Combined Feature-Based and Content-Based URL Analysis for Improved Phishing Detection

Malicious URLs are web addresses designed to deceive users and distribute harmful content. Understanding their structure and characteristics is crucial for protecting against online threats.

Why this topic?

Why We Chose This Project

1. **Growing Cybersecurity Threats**:
    - Phishing attacks are becoming increasingly sophisticated and prevalent, posing significant risks to individuals and organizations worldwide. With the rapid growth of internet usage, the need for effective phishing detection mechanisms has never been more critical.
2. **Financial and Personal Impacts**:
    - Phishing can lead to severe financial losses and compromise sensitive personal information. Victims of phishing attacks may suffer from identity theft, unauthorized financial transactions, and long-term impacts on credit scores and personal privacy.
3. **Need for Improved Detection Methods**:
    - Advanced machine learning methods provide a promising avenue for improving detection accuracy and reducing false positives and false negatives.

Understanding URL Structure

Scheme

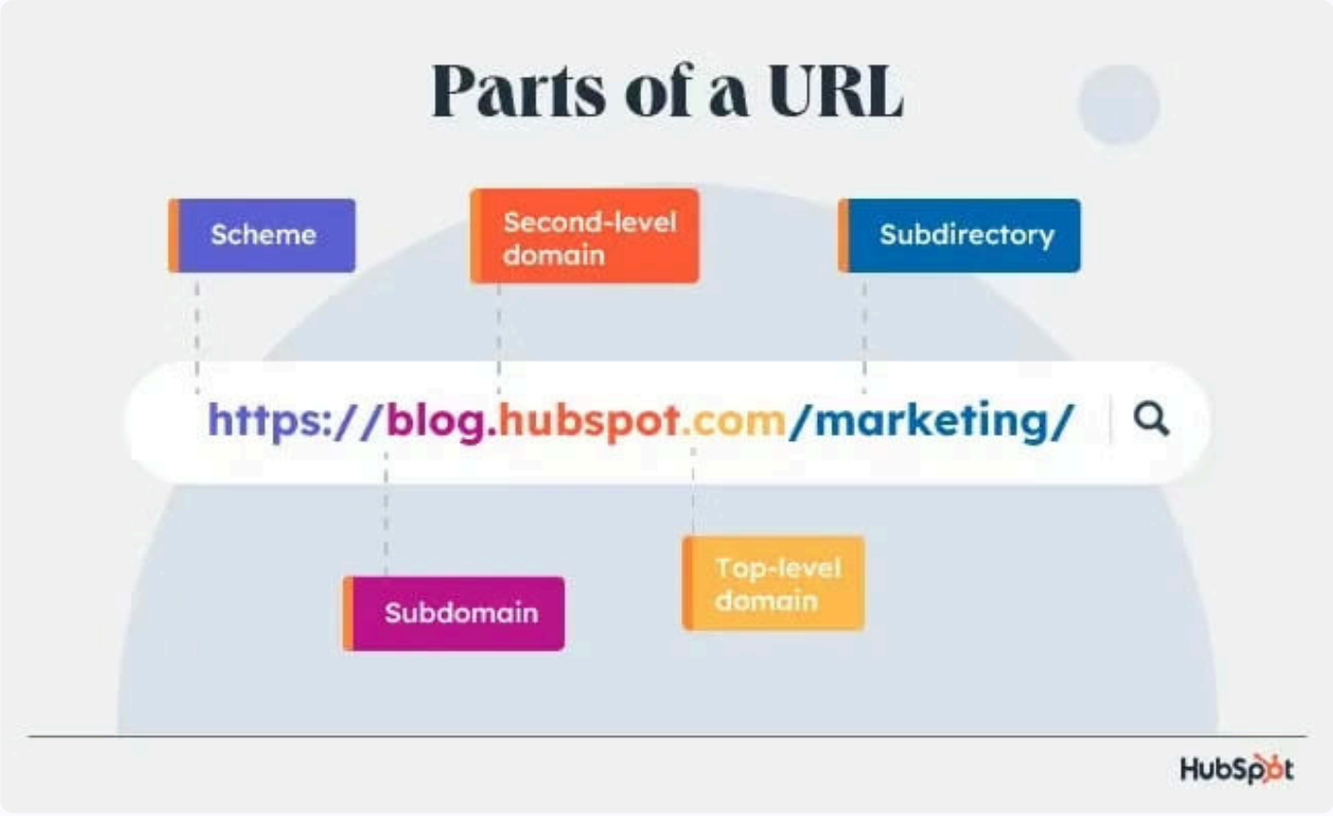
The protocol used, such as HTTP or HTTPS.

Domain

The unique web address associated with a website.

Path

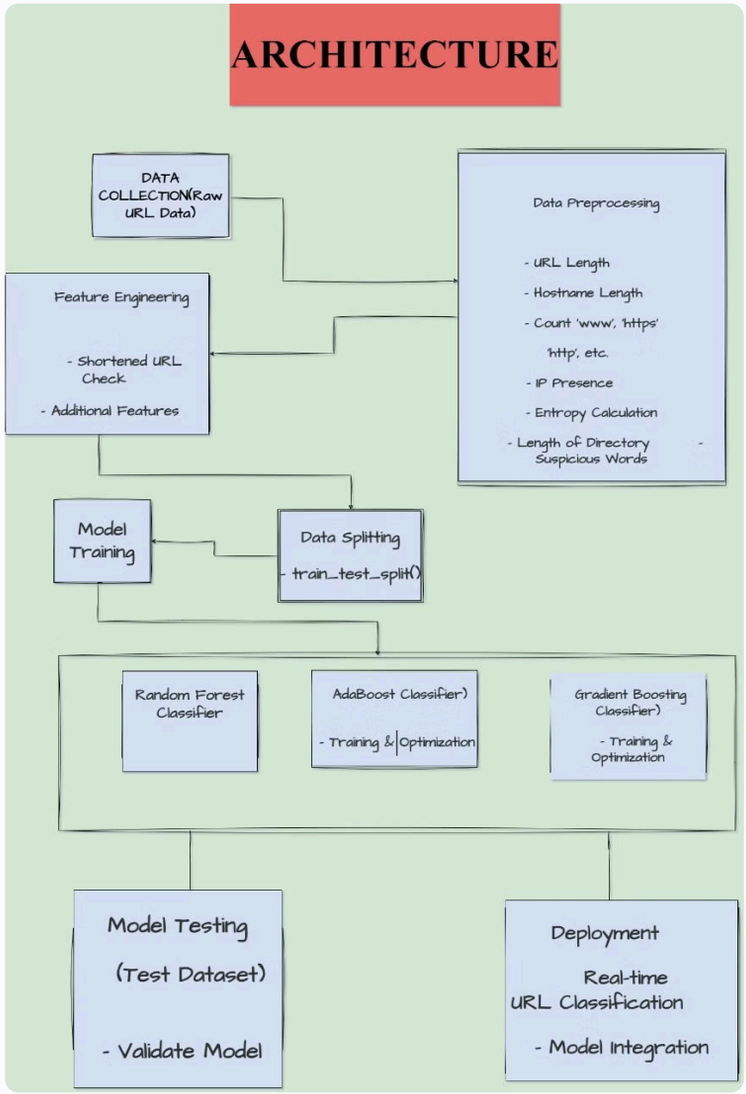
The specific location of a resource within a website.



Techniques

Type of Feature	Description	Pros and Cons
Lexical features [13]	URL is generally a string. Different features are generated based on the URL string.	• Easy and Safer to use. • Lexical feature alone is not enough to detect the malicious URL because, most of the malicious URLs are looking like benign URLs. • Assigning proper weightage to features requires in depth domain knowledge.
Host or Web Server based features [13]	Host feature includes, location of web server, ip address, DNS information etc	• Safer to process. Most of malicious URLs are short living in general. • Additional processing time is required to obtain the host related information.
Content based features. [14]	Web page content are analyzed, and features are extracted. Basically, it includes javascript features, HTML features, page size, visual similarity features [15].	• Mostly phishing websites are dynamic in nature. So, to identify malicious web page by analysing its content is most reliable method • Extracting different features from a page requires more processing time and not safer to process the malicious content. It may affect the processing system also.
Reputation based features [13]	Reputation based feature uses URL to check its merits with third party servers Reputation Features includes page rank, link popularity score, social reputation features [13]	• Safer to process and plays a vital role in detecting malicious URLs. • Additional processing time is required to access the third-party server.

Architecture



What features makes it Malicious?

1

URL length(url):

Purpose: Measures the length of the entire URL string.

Usage: Malicious URLs might be excessively long (e.g., due to appended parameters or directory structures) to hide the true destination or to obfuscate.

2

hostname_len(url):

Purpose: Measures the length of the hostname part of the URL.

Usage: Unusually long or short hostnames can indicate suspicious activity. For example, a very long hostname might be an attempt to make a phishing URL look more legitimate (e.g., `secure-bank-login-verification-account-confirmation.com`).

3

count_www(url):

Purpose: Counts occurrences of 'www' in the URL.

Usage: While 'www' itself isn't inherently suspicious, unusual counts (e.g., multiple instances or absence where expected) might indicate attempts to spoof a legitimate domain.

4

count_https(url):

Purpose: Counts occurrences of 'https' in the URL.

Usage: Legitimate URLs typically use HTTPS to secure connections. Malicious URLs might mix protocols (e.g., redirect from HTTPS to HTTP) or use non-standard HTTPS patterns (e.g., `https//secure-site.com`).

5

count_http(url):

Purpose: Counts occurrences of 'http' in the URL.

Usage: Malicious URLs might use HTTP instead of HTTPS to avoid encryption, or mix protocols to deceive users about the security of the connection.

6

count_dot(url):

Purpose: Counts the number of '.' (dots) in the URL.

Usage: URLs with an unusually high number of dots (e.g., `http://123.456.789.123/login.php`) might indicate the use of IP addresses or complex subdomains to deceive users.

1 count_per(url):

Purpose: Counts occurrences of '%' (percent signs) in the URL.

Usage: Percent encoding (%XX) can be used to obfuscate malicious payloads or bypass URL filtering mechanisms.

2 count_ques(url):

Purpose: Counts occurrences of '?' (question marks) in the URL.

Usage: Used in query strings, excessive use of '?' or unusual placement might indicate attempts to pass parameters to malicious scripts or redirect users.

3 count_hyphen(url):

Purpose: Counts occurrences of '-' (hyphens) in the URL.

Usage: Hyphens can be used in domain names to mimic legitimate URLs or create subdomains (e.g., `secure-login-example.com` vs. `secure-login-example.com`).

4 having_ip_address(url):

Purpose: A binary flag indicating if the URL contains an IP address instead of a domain name.

Usage: URLs using IP addresses can be indicative of phishing attempts or attempts to bypass domain blacklists by avoiding domain names.

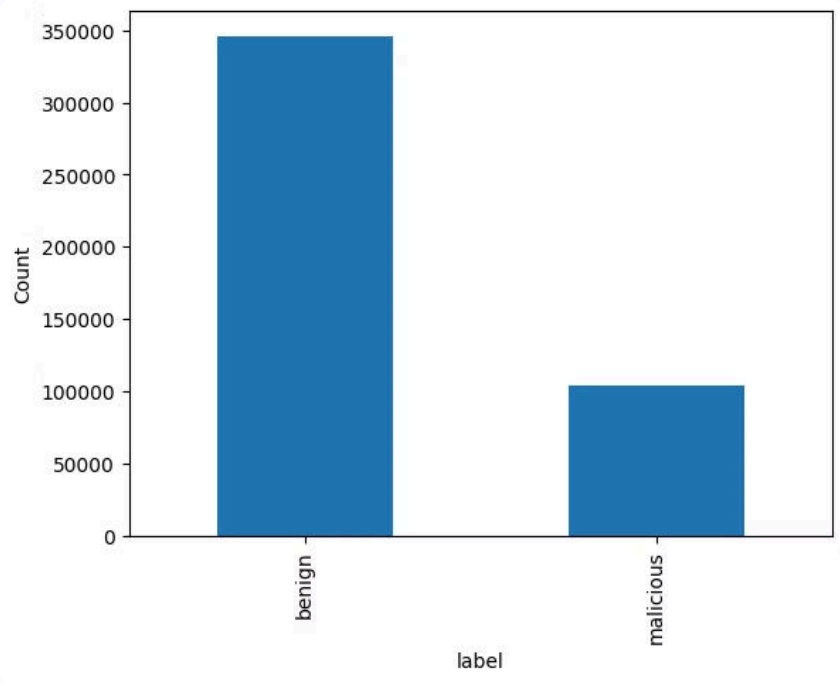
5 calculate_entropy(url):

Purpose: Calculates the entropy (information entropy) of the URL.

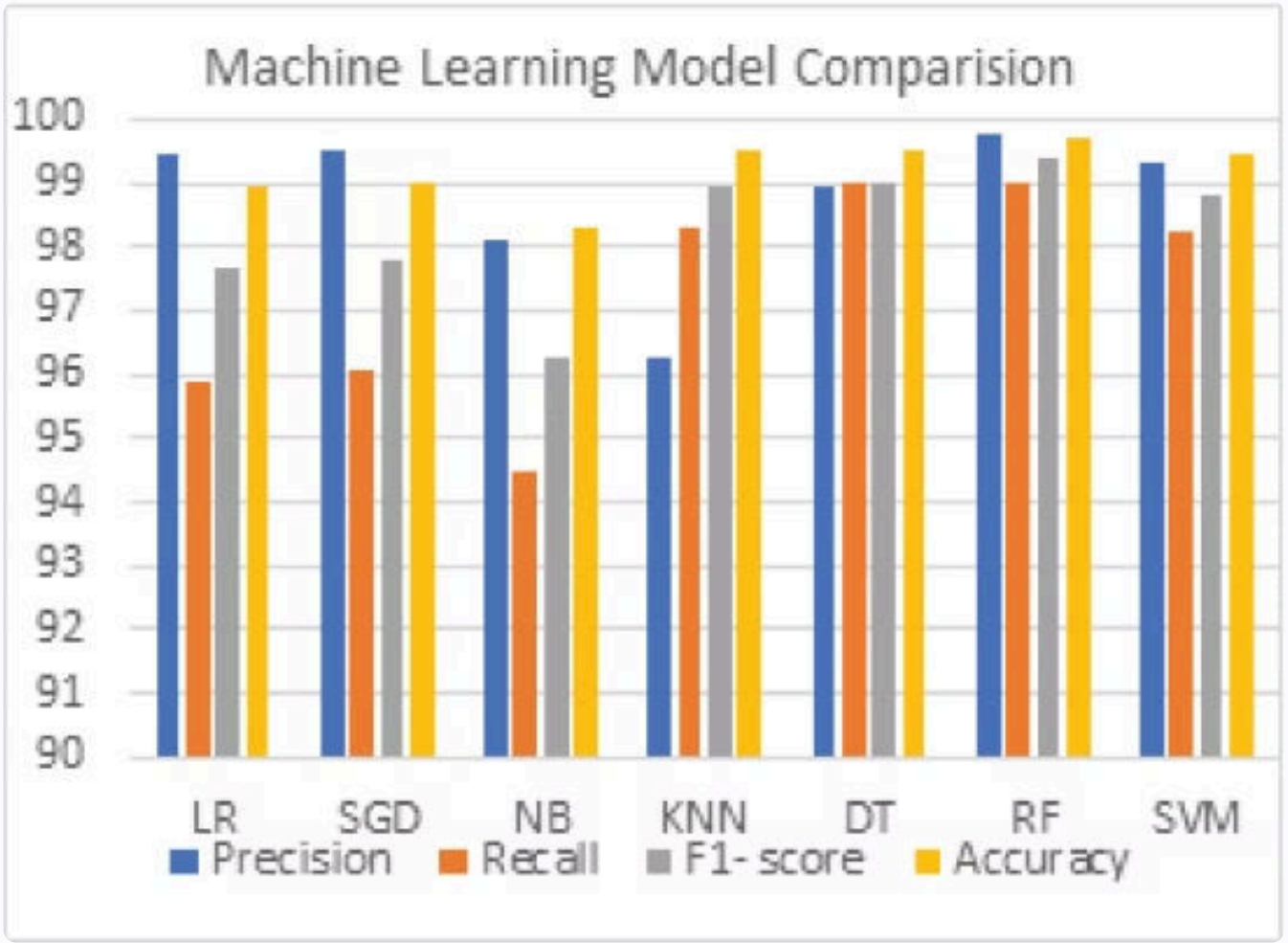
Usage: High entropy values can indicate randomness or complexity in the URL structure, which might be used to obfuscate malicious intent.

dataset

Dataset has been taken from Kaggle that contains records of malicious and benign URLs which can be used analysis or building classifiers.



RandomForest



HyperParameter Tuning

a `RandomForestClassifier` is initialized with specified hyperparameters (`n_estimators`, `max_depth`, `min_samples_split`). Randomized search cross-validation (`RandomizedSearchCV`) is then employed to explore different combinations of these hyperparameters, aiming to maximize model accuracy on unseen data. The best performing set of hyperparameters is identified based on cross-validation results

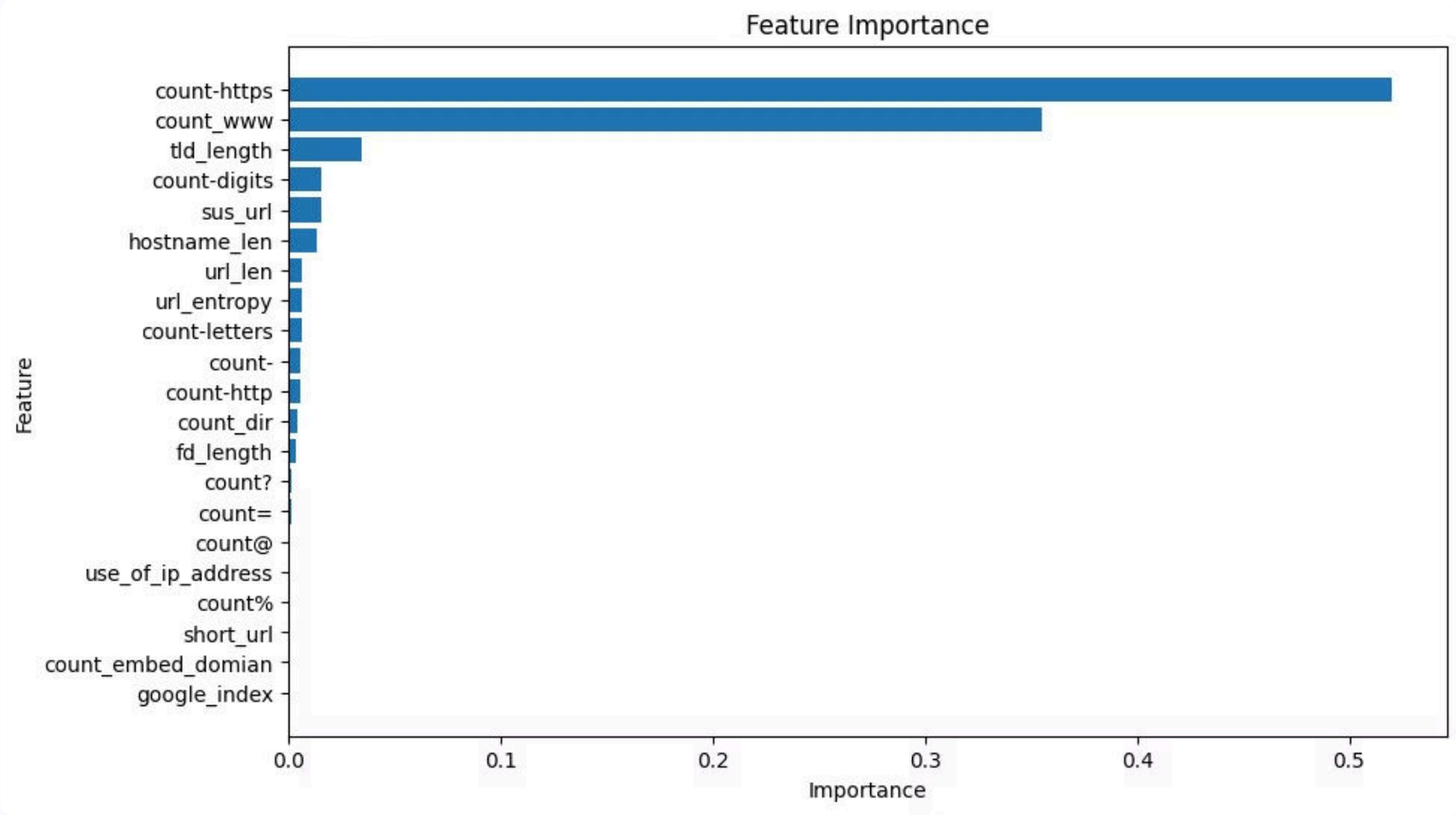
```
Best parameters found:  {'n_estimators': 200, 'min_samples_split': 2, 'max_depth': 20}
Best cross-validation accuracy:  0.9974870883545288
Test set accuracy:  0.9972122262206229
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     69148
           1       1.00      0.99      0.99     20888

    accuracy                           1.00     90036
   macro avg       1.00      1.00      1.00     90036
weighted avg       1.00      1.00      1.00     90036
```

Feature Importance

`random_search.best_estimator_.feature_importances_` provides a ranked list of feature importances for the best performing random forest classifier identified through randomized search cross-validation. Each value in the array represents the relative importance of a feature (e.g., specific words or tokens in TF-IDF) in predicting whether a URL is legitimate or phishing.



Ensembling

Ensemble learning combines multiple machine learning models to improve the overall performance of the system. By leveraging the strengths of different models, ensemble methods can reduce the variance and bias inherent in individual models, leading to more accurate and robust predictions.

1. RandomForestClassifier
2. AdaBoostClassifier
3. GradientBoostClassifier

Why those Models?

Model	Accuracy
KNeighborsClassifier	0.972
RandomForest	0.983
DecisionTree	0.973
GaussianNB	0.830

The accuracies of the AdaBoost, CatBoost, and Gradient Boosting Classifier models were 99%, 98%, and 96%, respectively.

Boosting vs Bagging

Feature	Boosting	Bagging
Basic Idea	Sequentially train models to correct errors of previous models	Train multiple models independently and combine their predictions
Model Dependency	Models are dependent as later models focus on correcting errors of earlier ones	Models are independent, trained on different subsets of data
Performance	Often achieves higher accuracy, especially in difficult cases	Reduces overfitting and improves stability, but may sacrifice some accuracy

Accuracy of Ensemble

```
Accuracy: 0.9969345595095295
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     69148
           1       1.00      0.99      0.99     20888

    accuracy                           1.00     90036
   macro avg       1.00      1.00      1.00     90036
weighted avg       1.00      1.00      1.00     90036
```

Content-based analysis

Since content-based analysis looks directly at the content of the URL (such as the webpage HTML and text), it can provide real-time assessments of URLs as they are accessed. This is crucial for rapidly identifying malicious or inappropriate content.

```
def classify_url(url):
    webpage_content = fetch_webpage_content(url)
    if webpage_content:
        # Use BeautifulSoup to parse HTML content
        soup = BeautifulSoup(webpage_content, 'html.parser')
        # Extract text content from webpage
        text_content = soup.get_text()
        # Train a simple classifier (e.g., Multinomial Naive Bayes) using TF-IDF vectors of text content
        vectorizer = TfidfVectorizer()
        X = vectorizer.fit_transform([text_content])
        # Train a simple classifier (e.g., Multinomial Naive Bayes)
        clf = MultinomialNB()
        clf.fit(X, [0])  # Dummy label since we only have one sample
        # Predict class (0: legitimate, 1: phishing)
        prediction = clf.predict(X)[0]
        return "Legitimate" if prediction == 0 else "Phishing"
    else:
        return "Phishing"

# Streamlit UI
```

Hybrid Model

- This method integrates two distinct approaches to effectively classify websites as legitimate or potentially malicious (spam). Initially, all websites undergo feature-based analysis.
- Websites predicted as legitimate proceed directly to the content-based analysis phase, where the textual content of the webpage is thoroughly examined. This includes parsing the HTML, extracting visible text, and applying machine learning techniques like TF-IDF vectorization and classification algorithms (e.g., Multinomial Naive Bayes) to further verify their legitimacy.

```
url = st.text_input("Enter URL:", "")
if url:
    result = predict_url_spam(url)
    if(result==0)
    {
        classification = classify_url(url)
    }
    else
    {
        classification = "Phishing"
    }
    st.write(f"Classification for {url}: {classification}")

    if classification == "Phishing":
        st.error("This website might be a phishing website. Be cautious!")
    else:
        st.success("This website seems legitimate.")
```

Conclusion

- Our integrated approach combines ensemble methods like Random Forest, AdaBoost, and Gradient Boosting for feature-based analysis, achieving a remarkable 99.69% accuracy in distinguishing between legitimate and potentially malicious websites based on domain age, URL structure, and server reputation.
- This robust initial classification is further validated through content-based analysis using TF-IDF vectorization and Multinomial Naive Bayes, ensuring deep scrutiny of textual content to enhance accuracy and minimize false positives.
- Together, these methodologies strengthen cybersecurity defenses by efficiently identifying threats and safeguarding users against malicious online activities.

Real-Time URL Monitoring and Alerting

Continuous Scanning	Analyzing URLs in real-time to identify potential threats.
Automated Alerts	Notifying security teams about detected malicious URLs.
Incident Response	Initiating appropriate actions to mitigate and contain the threat.

Best Practices for Malicious URL Prevention

User Education

Training employees to recognize and avoid suspicious URLs.

Multilayered Defense

Implementing a combination of security measures for comprehensive protection.

Continuous Monitoring

Regularly reviewing and updating security controls to stay ahead of evolving threats.

References

https://ieeexplore-ieee-org-uceou.knimbus.com/document/9777221

https://ieeexplore-ieee-org-uceou.knimbus.com/document/9777221

PDF file