Email Phishing Detection Using Machine Learning and NLP

Team Members and Roles

Muhammad Abdullah - Data Preprocessing, Feature Engineering

Talal Joyia - Model Training, Evaluation Metrics

Bilal Hashmi - Baseline Development, Report Compilation

1. Introduction, Objectives, and Problem Statement

Phishing emails trick users into revealing sensitive data like passwords and banking credentials.
This project aims to build a machine learning-based email phishing detection system that classifies emails as phishing or legitimate using NLP techniques and metadata features. The main objective is to minimize false negatives by maximizing recall without compromising model simplicity and speed.

2. Methodology

The methodology includes the following phases:

- Preprocessing: Removing HTML tags, cleaning text, and applying lemmatization using NLTK.
- Feature Engineering: Extracting TF-IDF vectors, counting URLs, detecting suspicious domains, and checking punctuation patterns.
- Model Training: Logistic Regression, Random Forest, and XGBoost trained with class weighting and SMOTE for imbalance handling.
- Evaluation: Metrics include Precision, Recall, F1-Score, and ROC-AUC, prioritizing Recall to catch phishing emails.

3. Dataset Description

The dataset used is 'spam.csv' from Kaggle. It contains labeled email data with fields like subject, body, and label.

- Preprocessing steps: Removing nulls, duplicates, and ensuring consistent fields.
- Feature extraction: Subject line, email body, number of URLs, HTML presence, and lexical patterns.

4. Experiments Conducted

- Baseline Model: Keyword-based heuristic detection using common phishing terms.
- Logistic Regression: As a linear baseline.
- Random Forest: Ensemble model with higher robustness.
- XGBoost: Tuned for performance with imbalanced data handling.
- Hyperparameter tuning done using GridSearchCV (time permitting).

5. Evaluation and Analysis

Metrics Used:

- Precision, Recall, F1-Score, Confusion Matrix, ROC-AUC
- Emphasis on Recall to reduce false negatives.
- XGBoost gave the best trade-off between Recall and Precision.

6. Ethical Considerations

- Only public, anonymized datasets used.
- No real emails or personally identifiable information (PII) stored.
- Results do not generalize to all languages or zero-day phishing attacks.

7. Key Learnings and Reflections

- Learned how NLP and metadata features can work together to detect phishing.
- SMOTE was effective in balancing the dataset.
- Realized the trade-offs between recall and precision in cybersecurity applications.
- Model interpretability (via SHAP) offers insights into phishing indicators.

8. Tools and Technologies

- Python (Colab Notebook)
- Libraries: scikit-learn, XGBoost, NLTK, imbalanced-learn, BeautifulSoup, SHAP (optional)
- Environment: Google Colab