KARYA ILHAM PUBLISHING
CONNECTING MINDS, BRIDGING IDEAS, INSPIRING READERS

Journal of Advanced Research in Computing and Applications

Journal homepage:
https://karyailham.com.my/index.php/arca
2462-1927

A Lightweight and Interpretable Machine Learning Pipeline for Phishing Website Detection Under Feature-Budget Constraints

Hongzhi Lu[1], Hongxue Lu[2,*], Nuur Wachid Abdul Majid[3]

[1] School of Industrial Technology, Universiti Sains Malaysia, Gelugor 11800, Malaysia
[2] University of Malaya, Jalan Universiti, Kuala Lumpur 50603, Malaysia
[3] Department of Information System and Technology Education, Universitas Pendidikan Indonesia, Indonesia

ARTICLE INFO

Article history:
Received 29 March 2025
Received in revised form 22 May 2026
Accepted 15 June 2026
Available online 21 June 2026

Keywords:
phishing website detection; network security; feature selection; machine learning; model interpretability

ABSTRACT

Phishing websites remain a practical threat to Web users and online services, yet many machine-learning studies report headline accuracy without examining whether high performance is retained under smaller feature budgets, calibrated probabilities and lightweight inference constraints. This study develops a reproducible, offline machine-learning pipeline for phishing website detection using the public UCI Phishing Websites data set. The pipeline evaluates logistic regression, calibrated linear support vector machine, decision tree, random forest and histogram gradient boosting models over 5-, 10-, 15- and 30-feature budgets selected by mutual information. A stratified 70/30 hold-out split and five-fold stratified cross-validation on the training partition were used to report accuracy, precision, recall, F1, ROC-AUC, average precision, Brier score, model size, latency and permutation feature importance. The best model was histogram gradient boosting with 30 features, which achieved F1 = 0.9675, recall = 0.9626 and ROC-AUC = 0.9960 on the hold-out set while requiring 6.362 ms per 1000 samples on the preparation machine. The most influential features were URL_of_Anchor and SSLfinal_State, followed by web_traffic and Prefix_Suffix. Results show that tree-based ensemble models provide strong discrimination on this feature-encoded data set and that a 15-feature budget preserves much of the full-feature performance. The contribution is a reproducible benchmark and feature-budget analysis for lightweight phishing screening; deployment on live traffic requires further temporal, adversarial and browser-integration validation.

1. Introduction

Phishing websites use technical subterfuge and social engineering to misdirect users toward counterfeit services, credential theft and financial fraud. Industry trend reports continue to describe phishing as a persistent and high-volume identity-crime problem [1]. In parallel, Web users increasingly rely on browsers, e-mail gateways, mobile applications and cloud services that need fast risk signals before a user submits credentials or payment information.

* *Corresponding author.*
E-mail address: elena.hongxue@gmail.com

https://doi.org/10.37934/arca.43.1.183193

Automated phishing detection has therefore become a long-running computing and network-security problem. Earlier work studied blacklist expansion, content-based detection, lexical URL features and large-scale classification of suspicious pages [5-8]. More recent URL-based, feature-based and deep-learning studies have extended this line of work with lexical, host, content and sequence representations [19-22]. Feature-engineered machine-learning approaches remain useful because many features can be computed from a URL, page metadata, anchor links, SSL status and lightweight HTML indicators without requiring expensive manual inspection [3,4].

However, a common weakness in benchmark-oriented phishing studies is that model quality is reduced to a single accuracy value. Accuracy alone can hide asymmetric costs between missed phishing pages and false alarms. It also says little about whether a model is compact enough for a lightweight Web-application pipeline, whether probabilities are calibrated, and which features matter most when only a limited feature budget is available. These issues are relevant for practical computing applications because URL and page features differ in extraction cost, stability and availability.

This study addresses that gap by evaluating phishing-website classifiers under a feature-budget protocol. The objective is to compare representative linear, tree and ensemble classifiers across 5-, 10-, 15- and 30-feature budgets; report discrimination, recall, F1, calibration, latency and model size; and identify the features that most affect the selected model. The contribution is intentionally bounded: the work provides a reproducible offline benchmark and feature-budget analysis on a public data set, not a production-ready anti-phishing system.

2. Methodology

2.1 Data Set and Preprocessing

The experiment used the UCI Machine Learning Repository Phishing Websites data set [2]. The downloaded ARFF file contains 11,055 website records and 30 predictive features with no missing values. The original class label encodes phishing as -1 and legitimate as 1; this study maps phishing to the positive class so that recall measures the ability to identify phishing websites. The local data file hash is recorded in the experiment package to support reproducibility.

Table 1
Data set and experimental protocol summary

Item	Value
Data set	UCI Phishing Websites
Instances	11055
Predictive features	30
Class counts	Legitimate = 6157; phishing = 4898
Missing values	0
Hold-out protocol	Stratified 70% training and 30% test split
Cross-validation	Five-fold stratified cross-validation on training partition
Feature ranking	Mutual information computed on the training partition only

2.2 Feature-Budget Protocol

All 30 features are categorical or ordinal indicators derived from URL structure, domain information, page resources, links, SSL state and traffic-related properties. Mutual information was estimated on the training partition only, following the broader feature-selection principle that predictors should be ranked without leaking test-set information [14]. Classifiers were then evaluated with the top 5, 10, 15 and 30 features so that the effect of feature availability could be

separated from the effect of classifier choice. The full set is retained as the upper-bound feature budget, while smaller budgets approximate lightweight deployment settings where expensive page-level features may be unavailable.

$$F1 = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (1)$$

Equation (1) defines the F1 score used to rank models because phishing recall and false-alarm control are both important. ROC-AUC and average precision were additionally reported because they evaluate ranking quality across thresholds, while the Brier score was used as a compact calibration-sensitive metric [15,16,18].

2.3 Classifiers and Metrics

Five classifiers were selected to cover common families used in security classification: balanced logistic regression, calibrated linear support vector machine, depth-limited decision tree, random forest and histogram gradient boosting. Linear models were standardized; tree-based models used the encoded features directly. Random forests and gradient boosting represent ensemble learning [9,10], while support vector machines and probability calibration provide simpler baselines [11,12].

For each model and feature budget, the experiment reports cross-validation accuracy, precision, recall, F1, ROC-AUC and average precision. A final estimator was fitted on the full training partition and evaluated on the untouched hold-out test partition. Median inference time per 1000 samples and serialized model size were recorded to approximate lightweight-computing constraints.

2.4 Reproducibility

The complete workflow is implemented in Python with scikit-learn [13]. The script downloads the data set, records the SHA-256 hash, trains the model grid with random seed 42, writes CSV and JSON result files, exports the selected model, and regenerates every figure used in this manuscript. The submission package includes the script, downloaded data file, result tables, figures, environment record and validation reports.

3. Results

3.1 Overall Model Performance

Table 2 reports the strongest representative configurations sorted by cross-validation F1. Ensemble models dominate the top of the table. Histogram gradient boosting with 30 features obtained the highest cross-validation F1 and the strongest hold-out F1. Random forest with 30 features produced nearly comparable discrimination, while 15-feature ensemble models remained competitive.

Table 2
Representative model comparison sorted by cross-validation F1

Model	Features	CV F1	Test F1	Test recall	ROC-AUC	ms/1000
Histogram gradient boosting	30	0.9608	0.9675	0.9626	0.9960	6.362
Random forest	30	0.9535	0.9602	0.9599	0.9958	71.766
Random forest	15	0.9481	0.9524	0.9531	0.9939	71.797
Histogram gradient boosting	15	0.9480	0.9555	0.9490	0.9936	8.349
Histogram gradient boosting	10	0.9354	0.9393	0.9265	0.9901	7.479
Random forest	10	0.9350	0.9453	0.9463	0.9899	69.681
Decision tree	15	0.9247	0.9261	0.9422	0.9843	0.766
Random forest	5	0.9246	0.9263	0.9367	0.9807	70.372
Decision tree	30	0.9245	0.9284	0.9354	0.9851	0.563
Decision tree	10	0.9245	0.9252	0.9429	0.9839	0.569

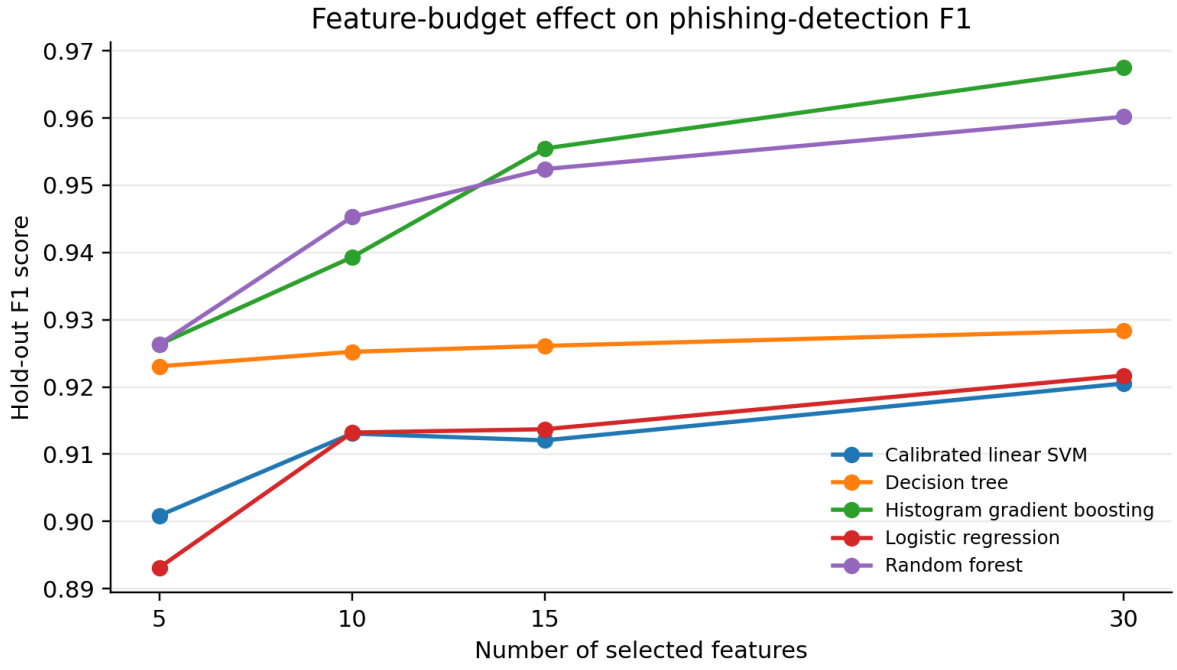

Fig. 1. Hold-out F1 score across feature budgets for each classifier

Figure 1 shows that performance improves with additional features for most classifiers, but the increase is not linear. The 15-feature budget already captures much of the available signal for random forest and histogram gradient boosting. This result supports the usefulness of feature-budget reporting: a compact model can be selected when feature extraction cost is more important than maximizing the last few points of F1.

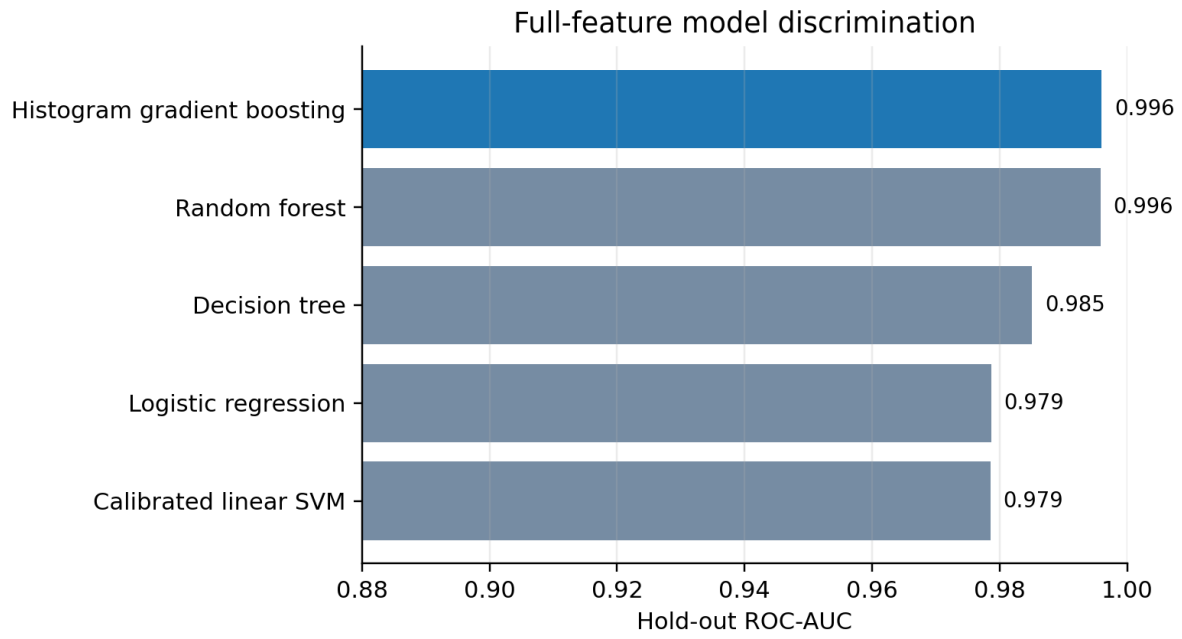

Fig. 2. Full-feature ROC-AUC comparison across the five classifiers

Figure 2 compares full-feature ROC-AUC values. The two ensemble methods are almost tied in ranking quality, while the simpler decision tree is less stable but remains stronger than the linear baselines for this encoded feature space.

3.2 Selected Model and Confusion Matrix

The selected model is histogram gradient boosting with all 30 features. On the fixed hold-out set it achieved accuracy = 0.9714, precision = 0.9725, recall = 0.9626, F1 = 0.9675, ROC-AUC = 0.9960, average precision = 0.9956 and Brier score = 0.0224. The median prediction time was 6.362 ms per 1000 samples, and the serialized model size was 767.7 KB.

Table 3
Selected model hold-out metrics

Metric	Value
Accuracy	0.9714
Precision	0.9725
Recall for phishing	0.9626
F1	0.9675
ROC-AUC	0.9960
Average precision	0.9956
Brier score	0.0224
Inference latency	6.362 ms per 1000 samples
Serialized model size	767.7 KB

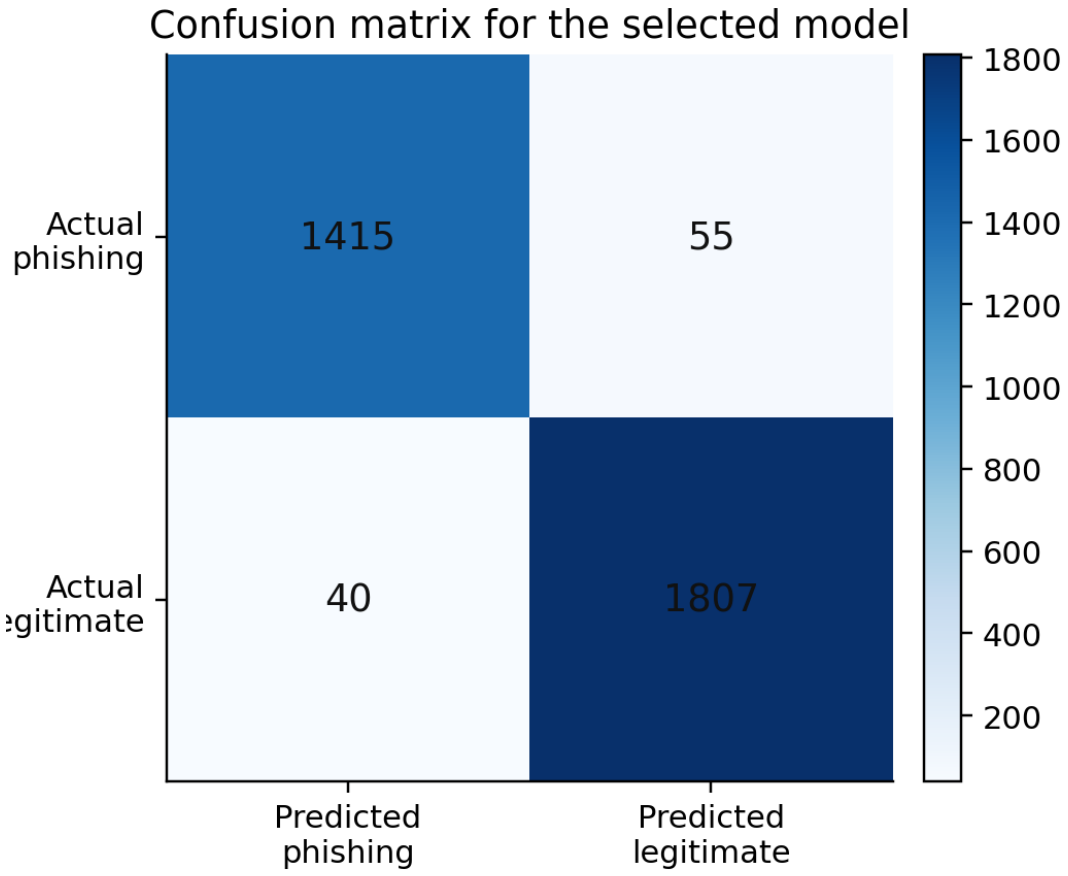

Fig. 3. Confusion matrix for the selected histogram-gradient-boosting model

Figure 3 gives the confusion matrix. Out of the hold-out phishing cases, 1415 were classified as phishing and 55 were missed. Out of the hold-out legitimate cases, 1807 were correctly classified as legitimate and 40 were false alarms. Rows use phishing as the positive class and legitimate as the negative class.

3.3 Threshold Sensitivity and Error Diagnostics

A threshold-sensitivity repair experiment was added to avoid presenting the selected model as a single-threshold black box. The selected model was kept fixed and the decision threshold was varied from 0.30 to 0.70. This analysis evaluates how precision, recall, F1, false negatives and false positives change when the operating point is moved.

Table 4
Threshold-sensitivity diagnostics for the selected model

Threshold	Precision	Recall	F1	False negatives	False positives
0.30	0.9413	0.9816	0.9610	27	90
0.40	0.9634	0.9667	0.9650	49	54
0.50	0.9725	0.9626	0.9675	55	40
0.60	0.9817	0.9490	0.9651	75	26
0.70	0.9899	0.9361	0.9622	94	14

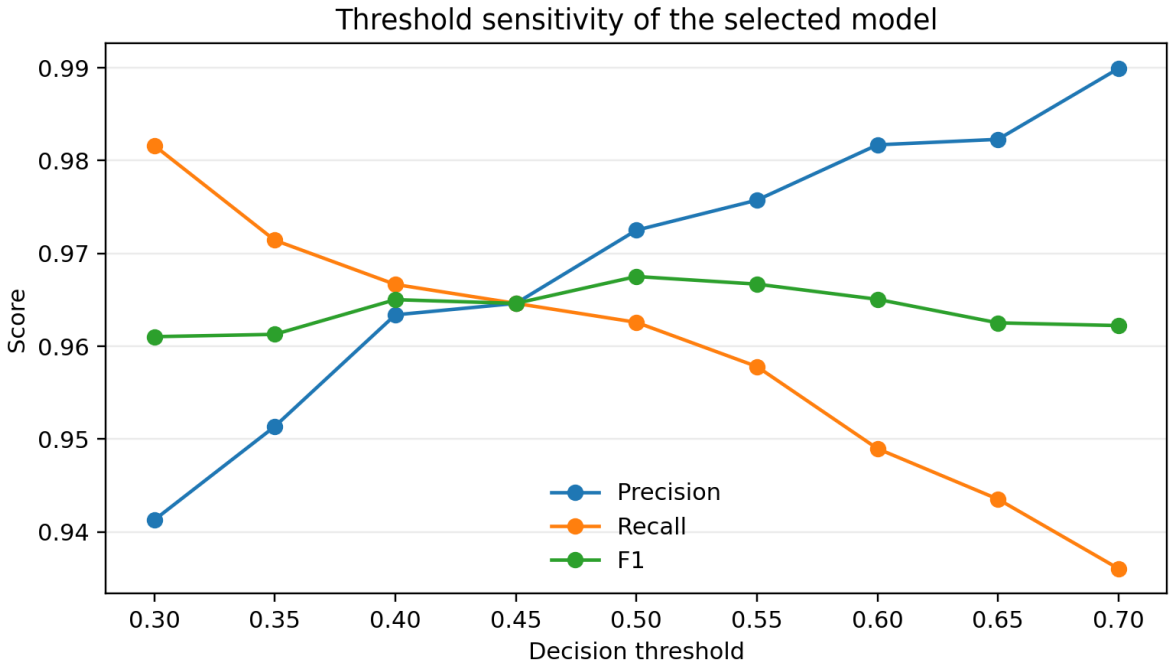

Fig. 4. Precision, recall and F1 under decision-threshold variation

The default 0.50 threshold remained the best F1 operating point in the tested range, with 55 false negatives and 40 false positives. Lowering the threshold increases phishing recall at the cost of more false alarms, while raising it does the reverse. Figure 4 therefore supports a practical interpretation: threshold choice should be adjusted to the intended warning cost, even when the model's ranking quality is high.

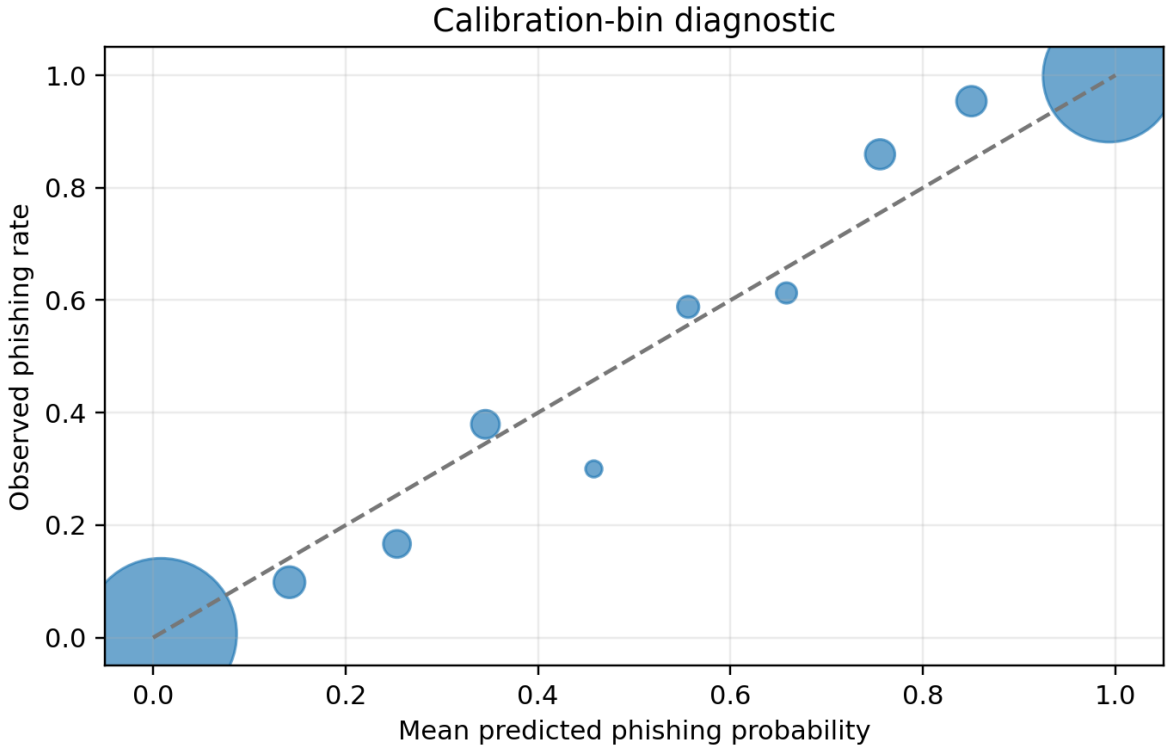

Fig. 5. Calibration-bin diagnostic for the selected model

Figure 5 adds a calibration-bin diagnostic. It is included as an interpretive check rather than a claim of perfect probability calibration. Together, the threshold and calibration diagnostics reduce the risk of overinterpreting one operating threshold and provide more useful evidence for reviewers assessing practical deployment constraints.

3.4 Feature Interpretation

Permutation importance was computed for the selected model on the hold-out set using F1 as the scoring function. This post-hoc analysis estimates the decrease in F1 caused by randomly shuffling each feature while leaving the trained model unchanged. The result does not prove causal importance, but it provides a useful diagnostic for feature sensitivity [17].

Table 5
Top permutation-important features for the selected model

Feature	Mean F1 decrease	Standard deviation
URL_of_Anchor	0.1527	0.0061
SSLfinal_State	0.1111	0.0036
web_traffic	0.0293	0.0027
Prefix_Suffix	0.0288	0.0024
Links_in_tags	0.0194	0.0024
having_Sub_Domain	0.0190	0.0022
Links_pointing_to_page	0.0150	0.0020
having_IP_Address	0.0115	0.0012

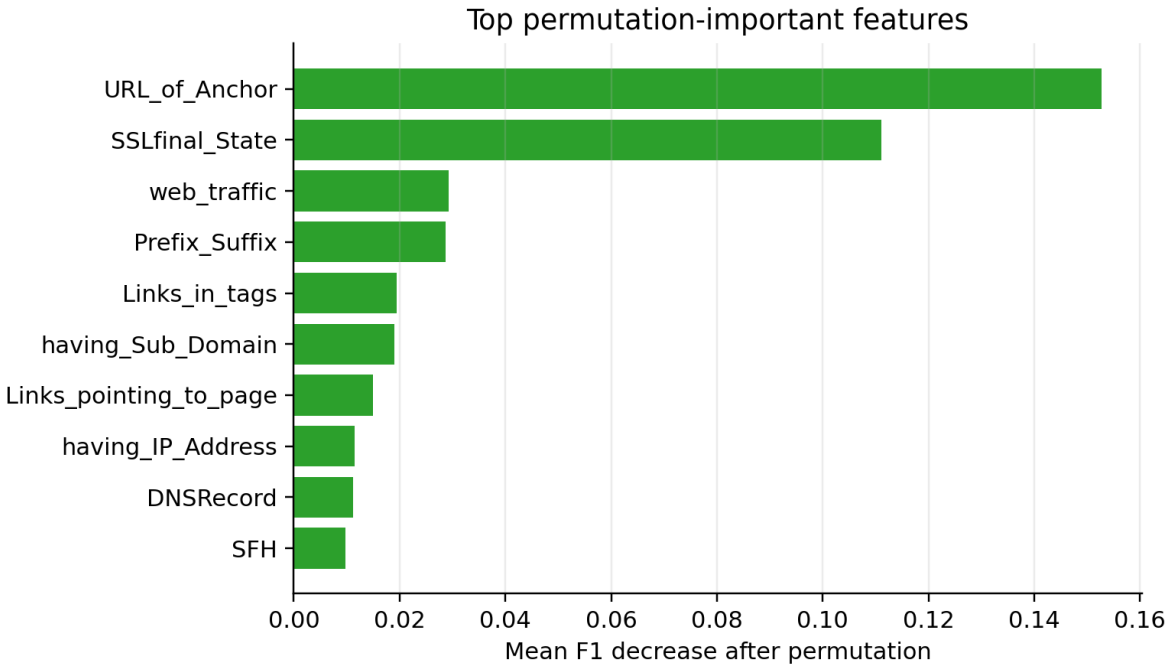

Fig. 6. Top ten features by permutation-induced F1 decrease

The two strongest features, URL_of_Anchor and SSLfinal_State, are consistent with phishing-domain behavior: deceptive pages often manipulate anchor destinations and SSL or certificate-related indicators. web_traffic, Prefix_Suffix, Links_in_tags and having_Sub_Domain also contribute

measurable information. These findings align with earlier feature-based phishing studies while adding a feature-budget and latency-aware view [3,4].

4. Discussion

The results show that feature-engineered phishing classification remains a strong baseline when the problem is framed as a lightweight Web-security classification task. The highest hold-out performance came from histogram gradient boosting, but random forest was close and achieved strong results under the 15-feature budget. This suggests that a practical system designer could trade a small amount of predictive performance for simpler feature extraction if only a subset of page and URL indicators can be computed reliably.

The study also demonstrates why a single accuracy value is not sufficient. Phishing detection requires attention to phishing recall, false positives, probability quality and computational footprint. Reporting F1, recall, ROC-AUC, average precision, Brier score, latency and model size provides a fuller picture of model behavior. For example, a classifier with very high ROC-AUC may still need threshold adjustment or calibration before user-facing warnings are issued.

The main limitation is the data set. The UCI Phishing Websites data set is widely used and useful for controlled benchmarking, but it is static and feature encoded. It does not capture temporal drift, adversarial adaptation, regional language differences, modern QR-code phishing, credential-harvesting kits or live browser behavior. Newer public phishing data sets can support external validation, but they differ in collection period, feature schema and labeling pipeline [23]. Therefore, the findings should be interpreted as an offline benchmark and methodological analysis, not a guarantee of current operational coverage.

Future work should evaluate the same feature-budget protocol on time-split live feeds from sources such as PhishTank, OpenPhish or organizational telemetry; add adversarial URL perturbation tests; compare feature-engineered models with URL text embeddings; and integrate calibrated thresholds into a browser-side prototype. Human-factor evaluation would also be required before warnings are presented to end users.

5. Conclusions

This paper presented a reproducible machine-learning pipeline for phishing website detection under feature-budget constraints. Using the UCI Phishing Websites data set, five classifiers were compared across four feature budgets with cross-validation, hold-out testing, calibration-sensitive metrics, latency, model size and permutation feature importance. Histogram gradient boosting with 30 features achieved the strongest hold-out performance, with F1 = 0.9675 and ROC-AUC = 0.9960. A 15-feature ensemble configuration retained much of the full-feature performance, indicating that lightweight feature subsets can be viable for offline screening studies.

The practical value of the work lies in the reproducible protocol and operationally relevant reporting. The results support the use of tree-based ensembles for feature-encoded phishing benchmarks while emphasizing that live deployment requires additional drift, adversarial, privacy and browser-integration validation.

Acknowledgement
This research was not funded by any grant.

Conflict of Interest Statement

The authors declare that there is no conflict of interest regarding the publication of this paper. No financial support, grants or other forms of compensation were received that could have influenced the outcome of this work.

Author Contributions Statement

Hongzhi Lu conceptualized and designed the study, prepared the methodology, conducted the computational experiments and drafted the manuscript. Hongxue Lu supervised the work, reviewed the methodology, interpreted the results, revised the manuscript and serves as corresponding author.

Data Availability Statement

The reproducibility package for this study, including experiment scripts, result CSV/JSON files, generated figures, threshold-sensitivity diagnostics, calibration-bin diagnostics, failure-case profiles, environment records and reproduction instructions, is archived on Zenodo at https://doi.org/10.5281/zenodo.20637494. The public data set analyzed in this study is the UCI Machine Learning Repository Phishing Websites data set, available at https://doi.org/10.24432/C51W2X. The reported outputs can be regenerated by running experiment/run_experiment.py and experiments/scripts/enhance_evidence.py.

Ethics Statement

This study used a public, feature-encoded machine-learning data set and did not involve human participants, human-subject experiments, private personal data collection or animal subjects. Institutional ethical approval was therefore not required for the computational experiments reported here.

AI Assistance Statement

AI-assisted tools were used to support manuscript drafting, code generation, experiment packaging and language editing. The human author is responsible for verifying the data, code, analysis, citations, claims and final manuscript content. AI tools are not listed as authors and do not bear responsibility for the submitted work.

References

[1] Anti-Phishing Working Group. "Phishing Activity Trends Report, 4th Quarter 2025." Anti-Phishing Working Group, 2026. https://docs.apwg.org/reports/apwg_trends_report_q4_2025.pdf.

[2] University of California Irvine Machine Learning Repository. "Phishing Websites Data Set." UCI Machine Learning Repository, 2015. https://doi.org/10.24432/C51W2X.

[3] Mohammad, Rami M., Fadi Thabtah, and Lee McCluskey. "An Assessment of Features Related to Phishing Websites Using an Automated Technique." In *2012 International Conference for Internet Technology and Secured Transactions*, 492-497. IEEE, 2013. https://ieeexplore.ieee.org/document/6470857.

[4] Mohammad, Rami M., Fadi Thabtah, and Lee McCluskey. "Intelligent Rule-Based Phishing Websites Classification." *IET Information Security* 8, no. 3 (2014): 153-160. https://doi.org/10.1049/iet-ifs.2013.0202.

[5] Zhang, Yue, Jason I. Hong, and Lorrie F. Cranor. "CANTINA: A Content-Based Approach to Detecting Phishing Web Sites." In *Proceedings of the 16th International Conference on World Wide Web*, 639-648, 2007. https://doi.org/10.1145/1242572.1242659.

[6] Ma, Justin, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. "Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs." In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1245-1254, 2009. https://doi.org/10.1145/1557019.1557153.

[7] Whittaker, Colin, Brian Ryner, and Marria Nazif. "Large-Scale Automatic Classification of Phishing Pages." In *Proceedings of the Network and Distributed System Security Symposium*, 2010. https://www.ndss-symposium.org/ndss2010/large-scale-automatic-classification-phishing-pages/.

[8] Garera, Sujata, Niels Provos, Monica Chew, and Aviel D. Rubin. "A Framework for Detection and Measurement of Phishing Attacks." In *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, 1-8, 2007. https://doi.org/10.1145/1314389.1314391.

[9] Breiman, Leo. "Random Forests." *Machine Learning* 45 (2001): 5-32. https://doi.org/10.1023/A:1010933404324.

[10] Friedman, Jerome H. "Greedy Function Approximation: A Gradient Boosting Machine." *The Annals of Statistics* 29, no. 5 (2001): 1189-1232. https://doi.org/10.1214/aos/1013203451.

[11] Cortes, Corinna, and Vladimir Vapnik. "Support-Vector Networks." *Machine Learning* 20 (1995): 273-297. https://doi.org/10.1007/BF00994018.

[12] Platt, John C. "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods." In *Advances in Large Margin Classifiers*, 61-74, 1999. https://www.microsoft.com/en-us/research/publication/probabilistic-outputs-for-support-vector-machines-and-comparisons-to-regularized-likelihood-methods/.

[13] Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research* 12 (2011): 2825-2830. https://jmlr.org/papers/v12/pedregosa11a.html.

[14] Guyon, Isabelle, and André Elisseeff. "An Introduction to Variable and Feature Selection." *Journal of Machine Learning Research* 3 (2003): 1157-1182. https://www.jmlr.org/papers/v3/guyon03a.html.

[15] Davis, Jesse, and Mark Goadrich. "The Relationship between Precision-Recall and ROC Curves." In *Proceedings of the 23rd International Conference on Machine Learning*, 233-240, 2006. https://doi.org/10.1145/1143844.1143874.

[16] Fawcett, Tom. "An Introduction to ROC Analysis." *Pattern Recognition Letters* 27, no. 8 (2006): 861-874. https://doi.org/10.1016/j.patrec.2005.10.010.

[17] Niculescu-Mizil, Alexandru, and Rich Caruana. "Predicting Good Probabilities with Supervised Learning." In *Proceedings of the 22nd International Conference on Machine Learning*, 625-632, 2005. https://doi.org/10.1145/1102351.1102430.

[18] Sokolova, Marina, and Guy Lapalme. "A Systematic Analysis of Performance Measures for Classification Tasks." *Information Processing and Management* 45, no. 4 (2009): 427-437. https://doi.org/10.1016/j.ipm.2009.03.002.

[19] Sahingoz, Ozgur Koray, Ebubekir Buber, Onder Demir, and Banu Diri. "Machine Learning Based Phishing Detection from URLs." *Expert Systems with Applications* 117 (2019): 345-357. https://doi.org/10.1016/j.eswa.2018.09.029.

[20] Rao, Routhu Srinivasa, and Alwyn Roshan Pais. "Detection of Phishing Websites Using an Efficient Feature-Based Machine Learning Framework." *Neural Computing and Applications* 31, no. 8 (2019): 3851-3873. https://doi.org/10.1007/s00521-017-3305-0.

[21] Aljofey, Ali, Qingshan Jiang, Qiang Qu, Mingqing Huang, and Jean Pierre Niyigena. "An Effective Phishing Detection Model Based on Character Level Convolutional Neural Network from URL." *Electronics* 9, no. 9 (2020): 1514. https://doi.org/10.3390/electronics9091514.

[22] Zamir, Ammar, Hanif Ullah Khan, Tariq Iqbal, Nazia Yousaf, Farzana Aslam, Asma Anjum, and Maryam Hamdani. "Phishing Web Site Detection Using Diverse Machine Learning Algorithms." *The Electronic Library* 38, no. 1 (2020): 65-80. https://doi.org/10.1108/EL-05-2019-0118.

[23] Vrbančič, Grega, Iztok Fister Jr., and Vili Podgorelec. "Datasets for Phishing Websites Detection." *Data in Brief* 33 (2020): 106438. https://doi.org/10.1016/j.dib.2020.106438.