailing whitespace is removed. Records are sorted by identifier year and then lexicographic CVE ID; the first record per normalized description is retained. This removes 42 repeated descriptions and leaves 1,272 records. No normalized-description group has conflicting binary labels. This procedure limits exact-copy overlap, but does not eliminate semantically similar advisories or vulnerabilities in the same product family.

The main robustness partition uses CVE identifier years up to 2023 for training and identifier year 2024 for testing. The resulting sizes are 882 and 390. Identifier year is a grouping variable only. The CVE Program explains that the year in an identifier can reflect assignment or public disclosure [9]; it is not a disclosure timestamp. Accordingly, this experiment is called an identifier-year holdout, not chronological validation. It cannot establish that every training feature was available before every test disclosure, or remove leakage from later edits to vulnerability descriptions.

All learned transformations are fitted on the training partition. Model settings and the probability threshold of 0.5 are fixed before the new comparison; no test-set tuning, oversampling, class weighting, threshold optimization or calibration is performed. A secondary stratified 80/20 split of

the same deduplicated corpus, with seed 42 and training-only transformations, measures sensitivity to splitting. It is not the main evidence for prospective performance. The comparison changes sample composition and prevalence and therefore does not isolate a pure temporal effect.

2.5.Models and uncertainty

The main models share the same feature matrices. LightGBM uses 100 trees, learning rate 0.05, 15 leaves and a minimum of 20 training examples per leaf. Random Forest uses 300 trees, minimum leaf size 2 and square-root feature sampling. XGBoost uses 100 trees, depth 3 and learning rate 0.05, without row or column subsampling. CatBoost uses 100 iterations, depth 4 and learning rate 0.05. Logistic regression uses C = 1 and up to 2,000 iterations. Random seeds are 42. No claim of equal optimization budgets or optimal hyperparameters is made. Logistic regression replaces the original QDA comparator because the revised text representation is sparse and high-dimensional.

We report accuracy, positive-class precision, recall, F1, ROC-AUC and average precision (AP). AP is the non-interpolated precision-recall summary calculated by scikit-learn. The majority baseline predicts no positives at threshold 0.5 and assigns the training prevalence as a constant ranking score. A CVSS-threshold oracle is reported separately as a definitional check, not a deployable learned competitor when the score is unavailable. Undefined precision and F1 are set to zero for models predicting no positives.

Uncertainty is estimated using 2,000 ordinary bootstrap resamples of the fixed test records, seed 20260919, with percentile 95% intervals. Resamples lacking either class are skipped. The operating-system AP difference uses paired resamples, applying the same sampled test indices to both predictions. These intervals are conditional on the trained models and this partition; they do not include retraining variability, product clustering or uncertainty from the sampling process. They are descriptive intervals rather than proof of population-level significance.

2.6.Interpretation procedure

TreeSHAP contributions[10] are calculated with LightGBM pred_contrib on held-out records. The built-in tree-path-dependent attribution sums to the raw model margin: $f(x) = \phi_0 + \Sigma_j \phi_j(x)$, and the class score is $1/(1 + \exp(-f(x)))$. Contributions are reported in log-odds units, not probability points. Global importance is the test-set mean of absolute contributions. For a feature group we first sum its signed contributions within each record and then take the mean absolute value; this differs from summing individual global importances.

We examine dependence plots for OS_count and the highest-ranked individual feature. To make case selection reproducible, we select the highest-score true positive and false positive, and the lowest-score false negative and true negative at threshold 0.5. These deliberately extreme examples illustrate behavior and errors; they are not a representative case sample. SHAP describes the fitted model, not causal drivers of attacks or operating-system security.

3.Experimental results and analysis

3.1.Dataset audit and distributions

The available CSV contains 1,314 distinct CVE identifiers and five original columns. The original manuscript attributes these records to NVD, but the extraction query, sampling criteria and collection date are unavailable. We therefore treat this file as a fixed convenience sample and do not claim that it represents the complete NVD population. All vector strings have a CVSS:3.1 prefix. There are 236 distinct complete vector strings, 1,275 distinct raw descriptions and 1,272 normalized descriptions.

Table 1. Raw dataset audit

Item	Value
Records / unique CVEs	1,314 / 1,314
Above-9 / at-most-9	165 / 1,149
Positive prevalence	12.56%
Negative-to-positive ratio	6.96 : 1
Missing OS field	883 (67.20%)
Missing other four fields	0
CVSS mean ± SD	6.79 ± 1.79
CVSS median [Q1, Q3]	6.7 [5.5, 7.8]
CVSS range	2.2–10.0
OS count median [Q1, Q3]	0 [0, 1]
OS count mean / maximum	4.04 / 356
AV N / L / A / P	893 / 343 / 61 / 17

The CVE identifier-year counts are: 2010 (1), 2015 (1), 2016 (1), 2017 (1), 2018 (2), 2020 (11), 2021 (4), 2022 (39), 2023 (852) and 2024 (402). Thus, 95.43% of records carry 2023 or 2024 identifiers. This is not evidence of continuous disclosure coverage from 2010 to 2024. Attack-vector proportions are 67.96% Network, 26.10% Local, 4.64% Adjacent and 1.29% Physical. OS_count is highly skewed: 883 records have zero recorded entries, 210 have one, 44 have two and 177 have three or more. Figure 1 summarizes the score distribution and missingness.

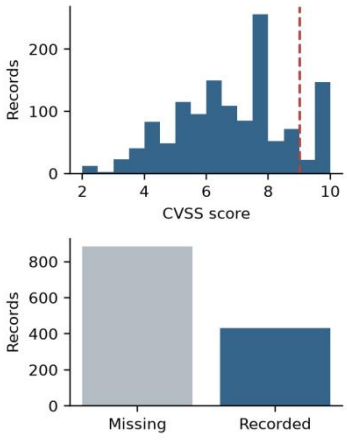


Fig 1. Raw CVSS distribution and OS-field completeness

Note: The dashed line marks 9; positivity requires a score strictly greater than 9.

After deduplication, the training set has 138 positives and 744 negatives (15.65% positive); the test set has 24 positives and 366 negatives (6.15% positive). OS metadata is missing in 597/882 training records (67.69%) and 277/390 test records (71.03%). The change in prevalence and the small number of test positives are essential to interpreting classification metrics. The original random test set contained 263 records and 33 positives; it was the holdout, not the entire dataset.

3.2.Model comparison on the identifier-year holdout

Table 2a. Threshold metrics on 390 test records

Model	Acc.	Prec.	Recall
LightGBM	0.9564	0.7692	0.4167
Random forest	0.9564	1.0000	0.2917
XGBoost	0.9590	0.9000	0.3750
CatBoost	0.9590	1.0000	0.3333
Logistic reg.	0.9487	0.7500	0.2500
Majority	0.9385	0.0000	0.0000

Table 2b. F1 and ranking metrics on the same records

Model	F1	AUC	AP
LightGBM	0.5405	0.8412	0.4821
Random forest	0.4516	0.8699	0.5159
XGBoost	0.5294	0.8152	0.4603
CatBoost	0.5000	0.8589	0.5196
Logistic reg.	0.3750	0.8881	0.5418
Majority	0.0000	0.5000	0.0615

LightGBM identifies 10 of the 24 positives, with 3 false positives and 14 false negatives. Its accuracy is 95.64%, only 1.79 percentage points above the majority baseline, while recall is 41.67%. Figure 2 reports ranking curves and Figure 3 shows the confusion matrix. XGBoost and CatBoost have slightly higher accuracy (95.90%). Logistic regression has the highest ROC-AUC (0.8881) and AP (0.5418), whereas LightGBM has the highest F1 at the fixed threshold (0.5405). Different metrics favor different models; the results do not establish overall LightGBM superiority.

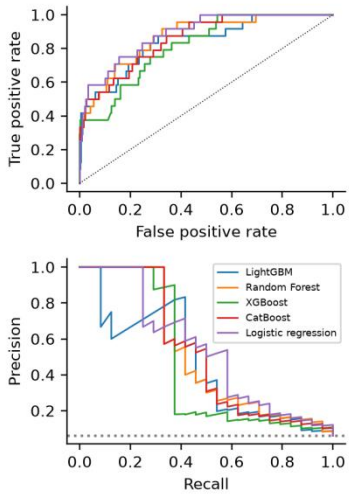


Fig 2. ROC and precision-recall curves on the identifier-year holdout

Note: The horizontal PR reference is test prevalence, 0.0615

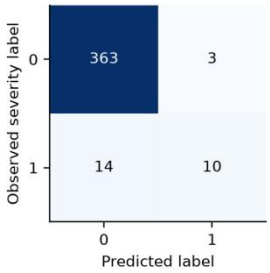


Fig 3. LightGBM confusion matrix

Note: Rows are observed score-derived classes; columns are predictions at 0.5.

Table 3. LightGBM bootstrap uncertainty

Metric	Estimate	95% interval
Accuracy	0.9564	0.936–0.977
Recall	0.4167	0.219–0.611
F1	0.5405	0.324–0.718
AUC	0.8412	0.758–0.920
AP	0.4821	0.282–0.673

The recall interval is wide (0.219–0.611), as is the AP interval (0.282–0.673). With only 24 positive test records, one additional true positive changes recall by 4.17 percentage points. The high accuracy is therefore insufficient evidence of reliable minority-class identification. The CVSS-threshold oracle attains 1.0000 on all six metrics by definition. This confirms that the task is label reconstruction when the score is available, rather than an independent measure of exploitation.

3.3.Operating-system ablation and pipeline sensitivity

Table 4. LightGBM feature ablation

Inputs	F1	AUC	AP
Text only	0.4615	0.7508	0.3828
AV, text	0.4737	0.8435	0.5123
AV, text, OS	0.5405	0.8412	0.4821
OS only	0.0000	0.6318	0.1110

Adding OS_count and OS_missing to AV plus text increases F1 from 0.4737 to 0.5405 and recall from 0.3750 to 0.4167. However, ROC-AUC decreases from 0.8435 to 0.8412 and AP decreases from 0.5123 to 0.4821. The paired AP change is −0.0303, with a 95% interval from −0.1077 to 0.0409. The interval includes zero, and ranking and threshold metrics disagree. We therefore withdraw the claim that operating-system attributes consistently improve performance. OS-only features predict no positives at 0.5; their AP of 0.1110 is weak despite exceeding the prevalence baseline.

The corrected random split contains 1,017 training and 255 test records, including 32 test positives. Its LightGBM accuracy, recall, ROC-AUC and AP are 0.9137, 0.4375, 0.9448 and 0.7770. The identifier-year holdout has lower ranking performance, but higher accuracy because the class balance and record composition differ. This contrast demonstrates evaluation sensitivity; it does not justify a causal claim about the exact amount of temporal leakage.

Re-running the original four-feature pipeline gives accuracy 0.9848, precision 0.9677, recall 0.9091, F1 0.9375

and ROC-AUC 0.9821. These reproduce the original performance table; its narrative recall of 87% was inconsistent with the table and is removed. The reproduced confusion matrix is TN = 229, FP = 1, FN = 3 and TP = 30. These figures are retained only to document the audit and are not the revised main results.

As an additional diagnostic, one-hot encoding the complete CVSS vector alone, with the same identifier-year partition and revised LightGBM settings, produces accuracy 0.9769 and F1 0.7692. Such performance concerns reconstruction of a score-derived class. Complete vectors contain the metrics used to define that class, so this result cannot validate exploitation forecasting. Differences between the legacy and corrected results also reflect representation, deduplication, split and model-setting changes; this audit does not attribute the whole difference to any single source of leakage.

3.4. Global and dependence explanations

The largest mean absolute TreeSHAP contribution is AV_N (0.7491 log-odds), followed by the terms cross (0.2390), vulnerability (0.1989) and attack (0.1907). Figure 4 shows the ten leading individual features. Grouped mean absolute contributions are 0.7649 for AV, 1.0299 for text and only 0.00935 for OS metadata. Group magnitudes depend on grouping and correlation and are not directly additive. Their ordering supports a predominantly attack-vector and lexical model in this sample, not the earlier claim of a substantial environmental contribution.

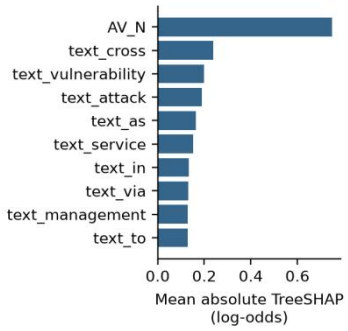


Fig 4. Global TreeSHAP importance on the held-out records

Note: Contributions explain the score-derived class in log-odds units.

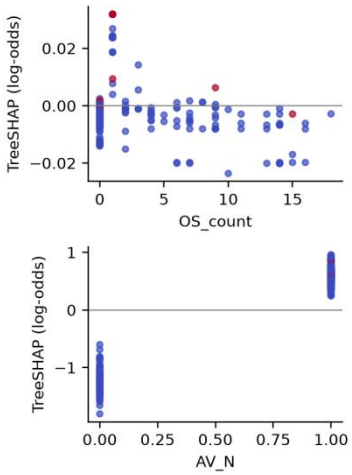


Fig 5. Dependence plots for OS_count and AV_N

Note: Red indicates above-9 severity and blue indicates the at-most-9 class.

Figure 5 compares OS_count with its own contribution and AV_N with its contribution. The AV_N plot shows that the direction and magnitude of attribution vary with the remaining features, illustrating conditional rather than purely global importance. The OS_count contributions are small and do not establish a reliable monotonic relation between the number of recorded entries and severity. Missingness, entry-count skew and the absence of normalized system identities prevent a security-mechanism interpretation. Colors identify the observed severity class, not whether exploitation occurred.

3.5. Concrete predictions and failure cases

For every test record, the sum of the baseline and feature contributions reconstructs the raw margin to within 6.3×10^{-15}. The common baseline is −3.1493 log-odds. The following cases are taken from the supplied snapshot; their scores and descriptions are not claimed to be current NVD records.

The selected true positive, CVE-2024-22088, describes a use-after-free in Lotos WebServer and has supplied CVSS 9.8. Its model score is 0.7648. AV_N contributes +0.9465, use +0.7002 and via +0.4619 log-odds. Together with the remaining terms, these move the raw margin to +1.1794. The OS field is missing. The prediction is correct for the severity label but says nothing about observed exploitation of this vulnerability.

The selected false negative, CVE-2024-22051, describes a CommonMarker integer overflow with possible memory corruption and remote code execution. Despite supplied CVSS 9.8 and Network AV, the model score is only 0.0186. The words that and an contribute −0.8294 and −0.6312; AV_N (+0.4867), execution (+0.2306) and remote (+0.1841) do not offset the other negative contributions. The final raw margin is −3.9660. Dependence on common function words suggests unstable lexical associations and limits the claim that the representation captures meaningful attack semantics.

The selected false positive, CVE-2024-21747, describes SQL injection in WP ERP and has supplied CVSS 7.6, hence label 0. Its score is 0.6664. Network AV (+0.8690), command (+0.7520) and injection (+0.3748) contribute toward a positive prediction, although the true score-derived class is negative. This case shows that attack-related language can be insufficient to distinguish the above-9 threshold; it must not be described as a false alarm about an actual attack.

The selected true negative, CVE-2024-20710, concerns an out-of-bounds read in Adobe Substance 3D Stager and has supplied CVSS 5.5. Its model score is 0.0019. Absence of Network AV contributes −0.9608, and that contributes −0.6129, producing a raw margin of −6.2657 after all contributions. The record has two OS entries. Together, these examples demonstrate what the classifier uses, including errors, rather than converting feature attribution into causal vulnerability explanations.

4. Discussion and limitations

The central finding is that a reproducible high accuracy can answer the wrong substantive question. The original target measures severity, the complete-vector feature is tightly

linked to that target, and the original evaluation also learns text statistics from the test corpus. Correcting these issues leads to a narrower and less favorable empirical claim. The revised study supplies a transparent baseline and a cautionary case for applied vulnerability learning, rather than evidence that operating-system metadata improves exploitation prediction.

Several limits remain. First, no independent exploitation labels or observation windows exist, so we cannot assess functional exploit development or in-the-wild attacks. Second, CVE-year grouping cannot replace disclosure-date splitting or historical feature snapshots. Third, the file is a small convenience sample with only 24 positive test records, unknown collection criteria and heavy concentration in 2023–2024 identifiers. Fourth, exact-description deduplication leaves possible vendor and advisory-family dependence. Fifth, OS metadata is mostly missing, unnormalized and represented by coarse counts. Sixth, the single fixed configuration and split do not establish an optimal model or a stable ranking across samples.

The experiments also retain a partial connection to the target through AV. This is appropriate to the declared severity-classification diagnostic but unsuitable as independent proof of threat prediction. When the actual CVSS score is known, thresholding it is preferable to learning this label. Where the score is unavailable, this study still lacks the timing evidence needed to demonstrate an operational early-warning use case. The results should therefore not be deployed as exploitation probabilities or as a standalone remediation policy.

5.Conclusion and targeted future work

This study redefines the outcome explicitly and reassesses LightGBM using the available evidence. On a deduplicated identifier-year holdout, the model achieves 95.64% accuracy but only 41.67% recall, with AP 0.4821. Operating-system metadata does not provide consistent additional ranking value, and logistic regression exceeds LightGBM on ROC-AUC and AP. Case-level TreeSHAP analysis reveals both meaningful attack-related terms and fragile lexical associations. These findings support a restricted CVSS-severity classification study; they do not validate vulnerability exploitability prediction.

Future work follows directly from the observed limitations. First, an exploitation study must collect independently sourced events, distinguish public exploit code from confirmed exploitation, record disclosure and observation dates, and define a prediction horizon before fitting models. Non-observed cases should not automatically be treated as proven negatives. Training on past disclosures and testing on later disclosures must use feature snapshots available at the prediction time; this is the prerequisite for restoring an exploitation claim.

Second, the operating-system hypothesis requires normalized CPE or product/version mappings, explicit missingness provenance and vendor- or product-group evaluation. With 67.20% missing OS fields and an inconclusive paired AP difference, expanding environmental features without validating their coverage is not justified. Third, the false-negative analysis motivates comparing stop-word filtering and richer text representations within training-only validation, while assessing related-advisory overlap. Threshold selection should target recall and false-positive cost on a validation set rather than optimize the final test set. A larger independent positive test sample is needed before drawing stable deployment conclusions.

References

[1] FIRST. Common Vulnerability Scoring System v3.1: Specification Document[Z/OL]. https://www.first.org/cvss/v3-1/specification-document. Accessed 19 September 2026.

[2] FIRST. Common Vulnerability Scoring System v3.1: User Guide[Z/OL]. https://www.first.org/cvss/v3-1/user-guide. Accessed 19 September 2026.

[3] JACOBS J, ROMANOSKY S, EDWARDS B, et al. Exploit Prediction Scoring System (EPSS). Digital Threats: Research and Practice, 2021, 2(3): 1-17.

[4] JACOBS J, ROMANOSKY S, SUCIU O, et al. Enhancing Vulnerability Prioritization: Data-Driven Exploit Predictions with Community-Driven Insights[Z/OL]. arXiv:2302.14172, 2023.

[5] SUCIU O, NELSON C, LYU Z, et al. Expected Exploitability: Predicting the Development of Functional Vulnerability Exploits//31st USENIX Security Symposium. Berkeley, CA: USENIX Association, 2022: 377-394.

[6] KE G, MENG Q, FINLEY T, et al. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Advances in Neural Information Processing Systems, 2017, 30.

[7] LIU Z. Explainable Machine Learning for Telecom Customer Churn Prediction and Actionable Retention Strategies. Applied Artificial Intelligence Research, 2026, 2(1).

[8] WANG X. A Case Study of the Influence of Multifarious Factors on Traffic Flow Forecasting. Applied Artificial Intelligence Research, 2026, 2(2).

[9] CVE Program. Frequently Asked Questions, archived CVE website: CVE identifier year[Z/OL]. https://www.cve.org/Resources/Media/Archives/OldWebsite/about/faqs.html. Accessed 19 September 2026.

[10] LUNDBERG S M, LEE S I. A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems, 2017, 30.