Phishing Website Detection

1. Introduction

In recent years, phishing is one of the serious and widespread forms of cybercrime. Phishing websites are malicious web pages that mimic legitimate sites to trick users to enter their sensitive information including passwords, personal details and financial accounts. This problem is particularly interesting because cybersecurity is now a critical concern globally and there is an increasing number of threats targeting individuals and organisations. Phishing detection systems can play a vital role in the early cyber threats. The goal of this project is to develop machine learning models to identify the phishing web based on features like URL, Title and length of HTTPs etc. This dataset provides a rich bias for training classification models. Classifiers including Naive Bayes, Decision Tree and Support Vector Machine will be trained and evaluated, this report will discuss the methodology to train the model. The report aims to compare and evaluate the performance of different models for distinguishing between legitimate and phishing websites.

2. Literature Review

Phishing detection remains active in cybersecurity research because of the rising prevalence of online scams. The number of unique website phishing has increased significantly in recent years, it skyrocketed after 2020 and the website phishing attacks are higher than other phishing attacks like email phishings (Prasad & Chandra, 2023). However, the traditional blacklist detection systems often fail to keep up, which means attackers can easily change the top domain of the website to avoid it (Prakash et al., 2010). Thus, this increased adoption of machine learning (ML) approaches for accurate phishing detection.

This project used the dataset called “PhiUSIIL Phishing URL (Website) ” from the UC Irvine Machine Learning Repository (Prasad & Chandra, 2023). The dataset used for the project contains a rich set of the features(more than 50 features) including all sorts of the information for the website covering from the text-based data like titles and domain names, boolean data including “is HTTPS” and numerical data like length of the URLs and lines of the code. The data comprises 134,850 legitimate and 100,945 phishing websites, which contains more than 200,000 instances. This dataset enables training the models to generalise patterns because those features and instances are meaningful for the classification. It has been used for several research projects and useful for training supervised learning because of its diverse features and formatted label.

3. Methods

3..1 Data Acquisition and Subsampling

Dataset: 235,795 total URLs (134,850 legitimate; 100,945 phishing) from the UCI repository.
Subsampling Justification: Randomly selected 5,500 instances (3,158 legitimate; 2,342 phishing) to ensure the randomness and no bias within the computational constraints.

3.2 Data Pre-processing

Before the model training, several data processing were performed to make sure the data is clean and structured. Raw text cannot be interpreted by most ML models directly.

1. *Missing Values & Duplicates*: The dataset contained no missing values, which is confirmed by dropping duplicates and missing count.
2. *Text-Based Features Handling*: All non-numerical data is pulled out for the further processing including “FILENAME, URL, Domain, TLD, Title”.
3. *Dataset Sizing*: the database has 235,796 instances for phishing data, reducing this size by taking a subset of 5,500 instances.

3.3 Feature Construction

To enhance the classifier’s performance transferring data like raw into meaningful numerical features is necessary. Some features like changing raw texts into numbers or vectors are constructed to interpret easily by the model.

A. TLD processing

1. *TLD_Top10%*: Constructed from TLD identified 10 most frequent TLD booleans covering more than 70% of the data.
2. Enumerate the unique values for the TLD by changing raw text into numerical identifiers.

3. *TLD_target_encoded* : calculated the mean phishing probability for each TLD. This feature reflects both frequency and historical risk.

$$TLD\_target\_encoded = \frac{Number\ of\ phishing\ websites\ with\ TLD}{Total\ website\ with\ TLD}$$

B. URL processing: Counted the suspicious substring occurrences from the research by Jalil, Usman and Fong (2022).

C. Title processing:

1. Word Embedding using the Word2Vec and reduced to 3 dimensions using SVD.
2. Sentiment Analysis: titles were analysed using TextBlob to extract sentiment polarity and subjectivity scores capturing emotional cues that may hint at phishing intent.

D. Numerical Feature Transformation: Some of the numeric features showed skewed(mostly right skewed) and were transformed to normalise.

1. *Log Transformation*: URLLength, LineOfCode, LargestLineLength, NoOfImage, NoOfCSS, and NoOfObfuscatedChar. Features have a very high skewness, for example “LineOfCode” has 44 and “NoOfObfuscatedChar” has 36. Those features will affect the model performance.
2. *Z-score Standardization*: DomainLength to standardise its scale.

E. Financial Flag : A new boolean feature” FinancialFlag” was created by calculating a logical “ XOR ” operation across the “Bank”, “Pay”, “Crypto”, and “HasPasswordField” features. This combines all the presence of financial-related information in a more compact way.

$$FinancialFlag = XOR(\ Bank \mid Pay \mid Crypto \mid HasPasswordField\ )$$

3.4 Feature Selection

In this project, three filter methods of the feature selection have been used to improve model generalizability, especially reduce the noise and combat overfitting. This dataset contains 63 features after features construction, this can result in confusion with the model and overfitting. Three methods are applied in order to provide more robust feature selections. Correlation filtering method removes the redundancy of the data, Mutual Information gets the general dependence and Chi-Square ensures statistically relationships. After all feature selection, 24 features have been kept.

1. *Correlation filtering*
   High correlated features contain redundant information, this will inflate some certain patterns and reduce model interpretability. So when two features have a correlation coefficient more than 0.7, one of them will be removed. The advantages of using correlation filtering is that it is simple and fast to compute and reduces the redundancy of the data. However, it only captures the linear relationships of some data and this method does not consider the target label of the phishing websites, which might remove some relevance. The average correlation is low overall, which is 0.146. From the heatmap, there are a number of the features that show high correlation like “URLSimilarityIndex” and “ObfuscationRatio”; those features have been removed.
2. *Mutual Information(MI)*
   Mutual Information measures the dependency between features and label(target) and captures the non-linear relations that are missed out by the correlation. The key metric of this method is the MI score, the higher means stronger dependency. The strengths of MI are it works for all data types and captures linear and non-linear relationships, but it is computationally slower for large datasets and less intuitive. The highest MI score is feature “LineOfCode”, which has MI score 0.6.
3. *Chi-Square*
   The purpose of the Chi-Square is to test the independence between categorical features and target. The metrics for this method are Chi-square statistics and p-value. This method is fast and interpretable for categorical data and effective for classification tasks like the classification of the phishing website. The selection process here is taking features that have p value less than 0.05.

Table 1 : Feature Selection Process

Method	Features Removed	Features Retained	Notes
Correlation	21	42	Redundancy Removal
Mutual Information	14	28	Non-linear dependency
Chi-Square	4	24	statistical relevance

3.5 Algorithms / Training Models

This project evaluates three machine learning algorithms including Naive Bayes, Decision Tree, and Support Vector Machine. These 3 models are selected based on the complexity and assumptions, NB has the high bias, DT has lower bias than it when limited to max_depth = 3 and SVM has the lowest bias here. SVM has the highest variance, DT has the medium variance and NB has the lowest variance among these algorithms. Those 3 algorithms are chosen for the phishing dataset because they are practical for imbalance data and computationally efficient. But for the better bias-variance spectrum, models like Random Forest and XGBoost should replace DT and NB. For this project to compare these models for better efficiency.

3.6 Cross-Validation:

K-fold cross validation is implemented, the dataset is split into 5 folds, each fold serves as the test set once, the remaining 4 folds are used for training. K=5 because this number balances the efficiency and reliability. The nested CV (k=3) is also used for hyperparameter tuning. This method is reliable and mitigates overfitting, but it requires a lot of time to compute. For this approach, 5(outer) x 3(inner)x hyperparameter x 3 models, this results in more than 100 total runs for the model fit. This means this approach will not work for the extremely larger dataset with millions of samples.

3.7 Nested Cross-Validation and hyperparameter Tuning:

The outer loop assesses the generalisation error and the inner loop performs the tuning with GridSearchCV each model.

Gaussian Naive Bayes(GNB): tuned using “var_smoothing”

Decision Tree(DT): tuned using “max_depth” and “min_samples_split”

Support Vector Classifier(SVC): “kerne” = linear/rbf, with the parameter “C” tuned 0.1, 1 and 10.

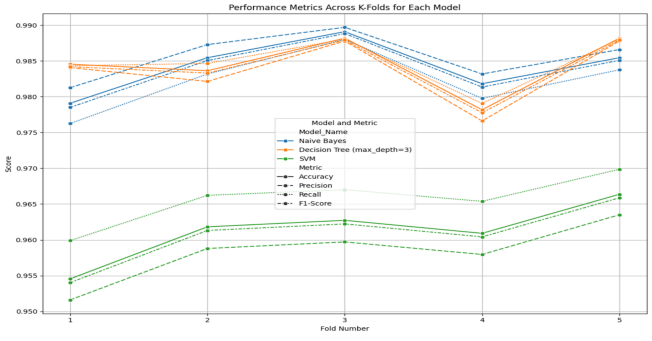
3.8 Metrics

Metrics including accuracy, precision, recall and F1 are used for the project to evaluate the performance of the model. Accuracy is the basic evaluation metric for examining the correctness of a model. Precision and Recall scores are similar as they all used the confusion matrix values. In this case, precision is the proportion of correctly predicted phishing websites out of all websites labelled as phishing, whereas, recall is the proportion of actual phishing websites that are correctly identified by models. And F1 score is the harmonic mean of precision and recall, which balanced precision and recall.

4. Results and Discussion

All three models perform very well in general, which all result in very high accuracy(>95%), the best model with the highest score is Decision Tree. This suggests tree structure captures non-linear feature interaction in the dataset, the depth constrained overfitting for the tree, but this could be due to the feature selection or construction step. And the Naive Bayes works well as DT. SVM is the lowest performance because the linear kernel can not capture the intricate.

Both DT and NB models perform nearly identically for the phishing website, with DT slightly better in Accuracy(0.9845 vs. 0.9841), Recall(0.9848 vs. 0.9821) and F1(0.9842 vs. 0.9837). NB achieves higher Precision(0.9856 vs 0.9836). This means NB minimises the false positives(FP) slightly better, it also shows it is highly cautious when predicting the website is malicious. The reason why NV works here is because the nature and simplicity generalise well in this dataset after the feature selection. The strong performance of DT and NB suggest the dataset’s feature ike (URLLength, IsHTTPS) from the feature selection process are either rule-based or approximately independent, aligning with DT and NB, respectively.

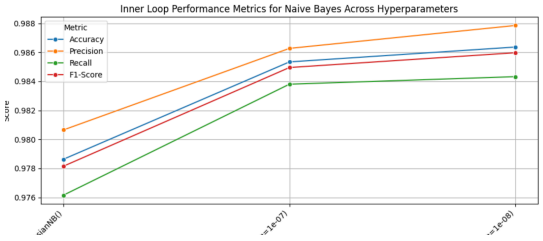


Graph 1: Performance Matrix Across K-Fold

Graph 2: Average Scores across Models

SVM lags behind in all metrics having all numbers around 0.96, this is likely because of the kerne; choice or regularisation of the hyperparameters. Those data that are non-linear could not be captured by the default configuration and SVM is more sensitive to feature scalings.

Across the folds for the cross-validation(see graph 2), it shows consistent performance for DT and NB, which results in a score from 0.97-0.99 and the third fold (fold 3) has the best results overall. This indicates robustness. However, SVM's lower and more variable scores shows that this model has higher sensitivity to training data splits. SVM requires significant tuning to match other models, but it is currently less practical in this data.



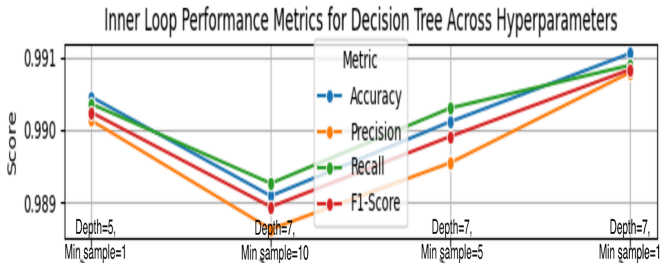
Graph 3: Naive Bayer Tuning

Table 2 : Model Comparison

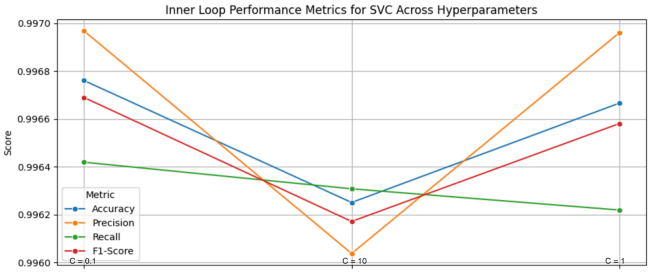
Model_Name	Accuracy	Precision	Recall	F1-Score
Decision Tree (depth=3)	0.9845	0.9836	0.9848	0.9842
Naive Bayes	0.9841	0.9856	0.9821	0.9837
SVM	0.9612	0.9583	0.9656	0.9607

The results from the nested cross validation and tuning, there is no big difference for the NB all around 0.98. But referring to the graph for NB, the var_smoothing still affects the performance of the model, hyperparameter "1e-8" has the best scores(see graph 3), default configuration is the lowest. This works well because increased var_smoothing means more regularisation, this smooths variance estimates and improves numerical stability. The hyperparameter for DT helps the performance improve by 1%(graph 4) for nearly all scores(accuracy, precision, recall and F1), the nested and tuning provides a strong performance improvement for the decision tree, and depth controls help prevent overfitting, different min_samples_split also helped stable the performance by suggesting decision boundaries.

After the cross-validate and hyperpremer tuning SVC performs the best(0.996). The main reason why SVC works well after tuning is its ability to find optimal separating hyperplanes in high-dimensional feature space; robust C values generalise well. The best score is when C=0.1(graph 5), for precision score C=0.1 and C=1 have similar results. Lower C has stricter regulation, which widens the margin. For this phishing dataset C=0.1 is ideal because it works well in balancing bias-variance trade-off.



Graph 4: Decision Tree Tuning



Graph 5: SVC Tuning

5. Conclusion

Overall, phishing detection remains a critical threat in today's digital landscape, this project has evaluated machine learning solutions by feature engineering. Through cross-validation, Decision Tree provided an excellent performance without tuning and Naive Baayes also offered pragmatic alternatives , but support vector machines delivered the highest performance after the nested cross validation and tuning. Looking forward, ensemble approaches and interpretability techniques could furthest enhance detection performance, integrating these machine learning models can detect threats earlier and help users from risk of phishing.

6. References

Abdelhamid, N., Ayesh, A., & Thabtah, F. (2014). Phishing detection based Associative Classification data mining. *Expert Systems with Applications*, *41*(13), 5948–5959.https://doi.org/10.1016/j.eswa.2014.03.019

Prakash, P., Kumar, M., Kompella, R. R., & Gupta, M. (2010). PhishNet: Predictive Blacklisting to Detect Phishing Attacks. *2010 Proceedings IEEE INFOCOM*. https://doi.org/10.1109/infcom.2010.5462216

Prasad, A., & Chandra, S. (2023). PhiUSIIL: A diverse security profile empowered phishing URL detection framework based on similarity index and incremental learning. *Computers & Security*, *136*, 103545. https://doi.org/10.1016/j.cose.2023.103545