Malicious URL Detection using Machine Learning

Shakti Kinger*, Pranav Nirmal, Ayush Shrivastav, Aniket Sharma, and Sakshi Saindane

Dr. Vishwananth Karad MIT- WPU, S.No.124, Paud Road, Kothrud, Pune, 411038, Maharashtra, India

shakti.kinger@mitwpu.edu.in
nirmalpranav187@gm ail.com
ayushnshrivastav@gm ail.com
aniketsharma2468@g mail.com
sakshi.saindane15@g mail.com

Abstract. Malicious websites that can hurt users and steal personal information are now a serious problem that requires immediate attention. Machine learning techniques are widely utilised to accurately identify and categorise such websites. These models examine the static elements of the websites to find any potential flaws or hidden exploit code. This study assesses the effectiveness and accuracy of different machine learning models for identifying dangerous URLs. The autoencoder neural network had the lowest accuracy, at 61.50%, followed by the MLP model with an accuracy of 85.50%, and XGBoost with an accuracy of 85.35%. The study shows how well machine learning models work at identifying and stopping the spread of harmful websites. This study highlights the significance of using machine learning techniques to protect users from potential harm.

Keywords: Malicious URL detection, feature extraction, feature selection, machine learning

1 INTRODUCTION

The pace of technological innovation is accelerating in the modern world. The Internet has had a comparable evolution to that of technology. The use of the internet for social and commercial purposes is rapidly expanding. The prevalence of online use for these objectives broadens the potential for cybercrime. Attackers rise in number in direct proportion to connection and user growth. Victims include the government, business, and people. Predicting future dangers and their nature is a tough and, in many cases, intractable undertaking. One of the main risks to cyber security is malware or malicious websites. While malicious URLs in particular pose a significant danger to online security. A frequent and significant risk to cyber security is a malicious URL. Malicious URLs host content that is anomalous, such as spam, phishing attempts, user exploitation, etc. They let unauthorised users be attacked by cars. Every year, they experience enormous financial losses of billions of dollars. First and foremost, it is crucial to regularly identify and respond to such assaults for security [1] . Usually, large blacklists

are used to perform such detections [1] . Exhaustive blacklists are impractical in real life [1] . The billions of URLs that people encounter on a daily basis cannot be addressed by the crude detection algorithms now in use. Large-scale binary classification problems are addressed using machine learning techniques [2] . The classification of good and bad URLs can be done with great accuracy using a variety of classifiers in machine learning [3, 1, 2] . Additionally, Huang et al. [4] uses a greedy selection approach to identify malicious URLs. The experimental investigation on URL detection using machine learning algorithms by Liu et al. is similar [5] . Using a Decision Tree approach, Vu et al. conducts cost-sensitive malicious URL detection [6] . In this empirical study, we carry out the following tasks: (a) gathering a dataset made up of a large number of URLs, including both malicious and non-malicious URLs; (b) dividing the collected dataset into two subsets in the ratio of 80:20 for training purposes and testing purposes; (c) extracting features from the training data classified into lexical features, network-based features, and host-based features; and (d) training the system using the training data and machine learning algorithm In order to detect malicious URLs, and want to provide a thorough examination using machine learning techniques like the Decision Tree, Random Forest, Autoencoder, XGBoost, Multilayer perceptrons and SVM algorithms.

This paper provides an overview of the problem being addressed, the prevalence and dangers of malicious URLs, and the importance of being able to detect them quickly and accurately. It also provides background information on machine learning models and their application to cybersecurity. The proposed system section would describe the methodology and approach used in the current study, the types of data used, and the specific machine learning algorithms used to classify the URLs. The results and decision section would present the findings of the study, including statistical analyses and visual representations of the data. The references section would list all the sources cited in the paper, including books, articles, and other resources. hj

2 RELATED WORK

In the realm of detecting rogue websites, numerous researchers have proposed using machine learning, data mining, and even deep learning approaches. Key characteristic features of web pages, such as network traffic data, content traits, lexical characteristics of URLs, and even domain name system (DNS) data, are essential to the effectiveness of these strategies [2] . While a combination of these features provides better results, obtaining them can be expensive and time-consuming. This can involve downloading entire web pages [7] or searching through numerous DNS servers and ISPs to find enrichment data like geo-location, registration records, and network information [6], which may be unsuitable for real-time systems due to issues of network latency [8] [9] .

Despite the difficulties involved in using non-lexical features to detect malicious URLs in real-time despite their high accuracy values [8] [7] , previous research explored using only lexical features of URLs, and it has been demonstrated that they can accurately detect malicious webpages in real-time systems . The model and dataset utilised,

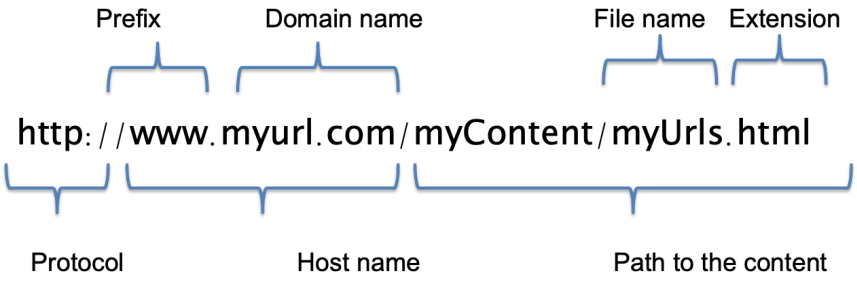

Fig. 1. URLtokenization

however, have a significant impact on the performance reported across the research, even in systems that just use lexical-based characteristics.

For instance, in Decisions Tree Learning Method Based on Three-Way Decisions [5] , the authors assessed three supervised machine learning models (k-nearest neighbour, support vector machine (SVM), and naive bayes classifier) in addition to two unsupervised machine learning models (k-means and affinity propagation). The results demonstrated that supervised models performed marginally better than unsupervised machine learning models. The quantity of data utilised might also be a concern. [10] combines association rule mining with a multitude of machine learning models to classify URLs as hazardous or benign depending on characteristics extracted from the URL. The authors employed the Synthetic Minority Over-Sampling Technique to overcome the class disparity and short dataset size (SMOTE) [11] . The majority of the models greatly improved following class balancing and the resulting increase in dataset size, as shown by a comparison of their performance before and after the process.

While many researchers have classified URLs as harmful using well-known machine learning approaches, several studies have just compared ML models using their datasets. This method can produce inaccurate results that are due to the dataset used rather than the analysis done, as shown in a number of past research projects in different fields. To avoid abandoning essential qualities of malicious website detection systems, such as speed, high accuracy, and low false-negative rates, it is imperative for this study to comprehend how machine learning models generalise over a range of datasets. This paper's main contribution is to help academic scholars and security experts decide which strategies work best in certain situations.

.

3 PROPOSED SYSTEM

For classification purposes, the proposed system employs Machine Learning algorithms and analyses using the various features received from the URL. The full flow diagram of our suggested system, shown in Figure 2 , includes the phases of data collection, feature extraction, data preprocessing, model training, model testing, model selection, and final output phase. It also represents the point of origin for data collection, the features that

were extracted, the models that were applied for classification, and the output, which consists of precision and classification outcome.

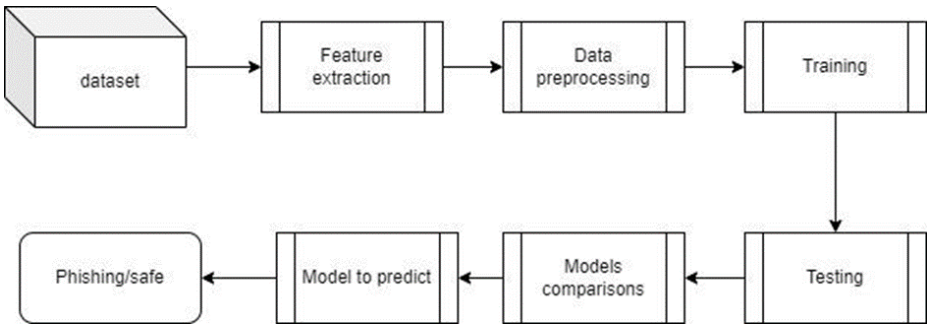

Fig. 2. System Architecture

3.1 Data Collection and Preparation

Data Gathering trustworthy and illuminating datasets is a crucial step in solving learning-based classification or regression problems. For training and testing purposes, data with labels that include both harmful and non-malicious URLs must be gathered from a reputable source in order to improve classification accuracy.

1. Randomly selecting URLs: The dataset from http://data.phishtank.com/data/online-valid.csv, which contained hundreds of phishing URLs, was taken into consideration. To reduce the possibility of data imbalance, a margin value of 10,000 phishing URLs and 5000 real URLs was taken into account. We chose 5000 samples at random from the dataframe.
2. Feature Selection and Extraction: When dealing with a large dataset, the phase of feature selection and extraction is crucial and challenging. A feature in machine learning is a distinct measurable quality, trait, or attribute of a phenomenon being observed. For effective algorithms, selecting informative, differentiating, and independent features is a crucial step.
3. Tokenizing the URLs: The dataset's "URLs" column was tokenized. A total of 5000 reliable URLs were stored and utilised to extract model training features.
4. Feature Extraction: The URLs were processed using the following functions to extract features and shown in Figure 3 :

–
  - Domain Extraction: This feature extraction function pulls out the URL's domain. It provides details about the URL's origin.
  - Check for IP Address (Have IP): This function determines whether the URL has an IP address. If the URL points directly to an IP address rather than a domain name, this can help determine that.

- Check for the "@" symbol (Have At): This function determines whether the URL contains the "@" symbol. The '@' mark in a URL can be a red flag that it isn't a standard URL and should raise suspicion.
- URL Length Categorization (URL Length): This function calculates the URL's length and classifies it. The intricacy and structure of the URL can be inferred from the URL's length.
- Number of slashes in the URL (URL Depth): This method returns the quantity of slashes in the URL. The amount of slashes in the URL might provide insight into its structure and depth.
- Check for Redirection '//' (Redirection): This function determines whether the URL contains the redirection character. Redirections in URLs can mean that a user is being sent to a different website, which raises several warning flags.

The function that determines whether the "HTTPS" token is present in the domain portion of the URL is called "Presence of HTTPS Token in Domain Part of URL" (https Domain). Although the existence of "HTTPS" can suggest that a website is safe, it does not ensure the website's reliability.

–
- Check for Shortening Services (Tiny URL), determines whether the URL is being shortened. The original URL may be hidden when using a shortening service, which makes it more challenging to verify the legitimacy of a website.
- Prefix/Suffix Separation in Domain (Prefix/Suffix): This function determines whether the domain name contains a prefix or suffix that is divided by a hyphen (-). This may be a sign that the domain name was made with an ulterior motive and is not genuine.
- Web Traffic (Web Traffic): This feature offers data on the volume of traffic that a website receives. A website's high traffic volume may be a sign that it is a trustworthy website.
- Domain Age (Domain Survival Time): This function offers data about the Domain Survival Time. It computes the discrepancy between creation and termination times. Long-running domains might be viewed as having a higher level of authenticity.
- Domain End Time (Domain End): This function determines how much time has passed since the domain was terminated. A domain that is about to expire can be viewed as less trustworthy.
- The IFrame Redirection (iFrame) function determines if the URL employs iFrame redirection. When an iFrame redirection is used, it may be a sign that the user is being taken to a separate website, which raises several red flags.
- Status Bar Effect of Mouse Over (Mouse Over): This function examines the status bar effect of mouse over. The status bar's behaviour can reveal details about the legitimacy of a website.
- Right Click Attribute Status (Right Click): This function determines the right click attribute's current state. The right click attribute's behaviour can reveal details about the legitimacy of a website.

–
- Number of Forwardings (Web Forwards): This function determines how many forwardings are in the top of the form.

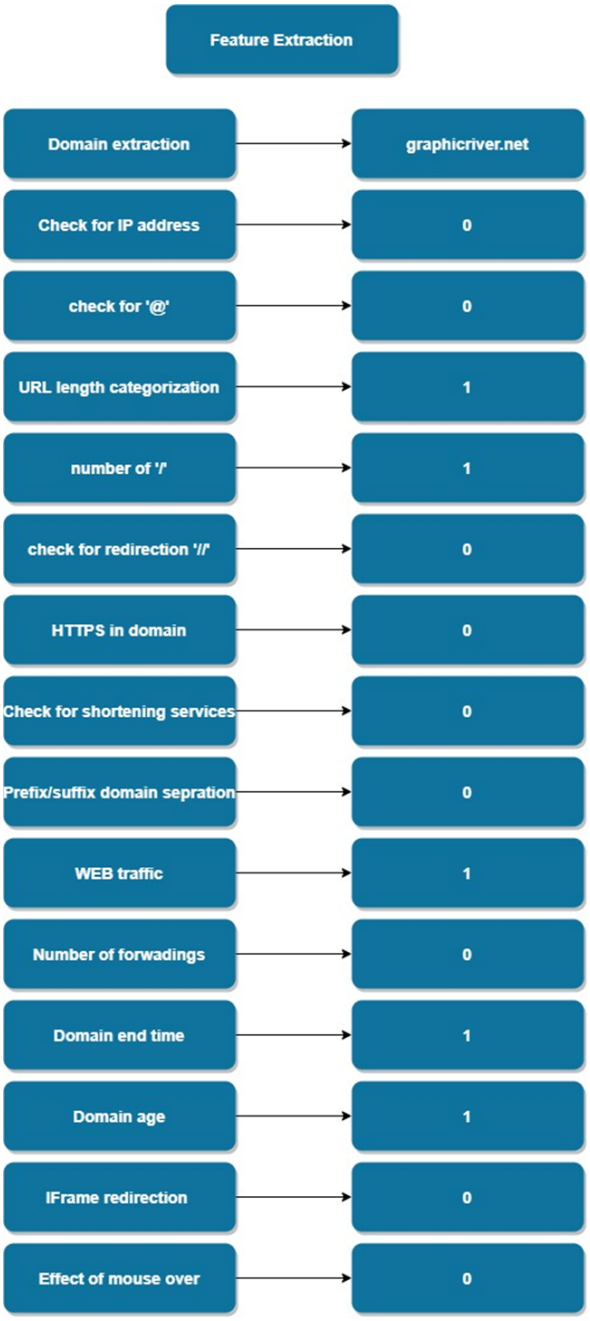

Fig. 3. FeatureExtraction

- Creation of the Legal Dataset: Using the aforementioned functions, the characteristics were retrieved from the 5000 Legal URLs. The genuine dataset was updated with the returning values, which were either 1 or 0.

Phishing Dataset Creation: The phishing URL dataset was used to extract features using the same feature extraction techniques. The phishing dataset was updated with the returned results, which were either 1 or 0.

URL Data Creation: The urldata.csv file was made by concatenating the legitimate and phishing datasets. The model was then trained using this file.

Important details about the structure and behaviour of the URLs were provided by the extracted characteristics, which were significant for identifying malicious URLs. The feature extraction features made it easier to comprehend the existence of IP addresses, URL length, URL depth, redirection, HTTPS token presence, and other significant characteristics.

3.2 Machine Learning Modelling

Our analysis was focused on this. It required transforming the uniform dataset into pertinent models. This sub-problem involved: (a) shortlisting machine learning models based on literature use for malicious website detection or similar tasks; (b)

reducing the list of shortlisted machine models to top models based on empirical performance in the literature related to malicious website detection and other well-known tasks; (c) choosing the metrics that would be used for training and validating the models; and (d) developing the machine learning models and training them on the training data, (e) saving the best result giving fine-tuned model real life predictions. Machine Learning models used are:

Decision Tree Classifier [12] : A tree-based supervised learning algorithm, a decision tree classifier generates a decision tree to make predictionsFigure 4 . Due to its ability to manage intricate data linkages and produce understandable findings, it is important in the challenge of malicious URL identification. By creating a decision tree that predicts a URL's class based on its attributes, the technique can be used to identify whether a URL is harmful or not.

Random Forest Classifier [13] : This ensemble learning approach builds numerous decision trees and averages their predictions to increase the model's accuracy and stabilityFigure 5 . Because it may improve accuracy and decrease the overfitting issue, it is important in the problem of malicious URL identification.

Multilayer Perceptrons (MLPs) [14] : Between the input and output layers of MLPs' feed-forward artificial neural networks are a number of hidden layersFigure 6 . Because of their ability to manage intricate non-linear correlations between the input attributes and the output class, they are important in the challenge of malicious URL identification. The intricate connections between the URL features and their class can be discovered using MLPs.

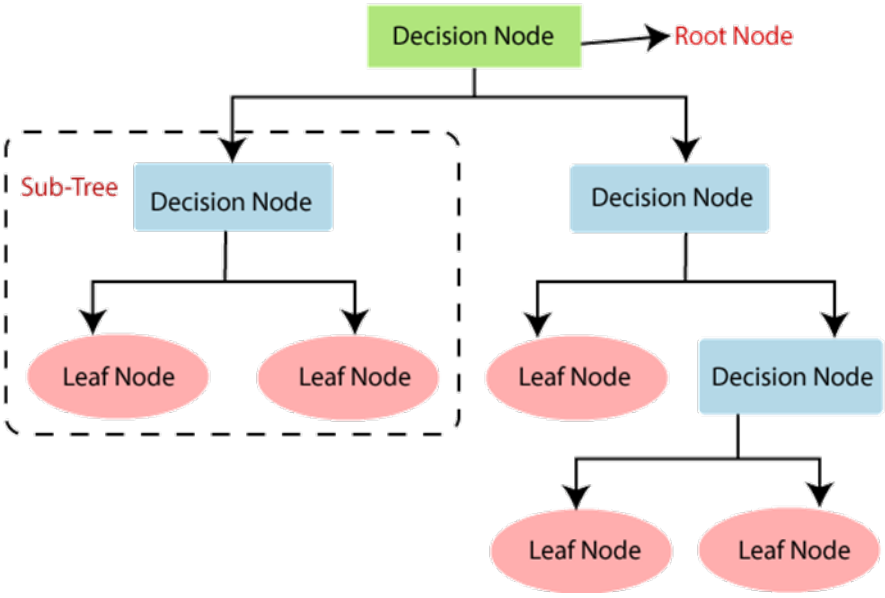

Fig. 4. DecisionTree graph

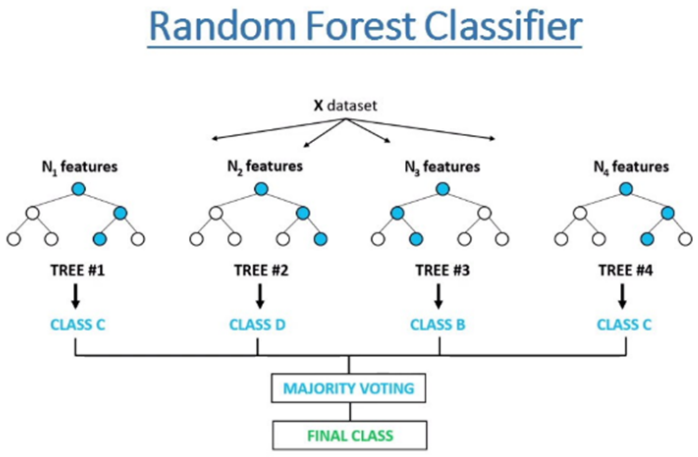

Fig. 5. RandomForest Classifier

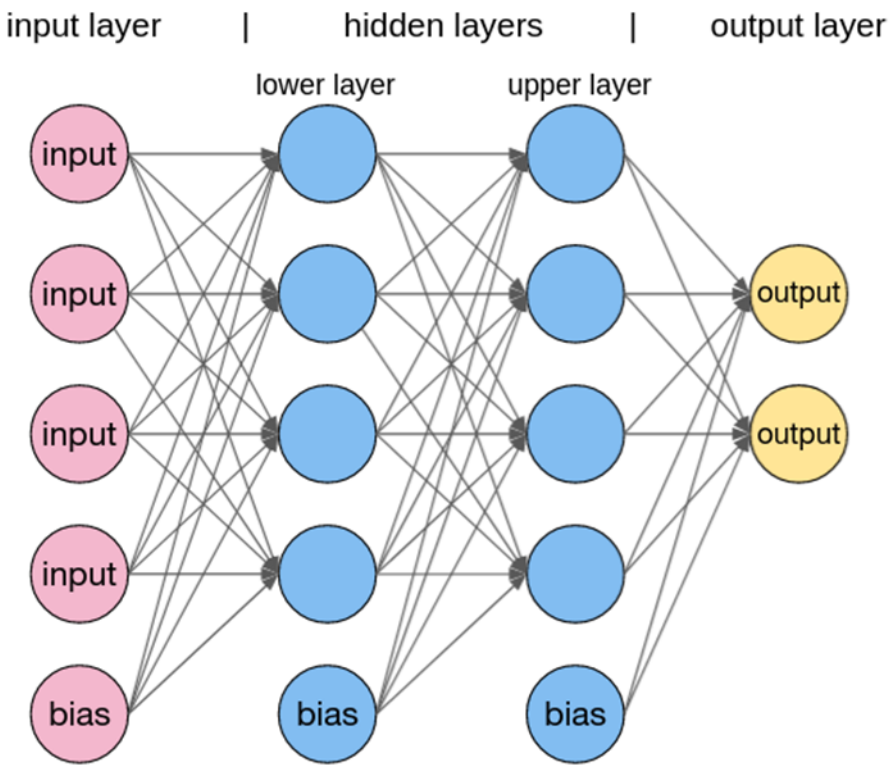

Fig. 6. MLPnetwork

XGBoost Classifier [15] : To increase prediction accuracy, XGBoost uses a gradient boosting tree technique to construct a tree-based modelFigure 7 . Because it can handle big datasets, deliver results that are easy to understand, and increase accuracy by combining the predictions of other trees, it is important in the malicious URL identification problem.

Autoencoder Neural Network [16] : Unsupervised deep learning methods known as autoencoder neural networks can learn to reassemble the input dataFigure 8 . Due to their ability to extract crucial elements from URL data that can be used to create prediction models, they are critical in the challenge of malicious URL identification. To lower the dimensionality of the URL data and enhance the predictive model's performance, autoencoders can be utilised as a pre-processing step

Support Vector Machines [17] These are linear classifiers that locate the largest margin hyperplane to divide the data into many classesFigure 9 . Because they can handle high-dimensional data and produce reliable findings even in the presence of outliers,

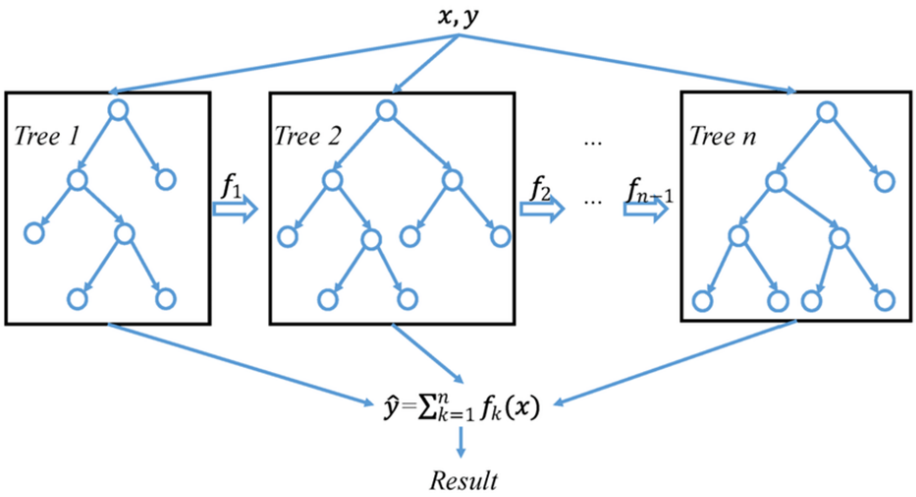

Fig. 7. XGBooststructure

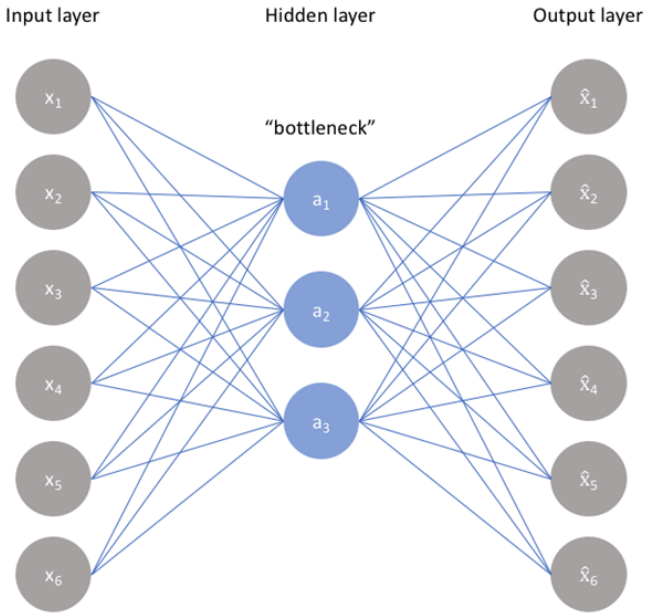

Fig. 8. AutoencoderNeural Network

they are important in the challenge of malicious URL detection. By identifying the maximum margin hyperplane that divides the URL data into distinct groups, SVM can be utilised to develop a predictive model for the problem of malicious URL identification.

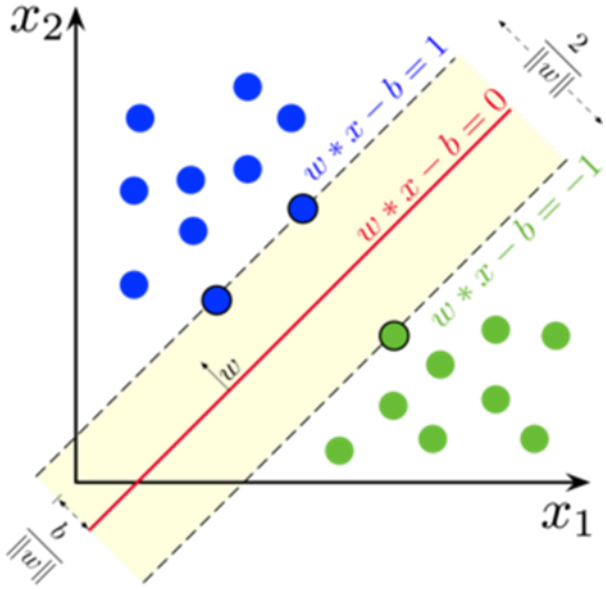

Fig. 9. SVMgraph

4 RESULTS AND DECISION

TThe extracted features from the labelled data, which includes both harmful and helpful URLs, are then used to separate the data into training and testing sets [8]. The training data are supplied to different models including Multilayer perceptrons, XGBoost, AutoEncoder, Random Forest, Decision Tree, and SVM after being passed through various methods for feature extraction and labelling. Additionally, the trained model is evaluated using the training dataset, which contains features without labels, and the model's correctness is determined. The training dataset comes before the tested dataset in the process above, which is repeated for 80:20 data splits. The flow diagram for the experiment is shown in Figure 2 . There are both benign and harmful URLs in the dataset. The three components of a URL are the protocol, domain name, and path. Numerous host-based features, lexical features, and site popularity features are derived from the URL. The following Source [8] is used to compile a collection of benign and malicious URLs.

4.1 Model Analysis

The training data's non-string columns are taken out. The data are delivered to SVM, XGBoost, Autoencoder Neural Network, Random Forest, Multilayer Perceptrons (MLPs), Decision Tree, and MLPs. The input supplied to the Decision Tree function is the number of trees. The fit function, which accepts training data and output from feature extraction as inputs, is used to train the model. Utilising test data and the cross val score function cross validation module, the trained model's accuracy is evaluated. The result includes the test data and accurately anticipated values.

The dataset is split 80-20 into train and test sets for each classification model. of the data in a decision tree with a maximum depth of 5 were predicted accurately. Accuracy was 81.5% and 82.4% for Random Forest classification in training and testing data. Autoencoder Neural Network. accuracy was about 84.55% for Multilayer Perceptrons and XGBoost, and 80% for SVM.

1. F1 score is a statistic that combines recall and accuracy into a single value. It is referred to as the harmonic mean of recall and accuracyEquation (1) .

$$F1\ Score = 2\ +\ (precision\ -\ recall)\ /\ (precision\ +\ recall) \tag{1}$$

To offer a fair assessment of a classifier's performance, the F1 score considers both accuracy and recall. It is helpful when the dataset is unbalanced, meaning one class has much more data than the other.

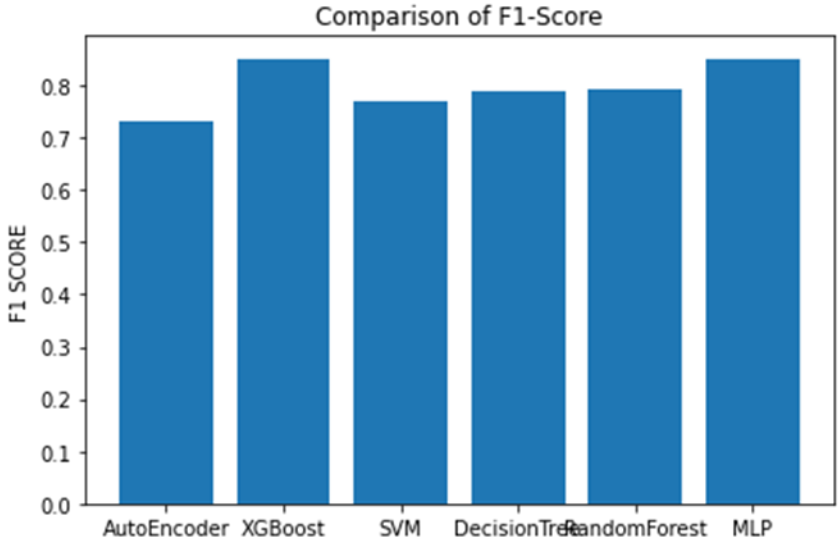

Fig. 10. F1-scoresof models

1. Accuracy is a statistic that counts the number of occurrences that were successfully categorisedEquation (2)

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{2}$$

Accuracy offers a general assessment of a classifier's effectiveness. It calculates the percentage of correctly classified instances, where TP (True Positive) represents the number of positively predicted instances that actually occurred, TN (True Negative) represents the number of negatively predicted instances actually occurring, FP (False Positive) represents the number of positively predicted instances that actually occurred, and FN (False Negative) represents the number of negatively predicted instances actually occurring.

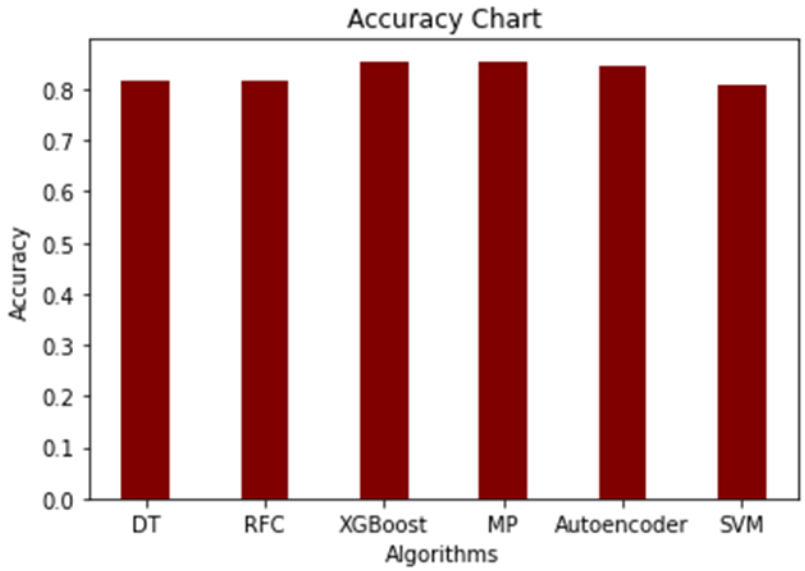

Fig. 11. Accuracyof machine learning models

3. Precision is a statistic that counts the proportion of successfully anticipated positive cases among all positive instances that were expectedEquation (3) .

$$Precision = TP/(TP + FP). \tag{3}$$

Precision measures how well a classifier is able to anticipate positive cases. Instances of TP (True Positive), the number of properly predicted positive instances, and FP (False Positive), the number of mistakenly forecasted positive instances, are used to calculate

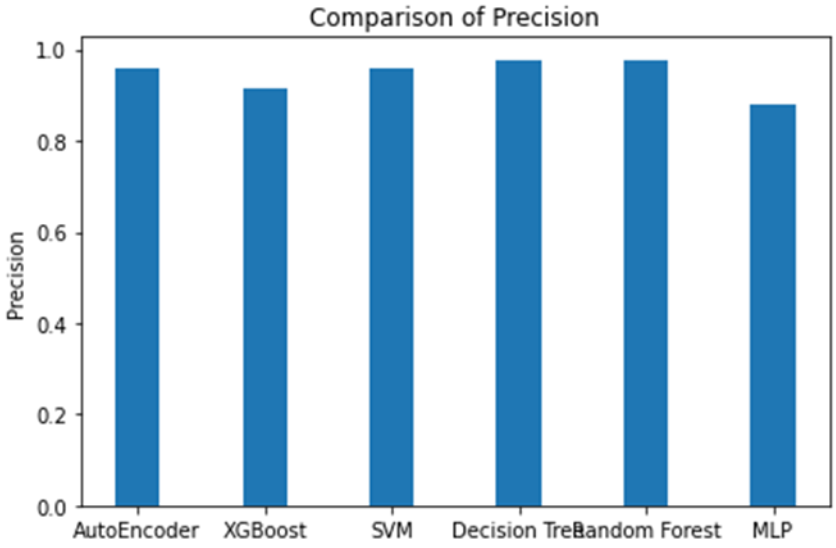

Fig. 12. Precisionsof all the models

the percentage of correctly predicted positive instances among all instances of expected positive instances.

4. Recall is a statistic that counts the proportion of positive cases that were properly anticipated out of all positive instancesEquation (4) .

$$Recall \ = \ TP \ / \ (TP \ + \ FN). \tag{4}$$

Recall measures how well a classifier can anticipate positive cases. TP (True Positive) is the number of properly predicted positive cases, and FN (False Negative) is the number of mistakenly forecasted negative instances. It calculates the proportion of correctly predicted positive instances among all real positive instances.

4.2 CONCLUSION

The majority of online criminal operations are based on malicious Web sites. The risks posed by malicious websites are significant, and end users must not be allowed to access them. Users should refrain from clicking on these Uniform Resource Locators (URL). A number of machine learning algorithms, including Multilayer Perceptrons, XGBoost, AutoEncoder, Random Forests, SVMs, and Decision Tree are deployed on training dataset to solve the binary classification issue of detecting dangerous URLs. Additionally, it has been shown that the Multilayer Perceptrons and XGBoost classifiers outperform the others for the specific challenge.

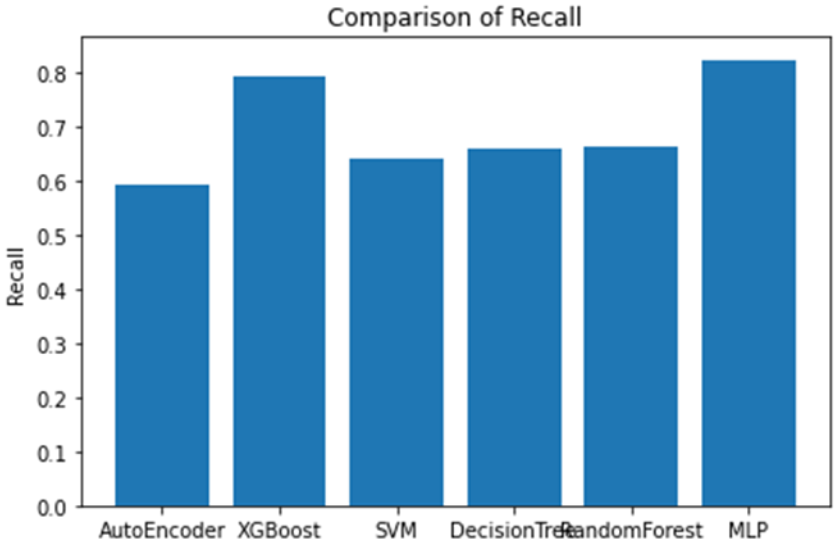

Fig. 13. Recalls ofall the models

Algorithm	Precision	Recall	F1-Score	Accuracy
Auto Encoder Neural Network	0.958155	0.591391	0.731368	0.781333
XGBoost	0.916475	0.792053	0.849734	0.859
Support Vector Machine	0.960396	0.642384	0.769841	0.806667
Decision Tree	0.976494	0.660265	0.787831	0.821
Random Forest	0.978537	0.664238	0.791321	0.823667
MultiLayer Perceptron	0.881728	0.824503	0.852156	0.856

Fig. 14. Algorithmwith their precision, recall, F1-score and accuracy

References

1. Mansoori, M., Welch, I., Fu, Q.: YALIH, yet another low interaction honeyclient. Proceedings of the Twelfth Australasian Information Security Conference **149**, 7–15 (2014)
2. Hassan, U., Ali, I., Abideen, R.H.U., Khan, Z., Kouatly, T.A., R: Significance of Machine Learning for Detection of Malicious Websites on an Unbalanced Dataset. Digital **2022**, 501–519
3. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Learning to detect malicious URLs. ACM Trans. Intell. Syst. Technol **24** (2011)
4. He, S., Li, B., Peng, H., Xin, J., Zhang, E.: An Effective Cost-Sensitive XGBoost Method for Malicious URLs Detection in Imbalanced Dataset. IEEE Access **9**, 93089–93096 (2021)

5. Liu, Y., Xu, J., Sun, L., Du, L.: Decisions Tree Learning Method Based on Three-Way Decisions. In: Yao, Y., Hu, Q., Yu, H., Grzymala-Busse, J. (eds.) Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing. vol. 9437. Springer (2015)
6. Vu, P., Nguyen, D., Turaga: Firstfilter: A cost-sensitive approach to malicious URL detection in large-scale enterprise networks. IBM Journal of Research and Development **60**(4), 1–4 (2016)
7. Manjeri, A.S., R, Mnv, K., Nair, A., C, P.: A Machine Learning Approach for Detecting Malicious Websites using URL Features. 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA) pp. 555–561 (2019)
8. Rout, R.R., Lingam, G., Somayajulu, D.V.L.N.: Detection of Malicious Social Bots Using Learning Automata With URL Features in Twitter Network. IEEE Transactions on Computational Social Systems **7**(4), 1004–1018 (2020)
9. & Dery, L., Shmueli, E.: BoostLR: A Boosting-Based Learning Ensemble for Label Ranking Tasks. IEEE Access **8** (2020)
10. Ahmed, A.A., Li, C.: Analyzing Data Remnant Remains on User Devices to Determine Probative Artifacts in Cloud Environment. Journal of forensic sciences **63**(1), 112–121 (2018)
11. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority over-Sampling Technique. J. Artif. Int. Res. **16**(1), 321–357 (jun 2002)
12. Navada, A., Ansari, A.N., Patil, S., Sonkamble, B.A.: Overview of use of decision tree algorithms in machine learning. In: 2011 IEEE Control and System Graduate Research Colloquium. pp. 37–42 (2011)
13. Breiman, L.: Random Forests. Machine Learning **45**, 5–32 (2001)
14. & Nazzal, J., & El-Emary, I., Najim, Salam: Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale. World Applied Sciences Journal. 5 (2008)
15. Chen, T., Guestrin, C.: XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). pp. 785–794. Association for Computing Machinery (2016)
16. Wang, W., Huang, Y., Wang, Y., Wang, L.: Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 496–503 (2014)
17. Evgeniou, Pontil, M.: (2001)