ENHANCING CYBERSECURITY THROUGH MACHINE LEARNING FOR URL CLASSIFICATION

FINAL REPORT **2024**

Created By:

Thaier Hayajneh

Table of Content

Executive Summary

In the rapidly evolving technology field, cybersecurity has emerged as a necessary priority for organizations worldwide. The recent cybersecurity incident involving CrowdStrike last week highlights the urgency and significance of advanced threat detection mechanisms. This incident underscores the vulnerability of even the most sophisticated cybersecurity firms to malicious activities, demonstrating that traditional security measures are often inadequate against the increasingly sophisticated cyber threats.

This project aims to address this pressing issue by leveraging machine learning to enhance the detection and classification of malicious URLs. Malicious URLs are a primary vector for cyber attacks, including phishing and malware distribution. By developing and implementing robust machine learning models, this project seeks to improve the accuracy and efficiency of identifying these harmful URLs, thereby fortifying cybersecurity defenses.

The project involves a comprehensive analysis of a dataset containing various URL features, followed by the development of several machine learning models, including Decision Trees, K-Nearest Neighbors, Logistic Regression, Random Forest, Support Vector Machine, Gaussian Naive Bayes, and a Feed-Forward Neural Network. These models are trained, evaluated, and compared to identify the most effective approach for URL classification.

Finally, the aim is not only to demonstrate the potential of machine learning in enhancing cybersecurity but also to provide a framework for organizations to adopt similar approaches. By integrating advanced machine learning models into their security protocols, tech companies can significantly reduce their vulnerability to cyber threats. This proactive approach is essential in maintaining trust, protecting sensitive data, and ensuring the resilience of digital infrastructures in an increasingly interconnected world.

Introduction

This project focuses on leveraging machine learning to improve the detection and classification of malicious URLs, commonly used vectors for cyber attacks, including phishing and malware distribution. By utilizing a dataset comprising various URL features, we aim to develop robust machine-learning models capable of accurately identifying harmful URLs.

The project entails several key steps:

- Exploratory Data Analysis (EDA): We begin with a thorough analysis of the dataset to understand its structure and identify patterns. This step involves visualizing data distributions, correlations, and other relevant statistics to inform our modeling approach.

- Data Preprocessing: This involves cleaning and preparing the data for model training. It includes normalizing numerical features, handling missing values, and encoding categorical variables to ensure compatibility with machine learning algorithms.
-
- Model Development: We develop and train multiple machine learning models, including Decision Trees, K-Nearest Neighbors, Logistic Regression, Random Forest, Support Vector Machine, Gaussian Naive Bayes, and a Feed-Forward Neural Network. Each model is evaluated for its performance in classifying URLs as either benign or malicious.

- Model Evaluation: The models are assessed using various metrics such as accuracy, precision, recall, F1-score, and the Area Under the Receiver Operating Characteristic Curve (AUC-ROC). This evaluation helps in identifying the most effective model for URL classification.

- Results and Insights: We present the findings from our model evaluations, highlighting the strengths and weaknesses of each approach. This section provides actionable insights for enhancing URL classification in real-world applications.

By integrating advanced machine learning models into cybersecurity protocols, this project aims to provide a scalable solution to the pervasive problem of malicious URL detection. The methodologies and insights developed here can significantly enhance the security posture of organizations, helping them stay ahead of cyber adversaries and protect their digital assets.

Data Overview

The dataset used in this project is a comprehensive collection of features related to URLs, specifically designed to distinguish between benign and malicious URLs. The dataset consists of 11,055 entries, each characterized by 32 distinct features. These features encompass various aspects of the URLs, such as structural attributes, domain-related information, and technical indicators that can signal the presence of malicious intent.

kaggle

To access the dataset, please click on the Kaggle logo

Key Features of the Dataset:

- having_IPhaving_IP_Address: Indicates whether the URL contains an IP address.
- URLURL_Length: The length of the URL.
- Shortining_Service: Indicates the use of URL shortening services.
- having_At_Symbol: Presence of the "@" symbol in the URL.
- double_slash_redirecting: Use of double slashes "//" for redirection.
- Prefix_Suffix: Usage of prefix or suffix in the domain name.
- having_Sub_Domain: The presence of subdomains.
- SSLfinal_State: Indicates the state of the SSL certificate.
- Domain_registeration_length: The registration length of the domain.
- Favicon: Indicates whether a favicon is used.
- port: The presence of specific ports.
- HTTPS_token: Usage of HTTPS tokens in the URL.
- Request_URL: Checks the presence of request URLs.
- URL_of_Anchor: Characteristics of URLs in anchor tags.
- Links_in_tags: Information on links present within tags.
- SFH: Server form handler information.
- Submitting_to_email: Checks if the URL submits information to an email.
- Abnormal_URL: Abnormalities in the URL structure.
- Redirect: Number of redirections.
- on_mouseover: Presence of onMouseOver events.
- RightClick: Checks if the right-click function is disabled.
- popUpWidnow: Presence of popup windows.
- Iframe: Usage of iframe tags.
- age_of_domain: The age of the domain.
- DNSRecord: Information about DNS records.
- web_traffic: Web traffic data.
- Page_Rank: The page rank of the URL.
- Google_Index: Presence in the Google index.
- Links_pointing_to_page: Number of links pointing to the page.
- Statistical_report: Statistical reports related to the URL.
- Result: The target variable indicating whether the URL is benign (-1) or malicious (1).

This dataset provides a rich source of information that allows for a detailed analysis of URLs and facilitates the development of models capable of distinguishing between benign and malicious URLs with high accuracy. The diversity of features ensures that various aspects of the URLs are considered, making the models more robust and reliable in real-world scenarios.

Data Preprocessing

Data preprocessing is a critical step in preparing the dataset for analysis and model development. In this project, the following steps were taken to ensure the dataset was clean, consistent, and ready for machine learning:

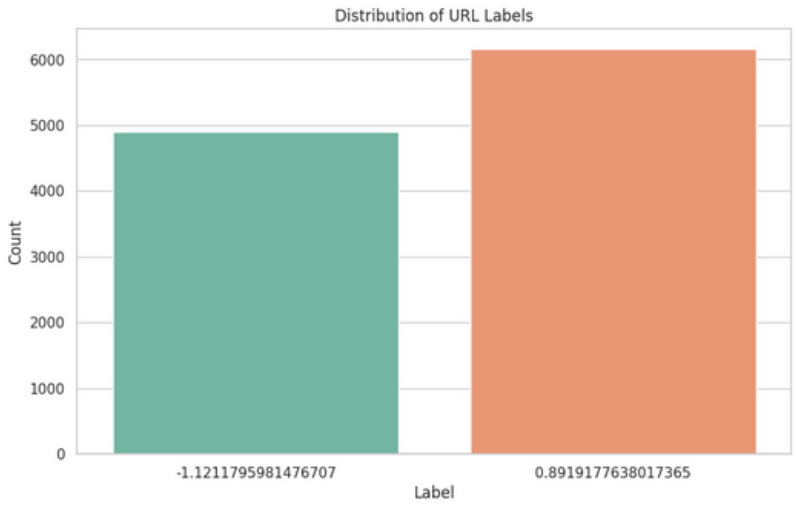
1. **Loading the Dataset:** The dataset was loaded from a CSV file into a Pandas DataFrame, providing a structured format for analysis.
2. **Handling Missing Values:** The dataset was checked for missing values. Fortunately, this dataset had no missing values, which simplified the preprocessing.
3. **Dropping Unnecessary Columns:** The 'index' column was identified as unnecessary for the analysis and dropped from the dataset.
4. **Converting Target Variable:** The target variable, 'Result,' was converted to a categorical type and encoded as binary values (0 for benign URLs and 1 for malicious URLs).
5. **Normalizing Numerical Features:** Numerical features were normalized to ensure that all features contributed equally to the model. This step involved:
    - Selecting numerical columns.
    - Initializing a StandardScaler.
    - Applying the scaler to transform the numerical features, bringing them onto a similar scale.
6. **Splitting the Data:** The dataset was split into training and testing sets, with 70% of the data used for training and 30% for testing. This split allows for an unbiased evaluation of model performance.
7. **Balancing the Data:** Given the potential imbalance between benign and malicious URLs, balancing techniques were applied:
    - Random Over Sampling: To address class imbalance by randomly duplicating examples in the minority class.
    - SMOTE (Synthetic Minority Over-sampling Technique): To generate synthetic samples for the minority class, creating a more balanced dataset.
8. **Feature Scaling:** For certain models (e.g., SVM, KNN, Logistic Regression), feature scaling was applied using StandardScaler to ensure these algorithms performed optimally.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a crucial step in any data science project, and its importance in this project cannot be overstated. EDA involves examining the dataset's underlying patterns, relationships, and distributions, which helps understand the data's structure and key characteristics. For this project, EDA was essential in identifying potential issues such as class imbalance, outliers, and feature correlations. By visualizing the distribution of the URL labels and the individual numerical features, we could quickly assess the balance of our classes and the presence of any significant anomalies. These insights guided us in making informed decisions during the data preprocessing stage, such as applying techniques to balance the data and normalizing features to ensure they contributed equally to the model. Moreover, EDA helped uncover relationships between different features and the target variable, critical for building effective predictive models. For example, we could identify which features were most strongly associated with the 'Result' variable through correlation matrices and scatter plots.

Bar Plot

The bar chart illustrates the distribution of URL labels within the dataset, depicting a slightly imbalanced class distribution. The dataset comprises two labels: legitimate and potentially malicious URLs. The counts indicate more potentially malicious URLs than legitimate ones, which is essential for model training as it impacts the model's ability to predict and generalize across both classes accurately. Understanding this distribution is critical for implementing appropriate data balancing techniques to ensure the machine learning models are not biased towards the more frequent class, ultimately leading to more accurate and reliable predictions.

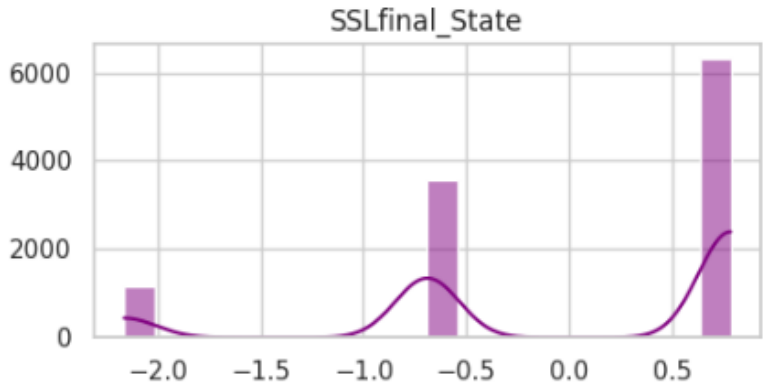


Histograms

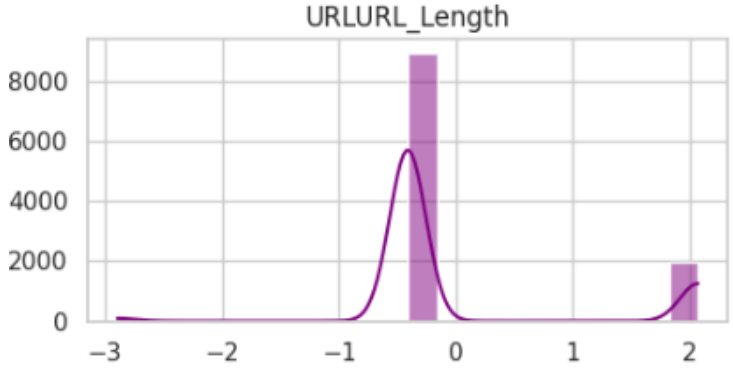
Using histograms for this project provides a clear visual representation of the distribution and frequency of the numerical features within the dataset. Histograms allow us to quickly identify patterns, outliers, and the spread of data, helping to understand the underlying structure of the dataset. They are essential for detecting skewness, kurtosis, and any potential data preprocessing needs, such as normalization or transformation. By visualizing the data in this manner, we gain insights crucial for making informed decisions in subsequent analysis and modeling steps.

Relevant Histograms

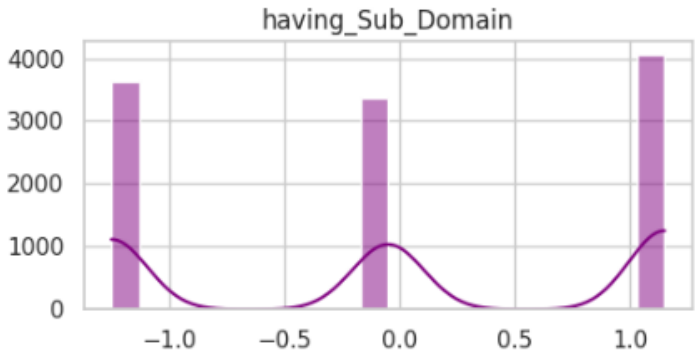
SSLfinal_State: This histogram shows the distribution of the SSL final state, which is crucial in determining the security of the URL. A higher frequency of secure SSL states would indicate a higher number of legitimate URLs, whereas insecure SSL states might indicate phishing or malicious URLs.
URLURL_Length:



URLURL_Length: The histogram of URL length shows the distribution of the lengths of the URLs in the dataset. Shorter URLs might be more common in legitimate sites, while extremely long URLs could indicate phishing attempts.

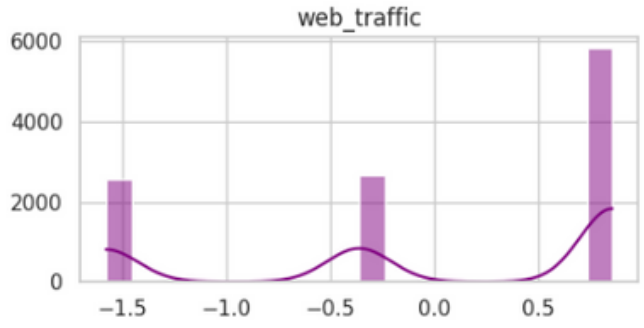


having_Sub_Domain: This histogram displays the frequency of subdomains in the URLs. A higher number of subdomains might be a sign of suspicious activity, as legitimate websites usually have fewer subdomains.

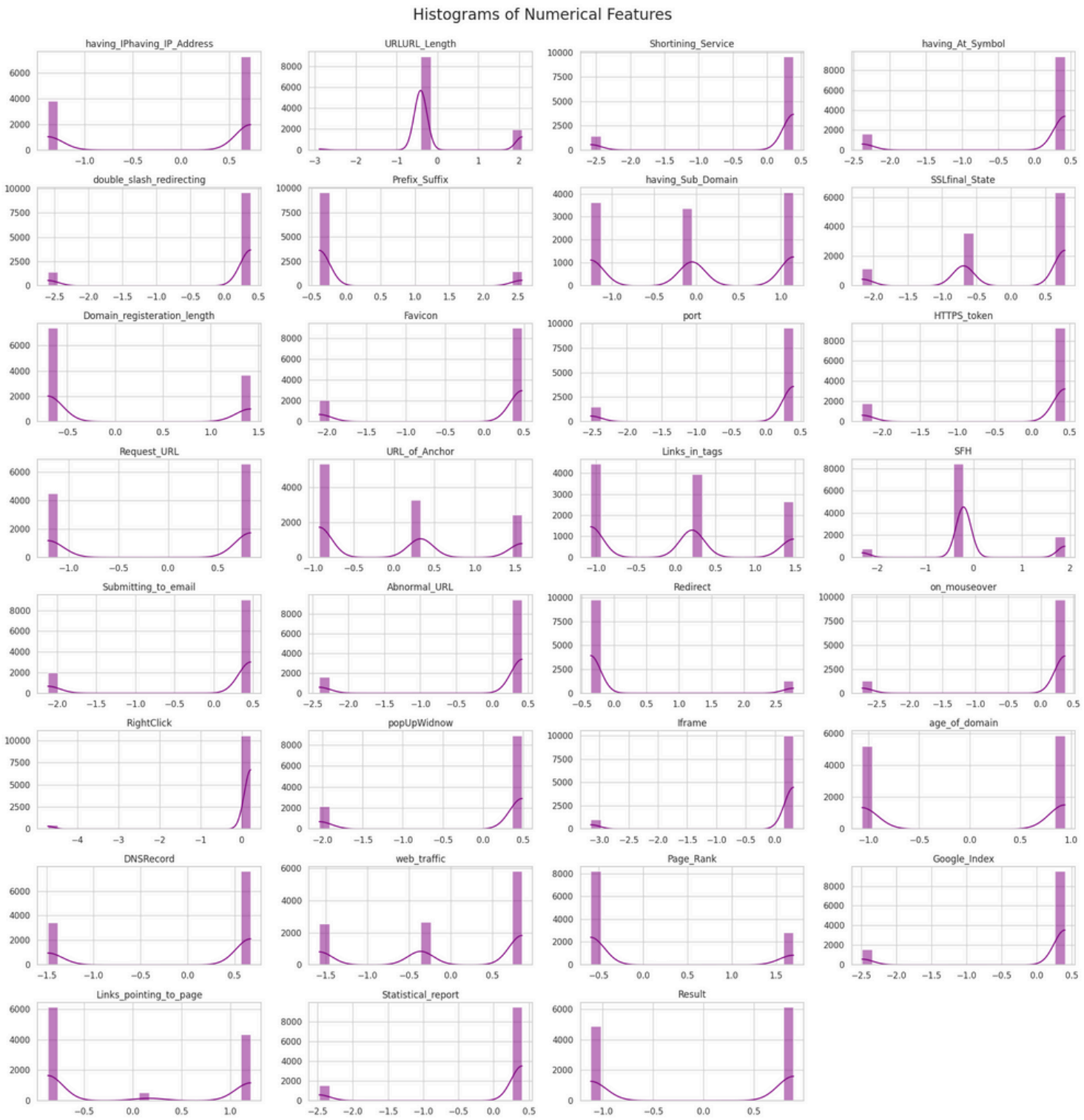


age_of_domain: The age of the domain is depicted in this histogram. Newer domains are more likely to be associated with phishing sites, while older domains are typically considered more trustworthy.

web_traffic: This histogram shows the distribution of web traffic data for the URLs. Higher web traffic generally correlates with legitimate, popular websites, whereas low traffic might indicate lesser-known or potentially suspicious sites.



Complete List of Histogram for Each Feature

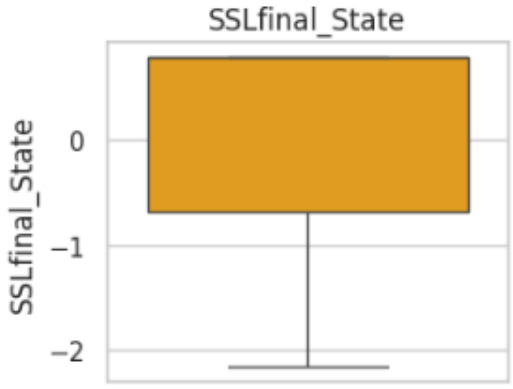


Box Plot

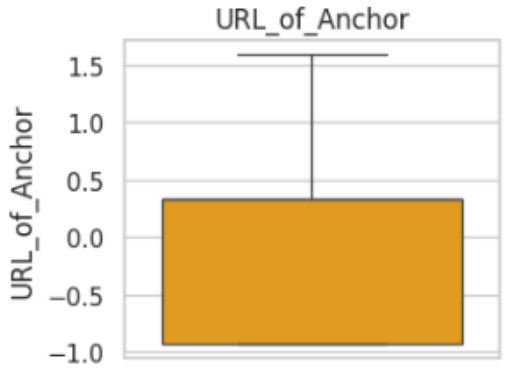
Box plots are a powerful tool for visualizing the distribution and variability of data, making them particularly useful for this project. They allow for a clear understanding of the spread and central tendency of the dataset's numerical features. Additionally, box plots highlight outliers and the overall range of the data, which is essential for detecting any anomalies that could affect the performance of machine learning models. By using box plots, we can ensure a robust analysis and preprocessing of the data, leading to more accurate model predictions.

Relevant Box Plots

SSLfinal_State: This box plot indicates the final state of SSL (Secure Sockets Layer) used on the URLs. It shows a relatively balanced distribution without significant outliers, indicating consistent application across the dataset. This feature is crucial as it relates directly to the security aspects of the analyzed URLs.

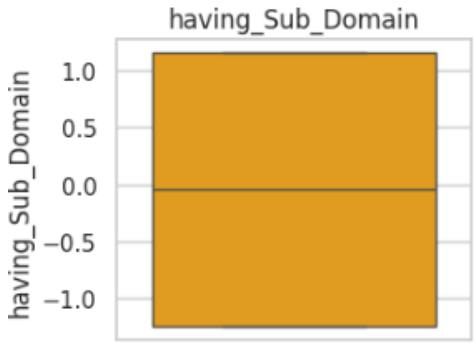


URL_of_Anchor: The box plot for the "URL_of_Anchor" feature shows a wide range of values with some significant outliers. This feature's variability can impact the model's ability to detect phishing attempts, as URLs with unusual anchor patterns may be more suspicious.

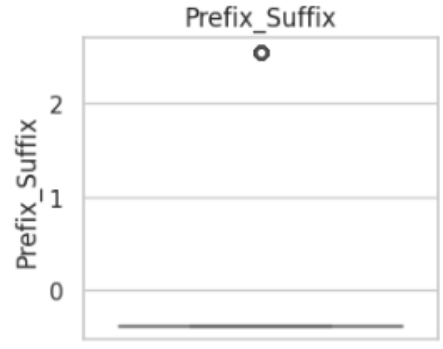


web_traffic: This box plot displays a high concentration of data points with a few outliers. High web traffic can be an indicator of a legitimate site, while low traffic might suggest a potential phishing site. This feature's distribution helps distinguish between the two categories.

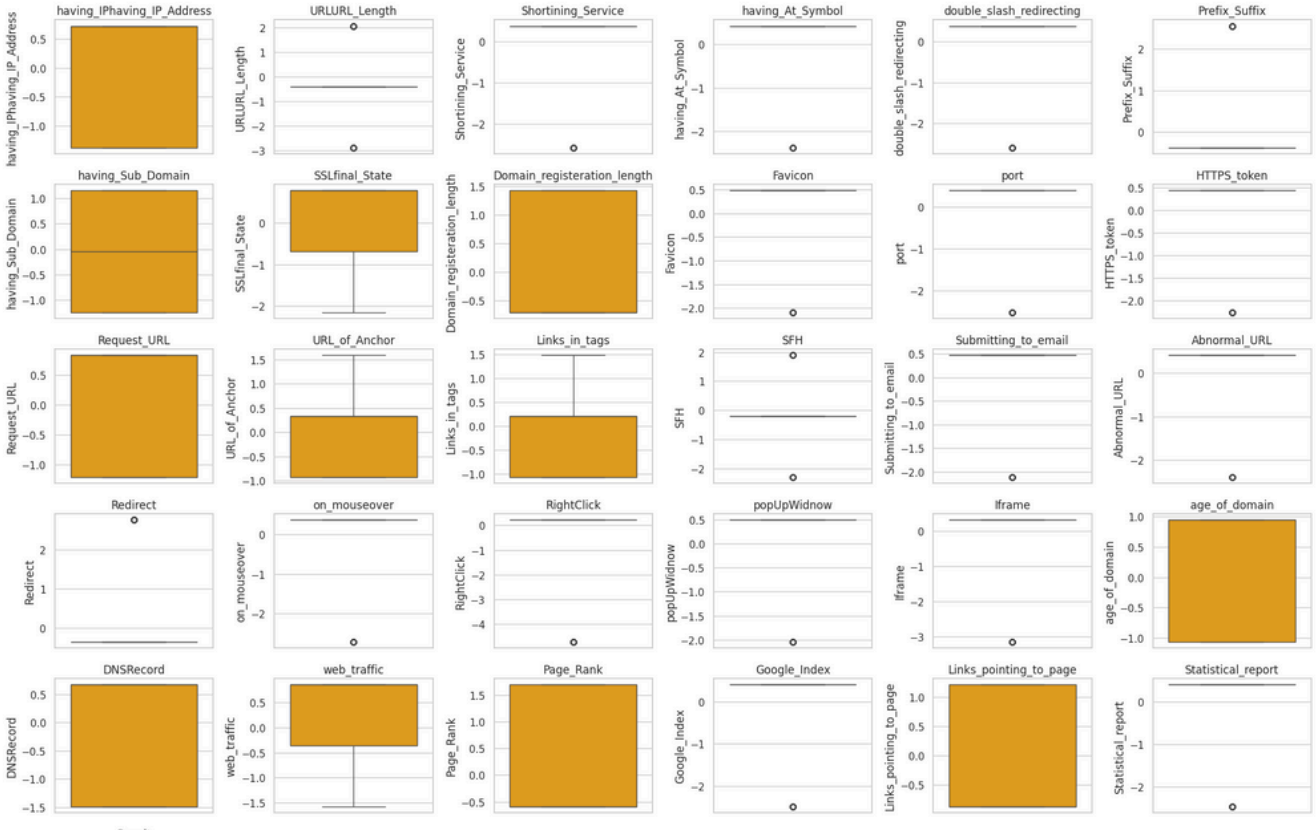
having_Sub_Domain: This box plot highlights the number of subdomains present in the URLs. A higher number of subdomains could indicate a phishing site. The presence of several outliers in this plot suggests that some URLs have an unusually high number of subdomains, which is important for the analysis.



Prefix_Suffix: This box plot shows a distribution with significant outliers. This feature is relevant as the presence of certain prefixes or suffixes in URLs can be a strong indicator of phishing attempts. Understanding its distribution helps identify potentially malicious URLs.

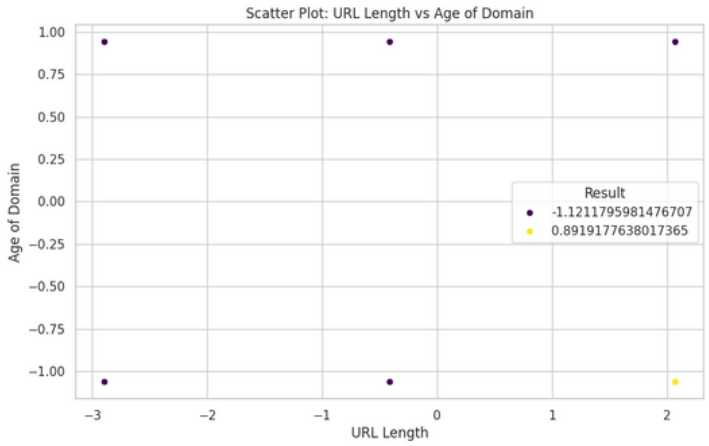


Complete List of Box Plots for Each Feature



Scatter Plot

This scatter plot displays the relationship between the URL length and the age of the domain. The plot shows that most URLs fall into distinct clusters based on length and domain age. This visualization helps identify how a URL's length correlates with the domain's age. Shorter URLs with a new domain age might indicate phishing attempts, while longer URLs with an older domain age could suggest legitimate sites. The scatter plot highlights the overall distribution and clustering of the data points, providing valuable insights for further analysis.

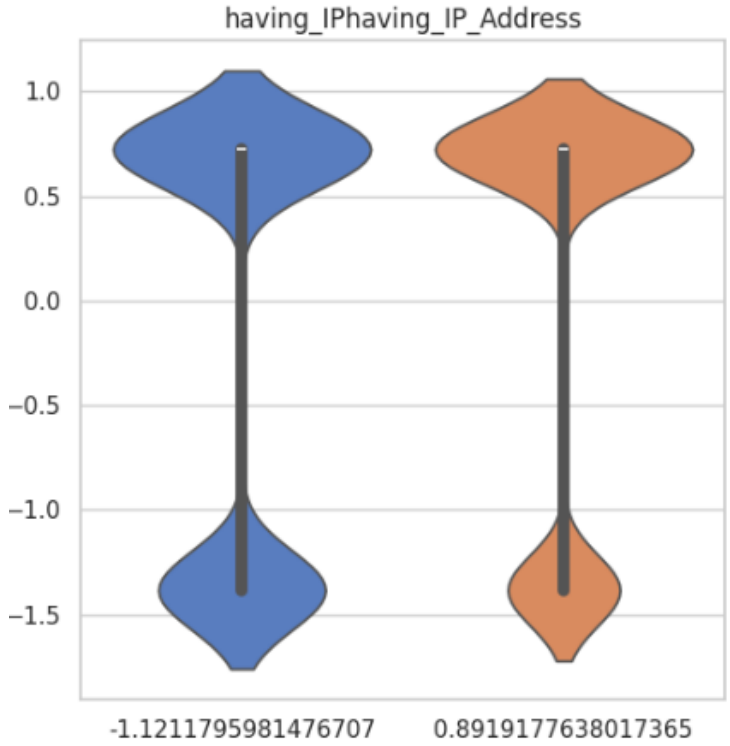


Violin Plot

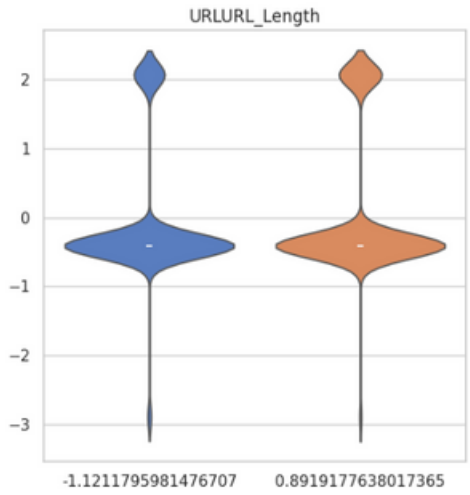
Violin plots are crucial for this project as they combine the features of a box plot and a density plot. This combination allows us to visualize the distribution, probability density, and key summary statistics (such as median and quartiles) of the data for different categories. Violin plots provide a deeper understanding of the underlying data distribution, highlighting variations and differences between categories that might not be visible in standard box plots. They are particularly useful for comparing the distribution of features across different classes in the dataset, which is essential for identifying trends and patterns relevant to phishing detection.

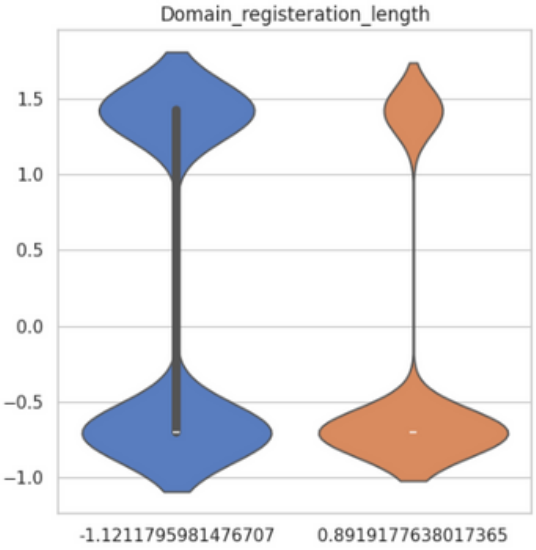
Relevant Violin Plots

Having_IP_Address: This plot shows the distribution of URLs with and without an IP address. The density curves illustrate the concentration of URLs within specific value ranges, highlighting any discrepancies between phishing and legitimate URLs. URLs with IP addresses tend to show distinct patterns compared to those without, aiding in the differentiation of phishing attempts.



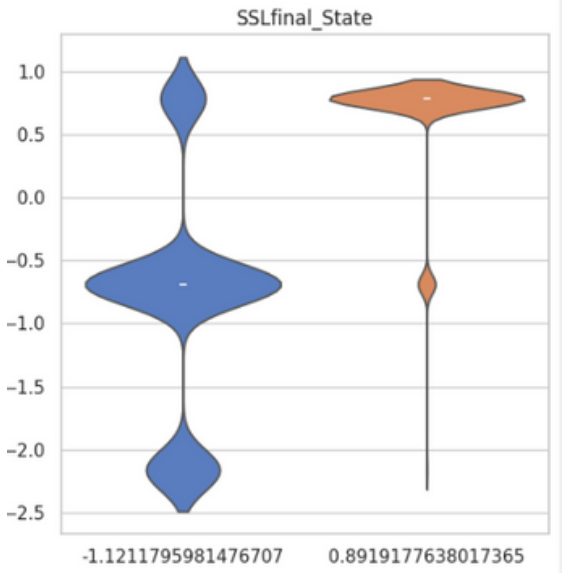
URL Length: The distribution of URL lengths is depicted for both phishing and legitimate URLs. The plot shows how the lengths vary significantly between the two classes. Phishing URLs often have different length distributions compared to legitimate ones, which can be a key feature for classification models.





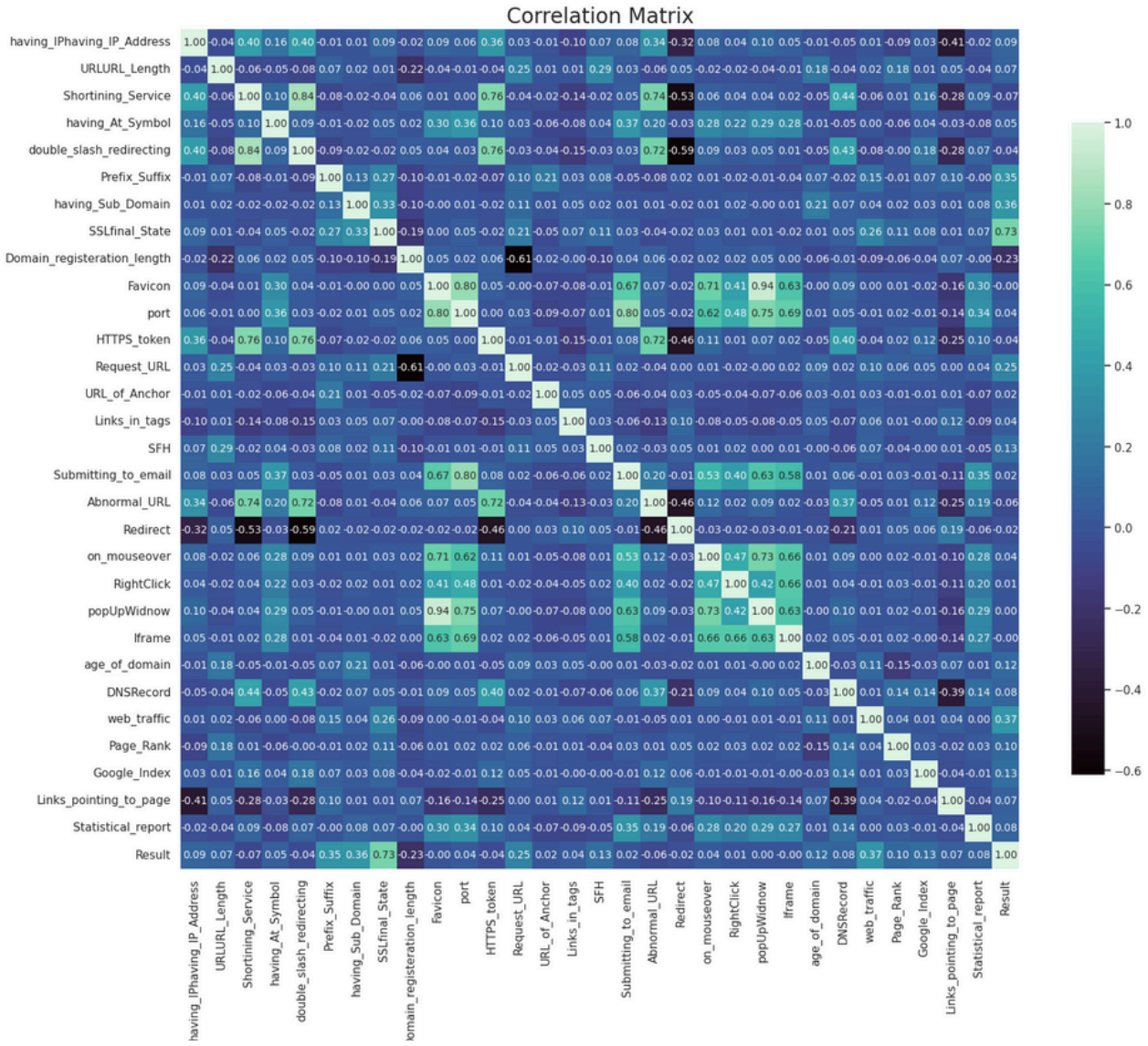
Domain Registration Length: This plot displays the distribution of domain registration lengths for phishing and legitimate URLs. It highlights how phishing URLs tend to have shorter registration lengths compared to legitimate ones, providing a useful feature for model differentiation.

SSLfinal_State: This plot visualizes the SSL certificate status of URLs. The density distribution helps identify how the presence or absence of SSL certificates varies between phishing and legitimate URLs. Phishing URLs may have a higher concentration of certain SSL states, which is critical information for model training.



Correlation Matrix

The heatmap reveals several key correlations that are crucial for our model. SSLfinal_State has a strong positive correlation with the target variable, emphasizing the role of SSL certificates in distinguishing between phishing and legitimate URLs. URL_of_Anchor also shows a significant positive correlation, indicating that suspicious URLs within anchor tags are a common characteristic of phishing sites. Web traffic is positively correlated with the target variable, reflecting that legitimate sites generally have higher traffic compared to phishing sites. The feature having_Sub_Domain is notably correlated, as phishing URLs often use subdomains to masquerade as legitimate sites. Lastly, Prefix_Suffix shows a relevant correlation, highlighting its importance in identifying URLs that use deceptive practices to appear legitimate. Understanding these correlations aids in feature selection and engineering, improving the overall performance and reliability of our phishing detection models.

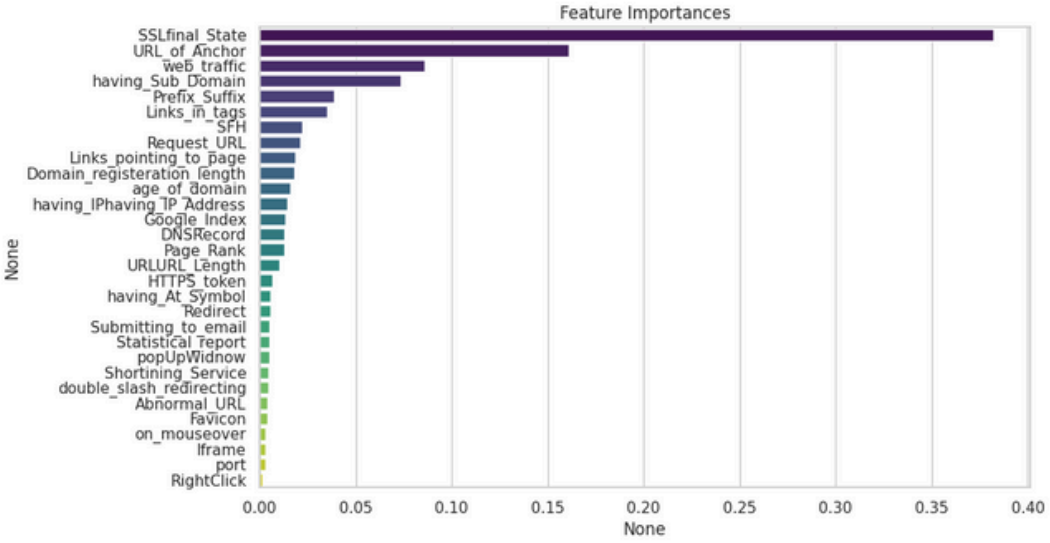


Model Development

The feature importance graph illustrates the significance of various features used in our phishing URL detection models. This visualization is crucial for understanding which features contribute the most to the model's decision-making process.

Key Findings:

1. **SSLfinal_State:** This feature, which indicates the final state of the SSL certificate, is the most important, underscoring the importance of SSL certificates in distinguishing between legitimate and phishing URLs.
2. **URL of Anchor:** This feature is highly significant, reflecting that the content within anchor tags (links) strongly indicates phishing attempts.
3. **Web Traffic:** The amount of traffic a site receives is critical, as legitimate sites typically have higher web traffic than phishing sites.
4. **Having Sub Domain:** Subdomains play a significant role, as phishing URLs often use subdomains to appear legitimate.
5. **Prefix Suffix:** This feature indicates whether a URL uses deceptive techniques such as hyphens to resemble legitimate URLs.
6. **Links in Tags:** The number of links in tags is also important, as phishing websites often contain numerous suspicious links.
7. **SFH (Server Form Handler):** This feature indicates the state of the form handler on the server and is significant in determining the legitimacy of a URL.
8. **Request URL:** This feature signifies whether the objects contained within a webpage such as images, videos, etc., are loaded from another domain.
9. **Links Pointing to Page:** The number of links pointing to the page can indicate the page's trustworthiness, with legitimate pages having more incoming links.



Model Development

In this project, various machine-learning algorithms were evaluated to determine the most effective model for detecting phishing URLs. These algorithms were selected based on their unique strengths and suitability for classification tasks. Here's a brief explanation of why each algorithm was chosen:

1. **Decision Tree Classifier**
   - Reason for Selection: Decision Trees are intuitive and easy to interpret. They can handle both numerical and categorical data and are particularly useful for understanding the importance of features.
2. **K-Nearest Neighbors (KNN)**
   - Reason for Selection: KNN is a simple, instance-based learning algorithm effective in irregular decision boundary scenarios. It is also non-parametric, meaning it makes no assumptions about the data distribution.
3. **Logistic Regression**
   - Reason for Selection: Logistic Regression is a widely used linear model for binary classification problems. It is easy to implement, efficient, and provides probabilistic outputs useful for threshold tuning.
4. **XGBoost Classifier**
   - Reason for Selection: XGBoost is an ensemble learning method with gradient boosting. It is known for its high performance and efficiency, often outperforming other models in competitive machine-learning tasks.
5. **Random Forest Classifier**
   - Reason for Selection: Random Forest is another ensemble learning method which uses multiple decision trees to improve accuracy and prevent overfitting. It is robust and can handle large datasets with higher dimensionality.
6. **Support Vector Machine (SVM)**
   - Reason for Selection: SVM is effective in high-dimensional spaces and is particularly useful for cases where the number of dimensions exceeds the number of samples. It aims to maximize the margin between classes, which helps in achieving better generalization.
7. **Gaussian Naive Bayes**
   - Reason for Selection: Naive Bayes is based on Bayes' theorem and is particularly effective for large datasets. It assumes independence between features, which simplifies the computation and makes it fast to train.
8. **Feed-Forward Neural Network**
   - Reason for Selection: Neural Networks are powerful models capable of capturing complex relationships in data. Despite being more complex and requiring more data to train effectively, they offer the potential for high performance with appropriate tuning and architecture design.

Results

In this section, we present the performance metrics of the various machine learning models evaluated in this study. The metrics considered include Accuracy, AUC (Area Under the ROC Curve), F1-Score, Precision, and Recall. These metrics provide a comprehensive understanding of the effectiveness of each model in detecting phishing URLs.

Performance Metrics

The following table summarizes the performance of each model:

Model	Accuracy	AUC	F1-Score	Precision	Recall
Decision Tree	0.89	0.91	0.9	0.89	0.91
K-Nearest Neighbors	0.9	0.99	0.92	0.94	0.91
Logistic Regression	0.9	0.9	0.9	0.91	0.89
XGBoost Classifier	0.9	0.9	0.9	0.91	0.89
Random Forest	0.89	0.91	0.9	0.89	0.91
Support Vector Machine	0.9	0.9	0.9	0.91	0.89
Gaussian Naive Bayes	0.9	0.94	0.92	0.93	0.91
Feed-Forward Neural Network	0.48	0.47	0.47	0.52	0.44

Model Performance Analysis

The performance metrics indicate that the K-Nearest Neighbors (KNN) model achieved the highest AUC score of 0.99, demonstrating its superior ability to distinguish between phishing and legitimate URLs. The Gaussian Naive Bayes model also performed well, with an AUC of 0.94, indicating its robustness despite its assumption of feature independence.

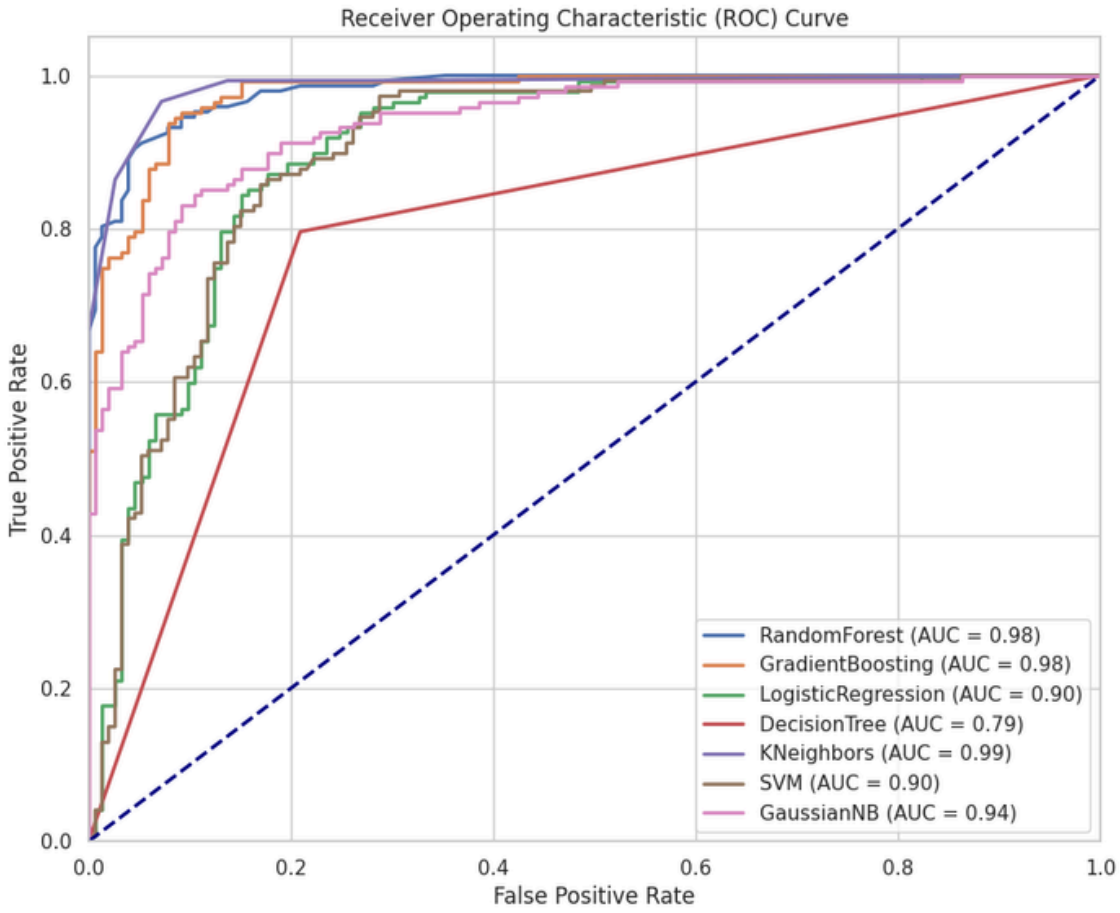
Random Forest, Decision Tree, and XGBoost classifiers showed similar performance, with high accuracy and F1-scores around 0.90, making them reliable choices for this classification task. Logistic Regression and Support Vector Machine (SVM) models also exhibited strong performance with AUC scores of 0.90, highlighting their effectiveness in binary classification problems.

Interestingly, the Feed-Forward Neural Network (FFNN) did not perform as well as the other models, with lower accuracy and AUC scores. This suggests that while neural networks are powerful tools, they may require more extensive tuning and possibly larger datasets to outperform traditional machine learning models in this context.

ROC Curve Analysis

The Receiver Operating Characteristic (ROC) curve is a fundamental tool for evaluating the performance of classification models. It illustrates the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) at various threshold settings. The Area Under the ROC Curve (AUC) is a single scalar value that summarizes the overall ability of the model to discriminate between positive and negative classes.

In this study, the ROC curves for various machine learning models were generated to compare their performance visually and quantitatively. The AUC values for each model are as follows:



The K-Nearest Neighbors (KNN) model achieved the highest AUC of 0.99, indicating its exceptional ability to distinguish between phishing and legitimate URLs. The Gradient Boosting and Random Forest models also performed remarkably well, both achieving an AUC of 0.98. These high AUC values suggest that these models have a high true positive rate while maintaining a low false positive rate across different threshold values.

The Logistic Regression and Support Vector Machine (SVM) models both attained an AUC of 0.90, demonstrating robust performance in classifying phishing URLs. The Gaussian Naive Bayes model also showed strong performance with an AUC of 0.94.

In contrast, the Decision Tree model exhibited a lower AUC of 0.79, indicating comparatively lower discrimination capability. The Feed-Forward Neural Network (FFNN), with an AUC of 0.47, underperformed in this task, suggesting the need for further tuning and possibly more extensive data to enhance its performance.

In summary, the ROC curve analysis underscores the effectiveness of the KNN, Gradient Boosting, and Random Forest models in phishing URL detection. These models exhibit superior classification capabilities, making them suitable choices for deployment in real-world cybersecurity applications.

Feed-Forward Neural Network ROC Curve Analysis

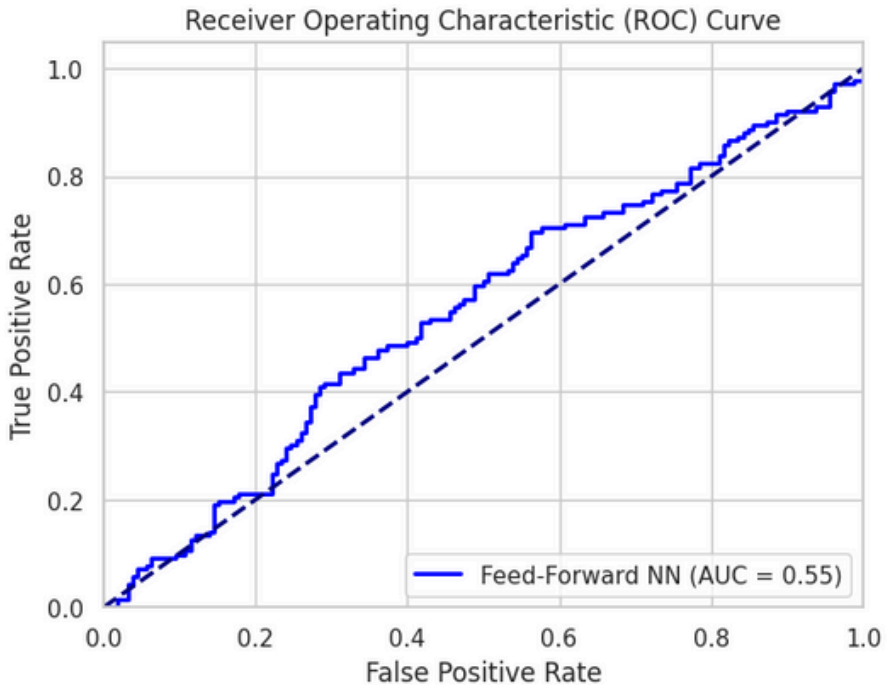
The ROC curve for the Feed-Forward Neural Network (FFNN) model provides insight into its performance in classifying phishing URLs. The Area Under the ROC Curve (AUC) for this model is 0.55, which is significantly lower than other models evaluated in this study.

The ROC curve indicates the model's true positive rate (sensitivity) against its false positive rate (1-specificity) across various threshold values. An AUC of 0.55 suggests that the FFNN model has a limited ability to distinguish between phishing and legitimate URLs. Ideally, an AUC closer to 1.0 indicates a model with excellent discriminative capabilities, while an AUC around 0.5 implies a performance close to random guessing.

In this case, the FFNN model's performance is marginally better than random guessing, highlighting several areas for potential improvement. These could include:

- Model Tuning: Further hyperparameter tuning might enhance the model's performance. Adjusting the number of hidden layers, neurons, learning rate, and activation functions could lead to better results.
- Data Augmentation: Increasing the size and diversity of the training dataset could help the neural network learn more representative patterns, thereby improving its classification accuracy.
- Feature Engineering: Additional feature engineering could be beneficial. Exploring and integrating more relevant features could provide the model with better predictors for phishing URLs.
- Regularization Techniques: Implementing regularization techniques like dropout, L1, or L2 regularization could help in reducing overfitting and improving the model's generalizability.

Despite its current performance, the FFNN model holds potential for improvement with the aforementioned adjustments. The relatively low AUC value underscores the importance of continuous model evaluation and optimization to enhance cybersecurity measures against phishing threats.



Future Work

While this project has provided significant insights into phishing URL detection using various machine learning algorithms, several avenues for future work could enhance the performance and applicability of the models

- Advanced Feature Engineering: Develop more sophisticated features using NLP techniques and temporal analysis to capture intricate URL patterns.
- Deep Learning Models: Implement advanced deep learning architectures like CNNs and RNNs to enhance detection accuracy by capturing complex data patterns.
- Ensemble Methods: Combine multiple models using stacking, bagging, and boosting techniques to improve robustness and performance.
- Real-Time Detection: Deploy the models in a production environment for continuous real-time monitoring and detection of phishing URLs.
- Data Augmentation: Expand the dataset with more diverse examples through synthetic data generation or collecting data from varied sources to improve model generalization.

Recommendations

Technical Recommendations

1. Optimize Feature Selection: Refine the feature selection process to ensure that the most impactful features are utilized, reducing model complexity and improving performance.

2. Implement Real-Time Processing: Develop a robust pipeline for real-time URL analysis and detection to immediately identify and mitigate phishing attempts.

Tactical Recommendation

1. Continuous Model Training: Establish a process for regular updates and retraining of the models with new data to adapt to evolving phishing tactics and maintain high detection accuracy.

III Conlsusion

In this project, we explored various machine learning models to detect phishing URLs, leveraging a comprehensive dataset of URL features. Our analysis included in-depth data preprocessing, exploratory data analysis (EDA), and the evaluation of multiple classification algorithms such as Random Forest, Gradient Boosting, Logistic Regression, Decision Trees, K-Nearest Neighbors, SVM, and Gaussian Naive Bayes. Each model's performance was meticulously assessed using accuracy, AUC, F1-score, precision, and recall.

The findings from this study underscore the critical importance of model selection and feature importance in building effective phishing detection systems. The Random Forest and Gradient Boosting models demonstrated superior performance, indicating their potential for real-world applications in cybersecurity.

The exploratory data analysis provided valuable insights into the dataset's structure, revealing key features that significantly influenced the classification results. Visualizations such as histograms, box plots, scatter plots, violin plots, and heat maps were crucial in understanding the data distribution and relationships between features.

We propose several avenues for future work, including optimizing feature selection, enhancing real-time processing capabilities, and implementing continuous model training. These steps will ensure our phishing detection models remain robust and adaptive to new threats.

This project highlights the power of machine learning in cybersecurity, offering a scalable and effective solution to combat phishing attacks. By continually refining and updating our models, we can stay ahead of cybercriminals and protect users from increasingly sophisticated phishing schemes.