Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

A Study on the Best Classification Method for an Intelligent Phishing Website Detection System

Nor Hapiza Mohd Ariffin[1,3,*], Muhammad Imtiaz Mohamed Iqbal[2], Marina Yusoff[3], Nurul Akhmal Mohd Zulkefli[4]

1 Faculty of Business, Sohar University, Sohar, Oman
2 Software Engineering Department, Motorola Solutions, Malaysia
3 Institute for Big Data Analytics and Artificial Intelligence (IBDAAI), University Teknologi MARA (UiTM), Selangor Darul Ehsan, Malaysia
4 Dhofar University, Oman

ARTICLE INFO

Article history:
Received 18 November 2023
Received in revised form 11 April 2024
Accepted 15 June 2024
Available online 15 July 2024

Keywords:
Phishing; classification algorithms; intelligent systems; machine learning

ABSTRACT

It is impossible to imagine our lives without the internet, but it has also meant that malicious acts such as phishing can be carried out anonymously. Phishers use social engineering or fake websites to trick their victims into giving them personal information such as credit card numbers, bank passwords and other sensitive information. However, the number of phishing attacks has increased significantly in the last year, and current methods of detecting phishing are ineffective. This study focuses on identifying features of phishing websites, evaluating the best dataset and method for applying machine learning classification algorithms, and developing a prototype phishing detection system using the best classification algorithm model. This study investigated the decision tree, logistic regression, and machine learning classification algorithm (k-nearest neighbours). In this study, the waterfall methodology of the system development life cycle (SDLC) was used. All approaches, strategies, tools and relevant theories were explored to provide an overview and understanding of this study. An extensive literature review was conducted to develop the model and problem statement. Data was collected through an open-source licenced website. In addition, the data was pre-processed before training and building the model to ensure no noisy data was present. The parameters of the three models, K-nearest neighbours, decision tree and logistic regression, were adjusted to obtain the best possible model result. The models were then evaluated against the confusion matrix, accuracy, precision, recall, f1 score and decision tree to determine the best classification model for phishing and legitimate websites. The models are fine-tuned with the best parameters for each to achieve an optimal result for phishing detection. After evaluating each model, the decision trees were found to be the most accurate in classifying phishing websites, with an accuracy of 95%. In the future, the system can be improved through different approaches, such as Deep Learning and a fully developed web-based system that can be used in the real world.

[*] *Corresponding author.*
E-mail address: nariffin@su.edu.om

https://doi.org/10.37934/araset.48.2.197210

1. Introduction

Phishing is a significant cybercrime problem in which a perpetrator attempts to steal an internet user's personal information [1]. This is a cyber-attack in which internet users are usually tricked into obtaining their personal data such as credit card details, bank passwords and other sensitive information with the help of disguised emails. It is an attack that targets the user and not the computer. According to Aburrous *et al.,* [2], phishing is a relatively new online crime. Because of its association with technological and social issues, phishing websites are difficult to understand and analyse. The immediate impact of phishing websites is the misuse of information by compromising user data, leading to financial loss or loss of goods for victims. Phishing is a fast-growing cybercrime compared to other online threats, such as hacker attacks and viruses. The most effective and efficient method of detecting phishing websites is using modern technologies such as machine learning [3]. These approaches attempt to analyse the information of a URL and its corresponding websites or web pages to predict new URLs that could be malicious [4]. Detecting new phishing websites is easier because machine learning can be predictive and detect them immediately rather than taking time to detect them. Detecting phishing websites is critical for both home users and businesses. This system is designed to trigger an alert about a new website that is not secure and can be considered a phishing website to prevent internet users' data from being stolen. Also, this system would allow new internet users to access any website without fear.

Many Internet users have been misled by phishing attempts that pose as legitimate websites and steal private information and financial data [5]. Numerous anti-phishing techniques and systems have been developed to protect users from phishing, each with a different strategy, e.g., client-side and server-side security.

Many studies have shown that numerous phishing detection systems are designed to deal with and protect against phishing attacks. The classification of phishing websites is currently based on blacklisting and whitelisting methods. The blacklisting method uses reports from users or companies to detect phishing websites, which are then stored in a database. However, since most phishing websites are ephemeral and usually persist for less than 20 hours, and URLs are often changed quickly, the blacklisting method cannot detect phishing [6]. Furthermore, the system does not protect against a phishing attack targeting a specific individual.

While a system developed using the whitelisting method is only used to identify known good websites, the user must check the user interface every time they visit a website. There is also the possibility that the whitelisting method will result in a phishing website being classified as a safe website. Blacklisting and whitelisting show that both blacklisting and whitelisting are inefficient because users cannot detect new phishing threats if the list is not updated, as phishing websites are usually short-lived [7].

Next, phishing attacks are identified by examining the website's content using the content-based method. Password fields, spelling errors, image sources, links, embedded links and other URL and host-based features are used in this technique [8]. Two examples of content-based approaches are CANTINA and SpoofGuard [9]. Based on Afroz and Greenstadt [5], it is claimed that this method can detect phishing attacks but can also easily avoid them by rearranging the HTML components without changing the website's design. Finally, a study by Chen et al., [10] used a visuality-based approach to detect phishing websites by taking screenshots. To characterise the images of websites, they used the Contrast Context Histogram (CCH) and the k-mean method to cluster the nearest key points. Their method has a 95-99 per cent accuracy rate with only 0.1 per cent false positives. According to them, screenshot analysis is ineffective and inefficient in detecting online phishing.

The current standard method is the blacklist approach, which leads to inefficiency because the blacklist cannot detect new threats from a phisher until the list is updated, as phishing websites are short-lived [7]. Machine learning is a better and more practical solution; phishing activities were monitored between 2004 and 2022. The Anti- Phishing Working Group (APWG) has been monitoring phishing attacks [11]. The numbers it reports have increased tremendously. In the fourth quarter of the first year of 2004, the Anti-Phishing Working Group (APWG) recorded 1,609 phishing attacks per month, compared to an average of 92,564 phishing attacks per month in the fourth quarter of 2016, an increase of 5,753%. The APWG reported an increase in the number of phishing attacks over time. In the third quarter of 2022, APWG observed 1,270,883 total phishing attacks, a new record and the worst quarter for phishing that APWG has ever observed. This is due to new internet users who do not know how to distinguish between genuine and phishing sites. In addition, most phishing attacks targeted payment services, and attacks against the financial sector represented 23.2% of all phishing attacks. This was due to leaks and the lack of a list of secure websites, and vice versa. Moreover, as new phishing tactics are constantly being developed, phishing detection techniques suffer from low detection accuracy and a high rate of false positives [12,13]. Furthermore, the most widely used blacklist-based method is ineffective in responding to phishing attacks, as registering new domains has become easier, and no comprehensive blacklist can guarantee a fully up-to-date database [14,15].

This study has three objectives: first, to determine the best parameters for the logistic regression, decision tree and k-nearest neighbours classification algorithms to develop a prototype phishing website detection system; second, to evaluate the logistic regression, decision tree and k-nearest neighbours classification algorithms; and third, to develop a prototype phishing website detection system using the best classification algorithm model. This study focuses on developing a phishing website detection system using an existing dataset from the online database Kaggle.com, which consists of 10,000 websites, of which 5,000 are phishing websites and another 5,000 are legitimate, to train machine learning. This study focuses on internet users to prevent their data from being shared or stolen on phishing websites. In general, this research aims to find out which websites could be phishing.

2. Methodology

The waterfall model is chosen to put the SDLC methodology into action. The waterfall method assumes all requirements can be captured in the requirements phase [16]. One of the most important aspects of conducting good research is the availability of accurate data. Without it, it would be difficult to conduct research. Even if it seems likely, the result should be questioned because the study does not meet certain criteria. Since numerous sources provide data for free, it is easy to obtain a large amount of data nowadays.

On the other hand, many companies only make the most reliable data sets available for purchase. To solve this problem, the dataset for this study was obtained under an open-source licence from Kaggle.com [17]. There are 11,054 websites used in the datasets for this study. The phishing website datasets can be obtained from the Kaggle website as part of the data collection process and are ready for pre-processing in the following step. The data pre-processing stage describes how the datasets are prepared for the model developed in the modelling stage. Before predictive analysis can begin, the understanding of the data must be familiarised with the features or functions of each feature and their relationship to the class label. This method allows this study to visualise and present results that would not have been possible by simply reading the original dataset [18].

2.1 Data Comprehension

It is necessary to record each attribute after obtaining sufficient data, as shown in Table 1. In this study, the data is thoroughly examined to understand it better. Since this dataset lacks a class label, feature extraction and classification are used.

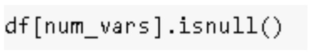
Table 1
Data Comprehension

No.	Attributes	Description
1.	Index	Index list of websites (E.g., 1-11,054)
2.	UsingIP	Using IP-address instead of a registered domain
3.	LongURL	URL Address Length (Long)
4.	ShortURL	URL Address Length (Short)
5.	Symbol@	Symbol (@) included in URL
6.	Redirecting//	Webpage with a redirect link
7.	PrefixSuffix-	Prefixes or suffixes separated by (-) in URLs
8.	SubDomains	An additional part to the main domain name (E.g., store(subdomain). yourwebsite.com)
9.	HTTPS	Additional “HTTPS” token to the domain part of the URL
10.	DomainRegLen	Registered domain length (When the domain is registered)
11.	NonStdPort	Non-Standard Port (E.g., Using port 8080 instead of 80)
12.	HTTPSDomainURL	Domain Session
13.	RequestURL	Examines external objects contained within the webpage are loaded from another domain or own domain
14.	AnchorURL	Examine anchor tag in a webpage (<a>)
15.	LinksInScriptTags	Examine the link's source code
16.	ServerFormHandler	Checks empty string within a domain name
17.	InfoEmail	Check the mail function to see where the personal information is sent to (E.g., mailto: functions)
18.	AbnormalURL	Extracted from a database (E.g., Identity is usually included in the URL of a legal website.)
19.	WebsiteForwarding	Redirect to the website URL
20.	StatusBarCust	The status bar of the current web page
21.	DisableRightClick	Examines JavaScript disabler right click to prevent examine source code
22.	UsingPopupWindow	Checks submit personal information using a popup window
23.	IframeRedirection	Iframe tag to hide additional information redirection
24.	AgeofDomain	Review the age of the domain lived
25.	DNSRecording	Checks DNS records from a database
26.	WebsiteTraffic	The popularity of a website by the number of visitors and how many times pages are visited
27.	PageRank	Determine a website’s page rank value ranging from “0” to “1”
28.	GoogleIndex	Checks if a website is indexed by Google.
29.	LinksPointingToPage	The number of links referring to a website shows its level of legitimacy.
30.	StatsReport	Checks statistical report from several trusted parties
31.	Class	Categorised into (1, -1) 1 means phishing website and -1 means legitimate website

Initially, this data set contained a total of 31 columns. It is cleaned in the data pre-processing using the correlation matrix to remove unimportant features and avoid noise.

2.2 Data Pre-Processing

Data pre-processing is a technique for cleaning data using a Microsoft Excel spreadsheet or a Python script [19]. Fortunately, a Python script has been developed to help us clean data efficiently. These tools are crucial for data analysts when dealing with extraneous data or missing values. It is necessary to check if there are missing values in the dataset to avoid unnecessary errors when training the model. As shown in Figure 1, we can use the method "IsNull ()" to determine all fields with missing values. If a field contains missing values, this method returns True; if it does not, it returns False.

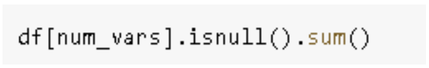


```
df[num_vars].isnull()
```

Fig. 1. Snippet Code of IsNull() method

Since our dataset contains more than 11 054 rows considered large, it will be difficult to identify which columns have missing values clearly. We also need to resolve it with the sum-zero method to see clearly.

Figure 2 shows the code snippet to check the number of missing values for each dataset column. The dataset is ready for further pre-processing using the feature selection method to remove noisy data that could affect the accuracy of a model. The unimportant features are removed during the feature selection process using a correlation matrix to avoid noise and achieve higher accuracy. Table 2 below shows the removed features. This dataset is then further reduced to 20 columns. Table 2 shows the data removed from the dataset using feature selection and considered noisy data due to its low correlation. The next step is to build and train the model.



```
df[num_vars].isnull().sum()
```

Fig. 2. Snippet Code of Missing Value Count

Table 2
Removed Attributes

No.	Attributes	Reason
1.	Index	Unique Data
2.	LongURL	Low Correlation Noisy Data
3.	DomainRegLen	Low Correlation Noisy Data
4.	RequestURL	Low Correlation Noisy Data
5.	LinksInScriptTags	Low Correlation Noisy Data
6.	WebsiteForwarding	Low Correlation Noisy Data
7.	AgeofDomain	Low Correlation Noisy Data
8.	PageRank	Low Correlation Noisy Data
9.	GoogleIndex	Low Correlation Noisy Data
10.	LinksPointingToPage	Low Correlation Noisy Data
11.	Class	Unique Data

2.3 Modelling

In the modelling phase, three machine learning classifiers were used to compare accuracy and performance results, as shown in Figure 3. The classifiers chosen were K-nearest neighbours, decision trees and logistic regression. The dataset was split into two parts: a training dataset and a test dataset with a ratio of 80:20, i.e. 80% training and 20% test. In the appendix, you will find the code for the snippets. In this phase, the most important parameters of each classifier model are tested and visualised in line graphs to show which parameter provides the highest accuracy and lowest error to evaluate the model best. The accuracy of each parameter is visualised in line graphs.

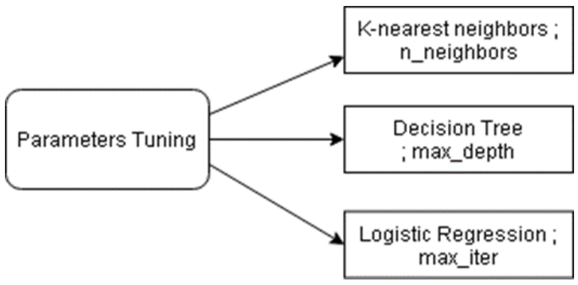

Parameters Tuning
K-nearest neighbors ; n_neighbors
Decision Tree ; max_depth
Logistic Regression ; max_iter

Fig. 3. Parameters tuning

For k-nearest neighbours, the parameter set to give the best possible result is 'neighbours'. It specifies the number of neighbours to be used for classifying new data. The parameters are tuned for the decision tree to give the best possible result max_depth. It specifies the maximum depth of the tree. The parameter max_iter is tuned for logistic regression to obtain the best possible result. It specifies the maximum number of iterations the solver needs to converge.

2.4 Evaluation

A performance evaluation is required to ensure the model is successful [20]. The Decision Tree, K-Nearest Neighbours and Logistic Regression models are evaluated on four criteria, namely precision, accuracy, recall and f1 score. The confusion matrix is a table used to obtain true positive (TP), true negative (TN), false positive (FP) and false negative (FN) classification results. A table illustrating the confusion matrix is shown in Table 3 below.

Table 3
Confusion matrix table

Predicted Value	Actual Value	
	True Positive (TP)	False Positive (FP)
	False Negative (FN)	True Negative (TN)

From the confusion matrix table, precision, accuracy, recognition score, and f1 score can be calculated. The equations for precision, accuracy, recall and f1-score are shown in Eq. (1), Eq. (2), Eq. (3) and Eq. (4) respectively.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{1}$$

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \tag{2}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{3}$$

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision+Recall} \tag{4}$$

Precision is the number of correct positive predictions. A higher precision rate indicates that the proportion of correctly predicted positive observations to the total number of predicted positive observations is high, resulting in a low false positive rate. On the other hand, recall is the percentage of detected positive cases. This is due to the classifier's ability to flag the positive observations correctly. The F1 score is the average value between precision and recall, with a high F1 score indicating better model performance. This is the same as precision, the percentage the model can predict the phishing website based on its classification.

2.5 System Design

This phase, called system design, explains how to build the system architecture, system flowchart and interface design to provide clarity for the research project.

The prototype for the current research project is shown in full in Figure 4. The system consists of a frontend, i.e. a graphical user interface, and a backend and frontend, both written in Python IDLE. The pre-processing includes all the previous steps, which are first done with the raw dataset. Using the pre-processed data, a model with the best parameters is created and trained (k-nearest neighbours, decision tree and logistic regression) [21]. The best model is then used and imported into IDLE via the backend after being checked for accuracy and precision. After pre-processing, the data is sent to the system so that Python IDLE can extract features.

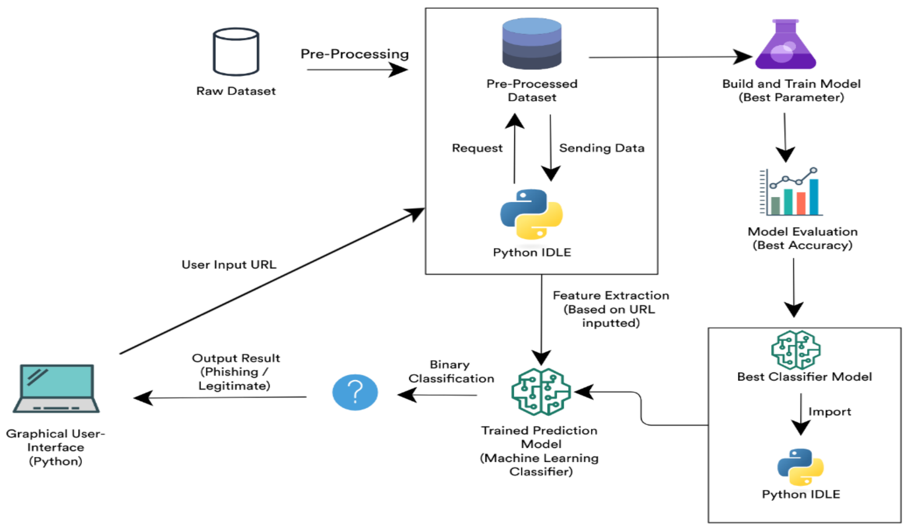

Raw Dataset
Pre-Processing
Pre-Processed Dataset
Request
Sending Data
Python IDLE
Build and Train Model (Best Parameter)
Model Evaluation (Best Accuracy)
User Input URL
Feature Extraction (Based on URL inputted)
Best Classifier Model
Import
Python IDLE
Output Result (Phishing / Legitimate)
Binary Classification
Trained Prediction Model (Machine Learning Classifier)
Graphical User-Interface (Python)

Fig. 4. System prototype architecture diagram

The user enters the URL they want to check when using the system. The system then takes the features from the dataset and examines each URL that the user has entered so that machine learning can determine whether it is a legitimate or a phishing site. In the end, the machine learning classifies the URLs entered by the user as a binary number (1, -1) based on the feature extraction and sends it to the user and to the front end to show the result as "phishing" or "legitimate".

Figure 5 shows the system where a user first logs in with Phyton IDLE. The user enters the desired URL (e.g. www.youtube.com) into the input field. The system then gathers feature information from the URL, applies a predictive model to determine whether the URL is phishing, and outputs the classification. Users can exit the system if they cannot continue typing in the URL to be checked.

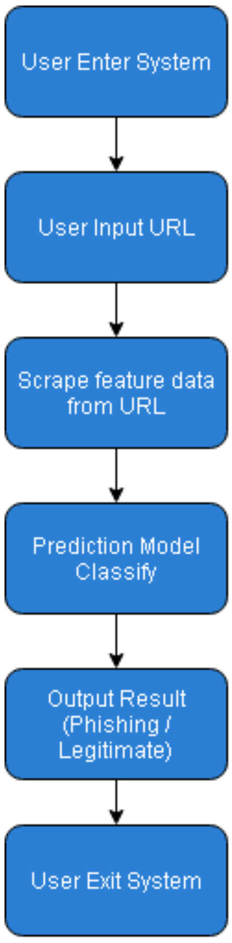

User Enter System
User Input URL
Scrape feature data from URL
Prediction Model Classify
Output Result (Phishing / Legitimate)
User Exit System

Fig. 5. System flowcharts

2.6 System Development

The system was developed using Python IDLE and is delivered through a graphical user interface (GUI) that allows any user to test the system's functionality. The GUI also allows the user to see the result of each prediction for each URL entered. The system is then evaluated against other previously published work. Figure 6 shows the architecture of the prototype system. The dataset and trained model are imported and the back end and front-end are developed using Python as the graphical user interface (GUI).

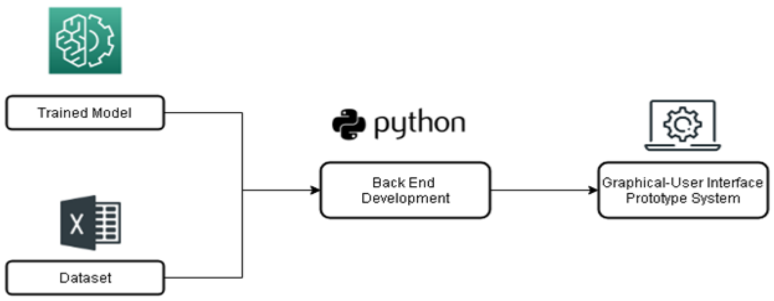


Fig. 6. GUI prototype system architecture

3. Results

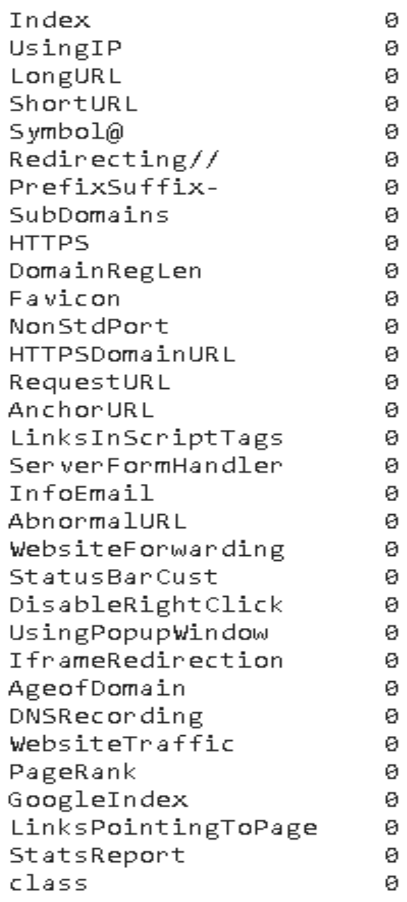
The results of the experiment were discussed in this section. The study of sentiment and the machine learning methods used in the project are discussed. In addition, the results and performance of each machine learning technique.

3.1 Data Collection

There are several sources of data for phishing websites. The most complete and free data set can be obtained from a licenced website called Kaggle.com. This data covers almost every aspect of a website to check whether a website is legitimate or not. In total, there are 32 columns and 11,054 pieces of data. The dataset was saved in CSV file format.

3.2 Data Pre-Processing

After collecting sufficient data, the next step is to prepare it for model training and testing through pre-processing. All data pre-processing is done with a Python script. The data pre-processing steps are to detect and replace missing values when missing values are found in the (zero) column and to perform feature selection using a correlation matrix to remove noisy data for cleaning. The raw dataset is checked using the isnull() method to detect missing values, as mentioned earlier. Each column returns true if missing values are detected in the dataset and false if there are no missing values. "False" is displayed for all columns in the raw dataset, but since the raw dataset contains 32 columns and 11,054 pieces of data, we need a unique count of the missing values for each column. Figure 7 shows that every column in the dataset from 'Index' to 'Class' displays 0, meaning no missing values were detected in the raw dataset. This shows that the dataset does not contain any unfilled or invalid values for each data.



```
Index                  0
UsingIP                0
LongURL                0
ShortURL               0
Symbol@                0
Redirecting//          0
PrefixSuffix-          0
SubDomains             0
HTTPS                  0
DomainRegLen           0
Favicon                0
NonStdPort             0
HTTPSDomainURL         0
RequestURL             0
AnchorURL              0
LinksInScriptTags      0
ServerFormHandler      0
InfoEmail              0
AbnormalURL            0
WebsiteForwarding      0
StatusBarCust          0
DisableRightClick      0
UsingPopupWindow       0
IframeRedirection      0
AgeofDomain            0
DNSRecording           0
WebsiteTraffic         0
PageRank               0
GoogleIndex            0
LinksPointingToPage    0
StatsReport            0
class                  0
```

Fig. 7. Result missing value count

3.3 Feature Selection

To achieve a better result and remove noisy data, the feature selection of the data set is done using a correlation matrix [22]. A higher correlation between features means the feature is needed and important for building and training the model. The correlation diagram shows that there are many low-correlated or considered "unimportant" features for this dataset. Index and class have been removed as they are unique in the correlation matrix. It can be seen that LongURL is barely correlated with any feature other than ServerFormHandler, which is only 0.41. Other features such as DomainRegLen, LinksInScriptTags, websiteForwarding, AgeofDomain, PageRank, GoogleIndex, LinksPointingToPage can be considered noisy data as they have the lowest correlation with other features. Figure 8 shows the correlation matrix of the preprocessed dataset. The figure clearly shows that all features are highly correlated with each other, especially UsingIP, ShortURL, Symbol@, Redirecting//, Favicon, NonSTDPort, HTTPSDomainURL, InfoEmail, AbnormalURL, StatusBarCust, DisableRightClick, UsingPopupWindow and IframeDirection mostly have high correlation values with other features. The low correlation features were removed and saved in CSV file format. Originally, there were 32 columns, including the unique columns "Index" and "Class". After pre-processing, the dataset was reduced to 21 columns without "index" and "class". A total of 10 columns were deleted due to low correlation with other characteristics.

3.4 Modelling

Moving on, after the dataset is pre-processed. The dataset is split into 80% and 20% for testing and training. The model is ready to be built and trained, and parameter tuning will be done for each model (k-nearest neighbours, decision tree, logistic regression) to gain optimal results for each model by tuning their most important parameters (1 – 15). The model will be compared using accuracy, precision, recall, and f1-score, as well as a bar chart, for clear results to compare each model.

Figure 8 compares the accuracy of the model k-nearest neighbours, decision tree and logistic regression. The accuracy of k-nearest neighbours is 0.9299, a decision tree is 0.9489 and logistic regression is 0.9104. As the bar chart shows, the highest accuracy achieved by decision tree and the lowest accuracy is logistic regression.

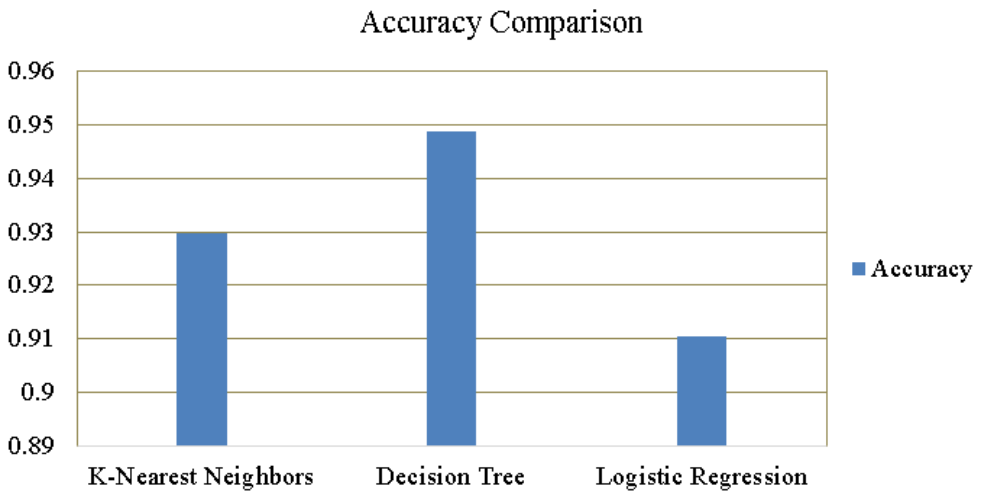


Fig. 8. Accuracy comparison

Figure 9 compares the precision of the model k-nearest neighbours, decision tree and logistic regression. The precision of k-nearest neighbours is 0.9304, a decision tree is 0.9496, and logistic regression is 0.9113. As the bar chart shows, the highest accuracy achieved by decision tree and the lowest accuracy is logistic regression.

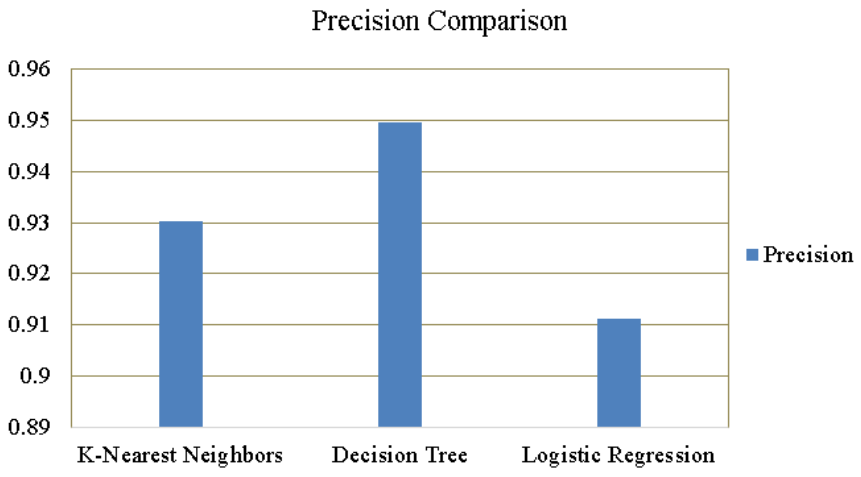


Fig. 9. Precision comparison

Figure 10 compares the recall of the model k-nearest neighbours, decision tree and logistic regression. Recall of k-nearest neighbours is 0.9270, a decision tree is 0.9494, and logistic regression is 0.9064. As the bar chart shows, the highest accuracy achieved by the decision tree and the lowest accuracy is logistic regression.

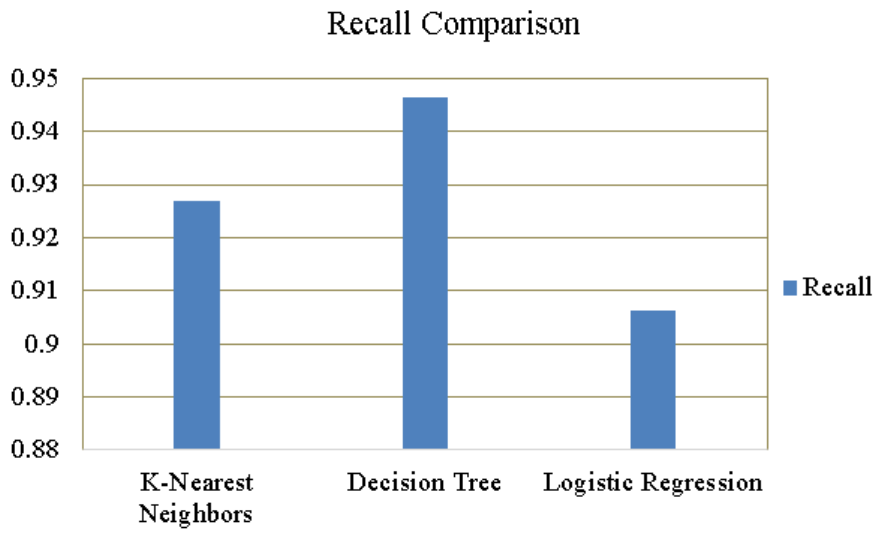


Fig. 10. Precision comparison

Figure 11 below compares the f1-score of the model k-nearest neighbours, decision tree and logistic regression. F1-score of k-nearest neighbours is 0.9270, decision tree is 0.9494, and logistic regression is 0.9064. As the bar chart shows, the highest accuracy achieved by decision tree and the lowest accuracy is logistic regression.

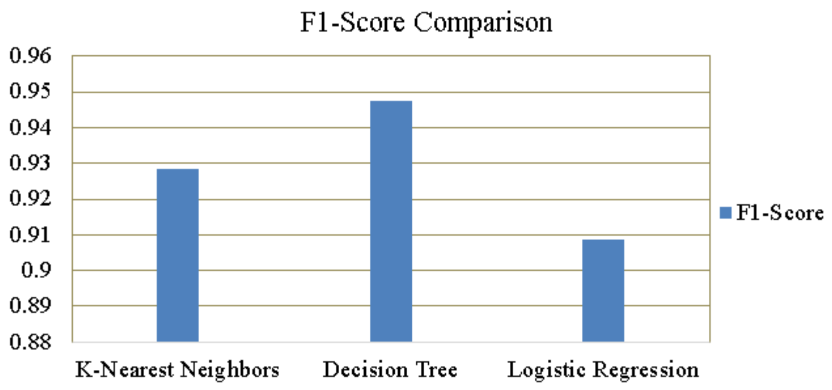


Fig. 11. F1-score comparison

Figure 12 above compares the whole evaluation model, including the model k-nearest neighbours, decision tree, and logistic regression. The k-nearest neighbours in the chart are represented by blue, the decision tree is represented by red, and logistic regression represents green. As we can see from the bar chart, a decision tree has the highest accuracy, precision, recall and f1-score compared to logistic regression and k-nearest neighbours, while Logistic regression has the lowest accuracy, precision, recall and f1-score compared to the decision tree and k-nearest neighbours.

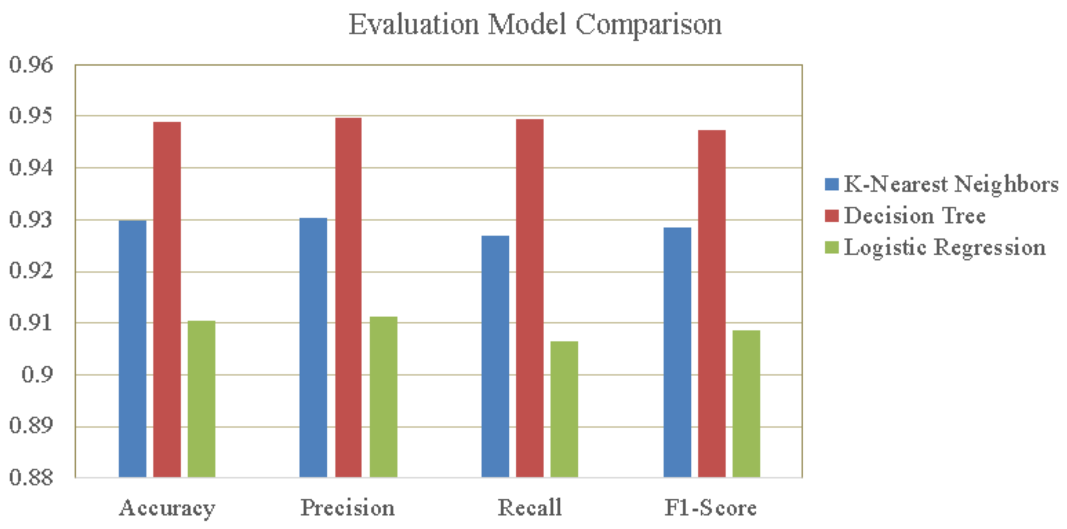


Fig. 12. Evaluation model comparison

The best classifier model for phishing detection is the decision tree, which has the highest accuracy, precision, recall and f1-score compared to others. The model can classify the phishing website with a 95% average accuracy. This shows that the model developed is robust and can perform well in real-life applications.

4. Conclusions

This research used Three classification algorithms: K-nearest neighbours, decision tree and logistic regression. The three models are compared to determine which provides the best accuracy in detecting a fake URL. The models are fine-tuned with the best parameters for each to achieve an optimal result for phishing detection. After evaluating each model, the decision trees were the most accurate in classifying phishing websites.

References

[1] Jain, Ankit Kumar, and Brij B. Gupta. "Phishing detection: analysis of visual similarity based approaches." *Security and Communication Networks* 2017 (2017). https://doi.org/10.1155/2017/5421046

[2] Aburrous, Maher, M. Alamgir Hossain, Fadi Thabatah, and Keshav Dahal. "Intelligent phishing website detection system using fuzzy techniques." In *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*, pp. 1-6. IEEE, 2008. https://doi.org/10.1109/ICTTA.2008.4530019

[3] Mahajan, Rishikesh, and Irfan Siddavatam. "Phishing website detection using machine learning algorithms." *International Journal of Computer Applications* 181, no. 23 (2018): 45-47. https://doi.org/10.5120/ijca2018918026

[4] Hoi, Steven C. H., Doyen Sahoo, Jing Lu, and Peilin Zhao. "Online learning: A comprehensive survey." *Neurocomputing* 459 (2021): 249-289. https://doi.org/10.1016/j.neucom.2021.04.112

[5] Afroz, Sadia, and Rachel Greenstadt. "Phishzoo: Detecting phishing websites by looking at them." In *2011 IEEE Fifth International Conference on Semantic Computing*, pp. 368-375. IEEE, 2011. https://doi.org/10.1109/ICSC.2011.52

[6] Moore, Tyler, and Richard Clayton. "Examining the impact of website take-down on phishing." In *Proceedings of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit*, pp. 1-13. 2007. https://doi.org/10.1145/1299015.1299016

[7] Mao, Jian, Pei Li, Kun Li, Tao Wei, and Zhenkai Liang. "BaitAlarm: detecting phishing sites using similarity in fundamental visual features." In *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, pp. 790-795. IEEE, 2013. https://doi.org/10.1109/INCoS.2013.151

[8] Abiantoro, Deandra, and Dana Sulistyo Kusumo. "Analysis of web content quality information on the Koseeker website using the web content audit method and ParseHub tools." In *2020 8th International Conference on Information and Communication Technology (ICoICT)*, pp. 1-6. IEEE, 2020. https://doi.org/10.1109/ICoICT49345.2020.9166396

[9] Chorghe, Nikita, Akshay Jain, Shraddha Mali, and Prathmesh Gunjgur. "Identity Theft Prediction Using Game Theory." In *ITM Web of Conferences*, vol. 32, p. 03022. EDP Sciences, 2020. https://doi.org/10.1051/itmconf/20203203022

[10] Chen, Kuan-Ta, Jau-Yuan Chen, Chun-Rong Huang, and Chu-Song Chen. "Fighting phishing with discriminative keypoint features." *IEEE Internet Computing* 13, no. 3 (2009): 56-63. https://doi.org/10.1109/MIC.2009.59

[11] Anti-Phishing Working Group. "Phishing Activity Trends Reports." *APWG*. Accessed March 28, 2023. https://apwg.org/trendsreports/.

[12] Mishra, Anshumaan, and Fancy Fancy. "Efficient detection of phishing hyperlinks using machine learning." *International Journal on Cybernetics & Informatics (IJCI)* 10, no. 1/2 (2021): 23-33. https://doi.org/10.5121/ijci.2021.100204

[13] Ali, Shujaat. "Phishing Attacks: Detection and Prevention." *Engineering Engrxiv Archive Preprint* (2022). https://doi.org/10.31224/2405

[14] Yahya, Farashazillah, Ryan Isaac W. Mahibol, Chong Kim Ying, Magnus Bin Anai, Sidney Allister Frankie, Eric Ling Nin Wei, and Rio Guntur Utomo. "Detection of phising websites using machine learning approaches." In *2021 International Conference on Data Science and Its Applications (ICoDSA)*, pp. 40-47. IEEE, 2021. https://doi.org/10.1109/ICoDSA53588.2021.9617482

[15] Ridho, M. Rasyid, and Hilal H. Nuha. "Application of Extreme Learning Machine (ELM) Classification in Detecting Phishing Sites." In *2022 5th International Conference of Computer and Informatics Engineering (IC2IE)*, pp. 60-64. IEEE, 2022. https://doi.org/10.1109/IC2IE56416.2022.9970191

[16] Pal, Kamalendu, and Bill Karakostas. "Software Testing Under Agile, Scrum, and DevOps." In *Research Anthology on Agile Software, Software Development, and Testing*, pp. 1059-1076. IGI Global, 2022. https://doi.org/10.4018/978-1-6684-3702-5.ch053

[17] Al-Taie, Mohammed Zuhair, Naomie Salim, and Adekunle Isiaka Obasa. "Successful Data Science Projects: Lessons Learned from Kaggle Competition." *Kurdistan Journal of Applied Research* 2, no. 3 (2017): 40-49. https://doi.org/10.24017/science.2017.3.18

[18] Devi, G. Malini, and Marlapudi Apurupa. "A CMiner Algorithm based Mining Technique to Extract Competitors for Kaggle Dataset." *International Journal of Engineering Research & Technology* 9, no. 7 (2020): 1310-1314. https://doi.org/10.17577/IJERTV9IS070567

[19] Hari, R., and A. Puce. "Data Acquisition and Preprocessing." *MEG-EEG Primer* (2017): 89-97. https://doi.org/10.1093/med/9780190497774.003.0007

[20] Kubat, Miroslav. *An Introduction to Machine Learning*. Springer Nature, 2021. https://doi.org/10.1007/978-3-030-81935-4

[21] Chen, Junde, and Defu Zhang. "Comparison analysis for classification algorithm in data mining and the study of model use." In *AIP Conference Proceedings*, vol. 1955, no. 1, p. 040173. AIP Publishing LLC, 2018. https://doi.org/10.1063/1.5033837

[22] Devi, S. Gayathri, and M. Sabrigiriraj. "Feature selection, online feature selection techniques for big data classification:-a review." In *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pp. 1-9. IEEE, 2018. https://doi.org/10.1109/ICCTCT.2018.8550928