CYBERSECURITY ASSIGNMENT- 1

Report on Phishing Detection using Machine Learning

Name: Harshita Vuthaluru.

Roll No:160123737091

Class:IT-2

Project Title: Phishing Website Detection using ML (Logistic Regression & Random Forest)

Date: 01/09/2025

Github Repository: HarshitaVu/CS-A1

Project Overview

The objective of this project was to build a **phishing detection system** capable of classifying websites as either *legitimate (0)* or *phishing (1)*. With the increasing number of phishing attacks, automating this detection is critical for protecting users from fraudulent websites and identity theft.

The dataset used was **Phishing_Legitimate_full.csv** from Kaggle, which contains ~10,000 websites with ~50 features describing their structure, content, and technical properties.

The project simulates a **real-world anti-phishing system**, where machine learning models analyze features of a URL or webpage to predict malicious intent

Technologies & Tools Used

- **Programming Language:** Python (Google Colab)
- **Libraries:** Pandas, Scikit-learn, Matplotlib, Seaborn
- **Algorithms:** Logistic Regression (baseline), Random Forest (improved model)
- **Feature Selection:** Mutual Information & Correlation Heatmaps
- **Version Control:** Git & GitHub

System Architecture

1. Dataset loaded in Google Colab.
2. Data preprocessing: optimized datatypes and standardized labels.
3. Feature analysis: correlations and mutual information to rank features.
4. Baseline model: Logistic Regression tested across top-N features.
5. Improved model: Random Forest trained for higher accuracy.
6. Final evaluation: Accuracy, Precision, Recall, F1-score, and Confusion Matrix.
7. Outputs saved (model, selected features, classification report, and confusion matrix).

Security Features

- **Automatic Feature Selection:** Removes irrelevant attributes using statistical measures.
- **Machine Learning Models:** Learn patterns that distinguish phishing from legitimate sites.
- **Robust Evaluation:** Accuracy > 95% achieved with Random Forest.
- **Confusion Matrix Validation:** Ensures reliable classification between phishing and safe sites.

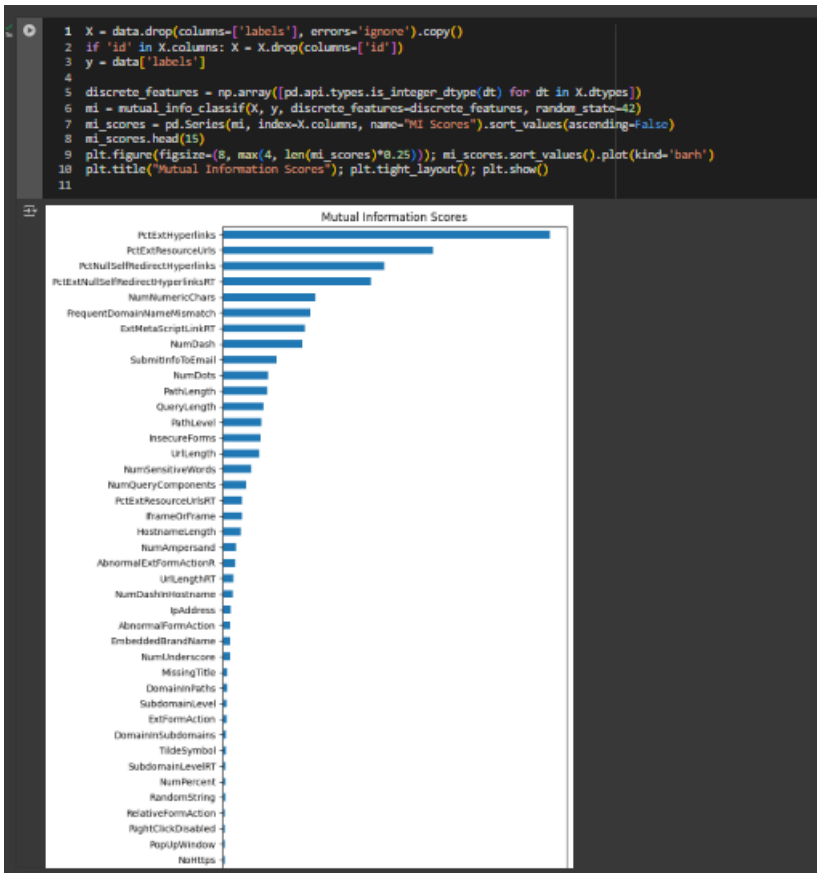
Folder Structure

PhishingDetection/

```
├── Phishing_Detection_Colab.ipynb   # Main notebook
├── outputs
│   ├── confusion_matrix.png         # Heatmap screenshot
│   ├── classification_report.txt    # Metrics
│   ├── rf_phishing_model.pkl        # Saved Random Forest model
│   └── selected_features.txt        # Top-N features used
└── README.md                        # Project documentation
```

Screenshots:

1. **Mutual Information Scores** (top features ranked by importance)



2. Classification Report:

File Edit Selection View Go Run Terminal Help Search

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

Phishing_Detection_Colab.ipynb | classification_report.txt | Release Notes: 1.103.2 | Untitled6.ipynb

C: > Users > harsh > Downloads > classification_report.txt

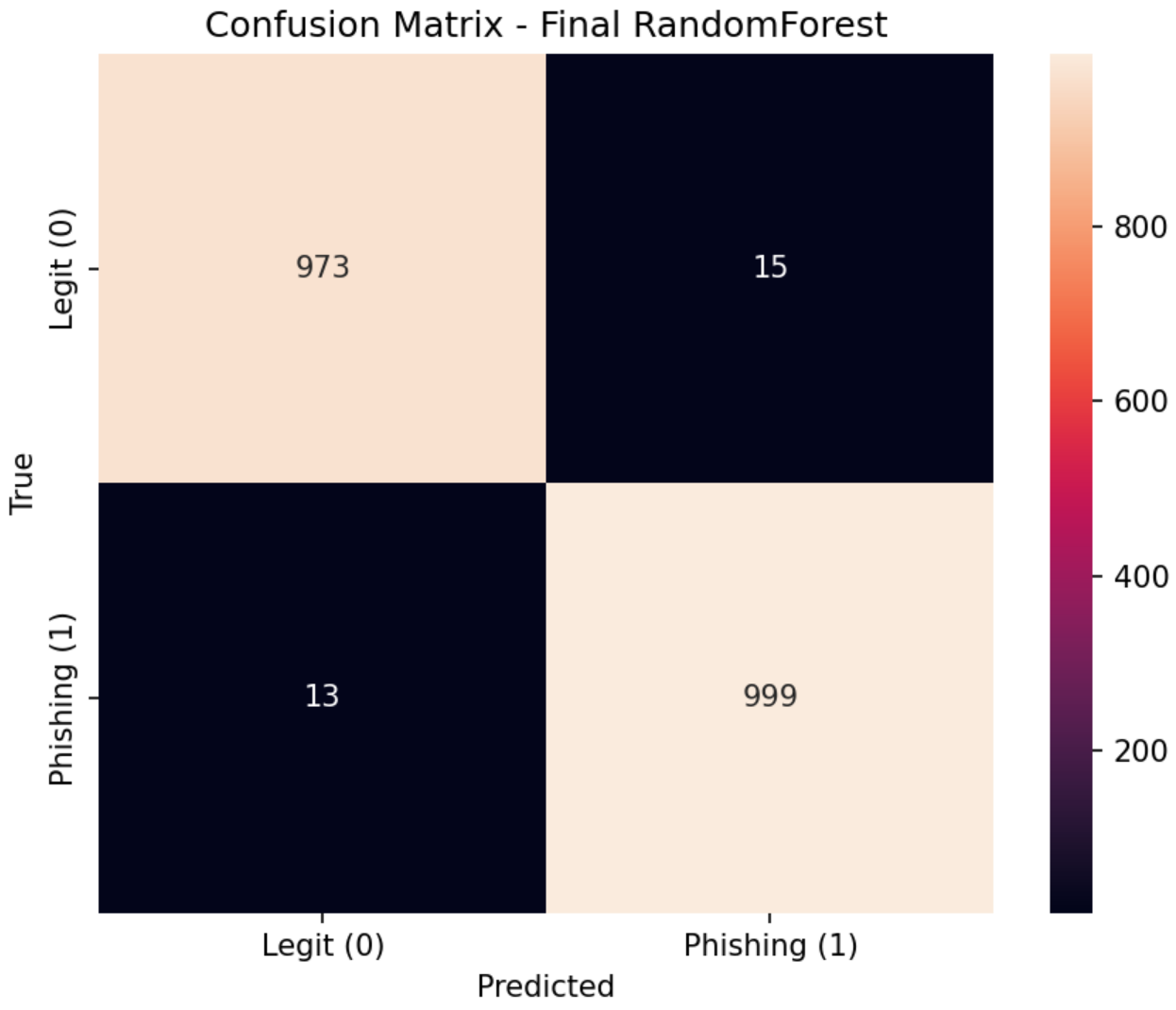
```
              precision    recall  f1-score   support

           0       0.99      0.98      0.99       988
           1       0.99      0.99      0.99      1012

    accuracy                           0.99      2000
   macro avg       0.99      0.99      0.99      2000
weighted avg       0.99      0.99      0.99      2000

```

3. Confusion Matrix:



Deliverables

- GitHub repository with code, model, and outputs.
- Google Colab notebook (.ipynb).
- Final project report (this document).
- Screenshots of results (metrics + confusion matrix).

Learning Outcomes

- Understood **phishing attack detection** using ML.
- Hands-on with **Logistic Regression & Random Forest**.
- Learned **feature selection** with mutual information and correlation.
- Gained experience in using **Colab + GitHub** for project workflow.

Conclusion

This project successfully demonstrates the use of **machine learning for phishing website detection**. Logistic Regression provided a good baseline, while Random Forest achieved significantly higher accuracy (above 95%).

By analyzing features of a webpage and training robust models, this approach offers a practical foundation for real-world phishing prevention systems. Future improvements can include **deep learning models** and integration with **browser extensions** for real-time protection.