Düzce University
Journal of Science & Technology

Research Article

A Novel Model Based on Ensemble Learning for Phishing Attack

Aykut KARAKAYA [a,*], Ahmet ULU [b]

[a] *Department of Computer Technologies, Zonguldak Bulent Ecevit University, Zonguldak, TURKEY*
[b] *Department of Computer Engineering, Artvin Coruh University, Artvin, TURKEY*
* *Corresponding author's e-mail address: aykut.karakaya@bil.omu.edu.tr*
DOI: 10.29130/dubited.1426401

ABSTRACT

With the increase in the speed of the internet environment and the development of the infrastructures used, people have started to perform most of their work online. As much as this makes life easier, it also increases the possibility of being attacked by malicious people. Attackers can activate a phishing attack that aims to steal information from victims by creating copied, fake websites. While this attack is very old and somewhat simple, it can still be effective due to low IT literacy. People can enter their information on these fake websites out of spontaneity or ignorance or good intentions and be exposed to Phishing attacks. The compromise of a user's account information also puts at risk the security of the organization or institution to which it is connected. In this study, we propose a new machine learning-based ensemble model with feature selection methods to detect phishing attacks. Also, an ablation study is presented to measure the effect of different feature selection methods. The proposed model which we named as NaiveStackingSymmetric (NSS) is analyzed using the widely used accuracy (ACC), the area under curve (AUC), and F-score metrics as well as the polygon area metric (PAM), and it is shown that it outperforms other studies in the literature using the same dataset.

Keywords: *Phishing attack, ensemble learning, malicious URL, stacking, information security*

Kimlik Avı Saldırısı için Ensemble Öğrenmesine Dayalı Yeni Bir Model

ÖZ

İnternet ortamının hızının artması ve kullanılan altyapıların gelişmesiyle birlikte, insanlar çoğu işlerini çevrimiçi olarak gerçekleştirmeye başlamıştır. Bu durum hayatı kolaylaştırırken, kötü niyetli kişiler tarafından saldırıya maruz kalma olasılığını artırmaktadır. Bu saldırılardan biri de kimlik avıdır. Kimlik avı saldırısında saldırganlar, kopyalanmış, sahte web siteleri oluşturarak kullanıcılardan bilgi çalmayı amaçlamaktadır. Bu saldırı nispeten eski ve kolay olmasına rağmen, düşük bilgi teknolojileri okuryazarlığı nedeniyle hâlâ etkili olabilmektedir. Kullanıcılar, bu sahte web sitelerine anlık tepki, bilgisizlik veya iyi niyetle bilgilerini girebilmekte ve kimlik avı saldırılarına maruz kalabilmektedir. Bir kullanıcının hesap bilgilerinin tehlikeye girmesi, bağlı olduğu kuruluşun veya kurumun güvenliğini de riske atmaktadır. Bu çalışmada, kimlik avı saldırılarını tespit etmek için yeni bir makine öğrenimi tabanlı topluluk (ensemble) model öneriyoruz. Ayrıca, farklı özellik seçimi yöntemlerinin etkisini ölçmek için bir ablasyon çalışmaları sunuyoruz. NaiveStackingSymmetric (NSS) olarak adlandırdığımız model doğruluk (ACC), eğri altındaki alan (AUC) ve F-skor metrikleri ile çokgen alan metriği (PAM) kullanılarak analiz edilmekte ve aynı veri kümesini kullanan diğer çalışmalara göre daha iyi sonuçlara sahip olduğu gösterilmektedir.

Anahtar Kelimeler: *Kimlik avı saldırısı, ensemble öğrenme, kötücül URL, stacking, bilgi güvenliği*

Received: 27/01/2024, Revised: 05/02/2024, Accepted: 08/04/2024

I. INTRODUCTION

Developing technology allows people to meet many of their needs via the internet. Although it is valuable in terms of time and comfort, it also creates an environment for being exposed to more attacks. Attackers can create a large number of attacks using the internet and web environment, and their victims can trigger these attacks. One of the most encountered is Phishing attacks based on URLs with copy websites.

Although many protocols have been developed for cyber attacks, the need for systems and the number of threats that may occur are increasing at a similar level [1], [2]. As an example of an advanced social engineering attack, a phishing attack is one of the oldest types of attacks in internet history [3]. It is generally based on sending fake e-mails containing gifts, discount vouchers, and e-invoices to victims' e-mail boxes, causing the user to click on links in the e-mail or files containing malicious software. With the clicked link, the user is directed to a fake website created by the attacker, which is very similar to the legitimate website, and is asked to enter the account information. With this attack, the attacker aims to capture the victim's passwords, credentials, bank account information, or other sensitive information. In order to be protected from a phishing attack, precautions such as different passwords on each platform, not clicking on shortened URL links, not logging into the system without making sure that the website that seems to be legitimate is safe, and not responding to e-mails that ask for personal information should be taken. However, the scenarios in which these types of attacks are successful, which try to take advantage of people's momentary distraction or ignorance, are not to be underestimated.

A. RELATED WORK

The implementation of the phishing attack dates back to almost as old as the early times of the web service. Although the techniques are different today, their purposes are basically the same. In this section, the methods and results of current studies in the literature are given.

In [4], Almomani et al. have made a comparison with different machine-learning algorithms to detect phishing websites using semantic features. For this purpose, the 10 most effective semantic features have been tested with 16 machine learning methods and it has been stated that GradientBoostingClassifier and RandomForestClassifier methods give the best results with approximately %97. In [5], data preprocessing has been performed with adaptive synthetic sampling, and phishing attacks have been detected with a hybrid structure using S-shaped and V-shaped transfer functions. The k parameter of KNN (K-nearest-neighbours) is optimized. According to the polygon area metric [6], it is stated that a accuracy of 97.044 is achieved.

By detecting phishing in [7], machine learning performance results have been analyzed to help users identify fake websites. Accordingly, random forest and gradient boosting with XGBoost models have been stated to be the best model with %97.3. In [8], a dataset has been created by considering URL feature extraction, word analysis, and TinyURL approaches for phishing and tested with machine learning models. Accordingly, it has been emphasized that extra tree and deep neural network (DNN) gave the best results with %98. A phishing website detection model is proposed in [9], which is based on machine learning and takes into account the characteristics of the URL, the source code, and the threat intelligence of the websites. Accordingly, it is stated that Random Forest, Extra Tree, and Decision Tree models showed %97.56, %97.33, and %97.29 accuracies respectively.

A supervised learning approach that uses deep learning algorithms to detect phishing websites is proposed in [10]. It is stated that the standard neural network model achieves %94.8 accuracy and the CNN (Conv2D) model %93.6 accuracy. In [11], for malicious URL detection, after feature selection has been made on the dataset, LR (Linear Regression), SVM(Support Vector Machine), and KNN models have been tested. In the results, it is stated that LR achieved %92, KNN %93, and SVM %94 accuracy. In [12], mitigations against the most common web application attacks are set, and the web

administrator is provided with ways to detect phishing links which is a social engineering attack, the study also demonstrates the generation of web application logs that simplifies the process of analyzing the actions of abnormal users to show when behavior is out of bounds, out of scope, or against the rules. It is stated that Random forest, logistic regression, and SVM models have performed using the dataset in UCI, and the highest performance resulted as %94.13 by SVM. Then, with the data set they obtained from OpenPhish and Phishtank sites, it is stated that the highest performance was %98.86 by LSTM (Long Short-Term Memory).

In [13], two datasets with 30 and 48 features have been combined to identify 18 common features to detect phishing websites. Feature selection methods have been applied to reduce this to 13. When the random forest algorithm has been applied to these two datasets differently, it has been stated that the 48-attribute dataset has given better results than the 30-featured dataset with %93.7 accuracy. In [14], 5 machine learning-based experiments have been carried out for phishing website detection. It has been stated that the success rate of the approach that gave the best results from these experiments was %95.7. In order to detect phishing websites with common features in [15], data was obtained from Phishtank and compared to SVM, bayes, and neural network methods. It is stated that the neural network gives the best accuracy with an accuracy of %99.16.

In [16], machine learning-based models have been examined to detect phishing websites. F-score, ROC, and AUC parameters have been used as criteria. As a result, it has been stated that the SVM-supported Adaboost method has given the best result with %97.61. Using Random Forest in [17] is intended to detect whether a website is phishing or legitimate. It has been emphasized that the result obtained after the feature extraction techniques was %97.27 accuracy. In [18], phishing has been detected with the 5-layer PhiDMA (Phishing Detection using Multi-filter Approach) method. As a result of the experiments, it has been stated that %92.72 accuracy was achieved in detecting phishing sites.

The meta-algorithm plugin is proposed in [19] to support the improvement of classification performance for the development of various web phishing detection systems. It is stated that %97.5 accuracy was achieved by using the stacking process. In [20], different classification models were compared using different feature selection methods. It is seen that the performance of the dataset with the feature selection methods applied has decreased compared to the original dataset. As a result of the comparison, it seems that the ID3 (Iterative Dichotomiser 3) method, which is the decision tree without feature selection, has the best accuracy with %96.73.

In [21], various machine learning algorithms is aimed at predicting whether a website is phishing or legitimate are examined. It was stated that the Random Forest method with PCA (Principal Component Analysis) applied has given the best accuracy with %98.4. An intelligent system that uses data mining to detect phishing attacks is proposed in [22]. As criteria, accuracy, AUC, and F-score are used. In the experimental results, it is stated that the method with the highest accuracy was Random Forest with %97.36. Machine learning models were compared to detect a phishing attack in [23]. As a result of this comparison, it is emphasized that the Random Forest method, which applied PSO (Particle swarm optimization) feature selection, gave the best accuracy with %95.2.

In this study, a new stacking-based machine-learning model that is one of the ensemble learning types for phishing attacks is proposed. To accelerate the performance of the proposed model whose name is NSS (NaiveStackingSymmetric) a feature selection method is applied to the dataset. Besides, examining the effect of feature selection on classification results, we have conducted an ablation study. To this purpose, two filter approaches and two wrapper approaches which are based on feature selection algorithms are chosen. The proposed new model presented outperforming results compared with state-of-art methods under different metrics.

B. MOTIVATION AND CONTRIBUTIONS

Cyber threats and attacks are among the most important problems of today's world. The phishing attack is one of the most common of these threats because it does not require high technical knowledge to carry out. Although it is thought to be easy to protect against these attacks, it can lead to bad consequences if people are exposed to these attacks as a result of possible carelessness. In this study, we propose a machine learning-based model to detect whether a website contains a phishing attack. The main motivation of this study is the topicality of the attack type, the widespread use of the attack, the high probability of exposure, and the scarcity of machine learning-based systems with high performance.

The contributions of this study can be summarized as follows:

- A detailed literature review on the subject is conducted and evaluated together with the accuracy rates and discussed the methods that are mostly using the same dataset.
- Using the stacking method, one of the ensemble methods, a new machine learning-based model is proposed for phishing detection.
- In order to improve the performance of the NSS, the feature selection method is used. An ablation study is also presented to evaluate the effect of feature selection methods on the NSS. In this ablation study, a comprehensive analysis is performed for 4 different selection methods, two of which are filter approaches and two are wrapper approaches.
- The NSS is evaluated under the ACC, AUC, F-score, and polygon area metric(PAM). It can be seen that the proposed stacking ensemble model with feature selection outperforms compared with state-of-the-art denoising methods.

C. ORGANIZATION

The literature review and contributions of the paper are presented in the previous sections. The following sections are organized as follows. Section 2 contains preliminaries describing the methods used in the paper, Section 3 details the dataset used and the proposed methodology, Section 4 presents the results obtained and a discussion for the analysis of these results, section 5 contains directions for future work, and the last section concludes the paper.

II. PRELIMINARIES

This section provides a detailed preliminary overview of the methods used in the proposed machine learning model. A detailed explanation of k-means, random forest, modlem, and naive bayes methods used in building the stacking-based machine learning model is given. In addition, the details of the feature selection methods which are genetic search, particle swarm optimization, significance attribute evaluation, and symmetrical uncertainty attribute evaluation in the preprocessing section are also explained in this section.

A. CLASSIFICATION ALGORITHMS

A. 1. k-Nearest-Neighbours (kNN)

In classification using k-nn, the distance of each data in the dataset is calculated. However, for a given data, only k points of the other data are taken into account. These k points are the points that are closest to the point whose distance is calculated compared to the other data. The k value is chosen in advance. Too high a value causes dissimilar data to be assigned to the same class, too small a value causes data that should be in the same class to be assigned to different classes.

Algorithm 1. *KNN algorithm* [24]

Initialization. Training data (X); class labels (Y); number of nearest neighbors (K)

Foreach sample X in the test data **do**

Calculate the distance: $d(x, X) = \sqrt{\sum_{i=1}^{n}(x_i - X_i)^2}$

Classify x in the majority class: $C(x_i) = argmax_k \sum_{X_i \in KNN} C(X_j, Y_K)$

Output. Class of a test sample x

As a working principle, a distance measurement method is first determined. The most commonly used one is the Euclidean distance. The k points closest to each other are identified. The class closest to the group is determined and the group is labeled with that class. The general structure is given in Algorithm 1. The performance of the KNN classifier algorithm also depends on the value of K [25]. Usually, the optimal value of k is determined empirically.

A. 2. Random Forest

Breiman first introduced the random forest (RF) algorithm, which has since become a widely used nonparametric classification and regression tool for developing prediction rules based on various types of predictor variables without making any assumptions about how they will be associated with the response variable [26]. For classification and regression problems, RF can be used; RF combines the output of various decision trees (DT) to produce a singular outcome. That is why, it is referred to as an "ensemble learning" approach to reduce the overfitting of DT.

Tree-based models iteratively split the dataset into two groups until a certain predefined stopping criterion is met. Depending on how the splitting and stopping criteria are set, decision trees can be designed for both classification and regression tasks. In both cases, the subset of variables chosen to split the node is generated according to a predetermined splitting criterion formulated as an optimization problem [27]. Entropy, a practical application of Shannon's source coding theorem, is widely used as a splitting criterion in classification. The entropy formula is given in Equation 1.

$$E = -\sum_{i=1}^{c} p_i \times log(p_i) \quad (1)$$

Here c represents the number of unique classes, and p_i represents the prior probability of each class. The value of E is maximized to get the most information in each part of the decision tree. The disadvantage of decision trees is that they cause too much overfitting. This leads to a low accuracy of the overall estimation. Building numerous separate trees while just taking into account a portion of the observations can improve generalization accuracy. The random-subspace method was first proposed by Ho, and then expanded and formally published as the random forest by Breiman [27]. The random forest model is a community-based learning algorithm. Estimates are averaged over many individual trees. Trees are built on bootstrap instances rather than the original instance and This reduces the overfitting. The random forest method is illustrated in Algorithm 2.

Algorithm 2. *Random forest algorithm*

Initialization. Training data (D), subtrees (B)
For $i \leftarrow 1$ to B **do**
  Draw a bootstrap sample of size N from D
  While node size ≠ minimum node size **do**
    Randomly select a subset of m predictor variables from total p
    For $j \leftarrow 1$ to m **do**
      If jth predictor optimizes splitting criterion **then**
        Split internal node into two child nodes

break

Output. The ensemble tree of all B subtrees is created.

Random forest structures, which are a collection of decision trees, perform better than individual decision trees. Compared to decision trees, the random forest algorithm more precisely predicts the mistake rate. According to mathematical proof, the error rate always decreases as the number of trees rises [26]. The size of the subset of predictor variables, m, in the random forest algorithm, is essential for regulating the final depth of the trees. Therefore, it is a parameter that should be adjusted during model selection.

A. 3. Modlem

One of the key objectives in machine learning, data mining, and rough set theory is the discovery of rules from examples. As one of them, the Modlem algorithm develops rules using rough set theory and it is suited to deal with numerical and imperfect data [28]. It is a sequential covering algorithm that generates the smallest possible collection of unordered rules. It repeatedly looks for the best rule for a given class, deletes any positive instances from the learning set that have been covered by that rule, and repeats the process until all examples from that class have been covered. For every single class, the procedure is repeated. Finding the best condition is the first step in building a single rule, and adding further conditions is done so until a stopping requirement is satisfied. The direct processing of numerical attribute values (without pre-discretization) and missing values makes up Modlem's unique feature. Additionally, it can be used to handle inconsistent or noisy instances using rule pruning or rough estimates. The Modlem method is shown in Algorithm 3.

Algorithm 3. *Modlem algorithm*

Initialization. A set of positive examples from a given decision concept (B), an evaluation measure ($criterion$)
$G := B$; a temporary set of rules covered by generated rules
$R := \emptyset$
While $G \neq \emptyset$ **do**
  $T := \emptyset$; a candidate for a rule condition part
  $S := U$; a set of objects currently covered by T
  While $T = \emptyset$ or not ($[T] \subseteq B$) **do**
    $t := \emptyset$; a candidate for an elementary condition
    Foreach attribute $q \in C$ **do**
      Find best conditions with q and S, assign to new_t
      If Better($new_t, t, criterion$) **then**
        $t := new_t$; evaluate if a new condition is better than previous one according to the chosen evaluation measure
    $T := T \cup \{t\}$; add the best condition to the candidate rule
    $S := S \cap [t]$; focus on examples covered by the candidate
  Foreach elementary condition $t \in T$ **do**
    If $[T - t] \subseteq B$ **then**
      $T := T - \{t\}$; test a rule minimality
    $R := R \cup \{T\}$; store a rule
    $G := B - \cup_{T \in R}[T]$; remove already covered examples
Foreach $T \in R$ **do**
  If $\cup_{T' \in R-T}[T'] = B$ **then**
    $R := R - T$; test minimality of the rule set
Output. R single local covering of B, treated here as rule condition parts

A. 4. Naive Bayes

Naive Bayes (NB) is a straightforward learning algorithm that makes use of the Bayes rule and the fundamental presumption that, given the class, the attributes are conditionally independent [29]. Despite the fact that in practice this independence assumption is frequently broken, naive Bayes frequently produces competitive classification accuracy. This, together with its computational effectiveness and numerous other appealing characteristics, contributes to Naive Bayes' widespread use in practice.

Given a training dataset D_{train} of t classified objects, Naive Bayes estimates the probability $P(y|x)$ that a new instance $x = \{x_1, x_2, \ldots, x_a\}$ belongs to a class y. Where x_i represents the value of attribute X_i, $y \in \{1, \ldots, c\}$ represents the value of class variable Y [30]. D_{test} is the test dataset, c is the number of classes, a is the number of attributes.

The definition of conditional probability is $P(y|x) = P(y, x)/P(x)$. Taking $P(x)$ as the normalization constant, it makes sense to estimate the joint probability $P(y, x)$. If there are not enough x samples in the training data, an accurate estimate of $P(y, x)$ cannot be obtained directly. It is necessary to infer these estimates from observations of lower-dimensional probabilities in the data [30]. Accordingly, redefining conditional probabilities yields Equation 2.

$$P(y, x) = P(y)P(x|y) \tag{2}$$

If the number of classes k is not too large, $P(y)$ in Equation 2 can be accurately estimated from the sample frequencies. To compute $P(x|y)$ based on low-dimensional probabilities, it is factorized by the chain rule in Equation 3.

$$P(x|y) = \prod_{i=1}^{a} P(x_i|x_1, x_2, \ldots, x_{i-1}, y) \tag{3}$$

Equation 3 is optimal in theory. However, for datasets with a large number of features, the conditional probability $P(x_i|x_1, x_2, \ldots, x_{i-1}, y)$ cannot be estimated accurately enough because the feature dependency arcs are too large, leading to high complexity. Consequently, Naive Bayes assumes that the attributes of a given class are independent of each other. Thus Equation 4 It simplifies the calculation of $P(x|y)$.

$$P(x|y) = \prod_{i=1}^{a} P(x_i|y) \tag{4}$$

As a result, Naive Bayes calculates the joint probability $P(y, x)$ according to Equation 5.

$$P_{NB}(y, x) = P(y) \prod_{i=1}^{a} P(x_i|y) \tag{5}$$

Thus, Naive Bayes classifies a new instance of x by choosing it as in Equation 6.

$$argmax_y(P'(y) \prod_{i=1}^{a} P'(x_i|y)) \tag{6}$$

Here $P'(y)$ and $P'(x_i|y)$ are estimates of the probabilities derived from the frequencies of their respective arguments in the training sample with possible corrections. The training process of NB is given in Algorithm 4.

Algorithm 4. *Naive Bayes training process* [30]

Initialization. $Count$: Table of observed counts of combination of 1 attribute value and the class label
For $instance \in D_{train}$ **do**
  Get the value of class variable in $instance$, suppose it is the y^{th} value
  For $X_i, i \in \{1,2,3,\ldots,a\}$ **do**
    Get the value of attribute X_i in $instance$, suppose it is the j^{th} value
    Increase the element in $Count$ with index (i, j, y) by 1

B. FEATURE SELECTION METHODS

B. 1. Genetic Search

The Genetic Algorithm (GA) is an evolutionary algorithm (EA) that promotes the survival of the fittest and was influenced by Charles Darwin's idea of natural selection [31]. According to the principle of natural selection, only the fittest individuals are chosen to have children. To increase the likelihood of survival, the traits of the fittest parents are subsequently transferred to their kids through cross-over and mutation. The natural selection process, such as selection, cross-over, and mutation, is biologically inspired, and genetic algorithms mimic this process to produce high-quality optimization solutions. There are five phases in a genetic algorithm:

- Initial population: Given that each individual is represented binary, the population is a binary matrix where the rows represent the randomly chosen individuals and the columns represent the potential predictors. With a random selection of 0 and 1, for each entry, an initial population with a predetermined number of people is formed.
- Fitness function: Each member of the population has their fitness value determined using a predetermined fitness function. For the following generation, the person with the lowest prediction error and the fewest predictors has been chosen [32].
- Selection: Through crossover and mutation processes, the elite individuals who have been chosen based on their fitness value are chosen as parents to create offspring.
- Crossover: By transferring entries between two chosen parents from the previous stage, a new generation is created using this process.
- Mutation: This procedure, which is used after crossover, assesses whether a person should be modified in the following generation and ensures that no predictors have been permanently eliminated from the GA population.

The flow chart of the genetic search algorithm is shown in Figure 1.

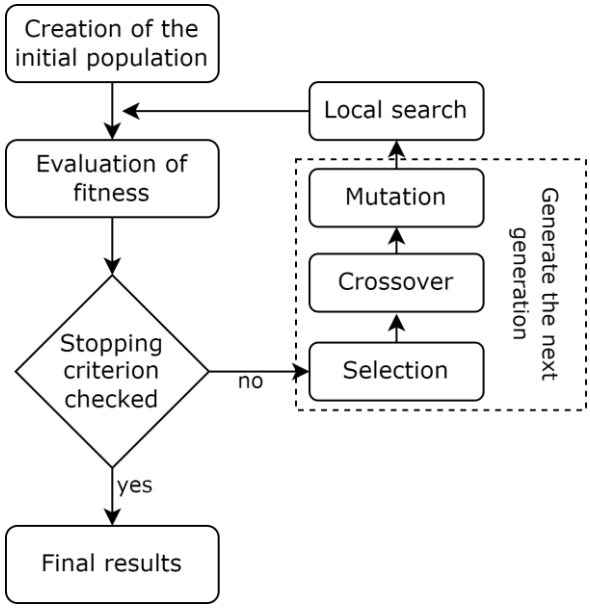


Figure 1. *Genetic search algorithm*

B. 2. Particle Swarm Optimization

PSO is an intelligent evolutionary computer system that is influenced by the social behavior of creatures like a flock of birds finding food sources. Kennedy and Eberhart introduced it in 1995 [33]. It is a computer strategy that resolves a problem by repeatedly attempting to enhance a candidate arrangement with regard to a certain percentage of value [34]. PSO's goal is to collaborate and share data among the particles or molecules in a group that may be thought of as a population in order to identify the best solution. A particle is a constituent or element of the swarm's population. In order to locate promising scene territories, the swarm must fly over the hunting area. Every particle is randomly initialized, has a searching space across which it searches for food, and carries both velocity and position information. Each particle is aware of both its own optimal position P_{best} and the optimal position within the group of particles G_{best}. Using the following Eq. 7 and 8, the velocity and position of each particle are updated after each iteration.

$$V_i^{t+1} = W * V_i^t + c_1 * r_1(P_{best}^t - X_i) + c_2 * r_2(G_{best}^t - X_i) \quad (7)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (8)$$

where V_i is velocity, X is position, t is iteration, W is inertia weight, c is cognitive constant, r is random number. The steps of the PSO algorithm can be summarized as follows:

- (1) Generate the initial position randomly
- (2) Calculate the parameters of each particle
- (3) Evaluate each particle via fitness function(objective function)
- (4) Calculate global ve particle best values
- (5) Update the velocity and position of each particle
- (6) Go step 2 until the stopping criteria is satisfied

B. 3. Significance Attribute Evaluation (SAE)

A feature ranking technique called significance attribute evaluation determines an attribute's effect by computing its conditional probability-based significance as a two-way function (feature-classes and

classes-feature association) [35]. Feature-classes(FC) and classes-feature(CF) association can be defined as follows:

$$FC = \left(\frac{1}{m}\sum_{i=1}^{m}\gamma^{i}\right) \tag{9}$$

$$CF = \left(\frac{1}{k}\right) \times \left(\sum_{j=1}^{n}\delta^{i}\right) - 1.0 \tag{10}$$

where m is the number of unique features, γ is the discriminating power, δ is the separability of a single feature with regard to class j and n is the total class number. SAE is calculated as an average of FC and CF as follows:

$$SAE = \frac{FC + CF}{2} \tag{11}$$

B. 4. Symmetrical Uncertainty Attribute Evaluation (SUAE)

Mutual information is a fundamental method for calculating the degree of correlation between two features. It is described as the difference between the joint entropy and the sum of the marginal entropies. The mutual information for two completely independent items is always 0. Most feature selection systems based on mutual information use symmetric uncertainty (SU), one of the best feature selection approaches [36]. By calculating the relationship between the feature and the target class, symmetric uncertainty can be utilized to determine the fitness of features for feature selection. A feature that has a high SU value is given a lot of importance. The definition of symmetric uncertainty can be done as follows:

$$IG(A|B) = E(A) - E(A|B) \tag{12}$$

$$SU(A,B) = 2 \times \frac{IG(A,B)}{E(A) + E(B)} \tag{13}$$

where $E(A)$ and $E(B)$ are the entropy of features A and B, $E(A|B)$ is the joint probability and $IG(A|B)$ is the information gain of A under B.

III. PROPOSED MODEL

This section contains the details and analysis of the proposed model. The article proposes a novel ensemble learning-based model for detecting malicious URL and phishing websites. For this purpose, the determination of the dataset, the feature selection methods, the establishment of the classification model and the details of the algorithms used are explained.

A. DATASET DESCRIPTION

In order to train and test the proposed model that named as NSS, first of all, accurate and reliable datasets are needed. In this paper, "Phising Website Features" [37] dataset from the UCI dataset pool was used in order to be reliable and comparable. The dataset has 30 input attributes, 1 output attribute, and 11055 record data. After detailed analysis of the dataset, Figure 2 is created, which includes the

value ranges determined for each attribute. The dataset is located in the data store with normalization applied. In this study, the dataset is parsed for %70 training and %30 testing.

No	Feature	Value	Description
1	having_IP_Address	{-1,1}	Having an IP address in the URL *(yes, P | no, L)*
2	URL_Length	{1,0,-1}	URL length *(<54, L | >=54 and <=75, S | otherwise, P)*
3	Shortining_Service	{1,-1}	Using URL shortening services "TinyURL" *(yes, P | otherwise, L)*
4	having_At_Symbol	{1,-1}	URL's having "@" symbol *(yes, P | otherwise, L)*
5	double_slash_redirecting	{-1,1}	Redirecting using "//", except the first "https://" *(URL index>7, P | otherwise, L)*
6	Prefix_Suffix	{-1,1}	Adding prefix or suffix separated by (-) to the domain *(yes, P | otherwise, L)*
7	having_Sub_Domain	{-1,0,1}	Sub/Multisub domain, except for extensions *(dots in domain=1, L | 2, S | otherwise, P)*
8	SSLfinal_State	{-1,1,0}	Use HTTPS - issuer is trusted - age of certificate *(yes-yes and >=1year, L | yes-no, S | otherwise, P)*
9	Domain_registeration_length	{-1,1}	Domain registration length *(domain expires on<=1year, P | otherwise, L)*
10	Favicon	{1,-1}	Favicon loaded from external domain *(yes, P | otherwise, L)*
11	port	{1,-1}	Only needed ports should be open *(Only required ports are open, L | otherwise, P)*
12	HTTPS_token	{-1,1}	Using HTTP token in domain part of the URL *(yes, P | otherwise, L)*
13	Request_URL	{1,-1}	Most of objects in a webpage are the same domain *(Request URL <22%, L | >=22% and <=61%, S | otherwise, P)*
14	URL_of_Anchor	{-1,0,1}	Using the <a> tag, similar to URL request *(URL of Anchor <31%, L | >=31% and <=67%, S | otherwise, P)*
15	Links_in_tags	{1,-1,0}	Use of links in <Meta>, <Script>, and <Link> tags *(<17%, L | >=17% and <=81%, S | otherwise, P)*
16	SFH	{-1,1,0}	Server Form Handler *("about:blank" or is empty, P | refers to a different domain, S | otherwise, L)*
17	Submitting_to_email	{-1,1}	Using "mail()" or "mailto:" function to submit user information *(yes, P | otherwise, L)*
18	Abnormal_URL	{-1,1}	The host name is not included in URL *(yes, P | otherwise, L)*
19	Redirect	{0,1}	Number of redirect page *(<=1, L | >=2 and <4, S | otherwise, P)*
20	on_mouseover	{1,-1}	Changes status bar *(onMouseOver, P | it does't change, L)*
21	RightClick	{1,-1}	Blocking access to the source code of the web page, "event.button==2" *(right click disabled, P | otherwise, L)*
22	popUpWindow	{1,-1}	Pop up window contains text fields *(yes, P | otherwise, L)*
23	Iframe	{1,-1}	Using iframe redirection without frame borders *(yes, P | otherwise, L)*
24	age_of_domain	{-1,1}	Most phishing websites live for a short period of time *(age of domain >=6 months, L | otherwise, P)*
25	DNSRecord	{-1,1}	No DNS record for the domain *(yes, P | otherwise, L)*
26	web_traffic	{-1,0,1}	The popularity of the website - website rank *(<100k, L | >100k, S | otherwise, P)*
27	Page_Rank	{-1,1}	How important a web page is on the Internet - interval 0 and 1 *(<0.2, P | otherwise, L)*
28	Google_Index	{1,-1}	Webpage indexed by Google *(yes, L | otherwise, P)*
29	Links_pointing_to_page	{1,0,-1}	Legitimacy level - the number of links pointing to the web page *(=0, P | >0 and <=2, S | otherwise, L)*
30	Statistical_report	{-1,1}	Host belongs to top phishing IPs or top phishing domains *(yes, P | otherwise, L)*
Result		**{-1,1}**	**Phishing or legitimate decision of the website *(output)***
Abbreviations - P: Phishing, S: Suspicious, L: Legitimate			

Figure 2. *Dataset features and descriptions*

B. FEATURE SELECTION

Before the classification of the dataset, a feature selection is used to find the best relative feature and eliminate the redundant ones. There could be some redundant or useless attributes in a dataset containing features. A feature selection algorithm eliminates redundant and unnecessary features to choose the best set possible. The two major categories of feature selection approaches are the filter approach and the wrapper approach. The filter approach is a feature ranking technique that assesses relevant and nonredundant features in accordance with the inherent characteristics of the data without reference to the classification methods. Filter techniques have the benefits of being quick, scalable, and independent of a learning algorithm. The filter approach's drawbacks include neglecting the classifier's interaction and the prediction of feature dependencies. Another feature ranking technique is the wrapper strategy, which rates nonredundant and pertinent features in accordance with the classifier. The wrapper method's shortcomings include overfitting and time-consuming computing. The connection between feature subset search and classification algorithm is one benefit of wrapper techniques.

In this paper, instead of using a single feature selection algorithm, we have used four different methods which are two of their filter and the other two wrapper approach. Besides, we have analyzed the results as an ablation study. While as filter approaches, Significance Attribute Evaluation(SAE) and Symmetrical Uncertainty Attribute Evaluation (SUAE) have been chosen, as wrapper approaches Particle Swarm Optimization(PSO) and Genetic Search(GS) have been chosen.

C. CLASSIFICATION

In this section, details of the proposed stacking model are explained. For this reason, different classification algorithms are combined via another classification algorithm which is also known as a meta-learner in stacking methods. In addition to the stacking method, we also combine these classification algorithms with a voting approach which one of the ensemble learning methods for better analysis.

C. 1. Ensemble Learning

Bagging, stacking, and boosting are the three structures that make up ensemble learning. First, the data set for bagging is split into test and train groups (often with a ratio of 70/30). A certain number of bags are filled with random and repeated samples taken from the train data. Every sample bag receives training using recognized models. The outputs are averaged or voted on to make decisions. Similar to bagging, data is separated and randomly sampled in the boosting process. Each sample is trained independently and generates output in the bagging method, giving each model an equal opportunity to succeed. In contrast, in the boosting method, data that was incorrectly identified by one model is given priority [38].

Three sets of classifiers are created at once during the boosting process. Similar to bagging, the first and second classifiers are trained using various randomly selected portions of the data set. On data on which the first and second classifiers failed, the third classifier was trained. The majority vote technique is then paired with these three classifiers. On the other hand, the stacking decides based on the percentage of the feature space where the classifiers are successful. The outputs of the classifiers are combined with another classifier and the decision is made [39].

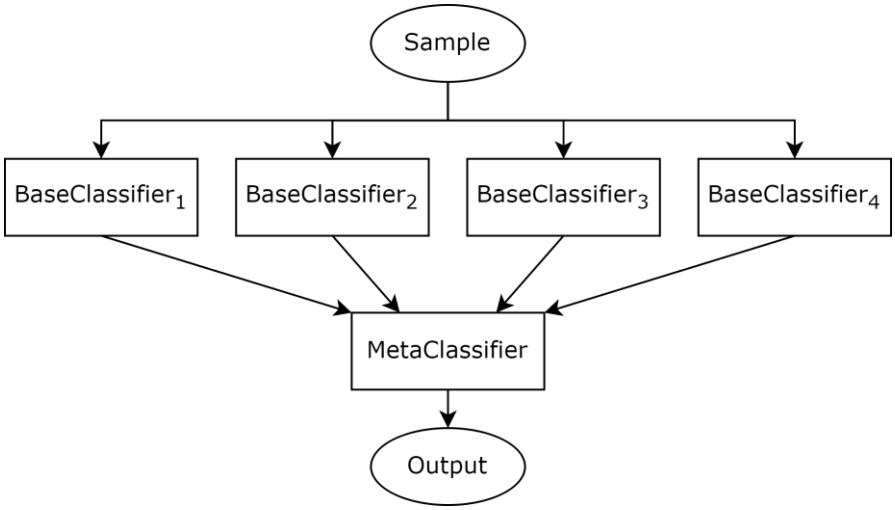


Figure 3. *Stacking process in the ensemble learning*

Stacking in ensemble learning is shown in Figure 3. Here there are four different basic classifier examples. Fewer or more classifiers can be used depending on the model design. The new incoming sample is evaluated in each classifier to be classified. The results from each are evaluated in a new metaclassifier. According to the result of the meta classifier, the sample data is marked with a class label [40].

One of the ensemble learning is also a voting classifier. A voting classifier's architecture is made up of n machine learning models, whose predictions are valued in both hard and soft ways. In a hard vote, the prediction that receives the most votes wins. The winning class will be the one with the highest weighted and averaged probability, on the other hand, because the Voting Classifier in soft mode takes into account the probabilities generated by each machine learning model.

C. 2. Details of Proposed Ensemble Model

We have described the details of the proposed ensemble model in this section. As a classification method, we have chosen a stacking ensemble learning algorithm. In this stacking method as a meta-learner, we prefer the Naive-Bayes and as heterogeneous weak learners, we prefer the K-NN, the Modlem, and the Random forest. So, the design stacking method consists of 4 different classification algorithms. The stacking process of the NSSis shown in Figure 4.

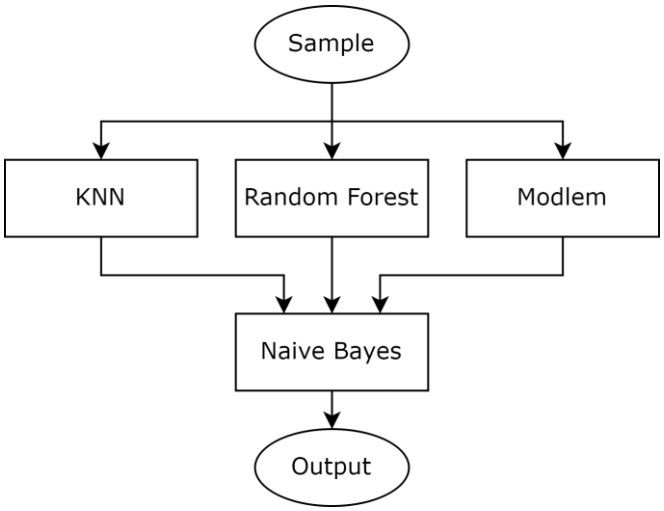


Figure 4. *Stacking process of the NSS*

The proposed stacking model is evaluated under different metrics. For this purpose, we have chosen PAM (the polygon area metric) in addition to well-known metrics like accuracy(ACC), the area under curve(AUC), and F-score. We also provided a confusion matrix to better analyze the classification results.

Firstly, to choose the best feature selection method, we conducted an ablation study. For this purpose, four different feature selection methods are chosen. As wrapper approaches we ran PSO and Genetic Search (GS) methods. As a classification algorithm to carry out PSO and GS, the multi-layer perceptron is preferred. While under PSO feature selection, 22 features are selected under 30 total features, under GS, 26 features are selected. For SUAE and SAE feature selection methods, 30 features are ranked and %10 pruning is done to prefer the most relevant features. When the features selected by both SUAE and SAE are analyzed, it is seen that two features which are "popUpWindow" and "Favicon" are selected as mostly irrelevant while the most relevant features selected by filter approaches are "SSL final State", "URL of Anchor" and "Prefix Suffix".

The only ensemble method is not stacking, there are also other methodologies like voting. Therefore for analysis, we also carried out another one of the ensemble models which is voting under the same feature selection algorithm(SUAE) and classification algorithms (KNN, Random Forest, Modlem).

IV. RESULTS AND DISCUSSION

This section contains detailed analysis, evaluation, and comparison of the results obtained. The NSS is evaluated according to different metrics. These metrics are produced according to the complexity matrix of the result obtained from the model. In complexity matrices, there are four states: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The first of the metrics evaluated according to these situations is Accuracy. Accuracy gives the ratio of correct predictions to total predictions. The mathematical formula of accuracy is given in Equation 14.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

Precision is the ratio of correctly predicted positive results to total positive results. The mathematical formula of precision is given in Equation 15.

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

Specificity is the percentage of correctly classified belonging to negative samples and its formula can be given as in Equation 16.

$$Specificity = \frac{TN}{TN + FP} \tag{16}$$

Sensitivity or in other words recall is the ratio of correctly predicted positive results to all results in the true class. The mathematical formula of recall is given in Equation 17.

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

The Jaccard similarity index (JI), also known as the Jaccard similarity coefficient, compares the sample in two sets to determine which samples are similar and which samples are different. JI can be calculated as in Equation 18.

$$JI = \frac{TP}{TP + FP + FN} \tag{18}$$

It gives the weighted harmonic average of F-score, precision and recall values. The F-score formula is given in Equation 19.

$$Fscore = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{19}$$

The area under the curve (AUC) is a graphical area metric that is calculated according to the ROC curve. The performance of a classification model at each classification threshold is represented graphically by the ROC curve. True Positive and False Positive rates at different thresholds are shown in this graph. On the other hand, The AUC is a metric obtained by measuring the entire two-dimensional area under the whole ROC curve. AUC can be calculated as in Equation 20.

$$AUC = \int_0^1 g(x)dx \tag{20}$$

where $g(x)$ is a ROC curve that is drawn with the true-positive rate and the false-positive rate for different cut-off points.

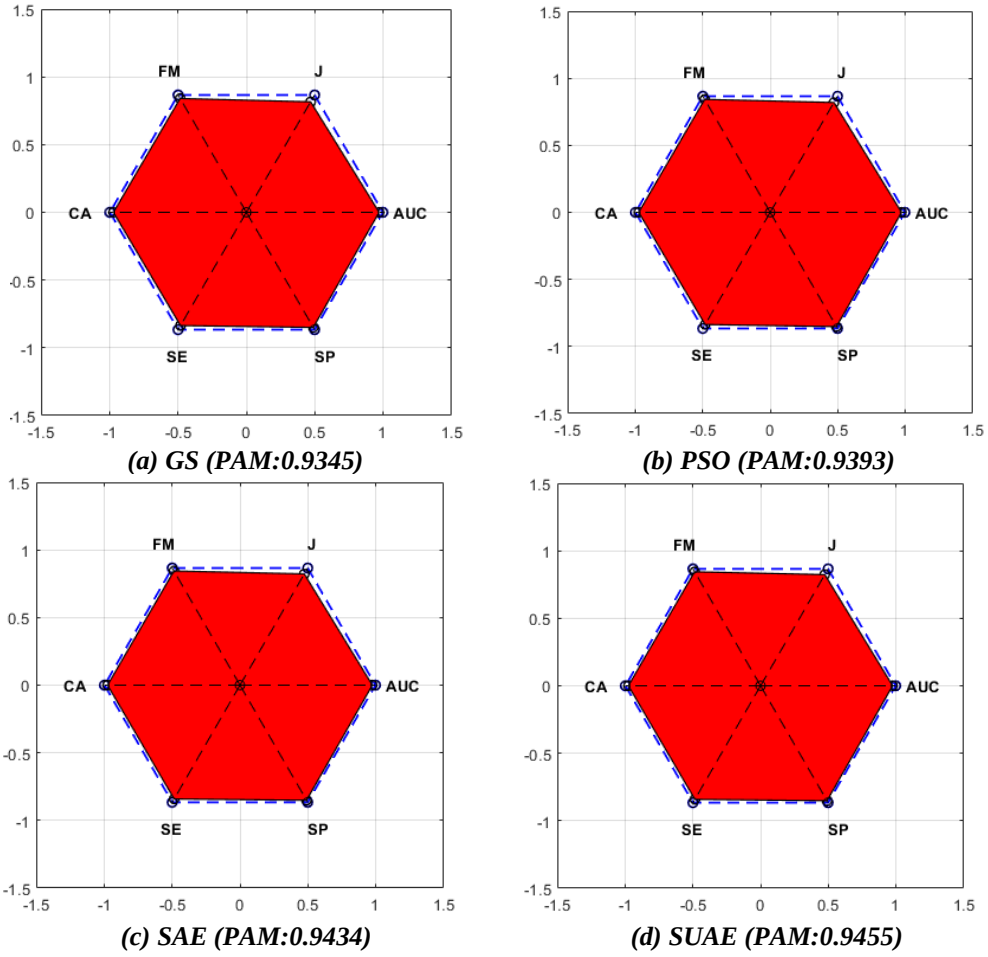
The Polygon Area Metric (PAM) is calculated using the regular hexagon area created by using six different metrics [6]. These metrics are accuracy, sensitivity, specificity, AUC, JI, and F-score. Basically, a regular hexagon is divided into 6 areas(triangle) and each of them fills these 6 metrics. Then, the percentage of filled area is calculated according to the Equation 21.

$$PAM = \frac{PA}{2.59807} \tag{21}$$

where PA is the filled area, and the number of 2.59807 is the area of the regular hexagon. As can be seen, the calculated PAM is ranging between $[0,1]$.

A. ABLATION STUDY

The proposed stacking model is trained with four different feature selection algorithms. The obtained accuracy values are %97.5271 and %97.3160 PSO and GS, respectively. While SAE gives %97.6779 accuracy, SUAE gives %97.7382 accuracy. For the NSS, under evaluation of F-score, with SUAE and SAE feature selection it gives 0.9744 and 0.9737 respectively, while it gives 0.9719 and 0.9695 respectively for PSO and GS. Besides, Under evaluation of AUC, while with SUAE and SAE feature selection it gives 0.9767 and 0.9762 respectively, with PSO and GS it gives 0.9742 and 0.9723 respectively. As for the PAM metric, for SUAE and SAE, it gives 0.9447 and 0.9434, while it gives 0.9393 and 0.9345 for PSO and GS. PAM results are also presented in Figure 5.



(a) GS (PAM:0.9345) *(b) PSO (PAM:0.9393)*

(c) SAE (PAM:0.9434) *(d) SUAE (PAM:0.9455)*

Figure 5. *PAM results for the NSS with four different feature selections*

For the proposed stacking model, all results are summarized in Table 1 for four different feature selection methods under the 4 different metrics which are accuracy, AUC, F-score, and PAM. Besides, confusion matrixes are provided in Figure 6. In this figure, the "A" presents a phishing class, and the "B" presents a legitimate class. It can be seen that filter approaches gave better results for the proposed stacking model. It is most likely that, wrapper methods can result in over-fitting results.

Table 1. *Results for the proposed method under four different feature selections*

Feature Selection Method	Accuracy (%)	AUC	F-Score	PAM
Genetic Search	97.3160	0.9723	0.9695	0.9345
PSO Search	97.5271	0.9742	0.9719	0.9393
SAE	97.6779	0.9762	0.9737	0.9434

SUAE	97.7684	0.9770	0.9747	0.9455

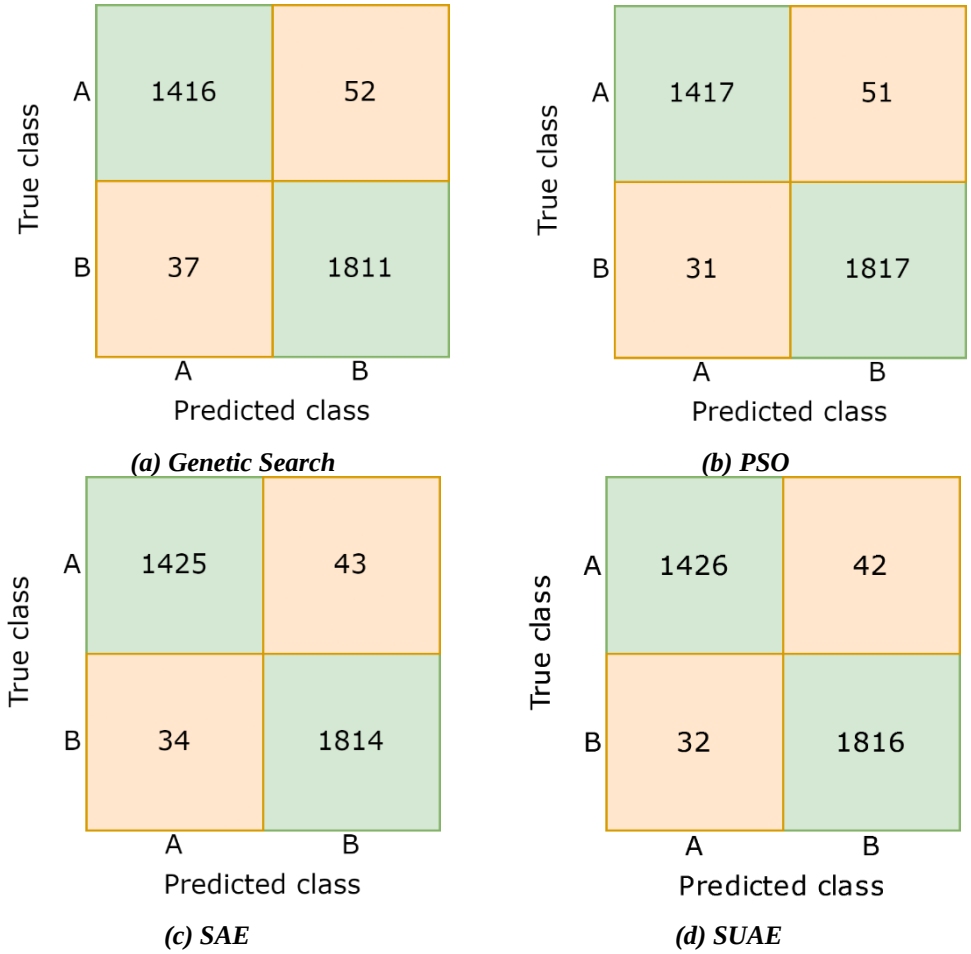


Figure 6. *Confusion matrices for proposed ensemble learning model under four feature selection algorithms*

The NSS with the SUAE feature selection method is chosen as the final proposed algorithm and named as NaiveStackingSymmetric model because the NaiveStackingSymmetric model gives %97.7382 accuracy, it also gives 0.9767, 0.9744, and 0.9447 under AUC, F-score, and PAM metrics respectively which are also better than other feature selection algorithms.

B. PROPOSED MODEL ANALYSIS

In the model where the best results are obtained, that is, the SUAE feature selection method is used, %10 pruning is applied for the best relevant features selection. Then, an ensemble learning model consisting of K-NN, Modlem, and random forest methods is applied and the results are stacked with Naive Bayes.

In order to provide an analysis of the classification methods used, it is presented as an ablation study by using hard voting and soft voting methods from ensemble methods as well as the stacking method. Accordingly, the results obtained using the accuracy, AUC, F-score and PAM metrics are shown in Table 2. Hard voting performs a little better than soft voting under all metrics. However, the stacking method using Naive Bayes as the meta classifier has still higher performance than both voting methods.

Table 2. *Results for the proposed method under three different meta classifiers*

Ensemble Methods	**Accuracy (%)**	**AUC**	**F-Score**	**PAM**

Hard Voting	97.0748	0.9689	0.9665	0.9277
Soft Voting	97.0446	0.9687	0.9662	0.9271
Stacking	97.7684	0.9770	0.9747	0.9455

According to these results, the stacking method is the most efficient compared to voting methods for K-NN, random forest, and modlem classification algorithms. In addition to the results presented in Table 2, the resulting confusion matrices using different ensemble methods are also indicated in Figure 7.

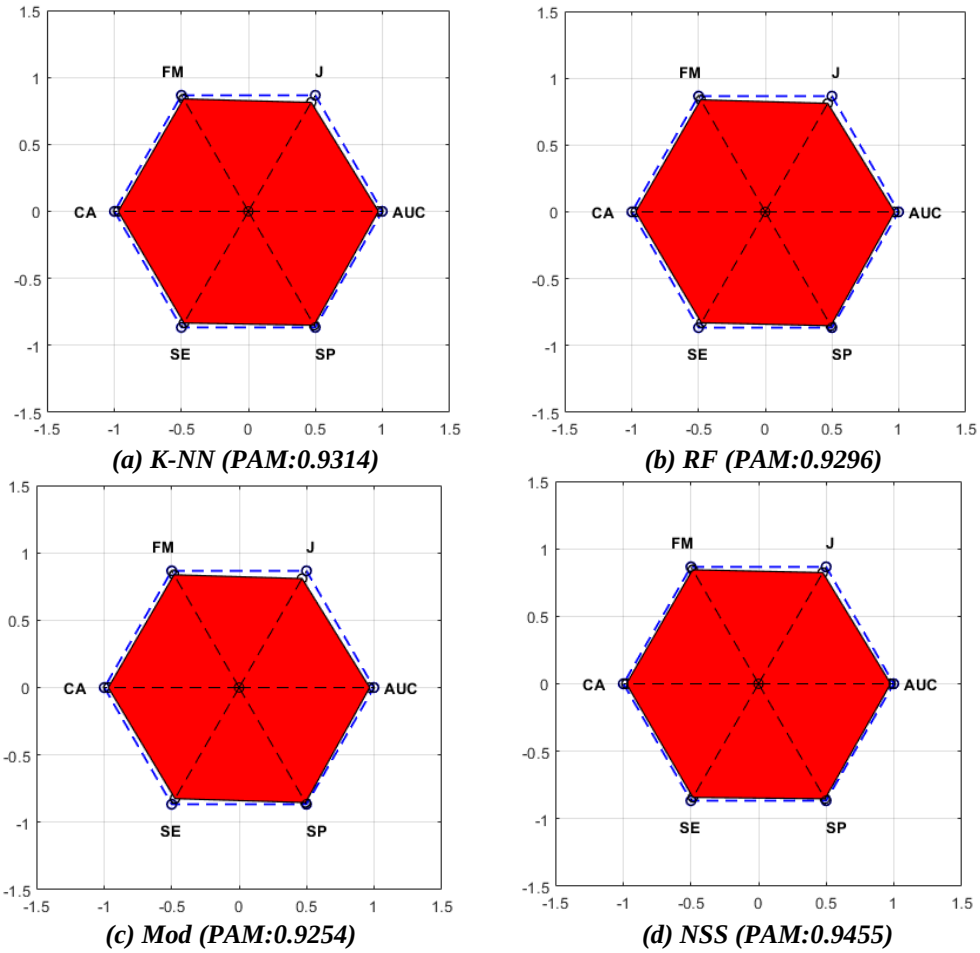
Figure 7. *Confusion matrices for proposed ensemble learning model under four feature selection algorithms*

We have also investigated the classification algorithms separately to see the effect of the stacking method on these algorithms. The results obtained when the K-NN, random forest, and the modlem classification algorithms used in the proposed ensemble method are applied to the dataset separately under the SUAE feature selection method are presented in Table 3. Under accuracy, AUC, F-score, and PAM metrics, K-NN gives the best results and the modlem gives the worst. While K-NN, random forest, and the modlem give accuracy %97.1954, %97.1351, %96.9843 respectively, stacking these methods with the naive bayes gives %97.7382. The proposed stacking method also outperforms the classification algorithms evaluated separately under other metrics. This shows that the proposed stacking method contributes classification problem effectively. The confusion matrices of the individual results and PAM results of the machine learning methods used in the proposed ensemble learning model are shown in Figure 9 and Figure 8 respectively.

Table 3. *Separate results of machine learning methods used in the proposed ensemble model*

ML Methods	**Accuracy (%)**	**AUC**	**F-Score**	**PAM**

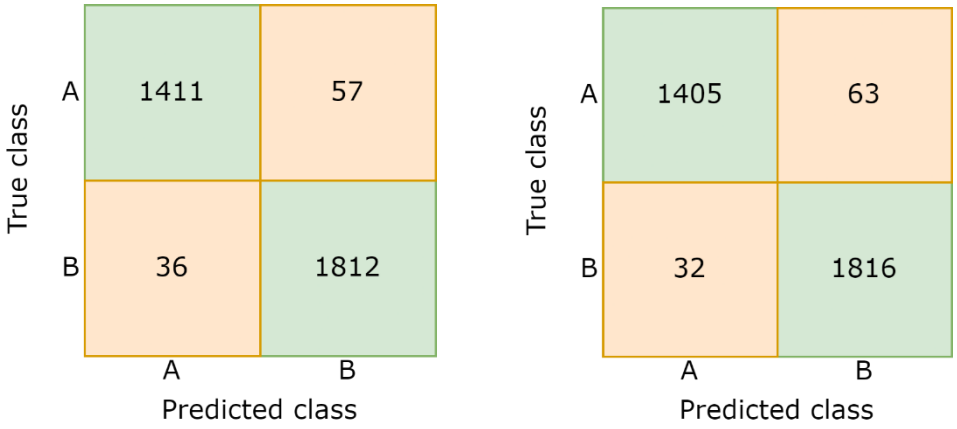
K-NN	97.1954	0.9708	0.9681	0.9314
Random Forest	97.1351	0.9699	0.9673	0.9296
Modlem	96.9843	0.9679	0.9654	0.9254
Proposed Ensemble Model	97.7684	0.9770	0.9747	0.9455



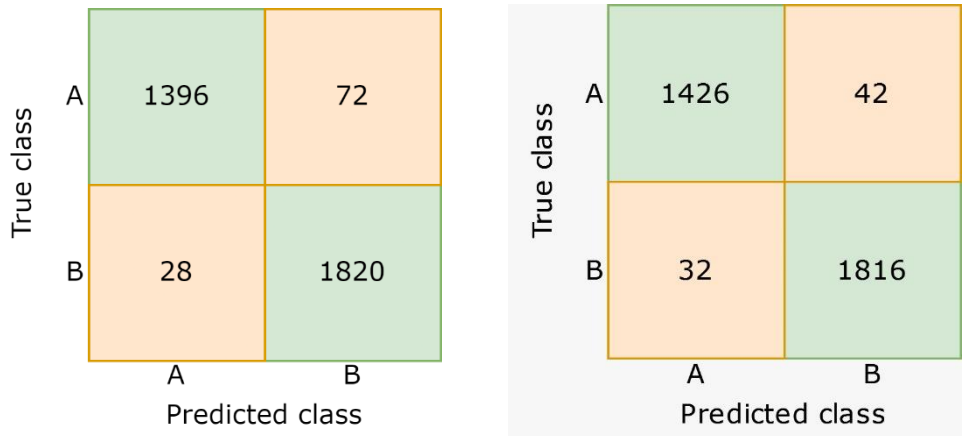
(a) K-NN (PAM:0.9314) *(b) RF (PAM:0.9296)*

(c) Mod (PAM:0.9254) *(d) NSS (PAM:0.9455)*

Figure 8. *PAM results for K-NN, Random Forest(RF), Modlem(Mod), and NSS(PM)*



(a) K-NN *(b) Random Forest*



(c) Modlem *(d) NSS*

Figure 9. *Confusion matrices for each of the machine learning methods used in the proposed ensemble learning model*

For SUAE and SAE feature selection methods, 30 features are ranked and %10 pruning is done to prefer the most relevant features. When the features selected by both SUAE and SAE are analyzed, it is seen that two features which are "popUpWindow" and "Favicon" are selected as mostly irrelevant while the most relevant features selected by filter approaches are "SSL final State", "URL of Anchor" and "Prefix Suffix". Figure 8d, Figure 6d, Figure 7c, Figure 9d and their associated tables relevant values indicate the proposed ensemble model. As seen from all comparisons, the proposed ensemble model for phishing detection achieves the best results.

The comparison of the proposed ensemble model with other studies using the dataset in the NSS is shown in Table 4. According to this table, the proposed ensemble model has higher performance than other approaches using the same dataset.

Table 4. *Summary of Related Works*

Ref.	Year	Models	Accuracy (%)
[4]	2022	Gradient boosting, random forest	97.0000
[5]	2022	S-shaped, V-shaped transfer function, KNN	97.0440
[41]	2022	XGBoost	97.0455
[7]	2022	Gradient boosting with XGBoost	97.3000
[9]	2022	Random forest, extra tree and decision tree	97.5600
[10]	2022	Standart neural network	94.8400
[11]	2022	LR, KNN, SVM	94.0000
[12]	2021	Random forest, logistic regression, SVM	94.1390
[14]	2021	CRAN-R, random forest	95.7000
[16]	2020	SVM + Adaboost	97.6100
[17]	2020	Random forest	97.2700
[18]	2020	Multi-filter	92.7200
[19]	2019	Stacking process	97.5000
[20]	2019	Decission tree (ID3)	96.7300
[22]	2017	Random forest	97.3600
[23]	2017	PSO feature selection, random forest	95.2000
NSS	2024	NaiveStackingSymmetric (ensemble) (proposed)	97.7684

A comparison of machine learning models performed using phishing datasets with different metrics such as F-score, AUC, precision, recall is given in Table 5. In this comparison, only studies that directly share the metrics or share the information and the values of the metrics are taken into

consideration. Only the accuracy values of other studies are given. In Table 5 there are studies that share more metric information or calculate metrics from the information provided.

Table 5. *More Detailed Summary of Related Works*

Ref.	Accuracy (%)	F-Score	AUC	Precision	Recall
[4]	97.0000	0.9685	0.9639	0.9622	0.9748
[5]	97.0440	0.9701	0.9704	0.9714	0.9695
[41]	97.0455	0.9736	N/A	0.9592	0.9794
[7]	97.3000	0.9740	N/A	0.9690	0.9820
[9]	97.5600	0.9722	N/A	0.9762	0.9682
[12]	94.1390	0.9343	N/A	0.9223	0.9466
[16]	97.6100	0.9760	0.9960	N/A	N/A
[17]	97.2700	0.9645	N/A	0.9456	0.9842
[18]	92.7200	0.9090	N/A	0.9124	0.9055
[22]	97.3600	0.9740	0.9940	N/A	N/A
NSS	97.7684	0.9744	0.9767	0.9827	0.9707

We have also investigated the selection of a splitting approaches for the training set. For this purpose, we have used 3 different splitting types which are 80/20 and 90/10 different from the used one in NSS which is 70/30. The results are presented in Table 6.

Table 6. *Different splitting approaches in training for NSS*

ML Methods	Accuracy (%)	AUC	F-Score	PAM
%70-30	97.7684	0.9770	0.9747	0.9455
%80-20	97.3768	0.9731	0.9709	0.9368
%90-10	98.0090	0.9796	0.9787	0.9525

V. FUTURE DIRECTIONS

In this section, there are directions for future studies on the subject. Machine learning approaches for cyber security are becoming more and more popular today. Therefore, the following directions are offered for future work.

- Models can be built and tested with a secure dataset for all other types of attacks, such as phishing.
- The rapid development of technology also increases cyber security needs and carries security strategies to different dimensions. With the emergence of quantum computers, many systems that are considered safe become insecure. Therefore, new protocols can be produced to make systems quantum resistant [42].
- With the development of 5G, 6G and IPv6 systems, the interest in Internet of Things (IoT) is constantly increasing. However, the nature of the wireless environment is insecure, leading to increased threats to IoT systems. In addition, the use of resource-constrained sensor devices increases the importance of lightweight security protocols. For this reason, machine learning-based security models can be developed for IoT systems that put a low load on the sensors [43].
- In mobile operating systems, security levels can be increased by detecting anomalies in data flow based on machine learning.
- Deep learning techniques, which are one of the popular topics of recent years and are frequently used in the field of image processing and whose use on numerical data are increasing, can be applied to phishing detection data sets [44].

VI. CONCLUSION

This paper proposes a new ensemble learning-based model for detecting whether a website is phishing or legitimate. Model training/test is performed on the dataset obtained from the UCI machine learning repository. Processes are performed using many methods and ablation studies are carried out. According to the results obtained, the best performance with %97.7382 accuracy belongs to the proposed ensemble model, which applies SUAE feature selection and stacks K-NN, random forest, and modlem approaches with Naive Bayes.

All other studies using the dataset are analyzed in detail and their results are compared with the proposed ensemble model. Here, not only accuracy but also the comparison is provided over different metrics such as AUC, F-score, precision, and recall. Based on the average performance of all these metrics, the proposed ensemble model has better performance than all other machine learning studies using the dataset.

In addition to a proposed novel model, the paper has an extensive literature review including machine learning-based phishing detection approaches and all other studies using the dataset. This shows that phishing attack is a very old and frequently used type of attack and reveals the necessity of taking precautions against this attack. Therefore, the NSS can effectively fills the gap in the literature. In future studies, we plan to develop deep learning-based cyber security models that have attracted great interest in recent years.

VII. REFERENCES

[1] A. Karakaya and S. Akleylek, "A survey on security threats and authentication approaches in wireless sensor networks," in 2018 6th International Symposium on Digital Forensic and Security (ISDFS), 2018, pp. 1–4. doi: 10.1109/ISDFS.2018.8355381.

[2] A. Karakaya and F. Arat, "A Survey on Security Requirements, Threats and Protocols in Industrial Internet of Things," International Journal of Information Security Science, vol. 10, no. 4. Şeref SAĞIROĞLU, pp. 138–152, 2021.

[3] K. Krombholz, H. Hobel, M. Huber, and E. Weippl, "Advanced social engineering attacks," J. Inf. Secur. Appl., vol. 22, pp. 113–122, 2015.

[4] A. Almomani et al., "Phishing website detection with semantic features based on machine learning classifiers: A comparative study," Int. J. Semant. Web Inf. Syst., vol. 18, no. 1, pp. 1–24, 2022.

[5] S. R. Sharma, B. Singh, and M. Kaur, "Improving the classification of phishing websites using a hybrid algorithm," Comput. Intell., vol. 38, no. 2, pp. 667–689, 2022.

[6] O. Aydemir, "A new performance evaluation metric for classifiers: polygon area metric," J. Classif., vol. 38, pp. 16–26, 2021.

[7] S. Maurya and A. Jain, "Malicious Website Detection Based on URL Classification: A Comparative Analysis," in Proceedings of Third International Conference on Computing, Communications, and Cyber-Security: IC4S 2021, 2022, pp. 249–260.

[8] H. Bouijij, A. Berqia, and H. Saliah-Hassan, "Phishing URL classification using Extra-Tree and DNN," in 2022 10th International Symposium on Digital Forensics and Security (ISDFS), 2022, pp. 1–6.

[9] J. V. Cubas and G. M. Niño, “Modelo de machine learning en la detección de sitios web phishing,” Rev. Ibérica Sist. e Tecnol. Informação, no. E52, pp. 161–173, 2022.

[10] M. A. A. Siddiq, M. Arifuzzaman, and M. S. Islam, “Phishing Website Detection using Deep Learning,” in Proceedings of the 2nd International Conference on Computing Advancements, 2022, pp. 83–88.

[11] W. Fadheel, W. Al-Mawee, and S. Carr, “On Phishing: URL Lexical and Network Traffic Features Analysis and Knowledge Extraction using Machine Learning Algorithms (A Comparison Study),” in 2022 5th International Conference on Data Science and Information Technology (DSIT), 2022, pp. 1–7.

[12] A. Hashim, R. Medani, and T. A. Attia, “Defences against web application attacks and detecting phishing links using machine learning,” in 2020 international conference on computer, control, electrical, and electronics engineering (ICCCEEE), 2021, pp. 1–6.

[13] S. Dangwal and A.-N. Moldovan, “Feature Selection for Machine Learning-based Phishing Websites Detection,” in 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), 2021, pp. 1–6.

[14] D. CJ and A. Gaurav, “Exposing model bias in machine learning revisiting the boy who cried wolf in the context of phishing detection,” J. Bus. Anal., vol. 4, no. 2, pp. 171–178, 2021.

[15] Z. Fan, “Detecting and Classifying Phishing Websites by Machine Learning,” in 2021 3rd International Conference on Applied Machine Learning (ICAML), 2021, pp. 48–51.

[16] A. Subasi and E. Kremic, “Comparison of adaboost with multiboosting for phishing website detection,” Procedia Comput. Sci., vol. 168, pp. 272–278, 2020.

[17] R. A. Kelkar and A. Vijayalakshmi, “ML BASED MODEL FOR PHISHING WEBSITE DETECTION,” challenge, vol. 7, no. 12, p. 2020.

[18] G. Sonowal and K. S. Kuppusamy, “PhiDMA--A phishing detection model with multi-filter approach,” J. King Saud Univ. Inf. Sci., vol. 32, no. 1, pp. 99–112, 2020.

[19] A. F. Nugraha and L. Rahman, “Meta-algorithms for improving classification performance in the web-phishing detection process,” in 2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), 2019, pp. 271–275.

[20] S. Adi, Y. Pristyanto, and A. Sunyoto, “The best features selection method and relevance variable for web phishing classification,” in 2019 International Conference on Information and Communications Technology (ICOIACT), 2019, pp. 578–583.

[21] I. Tyagi, J. Shad, S. Sharma, S. Gaur, and G. Kaur, “A novel machine learning approach to detect phishing websites,” in 2018 5th International conference on signal processing and integrated networks (SPIN), 2018, pp. 425–430.

[22] A. Subasi, E. Molah, F. Almkallawi, and T. J. Chaudhery, “Intelligent phishing website detection using random forest classifier,” in 2017 International conference on electrical and computing technologies and applications (ICECTA), 2017, pp. 1–5.

[23] D. R. Ibrahim and A. H. Hadi, “Phishing websites prediction using classification techniques,” in 2017 International Conference on New Trends in Computing Sciences (ICTCS), 2017, pp. 133–137.

[24] A. Almomany, W. R. Ayyad, and A. Jarrah, “Optimized implementation of an improved KNN classification algorithm using Intel FPGA platform: Covid-19 case study,” J. King Saud Univ. Inf. Sci., vol. 34, no. 6, pp. 3815–3827, 2022.

[25] Y. Liao and V. R. Vemuri, “Use of k-nearest neighbor classifier for intrusion detection,” Comput. \& Secur., vol. 21, no. 5, pp. 439–448, 2002.

[26] L. Breiman, “Random forests,” Mach. Learn., vol. 45, pp. 5–32, 2001.

[27] M. Schonlau and R. Y. Zou, “The random forest algorithm for statistical learning,” Stata J., vol. 20, no. 1, pp. 3–29, 2020.

[28] J. Stefanowski and others, “On rough set based approaches to induction of decision rules,” Rough sets Knowl. Discov., vol. 1, no. 1, pp. 500–529, 1998.

[29] G. I. Webb, E. Keogh, and R. Miikkulainen, “Na{\"\i}ve Bayes.,” Encycl. Mach. Learn., vol. 15, pp. 713–714, 2010.

[30] S. Chen, G. I. Webb, L. Liu, and X. Ma, “A novel selective naïve Bayes algorithm,” Knowledge-Based Syst., vol. 192, p. 105361, 2020, doi: https://doi.org/10.1016/j.knosys.2019.105361.

[31] D. E. Goldberg, Genetic algorithms. pearson education India, 2013.

[32] R. A. Welikala et al., “Genetic algorithm based feature selection combined with dual classification for the automated detection of proliferative diabetic retinopathy,” Comput. Med. Imaging Graph., vol. 43, pp. 64–77, 2015.

[33] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proceedings of ICNN’95-international conference on neural networks, 1995, pp. 1942–1948.

[34] A. Pradhan, S. K. Bisoy, and A. Das, “A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment,” J. King Saud Univ. Inf. Sci., vol. 34, no. 8, pp. 4888–4901, 2022.

[35] A. Ahmad and L. Dey, “A feature selection technique for classificatory analysis,” Pattern Recognit. Lett., vol. 26, no. 1, pp. 43–56, 2005.

[36] L. Yu and H. Liu, “Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution,” in Proceedings of the Twentieth International Conference on Machine Learning, AAAI Press, 2003, pp. 856–863.

[37] R. M. Mohammad, F. Thabtah, and L. McCluskey, “Phishing websites features,” Sch. Comput. Eng. Univ. Huddersf., 2015.

[38] A. Karakaya, A. Ulu, and S. Akleylek, “GOALALERT: A novel real-time technical team alert approach using machine learning on an IoT-based system in sports,” Microprocess. Microsyst., vol. 93, p. 104606, 2022, doi: https://doi.org/10.1016/j.micpro.2022.104606.

[39] R. Polikar, “Ensemble learning,” in Ensemble machine learning, Springer, 2012, pp. 1–34.

[40] O. Sagi and L. Rokach, “Ensemble learning: A survey,” Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 8, no. 4, p. e1249, 2018.

[41] M. F. Bin Karim, T. Hasan, N. Tazreen, S. Bin Hakim, and S. Tarannum, “An investigation of ML techniques to detect Phishing Websites by complexity reduction,” in 2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), 2022, pp. 144–149.

[42] E. Karacan, A. Karakaya, and S. Akleylek, “Quantum Secure Communication Between Service Provider and Sim,” IEEE Access, vol. 10, pp. 69135–69146, 2022, doi: 10.1109/ACCESS.2022.3186306.

[43] A. Karakaya and S. Akleylek, “A novel IoT-based health and tactical analysis model with fog computing,” PeerJ Comput. Sci., vol. 7, p. e342, 2021.

[44] A. Ulu, G. Yildiz, and B. Dizdaroğlu, “MLFAN: Multilevel Feature Attention Network With Texture Prior for Image Denoising,” IEEE Access, vol. 11, pp. 34260–34273, 2023, doi: 10.1109/ACCESS.2023.3264604.