Phishing URL Detection

Abhirup Das (2022019)
Krishna Shukla (2022254)

Himanshu Singh (2022217)
Shaurya Parashar (2022475)

1. Introduction

The internet has become an integral part of daily life, but it has also become a hub for cybercrime. Phishing attacks, in particular, have witnessed a significant surge in recent years. In 2022, phishing attacks increased by 47.2%, and in 2023, they rose by another 40%, costing consumers an estimated $8.8 billion. This shows the urgent need for effective methods to detect and prevent phishing attacks. Our GitHub page - Code

2. Problem Statement and Dataset Overview

Our work aims to develop a classification algorithm capable of accurately distinguishing phishing URLs from legitimate ones. By analyzing URL features and identifying patterns, we aim to establish a robust first line of defense against malicious web URLs.

The dataset (**source:** [1]) used comprises 235,795 instances, categorized into phishing (100,945) and legitimate (134,850) URLs.

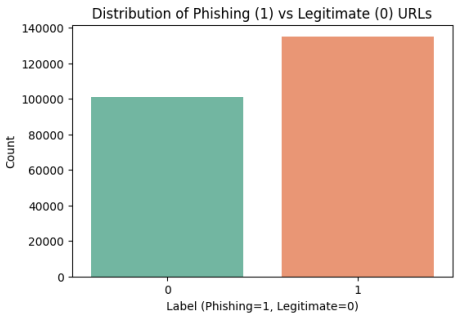

Figure 1. EDA - Class Distribution

It includes 54 features across three broad categories:

- **URL-Specific Features:** Length-related metrics (e.g., `URLLength`, `DomainLength`), security indicators (e.g., `IsHTTPS`, `IsDomainIP`), and URL-title correlation scores.
- **Page Content Features:** Includes elements like `NoOfImage`, `NoOfCSS`, `HasHiddenFields`, and sensitive keywords (e.g., `Bank`, `Pay`, `Crypto`).
- **Technical Features:** Structural attributes such as `LineOfCode`, `LargestLineLength`, and redirect behaviors (`NoOfURLRedirect`, `NoOfSelfRedirect`).

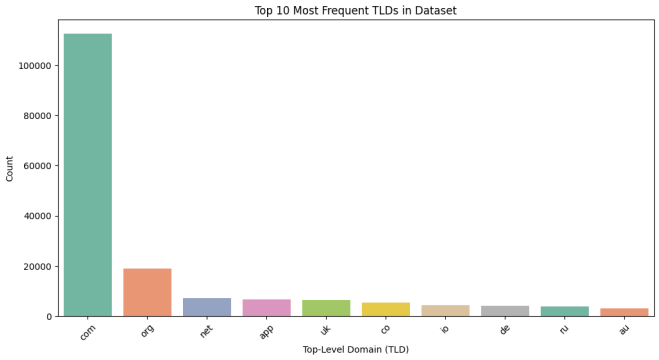

Figure 2. EDA - Frequency of TLD Occurrences

3. Dataset Challenges and Their Solutions

- **Outliers:** Certain features exhibited outliers that were addressed this using statistical techniques to detect and handle outliers.

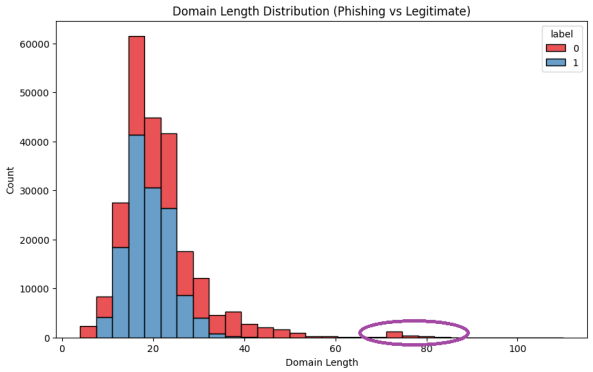

Figure 3. Detecting Outliers - Example 1

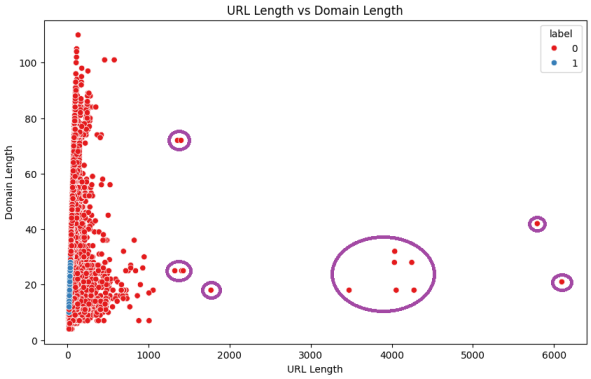

Figure 4. Detecting Outliers - Example 2

- **Feature Redundancy:** High correlation among some features led to redundancy. These were resolved by removing one of each pair of highly correlated features after thorough analysis. Ultimately, we reduced our dataset to 36 features.

- **Class Imbalance:** There was an imbalance in the original dataset with respect to the number of samples per label. We balanced this out by undersampling from the class with the higher number of samples to have a balanced dataset, with 92970 samples in each class.

- **Scalability:** Processing a large dataset posed computational challenges. Dimensionality reduction methods, such as CUR, SVD, and Sparse JL matrices, were applied to enhance scalability.

- **Further Pre-Processing:** To prepare the data for model training, we normalized the continuous features to ensure compatibility with machine learning models. We also performed label encoding to transform categorical features into numeric values.

4. Hypothesis Tests and Their Conclusions

To analyze feature significance, several hypothesis tests were conducted, with a level of significance of 5% in each. We took 10000 samples, sampled at random, and performed each of the following tests:

4.1. Paired t-Test:

Test: *Comparing Whether the Distribution of URL Lengths is same across Phishing and non-Phishing URLs*

Null Hypothesis: The distributions (Means) are the same.

Alternate Hypothesis: The distributions are not the same.

After performing the paired t-test, we got a test statistic of -37, much less than the critical value of -1.96, so we **reject the null hypothesis** and determine that **the mean lengths of URL lengths across phishing and non-phishing URLs do indeed differ**.

4.2. T-Test for Correlation Analysis:

Test 1: *Examining if there is a linear relationship between number of digits and letters in a URL.*

Null Hypothesis: There is no linear relationship.
Alternate Hypothesis: There is a linear relationship.

After performing the t-test for correlation coefficient, we got a test statistic of 53, much more than the critical value of 1.96, so we **reject the null hypothesis** and determine that **there is a linear relationship between number of digits and letters in a URL**.

Test 2: *Examining if there is a linear relationship between URL length and Domain Length.*

Null Hypothesis: There is no linear relationship.
Alternate Hypothesis: There is a linear relationship.

After performing the t-test for correlation coefficient, we got a test statistic of 49, again much more than the critical value of 1.96, so we **reject the null hypothesis** and determine that **there is a linear relationship between URL length and Domain Length**.

4.3. Chi-Squared Tests for Dependency:

Test 1: *Examining if a Phishing URL depends on it having a title.*

Null Hypothesis: They are independent.
Alternate Hypothesis: They are dependent on each other occurring.

We got a test statistic of 1904, much more than the critical value so we **reject the null hypothesis** and determine that **a phishing URL does indeed depend on it having a title**.

Test 2: *Examining if a Phishing URL depends on being an HTTPS link.*

Null Hypothesis: They are independent.
Alternate Hypothesis: They are dependent on each other.

Again, we get a statistic of 3620 so we **reject the null hypothesis** and determine that **a phishing URL does indeed depend on it being an HTTPS link**.

4.4. Broader Hypothesis Testing:

It aims to assess the relevance of each integer and float feature in predicting the label using various hypothesis tests learned in the course and conduct a gain analysis on binary features.

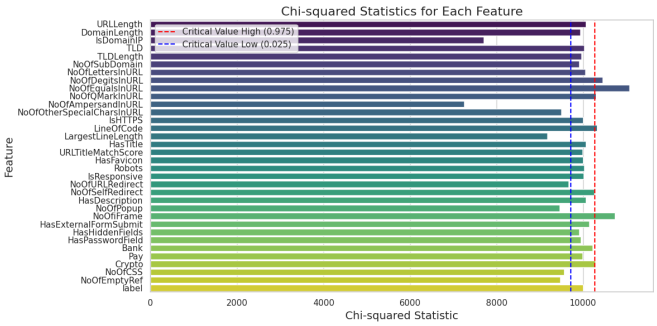

Figure 5. Broader Hypothesis Testing - Chi Squared Test

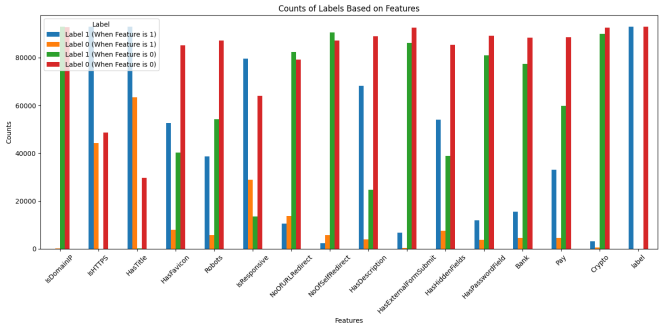

Figure 6. Broader Hypothesis Testing - Label vs Binary Feature Relevance

5. ML Models Used

We used the following machine learning models and trained them on our dataset, after optimizing them with the help of the GridSearchCV function in Python:

- **Random Forests:** Many smaller decision trees are created on a subset of the data and a subset of the features. Then, on any input, the mode of the output of each tree is the final prediction.
- **Gradient Boosting:** Gradient Boosting combines several weak learners into strong learners, in which each new model is trained to minimize the loss function of the previous model. The predictions of the new model are then added to the ensemble, and the process is repeated until a stopping criterion is met.
- **Logistic Regression:** It is a statistical model used for binary classification. It predicts the probability of an outcome (0 or 1) by applying the logistic (sigmoid) function to a linear combination of input features. The output is a probability, and a threshold (usually 0.5) is applied to classify the result.

6. Scaling Techniques and Benefits

Dimensionality reduction methods played a crucial role in optimizing performance:

- **Coreset:** A coreset is a smaller, weighted subset of the data that approximates the original dataset. This coreset is created by randomly sampling the data points, ensuring that the selected subset retains the key information needed for the task.
- **JL Matrix:** We transform the data into a lower dimension with the help of a JL matrix (whose entries are sampled from N(0, 1/k), where k is the size of the reduced number of dimensions, in

our case, k = 10). The JL lemma ensures that distance between points is maintained even after transformation, a useful property that maintains structure. We also try out a sparse JL matrix to reduce computation time.

- **SVD:** We transform the data into a lower dimension using SVD by reducing it to its top k singular components, with k = 10 This transformation preserves the most significant structure in the data, by capturing the principal directions of variation. This dimensionality reduction technique is particularly useful in maintaining the relationships and patterns within the data, which are crucial for classification tasks.

- **CUR:** It reduces the size of the feature space by selecting a subset of important columns (features) based on statistical leverage scores. We sample the most important columns and rows to perform the decomposition. Keeping the actual features in the dataset, we maintain the meaning the features had in the original dataset. This is in contrast to SVD-based techniques like PCA, where interpretability can be lost as the features are a combination of the original columns.

7. Model Performance on Scaled and Unscaled Datasets

We evaluated Random Forests, Gradient Boosting, and Logistic Regression on both scaled and unscaled datasets. The results are summarized below.

7.1. Unscaled Dataset

Algorithm	Train Accuracy	Test Accuracy	Runtime (in s)
Random Forests	100	99.975	6.4
Gradient Boosting	100	99.978	35.7
Logistic Regression	99.93	99.935	3.4

Table 1. Train and Test Accuracy For Different Algorithms on the Unscaled Dataset

7.2. Scaled Dataset

- **Coreset:**

Algorithm Coreset Size	Train Accuracy	Test Accuracy	Runtime (in s)
Random Forests 160	100	99.64	6
Logistic Regression 270	99.924	99.477	0.6

Table 2. Train and Test Accuracy For Different Algorithms on Dataset Scaled by Using Coresets

- **JL Matrix:**

Algorithm	Train Accuracy	Test Accuracy	Runtime
Random Forests	100	99.075	20.1s
Gradient Boosting	100	99.308	6m 11.6s
Random Forests (with Scaled JL)	100	99.875	7.8s
Gradient Boosting (with Scaled JL)	100	99.957	1m 5.9s

Table 3. Train and Test Accuracy For Different Algorithms using JL matrix

- **SVD:**

Algorithm	Train Accuracy	Test Accuracy	Runtime
Random Forests	100	99.59	17.4s
Gradient Boosting	100	99.67	4m 53.2s

Table 4. Train and Test Accuracy For Different Algorithms on Dataset Scaled Using SVD

- **CUR:**

Algorithm	Train Accuracy	Test Accuracy	Runtime
Random Forests	99.999	99.695	5.4s
Gradient Boosting	99.999	99.747	1m 20.9s

Table 5. Train and Test Accuracy For Different Algorithms on Dataset Scaled Using CUR

8. Conclusion and Future Work

This project successfully developed a phishing URL detection model with high accuracy and efficiency. Key takeaways include:

- Random Forests and Gradient Boosting consistently outperformed Logistic Regression in classification tasks, with Random Forests also being extremely lightweight.

- Dimensionality reduction techniques like Coresets reduced runtime significantly while maintaining a very high accuracy.

- In general, JL, CUR and SVD didn't improve efficiency substantially. However, sparse JL was found to be much better than the regular matrix.

Future Directions:

- Explore advanced machine learning and deep learning models, such as autoencoders, for improved scalability and accuracy.

- Incorporate real-time URL behavior analysis for dynamic phishing detection.

By advancing detection methods and promoting user awareness, we can create a safer online environment.

References

[1] Arvind Prasad and Shalini Chandra. PhiUSIIL Phishing URL (Website). UCI Machine Learning Repository, 2024. DOI: https://doi.org/10.1016/j.cose.2023.103545.