Report
Group Project (UCS420)

Topic: Phishing URL Detector

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Group Members:

Harshpreet (102317160)

Arshnoor Singh (102317161)

Ansh Madaan (102317159)

Index

Cognitive Engineering Project: Phishing URL Classifier

❖ Introduction & Problem Statement

The internet has become an integral part of modern life, but it also presents significant security risks. Phishing and malware attacks have become increasingly sophisticated, making it crucial to develop automated solutions for detecting malicious websites. This project aims to build a machine learning-based **Phishing & Malware Detector** that classifies URLs as **benign, phishing, defacement, or malware** based on various features extracted from the URL and WHOIS data.

❖ Dataset Overview

The dataset used for this project is **malicious_phish.csv**, which contains URLs labeled into four categories:

- **Benign (0):** Safe websites
- **Phishing (1):** Fake websites attempting to steal user credentials
- **Defacement (2):** Websites that have been hacked and defaced
- **Malware (3):** Websites hosting malicious software

The dataset includes over 6,00,000 URLs, providing a diverse set of examples for training our model.

❖ Technology Stack

The project is implemented using the following technologies and libraries:

- **Programming Language:** Python
- **Machine Learning Library:** XGBoost
- **Data Processing:** Pandas, NumPy
- **Feature Extraction:** tldextract, re, whois
- **Model Evaluation:** scikit-learn (train_test_split, classification_report, confusion_matrix)
- **Visualization:** Matplotlib, Seaborn
- **GUI Development:** Tkinter
- **Sound Alerts:** Winsound
- **Model Storage:** Joblib

❖ ML Model Implementation & Evaluation

Data Preprocessing

- URLs are cleaned by removing "http://" and "www."
- Labels are converted into numeric values for classification.

Feature Engineering

Features extracted from each URL include:

- **Structural Features:** URL length, number of dots, hyphens, slashes, and digits
- **WHOIS Data:** Domain age
- **Subdomain Analysis:** Subdomain depth
- **Trust Score:** Assigned based on domain age, subdomain depth, and presence of suspicious words
- **IP Address Check:** Determines if the URL contains an IP address instead of a domain

Model Training

The dataset is split into **80% training data** and **20% testing data** using stratified sampling. The **XGBoost Classifier** is used with the following hyperparameters:

- **n_estimators = 300**
- **learning_rate = 0.05**
- **max_depth = 8**
- **colsample_bytree = 0.85**
- **subsample = 0.9**
- **reg_lambda = 0.5**
- **eval_metric = 'logloss'**
- **tree_method = 'hist'**

Training is performed with a progress bar to monitor progress. The model is then saved using **joblib**.

Model Evaluation

- Predictions are made on the test dataset.
- Performance is measured using the Classification Report and Confusion Matrix.
- Results are visualized using Seaborn heatmaps.

❖ Results & Insights (Visualizations & Metrics)

- The **confusion matrix** indicates how well the model distinguishes between the four classes.
- The **classification report** provides precision, recall, and F1-score.
- High **accuracy and precision** are observed in detecting phishing and malware URLs.
- The model performs well but has some misclassifications, particularly between defacement and malware categories.

❖ Challenges & Future Improvements

Challenges Faced:

- Finding the best model for classification was challenging. Several machine learning models were tested before selecting XGBoost.
- Handling missing WHOIS data: Some domains did not have complete WHOIS records, affecting feature extraction.
- Balancing the dataset: The dataset had an imbalance among categories, requiring stratified sampling for better training.
- Feature selection and engineering: Determining the best URL-based features required experimentation.

Future Improvements:

- Integrate real-time WHOIS API to fetch live domain information.
- Use deep learning approaches like LSTMs or CNNs for better URL classification.
- Develop a browser extension for real-time phishing detection.
- Improve GUI experience with additional insights into detected threats.

❖ Conclusion & Learnings

- This project successfully demonstrates how machine learning can detect **phishing and malware URLs**.
- **XGBoost performed well** due to its handling of imbalanced datasets and ability to extract meaningful patterns.
- Feature engineering plays a crucial role in URL classification.
- There is potential for further improvement using **deep learning** and real-time data integration.

❖ References

- XGBoost Documentation: https://xgboost.readthedocs.io/
- WHOIS Python Library: https://pypi.org/project/python-whois/
- Tkinter GUI Documentation: https://docs.python.org/3/library/tkinter.html
- Dataset source: Malicious URLs Dataset on Kaggle