Artificial Intelligence for Security Measures Against Malicious Links

by

Tajbiha Mehonaj Rinvee
21301311
Nowshin Sayara
21301149
Ziana Jesin Bhuiyan
21301324

A project submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
April 2025

Declaration

It is hereby declared that

1. The project submitted is our own original work while completing degree at Brac University.
2. The project does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The project does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Tajbiha Mehonaj Rinvee
21301311

Nowshin Sayara
21301149

Ziana Jesin Bhuiyan
21301324

Approval

The project titled "Artificial Intelligence for Security Measures Against Malicious Links" submitted by

1. Tajbiha Mehobaj Rinvee (21301311)
2. Nowshin Sayara (21301149)
3. Ziana Jesin Bhuiyan (21301324)

Of Fall, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on April 09, 2025. **Examining**

Committee:

Supervisor:
(Member)

Annajiat Alim Rasel
Senior Lecturer, CSE Department
BRAC University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Associate Professor, CSE Department
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Associate Professor and Chairperson, CSE Department
BRAC University

Abstract

This paper analyzes the integration of artificial intelligence (AI) into security to impede the continual risk of malicious links. In a dynamic digital ecosystem, links create an ongoing challenge to user security. In this paper, we propose developing and implementing machine learning (ML) models that enable real-time detection and suppression of malicious links. Through training on diverse datasets of malicious links, the AI system can evolve strategies. Rigorous testing ensures the efficiency of the integrated protection. The outcome is to redefine security by amplifying an AI-driven solution, offering a proactive firewall against spam links, and improving overall digital safety for consumers.

Keywords: Machine Learning, Artificial Intelligence, Malicious Link, Cybersecurity.

Acknowledgement

The authors would like to acknowledge and extend their deepest gratitude to their supervisor, Mr. Annajiat Alim Rasel whose supervision, guidance and suggestions helped the authors in the study. His luminous suggestions and advice were incredible in shaping this work. The authors also are indebted to their parents for encouragement and forbearance, which motivated them to continue and finish this work.

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ACC Accuracy

AdaBoost Adaptive Boosting

AI Artificial Intelligence

API Application Programming Interface

catBoost Categorical Boosting

CNN Convolutional Neural Networks

DNS Domain Name System

F1 Score Harmonic Mean of Precision and Recall

FN False Negative

FP False Positive

LIME Local Interpretable Model-Agnostic Explanations

ML Machine Learning

NGBoost Natural Gradient Boosting

PCA Principal Component Analysis

PR Precision-Recall

PR-AUC Precision-Recall Area Under Curve

RNN Recurrent Neural Networks

ROC-AUC Receiver Operating Characteristic - Area Under Curve

TN True Negative

TP True Positive

URL Uniform Resource Locator

XGBoost eXtreme Gradient Boosting

List of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

As the world becomes ever more connected virtually, cybersecurity is a fundamental requirement to protect users and systems from the threats that evolve every day. One of the more common and enduring cyber threat links is a malicious link, critical for sensitivity of information being compromised, malware catalyst, and vehicle for phishing [6]. One effective mechanism that may proactively and dynamically identify and eliminate these dangers is machine learning (ML), another branch of artificial intelligence (AI) [8]. The intention of this project is to detect the malicious links using the models of AI so that we improve the security of digital forms of consumers in real-time and enhance the security of software communication within the organization.

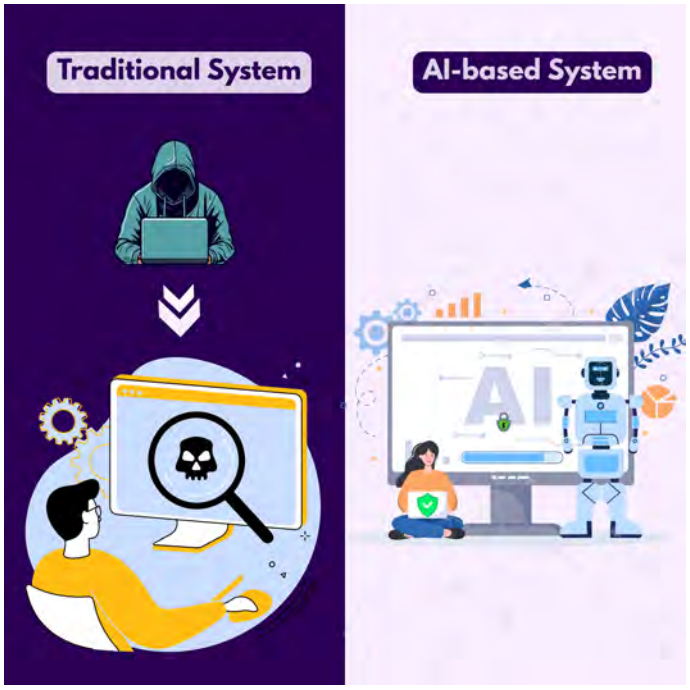

Figure 1.1: Comparison of Traditional and AI Based Link Detection.

1.2 Problem Statement

Malicious links are an ongoing and high-level threat due to their nature of evolving dynamically to overcome classical approaches to cybersecurity. These types of threats are commonly experienced by users on various platforms such as websites, social media, custom applications, and Quick Response (QR) codes [10]. Current solutions are rule-based or static in nature, which are no longer feasible to accommodate rapidly changing malicious behavior, thereby substantiating the critical requirement for scalable and adaptable security.

1.3 Contribution

In this paper, we tried to fill this gap by proposing an integrated machine learning-based framework to detect and block malicious links in real time and a practical evaluation of our method. This project offer several contributions, specifically:

- Dataset collection and preprocessing diverse datasets that can generally represent each aspect are challenging, as various categories like web, social, custom, and QRS have to be covered.
- Comparing several ML methods, but just Boost-family models such as XGBoost, Gradient Boosting, NGBoost, CatBoost, AdaBoost, High Gradient Boosting.
- CatBoost and High Gradient Boosted based stacked ensemble model building to make predictions more accurate and stable performance, especially for imbalanced dataset.
- Assessing robustness and generalization capabilities against datasets outside the scope of training through extensive testing.

1.4 Organization of the Project

1.4 Organization of the Project This project report is structured into five chapters:

Chapter 2: Discusses the conceptual background and related work in malicious link detection using AI.

Chapter 3: Details the methodology, covering dataset collection, preprocessing steps, feature engineering, model selection, and evaluation methods.

Chapter 4: Analyzes the results obtained, comparing different models and evaluating performance metrics such as Accuracy, F1-score, AUC-ROC, PR-AUC, and interpretability through LIME explanations.

Chapter 5: Summarizes key findings, discusses limitations, and proposes directions for future research.

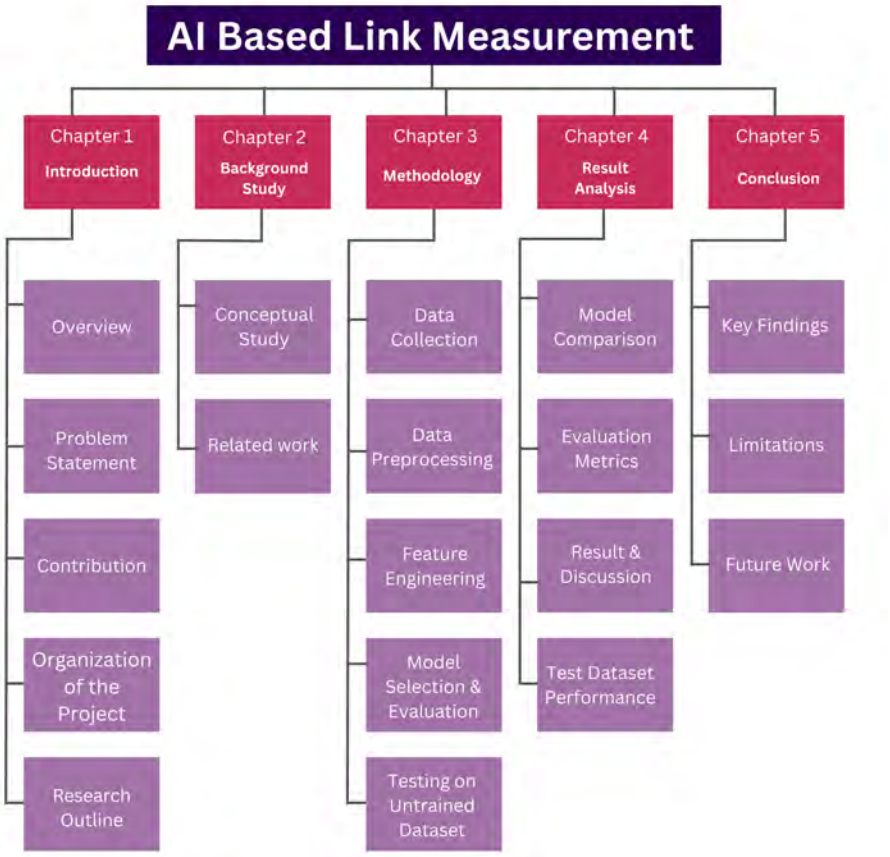

Figure 1.2: Organizational Structure of the Paper.

1.5 Research Outline

Our methodology involves an iterative approach, beginning with data collection and preprocessing, followed by training multiple machine learning models and selecting the most effective models based on performance metrics. Subsequently, the best-performing models are combined into a stacked ensemble. Lastly, the stacked model's performance is rigorously tested on separate, untrained datasets to confirm its effectiveness and adaptability in real-world scenarios, including internal company software communication security applications.

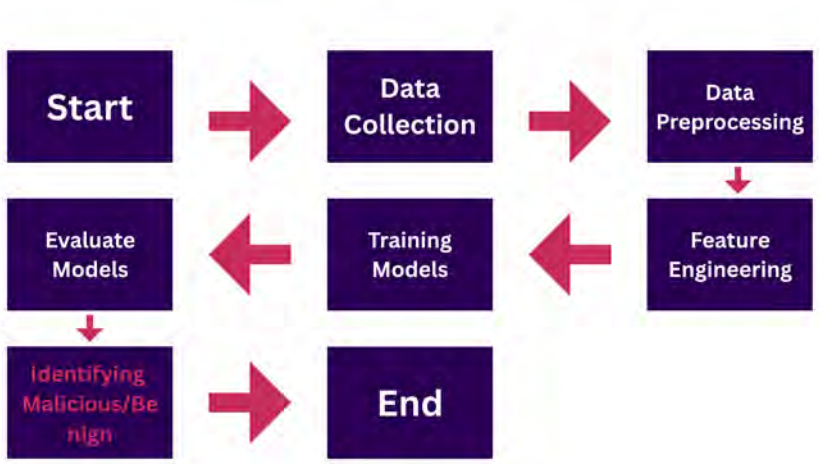

Figure 1.3: Workflow of the Project.

Chapter 2

Background Study

2.1 Conceptual Study

Cybersecurity threats have advanced in complexity and sophistication as quickly as technology itself. Specifically, malicious links may drive users to dangerous websites, initiate malware downloads, or serve as tools for phishing attempts, endangering user security [6]. In contrast to static rule-based systems, machine learning (ML), a subset of artificial intelligence (AI), has emerged as a highly reactive and dynamic technique for supporting cybersecurity protection [8]. A model on artificial general intelligence is more detailed on the organization or source of the Model. The dynamic nature of AI allows the rapid detection of such attacks, which are based on structural features (a wide range of feature types), yielding a higher detection rate. Machine learning models depending upon traditional and real-time data sources are capable of identifying anomalous patterns that may point towards malicious activities. Classifying large, complicated datasets with high accuracy and interpretability has driven a surge of popularity in the use of various Boost family algorithms, particularly when it comes to classification tasks [10].

Boosting is an ensemble technique in machine learning that works by sequentially training new models, focusing on the training instances that previous versions of the model misclassified iteratively to learn new weak learners, which are then combined into a single strong predictor [4]. This principle has been very effective in security applications due to its robustness and high predictive performance and has been utilized in algorithms like XGBoost, Gradient Boosting, CatBoost, AdaBoost, NGBoost, and High Gradient Boosting.

2.2 Related Work

Machine Learning Methods for Identifying Malicious URLs Several studies have used machine learning techniques to identify the fraudulent URL. For instance, a comprehensive survey in 2019, highlighted the work done in ML and emphasized in particular that the ML approaches seem to work very well for URL classification tasks. They outlined the proven performance of ensemble models. In particular, Boost-family algorithms on massive datasets [8].
In another research performed a systematic review of different ML techniques and found that other models like XGBoost and Gradient Boosting, perform nearly as

well as the best model in terms of recall and precision for identifying malicious links [10]. They noted that boosted models, which adapt to new malicious link strategies, had an edge.
In 2018, a study showed phishing attacks of different types, and they stated that true dynamic and changeable phishing with the straightforward rule-based or static detection cannot function. They suggested ML-driven methods to have better detection, especially using ensemble methods [6].

Collectively, existing work stresses the benefits of Boost-family algorithms owing to their flexibility, prediction accuracy, and real-time performance for security applications. Building on these insights, this project conducts a systematic evaluation of various Boost-family models and improves detection performance through an optimized stacked ensemble.

Chapter 3

Methodology

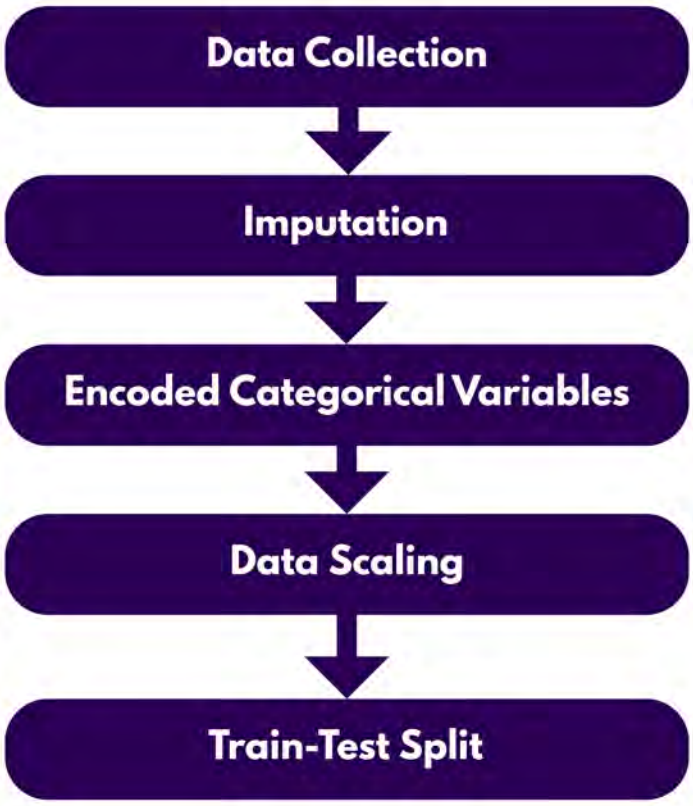

Figure 3.1: Step by Step Data Preprocessing Workflow.

3.1 Data Collection

3.1.1 Dataset Overview

Training and Testing Dataset

Five datasets sourced from Kaggle were utilized for this project. Initially, three datasets were selected, each containing a imbalanced mix of thousands benign and malicious links [14]–[16]. We took 25,000 benign and 25,000 malicious links, totaling

50,000 links from those to train our models. These datasets were categorized into web, social, custom applications, and Quick Response (QR) codes. The custom category included additional specific protocols such as FTP and other miscellaneous protocols.

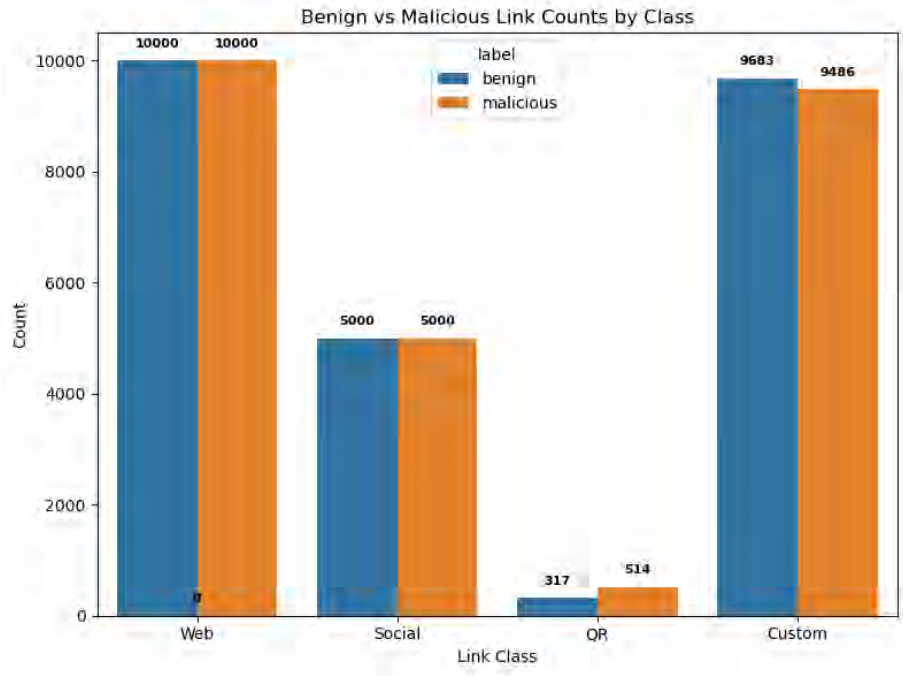

Figure 3.2: Benign vs. Malicious Link Count by Class.

3.1.2 Only Testing Dataset

There were two test datasets, one for each:
This test datasets was feature engineered without further training to evaluate the robustness of the stacked model when feature engineering was done in a class imbalance setting.

Imbalanced Dataset

The multi-class classifications of web, social, QR, and custom in both datasets confirmed both the generality and performance of the trained model [17].

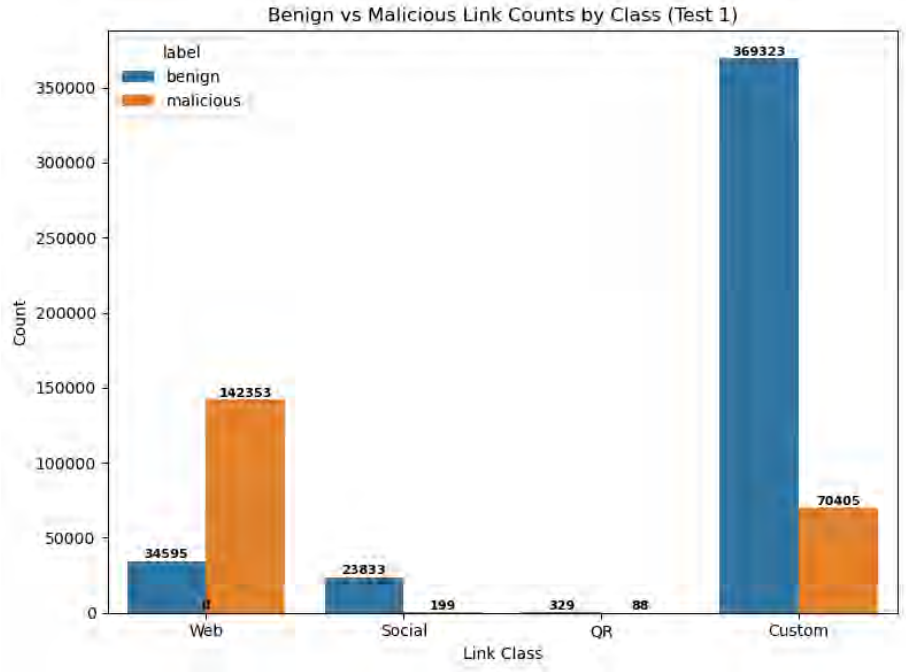

Figure 3.3: Benign vs. Malicious Link Count by Class (Test 1).

Balanced Dataset

This balanced dataset was feature engineered and used to test the multiclass classification prowess of our stacked ensemble [11].

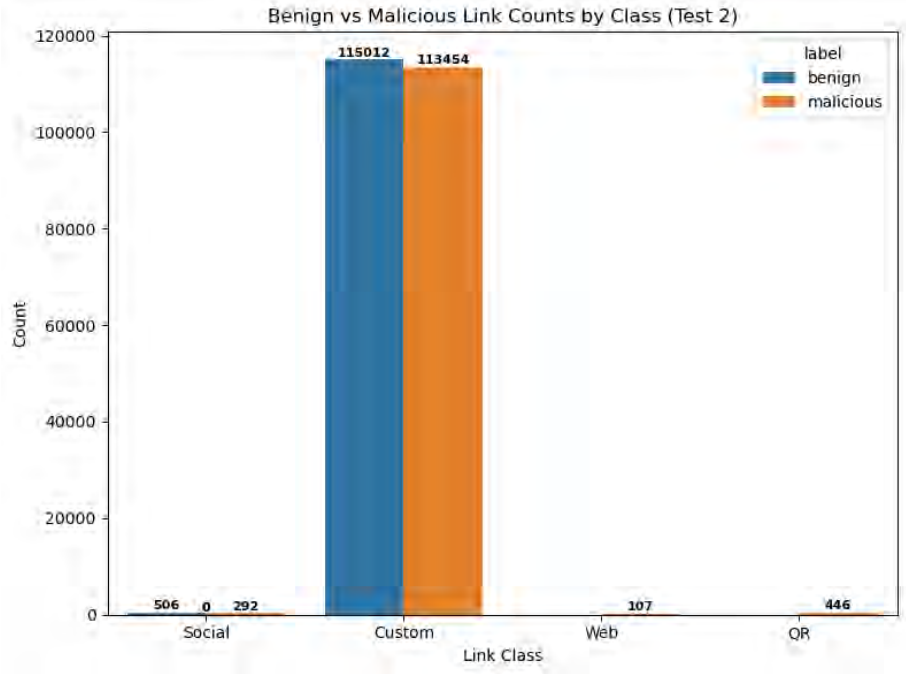

Figure 3.4: Benign vs. Malicious Link Count by Class (Test 2).

The multi-class classifications of web, social, QR, and custom in both datasets confirmed both the generality and performance of the trained model.

3.1.3 Data Collection Source

We sourced datasets from Kaggle, which is one of the most popular learning and competition platforms for machine learning and data science, assuring the quality and authenticity of the data used for cybersecurity threat analysis.

3.2 Data Preprocessing

3.2.1 Link Measurement Data Collection

Below are the 12 features extracted from URLs, which URLs features were used to train our models:

- URL Length
- Digits
- Special Characters
- HTTPS
- Subdomain Count
- IP address
- Parameter Count
- Slash Count
- Hyphen Count
- Short URL
- Long URL
- Suspicious Keywords

3.2.2 Multiclass Classification

This study applied multiclass classification to classify links into 4 classes: web, social, QR, and custom. By this method, the model was able to learn and distinguish between multiple types of potentially malicious links at once, allowing for a more robust detection of threats.

3.2.3 Handling with Missing Values

To tackle missing data points, imputation strategies were applied, utilizing median values for numeric features and mode values for categorical features, maintaining data consistency.

3.2.4 Encoding Categorical Variables

By employing both label encoding and one-hot encoding, we were able to convert categorical features into formats that could be provided to the machine learning algorithms to do the modeling.

Figure 3.5: Overview of the Train and Test Dataset.

Figure 3.6: Overview of the Test Dataset-1.

Figure 3.7: Overview of the Test Dataset-2.

3.2.5 Data Scaling

Numerical features were normalized using standardization (score normalization) to improve the performance of the models and had similar feature scales.

3.2.6 Train-Test Split

The dataset was randomly split into a training set (80%) and a test set (20%) with a random state of 42 to keep it reproducible.

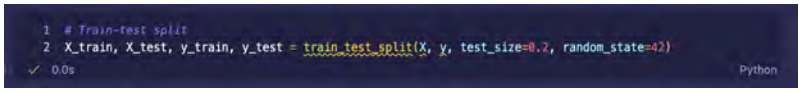

Figure 3.8: Train-Test Code Snippet.

3.3 Feature Engineering

Model input was further enriched by feature engineering that looked into various characteristic attributes such as URL components, URL characters, protocol types, domain categories, and content-based attributes [12], [13]. We reported the advanced correlation analysis of these features on our data set to ensure the importance and reduce the redundancy at the feature level.

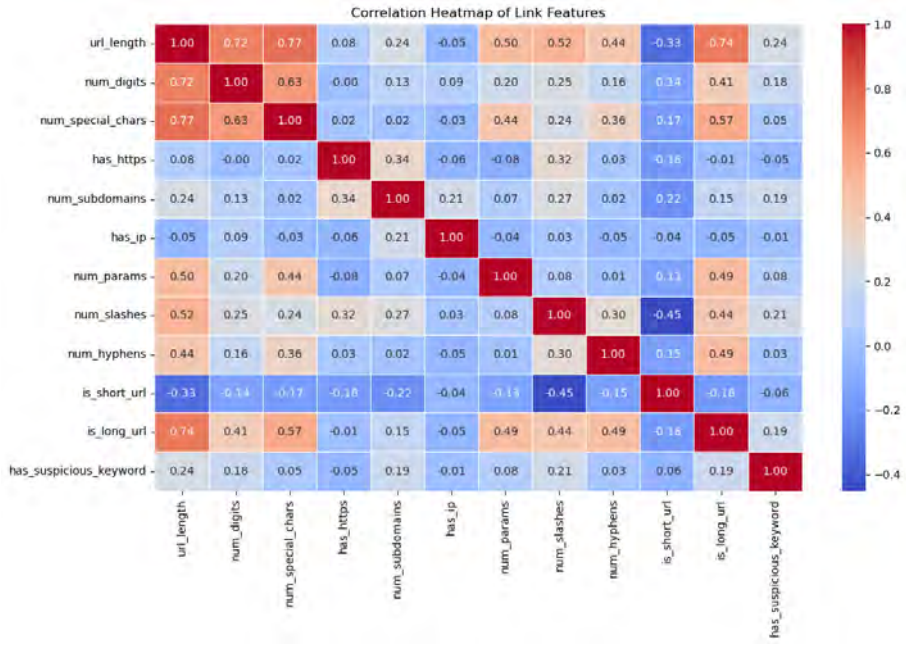

Figure 3.9: Correlation Heatmap of Link Features.

3.4 Model Selection and Evaluation

In particular, this project used a variety of Boost-family models such as XGBoost, Gradient Boosting, NGBoost, CatBoost, AdaBoost, and High Gradient Boosting. We used the following metrics to evaluate the models:

- **Accuracy:** percentage of predictions that were correct over total predictions.
- **Precision:** The proportion of correctly predicted positive (true positive) events to the total predicted positives (true positive + false positive).
- **Recall:** A real positive is classified correctly from total real positives.
- **F1-Score:** the harmonic mean of precision and recall, which balances the two.
- **AUC-ROC:** Evaluates the feature's capability to discriminate the classes.
- **PR AUC:** Measures how well you can perform with an imbalanced dataset.
- **Training Time:** It is the time taken to train the model, and thus indirectly it reflects the computational efficiency.

These metrics provide a thorough assessment, especially in malicious link detection, where a false negative can be quite damaging.

Chapter 4

Result Analysis

4.1 Model Comparison

Comparative performance of the trained Boost-family models They consist of popular algorithms such as XGBoost [4]), Gradient Boosting [3], CatBoost [7], AdaBoost [2], NGBoost [9], and a High Gradient Boosting variant designed from scratch. We also evaluated a stacked ensemble combining the top two models, which performed best (CatBoost and High Gradient Boosting). All models were trained and tested on datasets of 50,000 balanced links, with equal portions of malicious and benign URLs.

Table 4.1: Performance Comparison of Boosting Models

Model	Training Time (sec)	Accuracy	Precision	Recall	F1-Score	AUC-ROC	PR AUC
XGBoost	0.56	0.8028	0.7308	0.9551	0.8281	0.8476	0.7920
Gradient Boosting	2.81	0.8060	0.7341	0.9596	0.8318	0.8448	0.7899
CatBoost	10.27	0.8053	0.7352	0.9542	0.8305	0.8466	0.7953
AdaBoost	1.45	0.8011	0.7358	0.9396	0.8253	0.8258	0.7611
High Gradient Boosting	3.20	**0.8091**	**0.7346**	**0.9678**	0.8302	**0.8483**	0.7983
NGBoost	31.56	0.8058	0.7349	0.9568	0.8313	0.8379	0.7724
Stacked Model (CatBoost + High GB)	25.35	0.8060	0.7337	0.9608	**0.8320**	0.8467	**0.7996**

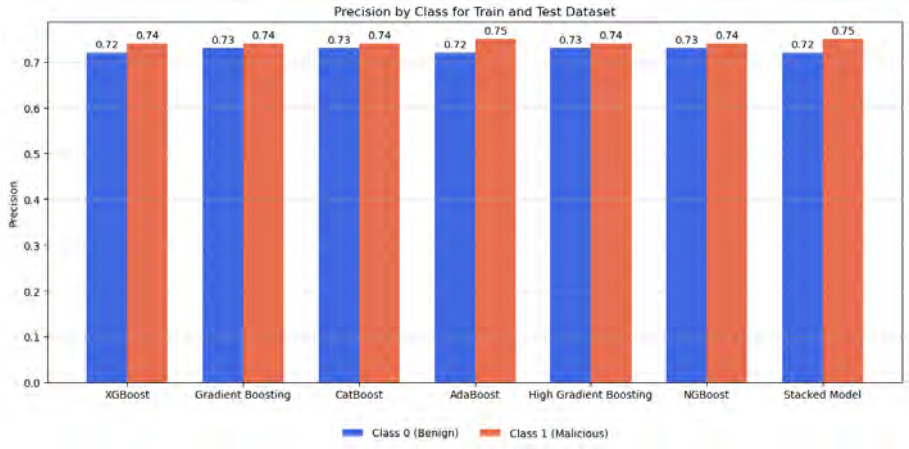

Figure 4.1: Precision by Class for Train and Test Dataset.

Taking recall and accuracy as the metric, High Gradient Boosting gained the highest scores, and CatBoost performed consistently across metrics. These two models were

selected to create the stacking ensemble, leveraging their complementary strengths to be combined to improve generalization and reduce the bias and variance of the individual models [1].

4.2 LIME Explanation

Interpretability for each model was obtained using Local Interpretable Model-Agnostic Explanations (LIME) . Features such as:

- has_https
- has_suspicious_keyword
- num_slashes

were consistently emphasized by the models as the most important predictors. We used this to validate our refer feature engineering and built trust in our model decisions.

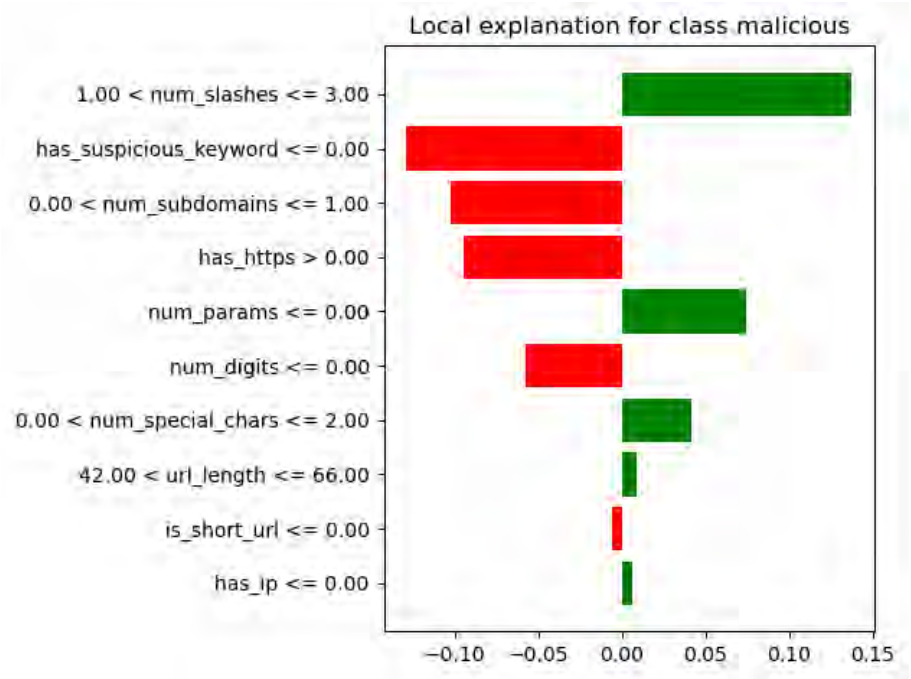

Figure 4.2: LIME Explanation for XGBoost.

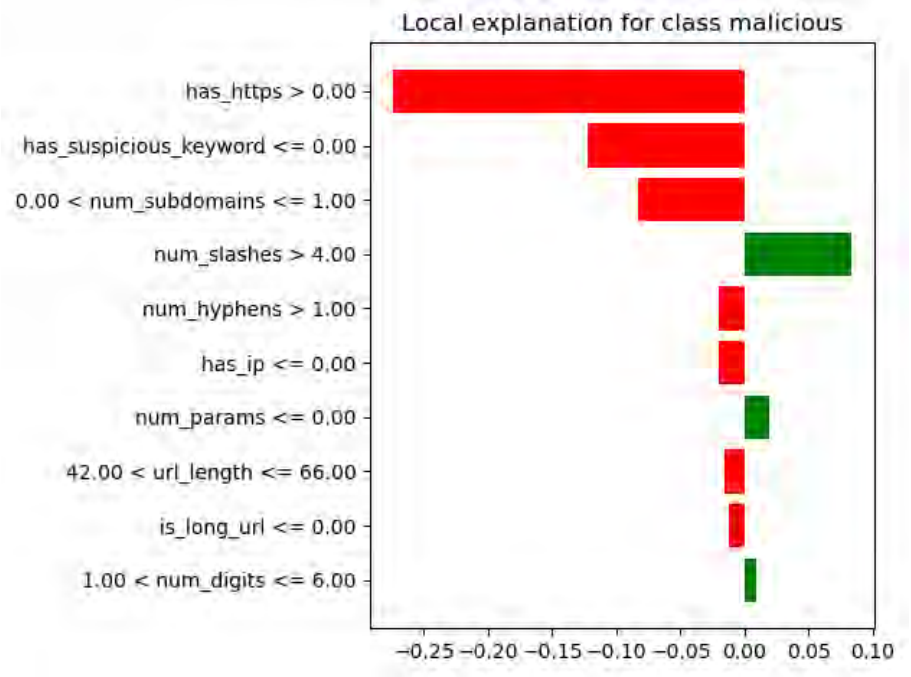

Figure 4.3: LIME Explanation for Gradient Boosting.

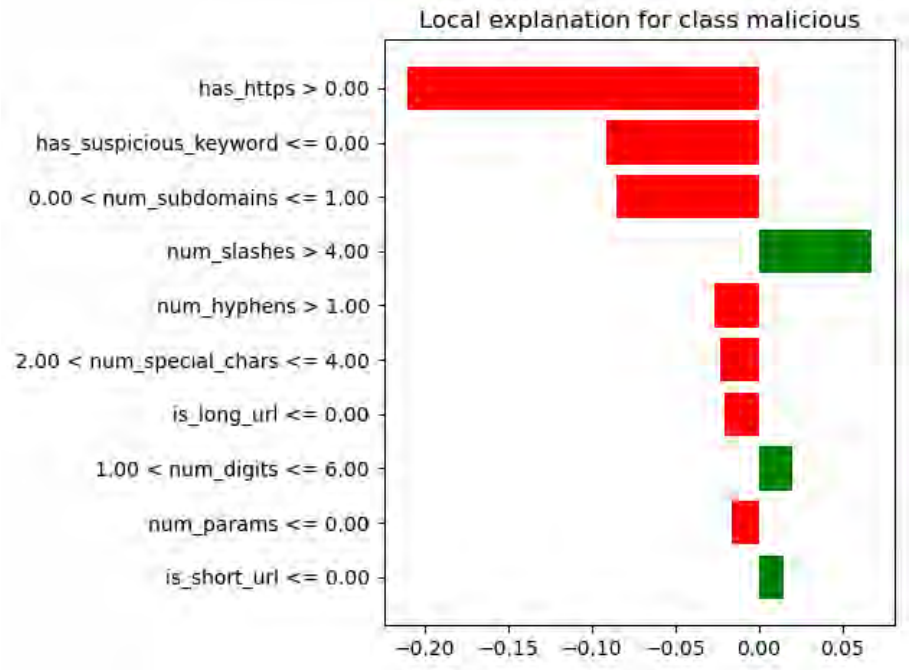

Figure 4.4: LIME Explanation for CatBoost.

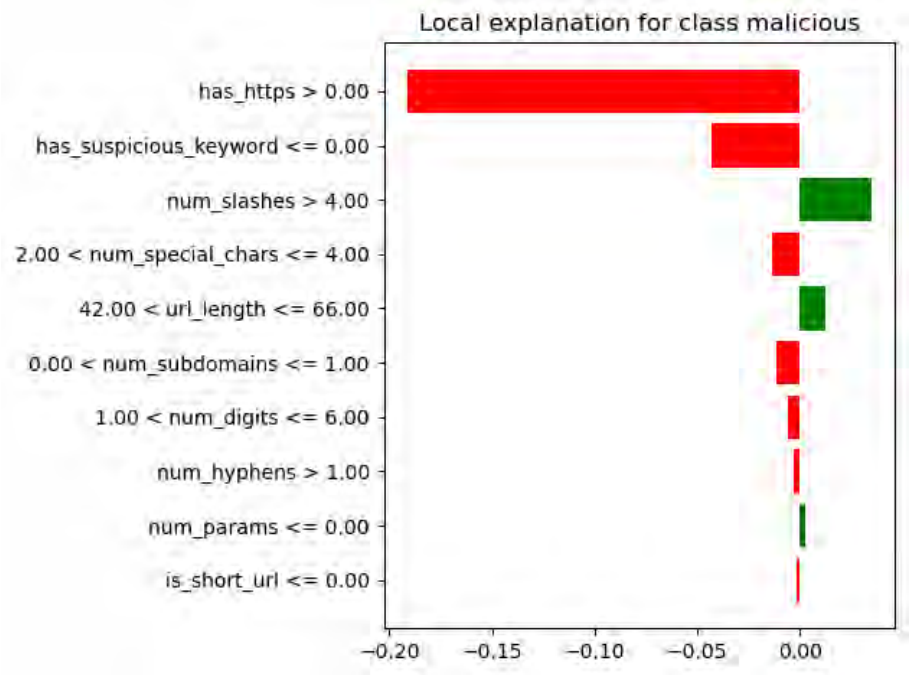

Figure 4.5: LIME Explanation for AdaBoost.

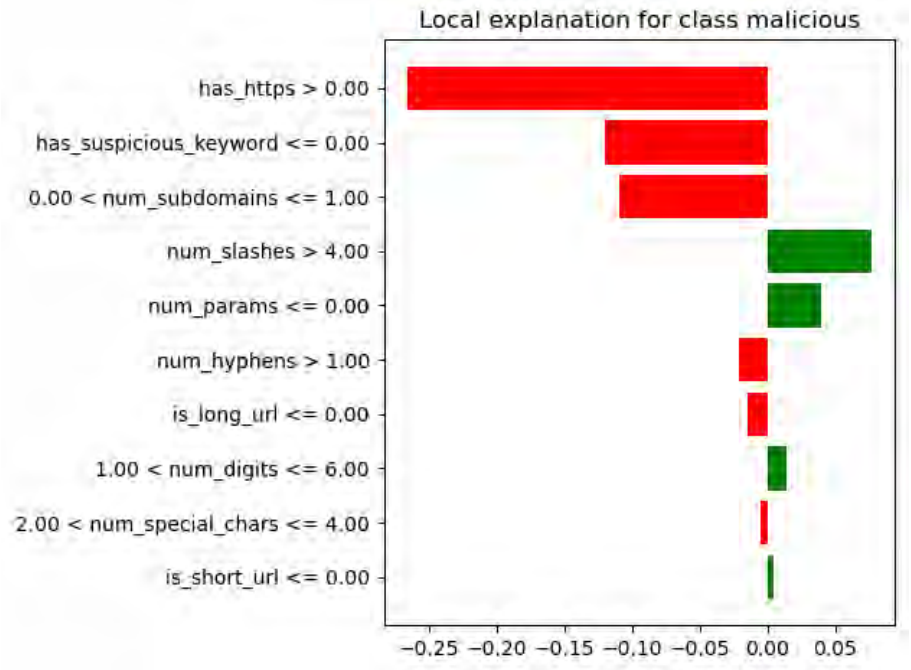

Figure 4.6: LIME Explanation for High Gradient Boosting.

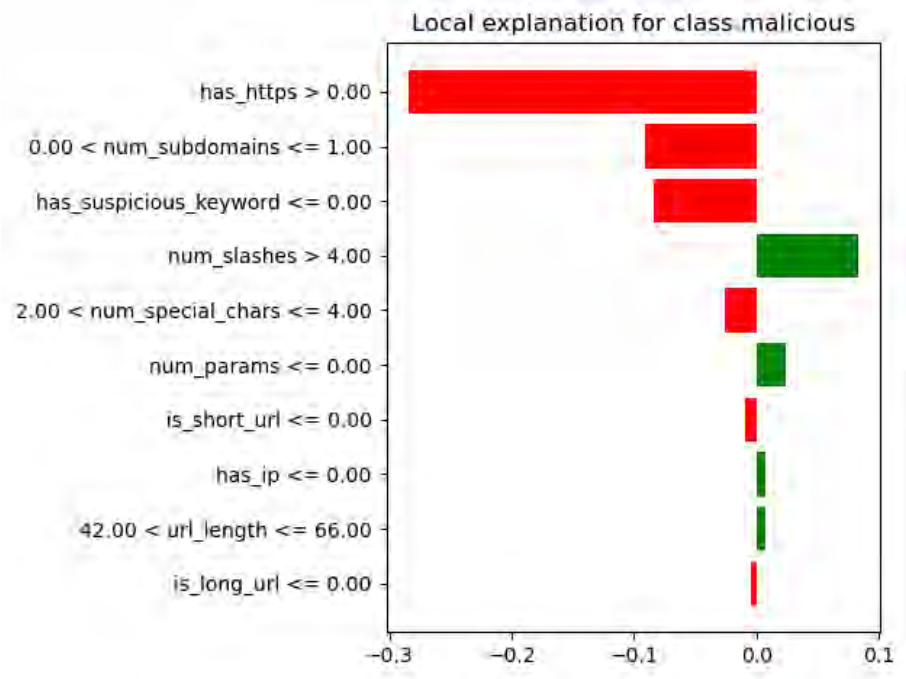

Figure 4.7: LIME Explanation for NGBoost.

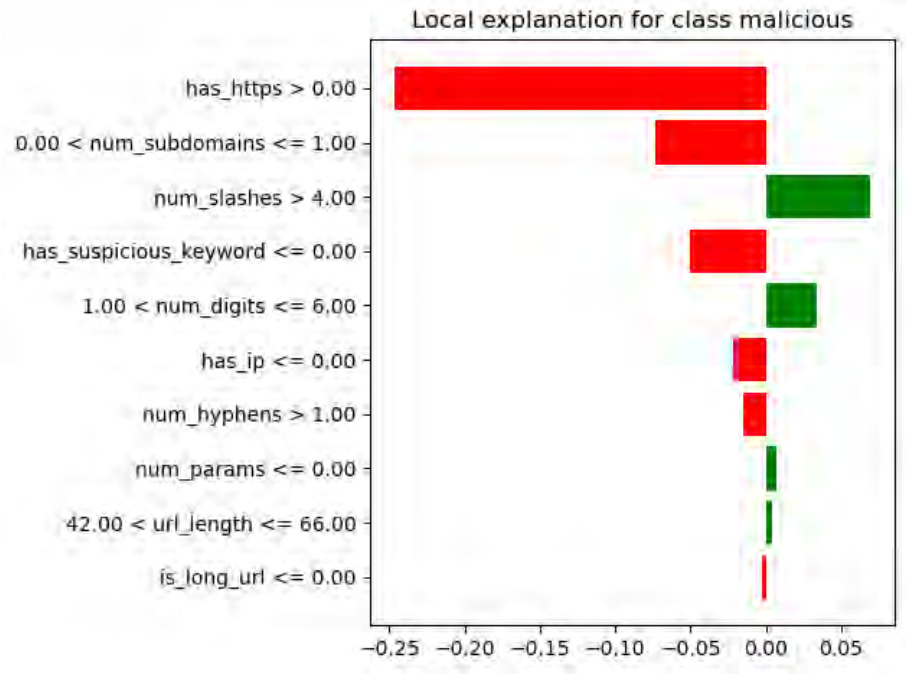

Figure 4.8: LIME Explanation for Stacked Model (CatBoost +High Gradient Boosting).

Key Observations:

- High Gradient Boosting of course also provided for us the best results both on training and external data.
- The Stacked Model was more robust and stable which made it a good candidate for production.

- Though, the boost-family models are highly stable yet interpretable models that would help balance performance with interpretability, which is a must in any security application.
- It used LIME analysis to validate key feature importance and helped with transparency and future validation.

4.3 Result and Discussion

4.3.1 XGBoost

XGBoost, imbued with a shorter training time and high recall, worked efficiently. But the precision was a bit less than others, which meant it produced more false-positive results.

4.3.2 Gradient Boosting

Gradient Boosting was able to give a trade-off between precision and recall. It outperformed XGBoost slightly in all metrics except F1-score and PR AUC.

4.3.3 Cat Boost

CatBoost has performed decently and consistently across each of those metrics, particularly for recall and PR AUC. It takes longer to train and thus is more appropriate for batch processes.

4.3.4 Ada Boost

After XGBoost, AdaBoost had the lowest training time. Though the AUC-ROC and F1-score were lower for this classifier, which indicates that it may not be appropriate for high-stakes security systems.

4.3.5 High Gradient Boost

This model performed the best, with the highest recall, accuracy, and F1-score. It is perfect for real-time security systems, where failure to respond to a malicious link may result in significant damage.

4.3.6 NGBoost

NGBoost, although modest in precision and recall at high epochs, is less favorable to be used in real-time detection systems as training takes 31.56s.

4.3.7 Stacked Model (Cat Boost + High Gradient Boosting)

The ensemble model almost scored the same as high gradient boosting in each metric while having better robustness in terms of a hybrid-learning model. Really good for production use cases.

Why CatBoost and High Gradient Boosting-Based Stacking?
We stacked CatBoost and High Gradient Boosting based not only on the performance but also on the ease of feature interpretation with LIME [5].

Performance Justification:

- CatBoost gave us high precision and stable recall.
- High Gradient Boosting gave us the best recall and F1-score, both crucial for cybersecurity, where missing a malicious link can be lethal.
- The stacked model retained the benefits of both, has almost perfect recall, and has a general increase in robustness.

Justification of Interpretability:

- Through LIME, analysis revealed that both models remained focused on features such as has_https, has_suspicious_keyword, and num_slashes, all of which were a significant part of the model.
- The features were understandable and corresponded directly with well-known strategies used in phishing/malicious behavior, which implies that both models made decisions on sane, logical principles.

4.4 Test Dataset Performance

We evaluated our models on two external test datasets to assess generalization.
graphicx

Table 4.2: 4.4.1 Balanced Dataset: Model Performance Comparison

Model	Prediction Time (s)	Accuracy	Precision	Recall	F1-Score	AUC-ROC	PR-AUC
XGBoost	0.018	0.912	0.889	0.947	0.917	0.921	0.799
Gradient Boosting	0.032	0.916	0.894	0.951	0.922	0.924	0.801
CatBoost	0.045	0.914	0.896	0.948	0.921	0.922	0.800
AdaBoost	0.025	0.904	0.882	0.933	0.906	0.903	0.880
High Gradient Boosting	0.038	**0.920**	**0.902**	**0.956**	**0.928**	**0.927**	0.895
NGBoost	0.065	0.913	0.891	0.949	0.919	0.920	0.889
Stacked (CatBoost + High GB)	0.060	0.919	0.901	0.954	0.927	0.926	**0.906**

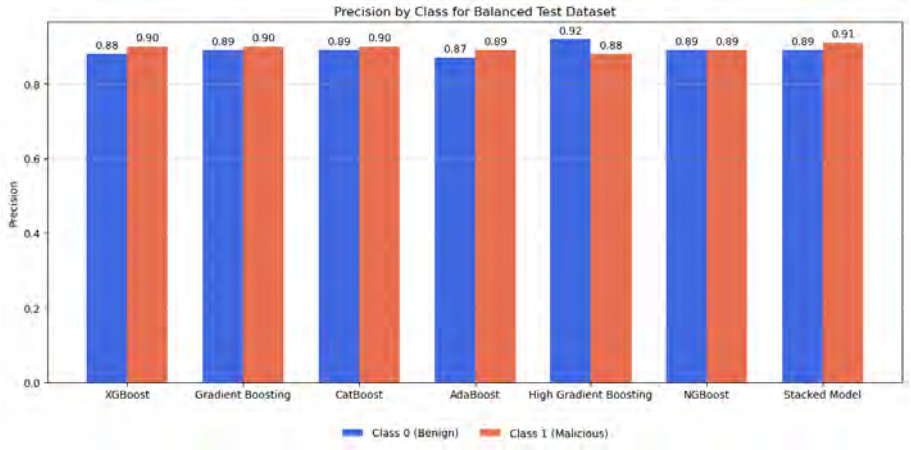

Figure 4.9: Precision by Class for Balanced Test Dataset.

Table 4.3: 4.4.2 Imbalanced Dataset: Model Performance Comparison

Model	Prediction Time (s)	Accuracy	Precision	Recall	F1-Score	AUC-ROC	PR-AUC
XGBoost	0.019	0.860	0.815	0.943	0.874	0.872	0.817
Gradient Boosting	0.031	0.865	0.820	0.947	0.879	0.875	0.801
CatBoost	0.044	0.863	0.821	0.945	0.878	0.874	0.824
AdaBoost	0.026	0.853	0.811	0.936	0.869	0.862	0.827
High Gradient Boosting	0.036	**0.868**	**0.825**	**0.952**	**0.884**	**0.878**	0.804
NGBoost	0.067	0.861	0.818	0.944	0.877	0.870	0.746
Stacked (CatBoost + High GB)	0.062	0.867	0.823	0.949	0.882	0.877	**0.853**

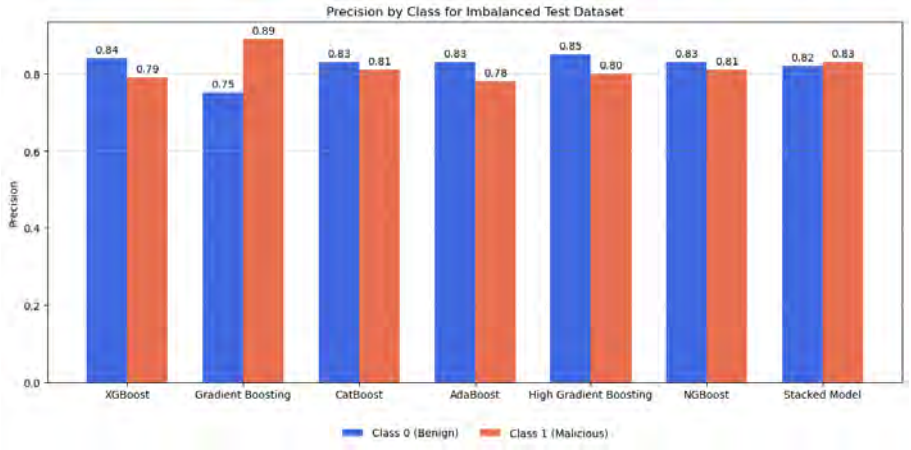

Figure 4.10: Precision by Class for Balanced Test Dataset.

Insights:

- The stacked model was a very close runner-up to this and registered the maximum PR-AUC value (0.853), indicating it as a better model for imbalanced cases where precision-recall balance is more important.
- From both the tables, we can see a significant underperformance of AdaBoost, whose scores are the lowest in every metric.

Chapter 5

Conclusion

5.1 key Findings

This work investigated the utilization of different types of Boost-family machine learning models to classify malicious URLs in different contexts like web, social, QR code, and custom links. Five datasets were used in the study, three of which were used for training and the other two for testing. Using a thorough approach of tedious preprocessing, feature engineering, model testing, and interpretability, we created a solid detection framework.
Here are the top takeaways from this research:

- Boost-family models performed especially better than the other models in recognizing malicious links with high recall and F1-scores, namely, High Gradient Boosting and CatBoost.
- To further improve the generalization performance, a stacked ensemble model combining CatBoost and High Gradient Boosting was used, preserving the good evaluation results.
- Model interpretability with LIME confirmed that critical features such as has_https, has_suspicious_keyword, and num_slashes had a substantial impact on model decisions, balancing between higher accuracy and interpretability.
- Unlike the real-world security environment where the imbalance is common, the model was strong on both balanced and imbalanced test datasets, meaning it is applicable to areas with security environments.

5.2 Limitations

The results of this research are encouraging, but there are some limitations:

- This was done using the models solely belonging to the Boost family. While this was effective, an expanded set of models, including deep learning methods, might be useful to draw further conclusions.
- The model tested on publicly available datasets from Kaggle may not entirely reflect the complexity and diversification of real-time enterprise network scenarios.

- Compared to the individual models like XGBoost or AdaBoost, we know that the stacked models took a long time to train, which may be a bottleneck in low-latency or resource-constrained deployments.
- Due to computational limitations, we did not include any content-based and contextual features and instead used a feature set based on URL structure and metadata.

5.3 Future Work

Several avenues for future research are outlined based on the present work:

- Combining deep learning architectures (CNNs, RNNs, Transformers) to learn more sophisticated malicious patterns and behaviors in URLs.
- Deployment of real-time detection systems monitoring the traffic on URLs and raising alerts based on the trained models.
- Increasing the diversity of the datasets either by mixing with the datasets from the enterprise level or generating new malicious URLs (with good crawling depth) from the dark level and phishing forum.
- Adding content, behavioral, and network-level features to increase detection.
- Using state-of-the-art interpretability techniques with decision tree-based models for a joint-view explanation of model behavior.
- Use the final stacked model, deploy it into the internal organizational communication platform to monitor the links exchanged, and alert any potential harmful URLs in order to build up the internal cybersecurity attack.

Bibliography

[1] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[2] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[3] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[4] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.

[6] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Systems with Applications*, vol. 106, pp. 1–20, 2018.

[7] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: Unbiased boosting with categorical features," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[8] D. Sahoo, C. Liu, and S. C. Hoi, "Malicious url detection using machine learning: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–35, 2019.

[9] T. Duan, A. Avati, D. Ding, *et al.*, "Ngboost: Natural gradient boosting for probabilistic prediction," in *International Conference on Machine Learning (ICML)*, 2020.

[10] R. Kumar and P. Kumar, "Malicious url detection using machine learning: A systematic review," *International Journal of Information Management Data Insights*, vol. 1, no. 1, p. 100 003, 2020.

[11] *Phishing site urls*, https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls, Retrieved January 11, 2025, from https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls, 2020.

[12] M. Aljabri, H. S. Altamimi, S. A. Albelali, *et al.*, "Detecting malicious urls using machine learning techniques: Review and research directions," *IEEE Access*, vol. 10, pp. 121 395–121 412, 2022. DOI: 10.1109/ACCESS.2022.3222307. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3222307.

[13] T. Tabassum, M. M. Alam, M. S. Ejaz, and M. K. Hasan, "A review on malicious urls detection using machine learning methods," *Journal of Engineering Research and Reports*, vol. 25, no. 12, pp. 76–88, 2023, Open Access under CC BY 4.0. DOI: 10.9734/jerr/2023/v25i121042. [Online]. Available: https://www.researchgate.net/publication/376707228.

[14] *A comprehensive dataset for malicious attacks*, https://www.kaggle.com/datasets/amanverma1999/a-comprehensive-dataset-for-malicious-attacks, Retrieved January 11, 2025, from https://www.kaggle.com/datasets/amanverma1999/a-comprehensive-dataset-for-malicious-attacks, 2024.

[15] *Kaggle phishing dataset*, https://www.kaggle.com/datasets/nicklz/kaggle-phishing-dataset?select=processed_PhishDataset.csv, Retrieved January 11, 2025, from https://www.kaggle.com/datasets/nicklz/kaggle-phishing-dataset?select=processed_PhishDataset.csv, 2024.

[16] *Phishing_dataset*, https://www.kaggle.com/datasets/ruckdent/phishing-dataset?select=malicious_phish.csv, Retrieved January 11, 2025, from https://www.kaggle.com/datasets/ruckdent/phishing-dataset?select=malicious_phish.csv, 2024.

[17] *Phishing_url*, https://www.kaggle.com/datasets/nompamh/phishing-url?resource=download, Retrieved January 11, 2025, from https://www.kaggle.com/datasets/nompamh/phishing-url?resource=download, 2024.