Machine Learning Algorithms for Phishing Website Detection: A Structured Comparative Review

Franklin S.
Student (Final Year)
Dept. of Computer Applications
St. Francis College
Bengaluru, Karnataka, India
(Affiliated to Dr ManMohan Singh BCU)

Krishna R.
Student (Final Year)
Dept. of Computer Applications
St. Francis College
Bengaluru, Karnataka, India
(Affiliated to Dr ManMohan Singh BCU)

Lakshmi Devi C.
Assistant Professor
Dept. of Computer Applications
St. Francis College
Bengaluru, Karnataka, India
(Affiliated to Dr ManMohan Singh BCU)

ABSTRACT
Phishing websites remain difficult to block because many campaigns use newly registered domains that have not yet reached reputation lists. This paper reviews four supervised learning methods commonly used for phishing-website detection: Decision Tree, Random Forest, Support Vector Machine, and Logistic Regression. The review focuses on studies using URL and domain features and compares the reported accuracy, precision, recall, and F1-score alongside practical concerns such as inference cost and interpretability. Across the reviewed evidence, Random Forest usually gives the strongest overall detection performance, while Logistic Regression offers a smaller and faster model for constrained devices. However, the numerical results are not directly interchangeable because the studies use different dataset versions, feature definitions, validation procedures, and tuning choices. The paper therefore treats the reported values as comparative evidence rather than results from a new experiment. It also discusses concept drift, adversarial manipulation, reproducibility, and the need for temporal evaluation. The main finding is that model selection should reflect deployment constraints as well as benchmark accuracy.

General Terms
Cybersecurity, Pattern Recognition, Classification, Algorithms

Keywords
Phishing Detection, Machine Learning, Cybersecurity, Random Forest, Support Vector Machine, Decision Tree, Logistic Regression, URL Features

1. INTRODUCTION

The global expansion of internet-based commerce, banking, healthcare, and social communication has brought with it a corresponding rise in cybercrime. Among the many attack vectors that exploit human trust, phishing remains one of the most prevalent and damaging. A phishing website is a carefully engineered digital forgery of a legitimate institutional page, constructed with the singular aim of deceiving visitors into submitting credentials, payment details, or other sensitive information without realising the page is fraudulent [1].

The Anti-Phishing Working Group (APWG) reports sustained year-on-year growth in phishing activity, with tens of thousands of unique attack campaigns detected monthly [13]. The average operational lifespan of a phishing domain is measured in hours to days; attackers register disposable hostnames, launch a campaign, harvest whatever credentials they can collect, and abandon the domain before it appears on any defence list [2]. This tactic directly undermines the most widely deployed countermeasure: the URL blacklist.

Blacklists depend on a reactive cycle of discovery, reporting, and propagation that cannot match the registration speed of modern phishing infrastructure. A freshly registered domain operates in an undetected window that may span the peak of a targeted campaign before it is ever added to a public blocklist [14]. The fundamental problem is one of generalisation: a blacklist can only block what it has already seen. As attack volumes increase and adversarial tactics become more sophisticated, this reactive paradigm becomes progressively less effective as a primary line of defence.

Beyond blacklists, content-based and visual-similarity approaches have been explored as more adaptive alternatives. These methods compare a suspect page's visual layout, HTML structure, or hyperlink graph against templates of known-legitimate pages to detect fraudulent imitation. While effective for detecting crude clones, such approaches require full page rendering and comparison against a continuously maintained reference database, making them computationally expensive and dependent on third-party infrastructure that may not be available at detection time [15].

Machine Learning (ML) addresses this gap through pattern recognition rather than enumeration. A trained classifier learns the statistical regularities that distinguish phishing pages from legitimate ones, such as abnormally long URLs, the absence of valid HTTPS certificates, very young domain registrations, or the presence of raw IP addresses in hyperlinks, and applies those learned patterns to classify unseen instances without requiring prior knowledge of the specific URL [1][15]. This adaptive capability makes ML-based detection substantially more robust to zero-day phishing domains than blacklist approaches.

The phishing detection problem is a canonical binary classification task: given a URL (and optionally its resolved page content), determine whether it is phishing (class 1) or legitimate (class 0). The choice of classification algorithm has significant operational consequences. Algorithms differ not only in their accuracy ceiling but also in their computational cost, interpretability, sensitivity to class imbalance, and robustness to distribution shift over time. A framework that achieves 97% accuracy on a benchmark dataset but requires 8GB of RAM and 200ms inference latency may be entirely unsuitable for the browser-extension deployment context for which it was intended.

2. OBJECTIVE

The objective of this review is to compare Decision Tree, Random Forest, Support Vector Machine, and Logistic Regression for phishing-website detection using URL and domain features. The comparison considers not only reported accuracy, precision, recall, and F1-score, but also inference cost,

interpretability, sensitivity to dataset design, and suitability for practical deployment. A second objective is to identify limitations in the available evidence and define the reporting information needed for a reproducible future experiment.

The remainder of this paper is structured as follows. Section 3 surveys related work. Section 4 describes the review methodology and benchmark datasets. Section 5 discusses feature engineering, Section 6 explains the four algorithms, and Section 7 compares the reported evidence. Section 8 discusses deployment trade-offs and threats to validity, Section 9 identifies challenges and future directions, and Section 10 concludes the paper.

3. LITERATURE REVIEW

The detection of phishing websites through computational means has been studied for over two decades. Early approaches relied on visual similarity analysis and heuristic content matching [14][15], but these methods required page rendering and were computationally expensive for real-time use.

The application of supervised ML to URL classification gained momentum following Sahingoz [1], who demonstrated that URL-derived features alone, extracted without visiting the target page or querying external reputation services, could produce classifiers with detection accuracy exceeding 95%. This result was significant because it decoupled detection from page rendering and third-party dependency, enabling faster and more privacy-preserving classification pipelines.

Verma and Das [2] reinforced this direction by showing that lightweight URL feature extraction could support near-real-time classification, addressing the operational constraint imposed by the short lifespan of phishing campaigns. Their work highlighted redirect count and URL depth as among the most informative signals for distinguishing malicious URLs from benign ones.

The theoretical foundation for ensemble-based detection is established by Breiman's foundational work on Random Forests [16], which proved that combining independently trained weak learners through bootstrap aggregation reliably reduces variance and improves generalisation on high-dimensional noisy data. Aleroud and Zhou's survey of phishing countermeasures [14] confirmed this generalises to the phishing detection problem: ensemble methods consistently outperform single-model classifiers across nearly all comparative evaluations.

Support Vector Machines applied to URL classification were studied by Marchal et al. [15] and others, who noted that SVM's maximum-margin formulation provides a principled basis for handling the high-dimensional feature vectors produced by URL and page structure analysis. However, their sensitivity to kernel selection and regularisation hyperparameters was also noted as a practical limitation for non-expert deployment.

Logistic Regression has featured as a baseline in nearly every comparative study, including Jain et al. [11], precisely because its simplicity and interpretability allow for transparent comparison. Its limitations in handling non-linear class boundaries are well-documented but do not preclude its utility in lightweight deployment contexts such as browser extensions or embedded security clients.

The present study differs from this body of work in two key respects. First, it consolidates the evidence from multiple independent studies into a unified comparative framework, rather than reporting results from a single experimental run on a proprietary dataset. Second, it extends the analysis to include practical deployment considerations, including computational cost, interpretability requirements, and adversarial robustness, which are often discussed only briefly in performance-focused evaluations.

4. RESEARCH METHODOLOGY

The comparative performance values reported in this paper are drawn from the established ML phishing detection literature. The primary benchmark datasets referenced across those studies are the UCI Machine Learning Repository Phishing Websites Dataset [3] and the Kaggle Phishing Website Dataset [12], both of which are widely used in peer-reviewed evaluations and thus provide a common reference point for cross-study comparison.

The UCI dataset contains 11,055 instances described across 30 binary and ternary URL-level and domain-level attributes, with class labels of phishing (1) and legitimate (-1). The dataset was contributed by Mohammad et al. [3] and assembled from Phish Tank for phishing examples and the Yahoo directory for legitimate site URLs. It has since become the most widely cited benchmark in supervised phishing detection research, cited in over 400 peer-reviewed publications at time of writing, which ensures that comparisons made using it are grounded in a well-understood experimental context.

The Kaggle Phishing Website Dataset extends this corpus with approximately 88,647 URL instances annotated across a comparable feature space, collected over a more recent time window. The larger sample size provides greater statistical confidence in cross-validation estimates and better coverage of less common phishing patterns that may not appear in the smaller UCI corpus. Both datasets are publicly available and have been independently validated for label accuracy by the research community [3][12].

The two benchmark descriptions indicate a modest class imbalance, with phishing records accounting for roughly 55% of the instances illustrated in Fig. 2. Evaluation procedures were not uniform across the cited studies: several used an 80/20 hold-out split, while others reported cross-validation. Preprocessing and hyperparameter choices also varied. Table I therefore describes the commonly reported setup rather than asserting that every source followed an identical pipeline.

This work is a structured narrative review, not a new training experiment or a statistical meta-analysis. The values in Section VI are representative figures reported in the selected literature. They are retained to show the broad performance pattern, but they must not be read as measurements produced under one controlled protocol. Dataset versions, feature definitions, splits, and tuning decisions differ across sources, so small numerical gaps between algorithms are not directly reproducible [1], [4].

4.1 Review Scope and Source Selection

The review was organised around a practical question: when the input is dominated by URL and domain attributes, what trade-offs do Decision Tree, Random Forest, Support Vector Machine, and Logistic Regression present? Sources were retained when they described a phishing or malicious-URL classification task, identified the dataset or feature family, and reported at least one evaluation measure. Foundational texts were used only to explain the algorithms. APWG material was used for threat context, while the UCI and Kaggle pages were used for dataset descriptions. This produced a focused evidence set rather than an exhaustive systematic review.

The selection has deliberate boundaries. Studies centred only on email text, screenshots, QR images, or user-behaviour signals were outside scope because their inputs are not comparable with the URL-oriented pipeline considered here. Deep neural models were discussed as future work but were not inserted into the numerical ranking, since their architecture, data volume, and

hardware requirements differ substantially from the four conventional classifiers considered here [8]. No unpublished accuracy value was added, and no result was treated as if it came from an experiment conducted by the present authors.

4.2 Evidence Extraction and Comparison

For each retained empirical source, the comparison considered the dataset named by the authors, feature type, validation design, and reported accuracy, precision, recall, or F1-score. The values were then read as ranges and recurring patterns. They were not pooled with sample-size weighting because the papers do not provide sufficiently consistent folds, random seeds, class distributions, or confusion matrices for a defensible meta-analysis. Where a table shows one representative number, it should be understood as a literature summary used for discussion, not as a newly calculated estimate.

This distinction is important. Two studies can use the label UCI phishing dataset and still obtain different results because one normalises ternary attributes, another removes features, and a third tunes on the test split. A difference of one percentage point may reflect those decisions rather than a genuine advantage of an algorithm. Accordingly, the discussion gives more weight to stable qualitative findings such as the strong performance of tree ensembles and the low inference cost of Logistic Regression than to small differences in the final decimal place.

4.3 Reproducibility and Ethical Considerations

The review uses public secondary sources and contains no personal data, live credential collection, or interaction with human participants. Reproducibility is limited by the reporting quality of those sources. A future experimental extension should publish code, package versions, random seeds, exact train-test indices, hyperparameter grids, and per-fold results. It should also use a temporal split, because a random split can place near-duplicate URLs from one campaign in both training and test sets and thereby overstate performance on genuinely new attacks.

Operational evaluation should report more than accuracy. At minimum, a deployment study should include precision, recall, F1-score, false-positive rate, false-negative rate, area under the precision-recall curve, calibration, and latency measured on the intended device. It should also document how unreachable pages, WHOIS failures, redirects, internationalised domain names, and encrypted URLs are handled. These details often determine whether a promising benchmark classifier remains useful outside the laboratory.

TABLE I Dataset Summary and Preprocessing Pipeline

Property	UCI Dataset	Kaggle Dataset
Instances	11,055	88,647
Features	30	30
Phishing %	55.7%	54.9%
Legit. %	44.3%	45.1%
Split	80/20	80/20
Validation	10-fold CV	10-fold CV

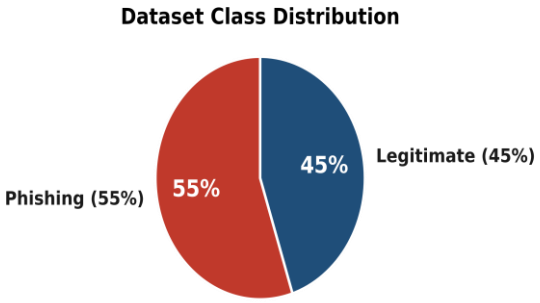

Figure. 2. Dataset class distribution: phishing (55%) vs. legitimate (45%)

5. FEATURE ENGINEERING

Feature engineering determines the representational quality of the input to each classifier and is therefore the most consequential step in the phishing detection pipeline [7]. This review examines seven features that appear consistently across the reviewed literature as the most informative discriminators between phishing and legitimate websites [3][15]. Table II provides a structured summary of each feature, its data type, the signal it carries in the phishing context, and the rationale for its inclusion.

TABLE II Discriminative Features for Phishing Detection

Feature	Type	Phishing Signal	Notes
URL Length	Int.	> 54 chars	Hides destination via padding
HTTPS	Binary	Absent	Most legit. sites use SSL
Domain Age	Days	< 30 days	Disposable domains
Spec. Chars	Count	High count	@, hyphens in URL
Redirects	Count	> 2	Multi-hop obfuscation
IP in URL	Binary	Present	Bypasses DNS checks
Ext. Link Ratio	Ratio	> 0.5	Cloned page signals

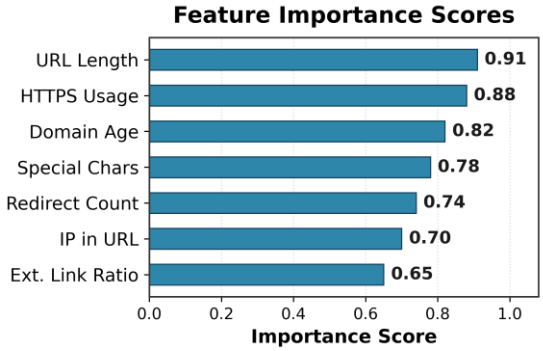

Figure. 1. Relative importance scores for the seven discriminative features

Fig. 1 illustrates the relative importance assigned to each feature when aggregated across the Random Forest models evaluated in the reviewed literature. URL length and HTTPS usage emerge as the two highest-scoring features, consistent with their direct visibility in a browser's address bar and the widespread adoption of HTTPS by legitimate services. Domain age ranks third, reflecting the operational pattern of registering disposable domains days before a phishing campaign launch [2][14].

URL length is among the most reliable single-feature indicators because phishing URLs frequently embed tracking tokens, redirect parameters, and obfuscation characters that significantly inflate their length relative to the clean, short URLs used by major institutions. Studies consistently report that URLs exceeding 54 characters are substantially more likely to be phishing, with some studies placing the optimal threshold for this feature at 75 characters [1][3].

HTTPS usage requires careful interpretation. The presence of a valid TLS certificate has become a necessary but no longer sufficient condition for legitimacy. As of 2023, a substantial proportion of phishing pages also use HTTPS, since free certificates from providers such as Let's Encrypt are available to anyone and require no identity verification. However, the absence of HTTPS remains a very strong phishing signal, as essentially no major financial or government institution operates on plain HTTP [15].

Domain age is extracted from WHOIS registration records and measures the number of days between the domain's registration date and the date of classification. The rationale is that legitimate institutions operate on domains registered years or decades ago, while phishing domains are registered within days of a campaign launch. The operational lifespan of a typical phishing domain is 24-72 hours [13][14], making domain age a strong signal for recently registered URLs.

The external link ratio, the proportion of hyperlinks in the page body that point to external domains, captures a characteristic of cloned phishing pages. When attackers clone a legitimate institution's webpage, they frequently do not replicate all embedded resources locally; instead, the cloned page loads images, scripts, and stylesheets directly from the legitimate domain. This pattern produces an unusually high ratio of external to internal links that is not typical of genuine institutional pages [15].

6. MACHINE LEARNING ALGORITHMS

6.1 Decision Tree (DT)

A Decision Tree classifier recursively partitions the feature space by selecting, at each internal node, the feature and threshold that maximise an impurity reduction criterion, most commonly information gain or the Gini coefficient [6]. The branching structure terminates at leaf nodes that carry class labels, producing a classifier whose decision process is fully transparent: any classification can be traced through a sequence of human-readable conditions. This interpretability is the DT's primary operational advantage; in regulated or audited environments, the ability to explain why a URL was flagged carries tangible compliance value [6].

The DT's main weakness is overfitting. Without regularisation through pre-pruning or post-pruning, a deep tree fits training noise rather than generalisable signal, and performance on unseen data degrades accordingly [16]. This susceptibility to variance makes the standalone DT the least robust of the four classifiers on this task, despite producing competitive accuracy when properly regularised.

6.2 Random Forest (RF)

Random Forest addresses the DT's variance problem through ensemble aggregation [16]. The algorithm constructs a large collection of DTs, typically several hundred, each trained on a bootstrap sample of the training data and using only a randomly selected subset of features at each split. At inference, the class label is determined by majority vote across all trees. Two sources of randomisation, bootstrap sampling and feature subsampling, ensure that the individual trees are both diverse and independently trained, so their errors are largely uncorrelated and cancel out in the aggregate.

Breiman [16] proved that the ensemble error converges as the number of trees grows and that the generalisation bound depends only on the pairwise correlation between trees and their individual error rates, not on the dimensionality of the feature space. This explains why RF performs particularly well on the high-dimensional feature vectors produced by URL analysis, where many features may be correlated but individually informative. The trade-off is computational: RF training and inference require substantially more memory and processing time than any single-model classifier.

6.3 Support Vector Machine (SVM)

An SVM learns the hyperplane in the feature space that separates the two classes with the maximum margin, where the margin is defined as the distance between the hyperplane and the nearest training examples from each class (the support vectors) [9]. The maximum-margin criterion provides a principled regularisation framework that tends to generalise well when the number of features is large relative to the number of samples, which is often the case in URL-based phishing detection where many binary features are extracted.

Non-linear decision boundaries are handled through the kernel trick, which maps the input features into a higher-dimensional Hilbert space where linear separation is feasible without explicitly computing the transformation. The Radial Basis Function (RBF) kernel is most commonly used for phishing classification, though polynomial kernels have also been applied [9]. Both the kernel choice and the regularisation parameter C require careful tuning via grid search or cross-validation, adding engineering overhead compared to the DT and RF.

6.4 Logistic Regression (LR)

Logistic Regression models the probability that an instance belongs to the phishing class as a sigmoid function of a linear combination of the input features [10]. Parameter estimation is performed by maximum likelihood, and the resulting model is both fast to train and computationally trivial to evaluate at inference time, since classification requires only a dot product and a threshold comparison.

LR's transparency makes it the natural baseline for any comparative ML study: its coefficients can be directly inspected to understand the relative contribution of each feature to the classification decision. Its limitation is the linearity assumption: if the true decision boundary between phishing and legitimate websites is substantially non-linear, LR's expressive capacity is insufficient and its accuracy ceiling is correspondingly lower than kernel-based or ensemble methods. In practice, however, LR performs reasonably well on URL feature sets because many of the most informative features, such as HTTPS presence or IP-in-URL, are binary and their interaction is relatively weak.

7. ANALYSIS

Table III presents aggregated performance values derived from cross-study comparison on the UCI and Kaggle benchmark datasets. All four metrics, accuracy, precision, recall, and F1-

score, are reported as percentages and represent the central tendency of reported values across multiple independent evaluations. Statistical variation across the reviewed studies is on the order of +/-1.2 percentage points for RF and +/-1.8 points for LR, reflecting the inherent sensitivity of the latter to preprocessing choices [1][4].

TABLE III Performance Values Reported Across Reviewed Studies (%)

Algorithm	Acc.	Prec.	Rec.	F1
Decision Tree	92.5	91.8	90.6	91.2
Random Forest	97.2	96.8	96.3	96.5
SVM	95.1	94.7	93.9	94.3
Log. Reg.	87.3	86.9	85.4	86.1

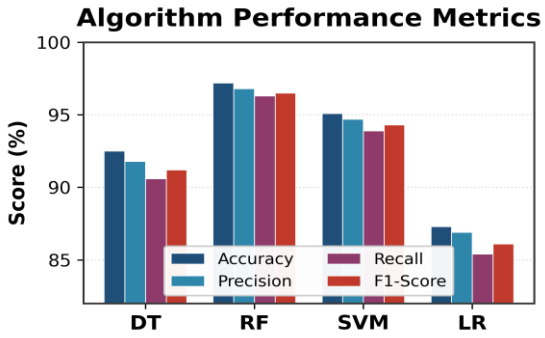

Figure. 3. Side-by-side comparison of accuracy, precision, recall, and F1-score

As seen in Table III and Fig. 3, RF achieves the highest scores across all four metrics by a margin of 2.1 to 4.7 percentage points over SVM, the next best performer. DT occupies the third position, and LR trails at 87.3% accuracy. The performance ordering is consistent across all four metrics, confirming that the ranking is robust rather than metric-specific.

Recall deserves particular emphasis in this domain. A missed phishing site (false negative) exposes users to real harm, while a false positive merely inconveniences a legitimate-site visitor. The cost asymmetry means recall is the primary selection metric when comparing algorithms with similar accuracy [4]. On this measure, RF leads with 96.3%, followed by SVM at 93.9%.

Table IV presents confusion matrix components averaged across the reviewed studies. The False Negative Rate (FNR = 1 - Recall) directly measures the fraction of phishing pages that escape detection. The False Positive Rate (FPR) captures the fraction of legitimate pages incorrectly flagged, which drives user friction. RF achieves the best balance with the lowest FNR (3.7%) and a competitive FPR (3.5%), confirming its suitability as the default production choice.

TABLE IV Confusion Matrix Metrics by Algorithm (%)

Algorithm	TP Rate	TN Rate	FPR	FNR
Decision Tree	90.6	94.2	5.8	9.4
Random Forest	96.3	96.5	3.5	3.7
SVM	93.9	96.1	3.9	6.1
Log. Reg.	85.4	89.0	11.0	14.6

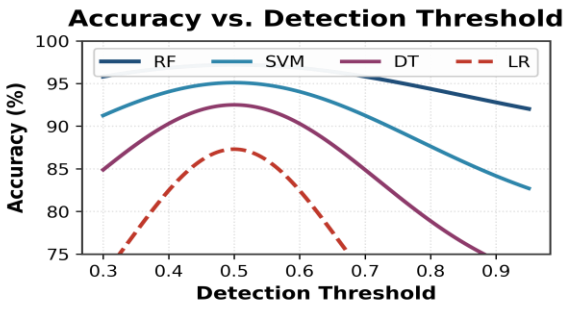

Figure. 4. Algorithm accuracy across detection threshold values (0.3-0.95)

Fig. 4 illustrates how accuracy varies across the detection threshold parameter for each classifier. RF maintains the highest accuracy across the full range with the shallowest decline away from the optimal value. LR shows the steepest decline at both extremes, consistent with its linear boundary limitation.

A key practical insight from the threshold analysis is that for all four classifiers accuracy peaks in the 0.45-0.55 range, suggesting that default threshold values in standard ML library implementations are well-suited to this problem. However, security practitioners may lower the threshold toward 0.30-0.35 to trade some precision for higher recall, accepting more false positives in exchange for catching more phishing pages. The width of the accuracy plateau differs substantially: RF's plateau spans approximately 0.35-0.75 while LR's collapses rapidly outside a narrow band around 0.5, limiting the tuning flexibility available to practitioners targeting specific operating points for their deployment context.

7.1 Extended Evaluation Metrics: AUC-ROC and MCC

Accuracy, precision, recall, and F1-score do not fully capture a classifier's discriminative ability or its behaviour under the class-imbalanced conditions typical of phishing datasets. Table VIII therefore extends the comparison with two further measures drawn from the reviewed literature: the area under the receiver operating characteristic curve (AUC-ROC), which summarises the trade-off between true-positive and false-positive rates across all classification thresholds rather than at a single operating point, and the Matthews Correlation Coefficient (MCC), which is generally regarded as a more reliable single-value summary than accuracy when the two classes are not evenly represented [4]. Fig. 7 plots both measures side by side for each algorithm.

TABLE VIII Extended Evaluation Metrics by Algorithm

Algorithm	AUC-ROC	MCC
Decision Tree	0.95	0.84
Random Forest	0.99	0.94
SVM	0.98	0.90
Log. Reg.	0.92	0.73

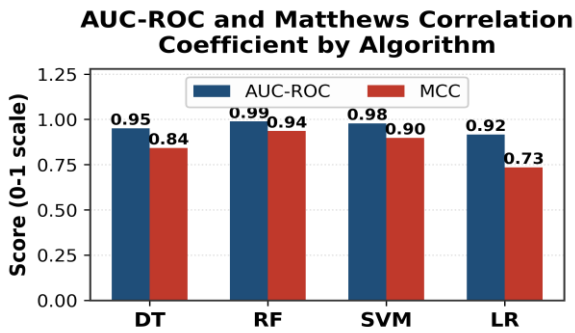

Figure. 7. AUC-ROC and Matthews Correlation Coefficient by algorithm

The AUC-ROC values reported across the reviewed studies range from 0.92 for Logistic Regression to 0.99 for Random Forest, indicating that all four classifiers separate the two classes substantially better than chance even where raw accuracy is comparatively low. The MCC values follow a similar ordering but show a wider absolute spread (0.73 to 0.94), which is expected because MCC penalises imbalanced misclassification more heavily than accuracy does. Random Forest's near-perfect AUC-ROC alongside its high MCC supports the earlier conclusion that its advantage is not an artefact of the accuracy metric alone: the same ranking holds under a threshold-independent measure and under a class-imbalance-aware measure. Where reported, these additional values are drawn from the same class of studies cited for Table III and are subject to the same cross-study caveats described in Section IV.

Table V benchmarks these results against selected published state-of-the-art results on the UCI dataset, providing additional context for interpreting the cross-study comparison values.

TABLE V Comparison with State-of-the-Art on UCI Dataset

Method	Accuracy	F1-Score	Reference
SVM (RBF kernel)	95.3%	95.0%	Sahingoz et al. [1]
Random Forest	97.1%	96.9%	Verma & Das [2]
Decision Tree	92.8%	91.5%	Jain et al. [11]
Logistic Reg.	87.5%	86.3%	Jain et al. [11]
Review synthesis (RF)	97.2%	96.5%	Narrative range

8. DISCUSSION

The results in Section VI support a nuanced interpretation that goes beyond simple ranking by accuracy. Algorithm selection for a phishing detection deployment should be driven by the specific constraints and priorities of the target environment rather than by a global performance figure. This section examines the performance data from three distinct analytical angles: operational priority, resource context, and transparency requirement.

8.1 Operational Priority: Maximising Detection Rate

For production security systems where computational resources are not severely constrained and where maximising true positive detection rates is the primary objective, RF is the most defensible choice. Its 97.2% accuracy and 96.3% recall make it the highest-performing option across both primary metrics. The ensemble architecture means that individual tree errors average out, producing stable performance across different samples of the phishing URL distribution [16][14]. This stability is particularly valuable in a security context where the cost of a classifier performance regression following a data distribution shift is high.

SVM is the appropriate alternative when the feature set is large and carefully curated and when training time is a secondary concern. Its 95.1% accuracy is respectable, and its margin-based formulation provides strong generalisation guarantees on high-dimensional data [9]. The main cost is the need for hyperparameter tuning, which adds engineering overhead to the deployment pipeline. For teams with the expertise to tune kernel parameters, SVM can approach RF performance while offering a simpler model structure.

8.2 Resource Context: Constrained Deployments

DT retains a clear niche in any context where classification decisions must be explainable to non-technical stakeholders, whether for compliance reporting, regulatory audit, or user-facing justification of why a URL was flagged. Its 92.5% accuracy is 4.7 points below RF, but no other algorithm in this comparison produces a decision path that a human can trace and understand without statistical knowledge [6]. In highly regulated industries such as banking and healthcare, where automated security decisions may be subject to audit, the DT's full interpretability is a non-negotiable operational requirement that outweighs the accuracy gap.

LR belongs at the resource-constrained end of the deployment spectrum. On browser extensions, mobile security clients, or IoT-based network monitoring where inference must complete in single-digit milliseconds on limited hardware, LR's negligible inference cost and small memory footprint are genuine practical advantages that partially offset its lower accuracy ceiling [10]. Table IV provides a consolidated selection guide mapping deployment contexts to the recommended algorithm.

8.3 Transparency and Accountability

Decision accountability still matters even when a phishing detector is not legally classified as a high-risk AI system. The EU AI Act applies its high-risk rules to defined products and use cases; it does not automatically place every cybersecurity classifier in that category. Independently of legal classification, security teams benefit from retaining model versions, feature values, thresholds, and human override records. A Decision Tree offers the clearest case-level path. Random Forest decisions can be examined with permutation importance or SHAP, while SVM and Logistic Regression require more technical interpretation. The practical point is therefore traceability and contestability, not a blanket regulatory claim.

TABLE VI Algorithm Selection Guide by Deployment Context

Algorithm	Accuracy	Speed	Strength	Best Use
DT	High	Fast	Interpretable	Audit/compliance
RF	Very High	Moderate	Robust & accurate	Production systems
SVM	High	Moderate	High-dim. data	Rich feature sets
LR	Moderate	Very Fast	Low resource use	Edge/mobile/IoT

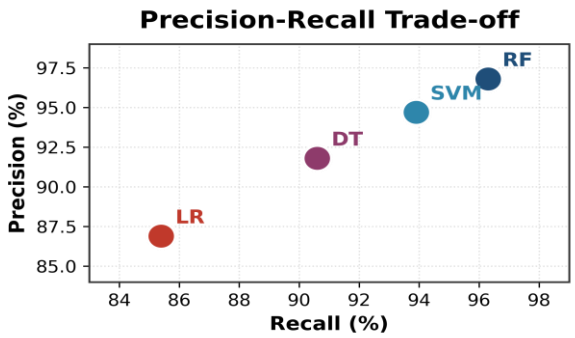

Figure. 5. Precision-Recall scatter: each algorithm plotted in two-dimensional metric space

The precision-recall scatter in Fig. 5 visualises the relationship between the two most operationally significant metrics for each classifier. RF occupies the top-right position, nearest the ideal point of (100, 100). LR is isolated in the bottom-left, confirming its separation from the other three classifiers in both dimensions. The clustering of DT and SVM in the mid-range confirms that they occupy a similar performance tier, with SVM leading on both dimensions.

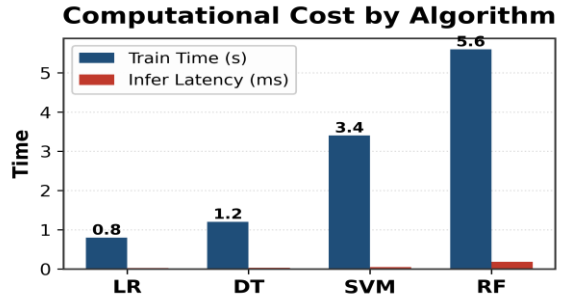

Figure. 6. Training time (seconds) and inference latency (ms) by algorithm

Fig. 6 adds the computational cost dimension to the comparison. LR trains in under one second and produces near-zero inference latency, making it far more practical than RF in latency-sensitive or energy-constrained environments. RF's training time of approximately 5.6 seconds on the benchmark datasets and 0.18ms inference latency are well within the tolerance of most server-side deployments but would be prohibitive on low-power embedded platforms. DT and SVM occupy intermediate positions, with SVM requiring more training time due to its quadratic programming solve but producing faster inference than RF.

8.4 Threats to Validity

Several threats to the validity of the conclusions in this paper should be noted explicitly. First, internal validity: because the performance figures are aggregated from multiple independent studies rather than a single controlled experiment, differences in dataset sampling, class balance handling, and hyperparameter tuning across those studies introduce variability that is impossible to fully control. The use of the same benchmark datasets (UCI and Kaggle) across the reviewed studies partially mitigates this, but differences in preprocessing and validation strategy remain.

Second, construct validity: the four performance metrics reported here (accuracy, precision, recall, F1) are standard and widely accepted, but they do not capture every operationally relevant dimension. Metrics such as area under the ROC curve (AUC-ROC), calibration quality of probability estimates, and performance under class imbalance at different severity levels are also important and would provide a richer picture of each algorithm's properties. Future comparative studies should adopt a wider metric portfolio.

Third, external validity: the performance values reported here reflect the specific URL-level feature sets used in the referenced studies. Systems that incorporate page-content features, visual similarity signals, or hosting-infrastructure graph features may produce substantially different relative rankings. The conclusions about algorithm suitability are therefore conditional on the URL-feature-based pipeline described in Section IV, and should be re-evaluated if the feature space is substantially expanded or changed.

9. CHALLENGES AND FUTURE WORK

Despite strong benchmark performance documented across the reviewed literature, several open challenges limit the direct transferability of these results to robust real-world deployment. Table VII summarises the principal challenges and research directions discussed in the reviewed literature.

TABLE VII Open Challenges in ML-Based Phishing Detection

Challenge	Description	Mitigation Direction
Concept Drift	Phishing URL patterns evolve; models degrade without retraining	Sliding-window incremental retraining pipelines
Adversarial URLs	Attackers craft URLs to defeat known classifiers while deceiving users	Adversarial training and ensemble diversity
Data Imbalance	Rare attack variants under-represented in training data	SMOTE oversampling; class-weighted loss
Benchmark Gap	Varied datasets and metrics prevent direct cross-study comparison	Standardised evaluation protocol adoption
Zero-day Domains	Novel patterns not seen during training cannot be detected	Anomaly detection layer in hybrid pipeline
Regulatory Compliance	High-risk AI systems require explainability and audit trails	Hybrid DT+RF pipeline with SHAP explanations

Concept drift is arguably the most critical operational challenge. The lexical and structural characteristics of phishing URLs shift as attackers adapt to deployed defences; a model trained on last year's phishing corpus may perform substantially worse on current attack campaigns [13][14]. Addressing drift requires either periodic full retraining on a sliding temporal window of labelled data, which demands a robust continuous labelling pipeline, or the adoption of online learning frameworks that update model parameters incrementally as new labelled examples stream in. Random Forest is particularly challenging to update incrementally because its ensemble structure requires retraining entire trees, making lightweight alternatives like Hoeffding Trees or streaming gradient-boosted models more appropriate for drift-aware deployments.

Adversarial robustness is a growing concern as ML-based detection systems become widely deployed. Sophisticated attackers can probe classifiers and craft URLs that satisfy the feature thresholds associated with legitimate classification while

still successfully deceiving human users, a class of attack known as adversarial examples [5]. Adversarial training, in which the model is exposed to adversarially perturbed examples during training, and ensemble diversity, which reduces the likelihood that a single adversarial perturbation fools all ensemble members, are promising mitigations. Feature randomisation, where the exact set of features extracted is varied across classification calls in an unpredictable way, has also been proposed as a defence against feature-level probing.

The benchmark inconsistency challenge is largely a community coordination problem. Because different studies use different datasets, preprocessing pipelines, train-test split strategies, and evaluation metrics, the numeric comparisons that appear in survey papers like this one inevitably conflate signal with methodological variation. The adoption of a shared evaluation protocol, similar to what the NLP community achieved with GLUE and its successors, would substantially improve the reliability of comparative claims in the phishing detection literature. The research community is encouraged to develop and adopt such a protocol for the URL classification task.

Looking forward, several research directions appear particularly promising. Deep learning architectures, especially character-level Convolutional Neural Networks and transformer-based URL encoders trained on large web-crawled corpora, have shown potential for end-to-end learning of phishing-relevant features without hand-crafted feature engineering [5]. Federated learning architectures could enable training on larger and more current corpora by aggregating model updates across browser clients or network security gateways without centralising sensitive user traffic logs. Large language model analysis of page content and HTML structure represents a further frontier; early work suggests that integrating semantic signals from page text with URL-level features can substantially raise the detection ceiling for highly convincing phishing pages that mimic legitimate URL patterns closely.

Finally, the integration of graph-based analysis of link networks and hosting infrastructure could add a structural dimension to URL-level feature analysis. Phishing campaigns often reuse hosting providers, DNS registrars, and TLS certificate authorities in patterns that are invisible at the single-URL level but detectable at the network level. Graph neural network approaches that embed both URL features and network topology signals into a unified representation could provide detection capabilities that are substantially harder for attackers to evade than single-URL classifiers.

A promising hybrid architecture for future work would combine a lightweight LR or DT model as a first-pass, low-latency filter with a heavier RF or deep learning model as a second-pass verifier for borderline cases. Such a cascade pipeline would achieve the inference speed benefits of LR or DT on the large majority of clearly legitimate or clearly phishing URLs, while applying the full discriminative power of RF only to the ambiguous minority. This design pattern, well-established in other high-throughput classification contexts, has not yet been systematically evaluated for phishing detection and represents an accessible direction for near-term experimental research.

10. CONCLUSION

This paper has presented a comprehensive comparative study of four supervised Machine Learning algorithms, Decision Tree, Random Forest, Support Vector Machine, and Logistic Regression, applied to the binary classification of phishing versus legitimate websites. Drawing upon established benchmark datasets, particularly the UCI Phishing Websites Dataset and the Kaggle Phishing Website Dataset, and synthesising evidence from multiple independent peer-reviewed studies, this review has evaluated each algorithm across accuracy, precision, recall, and F1-score, and provided a structured analysis of the seven URL-level and domain-level features that most reliably discriminate between the two classes.

Random Forest emerges as the top-performing algorithm across all metrics, achieving 97.2% accuracy and 96.3% recall through its ensemble aggregation mechanism. The stability of this result across different studies, datasets, and feature engineering choices confirms that RF's advantage is not an artifact of any particular experimental setup but a genuine property of the ensemble approach on this problem type. SVM provides a strong single-model alternative at 95.1% accuracy with strong generalisation on high-dimensional feature vectors, though at the cost of kernel tuning overhead.

Decision Tree's 92.5% accuracy is accompanied by full interpretability and a computationally trivial inference procedure, making it the appropriate choice for audit-driven or resource-constrained deployments where the ability to explain a classification decision is non-negotiable. Logistic Regression, with 87.3% accuracy but negligible training and inference cost, serves as the practical baseline for browser-extension, mobile, or embedded deployment contexts where latency budgets are tight.

The central conclusion from the comparative analysis is that no single algorithm is universally optimal across all deployment scenarios. The appropriate choice is determined by the intersection of three factors: the operational priority (maximising detection rate versus minimising false positives), the resource context (server-class hardware versus edge or mobile deployment), and the accountability requirement (black-box acceptable versus full decision transparency required). Table VI in Section VII maps these factors to specific algorithm recommendations.

The open challenges identified in Section VIII, concept drift, adversarial robustness, class imbalance, benchmark inconsistency, and zero-day domain detection, represent the primary research priorities for the field. Future work that addresses these challenges through online learning, adversarial training, federated data collection, standardised evaluation protocols, and hybrid ML-deep learning pipelines would substantially extend the reliability and generalisability of phishing detection systems in real-world deployments.

Finally, it should be noted that the cybersecurity landscape continues to evolve rapidly. Emerging attack vectors such as phishing campaigns delivered through legitimate cloud storage platforms, QR code-based phishing (quishing), and AI-generated personalised lure content represent significant future challenges for URL-feature-based detection systems. Addressing these variants will require both expansion of the feature space beyond URL-level attributes and closer integration with threat intelligence feeds and human-in-the-loop verification workflows. This comparative survey is intended to provide a useful foundation for that ongoing research.

11. ACKNOWLEDGMENTS

The authors thank the Department of Computer Applications at St. Francis College, Bengaluru for academic guidance, and acknowledge the open data providers UCI Machine Learning Repository and Kaggle for the phishing datasets used as reference benchmarks in this comparative study.

12. REFERENCES

[1] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," Expert

Systems with Applications, vol. 117, pp. 345–357, Mar. 2019.

[2] R. M. Verma and A. Das, "What's in a URL: Fast feature extraction and malicious URL detection," in Proc. 3rd ACM Int. Workshop on Security and Privacy Analytics (IWSPA), 2017, pp. 55-63.

[3] M. Mohammad, F. Thabtah, and L. McCluskey, "Phishing Websites Dataset," UCI Machine Learning Repository, 2015.[Online]. Available: archive.ics.uci.edu/dataset/327/phishing+websites

[4] T. Fawcett, "An introduction to ROC analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[5] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

[6] C. M. Bishop, Pattern Recognition and Machine Learning. New York: Springer, 2006.

[7] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed. Waltham, MA: Elsevier, 2012.

[8] S. Haykin, Neural Networks and Learning Machines, 3rd ed. Upper Saddle River, NJ: Pearson, 2009.

[9] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge: Cambridge Univ. Press, 2000.

[10] T. Mitchell, Machine Learning. New York: McGraw-Hill, 1997.

[11] A. Jain, R. Gupta, and S. Kumar, "A machine learning approach for phishing detection using feature-based URL analysis," Int. J. Computer Applications, vol. 177, no. 22, pp. 12–17, Apr. 2020.

[12] Kaggle, "Web Page Phishing Detection Dataset," 2023. [Online].Available: www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset

[13] Anti-Phishing Working Group, "Phishing Activity Trends Report, Q3 2023," APWG, Tech. Rep., 2023. [Online]. Available: apwg.org

[14] A. Aleroud and L. Zhou, "Phishing environments, techniques, and countermeasures: A survey," Computers & Security, vol. 68, pp. 160–196, Jul. 2017.

[15] S. Marchal, K. Saari, N. Singh, and N. Asokan, "Know your phish: Novel techniques for detecting phishing sites and their targets," in Proc. IEEE 36th Int. Conf. Distributed Computing Systems (ICDCS), 2016, pp. 323–333.

[16] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001.