International Research Journal of Education and Technology
Peer Reviewed Journal
ISSN 2581-7795

Spam Detection Using Machine Learning

Javed Ahamed B.M.S[1], Mayil Samy S[2], Naveen Pandian P[2],Praveen Kumar P M[2],

Vijayragu M[2]

[1]Assistant Professor, [2]UG Students – Final Year, Department of Computer Science and Engineering,Nandha College of Technology, Perundurai - 638052, Tamilnadu, India

Abstract

Now-a-days we get the message which contains "content" and "link."With the rapid growth of internet users, people are using them for illegal and unethical conducts, phishing, and fraud.This project will identify those spam messages and the phishing URLs on the messages by using techniques of machine learning. Apply the algorithm on our datasets and best algorithm is selected for the spam detection having the best precision and accuracy.

Keywords: Machine learning, spam message, spam mail, spam content, phishing attack, phishing url, web phishing, legitimate url, ham message, url detection, K-Nearest neighbours' classifier, logistic regression.

1. Introduction

Spam messages are almost always commercial and driven by a financial motive.They try to "promote and sell questionable goods",make false claims and deceive recipients into believing something that's not true.Phishing URLs allowing the attacker to observe everything while the victim is navigatingthe site, and transverse any additional security boundaries with the victim.Phishing awareness has become important at home and at the work place. For instance, from 2019 to 2022, these attacks have increased from 72% to 86% among businesses.In Modern world, Many Technologies are built and create to crack such Strategy. In this project, we must discuss and develop a detection application with the help of the Domain "Machine Learning" by using "Flask Framework" in Python and aware from such type of messages and sites.The method is usedtodetect phishing websites by updating the blacklisted URLs and the Internet Protocol (IP) to the antivirus database which is also known as "blacklist"method. To overcome the drawbacks of blacklist and heuristics-based method, many security researchers now focused on machine learning techniques. Machine learning technology consists of a many algorithms which requires past data to decide or prediction on future data. Using this technique, algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero-hour phishing websites. A Zero-day threat sometimes called a zero hour threat. It is one that has not been seen before and does not match any known malware signatures. This makes it impossible to detect by traditional signature-matching solutions.

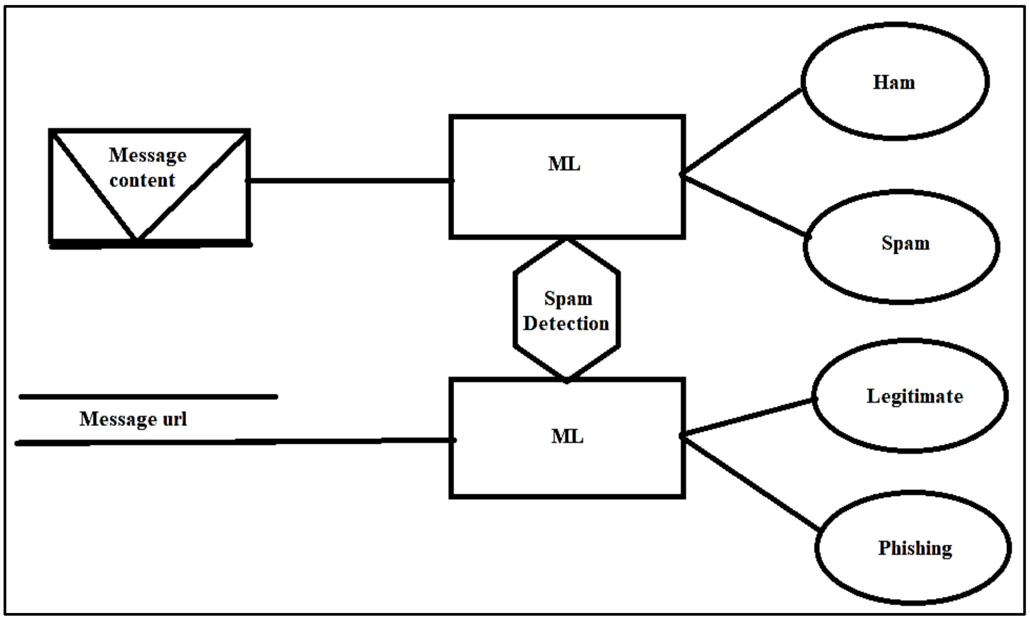

Figure 1Model Flow Chart

2. Literature Review

As we see, there are already some literatures are there by different authors. [1] Mohammed et al in 2013, Instead of using spam trigger words, which may fail, a lexicon-based approach is used to filter the data of the model SVM, K-Nearest Neighbor, Naive Bayes and Decision tree with the accuracy 85.96%. [2]Watcharenwong,Saikaew in 2017, Social features like comments etc., are combined with textual features yields better results using the Random Forest with the accuracy 91.3%. [3]Dewan,Kumaraguru in 2015, Automatic identification of spam text is done with 42 features using ML techniques using the Random Forest with the accuracy 86.9%.[4]Kumar et al in 2018, For effective spam identification, uses both univariate and multivariate distribution across user ratings using Random Forest, SVM, K-NearstNeighbor, Naive Bayes and Decision tree with the accuracy 76%. [5]Mani et al in 2018, The ensemble strategy aided in obtaining a higher accuracy score using Naive Bayer, Random Forest and SVM with the accuracy 87.68%.

3. Date Preprocessing

A. Dataset

This model used datasets form different online websites like [1]FiveThirtyEight, [2]BuzzFeed News, [3] kaggle, [4] Google Public Datasets, [5] Data.gov and some data are created by own thoughts. Using this dataset we can train the model and the name of the datasets we used are “mail data” which contains 5573 lines and 2 columns and the “dataset website” which contains 11055 lines and 32 columns. They are in both text and number format of data on it respectively.

B. Data Cleaning

This involves removing any errors, null values, duplicates, missing values, or outliers from the data set.

C. Feature Extraction

Feature extraction is an important step in building an effective spam detection system. The goal of feature extraction is to identify the most relevant and informative features that can help distinguish between legitimate and spam messages. Below features are used to detect the spam content.

Stop Words:These are often removed during the feature extraction process. Stop words are common words that are often removed from text because they are not informative and do not add much meaning to the text. Examples of stop words include "a", "an", "the", "and", "or", "in", "of", and "to".

Frequency of Certain Words:Spam messages often contain certain words or phrases that are not common in legitimate messages. By counting the frequency of these words, we can create a feature that can help distinguish between spam and legitimate messages.

Length of the Message:Spam messages are often shorter than legitimate messages. Therefore, the length of the message can be a useful feature for spam detection.

Presence of Certain Characters: Some spam messages use excessive punctuation, capitalization, or special characters. By counting the frequency of these characters, we can create a feature that can help identify spam messages.

Below features are used to detect the link or URL which is phishing or legitimate.

Having IP Address:legitimate websites typically use domain names in their URLs, while malicious websites may use IP addresses directly to hide their identity or to bypass domain-based filtering.

URL Length: legitimate URLs tend to be shorter and more concise, while fraudulent or phishing URLs may be longer and contain multiple subdomains or parameters.

Shortening Service: one can parse the URL and check whether it is hosted on a known shortening service domain (e.g., bit.ly, goo.gl, etc.). If the URL is hosted on a shortening service domain, the feature is set to 1, otherwise it is set to 0.

Having @ Symbol: The presence of "@" in a URL may indicate that the URL is an email address or that it is being used to bypass URL-based filtering. Therefore, it can be used as a feature to identify potentially suspicious URLs.

Double Slash Redirecting:it can be used by attackers to bypass filters or to redirect users to a different destination than intended. Therefore, it can be used as a feature to identify potentially suspicious URLs.

Prefix Suffix: Characters or words that appear before the domain name in a URL (e.g., "www" or "https://"), while suffixes are characters or words that appear after the domain name in a URL (e.g., ".com" or ".pup").

Having Sub Domain: Attackers may use subdomains that mimic legitimate websites to trick users into visiting a phishing site.One can simply parse the URL and count the number of subdomains. If there are one or more subdomains, the feature is set to 1, otherwise it is set to 0.

***SSL Final State*:** One can simply check if the URL begins with "https://" (indicating that SSL encryption is being used) or "http://" (indicating that SSL encryption is not being used).

Domain Registration Length: The idea behind using this feature is that legitimate websites tend to have longer registration periods, while fraudulent or spammy websites tend to have shorter registration periods.

Favicon: The idea behind using this feature is that legitimate websites tend to have custom favicons that are unique to their brand, while fraudulent or spammy websites tend to use generic favicons or no favicon at all.

Port: Ports are used to identify specific services or processes running on a server, and the default port for web traffic is 80 for HTTP and 443 for HTTPS. However, attackers may use non-standard ports to bypass firewall rules or to hide malicious activity.

HTTPS Token: The HTTPS token feature is set to 1 if the URL protocol is HTTPS, and 0 otherwise. The idea behind using this feature is that legitimate websites often use HTTPS to encrypt their communications and protect users' data, while fraudulent or phishing websites may not use HTTPS or may use invalid or self-signed certificates.

Request URL: One can simply parse the request URL and extract relevant information such as the path, query parameters, and fragment identifier. Additionally, one can perform further analysis on the request URL to check for known patterns or indicators of malicious activity. For example, known malicious domains or IP addresses may be present in the request URL.

URL of Anchor: Attackers may use anchor links to hide the true destination of a URL or to redirect users to a phishing site.To extract this feature, one can simply parse the HTML of the web page and collect all the anchor URLs.

Links in Tags: It can be used to identify potentially malicious or suspicious URLs. For example, attackers may use hidden links in HTML tags to redirect users to a phishing site or to download malware onto their devices.

SFH: One can simply check if the URL contains an iframe or another element that makes a request to a different domain than the one serving the original page. If such an element is present, the SFH feature is set to 1, indicating that there is a potential Same Origin Policy violation. If no such element is present, the SFH feature is set to 0, indicating that there is no potential Same Origin Policy violation. Additionally, one can further analyze the content and behavior of the element to determine if it is attempting to perform any malicious activities.

Submitting to Email: Attackers may use this technique to collect sensitive information from users, such as usernames, passwords, or other personal data. To extract this feature, one can simply check if the URL contains a web form or other input field that is configured to submit data to an email address. If such a field is present, the submitting to email feature is set to 1, indicating that there is a potential risk of data being sent to an email address. If no such field is present, the submitting to email feature is set to 0, indicating that there is no potential risk of data being sent to an email address.

Abnormal URL:One can check for URLs that include non-standard characters or symbols, or that use unusual or unexpected domain names or paths. Additionally, one can use machine learning algorithms or other techniques to identify patterns or features in abnormal URLs that are indicative of malicious or suspicious behavior. If an abnormal URL is detected, the abnormal URL feature is set to 1, indicating that there is a potential risk of the URL being malicious or suspicious. If no abnormal URLs are detected, the abnormal URL feature is set to 0, indicating that there is no potential risk of the URL being malicious or suspicious.

Redirect: To extract this feature, one can simply check if the URL contains a redirect or if there are multiple redirects involved in the process of accessing the final URL. If a redirect is detected, the redirect feature is set to 1, indicating that there is a potential risk of the URL being malicious or suspicious. If no redirects are detected, the redirect feature is set to 0, indicating that there is no potential risk of the URL being malicious or suspicious. Additionally, one can further analyze the content and behavior of the redirect to determine if it is attempting to redirect users to a malicious or phishing website.

On Mouse Over: Attackers may use this feature to display a URL that is different from the actual URL that the link will redirect to, or to execute malicious code when the user hovers over the link. To extract this feature, one can simply check if the URL contains JavaScript code that is executed when the user hovers over the link. If such code is present, the On Mouse Over feature is set to 1, indicating that there is a potential risk of the URL being malicious or suspicious. If no such code is present, the On Mouse Over feature is set to 0, indicating that there is no potential risk of the URL being malicious or suspicious.

Right Click: Attackers may use this feature to disable or modify the behavior of the right-click menu to prevent users from accessing important functionality, or to execute malicious code when the user right-clicks on a link or page.

Pop Up Window: It can be used to identify potentially malicious or suspicious URLs. Attackers may use this feature to open a new window or tab that displays a phishing website or to execute malicious code when the user clicks on the link or button.

Iframe: Extract features from it by analyzing the HTML code of the iframe. Here are some possible features that we could extract Source URL, Dimensions, Source domain, Embedded content type, Location on the webpage. To extract these features, you can use an HTML parser library such as Beautiful Soup in Python.

Age of Domain:To extract the age of a domain from a URL, you can use WHOIS records. WHOIS records contain information about domain registration, including the creation date of the domain.

DNS Record: To extract DNS records from a URL, you can use the " dnspython " library in Python.

Web Traffic: To extract web traffic information from a URL, we can use the requests library in Python to send a GET request to the URL and then extract relevant information from the response. Here are some possible features that we could extract Response time, Response status code, Response headers, Page content.

Page Rank: To rank web pages in the search engine results. It works by analyzing the number and quality of links to a page to determine its relevance and authority. In a binary classification task where the goal is to distinguish between URLs and non-URLs, the PageRank of a URL and its length or domain name can be used as features. Other features could include the presence of certain keywords or patterns in the URL.

Google Index: It involves extracting important information from the indexed web pages, such as the URL, page title, meta description, content, and other relevant attributes.

Links Pointing to Page**:**Links pointing to a page in a URL, we can use regular expressions to extract URLs from text data. It extracts all URLs from the text variable using the regular expression url_regex. Then, it filters the URLs to only keep those that start with the page_url variable, which is the URL of the page we are interested in. Finally, it prints the filtered URLs.

4. ML Algorithms

Machine Learning algorithms are used to build predictive models, which can be used for a variety of tasks, such as classification, clustering, regression, and anomaly detection. Here we used Supervised learning algorithms.These algorithms learn from labeled data and that are used for prediction tasks, such as classification or regression. Examples of supervised learning algorithms include decision trees, random forests, k-nearest neighbors, Naïve Bayes, support vector machines, and Logistic regression.

A. Logistic Regression:It is a simple, yet powerful algorithm that can be used for a wide range of classification problems. In logistic regression, the goal is to predict the probability of a binary outcome (e.g., yes/no, true/false, 0/1) based on one or more input variables (also known as features). The output of the logistic regression model is a probability value between 0 and 1, which represents the likelihood of the positive class (e.g., yes, true, 1). It is the algorithm works by estimating the coefficients of the input variables that maximize the likelihood of the observed data. This is typically done using maximum likelihood estimation, which involves finding the values of the coefficients that make the observed data most likely, given the model. To make a prediction using a logistic regression model, the input variables are first transformed into a linear combination of the coefficients using a weighted sum. This is then passed through a logistic function (also known as a sigmoid function), which maps the linear combination to a probability value between 0 and 1.

B. K-Nearest Neighbors:It uses the distance between data points to classify the new input data point. It is an instance-based learning algorithm because it does not learn a model from the training data, but instead stores the training data and uses it to make predictions on new data points. It loads the training data into memory and for a new input data point, calculate its distance to all the other data points in the training set. The most common distance metrics used are Euclidean distance and Manhattan distance. Then select the K nearest data points based on their distance to the new input data point. It classifies the new input data point based on the majority class of the K nearest neighbors. If K=1, the new input data point will be classified as the class of the nearest neighbor. Finally, it returns the predicted class of the new input data point.

C. Decision Tree:It is a popular and powerful algorithm in machine learning used for both classification and regression tasks. It is a supervised learning algorithm that works by recursively partitioning the feature space into smaller and smaller regions based on the values of the input features. At each node of the tree, the algorithm selects the feature that provides the best split between the classes or the most predictive feature for the regression target, based on some measure of purity, such as information gain, Gini impurity or variance reduction. Once it has been trained on a set of labeled data, it can be used to make predictions on new, unseen data. The decision tree starts at the root node and follows the branches of the tree based on the values of the input features until it reaches a leaf node, which corresponds to a predicted class or value for the regression target.

D. Support Vector Machines:It is a powerful and widely used algorithm in machine learning for classification, regression, and outlier detection. It is a supervised learning algorithm that works by finding the hyperplane that maximally separates the classes in the input feature space. The key idea behind it is to find a hyperplane that maximizes the margin between the two classes. The margin is defined as the distance between the hyperplane and the closest data points from each class. The hyperplane that maximizes the margin is the one that is most likely to generalize well to new, unseen data. In cases where the data is not linearly separable, it can use kernel functions to map the input features into a higher-dimensional feature space, where a linear hyperplane can be used to separate the classes. The most used kernel functions are the linear kernel, polynomial kernel, and Gaussian (RBF) kernel.

5. Implementation andResults

For implementing the classifier algorithms like logistic regression, we use Scikit-learn tool to import Machine learning algorithms. Then we divide the dataset into training set and testing set as 50:50, 70:30, 90:10 ratios respectively. Now we evaluate the performance of classifiers by using training set and testing set with different classifiers. The result show in Fig-2 which shows the classifiers performance table. The table contains accuracy score, false negative rate, and false positive rate. They can perform well in 90:10 ratio of data split for training and testing we used.

Dataset Split Ratio	Classifiers	Accuracy score	False Negative Rate	False Positive Rate
50: 50	Logistic Regression	96.75	3.35	2.91
	K-nearest Neighbors	96.74	3.25	2.98
	Decision Tree	96.74	3.63	2.13
70: 30	Logistic Regression	96.84	3.65	2.93
	K-nearest Neighbors	96.82	3.54	2.91
	Decision Tree	96.81	4.1	2.18
90: 10	Logistic Regression	97.591	3.18	2.68
	K-nearest Neighbors	97.03	3.14	2.61
	Decision Tree	97.14	4.63	2.66

Figure 2 Classifiers Performance Table

6. Conclusions

Finally, we concluded that the Logistic regression classifier has high accuracy than other classifiers. Therefore, we used that technique in spam content detection, it can tell the given content in message is ham or spam and we also used the k-nearest neighbors in URL detection, it can tell the given link or URL is phishing or legitimate.

References

1. Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 1532-1543.
2. Kaddoura, S., Alfandi, O., &Dahmani, N. (2020). A spam email detection mechanism for English language text emails using deep learning approach. 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 193-198.
3. Suryawanshi, Shubhangi & Goswami, Anurag & Patil, Pramod (2019). Email Spam Detection: An Empirical Comparative Study of Different ML and Ensemble Classifiers. 69-74.
4. Ameen A. K and B. Kaya, "Spam detection in online social networks by deep learning," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-4.
5. Mohamad, Masurah, and Ali Selamat. "An evaluation on the efficiency of hybrid feature selection in spam email classification". In Computer, Communications, and Control Technology (I4CT), 2015 International Conference on, pp. 227-231. IEEE, 2015.
6. Socher, R., Bauer, J., Manning, C. D., & Ng, A. Y. (2013). Parsing with compositional vector grammars. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, 1, 455-465.
7. Wei, F., & Nguyen, T. (2020). A lightweight deep neural model for sms spam detection. 2020 International Symposium on Networks, Computers and Communications (ISNCC), 1-6.
8. Roy, P. K., Singh, J. P., & Banerjee, S. (2020). Deep learning to filter SMS spam. Future Generation Computer Systems, 102, 524-533.
9. Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., & Alegre, E. (2022). A review of spam email detection: analysis of spammer strategies and the dataset shift problem. Artificial Intelligence Review, 1-29.
10. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American Society for information Science, 41(6), 391-407.
11. Bansal, C., & Sidhu, B. (2021). Machine Learning based Hybrid Approach for Email Spam Detection. 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) 1-4. IEEE.