TITLE: PHISHING URL DETECTION USING MACHINE LEARNING

Problem Statement:

Phishing attacks continue to be one of the most dangerous threats on the internet, tricking users into providing sensitive information through fake websites. These attacks typically involve fraudulent URLs that resemble legitimate sites. The goal of this project is to design and implement a machine learning-based system that can analyze URLs and classify them as either "Phishing" or "Safe" to help users avoid falling victim to such scams.

Team Members:

1. Shree Shivani. M (RRN:220071601241)
    - Implemented the machine learning model (Decision Tree).
    - Designed and handled the training pipeline (with GridSearchCV).
    - Evaluated model performance (accuracy, ROC, confusion matrix).

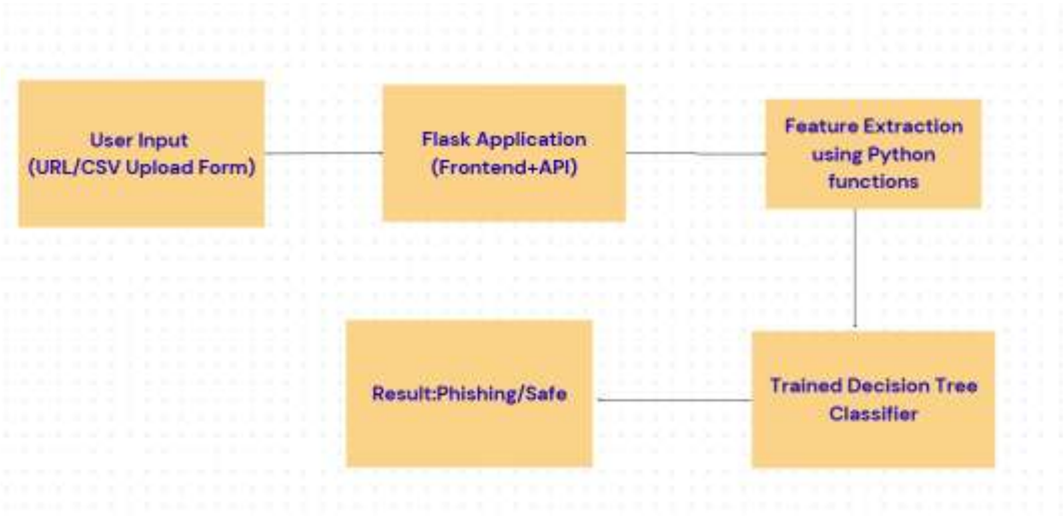
2. Yeshikasri. K (RRN:220071601272)
    - Built the front-end using HTML + Bootstrap.
    - Integrated the model using Flask and connected the user input pipeline.
    - Designed the UI to support both single URL and batch CSV upload.

Project Overview:

This project leverages a Decision Tree Classifier to detect phishing URLs by extracting various features from the URL string itself (e.g., length, number of special characters, keyword presence, entropy, etc.). The system includes:

- A Python-based training script for model creation.
- A Flask-based web dashboard for end-users to input URLs or upload CSV files.
- Data visualization tools such as Confusion Matrix and ROC Curve for model evaluation.

Architecture Diagram:



How the Code Works:

1. **Feature Extraction:**
    - URLs are parsed using urlparse().
    - Features such as URL length, presence of "@", digit count, entropy, and use of HTTPS are calculated.
    - Suspicious keywords (e.g., login, account, verify) are counted.

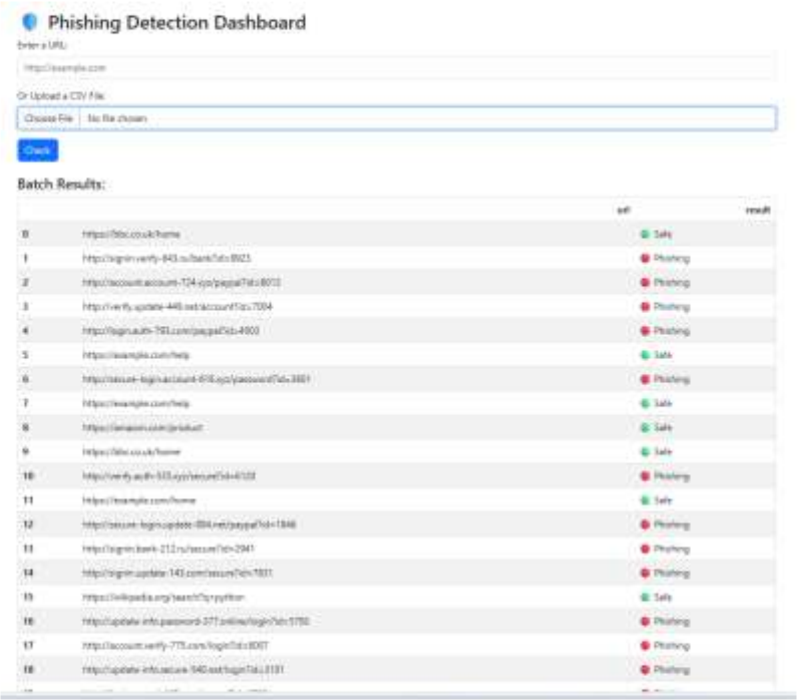
2. **Model Training (train.py):**
    - A Decision Tree Classifier is trained using the extracted features.
    - GridSearchCV is used to tune hyperparameters.
    - The model is saved using joblib into model/dt_model.pkl.

3. **Web Application (app.py):**
    - Built using Flask.
    - Accepts single URL input or CSV file uploads.
    - Extracts features, loads the trained model, and makes predictions.
    - Displays results on the web dashboard.

4. **Visualization:**
    - Confusion matrix and ROC curve are plotted.
    - Accuracy, precision, recall, and F1 score are calculated.

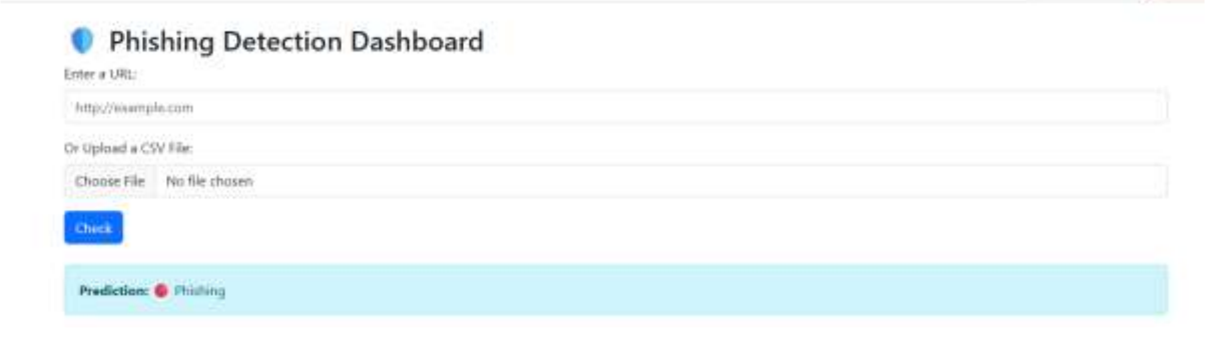
VDOP (Visual Data Output Presentation) Application:

- Users are presented with a simple, Bootstrap-styled web page.
- Results are color-coded:
    - Green dot and "Safe" for legitimate URLs.
    - Red dot and "Phishing" for malicious URLs.
- Upload CSV feature allows batch prediction.
- The application makes phishing detection accessible for non-technical users.

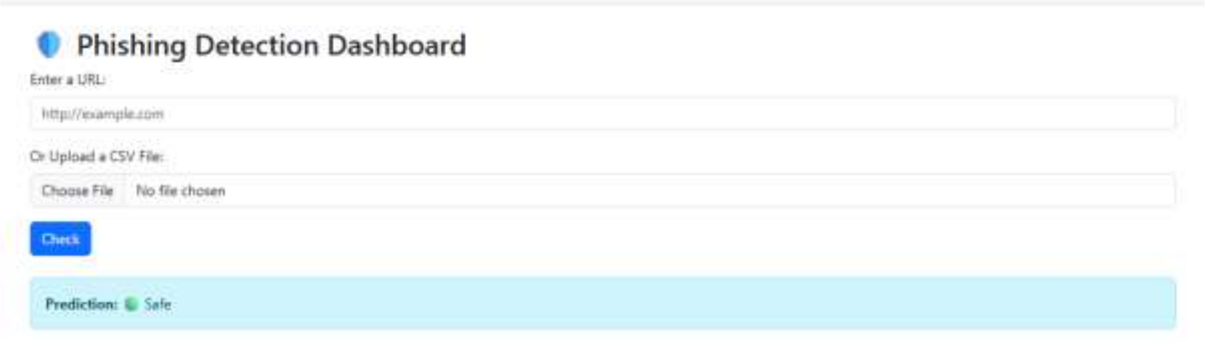


Batch Results:

	url	result
0	https://bbc.co.uk/home	🟢 Safe
1	http://signin.verify-843.ru/bank?id=8923	🔴 Phishing
2	http://account.account-124.xyz/paypal?id=8013	🔴 Phishing
3	http://verify.update-448.net/account?id=7004	🔴 Phishing
4	http://login.auth-793.com/paypal?id=4900	🔴 Phishing
5	https://example.com/help	🟢 Safe
6	http://secure-login.account-618.xyz/password?id=3881	🔴 Phishing
7	https://example.com/help	🟢 Safe
8	https://amazon.com/product	🟢 Safe
9	https://bbc.co.uk/home	🟢 Safe
10	http://verify.auth-533.xyz/secure?id=4133	🔴 Phishing
11	https://example.com/home	🟢 Safe
12	http://secure-login.update-884.net/paypal?id=1848	🔴 Phishing
13	http://signin.bank-212.ru/secure?id=2941	🔴 Phishing
14	http://signin.update-143.com/secure?id=7831	🔴 Phishing
15	https://wikipedia.org/search?q=python	🟢 Safe
16	http://update-info.password-377.online/login?id=5750	🔴 Phishing
17	http://account.verify-775.com/login?id=8067	🔴 Phishing
18	http://update-info.secure-940.net/login?id=3131	🔴 Phishing



Prediction: 🔴 Phishing

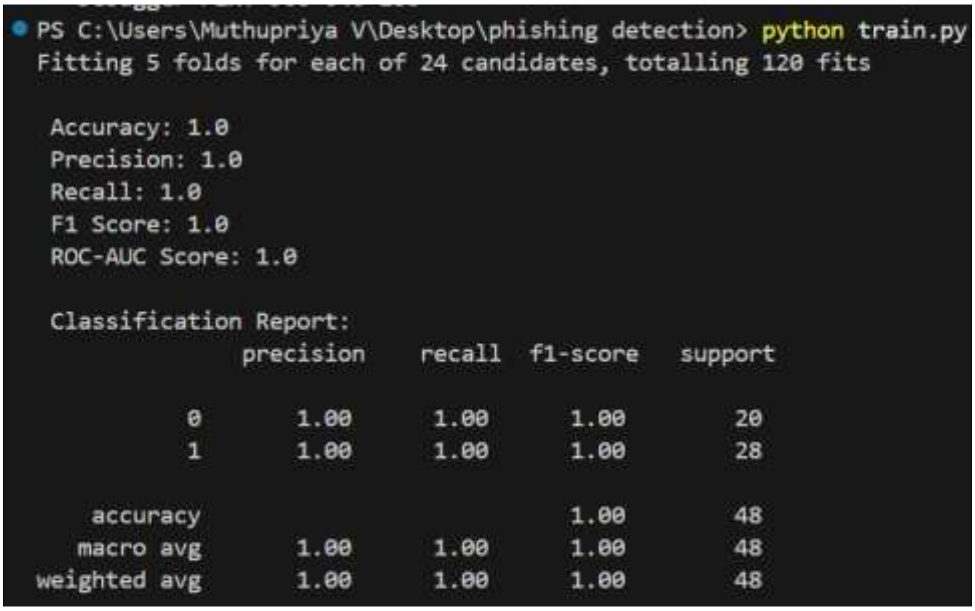


Prediction: 🟢 Safe

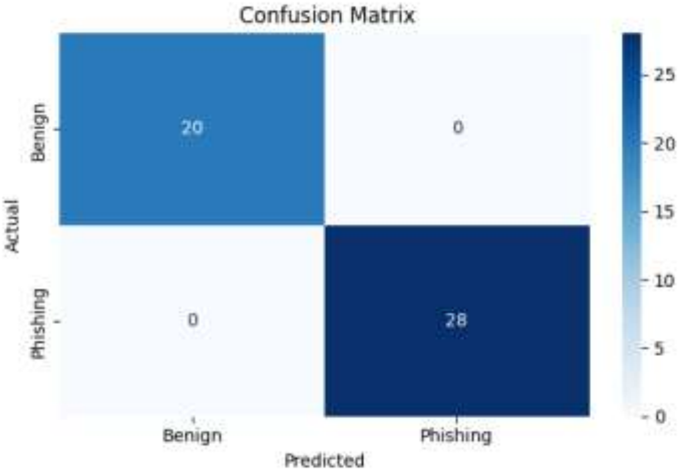
Model Performance:

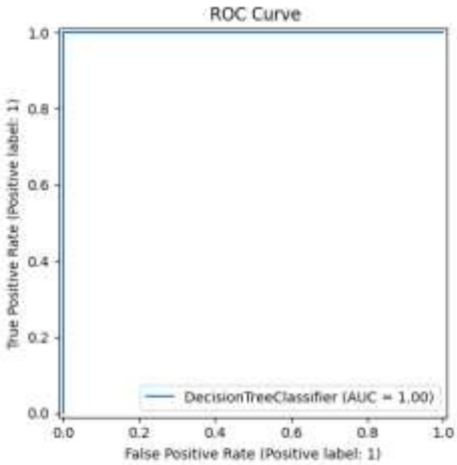
- **Accuracy:** 100%
- **AUC Score:** 1.0
- **Confusion Matrix:** Perfect classification (no false positives or false negatives).

Note: Although 100% accuracy sounds ideal, it can indicate overfitting due to the small dataset (241 rows).



```
PS C:\Users\Muthupriya V\Desktop\phishing detection> python train.py
Fitting 5 folds for each of 24 candidates, totalling 120 fits

 Accuracy: 1.0
 Precision: 1.0
 Recall: 1.0
 F1 Score: 1.0
 ROC-AUC Score: 1.0

 Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00        20
           1       1.00      1.00      1.00        28

    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48
```





Conclusion:

The phishing detection system demonstrates how machine learning can enhance online safety. Despite the limited dataset, the model shows excellent performance. The dashboard interface makes the tool user-friendly and practical for real-world use.

Future improvements may include:

- Using a larger and more diverse dataset.
- Exploring advanced ML models like Random Forest or XGBoost.
- Integrating with browsers for real-time URL scanning.

Technologies Used:

- Python
- Flask
- Scikit-learn
- Pandas
- Numpy
- Bootstrap (for UI)
- Joblib (for model saving/loading)

References:

- https://scikit-learn.org/
- https://flask.palletsprojects.com/
- https://phishstats.info/ (for phishing data)

Project Video:

https://drive.google.com/file/d/1c3B9d6rl37d8JcRQ9ySUaLvWKuz8DNWW/view?usp=sharing