Ibn Zohr University
Faculty of Sciences
Center of Excellence - Agadir

Master of Excellence

Data Analytics and Artificial Intelligence

Module: CYBER SECURITY

Project Report: Phishing URL Detection

Prepared by:

Hind ELQORACHI
Wafa JAAFAR
Asma MISBAH
Chaymae BELFAIK

Supervised by:

Pr.BOUGHROUS Monsef

Academic Year: 2025–2026

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 EXECUTIVE SUMMARY

Phishing attacks continue to rank among the most widespread and costly cybersecurity threats, with malicious URLs serving as a primary attack vector in credential theft, identity fraud, and financial exploitation. As phishing techniques evolve rapidly—leveraging URL obfuscation, domain spoofing, and dynamic redirection—traditional blacklist-based defenses and static rules are no longer sufficient. This growing complexity creates a clear need for intelligent, data-driven methodologies capable of generalizing beyond the threats we're familiar with.

This project's main objective is to develop an AI-powered phishing URL detection system by applying advanced data analytics techniques and machine learning models to a large corpus of labeled URLs. The work begins by acquiring a structured dataset enriched with comprehensive feature engineering, a crucial step in transforming raw URLs into numerical representations suitable for machine learning. Every URL is decomposed into multiple layers of information: structural patterns, lexical cues, domain metadata, and behavioral indicators.

From a data analytics perspective, feature exploration revealed strong correlations between phishing activity and characteristics such as URL length, redirection frequency, domain age, and HTTPS misconfigurations. Statistical summaries and visual analyses (such as distribution plots and heatmaps) provided early insight into how phishing URLs differentiate themselves from legitimate ones, guiding the refinement of the feature set and highlighting the features most likely to influence model learning.

Three supervised learning algorithms were selected for comparison: XGBoost, Random Forest, and Support Vector Classifier (SVC). Each model was chosen based on complementary strengths.

XGBoost, a gradient-boosting algorithm, is well-suited for handling heterogeneous features and capturing non-linear interactions.

Random Forest offers robustness to noise and strong generalization through ensemble averaging.

SVC provides a strong baseline, especially for high-dimensional feature vectors.

A rigorous data processing pipeline was implemented, including train–test splitting, normalization where relevant, and model calibration. To enhance reliability, probability-based predictions were analyzed, and a tuned probability threshold was introduced instead of relying on default 0.5 cutoffs. This allowed the classification boundary to be aligned with domain needs—for example, favoring higher recall when detecting potential phishing threats.

Evaluation metrics including accuracy, recall, precision, F1-score, ROC curves, and confusion matrices were used to assess model performance comprehensively. The ensemble models—XGBoost and Random Forest—outperformed the SVC classifier, demonstrating superior ability to detect subtle phishing patterns while minimizing false alarms. Feature importance analysis offered interpretability, revealing that domain age, URL depth, prefix–suffix patterns, and abnormal redirection behavior were among the strongest predictors.

To translate the model into an interactive tool, a Streamlit interface was developed, enabling real-time URL classification. The interface displays model predictions, phishing probability scores, and intuitive indicators reflecting the underlying analytics. This integration transforms the project from a theoretical study into a practical, operational system accessible to both technical and non-technical users.

In conclusion, this project demonstrates the effectiveness of combining data analytics, feature engineering, and machine learning for phishing URL detection. The findings confirm that a model-driven approach can capture behavioral signatures beyond human perception or rule-based filtering. The primary recommendation is to continuously enrich the dataset with recent phishing campaigns and periodically retrain the models, ensuring that learning remains aligned with evolving cyber-attack strategies. This work provides a solid foundation for an intelligent, scalable, and adaptive cybersecurity solution.

CHAPTER 2 PROBLEM STATEMENT & SCOPE

2.1 Context

Phishing attacks, particularly those based on malicious URLs, are among the most common and damaging cybersecurity threats. Attackers create URLs that closely mimic legitimate websites to deceive users into providing sensitive information such as passwords, credit card details, or corporate credentials. Studies show that over **90% of cyber-attacks start with phishing**, highlighting its pervasiveness and effectiveness.

2.2 Problem Definition

Despite awareness and existing security measures, users often cannot distinguish between legitimate and malicious URLs. Subtle manipulations—like replacing letters with numbers or using fake subdomains—make phishing URLs difficult to detect.

A notable example is the **Google Docs phishing attack of 2017**, where millions of users received emails with fake Google Docs links. Clicking the URL redirected users to a fraudulent OAuth page requesting Gmail access. Within minutes, compromised accounts propagated the malicious link to new victims, demonstrating the rapid spread and efficiency of such attacks.

2.3 Real-World Impact

Phishing URLs can cause significant consequences, including:

- **Data Breaches and Account Compromise:** Unauthorized access to email, cloud services, and corporate accounts.
- **Financial Losses:** Fraudulent transactions, invoice scams, unauthorized purchases, and ransomware incidents, leading to billions in global losses.

- **Identity Theft:** Personal information can be exploited to impersonate victims and open fraudulent accounts.
- **Reputational Damage:** Organizations lose customer trust and may face legal liabilities or negative publicity.

2.4 Scope of the Project

This project focuses on developing a **Phishing URL Detection System** with the following scope:

- **Dataset Preparation:** Using labeled phishing and legitimate URLs for training and evaluation.
- **Feature Extraction:** Analyzing lexical, structural, and domain-related characteristics of URLs.
- **Machine Learning Classification:** Developing models to distinguish phishing from legitimate URLs.
- **Model Evaluation:** Assessing accuracy, precision, recall, and F1-score.
- **Interactive Interface:** Allowing users to input URLs and receive immediate phishing detection results.
- **Limitations:** Focused solely on URL-based phishing, excluding SMS, voice, or malware-based attacks.

2.5 Summary

Phishing URLs are a widespread cybersecurity threat with serious financial, privacy, and reputational consequences. The Google Docs 2017 attack exemplifies how quickly malicious URLs can spread and compromise users. This project aims to mitigate these risks through an intelligent, machine learning–based detection system with an interactive interface for real-time URL analysis.

CHAPTER 3

TECHNICAL APPROACH & METHODOLOGY

This chapter summarizes the methodological pipeline followed throughout the project. The complete machine learning workflow is illustrated in Figure 3.1.

3.1 Pipeline Overview

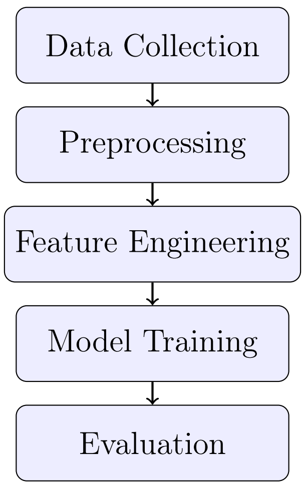

Figure 3.1: Machine Learning Pipeline for Phishing Detection

3.2 Dataset Description

The dataset employed in this project originates from a publicly available Kaggle repository and contains a substantial corpus of more than 800,000 URLs. This large-scale collection is intentionally curated to capture a wide diversity of online domains, ranging from highly trusted websites to malicious sources commonly involved in phishing campaigns. Approximately 52% of the URLs are labeled as legitimate, representing well-established and verifiable online services, whereas about 47% are classified as phishing, reflecting deceptive domains engineered to mimic trusted entities or exploit vulnerabilities in user behavior. Such a balanced distribution is particularly advantageous for supervised machine learning, as it mitigates the risks associated with class imbalance and enables fairer, more stable model learning.

Structurally, the dataset contains two primary fields: `url` and `status`. The `url` field stores the complete hyperlink string for each entry, serving as the raw input for feature extraction. The `status` field provides the corresponding binary label, where a value of 0 denotes a phishing URL and a value of 1 denotes a legitimate URL.

	url	status
167377	https://gd2qq8.duckdns.org/	0
136893	https://www.amzaon.co.ip.maesome.shop/a3lwMDY1P3	0
128254	https://www1.aupay.co.aumin-php.com	0
671400	43blrj6ry9.hohyzuketexppa.info/euuc7e03zb\nvel...	0
563260	linkedin.com/pub/hubert-lacroix/3/854/995	1
703686	https://www.amazon.co.uk/Technicolour-TG582n-P...	1
382056	hannalilja.se/tqvbsj/tsyq.php?djma=marker-pool...	1
566087	lrri.org/	1
445933	windmillsandeggnog.wordpress.com/	1
341359	ca.linkedin.com/pub/wayne-nelson/4/b70/792	1

Figure 3.2: Few rows of the phishing URL dataset

3.3 Modeling Tools and Techniques

- Python 3.10
- Pandas and NumPy for data manipulation and preprocessing
- Scikit-Learn modules for machine learning:
  - `sklearn.ensemble.RandomForestClassifier` for Random Forest
  - `sklearn.svm.SVC` for Support Vector Classifier
  - `sklearn.model_selection.train_test_split` for dataset splitting
  - `sklearn.preprocessing.StandardScaler` for feature scaling
  - `sklearn.metrics` for evaluation:
    - `accuracy_score`, `precision_score`
    - `recall_score`, `f1_score`
    - `roc_curve`, `auc` for ROC-AUC
    - `classification_report`, `confusion_matrix`
- Models training libraries:
  - `xgboost.XGBClassifier`
  - `sklearn.ensemble.RandomForestClassifier`
  - `sklearn.svc.SVC`

- Threshold tuning tools:
  - `numpy.linspace` for threshold grid search
  - ROC-based evaluation for optimal threshold selection
- Matplotlib and Seaborn for visual analytics
- Google Colab for cloud-based execution
- Streamlit for an interactive Interface.

CHAPTER 4

IMPLEMENTATION & TESTING

This chapter describes the implementation workflow for the phishing URL detection system. We detail the dataset construction, preprocessing, feature extraction, model development, evaluation, and threshold tuning applied to ensure unbiased performance.

4.1 Feature Extraction

Each URL in the dataset was transformed into a numerical representation via **25 handcrafted features** capturing lexical, structural, domain-based, and HTML/JS characteristics. Features that required WHOIS lookups or repeated network requests were minimized due to timeouts and performance concerns.

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	HTTPS	Shortener	Prefix_Suffix	Subdomain_Count	...	RightClick	Forwarding	Form_Tag	Suspicious_JS	Trusted_Domain	Manual_Shortener	Fake_Secure_Keyword	Suspicious_TLD	Suspicious_Path	Label
19204	NaN	0	0	0	2	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
10902	xoxodiana.com	0	0	0	3	0	0	0	0	1	...	1	0	0	0	0	0	0	0	1	0
19357	NaN	0	0	0	2	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
20864	NaN	0	0	0	5	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
32259	facebook.com	0	0	0	1	0	1	0	0	1	...	1	0	0	0	1	0	0	0	0	1
14019	NaN	0	0	1	5	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
37923	NaN	0	0	1	4	0	0	0	0	0	...	1	0	0	0	0	0	1	0	1	0
33383	NaN	0	0	0	2	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
19166	NaN	0	0	0	2	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	1
9696	iptvschedule.com	0	0	0	2	0	0	0	0	1	...	1	0	0	0	0	0	0	0	0	1

10 rows × 27 columns

Figure 4.1: Dataset after feature extraction (URL features dataset)

Table 4.1: Phishing URL Feature Extraction Overview (Compact)

Feature	**Category**	**Description**
Domain	Lexical	Extracted domain; optionally numeric encoded
Having IP	Lexical	1 if domain is an IP, 0 otherwise
@ Symbol	Lexical	1 if '@' exists in URL, 0 otherwise
URL Length	Lexical	1 if URL $\geq$ 54 chars, 0 otherwise
URL Depth	Lexical	Number of path segments
Redirection	Lexical	1 if '//' after protocol, 0 otherwise
HTTPS Usage	Lexical	1 if URL starts with https, 0 otherwise
URL Shortener	Lexical	1 if URL matches shortening service
Prefix/Suffix (-)	Lexical	1 if '-' in domain, 0 otherwise
Subdomain Count	Lexical	Number of subdomains (dots minus one)
Digits in Domain	Lexical	Count of digits in domain
Special Chars in Domain	Lexical	Count of non-alphanumeric, non-dot chars
Sensitive Keywords	Lexical	1 if URL contains keywords like secure, login, verify
Domain Age	Domain	1 if domain length $<$ 10, 0 otherwise
Domain Extension	Domain	1 if domain ends with .com/.org/.net, 0 otherwise
iFrame Tag	HTML/JS	1 if no `<iframe>`, 0 otherwise
Mouseover Event	HTML/JS	1 if onmouseover present, 0 otherwise
Right-click Disabled	HTML/JS	1 if right-click blocked, 0 otherwise
Window Location	HTML/JS	1 if `window.location` forwarding, 0 otherwise
Form Tag	HTML/JS	1 if `<form>` present, 0 otherwise
Suspicious JS	HTML/JS	1 if eval(), escape(), unescape() present
Trusted Domain	Manual	1 if domain is known trusted site, 0 otherwise
Manual Shortener	Manual	1 if domain matches URL shortener, 0 otherwise
Fake Secure Keyword	Manual	1 if URL has fake security words, 0 otherwise
Suspicious TLD	Manual	1 if domain ends with suspicious TLD, 0 otherwise
Suspicious Path	Manual	1 if path has suspicious keywords, 0 otherwise

4.2 Model Development

The URL features dataset was then split into training and test subsets using an 80/20 stratified split, ensuring that both classes were proportionally represented in the train and test sets.

4.2.1 Machine Learning Models

Three supervised models were trained to classify URLs:

1. **XGBoost Classifier:** A gradient boosting model known for handling heterogeneous features and capturing complex non-linear relationships. Model parameters included 100 estimators, maximum depth of 6, learning rate of 0.1, and subsampling ratio of 0.8.
2. **Random Forest Classifier:** An ensemble of decision trees that improves generalization and reduces overfitting. Key parameters were 100 trees, maximum features set to `sqrt`, and full utilization of CPU cores.
3. **Support Vector Classifier (SVC):** A kernel-based classifier with radial basis function (RBF) kernel, regularization parameter $C = 1.0$, and probability estimation enabled for threshold-based decision making.

4.2.2 Training Process

For each model, the following steps were performed:

1. Fit the model on the training set (X_{train}, y_{train}).
2. Ensure reproducibility by setting a fixed random seed.
3. For XGBoost and Random Forest, extract feature importance values to understand which features contributed most to model predictions.

4.2.3 Feature Importance

Feature importance was analyzed for both XGBoost and Random Forest to provide insight into the model's decision-making process. Importance scores indicate which URL characteristics are most predictive of phishing behavior.

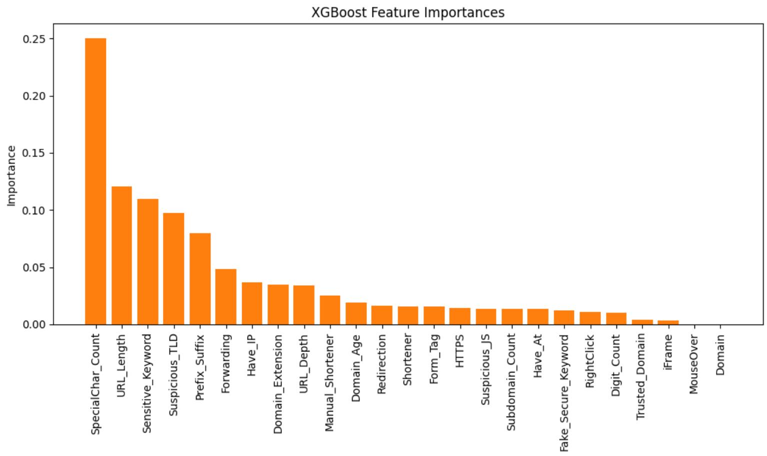

Figure 4.2: Feature Importance Plot for XGBoost Classifier

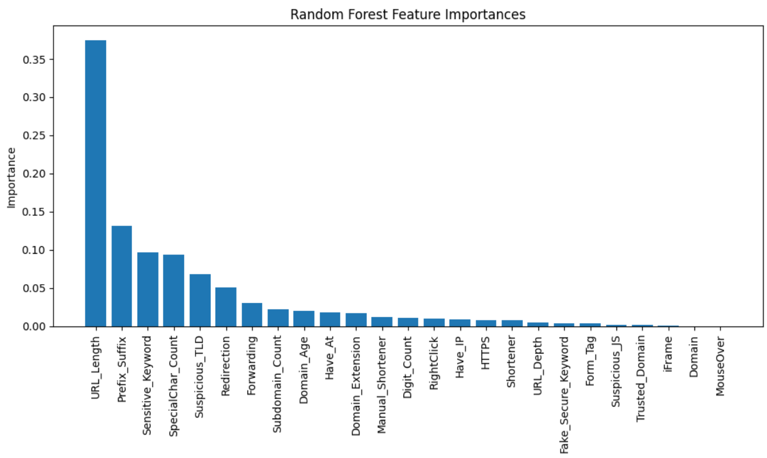

Figure 4.3: Feature Importance Plot for Random Forest Classifier

Key Observations from Feature Importance :

- **URL Depth** and **DNS Record** are the most influential features for both Random Forest and XGBoost classifiers.
- **Domain Age** and **iFrame** presence strongly impact model decisions, emphasizing the role of domain heuristics and HTML/JS behavior.
- Lexical indicators such as **Redirection patterns**, **'@' symbol**, and **URL shortening services** provide moderate predictive power.
- Behavioral features including **Mouse_Over events**, **Right-Click restrictions**, and **Web Forwarding** contribute consistently, though less prominently.
- Features like **Have_IP**, **https_Domain**, and **Web Traffic** have minimal influence on predictions, indicating they are less critical than structural and domain-related cues.

CHAPTER 5

END RESULTS & ANALYSIS

This chapter presents the evaluation and analysis of the machine learning models developed to classify URLs as *phishing* or *legitimate*. The models used include Random Forest, XGBoost, and SVM. We evaluated the models using standard metrics, confusion matrices, ROC curves, and comparison plots. Additionally, a simple user interface was implemented using Streamlit to demonstrate real-time URL classification.

5.1 Model Descriptions and Results

d

Table 5.1: Random Forest Evaluation Metrics

Class	Precision	Recall	F1-score
Legitimate (0)	0.86	0.78	0.82
Phishing (1)	0.80	0.87	0.83

Table 5.2: XGBoost Evaluation Metrics

Class	Precision	Recall	F1-score
Legitimate (0)	0.85	0.78	0.82
Phishing (1)	0.80	0.87	0.83

Table 5.3: SVM Evaluation Metrics

Class	Precision	Recall	F1-score
Legitimate (0)	0.85	0.77	0.81
Phishing (1)	0.79	0.86	0.83

Figure 5.1 shows the confusion matrices for all three models in a single figure. Each matrix visualizes the number of correct and incorrect predictions for both legitimate and phishing URLs. This provides insight into how well each model distinguishes between the two classes.

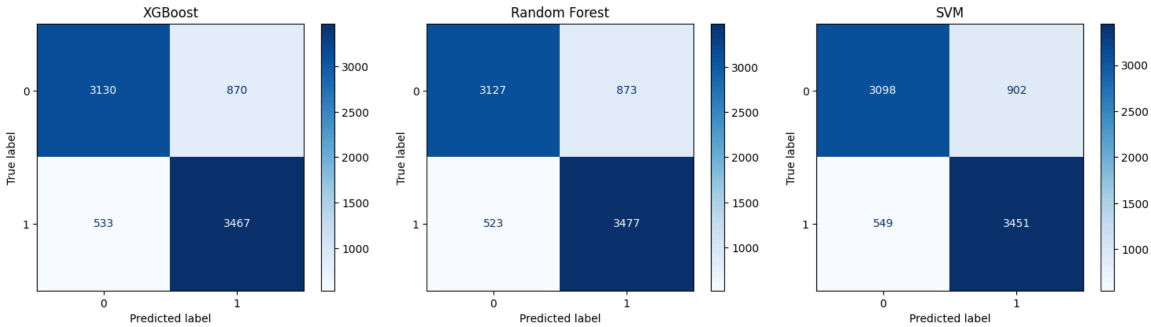

Figure 5.1: Confusion Matrices for Random Forest, XGBoost, and SVM

5.2 Model Comparison

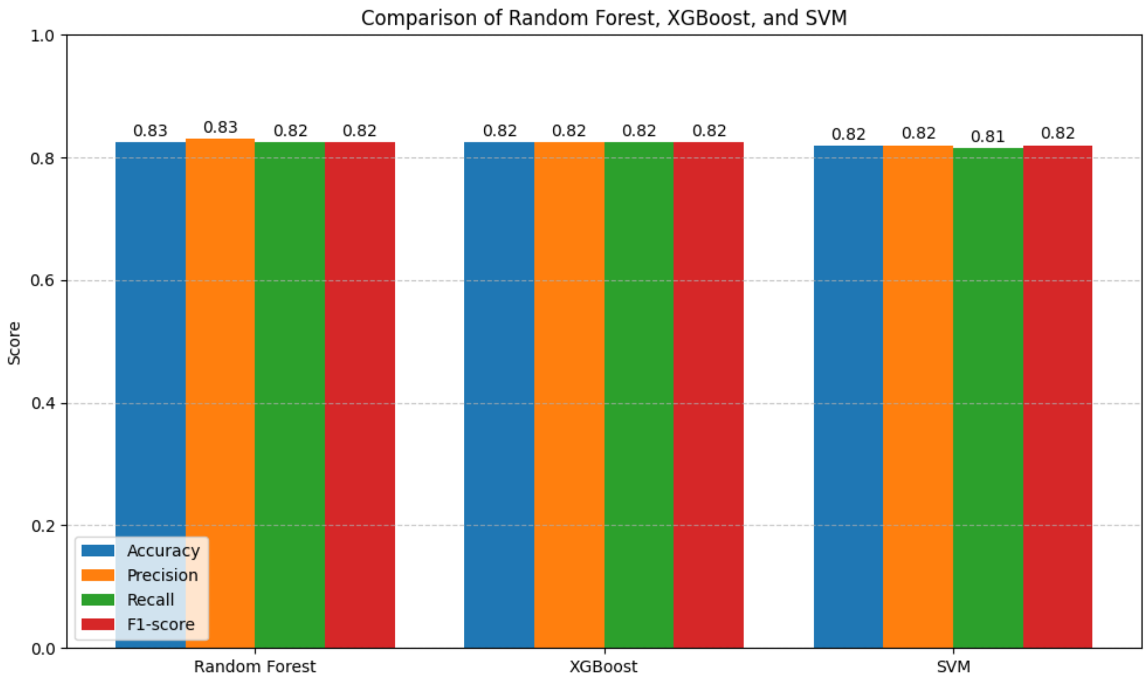

Figure 5.2: Comparison of Accuracy, Precision, Recall, F1-score for Random Forest, XGBoost, and SVM

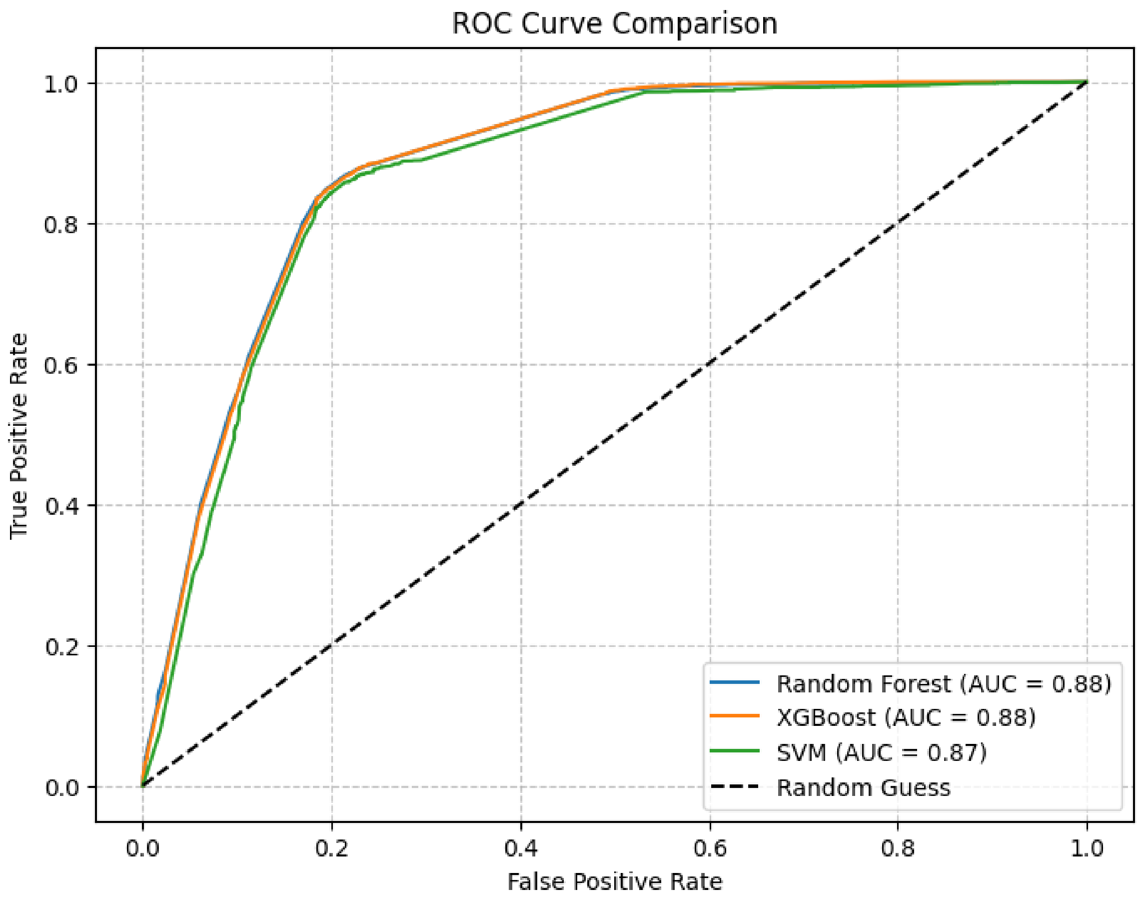

Figure 5.3: Comparison of ROC curves

5.3 User Interface Demonstration

A simple user interface was implemented using Streamlit to allow real-time phishing URL detection. The interface takes a URL as input and outputs whether it is classified as legitimate or phishing along with the reasons for the prediction.

Figure 5.4 shows a screenshot of the interface detecting a phishing URL, while Figure 5.5 shows a legitimate URL detection.

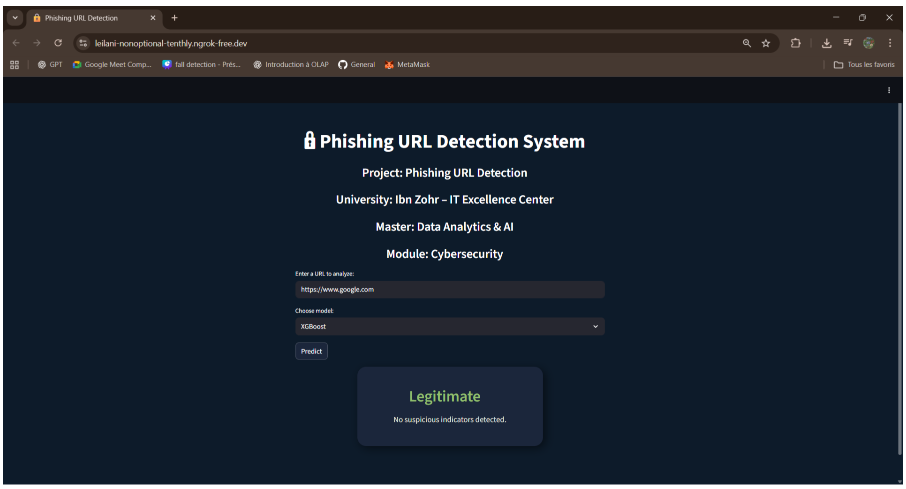

Figure 5.4: Streamlit Interface: Phishing URL Detection

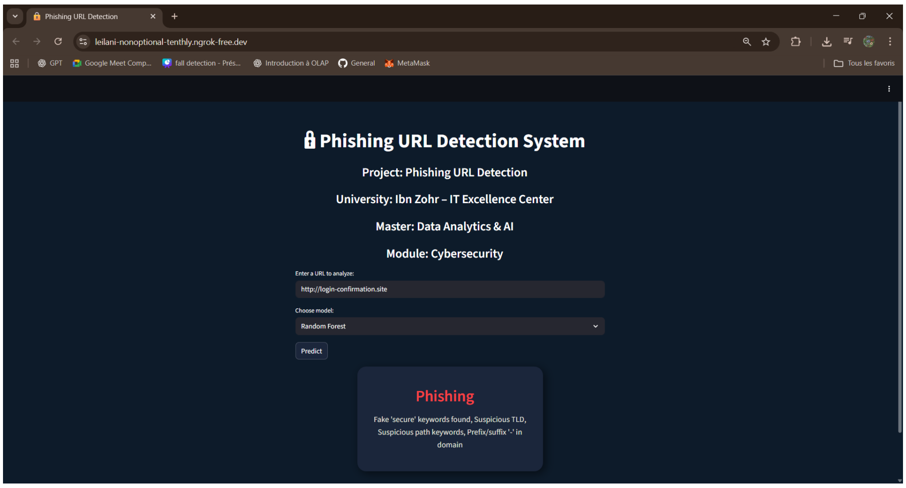

Figure 5.5: Streamlit Interface: Legitimate URL Detection

5.4 Summary

The evaluation confirms that all three models perform well for phishing URL detection. Random Forest and XGBoost show comparable performance with high recall for phishing URLs, making them suitable for cybersecurity applications. SVM provides slightly lower accuracy but still maintains good balance. The combined use of evaluation metrics, confusion matrices, ROC curves, and a simple interactive interface demonstrates both the effectiveness and interpretability of the proposed system.

CHAPTER 6

ACTIONABLE RECOMMENDATIONS & CONCLUSION

6.1 Actionable Recommendations

Based on the findings of this project, the following recommendations are proposed to enhance phishing URL detection and cybersecurity practices:

- **Real-time Implementation:** Deploy the trained models in email filters, web proxies, or browser extensions to detect phishing URLs in real-time.
- **User Awareness and Training:** Educate users about common phishing patterns, including suspicious domains, short URLs, unexpected redirections, and fake "secure" keywords.
- **Continuous Data Updates:** Regularly update the dataset with new phishing URLs and retrain the models to maintain detection accuracy and adapt to evolving phishing techniques.
- **Monitoring High-Risk Features:** Track indicators such as URLs containing IP addresses, very short domain age, unusual subdomain counts, or multiple numeric characters to flag potential attacks.
- **Layered Security Approach:** Combine machine learning-based detection with traditional cybersecurity tools, such as firewalls, antivirus software, and domain reputation services, to provide comprehensive protection.
- **Interface User Interaction:** Maintain a simple and user-friendly interface (e.g., Streamlit) for end-users or administrators to check URLs and understand why they are flagged.

6.2 Conclusion

This project successfully demonstrated the feasibility of automated phishing URL detection using machine learning models. Key achievements include:

- Training and evaluating three models—Random Forest, XGBoost, and SVM—on a balanced dataset of 40,000 URLs, achieving accuracies above 81%.
- Identifying URL lexical features, domain characteristics, and redirection patterns as strong indicators for phishing detection.
- Providing interpretable results through confusion matrices, ROC curves, and feature importance analysis.
- Developing a simple Streamlit interface that allows real-time URL analysis, making the system practical and accessible.

Overall, the project highlights that machine learning can be an effective and interpretable tool for phishing detection. With continuous updates, integration into security systems, and user education, such solutions can significantly reduce the risk posed by phishing attacks in real-world environments.

BIBLIOGRAPHY

[1] Harisudhan, 2023. *Phishing and Legitimate URLs Dataset*. Available online: Kaggle Dataset.

[2] Ars Technica, 2017. *Google phishing attack was foretold by researchers—and it may have used their code*. Available online: Ars Technica Article.

[3] Streamlit Inc., 2023. *Streamlit: The fastest way to build data apps in Python*. Available online: Streamlit Website.