Phish-Net: Catching Fake Sites Before They Catch You
(A machine learning-powered lie detector for websites — because not every “Login Here” link is your friend)

Abstract

The internet is full of surprises—some good, some malicious. This project focuses on identifying phishing websites using supervised machine learning techniques. Using a dataset of website attributes, we trained Random Forest and K-Nearest Neighbors (KNN) models to detect whether a given website is legitimate or a phishing attempt. The system achieved an impressive accuracy of 97.14% using Random Forest. The project includes data preprocessing, visualization, training/testing models, and evaluating results—all built with Python tools.

Objective

- To detect phishing websites using Machine Learning models.
- To compare Random Forest and KNN for classification accuracy.
- To visualize patterns in the dataset that relate to phishing indicators.
- To gain hands-on experience in real-world ML model development.

🧠 Technologies Used

Tech Stack	Purpose Description
Pandas	For loading, cleaning, and managing the dataset.
Scikit-learn	For building and evaluating ML models (Random Forest, KNN).
Matplotlib	For plotting graphs and visualizing the dataset.
Seaborn	For enhanced statistical plotting (pie, bar, heatmaps).
Jupyter Lab	The working environment for all coding and analysis.

⏳ Duration

Start: 4 Weeks before submission 😅
End: Night before the viva 😬
(Realistically: ~15–20 hours of focused effort over 2–3 weeks)

Procedure Description

1. **Data Import & Cleaning**
    - Imported dataset using Pandas
    - Removed null values and irrelevant features
2. **Data Processing & Encoding**
    - Converted categorical columns to numerical format using LabelEncoder
    - Split dataset into training and testing sets (70/30)
3. **Data Visualization**
    - Created pie charts, bar graphs, and correlation matrices
    - Used `seaborn` and `matplotlib` to find data insights
4. **Model Training**
    - Used `RandomForestClassifier` and `KNeighborsClassifier`
    - Evaluated using accuracy, sensitivity, specificity, F1-score
5. **Performance Comparison**
    - Random Forest: 97.14% accuracy
    - KNN: Slightly lower, but still solid

Results & Observations

- Random Forest provided better performance due to its ensemble nature.
- Visualizations showed phishing sites tend to have patterns in URL length, age, and redirection.
- Data preprocessing plays a huge role in model accuracy.

Conclusion

The project successfully predicted phishing websites using machine learning algorithms with high accuracy. The use of Random Forest proved more effective than KNN for this dataset. In the future, this system can be integrated into browser extensions or security tools to provide real-time phishing detection. This mini project also gave practical exposure to machine learning workflows from data to model.