Phishing Attack Detection on URLs' Using KNN, RF, DT With GA and K-fold Cross Validation Approach

Jun Chen Chong, Nah Yi Sim, Chia Wei Khoh, Law Teng Yi

New Era University College

DOI: https://dx.doi.org/10.47772/IJRISS.2025.9010134

Received: 30 December 2024; Revised: 05 January 2025; Accepted: 09 January 2025; Published: 07 February 2025

ABSTRACT

This research paper highlights a comprehensive study on phishing attack detection using machine learning algorithms which covers K-Nearest Neighbours (KNN), Random Forest and Decision Tree methods. Due to the ongoing rise of phishing attack, the needs for phishing attack detection method are necessary. This study used dataset downloaded from Kaggle, and then use various features to extract each URLs link retrieve from the dataset to generate another form of dataset for more successful detection. Next, employs K-Fold Cross-Validation methodology and Genetic Algorithms to optimise hyper parameter. The results show that both the Random Forest and Decision Tree models achieved perfect accuracy of 100%, while the KNN model achieved accuracy of 99.87%. The results underscore the effectiveness of machine learning techniques in enhancing phishing detection capabilities, contributing to improved cybersecurity measures.

Keywords: KNN, Random Forest, K-Fold Cross-Validation, Genetic Algorithms, Decision Tree, Phishing Attack

INTRODUCTION

In today's digital world, the connection between internet with individuals and businesses is inseparable. People spend most of their time interacting with IOTs such as smartphones, laptop, tablet to conduct various activities that include online shopping, sending emails, social media browsing, and retrieving information from the web. Due the large number of times expose to the internet, a range of online risks, including cybercrime, which is becoming a significant worry, despite all the benefits and convenience that the internet offers. Phishing is a misleading method whereby phisher fool users into giving important personal information, such as login passwords or financial details.

According to researcher, Siti Hawa Apandi definition of phishing attacks work by taking advantage of the victim's confidence. Phishers construct fake websites that closely mimic real ones in an attempt to trick users into sharing personal information. To trick users into entering their sensitive information, phishers design a website that looks like a social media platform or an online payment service. For example, once users key in their account information at the mimicked online banking website, the phisher will have to receive and collect users' input. Phishing attacks usually involve emails, ads, URLs, and malicious software, Spear phishing, email phishing, malware phishing, website phishing, and ads phishing. These fraudulent tactics have the possibility to significantly damage an organization's or an individual's finances and reputation.

Therefore, it is critical to create sophisticated detection techniques that are capable of keeping up with the constantly shifting strategies used by phishers. To protect against the increasing complexity of these attacks, traditional phishing detection techniques that depend on predefined rules and characteristics like URL structures are no longer sufficient enough. Thus, appropriate anti phishing attacks solution like machine learning where it provides a more accurate way to identify phishing attempts is required. Machine learning models able to adapt changes in unseen phishing strategies by learning from massive data sets.

This research mainly focuses on using 3 algorithm, K-NN, Random Forest, and Decision Tree to generate model

that can evaluate pre-processed phishing website URLs dataset acquire from Kaggle website based on different features to identify and detect suspicious website. Then, comparer these three algorithms according to their matrix performance to examine each algorithm strengths and weakness. [1]

Problem Statement

Through malicious URLs that take advantage of linkages in settings like web browsers, email clients, and social media platforms, phishing attacks assaults internet users. It is difficult for current URL-based detection systems to keep up with the ever-changing and dynamic nature of phishing tactics. To stop people from visiting dangerous websites, it is crucial to identify these malicious URLs early and accurately. Strong detection systems are essential for ensuring online safety since phishing attempts might happen whenever users click with shady websites.

Research Questions

1. How to filter URLs for phishing attack detection program?
2. Which algorithm provided the most accurate result?
3. What limitations in the URLs affect the performance of machine learning models for phishing detection?

Research Objectives

1. To achieve high success and accurate rate on phishing websites.
2. To compare and spot differences between each algorithm.

Research Hypothesis

When it comes to identifying phishing attempts on URLs, the effectiveness of the Decision Tree, Random Forest, and K-Nearest Neighbour algorithms will differ. Based on training and testing using K-fold cross-validation and performance measures, Random Forest should perform better than the other two in terms of accuracy, precision, and overall efficacy.

Research Scope

This project is all about taking a large set of data from Kaggle website and then do phishing attack detection on each URLs given in the csv file by generate and training model with machine learning algorithm such as decision tree, random forest, and k-nearest neighbour. In addition, Genetic Algorithm will be used as the search algorithm for finding the most optimization and decision-making across various domains. On the other hand, the methodology utilized is K-fold cross-validation. Then, compare each model result with the four-performance metrics.

Research Contribution

The goal of the research contribution to phishing attack detection is to advance cybersecurity and guarantee user have safer online interactions via through detection on URLs is phishing or legitimate website. The creation of sophisticated deep learning and machine learning models, feature engineering for spotting harmful URL patterns, and real-time detection systems that can stop consumers from interacting with phishing websites are some examples of contributions.

LITERATURE REVIEW

Introduction

According to several research papers, the phishing attack is commonly to be seen in a form of website. Besides,

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

plenty of technique has been developed on detecting phishing URLs. One of it is machines learning based approaches to find a sequence of pattern that match phishing website.

Phishing Attack Detection Technique

1. Phishing Attack*:* Phishing attack represent a common threat in cybersecurity field. It characterized by variety of techniques focus on deceiving individuals into exposing their own sensitive information. There are different types of phishing attacks like Social Engineering, Spear Phishing, Smishing, Text Phishing, Access Point Phishing, URL Phishing, Whaling, Clone Phishing, Deceptive Phishing, Drive-by Download, Phishing Kits, Denial of Service Attack, and Graphical User Interface (GUI) Phishing. [2]

2. URLs: URLs (Uniform Resource Locators) are a standardized format for specifying the location and access method of resources on the internet. A URL consists of various components, the <host> component being key as it determines the destination computer the browser contacts. The <host> is broken into subdomains, the domain, and the top-level domain (TLD), structured as <subdomain>. <domain>. <TLD>. For example, in "facebook.mobile.com," "com" is the TLD, "mobile" is the domain, and "Facebook" is the subdomain. The domain usually represents the organization's name, the TLD indicates the type or location of the organization. For example, ".gov" for government or ".com" for commercial sites. The subdomain often signifies a section of the website. Additionally, certain components of a URL, such as the <user> :< password> and <port>, are valid but less commonly seen in user-facing URLs. [3]

Figure 1: URL's Structure [3]

K-Fold Cross-Validation

K-fold cross-validation can be effectively applied to URL text in the context of phishing detection by treating the URLs as features within a dataset. In this case, the dataset would be made up of URLs that have been labelled as either legitimate or phishing. The dataset is separated into K subgroups for the K-fold cross-validation procedure, guaranteeing that each subset has a representative combination of both kinds of URLs. K-1 subsets are used to train the machine learning model, which then learns to recognise patterns and traits linked to phishing URLs, like dubious keywords, odd domain structures, or particular URL durations. The model's predictions are then verified using the remaining subset, enabling an evaluation of its precision and resilience in URL classification. To make sure that the model is not biassed towards any specific collection of URLs, this iterative procedure is repeated K times, with each URL being included in the validation set once. Researchers can get a more accurate assessment of the model's performance by using K-fold cross-validation to URL text. This will eventually result in higher detection rates for phishing attempts based on URL attributes.

The study [4] highlights the importance of cross-validation in improving model reliability and found that applying k-fold cross-validation improved the accuracy of ensemble classifiers, demonstrating the need for robust validation techniques in machine learning applications. This finding is consistent with the broader literature, which advocates cross-validation as a standard practice to mitigate overfitting and ensure generalisability.

Machine Learning Algorithms for Phishing Detection

1. K-Nearest Neighbours (KNN): KNN uses a measure of distance to classify URLs in the field of phishing detection by comparing a given URL's connection to known benign and phishing URLs. KNN is well known in classifying unlabelled data with features. It first set a number of k nearest data points called neighbours. Then, predict the result based on the distance between the query point and closest k neighbours. It is well known for its simplicity and adaptability which suitable for various dataset but limited by larger dimensions dataset when there are too many features. [5]

2. Decision Tree (DT): Using a tree-like model of decisions and their potential outcomes, decision trees are a popular machine learning technique. The approach creates stems that lead to decision nodes and leaf nodes that reflect class labels by repeatedly splitting the dataset according to feature values. In the context of detecting phishing websites, a decision tree (DT) model may be trained on a dataset of labelled phishing and legitimate websites [6]. The interpretability of decision trees is one of its main benefits; stakeholders can readily visualize and comprehend the final model. They efficiently and cohesively unite a series of basic tests, where a numeric feature is compared to a threshold value in each test [7]. Nevertheless, decision trees can overfit their models, especially if the hierarchy is deep down which might result in poor generalization on unknown data [8].

3. Random Forest (RF): By building numerous decision trees during training and generating the mode of their predictions, Random Forests improve on the Decision Tree methodology. The overfitting issue that just one Decision Trees frequently have is reduced by the ensemble approach. The algorithm's capacity to combine predictions from several trees, which enables it to identify an increased number of patterns in the data. Furthermore, Random Forests offer information on feature relevance that might direct additionally research and model improvement. The article highlights how crucial feature selection is in machine learning, highlighting how characteristics like "SSLState," "URLofAnchorExternal," and "Web Traffic" were essential to the model's performance [8]. Additionally, Random Forests have been effectively applied in phishing detection systems, demonstrating high accuracy and robustness in identifying malicious websites. For instance, a study analysing web-based phishing detection using Random Forest identified important URL features and showed improved detection performance with feature selection. Another researcher developed an anti-phishing browser based on Random Forest and rule extraction frameworks to identify webpage legitimacy. These applications underscore the versatility and effectiveness of Random Forests in cybersecurity domains [9].

Genetic Algorithms

The study [10] which has proposed GA-based feature selection method demonstrates superior performance compared to traditional methods, achieving higher recall and accuracy while significantly reducing the number of features required for effective phishing detection. The study highlights the unique ability of GAs to identify features that other methods may overlook, contributing to a more nuanced understanding of phishing detection.

Metric performance

Metric performance in phishing detection is primary calculating the accuracy, f1 score, recall, and precision on how successful the detection was. Among all of them it utilizes four basic term or formula which are TP (True Positive), TN (True Negative), FP (False Positive), TN (True Negative).

$$\mathrm{ACC} = \frac{\#\,\text{correctly classified samples}}{\#\,\text{all samples}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}$$

Figure 2: Formula Accuracy [11]

$$\text{REC} = \frac{\#\,\text{true positive samples}}{\#\,\text{samples classified positive}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Figure 3: Formula Recall [11]

$$\text{PREC} = \frac{\#\,\text{samples correctly classified}}{\#\,\text{samples assigned to class}} = \frac{\text{TC}}{\text{TC} + \text{FC}},$$

Figure 4: Formula Precision [11]

$$\text{F1} = 2 \times \frac{\textit{precision} \times \textit{recall}}{\textit{precision} + \textit{recall}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}$$

Figure 5: Formula F1 Recall [11]

Machine Learning Based Approach with Different Algorithm and Methodology

In the study by [12], Authors used a dataset of phishing webpages from Mendeley's online repository, which included 10,000 websites with binary class labels ('0' for phishing and '1' for legitimate), evenly divided between phishing and legitimate categories. They used feature extraction techniques like Principal Component Analysis (PCA) to reduce dimensionality while maintaining critical variance, Pearson and Shapiro Ranking to mathematically rank features according to their correlation with the target variable of interest, and Parallel Coordinates to visualize highly dimensional data and determine patterns in order to improve the performance of machine learning models. The authors used Random Forest, an ensemble technique that use multiple decision trees and Gini significance to evaluate features, and K-Nearest Neighbours (KNN), which classifies based on the majority class of nearest neighbours. The result from this research, RF hold discrimination threshold of 0.48, with precision, recall, and F1 score around 0.99. While KNN hold discrimination threshold of 0.50, with precision, recall, and F1 score around 0.82 to 0.89.

In the study by [8], Authors addresses the shortcomings of traditional strategies like blacklists and rule-based systems by using machine learning methods, particularly K-Nearest Neighbours (KNN), Decision Trees, and Random Forests, to improve the identification of phishing websites. The study shows that Random Forests had the greatest accuracy of 97%, followed by Decision Trees at 96% and KNN at 94%, using a dataset of 6,157 benign URLs and 4,898 phishing URLs with 30 different features. Metrics including precision, recall, and F1-score were used to assess the models. Random Forests demonstrated greater precision for phishing URLs with a recall of 0.95 and 0.97, demonstrating its resilience in detecting phishing attempts. The research also emphasizes the significance of some variables, especially "SSL_State," which became a crucial factor in the classification process, confirming the effectiveness of using machine learning based methods to enhance and automate phishing detection tactics.

In the study paper [13], authors employ the Random Forest, Decision Tree, and K-Nearest Neighbours (KNN) algorithms to assess how well they identify phishing websites. Using a dataset of 4,898 phishing websites and 6,157 authentic websites, they do a comprehensive comparative analysis, evaluating performance using metrics including accuracy, precision, recall, and F1 score. 10-fold cross-validation, which splits the dataset into ten sub-samples to enable reliable testing and modelling, is part of the process used for evaluating model performance.

The results highlight the advantages and disadvantages of each approach, eventually promoting the employment of ensemble techniques Random Forest in particular to improve detection capacities in the continuous fight against phishing attempts.

In the study paper [14], author use layered classification model for identifying phishing websites by utilizing the Random Forest and Decision Tree algorithms. According to the results of the study, the Decision Tree successfully distinguished between phishing and legitimate URLs, achieving an accuracy of 91% during the training phase and 90.4% during the testing phase. On the other hand, the Random Forest algorithm demonstrated its strong performance with training accuracy of 90.7% and testing accuracy of 90.6%. Overall, although the Decision Tree fared somewhat better than the Random Forest in terms of accuracy during testing, these results demonstrate the potential of both algorithms to improve phishing detection skills.

According to [15], have mentioned that the growing cybersecurity risks brought on by digital transformation, especially the increase in phishing attempts intended to steal private user information. In the research present about phishing attack detection using machine learning approach such as Random Forest and Decision Tree. The model utilized a phishing dataset from Kaggle and employed feature selection techniques such as Principal Component Analysis (PCA) to analyse the dataset's attributes. In the result of the research Random Forest algorithm achieved a maximum accuracy of 97%. The researchers evaluate performance using a confusion matrix, precision, recall, and F1 scores, finding that RF's ensemble approach yields superior results with lower variance and reduced overfitting.

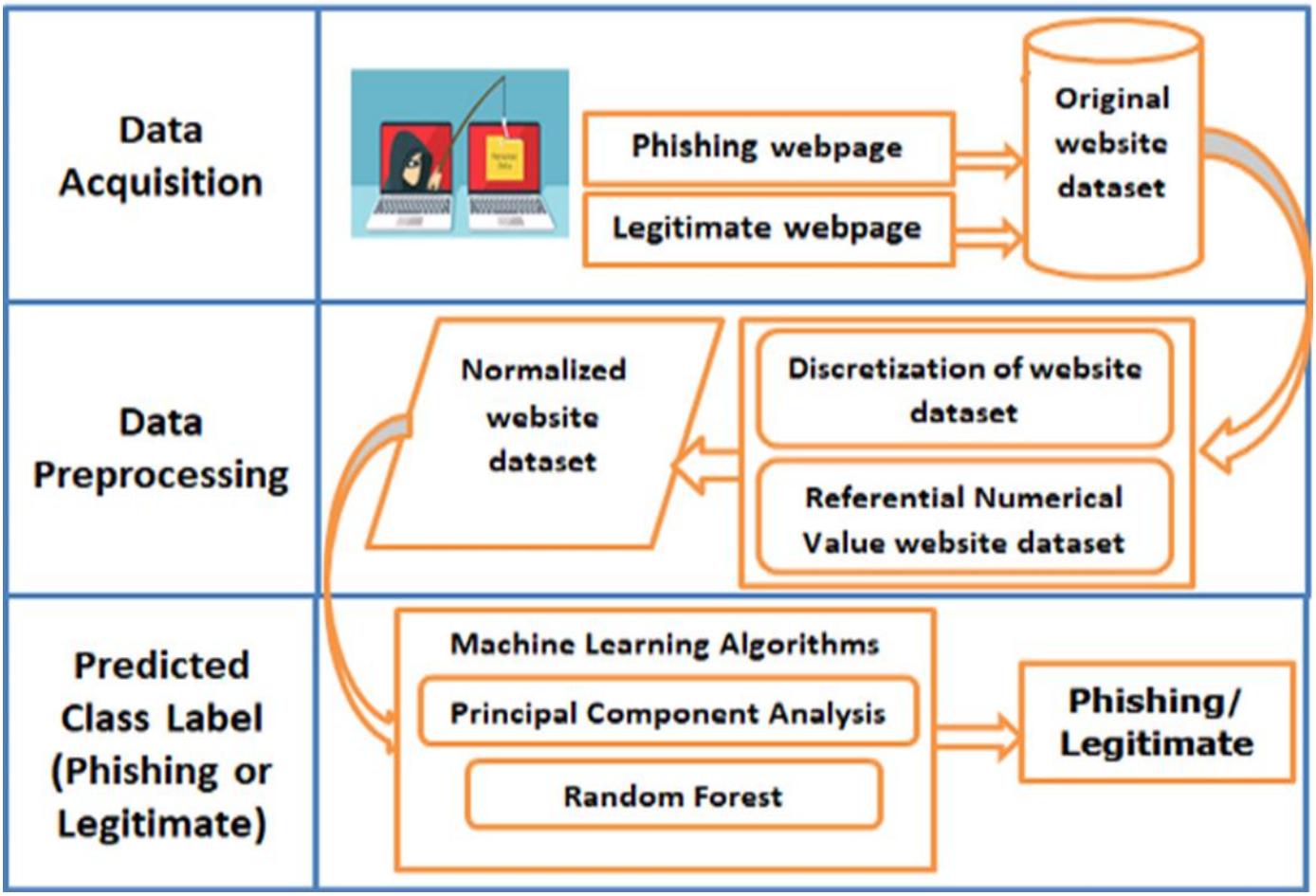

Figure 6: The proposed framework for phishing detection [15]

According to [6], phishing is the easiest method for gathering sensitive information from unwary people. To distinguish between legal and phishing URLs, the paper introduces using a machine learning approach for phishing attack detection such as Extreme Gradient Boosting, Decision Tree, Logistic Regression, Random Forest (RF), and Support Vector Machine. Two datasets were used: Dataset 1 (PhishTank) contained 10,000 URLs, evenly split between 5,000 phishing and 5,000 legitimate URLs, while Dataset 2 (UCI) consisted of

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

11,055 URLs, with 6,157 phishing and 4,898 legitimate examples. The K-fold, feature selection, and hyperparameter tuning methods were used to determine the models' accuracy on the datasets. Among the algorithms, RF provided an accuracy of 98.80% and 97.87% on the PhishTank dataset and UCI. Furthermore, on the PhishTank dataset, RF achieved the highest AUC-ROC value of 99.89% and the maximum precision, recall, and F1-score 99% each. The paper has found that machine learning algorithms yield high results from other state-of-the-art methods.

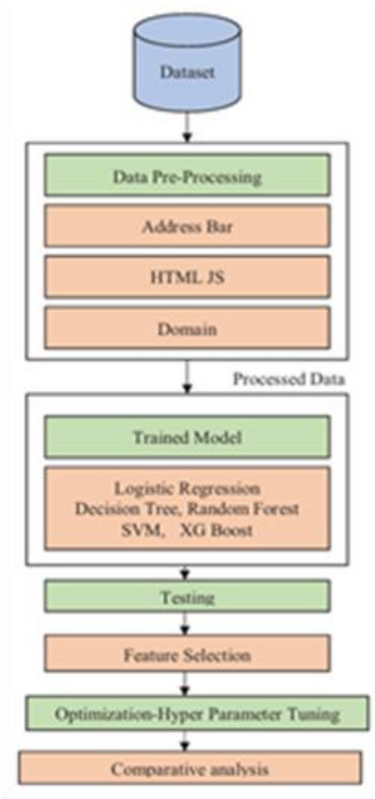

Figure 7: Proposed System Flow [6]

The study [4] addresses the critical issue of detecting phishing websites using machine learning techniques. The study uses two different datasets, one from the UCI Machine Learning Repository with 2,456 instances and another from Kaggle with 11,055 instances, both with 30 URL attributes and a result attribute indicating the phishing status. The authors conduct their analysis in three phases: first, they evaluate basic classifiers such as Logistic Regression, Decision Trees, and K-Nearest Neighbours; second, they apply ensemble classifiers including Bagging, Adaboost, and Boost; and finally, they evaluate the performance of these classifiers with and without k-fold cross-validation. The results show that ensemble classifiers significantly outperform basic classifiers, with the Extra Trees and Boost classifiers achieving accuracy rates of over 99% on the datasets. The study highlights the importance of cross-validation in improving model reliability and suggests that machine learning can play a vital role in combating phishing threats.

The study [16] presents a novel method for detecting phishing URLs using Long Short-Term Memory (LSTM) networks and a stacked generalization ensemble framework. The model uses multiple base classifiers, such as Decision Trees, Random Forest, and K-Nearest Neighbours (KNN), to form the first layer of prediction. These models individually analyse the dataset, interpreting hierarchical feature relationships, reducing overfitting, and capturing proximity-based classification patterns. The outputs are passed to an LSTM meta-learner, synthesizing predictions into a final decision. The model achieved a detection accuracy of 98.76% on benchmark phishing datasets, outperforming individual classifiers and outperforming traditional ensemble methods. The study underscores the importance of combining traditional machine learning algorithms with deep learning frameworks for cybersecurity challenges.

The study [17] explores the use of advanced machine learning techniques, including ensemble algorithms to improve phishing website detection. It also compares traditional classifiers like Random Forest, Decision Tree, and K-Nearest Neighbours (KNN) for comparison. Random Forest and Decision Tree models showed strong

performance in classifying phishing and legitimate URLs, while Decision Tree was praised for its simplicity and interpretability. Random Forest excelled in robustness and resistance to overfitting due to its ensemble nature. KNN showed strength in proximity-based classification but required careful hyperparameter tuning. AdaBoost, the highest-performing ensemble learning model, achieved an impressive ROC AUC score of 99%, highlighting the advantage of ensemble methods in achieving high accuracy and reliability.

The study [10] addresses the growing threat of phishing attacks, which have been increasing at a rate of over 150% per year, requiring effective detection systems. The authors propose a novel feature selection method using genetic algorithms (GAs) to optimise the identification of relevant features from a dataset of URLs, thereby improving the performance of machine learning models in detecting phishing attempts. The importance of feature selection as a pre-processing step to improve model accuracy, reduce computational cost and minimise training time highlighted in the study. The results of the study show that the proposed GA-based feature selection method significantly outperforms traditional methods such as Removing Features with Low Variance (RFLV), Recursive Feature Elimination (RFE), Sequential Feature Selection (SFS), and SelectFromModel (SFMFS) across various machine learning classifiers. Specifically, the Random Forest classifier achieved an accuracy of 92.93% and a recall of 89.05% using the selected features, while Boost achieved an accuracy of 91.53% and a recall of 87.24%. The GA method reduced the number of features from 73 to 44, resulting in a 36.62% reduction in inference time, from 1.0066 seconds to 0.6380 seconds. This reduction in computational cost, combined with improved performance metrics, highlights the effectiveness of the GA approach in improving phishing detection systems. The results suggest that GAs can effectively navigate the complexities of feature selection in high-dimensional datasets, making them a valuable tool in cybersecurity applications.

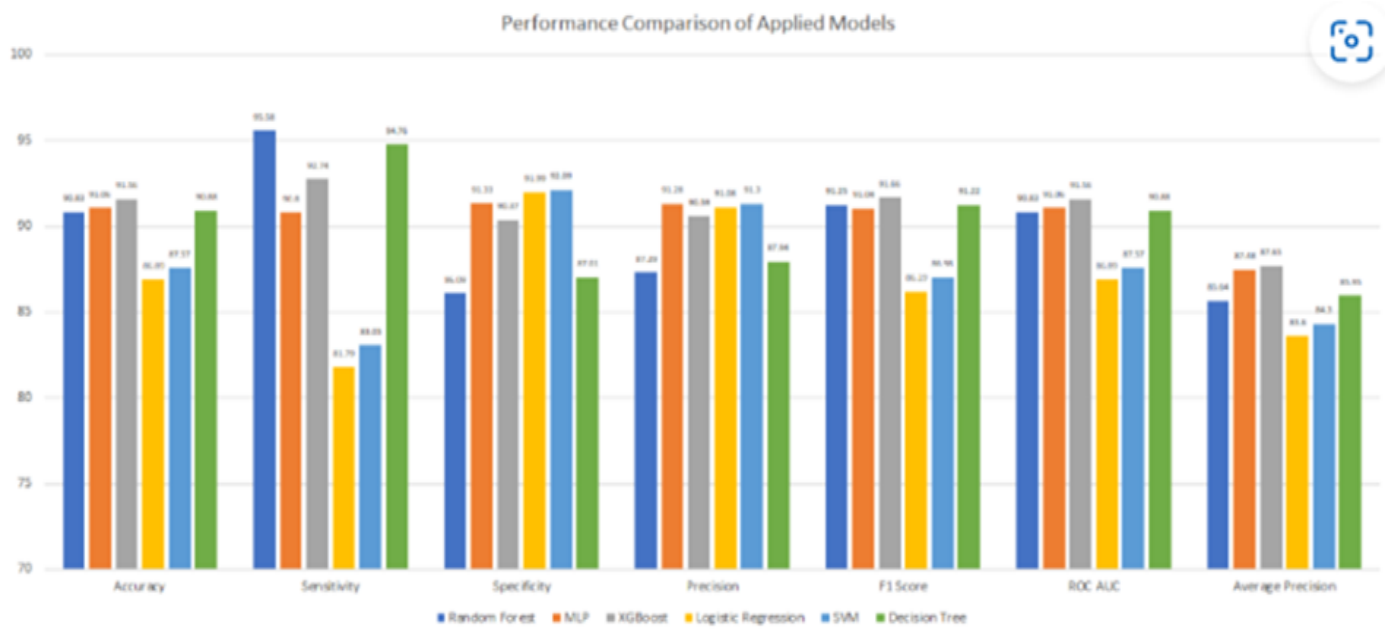

Figure 9: Performance Comparisons of Applied Models. [14]

The study [18] addresses the vital issue of phishing attacks in the internet business sector, specifically e-commerce and banking. The study uses a dataset of 1,353 samples with ten features related to URL classification that was obtained from the Kaggle website phishing data repository. The study actually analyses the model's performance using the K-Nearest Neighbour (KNN) method in a Python environment, finding that setting K to 10 yields the best accuracy of 87.82%. With a confusion matrix revealing only 7 misclassified observations out of 106 test cases, the results show that the model is successful in differentiating between phishing and legitimate URLs, indicating its potential as a strong tool for boosting online security against phishing threats.

The study [19] provides a comprehensive overview of existing methods and approaches to detect phishing attacks, emphasizing the evolution of cyber threats alongside technological advancements. It highlights the inadequacies of traditional techniques, such as blacklist-whitelist methods, which fail to address zero-day attacks effectively. The review discusses the application of supervised machine learning methods, which are well-suited for classification problems like phishing detection. It references various studies that have explored different

machine learning algorithms, including Random Forest, K-Nearest Neighbours (KNN), and Logistic Regression, to classify phishing and legitimate websites based on URL features and other characteristics. From the study, the random forest classifier performed with a precision of 97%, a recall 99%, and F1 Score is 97%.

The study [20] offers a fresh strategy for combating the growing danger of phishing attacks, which take advantage of people by posing as trustworthy websites in order to steal private data. In order to create a more successful sub-forest for phishing detection, the study presents an Optimised Decision Forest (ODF) method that uses a genetic algorithm (GA) to pick optimal and diverse individual trees. Three separate phishing datasets from the UCI repository are used to train and test the ODF using a variety of performance metrics, including accuracy, F-measure, Area Under the Curve (AUC), and Matthews Correlation Coefficient (MCC). The findings show that the ODF performs significantly better than a number of baseline classifiers, with low false-positive rates and detection accuracies of 98.37%, 97.26%, and 98.34% across the datasets. These results highlight the ODF method's potential as a strong way to improve website phishing detection and overcome the shortcomings of current anti-phishing tactics.

Comparison of KNN, RF, DT

Different performance features are revealed when the Decision Tree and Random Forest models are compared in the framework of identifying phishing attacks. The Decision Tree classifier's ability to distinguish between phishing and authentic websites using extracted features was demonstrated by its 91% training and 89.6% testing accuracy. On the other hand, the Random Forest model, which makes use of an ensemble of many decision trees, achieved testing and training accuracy of 89.8% and 90.8%, respectively. Although the accuracy of both models was comparable, Random Forest's ensemble technique offers superior generality and robustness, which makes it a popular option for many applications, especially when overfitting is an issue. [14]

Different performance features are revealed when K-Nearest Neighbours (KNN) and Random Forest are compared in the context of phishing website identification. For both phishing and genuine websites, KNN, which measures the distance between data points, obtained an Area Under the Curve (AUC) of 0.94, suggesting a moderate degree of prediction accuracy. On the other hand, the Random Forest classifier performed better, attaining perfect classification accuracy with an AUC of 1.00 for both classes. While KNN's performance is constrained by its dependence on local data points without the advantage of combined learning, Random Forest's ensemble approach which mixes numerous decision trees to enhance prediction reliability that highlights the algorithm's efficacy in handling complex datasets. [12]

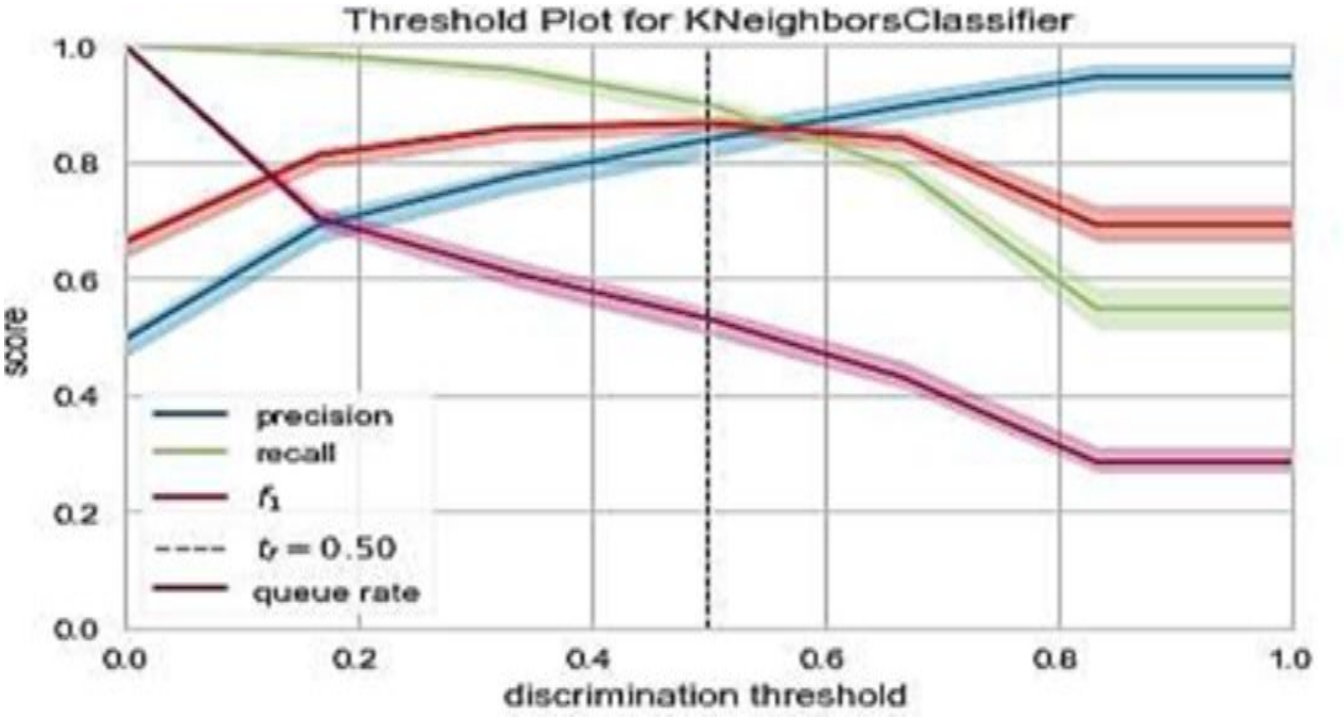

Figure 10: Threshold Plot for K Neighbors Classifier[12]

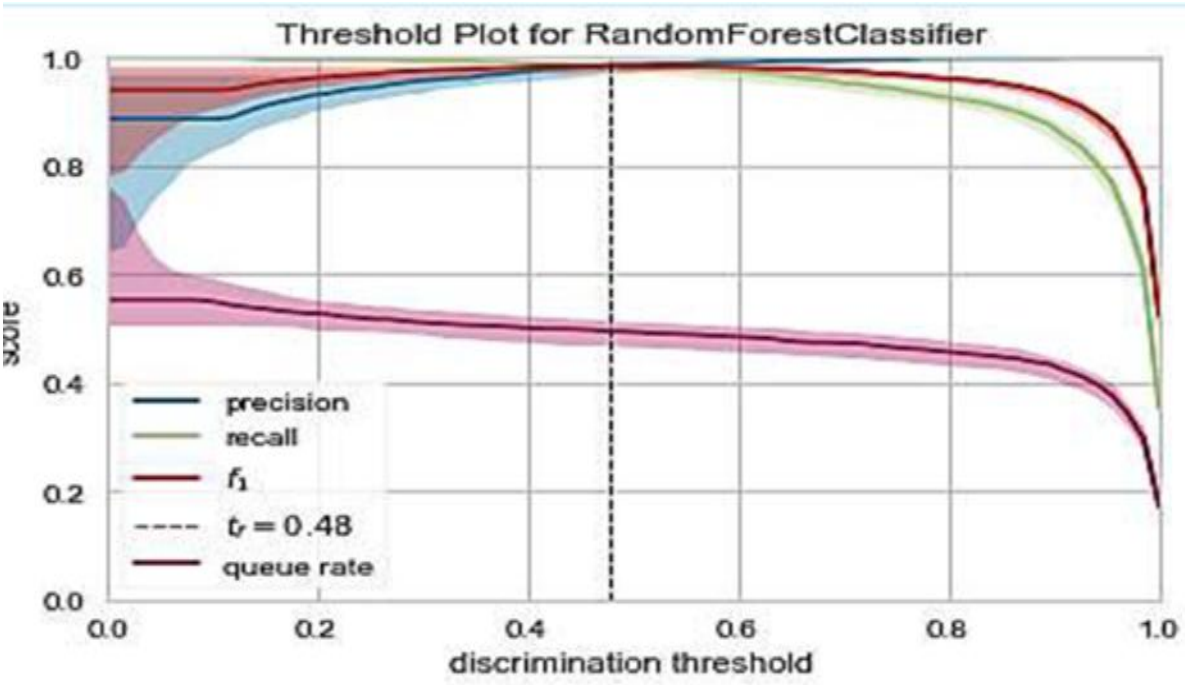

Figure 8: Threshold Plot for Random Forest Classifier.[12]

The study assesses the effectiveness of K-Nearest Neighbours (KNN), Decision Trees, and Random Forests in identifying phishing websites using a number of indicators. KNN performed dependable but somewhat less well than the other models, with a total accuracy of 94% and precision score of 0.94 for both benign and phishing URLs. With precision scores of 0.95 for phishing and 0.96 for benign URLs, Decision Trees showed its efficacy in categorization with a higher accuracy of 96%. Both were surpassed by Random Forests, which achieved the greatest accuracy of 97% and a precision score of 0.97 for phishing URLs and 0.96 for benign URLs. In a nutshell, The Random Forest model was the most successful of the three models since it demonstrated better recall and F1-scores, demonstrating its resilience and dependability in differentiating between phishing and legitimate URLs. [8]

Metric	SVM	KNN	Decision Trees	Random Forest
Precision (Class -1)	0.95	0.94	0.95	0.97
Precision (Class 1)	0.94	0.94	0.96	0.96
Recall (Class -1)	0.92	0.92	0.95	0.95
Recall (Class 1)	0.97	0.95	0.96	0.98
F1-score (Class -1)	0.94	0.93	0.95	0.96
F1-score (Class 1)	0.95	0.95	0.96	0.97
Overall Accuracy	0.95	0.94	0.96	0.97

Figure 11: Model Performance [8]

The most successful method was Random Forest, which had an accuracy of 97.27%. Its resistance against overfitting and computational efficiency are the reasons behind this. Despite its effectiveness, the Decision Tree classifier's lower accuracy of 96.60% demonstrated its limits in challenging classification tasks. Despite being part of the analysis, K-Nearest Neighbours had a 95.28% accuracy rate but struggled with scalability and sensitivity to distance measures. According to the results, ensemble techniques like Random Forest are very useful for phishing detection. To increase detection rates, the authors recommend more research into these models and feature engineering. [13]

classifier	train time (s)	test time(s)	accuracy	recall	precision	F1 score
logistic regression	0.080971	0.006414	0.926550	0.943968	0.925700	0.934704
decision tree	0.021452	0.003737	0.965988	0.971414	0.967681	0.969531
random forest	0.436126	0.021941	0.972682	0.981484	0.969852	0.975622
ada booster	0.336519	0.016766	0.936953	0.954362	0.933943	0.944032
KNN	0.112972	0.353562	0.952780	0.962968	0.952783	0.957827
neural network	9.088517	0.006925	0.969879	0.978723	0.967605	0.973112
SVM_linear	1.647538	0.053979	0.927726	0.945592	0.926268	0.935779
SVM_poly	1.048257	0.074207	0.949254	0.968816	0.941779	0.955083
SVM_rbf	1.341540	0.103329	0.952149	0.968815	0.946580	0.957543
SVM_sigmoid	1.344607	0.109696	0.827498	0.846515	0.844311	0.845305
gradient boosting	0.891888	0.005298	0.948621	0.962481	0.946234	0.954260
XGBoost	0.506072	0.006237	0.983235	0.981047	0.987235	0.976802

Figure 12: Classification Results for Different Methods [13]

The study's findings show that the Random Forest (RF) and Decision Tree (DT) algorithms perform noticeably differently in phishing detection. The Random Forest algorithm achieved an impressive accuracy of 98.80% on the PhishTank dataset and 97.87% on the UCI dataset, along with a precision, recall, and F1-score of 99% each on the PhishTank dataset. The Decision Tree method, on the other hand, showed lower performance metrics, with an accuracy of 96.58% and a propensity to overfit, which can result in decreased effectiveness on unseen data, despite being simpler and easier to understand. This comparison shows that the RF algorithm's ensemble method greatly improves its accuracy, precision, recall, and F1-score, making it a more robust and dependable option for phishing attack detection, even though the DT technique is easier to grasp. [6]

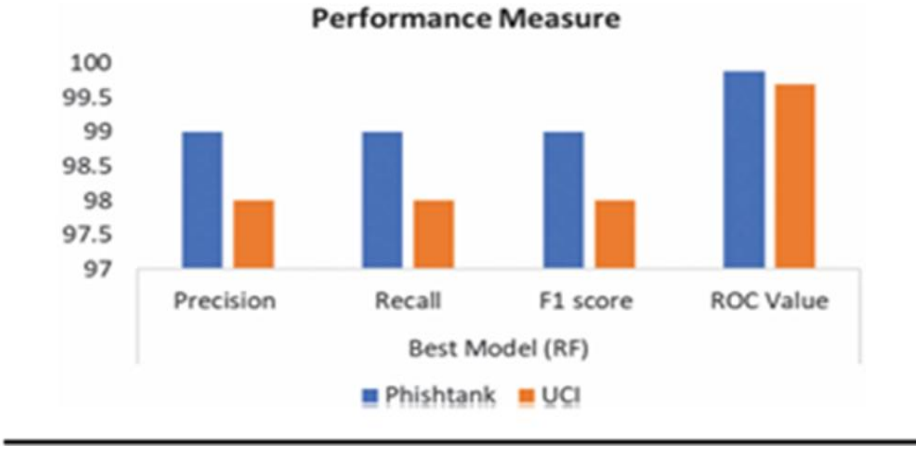

Figure 13: Comparative Performance Analysis-Phish tank And Uci Dataset [6]

The study's findings show that while both the Random Forest (RF) and Decision Tree (DT) algorithms are useful for detecting phishing attacks, their performance metrics vary greatly. With an F1 score of 90.8%, recall of 93.2%, precision of 88.5%, and accuracy of roughly 93.5%, the Decision Tree method performed well. The Random Forest algorithm, on the other hand, performed better than the Decision Tree, attaining a 97% accuracy rate, 96.9% precision, 96.5% recall, and 96.7% F1 score. These measurements show that the Random Forest's ensemble technique produces better accuracy and a better mix between precision and recall, making it more resilient against overfitting and more dependable for phishing classification, even though the Decision Tree offers respectable performance. [15]

ML algorithm	Accuracy	Recall	Precision	F1 Score
DT	0.9194	0.9384	0.8804	0.9084
RF	0.9696	0.4216	0.9689	0.5874

Figure 11: Comparison of result DT and RF [15]

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

METHOD AND MATERIALS

A set of URLs phishing attack dataset from kaggle.com which contain 235795 data from [21].

Procedure Detail

First go to any online platform database like Kaggle and obtain any relevant data based on the research title for us is phishing attack URL. The dataset used is the PhiUSIIL Phishing URL Dataset which consists of 235,795 entries, including 134,850 legitimate URLs and 100,945 phishing URLs. The features in the dataset are extracted from both the source code of webpages and the URLs themselves. These features include metrics such as URL length, domain attributes, and various URL-based similarity scores which are critical for identifying between phishing and legitimate URLs.

Secondly, conduct data processing before training model. Thus, the model are train in a formatted data which mean the URLs list from the dataset will be extract out with predefined features like URL length mention in [8].

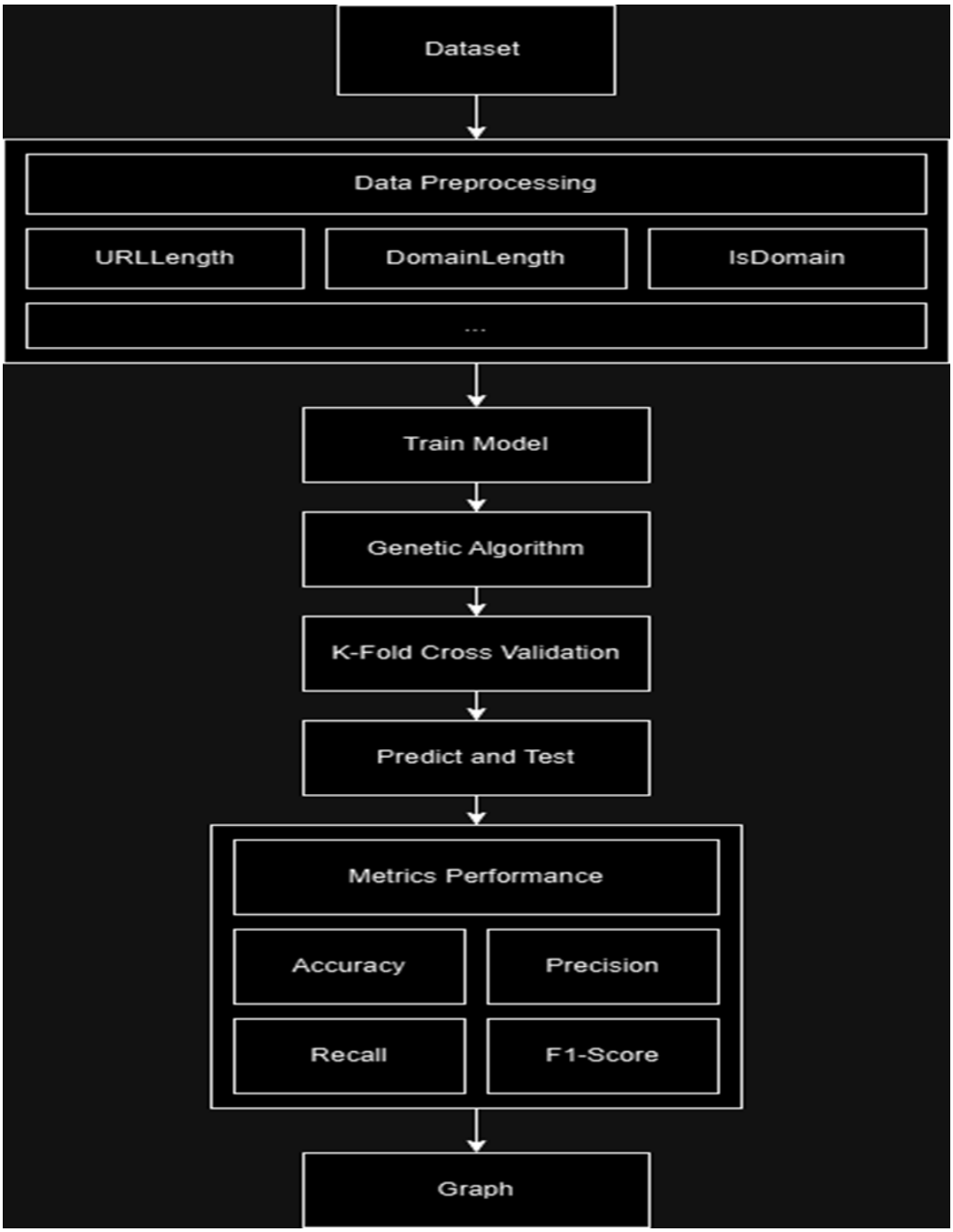

Figure 12: Procedure in Logical Order

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

However, not all features will be taken into account when majority of a features value is missing.

After extracting the data another new set of data will be generated. Then, train a model with 0.8 of the new data set and leave 0.2 for the testing. Using this splitting method is to strike a balance between having enough data to train the model and ensuring reliable evaluation. With 80% of the data used for training, the model has enough samples to understand the patterns and features required to recognise phishing attacks. The remaining 20% will be used for testing, which offers an unbiased assessment of the model's generalisability to new, untested data. In addition, make sure there is no NaN value or overfit in the dataset. Besides, which column will be x-axis or y-axis must be considered when training the model and make sure the same axis must use the same data type. For example, if x-axis displays integer value, then the column drop under x-axis must only consist of integer value.

Search algorithm is applied right after trained model phase for finding the best configuration or hyperparameter for the best prediction result. Genetic algorithm is applied one by one for each algorithm. In Genetic Algorithms, it consists of 8 components and they run in a sequential order which also indicate how to apply Genetic Algorithm. The 8 components are Initialization, Fitness Selection, Selection Operator, Crossover Operator, Mutation Operator, Local Optimization, Elitism, and Termination Criteria.

Initialization	Set a number of populations will be used for finding best solution. each population represent the combination of value. In this case, population value is the configuration hyperparameter of KNN, RF, and DT. In coding, number is assigned to n = 20.
Fitness Selection	Fitness is to show how well each solution or gene is by outputting a fitness value that calculated by the weight assigned for each hyperparameter. The hyperparameter use in the coding is the neighbours for Knn, estimators for RF, and max_depth for DT.
Selection Operator	Selection is about selecting a number of parents depend on their fitness value for crossover and mutation. In coding, Tournament Selection is selected due to balance selection are made compare to other method and 3 individuals will be compare and select the highest fitness value one.
Crossover Operator	Crossover is using selected 2 parent to create new offspring. Crossover also involve several techniques on how it creates the offspring. In coding, Blend Crossover is selected because of continuous of optimizing knn neighbours, RF estimators, and DT max_depth. This technique is taking 2 parent and blend 2 parent to make a new offspring value.
Mutation Operator	Mutation is randomly changing any gene. In coding, Uniform Integer Mutation is selected where one or more genes will be selected and remain them the same while other gene randomly selected for change.

5) K-Fold Cross Validation: K-Fold Cross Validation used for the prediction and testing along with the best hyperparameter for each algorithm from the search algorithm. The dataset was split into five equal-sized folds (k=5) in this step, and each model was trained on four of the folds while testing was done on the remaining fold. This procedure was repeated five times to ensure that each fold was used as the test set just once. Before applying cross-validation, the hyperparameters of each model were optimized using a genetic algorithm, which identified the best number of neighbours for KNN, the optimal number of estimators for Random Forest, and the ideal maximum depth for the Decision Tree. To evaluate the model's performance, the accuracy score was determined for each fold, and the mean accuracy and standard deviation for all folds were calculated.

This method gives a comprehensive evaluation of the model's performance by testing it on several subsets of the data. This ensures that no single train-test split will skew the evaluation. The cross-validation procedure

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

thoroughly tests the model's generalisation capabilities by using each fold as a test set only once. This helps in predicting the model's performance on unknown data. After hyperparameter tuning confirms the robustness of the optimised parameters, it ensures that the accuracy gain is stable across multiple data splits and not simply specific to a particular train-test split. The procedure reduces the risk of overfitting by training and assessing the model on various dataset segments several times. Both the mean accuracy and standard deviation provide precise information about the model's consistency and stability.

Next, calculate the metrics *performance* based on the prediction result and test in terms of Recall, accuracy, precision and f1 score. *After* calculation on the metrics performance, generate visualize graph like bar chart as the result of the project.

Model optimization and Evaluation

Genetic Algorithms (GA) are preferred over traditional optimization techniques for solving high-dimensional problems with complex search spaces. They prevent local optima by enabling the simultaneous investigation of several options. GAs is effective at selecting features, which lowers computational expenses and raises accuracy measurements. In one project, GA improved detection accuracy by optimizing KNN, Decision Tree, and Random Forest hyperparameters. To make sure the models were reliable and resilient, K-fold cross validation was employed. This method reduces the possibility of overfitting and offers a thorough assessment of the models' performance across various data splits by repeatedly splitting the dataset into training and testing subsets. Because of its ability to withstand overfitting and handle huge datasets with plenty of characteristics, Random Forest (RF) was selected. Several Decision Trees are used in its ensemble learning approach, which yields excellent accuracy metrics (100 percent in this research). RF's usefulness for phishing detection jobs is improved by its capacity to determine feature significance. The Decision Tree model was chosen for its simplicity and interpretability, making it an ideal choice for explaining classification results to stakeholders. However, it has limitations like overfitting, but using Genetic Algorithms for depth optimization ensures 100% accuracy. KNN is a useful and efficient approach for datasets with clearly defined clusters, offering a simple and straightforward classification rule, despite slightly lower accuracy than RF and DT. To make sure the models were reliable and resilient, K-fold cross-validation was employed. This method reduces the possibility of overfitting and offers a thorough assessment of the models' performance across various data splits by repeatedly splitting the dataset into training and testing subsets

RESULTS

```
Running Genetic Algorithm for KNN Optimization...
gen     nevals
0       20
1       13
2       9
3       14
4       15
5       11
6       12
7       10
8       12
9       14
10      6
Optimized Hyperparameter: 1
Optimized KNN Accuracy: 0.9986852986704553
```

Figure 16: Output of Optimise KNN Hyperparameter using Genetic Algorithm

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

Output from Google Colab

Figure 16 shows how the Genetic Algorithm (GA) optimisation for K-Nearest Neighbours (KNN) determined the ideal hyperparameter as k=1, which reflects the number of neighbours examined for classification. With a 99.87% accuracy rate, the model shows that the closest neighbour approach works quite well for this dataset. The high accuracy shows that KNN can efficiently categorise the data with little error when the hyper parameter is optimised.

Figure 17 shows how the GA optimisation for Random Forest determined the ideal hyper parameter (estimators) which gets the result with 115. The model reached a perfect accuracy of 100% with this setup. This outcome demonstrates how well the Random Forest model uses its ensemble learning methodology to minimis overfitting while capturing complicated patterns in the dataset.

```
Running Genetic Algorithm for RF Optimization...
/usr/local/lib/python3.10/dist-packages/deap/cre
  warnings.warn("A class named '{0}' has already
/usr/local/lib/python3.10/dist-packages/deap/cre
  warnings.warn("A class named '{0}' has already
gen     nevals
0       20
1       10
2       12
3       15
4       18
5       10
6       12
7       9
8       12
9       10
10      13
Optimized Hyperparameter: 115
Optimized RF Accuracy: 1.0
```

Figure 17: Output of Optimise Random Forest Hyperparameter using Genetic Algorithm

Figure 18 shows how the Genetic Algorithm optimised the Decision Tree (DT) model by selecting a maximum tree depth of 13. The model was able to reach 100% accuracy according to this setup. The outcome shows that the model is capable of capturing the complexity of the dataset without overfitting at a depth of 13. The Decision Tree can efficiently classify all samples while maintaining an appropriate level of model complexity to perform well when generalising to new data according to the optimised hyper parameter.

```
Running Genetic Algorithm for DT Optimization...
/usr/local/lib/python3.10/dist-packages/deap/cre
  warnings.warn("A class named '{0}' has already
/usr/local/lib/python3.10/dist-packages/deap/cre
  warnings.warn("A class named '{0}' has already
gen     nevals
0       20
1       8
2       6
3       11
4       12
5       12
6       12
7       16
8       7
9       5
10      15
Optimized Hyperparameter: 13
Optimized DT Accuracy: 1.0

Final Results:
Optimized KNN Accuracy: 0.9986852986704553
Optimized RF Accuracy: 1.0
Optimized DT Accuracy: 1.0
```

Figure 18: Output of Optimise Decision Tree Hyperparameter using Genetic Algorithm

Figure 19 shows the performance of the optimised K-Nearest Neighbours (KNN) model, Random Forest (RF) model, Decision Tree (DT) model and was evaluated using K-Fold Cross-Validation. The KNN model's strong consistency across several data splits was demonstrated by the findings, which showed a mean accuracy of 99.86% with a standard deviation of 0.0001. The reliability and consistency of the optimised KNN model for this dataset are confirmed by the low standard deviation, which indicates little variation in performance.

```
Performing K-Fold Cross-Validation for Optimized KNN...
KNeighborsClassifier (Optimized) - Mean Accuracy: 0.9986, Std Dev: 0.0001

Performing K-Fold Cross-Validation for Optimized Random Forest...
RandomForestClassifier (Optimized) - Mean Accuracy: 1.0000, Std Dev: 0.0000

Performing K-Fold Cross-Validation for Optimized Decision Tree...
DecisionTreeClassifier (Optimized) - Mean Accuracy: 1.0000, Std Dev: 0.0000

Final Results with K-Fold Cross-Validation:
Optimized KNN - Mean Accuracy: 0.9986, Std Dev: 0.0001
Optimized RF - Mean Accuracy: 1.0000, Std Dev: 0.0000
Optimized DT - Mean Accuracy: 1.0000, Std Dev: 0.0000
```

Figure 19: Evaluate performance of Optimised KNN, RF, and DT using K-fold Cross Validation

Besides that, the optimised Random Forest (RF) model achieved a perfect mean accuracy of 100% with a standard deviation of 0.0000. This outcome demonstrates how well the Random Forest model uses its ensemble learning methodology to minimise overfitting while capturing complicated patterns in the dataset.

The optimized Decision Tree (DT) model also achieved a perfect mean accuracy of 100%, with a standard deviation of 0.0000. This outcome shows that the DT model can correctly categorise every sample in the dataset when its depth is optimised. The model's reliability is confirmed by the zero-standard deviation, which shows constant performance across all validation folds.

Bar Graph

The bar chart compares the accuracy of three optimized machine learning algorithms which are K-Nearest Neighbours (KNN), Random Forest, and Decision Tree. All three algorithms show high accuracy, with KNN achieving a perfect score of 1.0. Further evaluation using metrics like precision, recall, or F1-score could provide deeper insights.

The bar chart compares the precision of three optimized three machine learning algorithms: K-Nearest Neighbour (KNN), Random Forest, and Decision Tree. All three achieved precision values close to 1.0, indicating high accuracy in classifying positive instances and minimizing false positives for the dataset. The third bar chart compares the recall of K-Nearest Neighbour (KNN), Random Forest, and Decision Tree, showing they, all achieve a recall of 1.0, indicating successful optimization in maximizing recall performance, crucial for applications requiring minimal false negatives.

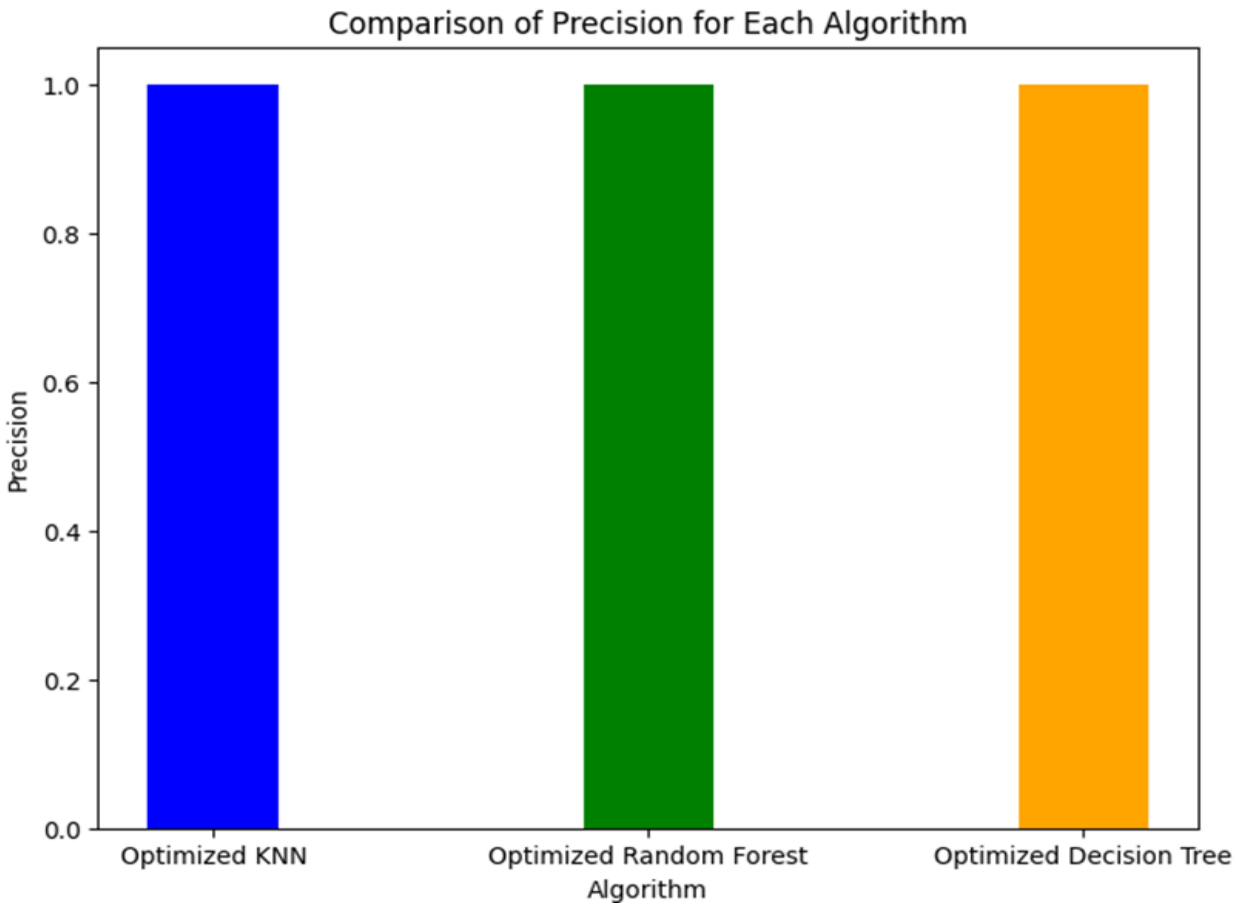

Figure 20.: Comparison of Precision for Each Algorithm

This bar chart compares the F1 Score of three algorithms, all achieving a perfect balance of precision and recall, indicating their effectiveness and well-tuned for minimizing false positives and false negatives, thereby proving their effectiveness in the task at hand. Both Random Forest (RF) and Decision Tree (DT) achieved perfect accuracy (100%) during both optimization and K-Fold Cross-Validation while K-Nearest Neighbours (KNN)

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume IX Issue I January 2025

achieved slightly lower accuracy (99.87%) during optimization and a mean accuracy of 99.86% during K-Fold Cross-Validation.

DISCUSSION

The study shows that machine learning algorithms—K-Nearest Neighbours (KNN), Random Forest (RF), and Decision Tree (DT) in particular—are good in identifying phishing attempts. Because of its ability to handle complicated datasets with resilience, RF obtained 100% accuracy, whereas KNN achieved 99.87%. Adjusting hyperparameters, especially the number of estimators, improves the model's performance even more. Using evolutionary algorithms to improve depth, decision trees reach 100% accuracy; but, when used alone, overfitting is more prevalent. With an accuracy of 99.87%, the K-Nearest Neighbours (KNN) model—which is based on proximity-based categorization—performed somewhat worse than RF and DT. Its hyperparameter adjustment guaranteed a low classification error. KNN's computational efficiency and ease of use make it appropriate for smaller datasets or low-resource environments. While DT is equivalent, RF's ensemble learning approach reduces bias, making it the most accurate and dependable method for phishing detection. According to the study, phishing attack risks can be considerably decreased by implementing RF or DT in practical situations.

Table I: Comparison Between Random Forest, Decision Tree, and K-Nearest Neighbours

Aspect	Random Forest (RF)	Decision Tree (DT)	K-Nearest Neighbor (KNN)
Accuracy	100% for both	100% for both	99.87% during optimization 99.86% during K-Fold Cross Validation
Optimization	Genetic Algorithm (GA) optimized *n_estimators = 115*	GA optimized *max_depth = 13*	GA optimized *k=1*
K-Fold Cross Validation (K-Fold CV) Mean Accuracy	100% (standard deviation: 0.0000)	100% (standard deviation: 0.0000)	99.86% (standard deviation: 0.0001)
Strengths	Robustness through ensemble learning; minimizes overfitting	Captures complexity of data with tree structure	Simplicity; effective for proximity-based classification
Limitations	Computationally expensive for large datasets	Higher risk of overfitting without optimization	Sensitive to *k* value and high dimensionality
Performance Consistency	Extremely high (no variance across K-Fold CV folds)	Extremely high (no variance across K-Fold CV folds)	High but slightly less stable compared to RF and DT

The versatility of these models is enhanced by genetic algorithms for hyperparameter tuning. The extensive dataset might not, however, cover every potential phishing scenario. To improve phishing detection skills, future

studies should concentrate on a variety of datasets, ensemble approaches, and deep learning models such as Long Short-Term Memory networks.

CONCLUSION

In conclusion, this project is all about examination of the performance of machines algorithms used on phishing attack URL detection. At the end, Random Forest and Decision Tree models achieve 100% with the used of Genetic algorithm and K-Fold Cross-Validation methodology. While K-Nearest Neighbours achieved a lower performance with a maximum accuracy of 99.87%. In addition, with the utilization of Genetic Algorithm increase the successful of detecting the legitimate and phishing website. K-Fold Cross Validation remove every prediction is continuously using one single reference only.

REFERENCES

1. S. Hawa Apandi, J. Sallim, and R. Mohd Sidek, "Types of anti-phishing solutions for phishing attack," in IOP Conference Series: Materials Science and Engineering, Institute of Physics Publishing, Jun. 2020. doi: 10.1088/1757-899X/769/1/012072.
2. M. F. Alghenaim, N. A. A. Bakar, F. Abdul Rahim, V. Z. Vanduhe, and G. Alkawsi, "Phishing Attack Types and Mitigation: A Survey," in Lecture Notes on Data Engineering and Communications Technologies, vol. 165, Springer Science and Business Media Deutschland GmbH, 2023, pp. 131–153. doi: 10.1007/978-981-99-0741-0_10.
3. S. Albakry, K. Vaniea, and M. K. Wolters, "What is this URL's Destination? Empirical Evaluation of Users' URL Reading," in Conference on Human Factors in Computing Systems - Proceedings, Association for Computing Machinery, Apr. 2020. doi: 10.1145/3313831.3376168.
4. A. Awasthi and N. Goel, "Phishing website prediction using base and ensemble classifier techniques with cross-validation," Cybersecurity, vol. 5, no. 1, Dec. 2022, doi: 10.1186/s42400-022-00126-9.
5. S. Uddin, I. Haque, H. Lu, M. A. Moni, and E. Gide, "Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction," Sci Rep, vol. 12, no. 1, Dec. 2022, doi: 10.1038/s41598-022-10358-x.
6. T. Choudhary, S. Mhapankar, R. Bhddha, A. Kharuk, and R. Patil, "A Machine Learning Approach for Phishing Attack Detection," Journal of Artificial Intelligence and Technology, vol. 3, no. 3, pp. 108–113, Jul. 2023, doi: 10.37965/jait.2023.0197.
7. B. Charbuty and A. Abdulazeez, "Classification Based on Decision Tree Algorithm for Machine Learning," Journal of Applied Science and Technology Trends, vol. 2, no. 01, pp. 20–28, Mar. 2021, doi: 10.38094/jastt20165.
8. O. Kayode-Ajala, "International Journal of Information and Cybersecurity Applying Machine Learning Algorithms for Detecting Phishing Websites: Applications of SVM, KNN, Decision Trees, and Random Forests."
9. R. Yang, K. Zheng, B. Wu, C. Wu, and X. Wang, "Phishing website detection based on deep convolutional neural network and random forest ensemble learning," Sensors, vol. 21, no. 24, Dec. 2021, doi: 10.3390/s21248281.
10. E. Kocyigit, M. Korkmaz, O. K. Sahingoz, and B. Diri, "Enhanced Feature Selection Using Genetic Algorithm for Machine-Learning-Based Phishing URL Detection," Applied Sciences (Switzerland), vol. 14, no. 14, Jul. 2024, doi: 10.3390/app14146081.
11. S. A. Hicks et al., "On evaluation metrics for medical applications of artificial intelligence," Sci Rep, vol. 12, no. 1, Dec. 2022, doi: 10.1038/s41598-022-09954-8.
12. S. Hossain, D. Sarma, and R. J. Chakma, "Machine Learning-Based Phishing Attack Detection," 2020. [Online]. Available: www.ijacsa.thesai.org
13. V. Shahrivari, M. M. Darabi, and M. Izadi, "Phishing Detection Using Machine Learning Techniques," Sep. 2020, [Online]. Available: http://arxiv.org/abs/2009.11116
14. M. W. Shaukat, R. Amin, M. M. A. Muslam, A. H. Alshehri, and J. Xie, "A Hybrid Approach for Alluring Ads Phishing Attack Detection Using Machine Learning," Sensors, vol. 23, no. 19, Oct. 2023, doi: 10.3390/s23198070.
15. M. N. Alam, D. Sarma, F. F. Lima, I. Saha, R. E. Ulfath, and S. Hossain, "Phishing attacks detection

using machine learning approach," in Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020, Institute of Electrical and Electronics Engineers Inc., Aug. 2020, pp. 1173–1179. doi: 10.1109/ICSSIT48917.2020.9214225.

16. S. Aslam, H. Aslam, A. Manzoor, C. Hui, and A. Rasool, "AntiPhishStack: LSTM-based Stacked Generalization Model for Optimized Phishing URL Detection," vol. 10, 2022, doi: 10.3390/xxxxx.
17. D. Kaibassova, M. Nurtay, A. Tau, and M. Kissina, "SOLVING THE PROBLEM OF DETECTING PHISHING WEBSITES USING ENSEMBLE LEARNING MODELS," Scientific Journal of Astana IT University, pp. 55–64, Dec. 2022, doi: 10.37943/12oyrs4391.
18. T. A. Assegie*, "K-Nearest Neighbour Based URL Identification Model for Phishing Attack Detection," Indian Journal of Artificial Intelligence and Neural Networking, vol. 1, no. 2, pp. 18–21, Apr. 2021, doi: 10.35940/ijainn. b1019.041221.
19. N. F. Abedin, R. Bawm, T. Sarwar, M. Saifuddin, M. A. Rahman, and S. Hossain, "Phishing attack detection using machine learning classification techniques," in Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 1125–1130. doi: 10.1109/ICISS49785.2020.9315895.
20. A. O. Balogun et al., "Optimized Decision Forest for Website Phishing Detection," in Lecture Notes in Networks and Systems, Springer Science and Business Media Deutschland GmbH, 2021, pp. 568–582. doi: 10.1007/978-3-030-90321-3_47.
21. Arvind Prasad, "Phishing URL - Websites Dataset (PhiUSIIL)," Kaggle. Accessed: Dec. 30, 2024. [Online]. Available: https://www.kaggle.com/datasets/kaggleprollc/phishing-url-websites-dataset-phiusiil/data?select=PhiUSIIL_Phishing_URL_Dataset.csv