Classification and Detection of Phishing Websites

DS 325 Final Project, Spring 2025

Ratul Pradhan

Introduction

Phishing attacks through websites, most often aimed to steal user information and sensitive data, mimic legitimate sites enticing users into believing their portals are secure. It is reported that nearly 80% of security incidents are attributed to phishing, with losses around $17,700 every minute due to these attacks.[1] Furthermore, 74% of these security breaches are attributed to human error as manual detection is heavily reliant on user judgement and personal pre-disposition, which can be slow and prone to errors.

This project seeks to analyze URL structure, page metadata and other features that may be extracted from a website and its content to flag potentially malicious sites. I hypothesize such sets of features can drive a machine learning model to effectively distinguish phishing from legitimate sites.

The dataset used for this project was retrieved from UC Irvine's Machine Learning Repository[2], under PhiUSIL Phishing URL (Website)[3], which has a supporting paper that aims to create a similar detection framework using incremental learning.[4] To ensure independent inquiry, the results and methodology of this paper were only viewed post feature engineering, model selection, and evaluation to ensure the results are unbiased by prior work.

A Random Forest Classifier model was engineered that achieved near-perfect accuracy (> 99%) with *URLSimilarityIndex, NoOfExternalRef,* and *LineOfCode* being the top three most significant features of importance, accounting for ~39% for the total impurity reduction, i.e. the model's decision-making power. The top ten features accounted for 82%.

[1] (Ucar)
[2] ("UCI Machine Learning Repository")
[3] (Prasad and Chandra)
[4] (Prasad and Chandra)

Methodology

The dataset was fetched via the *ucmilrepo*. It contains 235795 instances and 54 unique features, with a mix of real, categorial and numerical data. The features are a mix of original features (URL and metadata) and derived features extracted from the original.

Preprocessing

Data was first checked for null, duplicates, zero variance features, and class balance (~57% legitimate to 43% phishing)

The dataset has undergone significant to extract meaningful features from the original. However, for the feature 'Title', the dataset only creates 'URLTitleMatchScore' to identify the discrepancy between the URL and the webpage title.

For this reason, I engineer the following derived features:

- Extracted length, word count, average word length.
- Computed sentiment polarity/subjectivity via **TextBlob**; binned polarity into quartiles.
- Counted POS (Part-of-speech) tags (NN, JJ, VB, RB) using **spaCy**.
- Generated word-embedding averages from spaCy's en_core_web_sm.

Since the alphanumeric columns (Domain, Title, URL, TLD) were encoded, they were removed,

The dataset was then standardized and **Principle Component Analysis (PCA)** was applied to retain 95% of the variance (reducing 59 to 42 features/dimensions)

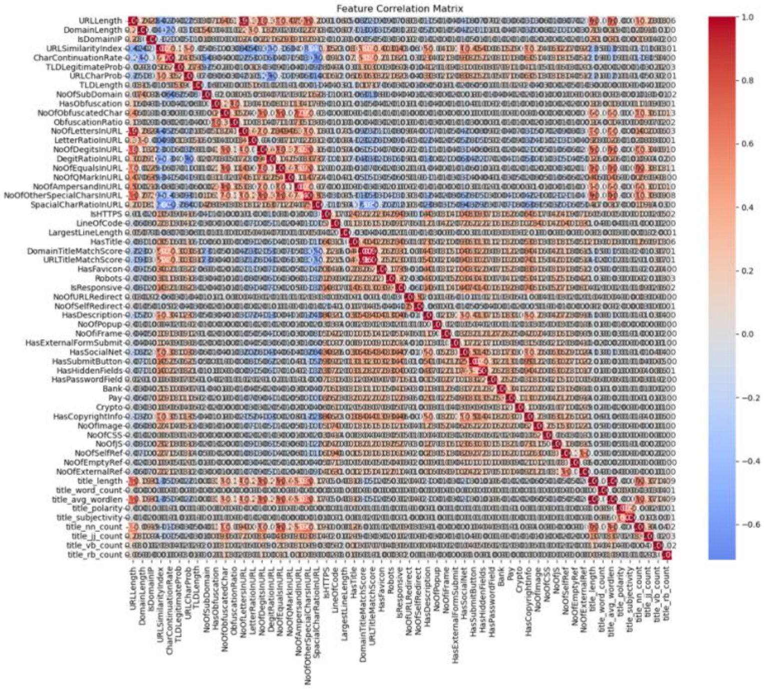
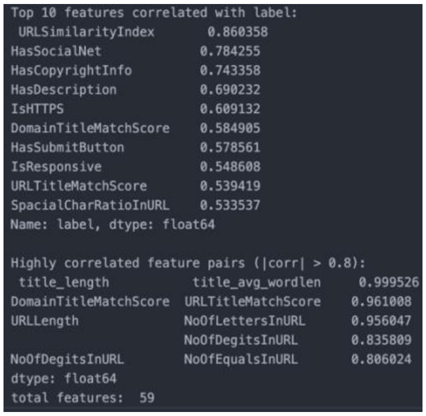


Fig 1. Correlation Matrix

Since the Correlational Matrix was completely unreadable, the top 10 significant and highly correlated pairs were printed



```
Top 10 features correlated with label:
 URLSimilarityIndex      0.860358
HasSocialNet            0.784255
HasCopyrightInfo        0.743358
HasDescription          0.690232
IsHTTPS                 0.609132
DomainTitleMatchScore   0.584905
HasSubmitButton         0.578561
IsResponsive            0.548608
URLTitleMatchScore      0.539419
SpacialCharRatioInURL   0.533537
Name: label, dtype: float64

Highly correlated feature pairs (|corr| > 0.8):
 title_length           title_avg_wordlen     0.999526
DomainTitleMatchScore  URLTitleMatchScore    0.961008
URLLength              NoOfLettersInURL      0.956047
                       NoOfDegitsInURL       0.835809
NoOfDegitsInURL        NoOfEqualsInURL       0.806024
dtype: float64
total features:  59
```

Fig 2. Simplified Correlation Matrix

Modelling and Experimentation

A Random Forests (RandomForestClasssifer) Classification model was utilized with hyperparameter tuning via a Grid Search with cross-validation (GridSearchCV 5-fold) over n_estimators, max_depth, max_features.

Random Forests are robust and interpretable ensemble learning method that can handle high-dimensional data effectively and have high interpretability via feature importance and permutation important metrics.

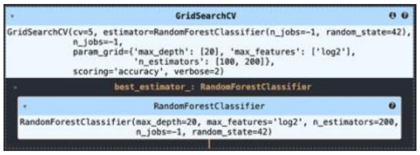
The experiments were run on a MacBook Pro, M3 Pro Chip with 18GB RAM.

4 experiments were run:

1. RandomForestClassifier on Principle Component Analysis reduced data (log2 splits, 100-200 trees) – Runtime: 3mins, 12secs
2. RandomForestClassifier on pre-processed 58 feature dataset (sqrt splits, 200-500 trees) – Runtime: 1min, 22secs
3. RandomForestClassifier on calculated top 5 features used in Exp 2. (using GridSearchCV.best_estimator_.feature_importance)
4. RandomForestClassifier on calculated top 5 features, excluding the first highest feature (URLSimilarityIndex)

Results

1. The first set of results are from running the RandomForestClassifier on Principle Component Analysis reduced data.



```
GridSearchCV
GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             n_jobs=-1,
             param_grid={'max_depth': [20], 'max_features': ['log2'],
                         'n_estimators': [100, 200]},
             scoring='accuracy', verbose=2)
best_estimator_: RandomForestClassifier
RandomForestClassifier
RandomForestClassifier(max_depth=20, max_features='log2', n_estimators=200,
                       n_jobs=-1, random_state=42)
```

Fig 3. Grid Search - RandomForestClassifier on Exp 1.

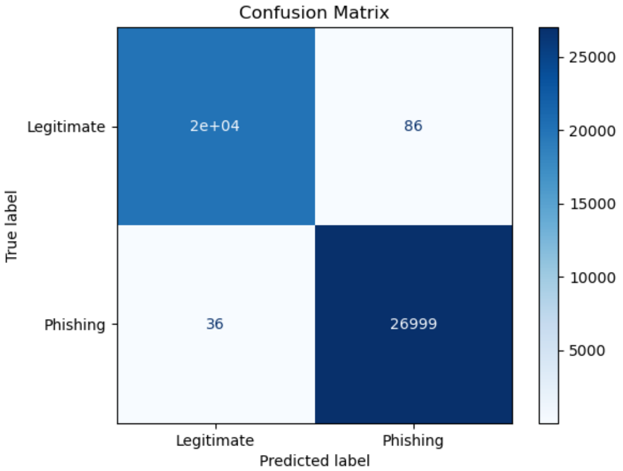
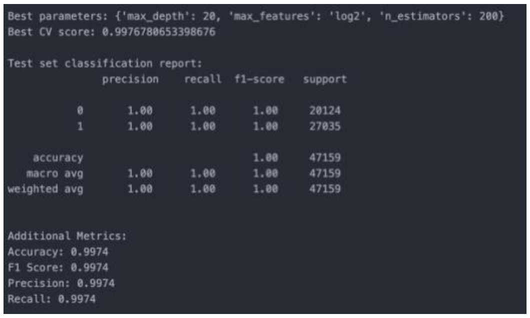


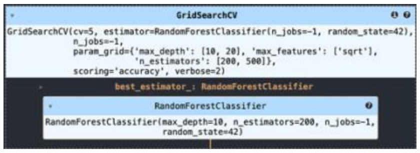
Fig 4. Confusion Matrix for Exp 1.



```
Best parameters: {'max_depth': 20, 'max_features': 'log2', 'n_estimators': 200}
Best CV score: 0.9976780653398676

Test set classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     20124
           1       1.00      1.00      1.00     27035

    accuracy                           1.00     47159
   macro avg       1.00      1.00      1.00     47159
weighted avg       1.00      1.00      1.00     47159


Additional Metrics:
Accuracy: 0.9974
F1 Score: 0.9974
Precision: 0.9974
Recall: 0.9974
```

Fig 5. Additional Metrics for Exp 1.

2. The second set of results are from running the RandomForestClassifier on pre-processed 58 feature dataset



```
GridSearchCV
GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             n_jobs=-1,
             param_grid={'max_depth': [10, 20], 'max_features': ['sqrt'],
                         'n_estimators': [200, 500]},
             scoring='accuracy', verbose=2)
best_estimator_: RandomForestClassifier
RandomForestClassifier
RandomForestClassifier(max_depth=10, n_estimators=200, n_jobs=-1,
                       random_state=42)
```

Fig 6. Grid Search - RandomForestClassifier on Exp 2.

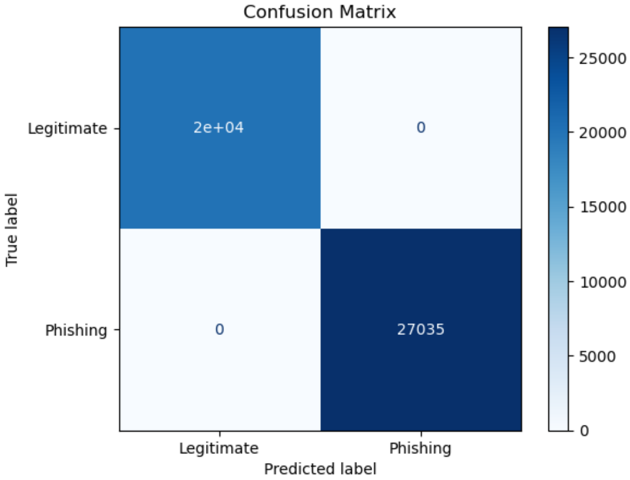
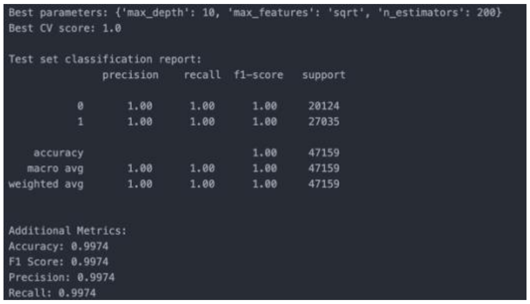


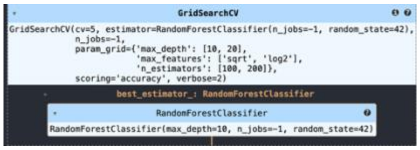
Fig 7. Confusion Matrix for Exp 2.



```
Best parameters: {'max_depth': 10, 'max_features': 'sqrt', 'n_estimators': 200}
Best CV score: 1.0

Test set classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     20124
           1       1.00      1.00      1.00     27035

    accuracy                           1.00     47159
   macro avg       1.00      1.00      1.00     47159
weighted avg       1.00      1.00      1.00     47159


Additional Metrics:
Accuracy: 0.9974
F1 Score: 0.9974
Precision: 0.9974
Recall: 0.9974
```

Fig 8. Additional Metrics for Exp 2.

3. The third set of results are from running the RandomForestClassifier on the top 5 features of important in Exp 2 (using GridSearchCV.best_estimator_.feature_importance)



```
GridSearchCV
GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             n_jobs=-1,
             param_grid={'max_depth': [10, 20],
                         'max_features': ['sqrt', 'log2'],
                         'n_estimators': [100, 200]},
             scoring='accuracy', verbose=2)
best_estimator_: RandomForestClassifier
RandomForestClassifier
RandomForestClassifier(max_depth=10, n_jobs=-1, random_state=42)
```

Fig 9. Grid Search - RandomForestClassifier on Exp 3.

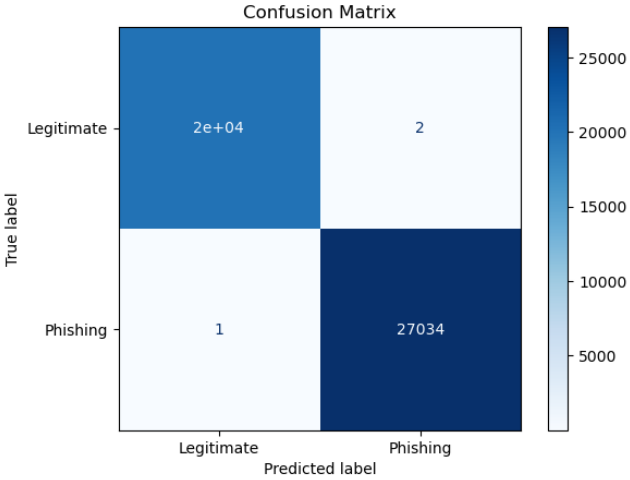
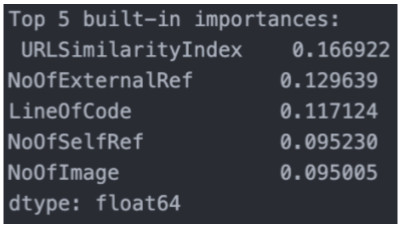
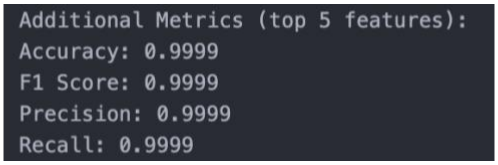


Fig 10. Confusion Matrix for Exp 3.



```
Top 5 built-in importances:
 URLSimilarityIndex    0.166922
NoOfExternalRef       0.129639
LineOfCode            0.117124
NoOfSelfRef           0.095230
NoOfImage             0.095005
dtype: float64
```

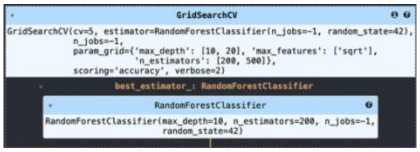
Fig 11. Top 5 features of importance found in Exp 2, used for Exp 3.



```
Additional Metrics (top 5 features):
Accuracy: 0.9999
F1 Score: 0.9999
Precision: 0.9999
Recall: 0.9999
```

Fig 12. Additional Metrics for Exp 3.

4. The third set of results are from running the RandomForestClassifier on the top 5 features of important in Exp 2, excluding the first highest feature (URLSimilarityIndex)



```
GridSearchCV
GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             n_jobs=-1,
             param_grid={'max_depth': [10, 20], 'max_features': ['sqrt'],
                         'n_estimators': [200, 500]},
             scoring='accuracy', verbose=2)
best_estimator_: RandomForestClassifier
RandomForestClassifier
RandomForestClassifier(max_depth=10, n_estimators=200, n_jobs=-1,
                       random_state=42)
```

Fig 13. Grid Search - RandomForestClassifier on Exp 4

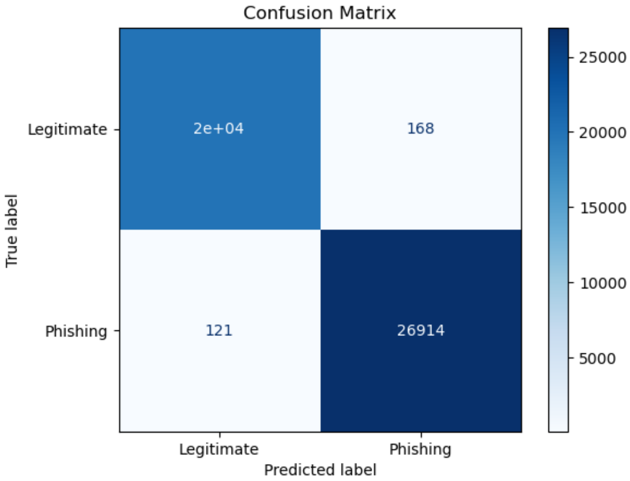
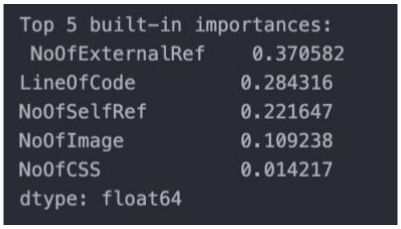
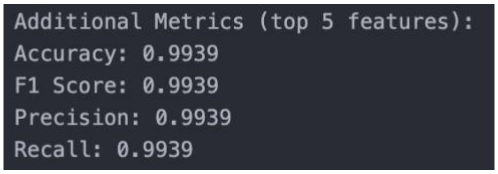


Fig 14. Confusion Matrix for Exp 4.



```
Top 5 built-in importances:
 NoOfExternalRef    0.370582
LineOfCode         0.284316
NoOfSelfRef        0.221647
NoOfImage          0.109238
NoOfCSS            0.014217
dtype: float64
```

Fig 15. Top 5 features of importance excluding URLSimilarityIndex.



```
Additional Metrics (top 5 features):
Accuracy: 0.9939
F1 Score: 0.9939
Precision: 0.9939
Recall: 0.9939
```

Fig 16. Additional Metrics for Exp 4.

Discussion

Experiment 1, with the PCA reduced features, when run on Random Forests achieved almost perfect predictions (greater than 99% accuracy, precision, recall and F1 score), where fewer than 1% of the phishing sites were misclassified (86 False Negatives, 36 False Positives, out of 47,159 websites) (Figure 7). This result suggested extremely high model effectiveness.

With the success of this model, experiment 2 focused on using the original pre-processed dataset without reduction, which resulted in perfect predictions, 47,159 websites being (20,124 legitimate, 27,035 phishing) (Figure 10).

It should also be noted that Exp 1 ran on 'log2' with 10 fits in GridSearchCV and Exp 2 on 'sqrt', with 20 fits. For this reason, I assumed Exp 1 would take significantly shorter model computation times but the contrary occurred, where Exp 1 took ~3mins and Exp 2 took ~1min 10secs.

I believe this is occurring for the following reasons:

1. PCA features are dense, double-precision floats. Hence, every threshold comparison is more expensive and doesn't exploit faster integer operations.
2. The original engineered features carry strong, interpretable signals (counts, flags, ratios) while PCA components are linear mixtures that may not easily split, requiring more work per node.

Using the feature importance function from GridSearchCV allowed inspection of each individual feature's fraction of the total impurity reduction across all trees that's attributable to that feature. (Figure 11).

URLSimilarityIndex of 0.1669 meant that this feature accounted for about 16.7 % of the model's total 'split-gain' when building all the trees, signaling it as the most powerful predictor.

Figure 17 shows the model's 'phishing' prediction changes too 'legitimate' when URLSimilarityIndex hits ~0.5.

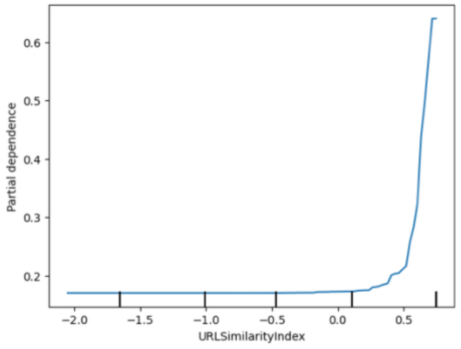


Figure 17 Partial Dependence of URLSimilarity Index

According to the attached paper[5] URLSimilarityIndex scores how closely a URL matches any of the top 10 million legitimate websites, where a perfect score matches a legitimate URL.

Using this information Exp 3 reran the classifier using the top 5 most important features to recreate the tree, which resulted in 2 false negatives and 1 false positive, from a test data set of

[5] (Prasad and Chandra)

47,159 sites, with every other website accurately predicted, highlighting that the top 5 features heavily influence the classifier.

Exp 4 removed URLSimilarityIndex and ran the classifier on the next top 5 features (Figure 15), which resulted in the worst model yet, albeit still very effective (Figure 14). This model has 168 false negatives and 121 false positives.

These results support the findings reported in the paper attached to the dataset, where they achieved 99.24% accuracy when experimented with a fully incremental training approach and 99.79% when experimented with a pre-training approach.

Some limitations of this project's approach are the following:

- The model is pre-trained, and dataset is static, with complete reliance on pre-computed URL and title features. Raw HTML, JavaScript behavior, SSL certificate age, for example were not incorporated.
- The UCI labels assume ground truth. However, some "legitimate" sites may host phishing kits or may been compromised after data collection.
- Random Forests may overestimate features with large variance.

Some future work for this project could investigate:

1. Creating an incremental approach as seen in the attached paper to update the model with new data.
2. Incorporate dynamic features such as SSL certificate age, DNS changes, HTML/CSS structure metrics and so on.

To conclude, this project, from feature engineering and model creation, showcases a framework which effectively helps detect phishing websites with high accuracy, interpretability, and efficiency.

Citations

Prasad, Arvind, and Shalini Chandra. "PhiUSIIL Phishing URL (Website)." *UCI Machine Learning Repository*, donated 3 Mar. 2024, https://archive.ics.uci.edu/dataset/967/phiusiil+phishing+url+dataset. Accessed 2 May 2025.

Prasad, Arvind, and Shalini Chandra. "PhiUSIIL: A Diverse Security Profile Empowered Phishing URL Detection Framework Based on Similarity Index and Incremental Learning." *Computers & Security*, vol. 136, Elsevier BV, 1 Jan. 2024, p. 103545, https://doi.org/10.1016/j.cose.2023.103545. Accessed 2 May 2025.

University of California, Irvine. *UCI Machine Learning Repository*. n.d. https://archive.ics.uci.edu/. Accessed 2 May 2025.

Ucar, Ozan. "Unveiling the Key Insights from the 2023 Data Breach Investigations Report (DBIR)." *Keepnet Labs*, 25 Jan. 2024, https://keepnetlabs.com/blog/unveiling-the-key-insights-from-the-2023-data-breach-investigations-report. Accessed 2 May 2025.

Note about AI Usage

Microsoft's Co-Pilot and OpenAI's ChatGpt were used in two specific ways in this project.

The first function was to provide inline comments and documentation, to ensure consistent readability in the Jupyter Notebook Pradhan_Phishing_Code.ipynb

The second function was to utilize these tools in an exploratory method experiment with function-chaining patterns and parameter configurations, heavily under Co-Pilot's @*explain* function.

All AI generated suggestions were reviewed, edited and corrected by the author.