AI-POWERED PHISHING EMAIL DETECTOR

PRESENTED BY PRIYALI POOJARI

INTRODUCTION

INTRODUCTION

What is Phishing?

- Phishing emails are designed to trick users into revealing sensitive information like passwords, credit card numbers, or login credentials.
- A cyber attack where attackers disguise themselves as trustworthy entities in electronic communications.

Why is Phishing Detection Important?

- Phishing causes tremendous financial and data losses globally. - Users often fail to accurately identify phishing emails due to growing sophistication.

Our Goal:

- Present an AI-based solution to detect phishing emails using a machine learning model.
- Build a simple and effective tool using Python, featuring both a Command-Line Interface (CLI) and a Graphical User Interface (GUI) for predictions, enhancing email security.

Problem

Phishing attacks are becoming more sophisticated, making it difficult for traditional email filters and users to detect them. Manual detection is unreliable, and there's a growing need for intelligent, real-time, and scalable protection.

- ❌ Prone to Errors:
  Human judgment often fails under pressure or due to lack of expertise.
- 🛡️ Outdated Filters:
  Traditional spam filters struggle to detect modern phishing techniques.
- ♻️ Constantly Evolving Tactics:
  Attackers continuously modify their strategies to bypass defenses.
- ⚡ Need for Real-Time Detection:
  Delays in detection can lead to major breaches and data loss.
- 🤖 Demand for Automation:
  Intelligent tools using machine learning offer scalable, reliable threat detection.

Looks Legitimate:
Phishing emails closely resemble authentic messages, making them hard to detect.

Rising Threats:
The volume and complexity of phishing attacks are increasing rapidly.

Time-Consuming:
Manual analysis of emails is slow and inefficient.

Phishing Email Incidents – Top 4 States (India)

11.5k

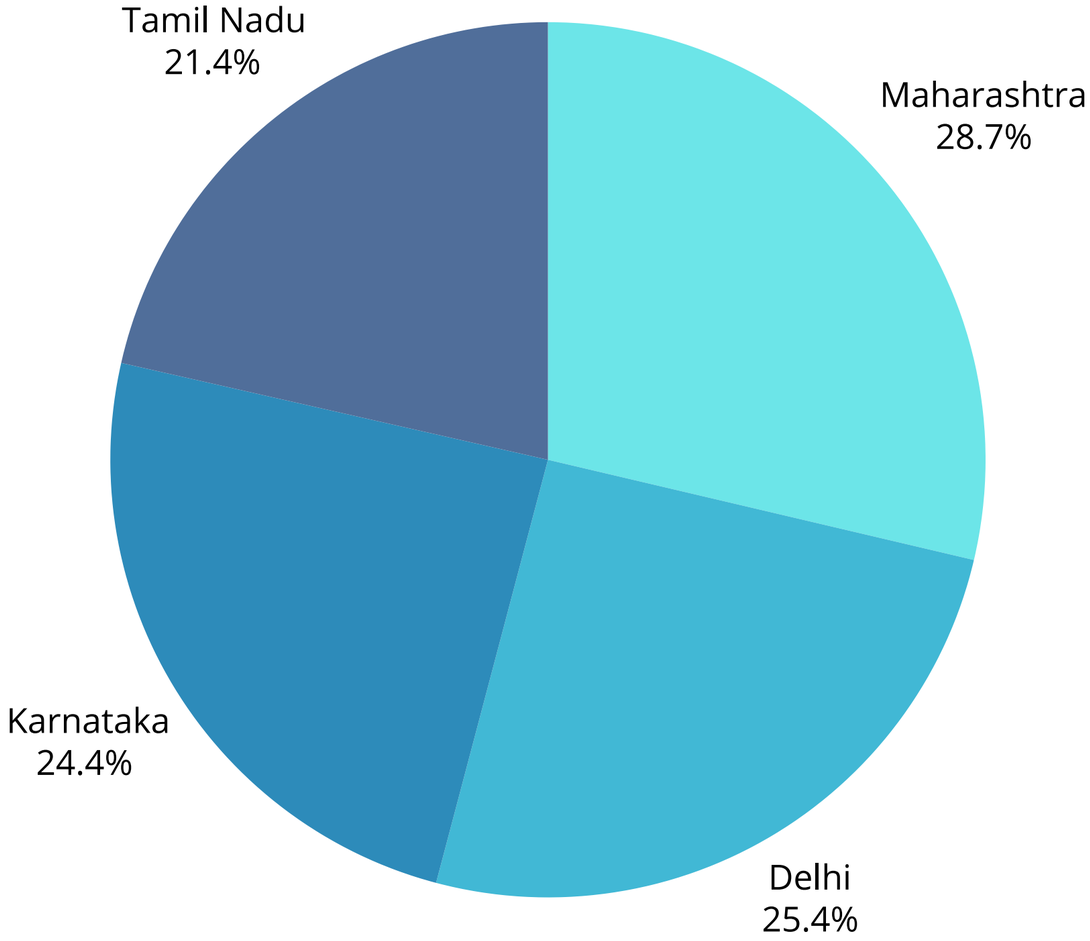
Maharashtra

High digital population and financial hubs attract more phishing attacks.

10.2k

Delhi

Heavy use of digital services and government targets increase incidents.



9.8k

Karnataka

Home to many tech firms; hackers exploit digital adoption.

8.6k

Tamil Nadu

Growing online service usage makes users vulnerable to sophisticated phishing.

Proposed Solution

- Implement a machine learning-based phishing email detector.

Approach:

- Use natural language processing techniques to convert email text into numeric features.
- Train a Random Forest classifier to discriminate phishing vs legitimate emails.

Benefits:

- Automates email classification.
- Helps users and organizations prevent phishing attacks effectively.

Data Preparation

Combined dataset:

- phishing email samples (labeled as 1).
- legitimate email samples (labeled as 0).

Examples:
Phishing: "Your account has been suspended. Click to verify."
Legitimate: "Security alert: Sign-in from new device detected."

Dataset format:
- Stored in a Pandas DataFrame with two columns: 'text', 'label'.

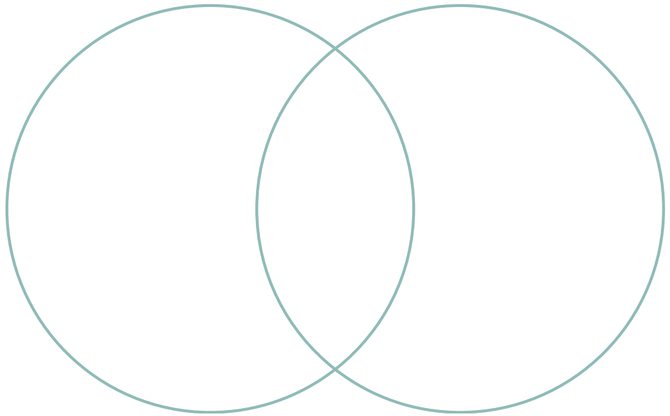
Feature Extraction
TF-IDF Vectorization

- Text data needs to be converted to numeric representations.
- Limited to top 3000 most relevant words.
- Using TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer:

1. It evaluates the importance of a word relative to the document and entire corpus.
2. Produces sparse matrix representation of email text features.

- Advantages:
    1. Captures meaningful words while down-weighing common words.
    2. Effective for textual classification tasks.

Model Training – Random Forest Classifier



Random Forest algorithm overview:
- Ensemble of decision trees.
- Uses majority voting to classify inputs.
- Reduces overfitting compared to single decision tree

Model setup:
- Number of trees (estimators): 100.
- Random state set for reproducibility.
- Training:
- Fit model on vectorized email content and labels (phishing or legitimate).

.

Model Saving & Reuse

Model and Vectorizer Persistence

- Trained Random Forest model and TF-IDF vectorizer saved using joblib
- File path:

  `model/phishing_model.pkl`

Why This Matters (Benefits)

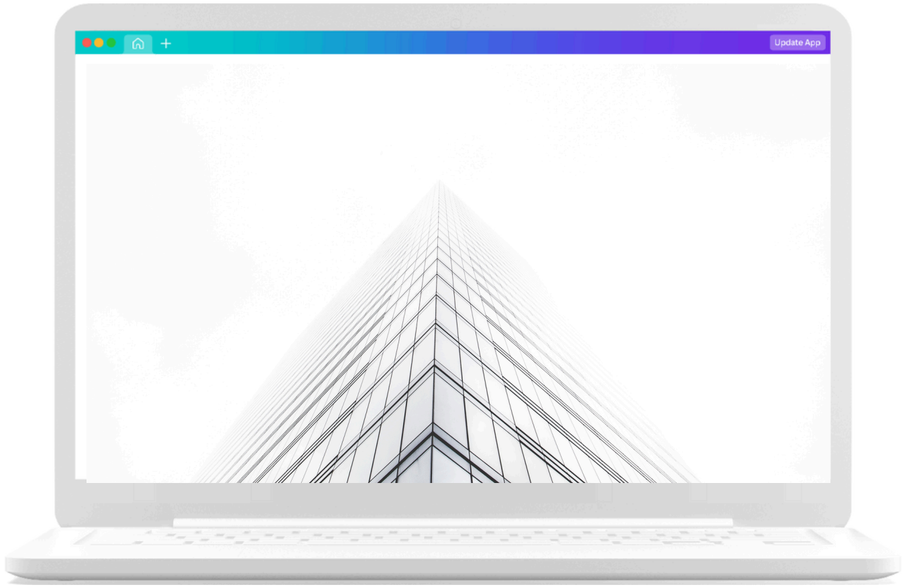
Reusability

- No need to retrain the model every time
- Saves compute resources and time

Consistency

- Ensures consistent predictions using the same vectorizer
- Prevents mismatched feature mapping

Prediction Functionality

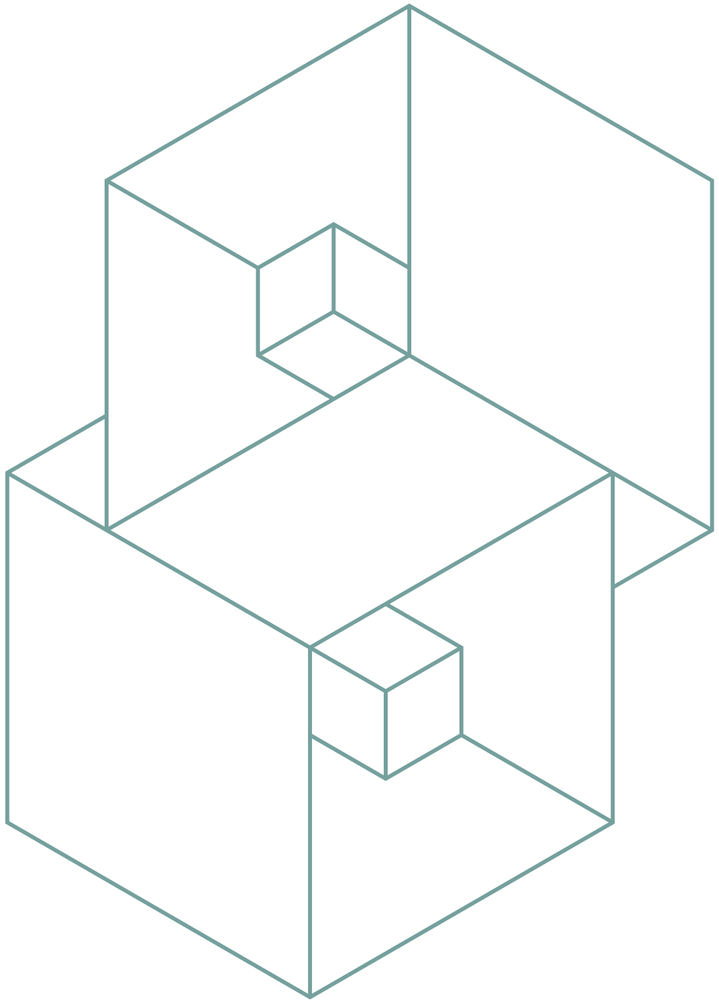


Command-line interface tool (`phishing_detector.py`):

1. Allows users to input email content via typing multiline text or loading from a `.txt` file.
2. Loads trained model and vectorizer.
3. Transforms input email text and predicts phishing or legitimate.

Output displays:

Emoji indicator ("🔴" = Phishing, "🟢" = Legitimate).



User-Friendly GUI (Tkinter)

- Desktop application interface for easy usage.
- Features:

1 . Text input area with scroll.
2. Buttons for Predict, Load from file, Clear, and Save Result.
3. Real-time prediction with display of confidence score.
4. Tooltips explaining button functionality.
5. Keyboard shortcuts for accessibility (e.g. Ctrl+P for Predict).

- Visual Style:

1. Lilac background with Georgia font and dark text input area.
2. Clean, modern look with clear labels and user guidance.

Real-World Use Cases

01

Email client integration:

Embed phishing detection into popular email clients (Outlook, Gmail add-ons).

02

Corporate security:

Use to automatically scan and quarantine suspicious emails in enterprise systems.

03

User education:

Help employees learn to recognize phishing emails by showing live predictions.

04

Incident response:

Quickly identify phishing attempts in large volumes of email traffic.

Future Enhancements

01

Expand dataset:
Collect and label more phishing and legitimate emails for higher model accuracy.

02

Use advanced models:
Experiment with deep learning models such as LSTM or transformers for context-aware detection.

03

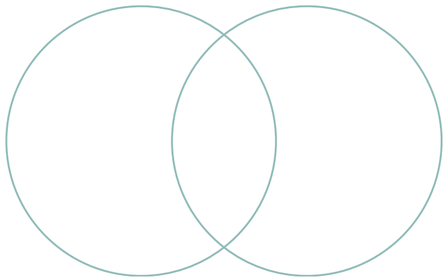
Feedback loop:
Implement user feedback feature for model refinement over time.

04

Feedback loop:
Implement user feedback feature for model refinement over time.

CONCLUSION

- Developed an AI-powered email phishing detection system based on Random Forest and TF-IDF.
- Provides both command-line and graphical user interface tools.
- Enhances email security by automating recognition of phishing attempts.
- Modular design allows easy extension and improvement.
- Important step towards safer email communication.



Thank you very much!

PRESENTED BY PRIYALI POOJARI