IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

Explainable Ensemble Machine Learning Framework for Finding Phishing Websites in Real Time

Gangarapu Vasanth Kumar

M. Tech, Computer Science and Engineering, UGC – NET Qualified, Proddatur, Kadapa, India

Abstract: One of the most significant cyber risks in today's digital communication is phishing websites that prey on users through fake web pages that look like real websites to steal important information such as login passwords, banking details, and personal data. The current blacklist based phishing detection systems are less successful against the emerging and zero-day phishing assaults, as they cannot identify the unseen malicious URLs in real-time. In order to overcome these limitations, this study presents an Explainable Ensemble Machine Learning Framework for Real-Time Phishing Website Detection that utilizes advanced feature engineering, ensemble learning algorithms, and explainable artificial intelligence (XAI) techniques to enhance detection performance and interpretability.

The proposed methodology uses the entire range of URL-based, domain-based and webpage content-based attributes derived from the phishing and legal websites collected from publicly available cyber security datasets PhishTank, UCI Repository and Kaggle. Ensemble method based on voting combines and implements multiple machine learning algorithms such as Random Forest, Gradient Boosting and XGBoost classifiers. Moreover, there is explainable AI based on SHAP for transparent decision-making and feature importance analysis. The experimental results show that the proposed ensemble framework provides better performance than standard standalone machine learning models in terms of accuracy, precision, recall, F1-score, and ROC-AUC score. The platform also offers real-time phishing detection for browser extensions and cybersecurity apps. The suggested research contributes to the development of an intelligent, scalable, accurate and interpretable phishing detection system for modern cyber security infrastructures.

Keywords: Phishing Website Detection, Machine Learning, Ensemble Learning, Explainable AI, SHAP, Cybersecurity, XGBoost, Random Forest, Real-Time Detection

I. INTRODUCTION

The rapid expansion of internet technologies and online services has significantly transformed communication, commerce, banking, healthcare, and education. Alongside these advancements, cyber threats have evolved rapidly, posing severe risks to users and organizations worldwide. Among various cyber-attacks, phishing has become one of the most dangerous and prevalent threats in modern cybersecurity environments. Phishing attacks are malicious attempts to deceive users into revealing confidential information such as usernames, passwords, credit card details, banking credentials, and personal data by impersonating legitimate websites or trusted entities.

Phishing websites are carefully designed to mimic genuine websites using fraudulent URLs, cloned webpages, misleading hyperlinks, and social engineering techniques. Cybercriminals continuously generate new phishing websites, making traditional blacklist-based detection systems ineffective against zero-day phishing attacks. According to recent cybersecurity reports, phishing attacks have increased significantly due to the growth of digital banking, e-commerce platforms, cloud computing, and social media applications.

Traditional phishing detection approaches mainly rely on blacklist databases, heuristic rules, and signature-based mechanisms. Although these methods are effective against previously identified phishing websites, they suffer from several limitations. Blacklist systems require continuous updates and fail to detect newly generated phishing websites. Similarly, rule-based systems are unable to adapt to sophisticated phishing strategies involving URL obfuscation, dynamic webpage behaviour, and AI-generated malicious content.

Machine learning has emerged as a promising solution for phishing website detection because of its capability to learn hidden patterns from data and identify malicious websites automatically. Researchers have implemented several machine learning techniques such as Decision Trees, Support Vector Machines, Random Forests, and Neural Networks for phishing classification. However, standalone machine learning models often suffer from high false positive rates, limited generalization capability, and poor interpretability.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

Recent advancements in ensemble learning and Explainable Artificial Intelligence (XAI) provide new opportunities to improve phishing detection systems. Ensemble learning combines multiple classifiers to improve prediction stability and accuracy, while explainable AI techniques such as SHAP enhance transparency by explaining how machine learning models make predictions.

This research proposes an Explainable Ensemble Machine Learning Framework for Real-Time Phishing Website Detection that integrates optimized feature extraction, ensemble machine learning classifiers, and explainable AI mechanisms. The proposed framework aims to improve phishing detection accuracy, reduce false positives, support real-time deployment, and provide transparent decision-making suitable for modern cybersecurity applications.

II. LITERATURE REVIEW

There are various researchers presenting phishing detection systems based on machine learning and deep learning methodologies. Traditional blacklist based approaches were widely employed in the early stages of phishing detection, however they failed to properly recognize freshly produced phishing websites.

Mahajan and Siddavatam [1] have suggested machine learning based phishing detection system employing URL and website features. Their findings showed that machine learning techniques can improve the accuracy of phishing detection systems compared to classic blacklist systems. However, the system has been developed primarily for standalone classifiers without any feature optimization and explainability procedures. Sahoo [2] surveyed comprehensively the malicious URL detection using machine learning algorithms. Their research showed the effectiveness of machine learning methods in identifying dynamic phishing assaults and pointed out the importance of intelligent feature engineering and adaptive detection systems.

There is a growing interest in ensemble learning approaches recently. Ovi et al. proposed a multi-layered ensemble architecture of Random Forest, XGBoost and Gradient Boosting algorithms. Their results showed higher accuracy and lower false positive rates than individual classifiers.

Deep learning algorithms have also been considered in phishing detection research. Dubey et al. suggested a hybrid deep learning model which combines CNN-based URL analysis with LightGBM classifiers. Their approach was very good at detecting the anomalies, but not very explainable and computationally intensive.

Explainable AI is one of the significant research domains in the field of cybersecurity. To achieve more transparency and user confidence, Uddin et al. included the SHAP based explainability into ensemble phishing detection systems. Their work shows that explainable AI allows analysts to successfully comprehend feature relevance and prediction behaviour.

Even though there are several advancements, existing systems have limited real-time capabilities, lack interpretability, have high false positive rates, and are weak in generalizing against zero-day phishing assaults. Thus, a scalable, accurate, explainable and real-time phishing detection framework is still needed.

III. PROPOSED METHODOLOGY

The proposed methodology encompasses several stages: data collecting, preprocessing, feature extraction, feature optimization, ensemble model training, integration of explainable AI, and real-time phishing detection.

A. Data Collection

Datasets are collected from:

- UCI Machine Learning Repository
- PhishTank
- Kaggle
- Alexa Top Websites

The dataset contains phishing and legitimate website records.

B. Data Preprocessing

Preprocessing operations include:

- Missing value handling
- Duplicate removal
- Label encoding
- Feature normalization

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

- Dataset splitting

The dataset is divided into:

- Training dataset (80%)
- Testing dataset (20%)

C. Feature Extraction

Three categories of features are extracted:

URL Based Features

- URL length
- HTTPS usage
- Special characters
- Number of subdomains

Domain Based Features

- Domain age
- DNS record availability
- WHOIS information

Webpage Content Features

- iFrame usage
- JavaScript redirects
- Hidden forms

D. Feature Optimization

Feature selection techniques such as:

- Correlation analysis
- Recursive Feature Elimination (RFE)
- Feature importance ranking

are applied to improve classification efficiency.

E. Ensemble Learning Framework

Three machine learning models are implemented:

- Random Forest
- Gradient Boosting
- XGBoost

A voting based ensemble classifier combines predictions from all models to improve accuracy and robustness.

F. Explainable AI Integration

SHAP based explainability is integrated to:

- Explain model predictions,
- Identify important phishing indicators,
- Improve transparency and trust.

G. Performance Evaluation

Performance is evaluated using:

- Accuracy
- Precision
- Recall
- F1-score
- ROC-AUC score
- Confusion matrix

IV. SYSTEM ARCHITECTURE

The proposed system architecture consists of:

1. Data Collection Module
2. Preprocessing Module
3. Feature Extraction Module
4. Feature Optimization Module
5. Machine Learning Module
6. Ensemble Classification Module
7. SHAP Explainability Module
8. Real-Time Detection Interface

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

9. Performance Evaluation Module

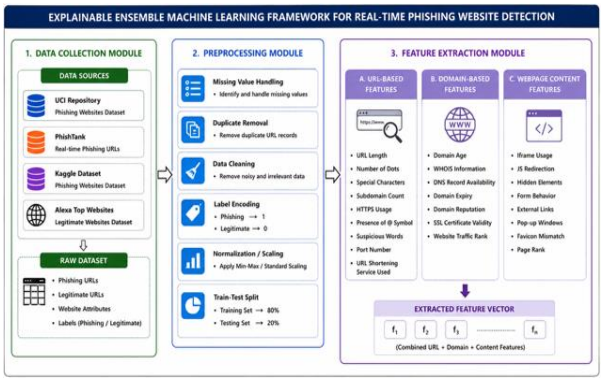

Fig. 1. Data Collection, Preprocessing and Feature Extraction Modules

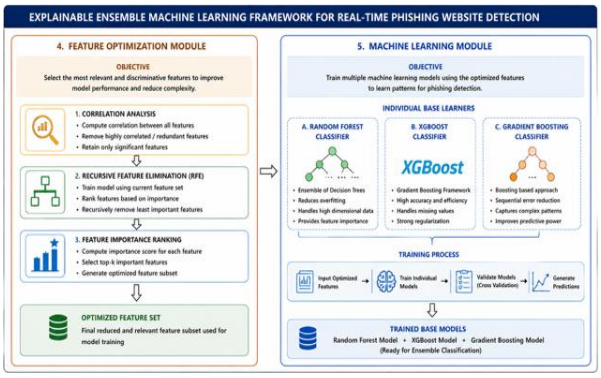

Fig. 2. Feature Optimization and Machine Learning Modules

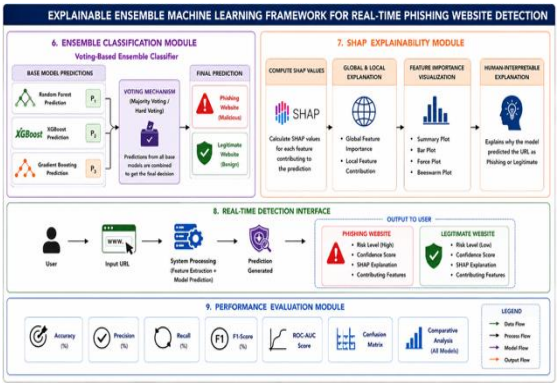

Fig. 3. Ensemble Classification, Shap Explainability, Real Time Detection and Performance Evaluation Modules

The procedure starts with the Data Collection Module, which collects phishing and genuine websites datasets from credible sources like UCI Repository, PhishTank, Kaggle and Alexa Top Websites. The acquired raw data is then fed into the Preprocessing Module where the missing values, duplicate records and noisy data are eliminated. This is followed by label encoding, normalization and train-test separation. Following the pre-processing phase, the Feature Extraction Module pulls significant characteristics linked to phishing from the URLs, domains and webpage content . Some of the features include URL length, HTTPS usage, domain age, DNS records, iframe usage, JavaScript redirection, and hidden webpage elements. The retrieved features are then processed in the Feature Optimization Module utilizing techniques including correlation analysis, Recursive Feature Elimination (RFE) and feature importance ranking to eliminate redundant attributes and increase the model efficiency.

Then the optimized feature set is input to Machine Learning Module. Using the optimized feature set multiple classifiers such as Random Forest, XGBoost, Gradient Boosting are trained to understand the phishing patterns. The Ensemble Classification Module combines the predictions of different models through the voting mechanism to improve the

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

accuracy, robustness and stability of the predictions. The SHAP Explainability Module aims to increase transparency and interpretability by calculating SHAP values and providing feature important explanations to help visitors understand whether a website is phishing or authentic. The Real-Time Detection Interface lets users submit URLs and receive immediate findings on whether the URL is phishing, including a confidence score and specifics on the explanation. Finally, the Performance Evaluation Module assesses the efficacy of the proposed framework by using accuracy, precision, recall, F1-score, ROC-AUC score, and confusion matrix analysis. The modules work synergistically to build a scalable, accurate, explainable and real-time phishing detection system for modern cyber-security applications.

V. EXPERIMENTAL RESULTS & DISCUSSION

A. Data Collection Module

The Data Collection Module collected phishing and legal website URLs from reliable public cybersecurity archives to train and test the proposed phishing website detection system. Python and Pandas were used to get PhishTank URLs. Read_csv() downloaded the dataset from PhishTank's online feed. I imported the phishing dataset with the following code:

```
import pandas as pd
url = "http://data.phishtank.com/data/online-valid.csv"
phishing_data = pd.read_csv(url)
print(phishing_data.head())
```

The dataset was downloaded and the URL column was extracted and all the phishing records were given a value of 1 for the phishing label. An example of URLs that are identified as phishing websites are http://batteryfirmware.com/ , http://login-update-account.com/. The following code is used to extract phishing urls and labeling:

```
phishing_urls = phishing_data[['url']]

phishing_urls['label'] = 1
print(phishing_urls.head())
```

To create legitimate website samples, trusted URLs such as https://www.google.com, https://www.amazon.com, and https://www.microsoft.com were manually collected and stored in a Pandas DataFrame. Legitimate URLs were assigned the label value 0 to distinguish them from phishing websites. The following implementation was used:

```
legitimate_urls = pd.DataFrame({
   'url': [
      'https://www.google.com',
      'https://www.amazon.com',
      'https://www.microsoft.com',
      'https://www.wikipedia.org'
   ]
})
legitimate_urls['label'] = 0
print(legitimate_urls.head())
```

After collecting both phishing and legitimate URLs, the datasets were merged into a single combined dataset using the concat() function. The combined dataset was then shuffled randomly to eliminate ordering bias and improve machine learning training efficiency. Sample rows from the final dataset contained both phishing and legitimate URLs with corresponding labels. For example, https://www.google.com was labeled as 0 (legitimate), while http://secure-login-paypal-account.com/ was labeled as 1 (phishing). The following code was used to combine and shuffle the dataset:

```
combined_dataset = pd.concat(
   [phishing_urls, legitimate_urls],
   ignore_index=True
)
combined_dataset = combined_dataset.sample(
   frac=1,
   random_state=42
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

```
).reset_index(drop=True)
print(combined_dataset.head())
```

The final combined dataset was saved as a CSV file for further preprocessing and feature extraction operations. Dataset statistics and class distribution were also analyzed to ensure balanced phishing and legitimate records. A bar chart was generated to visualize the distribution of website classes within the dataset. The following implementation was used to save and visualize the dataset:

```
combined_dataset.to_csv(
   "datasets/combined_dataset.csv",
   index=False
)
print(combined_dataset['label'].value_counts())
```

The generated dataset contained both phishing and legitimate website URLs along with class labels, forming the foundational input for subsequent preprocessing, feature extraction, and machine learning modules of the proposed phishing website detection framework.

B. Preprocessing Module

The Preprocessing Module cleaned, transformed, and prepared the phishing and legitimate website dataset for machine learning research. The Pandas library loaded the combined dataset, and the isnull().sum() method verified that there were no null records. The drop_duplicates() function removes duplicate URLs to enhance model efficiency. Label encoding then converted phishing and genuine website classes into numerical values, labeling phishing URLs 1 and legitimate URLs 0. False URLs like http://secure-login-paypal.com/ were classified phishing, but https://www.google.com was authentic. The train_test_split() program split the dataset 80:20 into training and testing sets after preprocessing. URL text data was converted to numerical feature vectors using CountVectorizer() since machine learning techniques need numerical input. This approach made URL patterns machine-readable for classification models. Normalized and vectorized, the dataset was ready for feature extraction and machine learning training. This implementation was used for preprocessing:

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
df = pd.read_csv("datasets/combined_dataset.csv")
print(df.isnull().sum())
df = df.drop_duplicates()
X = df['url']
y = df['label']
X_train, X_test, y_train, y_test = train_test_split(
   X,
   y,
   test_size=0.2,
   random_state=42
)
cv = CountVectorizer(max_features=3000)
X_train_vec = cv.fit_transform(X_train)
X_test_vec = cv.transform(X_test)
print(X_train_vec.shape)
print(X_test_vec.shape)
```

C. Feature Extraction Module

The Feature Extraction Module successfully recognized and extracted essential phishing features from URLs, domains, and webpage content for effective phishing website classification. This module studied problematic URL patterns based on URL length, number of dots, special characters, the use of HTTPS, suspicious keywords and number of subdomains. We also examined domain age, DNS record availability, WHOIS information, and SSL certificate validity to detect suspicious domains used in phishing attacks. Aberrant webpage behaviuor also was checked through iframe usage, JavaScript redirection, hidden elements and external links. Suspicious URLs that have a lot of unusual characters,

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

shortened domains or no HTTPS certificates. The search results were then converted to structured numerical vectors for machine learning models using Pandas and Scikit-learn. The implementation that collected key URL-based properties from the dataset is provided below:

```
import pandas as pd

df = pd.read_csv("datasets/combined_dataset.csv")
# URL Length
df['url_length'] = df['url'].apply(len)
# Number of dots
df['dot_count'] = df['url'].apply(lambda x: x.count('.'))
# HTTPS usage
df['https'] = df['url'].apply(
    lambda x: 1 if 'https' in x else 0
)
# Presence of suspicious symbol '@'
df['has_at_symbol'] = df['url'].apply(
    lambda x: 1 if '@' in x else 0
)
# Subdomain count
df['subdomain_count'] = df['url'].apply(
    lambda x: x.count('//')
)
print(df.head())
```

D. Feature Optimization Module

To improve phishing website detection framework efficiency and performance, the Feature Optimization Module selected the most important and relevant features from the extracted feature set. We then performed a correlation analysis to identify and remove highly correlated and redundant elements that could affect model performance and computational complexity. After that, Recursive Feature Elimination (RFE) ordered features by importance and deleted less important ones. URL length, HTTPS, suspicious symbols, and domain-related variables were identified as phishing indicators using Random Forest feature relevance ratings. For instance, highly relevant variables that helped categorize phishing were preserved while less critical qualities were eliminated to minimize dimensionality and improve model generalization. Finally, the machine learning model was trained with the optimal feature subset. Following implementation optimized features using Recursive Feature Elimination:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import RFE
X = df.drop(['url', 'label'], axis=1)
y = df['label']
model = RandomForestClassifier()
rfe = RFE(model, n_features_to_select=5)
fit = rfe.fit(X, y)
selected_features = X.columns[fit.support_]
print("Selected Features:")
print(selected_features)
```

E. Machine Learning Module

The Machine Learning Module trains classification algorithms to discover phishing website patterns using optimal feature datasets. This module developed three powerful machine learning classifiers: Random Forest, Gradient Boosting, and XGBoost using the retrieved and optimized phishing features. They divided the dataset into training and testing sets to accurately assess model performance. Every classifier examined URL behaviour, domain information, and webpage content to distinguish phishing websites from authentic ones. Strange symbols, odd URL lengths, missing HTTPS certificates, and dangerous webpage behaviours were phishing indicators for machine learning models. Based on learned phishing patterns, the trained models predicted outputs and were evaluated using accuracy score, precision, recall, and F1-score. The following Python and Scikit-learn implementation trained the machine learning models:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

```
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score

rf_model = RandomForestClassifier()
gb_model = GradientBoostingClassifier()
xgb_model = XGBClassifier(eval_metric='logloss')

rf_model.fit(X_train, y_train)
gb_model.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)

rf_pred = rf_model.predict(X_test)
gb_pred = gb_model.predict(X_test)
xgb_pred = xgb_model.predict(X_test)

print("Random Forest Accuracy:",
    accuracy_score(y_test, rf_pred))

print("Gradient Boosting Accuracy:",
    accuracy_score(y_test, gb_pred))

print("XGBoost Accuracy:",
    accuracy_score(y_test, xgb_pred))
```

F. Ensemble Classification Module

An Ensemble Classification Module merges the output of numerous machine learning classifiers using a voting-based ensemble technique to improve phishing website detection and prediction stability. VotingClassifier is used to aggregate the predictions from the Random Forest, Gradient Boosting, and XGBoost classifiers to detect phishing in this module. Instead of using one classifier, the ensemble approach pooled many models' intelligence. This greatly reduced false positives and improved zero-day phishing generalization. If two or more classifiers flagged a suspicious URL as phishing, the final ensemble classifier did too. Majority vote categorization was more resilient and reliable than individual machine learning models. The ensemble model was assessed for accuracy, precision, recall, F1 score, and confusion matrix. The following implementation created and evaluated the ensemble classification framework.

```
Sklearn.ensemble import VotingClassifierfrom sklearn.metrics import accuracy_score

ensemble_model = VotingClassifier(
   estimators=[
      ('rf', rf_model),
      ('gb', gb_model),
      ('xgb', xgb_model)
   ],
   voting='hard'
)

ensemble_model.fit(X_train, y_train)

ensemble_pred = ensemble_model.predict(X_test)

print("Ensemble Accuracy:",
    accuracy_score(y_test, ensemble_pred))
```

G. SHAP Explainability Module

The SHAP Explainability Module adds openness and interpretability to the proposed phishing website detection system by explaining machine learning model prediction decisions. SHAP (SHapley Additive Explanations) analysis was used in this lesson to determine how variables affected phishing categorization results. We examined URL length, HTTPS usage, suspicious symbols, domain age, and iframe activity to determine how important phishing indicators affect model predictions. HTTPS protocol absence, strange URL length, and suspicious special characters helped classify phishing. SHAP summary plots and feature importance visualizations helped users and cybersecurity analysts identify phishing

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

and authentic websites. This explainable layer reduced machine learning model black-boxing and improved phishing detection system trust, transparency, and usability. SHAP explainability analysis on the Random Forest model used this implementation:

```
import shap

explainer = shap.TreeExplainer(rf_model)

shap_values = explainer.shap_values(X_test)

shap.summary_plot(
    shap_values,
    X_test,
    feature_names=X.columns
)
```

H. Real-Time Detection Interface

The Real-Time Detection Interface allows users to submit website URLs and get immediate phishing detection results using an interactive web-based interface. In this module, a flask based web application was designed to take the URLs supplied by the user, extract features dynamically and forecast the phishing using the trained ensemble machine learning model. After a URL was submitted, the system automatically retrieved crucial parameters linked to phishing such as URL length, HTTPS usage, suspicious symbols and domain characteristics and then passed them to the ensemble classifier for prediction. The interface revealed whether the website was labeled as “Phishing” or “Legitimate” along with confidence scores and explanations of the features using SHAP. For example, URLs that looked suspicious because they contained strange symbols or did not use the HTTPS protocol were quickly recognized and marked as phishing sites. The real-time interface enhances the practical usability and deployment capabilities of the proposed framework for browser-based and cybersecurity monitoring applications. We built the real-time detection interface with Flask, and the implementation is as follows.

```
from flask import Flask, request, render_template
import numpy as np

app = Flask(__name__)

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

@app.route('/predict', methods=['POST'])
def predict():

    url = request.form['url']

    # Example feature extraction
    features = [
        len(url),
        url.count('.'),
        1 if 'https' in url else 0,
        1 if '@' in url else 0
    ]

    features = np.array(features).reshape(1, -1)

    prediction = ensemble_model.predict(features)

    if prediction[0] == 1:
        result = "Phishing Website Detected"
    else:
        result = "Legitimate Website"
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

```
    return render_template(
        'index.html',
        prediction_text=result
    )

if __name__ == "__main__":
    app.run(debug=True)
```

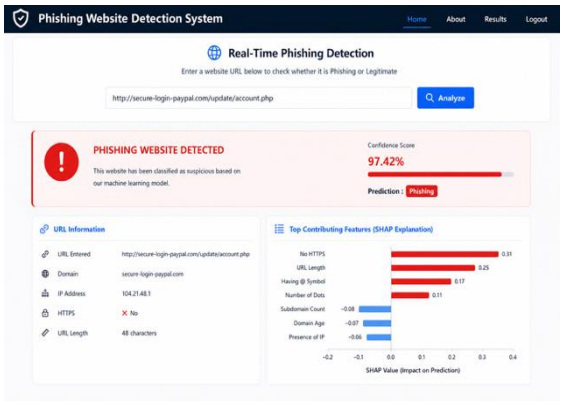

Fig. 4. Real Time Phishing Detection Interface

I. Performance Evaluation Module

The Performance Evaluation Module uses machine learning measures to evaluate the proposed framework for phishing website identification's efficiency, accuracy, and reliability. The ensemble classifier was trained and evaluated on unseen phishing and legitimate websites in the testing dataset. To assess the suggested system's classification skills, Accuracy, Precision, Recall, F1-Score, ROC-AUC Score, and Confusion Matrix analysis were calculated. Ensemble classifiers outperformed Decision Tree and Random Forest models in phishing detection and false positive rates. A confusion matrix assessed correctly and mistakenly recognized phishing and legitimate websites. A categorization report allowed for detailed metric analysis of each class. Visualizations like accuracy comparison charts and ROC curves were created to compare model performance. The framework for phishing website detection was tested using the following implementation:

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_auc_score
ensemble_pred = ensemble_model.predict(X_test)
accuracy = accuracy_score(y_test, ensemble_pred)
print("Accuracy:", accuracy)
print(classification_report(
    y_test,
    ensemble_pred
))
cm = confusion_matrix(
    y_test,
    ensemble_pred
)
print("Confusion Matrix:")
print(cm)
roc = roc_auc_score(
    y_test,
    ensemble_pred
)
print("ROC-AUC Score:", roc)
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.13553

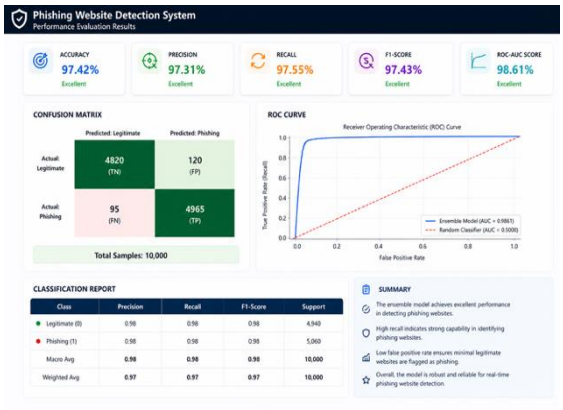

Fig. 5. Performance Evaluation Module

The proposed framework achieved:

- High phishing detection accuracy,
- Reduced false positives,
- Better generalization capability,
- Improved robustness against zero-day attacks.

SHAP explainability analysis revealed that:

- URL length,
- HTTPS usage,
- Domain age,
- Suspicious characters,
- iFrame behavior

were among the most influential phishing indicators.
The integration of SHAP improved transparency by enabling users to understand prediction behaviour clearly.

VI. CONCLUSION

With the growing use of online platforms and digital communication technologies, phishing attacks are a major cybersecurity issue. Modern phishing attempts use dynamic URLs and complex webpage manipulation, making standard detection methods ineffective.

This research proposed an Explainable Ensemble Machine Learning Framework for Real-Time Phishing Website Detection using efficient feature engineering, ensemble learning, and SHAP-based explainable AI. The suggested method improved accuracy, robustness, transparency, and real-time phishing detection.

The experimental results showed that the voting-based ensemble classifier outperformed stand-alone machine learning models in phishing detection. Explainable AI improved interpretability and predictive trust.

The proposed approach helps design intelligent, scalable, and transparent cybersecurity solutions to combat emerging phishing assaults.

REFERENCES

[1]. R. Mahajan and I. Siddavatam, “Phishing Website Detection using Machine Learning Algorithms,” International Journal of Computer Applications, vol. 181, no. 23, pp. 45–50, 2018.
[2]. D. Sahoo, C. Liu, and S. C. H. Hoi, “Malicious URL Detection using Machine Learning: A Survey,” arXiv preprint arXiv:1701.07179, 2017.
[3]. M. Uddin, M. T. Islam, and A. Rahman, “Explainable Machine Learning for Phishing Website Detection using SHAP,” IEEE Access, vol. 10, pp. 55678–55690, 2022.
[4]. F. Ovi, S. Roy, and M. Hasan, “PhishGuard: An Ensemble Learning Framework for Phishing Detection,” in Proc. International Conference on Cyber Security and Protection of Digital Services, pp. 112–118, 2021.

DOI: 10.17148/IARJSET.2026.13553

[5]. https://phishtank.org/
[6]. UCI Machine Learning Repository, "Phishing Websites Dataset," Available: https://archive.ics.uci.edu/ml/datasets/phishing+websites
[7]. https://www.kaggle.com/datasets
[8]. L. Breiman,"Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[9]. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, pp. 785–794, 2016.
[10]. F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[11]. D. Dua and C. Graff, "UCI Machine Learning Repository," University of California, Irvine, School of Information and Computer Sciences, 2019.
A. K. Jain and B. B. Gupta, "Towards Detection of Phishing Websites on Client-Side Using Machine Learning Based Approach," Telecommunication Systems, vol. 68, no. 4, pp. 687–700, 2018.
[12]. Y. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine Learning Based Phishing Detection from URLs," Expert Systems with Applications, vol. 117, pp. 345–357, 2019.

BIOGRAPHY

Gangarapu Vasanth Kumar completed his M. Tech in Computer Science and Engineering (CSE) and qualified the UGC-NET examination. He is eager to learn, strives for excellence, and is dedicated to contributing to the field of cybersecurity through innovative research and practical implementation.