Mini-Project Report

On

"Social engineering attack detector"

Submitted for the partial fulfilment of Fifth Semester subject **"Machine Learning"**

Course code: **CM303G$**

Submitted by: -

1.	Daksh Titarmare	2213063
2.	Vaishnavi Hemke	2213024
3.	Nandani Kshirsagar	2213034

Under the Guidance of

Dr. K.L Bawankule
Lecturer
Department of Computer Engineering

ODD-24

Department of Computer Engineering

Government Polytechnic, Nagpur

(An Autonomous Institute of Government of Maharashtra)

Certificate

This is certified that the Mini Project titled

“Social engineering attack detector ”

Carried out under guidance of

Dr. K.L Bawankule

Lecturer

Department of Computer Engineering

Submitted by: -

1.	Daksh Titarmare	2213063
2.	Vaishnavi Hemke	2213024
3.	Nandani Kshirsagar	2213034

As the partial fulfilment of Fifth Semester subject “MACHINE LEARNING”

Course code CM303G$^{\$}$ during the term ODD-24

Dr. K.L Bawankule

Guide and Lecturer

Department of Computer Engineering

Dr. M. V Sarode

Head of Department

Department of Computer Engineering

ODD-24

Department of Computer Engineering

Government Polytechnic, Nagpur
(An Autonomous Institute of Government of Maharashtra)

ACKNOWLEDGEMENT

We would like to place on record our deep sense of gratitude to our guide, **Dr. K.L. BAWANKULE** Department Of Computer Engineering for his generous guidance, helpful and useful suggestions, continuous encouragement and supervision throughout the course of present work.

We express our sincere gratitude to **Dr. M V SARODE**, Head of Department of Computer Engineering for his stimulating guidance. The success of any work depends on efforts of many individuals. We would like to take this opportunity to express out deep gratitude to all those who extended their support and have guided us to complete this project work.

We are extremely thankful to **Dr. R.P. MOGRE**, Principal for providing us infrastructural facilities work in, without which this work would not have been possible

1.	Daksh Titarmare	2213063
2.	Vaishnavi Hemke	2213024
3.	Nandani Kshirsagar	2213034

INTRODUCTION TO TOPIC

Phishing attacks are among the most prevalent cybersecurity threats, aiming to deceive users into revealing sensitive information through malicious websites or emails. Traditional detection methods, relying heavily on human intervention, often fail to keep up with the evolving tactics of attackers. This project leverages machine learning to automate phishing detection by analyzing URL features. By utilizing a Support Vector Machine (SVM) model, the system ensures efficient, accurate, and scalable detection of phishing threats.

Objectives:

1. **Automated Threat Detection:** Identify phishing URLs without manual intervention, enhancing security protocols.
2. **High Accuracy:** Minimize false positives and negatives for reliable threat identification.
3. **Scalable and Adaptable:** Deploy the model in various environments, from personal systems to large enterprises.
4. **Real-Time Processing:** Enable instant detection of threats for timely preventive actions.
5. **User-Friendly Implementation:** Provide an easy-to-deploy solution for non-technical users.

Methodology:

The project employs supervised learning with SVM as the classification algorithm. The methodology includes:

1. **Data Collection:**
   - A comprehensive dataset containing phishing and legitimate URLs, along with labeled features, is utilized.
2. **Data Preprocessing:**
   - Handled missing values by filling or dropping them as required.
   - Extracted and organized features for model training.
3. **Model Selection and Training (SVM):**
   - Chose a linear Support Vector Classifier (SVC) for its effectiveness in binary classification tasks.
   - Created a pipeline to preprocess data and streamline model training.
4. **Evaluation and Validation:**
   - Split the dataset into training and testing subsets.
   - Evaluated the model using accuracy, precision, recall, and classification reports to optimize performance.
5. **Deployment:**

- Saved the trained model for future use.
- Provided an interface for real-time URL prediction, identifying phishing threats instantly.

Applications:

1. **Cybersecurity Enhancement:** Strengthen defenses against phishing attacks in real-time.
2. **URL Filtering:** Automate identification of malicious URLs in emails and web traffic.
3. **Scalable Deployment:** Apply across personal devices, corporate environments, or internet service providers.
4. **User Awareness:** Assist in educating users by highlighting phishing traits in detected URLs.
5. **Preventive Measures:** Serve as a proactive tool for identifying and mitigating potential threats.

PROGRAM

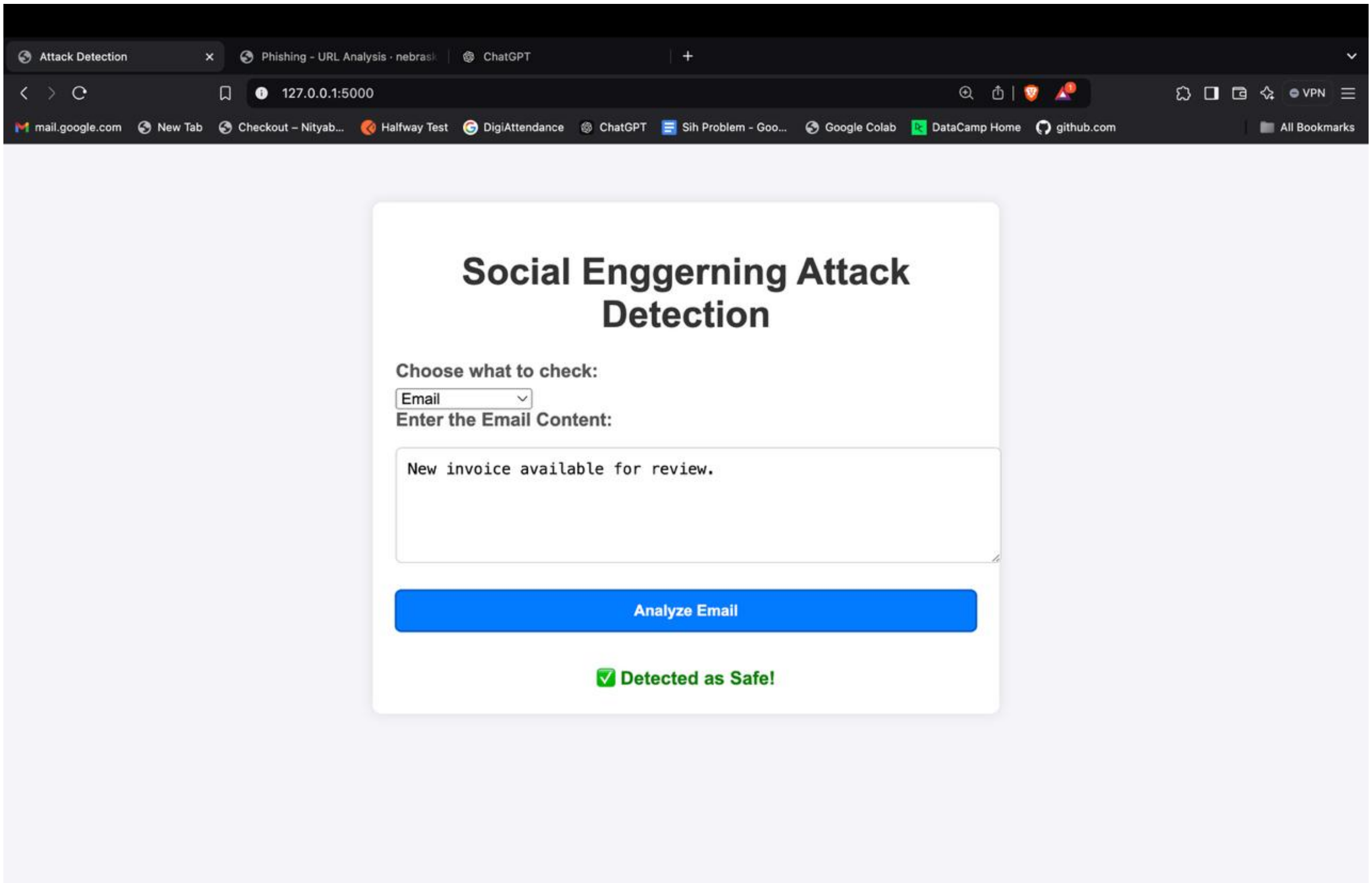
```
# Create a DataFrame from the data
import pandas as pd
df = pd.DataFrame(data, columns=['URL', 'Label'])

# Display the first few rows of the DataFrame
print(df.head())
df_email = pd.DataFrame(email_data, columns=['Email', 'Label'])
df_email
print(df.info())
print(df_email.info())
print(df['Label'].value_counts())
print(df_email['Label'].value_counts())
df['URL'] = df['URL'].str.lower().str.strip()
df_email['Email'] = df_email['Email'].str.lower().str.strip()
df['has_http'] = df['URL'].str.contains('http')
df['has_https'] = df['URL'].str.contains('https')
df['contains_login'] = df['URL'].str.contains('login')
df_email['contains_prize'] = df_email['Email'].str.contains('prize')
df_email['contains_urgent'] = df_email['Email'].str.contains('urgent')
df_email['contains_click'] = df_email['Email'].str.contains('click')
from sklearn.model_selection import train_test_split
# For URLs
X_url = df['URL']
y_url = df['Label']
X_url_train, X_url_test, y_url_train, y_url_test = train_test_split(X_url, y_url, test_size=0.2,
random_state=42)
# For Emails
X_email = df_email['Email']
y_email = df_email['Label']
X_email_train, X_email_test, y_email_train, y_email_test = train_test_split(X_email, y_email, test_size=0.2,
random_state=42)
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
# For URLs
url_pipeline = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', LogisticRegression())
])
url_pipeline.fit(X_url_train, y_url_train)
# For Emails
email_pipeline = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', LogisticRegression())
])
email_pipeline.fit(X_email_train, y_email_train)
```

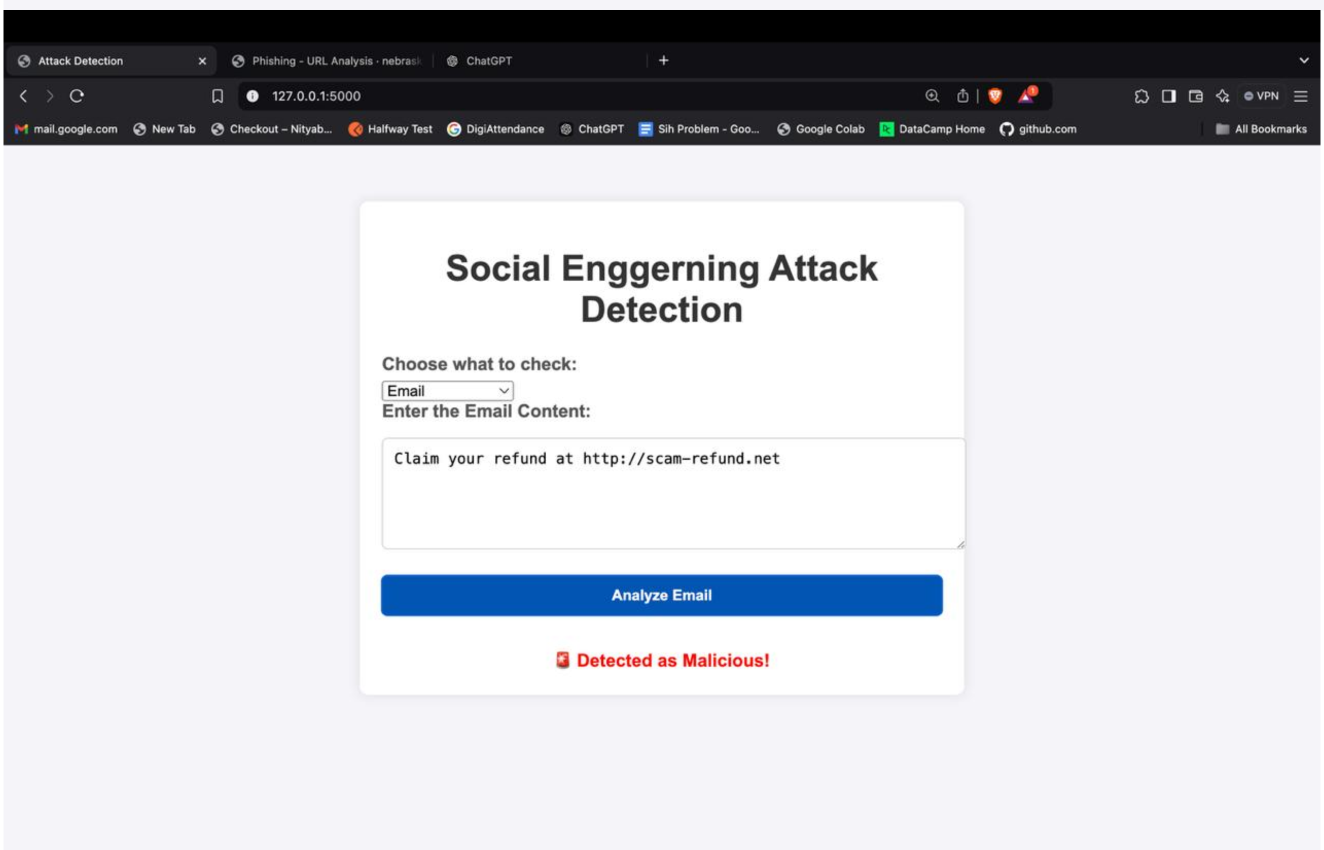
```
from sklearn.metrics import classification_report,accuracy_score

# URL Model Evaluation
y_url_pred = url_pipeline.predict(X_url_test)
print(classification_report(y_url_test, y_url_pred))
print(accuracy_score(y_url_test, y_url_pred)*100)

# Email Model Evaluation
y_email_pred = email_pipeline.predict(X_email_test)
print(classification_report(y_email_test, y_email_pred))
print(accuracy_score(y_email_test, y_email_pred)*100)

import joblib

# Save URL model
joblib.dump(url_pipeline, 'url_model.joblib')

# Save Email model
joblib.dump(email_pipeline, 'email_model.joblib')


from flask import Flask, request, jsonify,render_template
import joblib

# Initialize Flask app
app = Flask(_name_)

# Load models once during initialization
url_model = joblib.load('url_model.joblib')
email_model = joblib.load('email_model.joblib')

@app.route('/')
def home():
    return render_template('index.html')

# Endpoint for URL prediction
@app.route('/predict-url', methods=['POST'])
def predict_url():
    data = request.json
    if 'url' not in data:
        return jsonify({"error": "URL not provided"}), 400

    user_url = data['url']
    prediction = url_model.predict([user_url])[0]
    result = "malicious" if prediction == 1 else "safe"
    return jsonify({"url": user_url, "prediction": result})
```

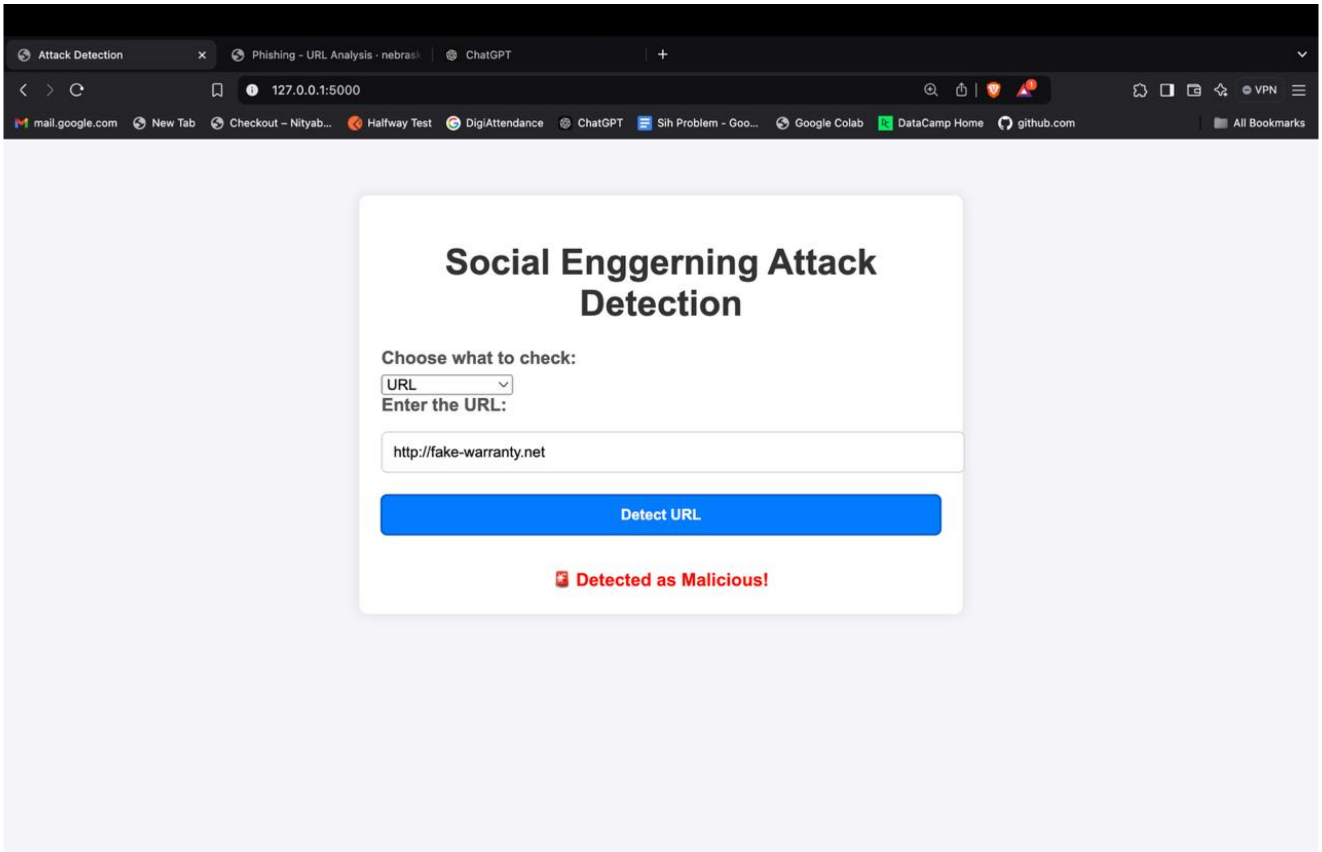
```
# Endpoint for Email prediction
@app.route('/predict-email', methods=['POST'])
def predict_email():
    data = request.json
    if 'email' not in data:
        return jsonify({"error": "Email content not provided"}), 400

    user_email = data['email']
    prediction = email_model.predict([user_email])[0]
    result = "malicious" if prediction == 1 else "safe"
    return jsonify({"email": user_email, "prediction": result})

if _name_ == '_main_':
    app.run(debug=True)
```

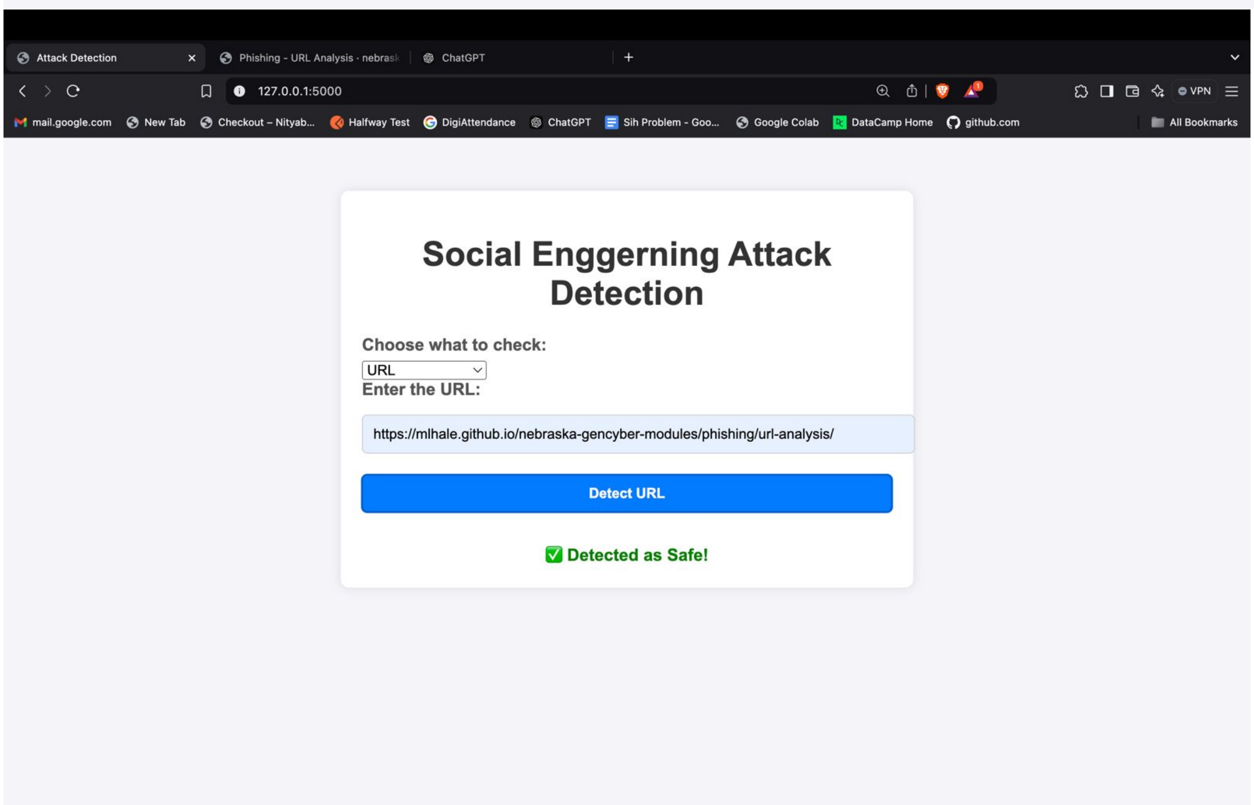
OUTPUT

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00         6

    accuracy                           1.00        19
   macro avg       1.00      1.00      1.00        19
weighted avg       1.00      1.00      1.00        19

100.0
              precision    recall  f1-score   support

           0       1.00      0.93      0.96        14
           1       0.96      1.00      0.98        25

    accuracy                           0.97        39
   macro avg       0.98      0.96      0.97        39
weighted avg       0.98      0.97      0.97        39

97.43589743589743
```


Attack Detection
Phishing - URL Analysis - nebrask
ChatGPT
127.0.0.1:5000
Social Enggerning Attack Detection
Choose what to check:
Email
Enter the Email Content:
New invoice available for review.
Analyze Email
Detected as Safe!


Attack Detection
Phishing - URL Analysis - nebrask
ChatGPT
127.0.0.1:5000
Social Enggerning Attack Detection
Choose what to check:
Email
Enter the Email Content:
Claim your refund at http://scam-refund.net
Analyze Email
Detected as Malicious!


Attack Detection
Phishing - URL Analysis - nebrask
ChatGPT
127.0.0.1:5000
Social Enggerning Attack Detection
Choose what to check:
URL
Enter the URL:
http://fake-warranty.net
Detect URL
Detected as Malicious!


Attack Detection
Phishing - URL Analysis - nebrask
ChatGPT
127.0.0.1:5000
Social Enggerning Attack Detection
Choose what to check:
URL
Enter the URL:
https://mlhale.github.io/nebraska-gencyber-modules/phishing/url-analysis/
Detect URL
✅ Detected as Safe!

CONCLUSION

The social engineering attack detector showcases the effectiveness of leveraging machine learning techniques, specifically Support Vector Machines (SVM), for accurately classifying URLs as phishing or legitimate. By preprocessing the data, extracting key features, and applying a supervised learning approach, the model achieves reliable detection performance. The evaluation metrics, including accuracy, precision, recall, and F1-score, confirm the system's robustness and reliability in identifying phishing threats in real-world scenarios.

This project not only demonstrates the potential of automated solutions in enhancing cybersecurity but also sets the stage for integrating advanced machine learning models into existing security infrastructures. With further refinement, such systems could adapt to evolving phishing tactics, analyze email content for social engineering attempts, and improve overall threat detection mechanisms, providing a scalable and proactive defense against cybercrime.