Machine Learning Project Report

Phishing Website Detection Using Classical and Ensemble Models

Group Members :
Paul ARTISI
Killian BEGU
Aboubakar BAOUCHI

1 Business Scope

Phishing is one of the most widespread and dangerous threats in modern cybersecurity. Attackers create fraudulent websites that imitate legitimate platforms in order to steal sensitive information such as login credentials, banking details, or personal data. As the volume and sophistication of phishing attacks continue to increase, traditional detection methods based on static rules have become insufficient.

This is where artificial intelligence becomes essential. Unlike manual approaches or signature-based systems, a machine learning model can automatically learn complex patterns from large amounts of phishing data. It is able to identify malicious behaviors and structural anomalies even when an attack does not match any known rule.

The benefits of integrating AI into phishing detection are significant:

- **Real-time detection:** a machine learning classifier can analyze a website and predict whether it is fraudulent almost instantly, which is essential to protect users before they fall victim to an attack.
- **Adaptation to new threats:** cybercriminals continually change their techniques. An AI model trained on thousands of examples can generalize better and detect new phishing strategies that do not yet appear in traditional signature databases.
- **Scalability:** AI systems can automatically analyze millions of URLs or websites without human intervention, which is critical for large companies, web browsers, and digital service providers.
- **Reduced false positives:** by learning from diverse features such as URL structure,, behavioral indicators, domain characteristics, machine learning improves the precision of detection compared to simple rule-based filters.
- **Continuous improvement:** the model can be retrained regularly with new data, allowing it to evolve alongside emerging threats.

This project is directly aligned with a real cybersecurity need: developing an intelligent system capable of automatically and accurately distinguishing legitimate websites from phishing websites. By learning from real-world web data, our solution aims to provide a robust and automated approach to strengthen user and organizational protection against phishing attacks.

2 Problem Formalisation and Methods

The objective of this project is to detect phishing websites using machine learning techniques. The task is formulated as a **binary supervised classification problem** in which each sample corresponds to a website described by 30 handcrafted features derived from URL structure, domain metadata, and webpage behavior indicators. Each sample is assigned one of two labels:

- **0**: legitimate website
- **1**: phishing website

Given a feature vector $x \in \mathbb{R}^{30}$, the goal is to learn a decision function

$$f(x) \to \{0, 1\}$$

that predicts whether the website is fraudulent or legitimate.

2.1 Supervised Learning Framework

Since ground-truth labels are provided, the project follows a supervised learning approach. The dataset is split into training and testing subsets using a stratified split to preserve the class proportions. The model is trained on the training data and evaluated on unseen test data to measure generalization performance.

2.2 Data Preparation and Preprocessing

To ensure the dataset is suitable for model training, several preprocessing steps were applied:

- **Missing value verification:** no missing values were detected.
- **Duplicate and conflict analysis:** more than 70% of samples were duplicates, which is expected due to the discrete nature of the features. Conflicting duplicate groups were identified and quantified.
- **Outlier detection:** Isolation Forest was used to detect unusual feature combinations, revealing that phishing samples contain more outliers than legitimate samples.
- **Target encoding:** labels originally encoded as {-1, 1} were mapped to {0, 1}.

- **Feature scaling:** StandardScaler was applied for algorithms requiring normalized inputs.

These steps ensure that the dataset is consistent, well-understood, and ready for machine learning.

2.3 Methods and Models

The first model implemented is a **Random Forest classifier**, used as a baseline due to its robustness, ability to handle discrete features, and strong performance on tabular datasets.

In this project, several additional models are implemented to compare different learning strategies. These include:

- Logistic Regression
- Support Vector Machine (SVM)
- K-Nearest Neighbors (KNN)
- Decision Tree
- Gradient Boosting

Each classifier is trained using the same stratified train–validation–test split, evaluated with standard metrics (accuracy, recall, F1-score), and compared against the tree-based baselines and the final ensemble model.

2.4 Rationale for the Approach

The chosen methodology ensures a comprehensive and rigorous treatment of the problem. By conducting an extensive exploratory analysis, we obtain a deep understanding of the dataset and its underlying characteristics. The preprocessing steps, including target encoding, scaling, and verification of class balance, allow the models to be trained under reliable and consistent conditions. Furthermore, using the same data splits for all classifiers guarantees that performance comparisons are fair and meaningful. This structured approach provides a solid foundation for the subsequent stages of the project, such as hyperparameter tuning, dimensionality reduction (PCA), and ensemble learning, which aim to further improve the overall predictive performance of the system.

3 Data Description and Exploration

The dataset used in this project contains 11,055 websites, each represented by 30 discrete features taking values in $\{-1, 0, 1\}$. Each feature is derived from a hand-crafted rule that maps a measurable property of the website to three categories: legitimate, suspicious, or phishing. For example, one feature is based on the length of the URL. Very long URLs are often used to hide the suspicious part of the address bar within a long string of characters.

By analysing the distribution of URL lengths in the dataset, a frequency-based rule was defined: URLs shorter than 54 characters are considered legitimate for this feature, URLs with a length between 54 and 75 characters are marked as suspicious, and URLs longer than 75 characters are classified as phishing. The corresponding feature then takes the value -1 for legitimate, 0 for suspicious, and 1 for phishing. Similar heuristic rules are applied to other structural and behavioural properties of the websites, such as the presence of an IP address in the URL, SSL certificate characteristics, redirection behaviour, the number of subdomains, domain age, web traffic, and several HTML or JavaScript interaction indicators.

3.1 Dataset Size and Structure

The feature matrix consists of 11,055 rows and 30 columns. All variables are numerical and discrete. The target variable is also discrete and originally encoded as $\{-1, 1\}$ before being mapped to $\{0, 1\}$ during preprocessing.

The class distribution is relatively balanced, with approximately 55.7% legitimate websites and 44.3% phishing websites. This near-balanced ratio ensures that no resampling or class-weight correction is required at this stage.

3.2 Missing Values

No missing values were found in the dataset, as all 30 features are fully populated for every entry:

```
Missing Values:  0
```

This confirms that the dataset is complete and ready for modelling without the need for imputation.

3.3 Duplicate Analysis

A detailed duplicate analysis was performed due to the discrete nature of the features. Because each feature takes only two or three possible values, many websites share identical feature patterns. The analysis revealed:

- 70.9% complete duplicate rows (same features and same label),
- 71.3% feature duplicates (same features, regardless of label),
- 5,785 unique feature combinations,
- 64 conflicting duplicate groups representing 3.2% of the samples, where the same feature pattern appears with different labels.

The high duplicate rate is expected, as phishing websites often mimic the same structural indicators, leading to repeated combinations. The presence of conflicting duplicate groups, however, mainly reflects a limitation of the feature representation rather than a temporal effect. In these cases, two websites share exactly the same engineered feature vector but

were assigned different labels, which suggests that additional information not captured in our dataset (such as page content, visual appearance, brand context, or external reputation signals) was used to decide whether a website is phishing or legitimate. As a consequence, part of the label uncertainty is irreducible for our models, since some aspects that drove the original labelling process are simply not observable through the available features.

3.4 Feature Cardinality

An analysis of the cardinality of each feature showed that:

- 22 features are binary (2 possible values),
- 8 features are ternary (3 possible values).

This explains the large number of duplicate combinations and the limited variability within the dataset.

3.5 Imbalanced Data

The dataset does not suffer from significant class imbalance:

```
Legitimate:  55.7%,  Phishing:  44.3%
```

The imbalance ratio of approximately 1.26:1 is mild and does not require corrective techniques such as oversampling or class weighting.

3.6 Outlier Analysis

Outliers were detected using the Isolation Forest algorithm with a contamination rate of 5%. The model identified 553 outliers, representing unusual or rare combinations of feature values.

Interestingly, phishing websites contained more outliers (7.2%) than legitimate websites (3.2%). This observation aligns with the intuition that phishing pages employ a wider variety of techniques and patterns to evade detection, while legitimate websites tend to follow more consistent structural norms.

Overall, the dataset is clean, well-balanced, and sufficiently rich for model training, with a detailed understanding of its structure provided by the exploratory analysis.

4 Methodology

4.1 Preprocessing

The preprocessing stage ensures that the dataset is properly structured for machine learning. The target variable, originally encoded using the values {-1, 1}, was converted to the binary format {0, 1} to improve consistency with standard classification algorithms and evaluation metrics. The dataset was then divided into a training set of 6,633 samples, a validation set

of 2,211 samples, and a test set of 2,211 samples using a two-step stratified split (60% / 20% / 20%), which preserves the original class distribution and prevents accidental imbalance during both training and evaluation.

This three-way split plays an important role in obtaining an objective assessment of model performance. The training set is used to fit the models and learn the parameters. The validation set is then used to select models and tune hyperparameters (for example through GridSearchCV) without ever touching the test data. This prevents information from the test set from leaking into the training process and artificially inflating performance. The test set is kept completely separate until the very end of the project and is used only once to estimate the final generalisation ability of the chosen model on truly unseen data, mimicking how the system would behave in real-world deployment.

Since the features of the dataset are discrete and share similar scales, most models are not sensitive to feature magnitude. However, algorithms such as logistic regression, support vector machines, and k-nearest neighbors require normalized inputs. For this reason, a standardization step was included in the pipeline so that scaled features are available whenever these models are introduced. Through these preprocessing steps, the dataset becomes clean, consistent, and ready for model training in a reproducible way.

4.2 Models Implemented

The final notebook implements six classical supervised models: Logistic Regression, a Support Vector Machine with an RBF kernel, k-Nearest Neighbors, a Decision Tree, a Random Forest, and a Gradient Boosting classifier. The initial baseline is the simplest linear model (Logistic Regression), which provides a reference level of performance. Tree-based methods, and in particular Random Forests and Gradient Boosting, are then used to capture non-linear interactions between the engineered features. Distance-based (KNN) and kernel-based (SVM) models complement this view by relying respectively on local neighborhoods in feature space and on implicit high-dimensional feature mappings. All models are trained on the same preprocessed training data and evaluated with identical metrics, which guarantees a rigorous and fair comparison of their behaviour on the validation and test sets.

4.3 Hyperparameters

To obtain a fair comparison between models and avoid overfitting, hyperparameters were either set using prior knowledge or tuned systematically. For the three best-performing tree-based models, Decision Tree, Random Forest, and Gradient Boosting, a GridSearchCV with 5-fold cross-validation was run on the combined training and validation set. The search explored different tree depths, numbers of estimators, and minimum numbers of samples required to split a node or form a leaf.

For the Random Forest, the best configuration selected by GridSearch uses 200 trees with a maximum depth of 20, a minimum of 2 samples to split a node, a minimum of 1 sample per leaf, and the square-root rule for feature subsampling. The tuned Decision Tree and Gradient Boosting models also rely on relatively shallow trees, which improves generalisation while keeping enough flexibility to fit the phishing detection task.

For the remaining models, simpler hyperparameter choices are adopted. Logistic Regression uses an increased maximum number of iterations to ensure convergence; the Support Vector Machine relies on an RBF kernel with a regularisation parameter set to $C = 1.0$; and the k-Nearest Neighbors classifier starts with $k = 5$ neighbours. These configurations provide strong baselines before any further hyperparameter refinement.

5 Results

5.1 Evaluation Metrics

To evaluate model performance, several standard metrics were computed, including accuracy, precision, recall, F1-score, and the confusion matrix. These metrics provide complementary perspectives on the classifier's ability to distinguish between legitimate and phishing websites, particularly in cases where the classes may exhibit subtle behavioural differences. Among the individual models, tuned tree-based ensembles such as Random Forest and Gradient Boosting achieve strong performance on the test set, with accuracies around **97.5%**. The best overall results are obtained with a Soft Voting ensemble that combines several tuned classifiers: it reaches a test accuracy of **97.69%** and an F1-score of **97.94%**. This confirms the suitability of ensemble methods for datasets composed of discrete, engineered features such as those used in phishing detection. All models are evaluated with the same set of metrics to ensure a consistent and fair comparison.

5.2 Overfitting / Underfitting Analysis

An examination of the baseline Random Forest classifier suggests that the model generalises well to unseen data. Its high accuracy on the test set, combined with the absence of significant performance degradation relative to the training phase, indicates that the model does not exhibit signs of overfitting despite its capacity to fit complex decision boundaries. This behaviour is expected from Random Forests, which mitigate overfitting through the aggregation of multiple independent decision trees. The ensemble structure reduces variance while preserving predictive strength, making the method particularly effective for tabular datasets with limited feature interactions. Comparing these algorithms will provide deeper insight into how different learning strategies capture the structural patterns of phishing websites.

6 Evaluation and Comparison with Baseline

All six individual classifiers and the ensemble models are evaluated on the held-out test set using the same metrics. Logistic Regression and the SVM provide strong linear and kernel baselines, but they are slightly outperformed by non-linear tree-based methods. Tuned Random Forest and Gradient Boosting both reach test accuracies around **97.5%** with F1-scores close to 0.978, confirming that ensembles of shallow trees are very well suited to this discrete tabular dataset.

Beyond individual models, higher-level ensemble combinations provide a small additional gain. The best overall performance is obtained with a Soft Voting classifier that aggregates

several tuned models. This ensemble achieves a test accuracy of **97.69%** and an F1-score of **97.94%**, with very high recall for phishing websites. Hard Voting and Stacking ensembles also reach very similar scores, showing that several model combinations are viable options.

Overall, the comparison confirms that the best trade-off between accuracy, robustness, and model complexity is provided by tree-based ensemble methods. In practice, we retain the Soft Voting ensemble as the final model for phishing website detection, while the tuned Random Forest remains a strong and slightly simpler baseline.

7 Limitations

Although the dataset and methodology provide a solid foundation for the detection of phishing websites, several limitations must be acknowledged. The features available in the dataset are entirely hand-engineered and discrete, taking values in {-1, 0, 1}. While this representation simplifies the modelling process, it does not capture the full semantic richness of URLs, domain information, or webpage behaviour. As a result, the models can only exploit the predefined patterns encoded in these features and may overlook more subtle or modern phishing strategies.

Another limitation arises from the presence of duplicate and conflicting samples. The dataset contains a very high proportion of duplicate rows, which is expected due to the limited cardinality of the features; however, approximately 3.2% of the samples share identical feature vectors but carry different labels. These inconsistencies introduce noise that may negatively affect model training, especially for classifiers sensitive to label uncertainty.

Moreover, the project does not incorporate deep learning techniques or NLP-based URL embeddings, primarily due to time constraints and the structure of the dataset. Such approaches could capture sequential patterns, character-level anomalies, or stylistic markers that engineered features fail to represent. Finally, the dataset does not include contemporary obfuscation techniques used in modern phishing attacks, such as Unicode homograph manipulation, punycode-based domains, or adversarial URL constructions. This restricts the generalisability of the models to real-world scenarios where attackers frequently adapt their strategies.

Overall, while the models perform well within the constraints of the dataset, these limitations highlight avenues for future improvement and more robust phishing detection systems.

8 Discussion and Conclusion

This project demonstrates that classical machine learning techniques can be highly effective for detecting phishing websites when provided with a structured set of engineered features. The Random Forest classifier, used as the baseline model, achieved an accuracy of 97.5% on the test set, confirming that ensemble methods perform particularly well on discrete, tabular datasets of this nature. Its strong generalisation ability, combined with minimal signs of overfitting, highlights the suitability of tree-based models for cybersecurity tasks involving behavioural and structural indicators.

The exploratory analysis played an essential role in guiding the modelling process. The

dataset was found to be clean, complete, and relatively well balanced, requiring no imputation or resampling. However, the high proportion of duplicate rows and the presence of conflicting labels also revealed some inherent limitations, suggesting that the dataset may not fully capture the diversity and ambiguity of real-world phishing attempts. Despite this, the Random Forest model proved robust to these imperfections and established a reliable baseline for subsequent experimentation.

While the current results are promising, they also leave room for future improvements. Extending the work to additional models, such as logistic regression, support vector machines, k-nearest neighbors, decision trees, and boosting methods, would provide a broader comparison and a deeper understanding of how different learning paradigms capture phishing patterns. More advanced approaches could also be explored, including deep learning architectures capable of analysing raw URLs or embeddings that encode semantic and sequential aspects of domain names. Incorporating features extracted from real-time webpage content or integrating more recent phishing strategies, such as obfuscated Unicode domains or adversarial URL manipulation, would further enhance the robustness and applicability of the model.

In conclusion, the project validates the effectiveness of traditional machine learning for phishing detection while highlighting both the strengths and limitations of the dataset. It establishes a solid methodological foundation upon which more sophisticated models and richer feature representations can be developed.