PROJECT REPORT
Phishing URL Detection Using Machine Learning

---

1. Introduction

Phishing has become one of the most common ways attackers trick people into sharing passwords, bank details, and personal information. Most phishing attempts start with a suspicious link that looks similar to a trusted website. Since manually checking every URL is not practical, this project aims to build a simple machine learning model that can identify whether a given URL is legitimate or phishing.

The purpose of this project is not to create an advanced cybersecurity tool, but to understand how machine learning can be applied to a real-world security problem using basic features. The entire project is built with beginner-friendly code and uses a small synthetic dataset created manually.

---

2. Problem Statement

Goal:

To classify URLs into two categories: legitimate and phishing, using simple, explainable machine learning techniques.

This involves:

- Creating a small dataset
- Extracting logical features from URLs
- Training a Decision Tree classifier
- Predicting whether a new URL is suspicious or safe

---

3. Dataset Description

Instead of using large external datasets, I created a small, clean dataset suitable for learning purposes.

The dataset contains two types of URLs:

a) Legitimate URLs:
Well-known and trusted websites, e.g.:

- Google
- YouTube

- GitHub
- Amazon
- Wikipedia
- VIT Bhopal official website

b) Phishing URLs:
Manually written based on common phishing patterns, e.g.:

- URLs containing “verify”, “login”, “refund”, “secure-update”
- Fake PayPal and Amazon pages
- Suspicious domains using .xyz, .info, etc.
- Long and misleading URLs pretending to be banking websites

A few variations were added by appending query strings (like ?id=1023).
The final dataset was stored in urls.csv with two columns: url and label.

---

4. Methodology

This project follows a simple step-by-step pipeline:

4.1 Feature Extraction

URLs are converted into numerical features for the ML model:

- Total URL length
- Number of dots .
- Number of slashes /
- Number of special characters (@, -, =, ?)
- Whether the URL starts with https://
- Length of the domain name
- Number of suspicious keywords in the URL

(Features calculated in utils.py)

4.2 Model Selection

Chosen Model: *Decision Tree Classifier*
Reasons:

- Easy to understand
- Works well for small datasets

- Provides explainable results
- Ideal for beginner-level projects

4.3 Training the Model

- Data split into training/testing (80:20 ratio)
- Model trained with extracted features
- Trained model saved as url_phish_dt.pkl

4.4 Prediction

A script (predict_url.py) created to test any URL from the command line.

---

5. Result and Analysis

Model performed well on both training and testing data.
Accuracy was high in a controlled environment.

Example prediction:
URL: http://paypal.verify-login.xyz/confirm
Prediction: *phishing*

Model correctly identified phishing URLs using patterns like:

- keyword “verify”
- “paypal”
- unusual domain .xyz
- missing HTTPS

Model also correctly labeled trusted sites:

- https://www.google.com → legitimate
- https://github.com → legitimate

---

6. Limitations

This project is for demonstration and has limitations:

1. The dataset is small and synthetic
2. Attackers may avoid obvious suspicious keywords
3. Real-world detection needs hundreds of features
4. Decision Trees may overfit small datasets

This tool is an academic project—not a production security system.

---

7. Future Improvements

Potential extensions:

- Use real phishing datasets (PhishTank/OpenPhish)
- Advanced features: token analysis, domain age, WHOIS info
- Ensemble models (Random Forest)
- NLP techniques for URL analysis
- Browser extension implementation