A content and URL analysis-based efficient approach to detect smishing SMS in intelligent systems

Article *in* International Journal of Intelligent Systems · September 2022

DOI: 10.1002/int.23035

CITATION
1

READS
64

6 authors, including:

Ankit kumar Jain
National Institute of Technology, Kurukshetra
57 PUBLICATIONS **1,456** CITATIONS
SEE PROFILE

Wadee S. Alhalabi
King Abdulaziz University - Effat University
50 PUBLICATIONS **797** CITATIONS
SEE PROFILE

Dr.Ammar Almomani
Skyline University College
95 PUBLICATIONS **1,721** CITATIONS
SEE PROFILE

Received: 17 July 2022 | Accepted: 10 August 2022

DOI: 10.1002/int.23035

RESEARCH ARTICLE

WILEY

A content and URL analysis-based efficient approach to detect smishing SMS in intelligent systems

Ankit K. Jain[1] | **Brij B. Gupta**[2,3,4,5] | **Kamaljeet Kaur**[1] | **Piyush Bhutani**[1] | **Wadee Alhalabi**[3] | **Ammar Almomani**[4,6]

[1]Department of Computer Engineering, National Institute of Technology, Kurukshetra, India

[2]International Center for AI and Cyber Security Research and Innovations & Department of Computer Science and Information Engineering, Asia University, Taichung, Taiwan

[3]Department of Computer Science, King Abdulaziz University, Jeddah, Saudi Arabia

[4]Research and Innovation Department, Skyline University College, Sharjah, UAE

[5]Lebanese American University, Beirut, Lebanon

[6]IT-department-Al-Huson University College, Al-Balqa Applied University, Irbid, Jordan

Correspondence
Brij B. Gupta, International Center for AI and Cyber Security Research and Innovations & Department of Computer Science and Information Engineering, Asia University, Taichung 413, Taiwan
Email: gupta.brij@gmail.com

Funding information
Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, Saudi Arabia, Grant/Award Number: RG-4-611-38

Abstract

Smishing is a combined form of short message service (SMS) and phishing in which a malicious text message or SMS is sent to mobile users. This form of attack has come to be a severe cyber-security difficulty and has triggered incredible monetary losses to the victims. Many antismishing solutions for mobile devices have been proposed till date but still, there is a lack of a full-fledged solution. Therefore, this paper proposes an efficient approach that analyzes text content and uniform resource locator (URL) presented in the SMS. We have integrated the URL phishing classifier with the text classifier to improve accuracy as some of the SMS contain the URL with no text or much less text. To find out rare words in a report, depending upon the frequency of term (TF) and the reciprocal of document frequency TF-inverse document frequency (IDF), a weighting framework TF-IDF is used. We have used two data sets for both text as well as for URL phishing classifier and used a synthetic minority oversampling technique to balance the training data. The voting classifier simply merges the findings of each classifier passed into it and predicts the output on the basis of voting. In proposed approach integrating KNN, RF, and ETC can detect smishing messages with a 99.03% accuracy and 98.94% precision rate which is

relatively efficient compared with existing ones like SmiDCA model which has the given accuracy of 96.40% using Random Forest classifier in BFSA, Feature-Based it has an accuracy of 98.74% and 94.20% true positive rate and Smishing Detector it shows an overall accuracy of 96.29%.

KEYWORDS
machine learning, mobile phishing, short message service, smishing, uniform resource locator

1 | INTRODUCTION

1.1 | Context

The smishing word is derived from SMS + phishing. Here SMS is short message service used to send and receive small text messages on mobile phones.[1] Smishing is defined as a malicious activity of sending text messages pretending to be from a reputable company to steal mobile users' personal and financial information.[2] As mobile phones are small in size, easily accessible, and have low production costs, which are increasing mobile phones' popularity.[3] In mobile-based phishing, SMS is the most used mode to send fake messages as these messages are short in length and difficult to predict. Attackers can steal information in many ways, that is, either by replying back or by letting the user click on the phish link which is attached with the message. When the user clicks on the phishing link, the user's information may be shared with the attacker without his knowledge or a malicious mobile application will be downloaded that will track all his movements thus harming user's personal life.[4,5] This has led to monetary damage to a large number of people and has even prompted self-destruction of its victims, creating a serious social issue.[6]

Figure 1 presents the life cycle of smishing attacks. Attackers set the message content as to be very real. Uniform resource locator (URLs) contained in the messages may have very high similarities with the actual/original company like facebok.com, amazon-firstreply.com, dehlivery.com.[7-9] This is done by misspelling one or two characters so the users accidentally consider it as a real/nonspam link. Messages content is based on the latest trends like COVID or can involve general messages including messages from banks or a delivery company.

1.2 | Problem definition

Smishing involves stealing of the user's financial details thus causing financial loss to the user. As per the FBI cybercrime report (2020) the number of smishing, phishing, vishing, and pharming victims was 241,342 and the total loss of money due to these attacks was more than $54 million.[10] Net banking facilitates users with having all bank activities with just a few clicks. Users can withdraw money, open accounts, create deposits, transfer money digitally, and so on. All this acts as an advantage for attackers to steal users' financial details by sending them fake

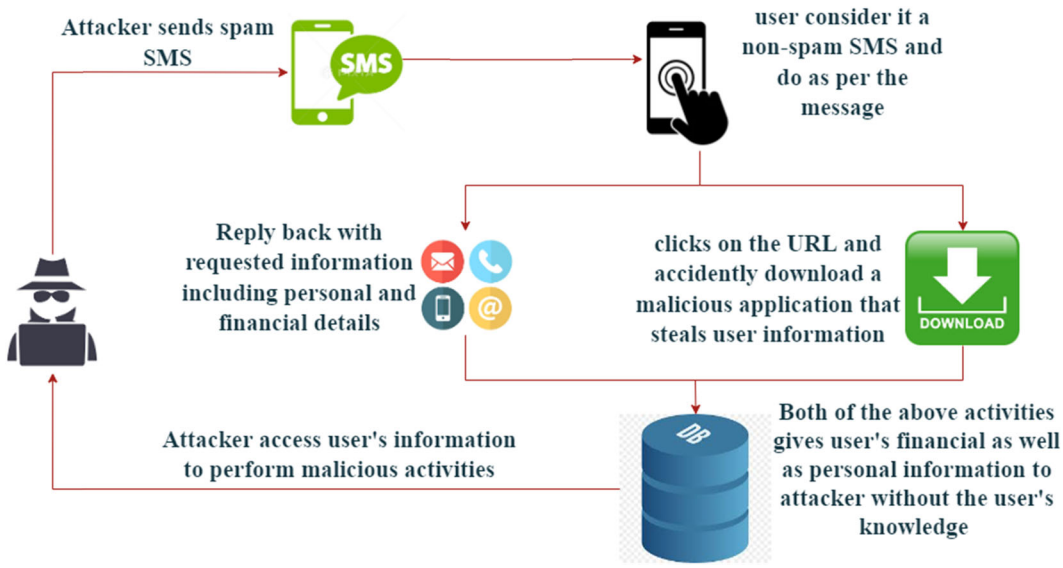

FIGURE 1 Lifecycle of smishing attacks [Color figure can be viewed at wileyonlinelibrary.com]

SMS. Thus, by obtaining the user's financial and personal details, attackers steal the user's money.[11] This section presents some reasons why people fall under smishing attacks.

Lack of awareness about smishing: Many people are not aware of this malicious activity of stealing information. Since there is very little knowledge about this, getting harmed by such attacks is very easy. As per the survey conducted in 2020, only 30% of the people correctly know about smishing, 21% of the people consider smishing as spam but there is a difference between smishing and spam and the rest (49%) do not know about smishing.[12]

Smishing is the most popular mobile-based phishing: With the availability of the Internet and mobile phones, smishing attacks or we can say attacks through mobile have increased drastically. Among all communications mediums, SMS is the only mode which has a high and fast view rate.[13] Figure 2 shows the mobile-based phishing using different modes.

Increasing number of attacks: As per the survey conducted in 2020, smishing attacks have increased to 328% in 2020.[14] Covid act as a boom in 2020 in increasing smishing attacks. All the information related to Covid is provided to user through SMS or internet. This acts as an opportunity for attackers to mislead people by sending fake SMS so that people will consider it real and hence fall in their trap.

Covid acts as a boost for smishing attacks: A lot of things have changed in covid. Digitalization is at its peak and covid acts as a boost for smishing attacks.[15,16] Updates about vaccine or covid-related information is done through message. Figure 3 shows a few examples of covid-based smishing messages. These types of smishing attacks include the message that involves:

- Where and how you can get Covid tests if you are feeling sick.
- Where and how you can get vaccines.
- Where and how you can download the vaccine certificates.
- How you can check if you came in contact with a covid positive person.
- How you can avail insurance.

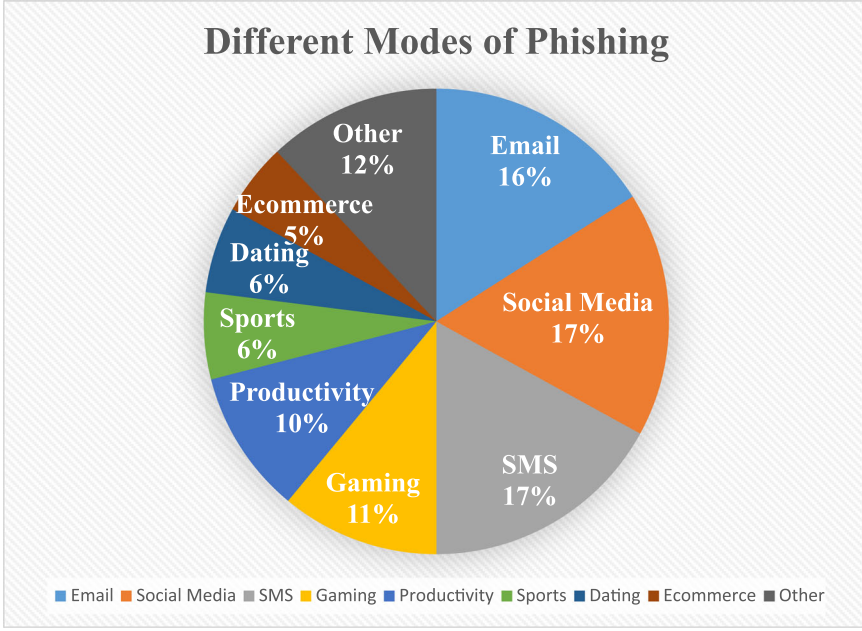

FIGURE 2 Mobile-based phishing using different modes [Color figure can be viewed at wileyonlinelibrary.com]

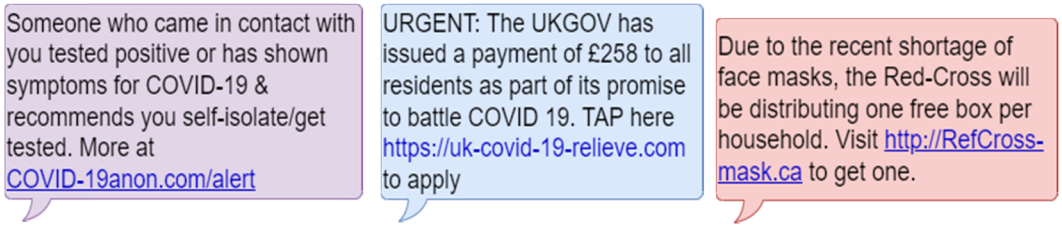

FIGURE 3 Examples of covid-based smishing messages [Color figure can be viewed at wileyonlinelibrary.com]

- Information about lockdown.
- How to isolate if you are tested positive.

Advancement in Ecommerce and hence delivery: With the advancement in ecommerce and availability of home delivery in almost all the areas, home deliveries are in trend.[17,18] Attackers use this opportunity and sends fake delivery message to user to steal information.[19]

1.3 | Proposed method

The proposed approach uses two different models for classifying the messages as smishing or nonsmishing, namely smishing text classifier and URL phishing classifier. Smishing Text Classifier classifies the message as smishing or nonsmishing based on the text content. This model is trained only on the text tokens that are extracted from the data set. Some of the SMS contain the URL with no text or much less text. So, it becomes very difficult to analyze or categorize such SMS. To overcome

that problem, the URL phishing classifier is integrated along with the text classifier to improve accuracy and precision. The training data set consists of two different types of data sets that are used for training the proposed model. Since the data is imbalanced which results in biased outcomes and no available paper has taken any steps for handling an imbalanced data set. Therefore, the proposed approach handles an imbalance data set using Synthetic Minority Oversampling Technique (SMOTE). The approach has used XGB, GBDT, RF, BgC, KNN, ETC, DT, LR, AdaBoost, BNB, MNB, SVC, and GNB machine learning algorithms to train the model. Further, we have calculated the accuracy of model on all data sets unscaled, scaled, and oversampled. The random forest model the has highest accuracy 98.75% on oversample data set among all the machine learning models.

1.4 | Major contributions

Following are the major contributions of the proposed method

- The proposed approach works efficiently on small text.
- We have balanced the datasets to improve the detection accuracy and precision rate.
- We have scaled the datasets to improve the working of machine learning algorithms
- A vectorization algorithm is used to vectorize text into a format more suitable for machine learning.
- We additionally trained phishing URLs for better detection accuracy.
- Selected features are independent of third-party service and message history.

The rest of the paper is organized as follows: The related work is discussed in Section 2, and the proposed method is discussed in Section 3. Section 4 provides the implementation and data set details. Section 5 provides the details of the results and observations. Finally, Section 6 concludes the paper and discusses some future work.

2 | RELATED WORK

This section presents various existing smishing detection approaches. Most of the existing approaches used content analysis. Various papers used traditional machine learning algorithms but some recent papers also used advanced machine learning algorithms and artificial intelligence. Various researchers have used different methods for URL classification, that is, either by making a URL classifier or by including a third-party application to check the validity of the URL.[20]

S-detector[21]: It is a security model for detecting smishing attacks on mobile devices by applying the proposed model to Naive Bayes classifier. It uses a statistical learning method to filter smishing messages.

Rule-based[22]: The classification approach used is decision tree, RIPPER, and PRISM. To filter smishing messages rule selection is essential. Riles which are selected should be correlated to the message type such that accuracy for detection of smishing messages can be increased. SMS messages are of less length, that is, they have a length limit and it contains only text, that is, SMS does not contain any attachments like images, docs, and so on. In comparison to email, there is no text limit and it can also contain attachments like images, docs, graphics, and so on. The rule-based approach is done in three phases namely preprocessing, rule extraction, and classifier training and testing. Data set used in rule-based is Almeida spam data set,[23] filtering of the data set is done manually to make a

new smishing data set of 5169 messages of which 4807 are ham and 362 are smishing messages. The result of this approach provides 99% true negative rate.

SmiDCA[24]: The SmiDCA model has reduced dimensionality. The machine learning based model evaluated the accuracy of 96.40% for the English data set with the help of a random forest classifier in BFSA. The proposed approach in this paper also includes a URL phishing detector for better smishing classification.

Feature-based[25]: This approach offers novel features that separate the smishing messages from the normal messages. The most suspicious keywords used by attackers are also identified. The data set used in this approach Almeida spam data set,[23] filtering of the data set is done manually to make a new smishing data set of 5169 messages of which 4807 are ham and 362 are smishing messages. This approach has an overall accuracy of 98.74%. It is very efficient for the detection of zero-hour attack. However, the authors not handled an imbalance data set.

Smishing-detector[26]: It is a security model to distinguish smishing through SMS content examination and URL behavior analysis. It decreases misleading positive outcomes by utilizing efficient strategies. The approach contains four modules specifically SMS content analyzer, URL filter, source code analyzer, and Apk download detector. Client assent taken while downloading the file is likewise analyzed in this module. The outcome of the proposed approach in this paper shows an exactness of 96.29%.

Smishing-detector implemented using Neural Network[27]: This paper presented an algorithm for the smishing-detector[28] model and implemented it using neural network. The author extracted the best seven features using a neural network and calculated the accuracy corresponding to each extracted feature. Neural network performed better than all other machine learning algorithms and increased the accuracy by 1.11%. Its final accuracy is 97.40%.

3 | PROPOSED APPROACH

The primary objective of our approach is to filter smishing messages from nonsmishing messages. To achieve our goal, the proposed method is the combination of two models namely, the smishing text classifier and the phishing URL classifier. Figure 4 shows the detailed architecture of the proposed method. We have used two different data sets for training the respective models, so all four steps will be performed for both models. Initially, when the message has been received, it will be scanned for the presence of the URL. If the URL is present, it will be forwarded to the URL phishing classifier in which all the four steps, as shown in Figure 4, will be executed. Otherwise, if the URL is not present, then the text message will perform all four steps according to the text phishing classifier.

In a text classifier, the first step is data cleaning in which we have labeled the data as well as removed unwanted text. Further, the cleaned data is forwarded to the preprocessing stage where the approach performs five steps: lowercase, tokenization, removal of special characters, removal of stopwords, and stemming as shown in Figure 5. After data preprocessing, the next stage is feature extraction in which the system extracted six different features from the text. The final stage is model building, which includes token analysis, vectorization, and preprocessing data set for training the model.

Similarly, the URL phishing classifier approach performs all four steps. The first step is data cleaning in which the system removes non-English entries, label the data and removes the null entries. The second stage is data preprocessing which includes tokenization and stemming. Further, in the third stage, the approach extracted six features from the URL. Finally, the model building stage includes token analysis, vectorization, and preprocessing data set for training the URL model.

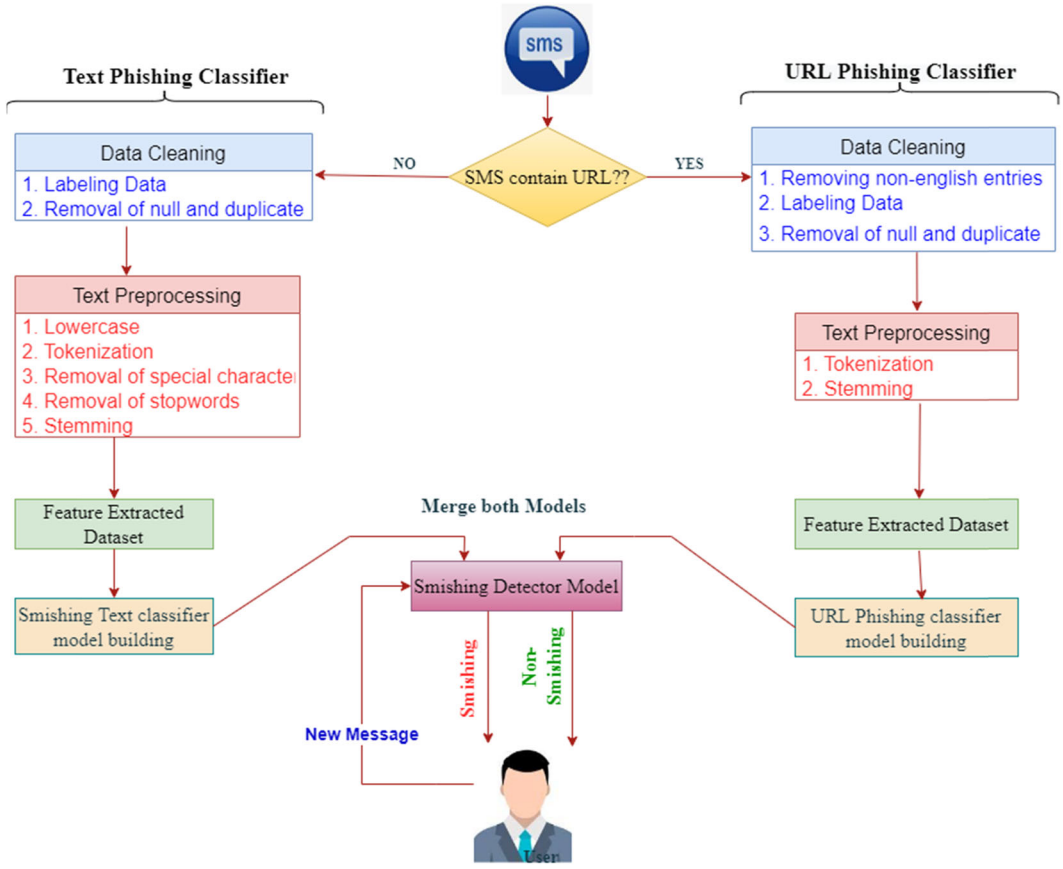

FIGURE 4 Detailed architecture of the proposed method [Color figure can be viewed at wileyonlinelibrary.com]

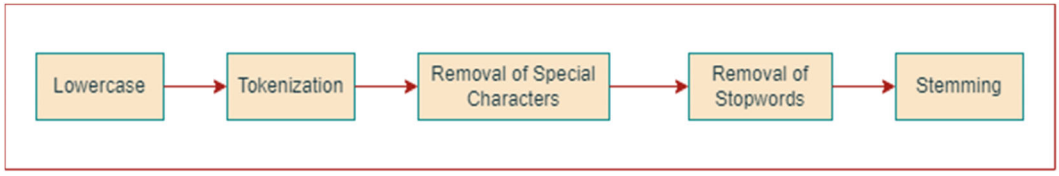

FIGURE 5 Steps involved in preprocessing of smishing messages [Color figure can be viewed at wileyonlinelibrary.com]

In addition to the above, system also performed oversampling on the data set to balance the data sets. As the training data set contains approximately 90% of the nonsmishing messages and 10% of the smishing messages, handling the data set was necessary. Otherwise, the classifier will give biased results. There are two ways to handle an imbalanced data set:

(a) Undersampling: In this type of handling, the data in the majority class is reduced whereas the data in the minority class remain as it is. In this method, as we reduce the data so there is a feature reduction. Hence, this method is not preferred.

(b) Oversampling: In this type, the majority class remains as it is whereas the minority class is increased to match the count of the majority class. In this, new data points are introduced which are very close to minority class points. In this type, there is no feature reduction, so this method is the preferred one.

The steps involved in the execution are explained below.

3.1 | Data cleaning

This process involves cleaning data for further preprocessing. It is a process of removing unwanted, null, incomplete, or duplicate data within the data set.[28]

3.2 | Data preprocessing

This phase involves preprocessing the data set for training the model, that is, converting the data from raw form into a usable form or understandable form.[29,30] Data preprocessing involves three phases, that is, data cleaning, data transformation, and data reduction. Data transformation involves normalization and attribute selection.[31] Data reduction involves dimensionality reduction. Figure 5 shows the steps of preprocessing.

Data preprocessing for smishing messages data set

The steps of preprocessing are:

(a) Lowercase: This involves converting the SMS into the lowercase for generalization
(b) Tokenization: This involves breaking the SMS into tokens. For our model, we use the nltk tokenizer to divide the string into a list of words/tokens.
(c) Removal of special characters: This phase involves the removal of special characters from the message. Tokens obtained from the above step contain special characters, which do not play any role in the classification of messages. Therefore, in this step, we remove special characters. Special Characters includes!"#$%&'()*+,-./:;<=>?@[\]^_'{|}~
(d) Removal of stopwords: Stopwords are the most common words which are used in grammar, that is, prepositions, pronouns, conjunctions, and so on.
(e) Stemming: Stemming is a process of reducing a word to a base word. To avoid duplicate entries, stemming is performed as a preprocessing phase on the tokens obtained from tokenization. Stemming of root word "like" includes likes, liked, liking, likely, and so on.

Data preprocessing for URLs:

(a) Tokenization: This involves breaking the URL into tokens. For our model, we use nltk tokenizer RegexTokenizer (r'[A-Za-z] to divide a string into a list of words/tokens.
(b) Stemming: Stemming is the process of reducing a word to a base word.[32] To avoid duplicate entries, stemming is performed as a preprocessing phase on the tokens obtained from tokenization. Stemming of the root word "like" includes likes, liked, liking, likely, and so on. In stemming tokens obtained from the tokenization of the URLs, we have used Snowball Stemmer.

3.3 | Exploratory data analysis

In this phase, data analysis is done to extract the features that are relevant for model training. Moreover, this phase also reduces the features which are not playing any role or do not have any major contribution in predicting the result.[33]

Exploratory data analysis for smishing text data set:

A message is classified as smishing based on different factors involving the length of the message, whether it contains any link or not, the number of abbreviations in it, and so on. Following are the different factors:

(a) Label counts: It is observed that data is imbalance, that is, 91.93% of the messages are nonsmishing while only 8.7% of the total messages are smishing.
(b) Dependency on number of characters in categorizing message: Figure 6 represents the count of smishing and nonsmishing messages based on number of characters in the message. From the graph, it is clear that smishing messages are neither of very fewer character counts, that is, less than 50 nor of very high character counts, that is, greater than 200. If the messages have very fewer characters or a large number of characters then it is more likely to fall under the nonsmishing category. For messages having average length, that is, between 100 and 200, these messages can be smishing or nonsmishing based on the text it contains.
(c) Dependency on number of words in categorizing message: Figure 7 presents the count of smishing and nonsmishing messages based on the number of words in the message. From the graph, it is clear that smishing messages are neither of very fewer word counts, that is, less than 5 nor of very high character counts, that is, greater than 50. For messages having an average word count, that is, between 10 and 45, these messages can be smishing or nonsmishing based on the text it contains.
(d) Dependency on number of sentences in categorizing message: Figure 8 represents the count of smishing and nonsmishing messages based on the number of sentences in the

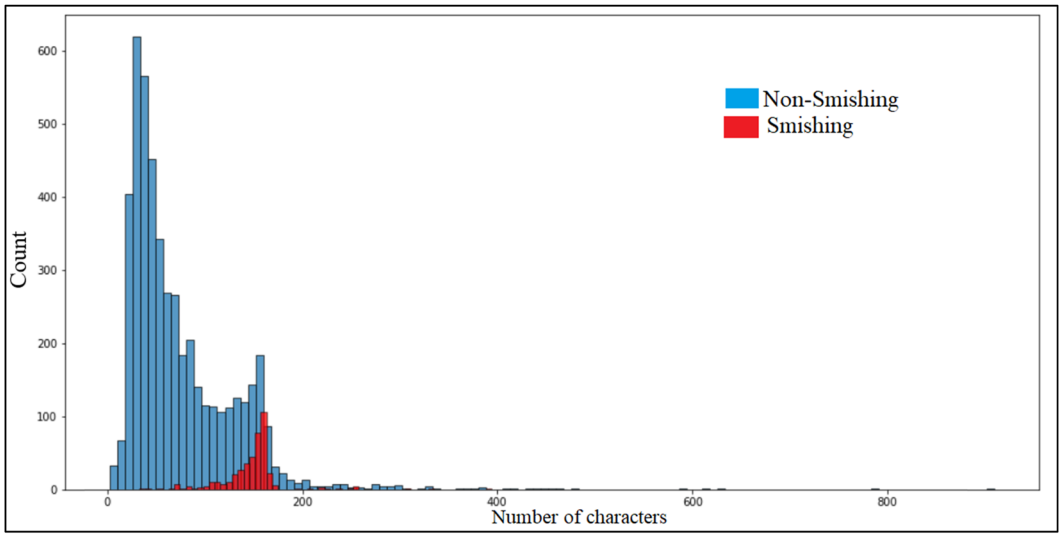

FIGURE 6 Count of smishing and nonsmishing messages based on number of characters [Color figure can be viewed at wileyonlinelibrary.com]

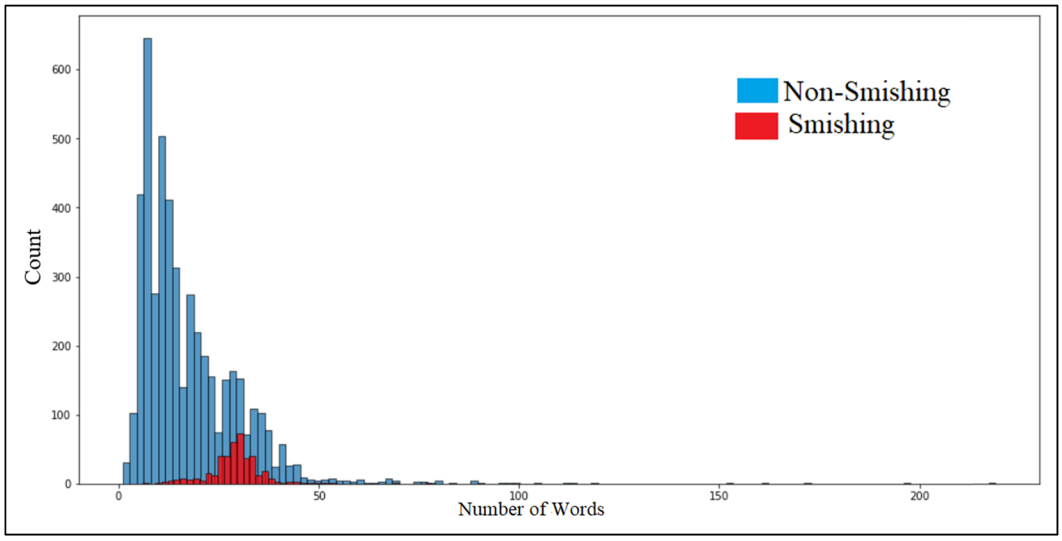

FIGURE 7 Count of smishing and nonsmishing messages based on number of words [Color figure can be viewed at wileyonlinelibrary.com]

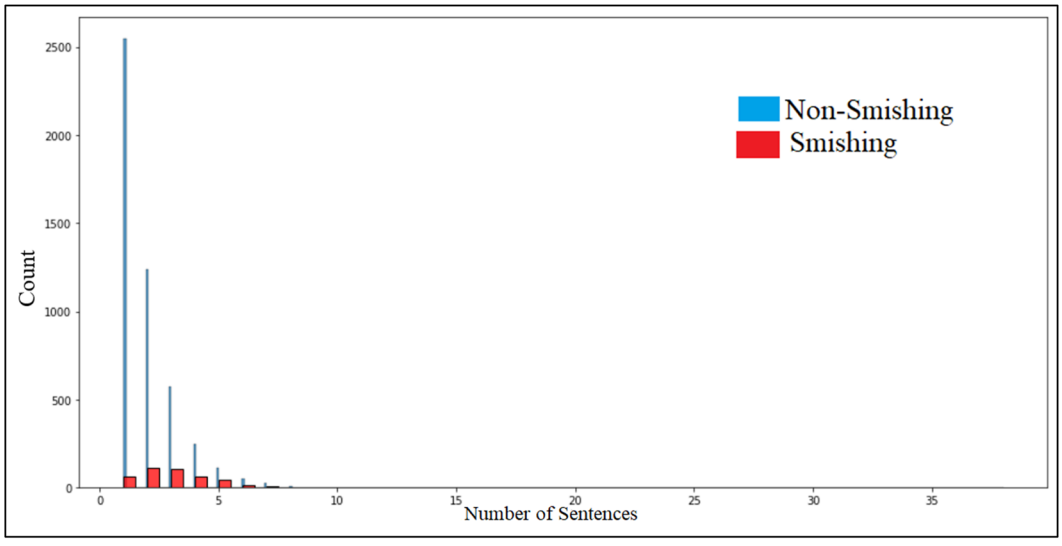

FIGURE 8 Count of smishing and nonsmishing messages based on number of sentences [Color figure can be viewed at wileyonlinelibrary.com]

message. Sentences for both smishing and nonsmishing messages are the same. Hence it can be difficult to classify the message on the basis of the number of sentences.

(e) Relation between Number of characters, words, and sentences: In Figure 9, it can be seen that the number of characters has a maximum dependency on target, that is, 0.34. The number of words and number of sentences has less influence on the target. As the number of characters and number of words has a very high dependency (0.96), so we

FIGURE 9 Heatmap of number of characters, number of words, and number of sentences [Color figure can be viewed at wileyonlinelibrary.com]

will select the number of characters as a feature and reduce other features, that is, the number of words. Between the number of characters and the number of sentences, the number of words has more influence on the target. Therefore, we chose the number of characters as a feature.

(f) Dependency of URLs in categorizing message

From our analysis, we found that 69% of the messages which contain URLs are smishing messages and 31% are nonsmishing messages.

Exploratory data analysis for URL phishing classifier:

(a) Label counts: It is observed that data is imbalanced, that is, 77% of the URLs are nonphishing while the rest are phishing.
(b) Dependency on the length of the URL in categorizing a URL: We analyze that URLs which have a length less than 100 have more chances of being a nonphishing URLs, that is, 80% of the URL whose length is less than 100 are nonphishing URLs. URLs with a length greater than 200 have a high probability of being phishing URLs, that is, on average, there are less than 2% chances of a nonphishing URL having a length greater than 200. So, from our analysis, we conclude that, if a URL has a length greater than 200, then it is more likely to be a phishing URL.
(c) Dependency on the number of question mark in categorizing URL: In our data set, we found that URL which contains no question mark or only one question mark is more likely to be a nonphishing URL. We conclude that if the URL contains two or more than two question marks then it is more likely to fall under the phishing category.
(d) Dependency on the number of Digits in categorizing URL: We observe that digit count is not playing much role in classifying URLs as phishing or nonphishing.
(e) Dependency on containing @ symbol in categorizing URL: 88% of the total URLs that contain "@" symbol belong to the phishing category while the rest belong to nonphishing.
(f) Dependency on the count of "." in domain name in categorizing URL: If the count of the dots (.) in the domain name is greater than 5, then the probability of the URL being phishing will be high.

3.4 | Model building

The proposed approach is an integration of two models, that is, a smishing text classifier and a URL phishing classifier. In this phase, we have built both models from the processed data set that we get from the above phase.

Model building for smishing text classifier:

(a) Tokens analyzation for smishing text classifier:

This phase of model building involves analyzing the tokens for smishing and nonsmishing messages. Figure 10 presents the word cloud for smishing messages. From the word cloud, we can easily observe words that are occurring maximum in smishing messages. Words like call, text, free, prize, claim, and so on, are having a maximum frequency in smishing messages. While training the model, we are using these words to categorize the message as smishing or nonsmishing. Figure 11 presents the top 30 words which are occurring maximum times in smishing messages. Works like call, ur, like, get, and so on, are having a maximum frequency in smishing messages. While training the model, we are using these words to categorize the message as smishing or nonsmishing.

FIGURE 10 Word cloud for smishing message [Color figure can be viewed at wileyonlinelibrary.com]

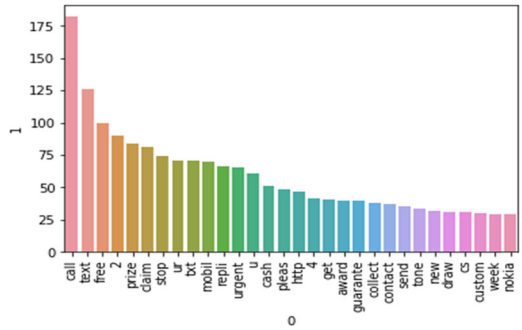

FIGURE 11 Frequency distribution for smishing messages [Color figure can be viewed at wileyonlinelibrary.com]

(b) Vectorizing and processing the data set for model training for Smishing Text Classifier:

This phase involves vectorizing the data set for different numbers of features, that is, 2000, 2500, 3000, 3500, and 4000. Vectorization means providing numeric value to data. Further, this step involves extracting the features for model building by getting text converted into numeric. By vectorizing, computation becomes much faster as compared to nonvectorized implementation.

3.4.1 | TF-IDF vectorization algorithm

TF-IDF stands for term frequency inverse document frequency. It is defined as the calculation of the relevancy of a word in a given corpus.[34] TF-IDF is probably the best measurement to decide how significant a term is to a text in a corpus. TF-IDF is a weighting framework that assigns a load to each word in a report depending upon the TF and the reciprocal of document frequency TF-IDF.[1]

Terminology:

TF: In document d, the recurrence addresses the number of occurrences of a given term/word t.

$$\mathrm{TF}(t, d) = \text{number of } t \text{ in } d/\text{total count of words in } d. \tag{1}$$

DF: It is the number of times the term t is occurring in the document set N.

$$\mathrm{DF}(t) = \text{Total occurrence of } t \text{ in the set documents } N.$$

IDF: It is used to find how relevant a particular word is. The IDF of the term is the count of documents in the corpus separated by the frequency of the text.

$$\mathrm{IDF}(t) = \log(N/\mathrm{DF}(t)). \tag{2}$$

where $N =$ number of documents, $N(t) =$ count of documents containing the term t

$$\text{TF-IDF}(t, d) = \mathrm{TF}(t, d)*\mathrm{IDF}(t). \tag{3}$$

The high TF-IDF score of a term means the term has a high frequency in a document, and low document frequency in the corpus. If the IDF value approaches 0, it means the term is appearing in almost all documents and making the TF-IDF closer to 0. If both IDF and TF values are high then TF-IDF will be high which means the word is very rare in the whole document but frequent in a document.

3.4.2 | Feature extraction

The features extracted for a model building comprise four categories with 14 unique different sets of features. Each set consists of its own distinguishing feature based on the phase it passed through. The feature extracted datasets are classified into four categories explained below:

(a) TF-IDF vectorized data set: Using TF-IDF vectorization, we have extracted top (1000, 2000, 2500, 3000, and 4000) rare words that are best influencing the result (smishing/nonsmishing).
(b) Scaled TF-IDF: It is the feature extracted data set with max (2500, 3000, and 3500) features and values in the features are scaled from 0 to 1 by using MinMaxScaler. MinMaxScaler is characterized by default hyperparameters. Once characterized, we can call the fit_transform() function and pass it to our data set to make a changed form of our data set.
(c) TF-IDF oversampled data: it is the same as the TF-IDF vectorized data set with oversampling. In this type, the majority data set remains as it is whereas the minority data set is increased to match the count of the majority data set.

We have used Synthetic Minority Over-sampling Technique (SMOTE)[35] to balance the data sets. It is an oversampling technique in view of making synthetic examples for the minority classes. The calculation takes every minority class test and presents manufactured examples along the line joining the ongoing case and a portion of its k closest neighbors from a similar class. Contingent upon how much oversampling is required, the calculation randomly picks the k closest neighbors of them and forms the sets of vectors that are utilized to make the synthetic tests. The new instances make bigger and denser choice areas. This assists classifiers with advancing more from the minority classes in those choice regions, as opposed to from the large classes surrounding those areas.

(d) Data set with additional features:

TF-IDF_3000_nchar: this contains the top 3000 words which are picked up by the TF-IDF vectorizer and along with that there is one additional feature "nchar" which is the count of characters in the message.

TF-IDF_3000_url: It is the same as above but contains "url present or not" as the 3001st feature.

3.4.3 | Building the smishing text classifier

Figure 12 presents the model which is involved in building a smishing classifier. Processed data set obtained from vectorizing, oversampling, and scaling is provided as training data for different models. Each model performs differently giving different accuracy and precision.

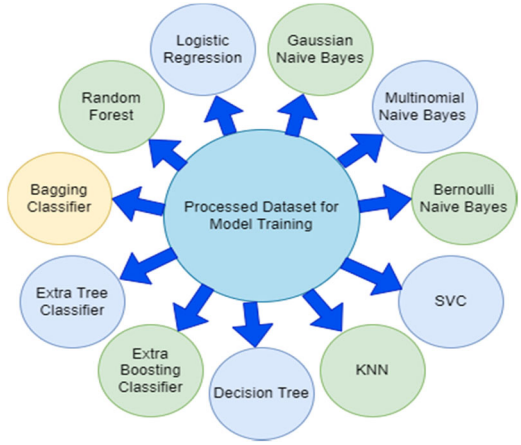

FIGURE 12 Models involved in building smishing classifier [Color figure can be viewed at wileyonlinelibrary.com]

Random forest, KNC, and Extra Tree Classifier perform exceptionally well on oversampled vectorized data with max of 3000 features. As these three models perform very well, they act as selection criteria for the voting classifier to obtain the result. The voting classifier simply aggregates the result of input models and predicts the result. Thus, giving better accuracy and precision.

3.4.4 | Model building: URL phishing classifier

(a) Tokens analyzation for URL phishing classifier

This phase of model building involves analyzing the tokens that were obtained after tokenizing the final formatted URL.

Figure 13 presents the word cloud for phishing URLs. We can observe words that are occurring maximum in phishing URLs. Works like net, admin, login, index, php, en, and so on, are maximum frequency words in phishing URLs. These words can also be considered phish hinted words. While training the model, we are using phish hinted words to categorize the URL as phishing or nonphishing.

(b) Vectorizing and processing the data set for model training

This phase involves vectorizing the data set that we obtained after data cleaning. This phase is also called feature extraction. In our proposed approach, we used a count vectorizer for feature extraction.[20] Count vectorizer converts the text or the collection of text into a list of tokens. This results in the formation of a sparse matrix with columns named as the tokens and the value representing the presence of the word in that row or text.

(c) Building the model

This phase involves building the model, that is, training and testing the most fitted model. Features from the above phase, that is, by count vectorizing are passed as a training data set for the model. For our proposed approach we chose the Logistic Regression Model, Multinomial

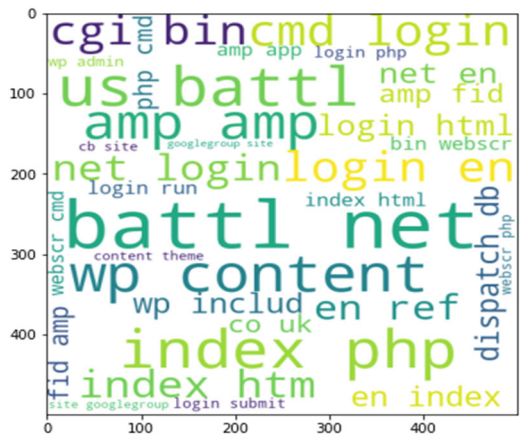

FIGURE 13 Word Cloud for phishing URL [Color figure can be viewed at wileyonlinelibrary.com]

Naïve Bayes, and Bernoulli Naive Bayes as the models for categorizing URLs as phishing or nonphishing.

4 | DATASET AND IMPLEMENTATION DETAILS

4.1 | Data set

We have used two data sets for both text as well as for URL phishing classifier. Firstly, for text classifier, we have used Almeida spam data set,[23] manually collected smishing messages and text extracted from images or screenshots available on the web. Second, the URL classifier data set consists of 507,195 unique entries in which 28% is phishing and the rest 72% is nonphishing. Detail of the data set is shown in Table 1.

4.2 | Implementation details

Given a data set of messages, we have trained and cross-validated a proposed machine-learning-based model, which is built by merging two classifiers namely, text phishing classifier and URL phishing classifier, to identify a message as smishing or nonsmishing. We have used the Pandas library in Python to access csv files that contain message content. We used UrlExtract in Python to extract URL from the text. Extracted data sets are further examined for nonsmishing contents: duplicates and null values are removed from the data set. In the text phishing classifier, we have used a count vectorizer and TFIDF vectorizer for the vectorization of data sets. MinMaxScanner is used to scale the datasets and SMOTE is used to balance the datasets. We have used XGB, GBDT, RF, BgC, KNN, ETC, DT, LR, AdaBoost, BNB, MNB, SVC, and GNB machine learning algorithms to train our model.[36–38] We have calculated the accuracy of model on all datasets unscaled, scaled, and oversampled.

TABLE 1 Data set description for Smishing text classifier and URL classifier

Smishing data set	URL data set
(a) Manually filtered Almeida spam to make a smishing data set which consists of 5169 messages out of which 362 are smishing messages and 4817 are nonsmishing message which also include spam message that does not harm user example promotional messages. (b) Manually collected smishing messages from a survey conducted online by the authors. This data set consists of 121 smishing messages. (c) Smishing messages extracted from images or screenshots available on the web on different sources like Pinterest, google photos, and so on. It includes 56 smishing messages. Text is extracted from these images using a python script.	The training data set consists of 507195 unique entries. The data contains two columns URL and its label. The label is categorized into two i.e., good or bad. (a) Good: It means the URL is safe to use. It does not contain any malicious stuff that will harm. (b) Bad: It indicates that the URL is not safe to use and hence a phishing site. Phishing URLs comprise 28% of the total data set and rest 72% of nonphishing URLs.

Further, in URL phishing classifier, URLs are fed as inputs to the Python script, and all the essential features are extracted and stored in text files. We have used Python Pandas library to access content from csv files and then check the language of URL as well as label the data as ham or spam. Further, we have done the data cleaning and preprocessing: for stemming we used snowball stemmer library of Python. Countvectorizer is used for vectorization. MNB, BNB, and LR machine learning models used to train our classifier. Finally, we have used the voting classifier in Python to merge these two classifiers.

5 | RESULTS AND OBSERVATIONS

5.1 | Accuracy of count vectorizer and TF-IDF vectorizer on the unscaled data set

Table 2 presents the accuracy obtained from all the models when each model is passed through the TF-IDF vectorized data set. The column "Accuracy" in the table represents the accuracy we get when we vectorized the training data set using a count vectorizer. "Accuracy_X" column represents the accuracy obtained from TF-IDF vectorized data set when maximum features are set to *X*. we represent vectorized datasets using count vectorizer and TF-IDF vectorizer with maximum 1000, 2000, 3000, and 4000 features. In Table 3, we can observe that TF-IDF vectorizer with 3000 maximum features is giving the best result.

TABLE 2 Accuracy of TF-IDF vectorized datasets with different models

Algorithm	Accuracy	Accuracy_1000	Accuracy_2000	Accuracy_3000	Accuracy_4000
XGB	0.959923664	0.961832061	0.958015267	0.960877863	0.960877863
GBDT	0.967557252	0.967557252	0.967557252	0.958969466	0.96278626
RF	0.961832061	0.972328244	0.969465649	0.96278626	0.963740458
BgC	0.97519084	0.958969466	0.96278626	0.964694656	0.963740458
KNN	0.938931298	0.944656489	0.942748092	0.938931298	0.936068702
ETC	0.959923664	0.961832061	0.961832061	0.963740458	0.960877863
DT	0.958015267	0.952290076	0.959923664	0.958969466	0.958969466
LR	0.96851145	0.960877863	0.961832061	0.960877863	0.957061069
AdaBoost	0.958969466	0.966603053	0.964694656	0.960877863	0.961832061
BNB	0.949427481	0.945610687	0.948473282	0.947519084	0.947519084
MNB	0.941793893	0.95610687	0.953244275	0.954198473	0.958015267
SVC	0.916030534	0.969465649	0.964694656	0.963740458	0.96278626
GNB	0.874045802	0.799618321	0.860687023	0.872137405	0.873091603

Abbreviations: AdaBoost, Adaptive Boosting; BgC, Bagging Classifier; BNB, Bernouli Naïve Bayes; DT, Decision Tree; ETC, Extra Tree Classifier; GBDT, Gradient Boosted Decision Tree; GNB, Gaussian Naïve Bayes; KNN, K-Nearest Neighbor; LR, Logistic Regression; MNB, Multinominal Naïve Bayes; RF, Random Forest; SVC, Support Vector Classifier; XGB, Extreme Gradient Boosting.

TABLE 3 Accuracy of TF-IDF vectorized data sets having a number of characters and URL as a feature

Algorithm	Accuracy_3000_n_char	Precision_3000_n_char	Accuracy_3000_url	Precision_3000_url
XGB	0.96278626	0.927272727	0.961832061	0.96
GBDT	0.960877863	0.868852459	0.961832061	0.942307692
RF	0.965648855	0.890625	0.963740458	0.913793103
BgC	0.964694656	0.788235294	0.963740458	0.824324324
KN	0.921755725	0.529411765	0.952290076	0.891304348
ETC	0.963740458	0.9	0.966603053	0.947368421
DT	0.958015267	0.862068966	0.958969466	0.852459016
LR	0.960877863	0.868852459	0.952290076	0.810344828
AdaBoost	0.964694656	0.826666667	0.969465649	0.855263158
BNB	0.947519084	0.656565657	0.950381679	0.67
MNB	0.923664122	1	0.961832061	0.838235294
SVC	0.917938931	0.41	0.964694656	0.788235294
GNB	0.877862595	0.36875	0.872137405	0.351851852

Abbreviations: AdaBoost, Adaptive Boosting; BgC, Bagging Classifier; BNB, Bernouli Naïve Bayes; DT, Decision Tree; ETC, Extra Tree Classifier; GBDT, Gradient Boosted Decision Tree; GNB, Gaussian Naïve Bayes; KNN, K-Nearest Neighbor; LR, Logistic Regression; MNB, Multinominal Naïve Bayes; RF, Random Forest; SVC, Support Vector Classifier; XGB, Extreme Gradient Boosting.

5.2 | Accuracy of TF-IDF vectorizer with additional feature URL and nchar

Table 3 represents the accuracy and precision of TF-IDF vectorized data sets with maximum features set to 3000 with an additional feature of the number of characters and presence of URL in the message. From Figure 10, it has been concluded that the number of characters has a maximum dependency on target, that is, 0.34. The number of words and number of sentences has less influence on the target. As the number of characters and number of words has a very high dependency (0.96), so we will select the number of characters as a feature. Here Accuracy_3000_n_char and Precision_3000_n_chars, represent the accuracy and precision obtained when TF-IDF vectorized data set with a maximum of 3000 features with number of characters in the message is passed to all the listed models.

5.3 | Accuracy of TF-IDF vectorizer on the scaled and oversampled data set

Table 4 presents the accuracy obtained when TF-IDF vectorized data set with scaling and oversampling is passed to different models. Here, we have selected maximum features as 2500, 3000, and 3500 as the TF-IDF vectorizer is giving the best result with 3000 features. Here we can observe that scaling does not much improve accuracy. To get the best working model, we have to find the best model for each set of features extracted from vectorized data sets.

From Table 5 it is clear that oversampling data set is giving maximum accuracy, that is, over 98% which is best among all the features extracted sets and models.

Figure 14 represents the Accuracy and Precision of oversampled TF-IDF vectorized data set with a maximum of 3000 features, it can be observed that KNN, Random Forest, and Extra Trees Classifier is giving the best result. As these three models perform very well, they act as selection criteria for the voting classifier to obtain the result.

Figure 15 represents the best accuracy and precision model for oversampled TF-IDF vectorized set for a maximum of 3000 features. These three models are giving the best accuracy and precision among all other models used in a different set of features extracted sets.

5.4 | Results for URL phishing classifier

The model is based on logistic regression as it is giving the best combination of accuracy and precision. The messages which contain the URL are passed to the URL phishing classifier. Here we are mainly concerned about precision so we chose Logistic regression as our model for the detection of phishing URLs from nonphishing URLs.

Figure 16 presents the accuracy and precision obtained from different models that are used for building URL phishing classifier. Here we can observe that Logistic Regression is giving best combination of accuracy and precision, that is, 97% accuracy and 96% precision.

URL phishing classifier is merged with Voting classifier simply aggregates the result of input models and predict the result. Thus, giving better accuracy and precision.

Figure 17 presents result obtained from the voting classifier when we integrate KNN, RF and ETC. Voting classifier simply merges the findings of each classifier passed into it and predicts the output on the basis of voting. The voting classifier is giving 99% accuracy and

TABLE 4 Accuracy and Precision obtained from TF-IDF vectorized datasets with scaling and oversampling

Algorithm	Accuracy_2500_scaled	Accuracy_3000_scaled	Accuracy_3500_scaled	Accuracy_2500_os	Accuracy_3000_os
XGB	0.960877863	0.960877863	0.961832061	0.959523	0.961598
GBDT	0.959923664	0.958969466	0.963740458	0.964193	0.963155
RF	0.972328244	0.96278626	0.966603053	0.984432	0.987545
BgC	0.966603053	0.964694656	0.964694656	0.96575	0.967307
KN	0.938931298	0.937977099	0.937977099	0.982875	0.98547
ETC	0.958015267	0.963740458	0.961832061	0.985989	0.988064
DT	0.957061069	0.958969466	0.958969466	0.932538	0.925791
LR	0.96278626	0.96278626	0.958969466	0.975091	0.974053
AdaBoost	0.965648855	0.960877863	0.96278626	0.974053	0.969382
BNB	0.947519084	0.947519084	0.949427481	0.973015	0.97561
MNB	0.951335878	0.955152672	0.951335878	0.954333	0.956409
SVC	0.953244275	0.955152672	0.950381679	0.982356	0.982875
GNB	0.865458015	0.868320611	0.870229008	0.928905	0.933576

Abbreviations: AdaBoost, Adaptive Boosting; BgC, Bagging Classifier; BNB, Bernouli Naïve Bayes; DT, Decision Tree; ETC, Extra Tree Classifier; GBDT, Gradient Boosted Decision Tree; GNB, Gaussian Naïve Bayes; KNN, K-Nearest Neighbor; LR, Logistic Regression; MNB, Multinominal Naïve Bayes; RF, Random Forest; SVC, Support Vector Classifier; XGB, Extreme Gradient Boosting.

TABLE 5 Comparison of the proposed model with other proposed systems

Security requirements	S-detector[21]	Rule-based[22]	SmiDCA[24]	Feature-based[25]	Smishing detector[26]	Proposed approach
Classification approach	Naïve Bayes	Decision Tree, RIPPER and PRISM	Random Forest, Decision Tree, AdaBoost and SVM	Neural Network, Naïve Bayes, SVM, Logistic Regression	Naïve Bayes	Voting Classifier (KNN, Random Forest, Extra Tree Classifier)
Data set used	N.A	Almeida spam data set,[23] filtering of the data set is done manually to make a new smishing data set of 5169 messages of which 4807 are ham and 362 are smishing messages	Almeida spam data set[23] contain total of 5574 messages out of which 4827 are ham and rest 747 are spam messages	Almeida spam data set,[23] filtering of the data set is done manually to make a new smishing data set of 5169 messages of which 4807 are ham and 362 are smishing messages	Almeida spam data set,[23] filtering of the data set is done manually to make a new smishing data set of 5858 messages of which 5320 are ham and 538 are smishing messages	Almeida spam data set[23] manually filtered, smishing messages obtained from survey and messages extracted from screenshots uploaded on various sources to make a new smishing data set of 5549 messages of which 413 are smishing messages
Result	N/A	99% True Negative Rate	Model has the given accuracy of 96.40% using Random Forest classifier in BFSA	It has an accuracy of 98.74% and 94.20% true positive rate	It shows an overall accuracy of 96.29%.	The proposed approach is giving 99.03% of accuracy and 98.94% of precision
URL	Yes	Yes	Yes	Yes	Yes	Yes
Imbalanced data handling	No	No	No	No	No	Yes

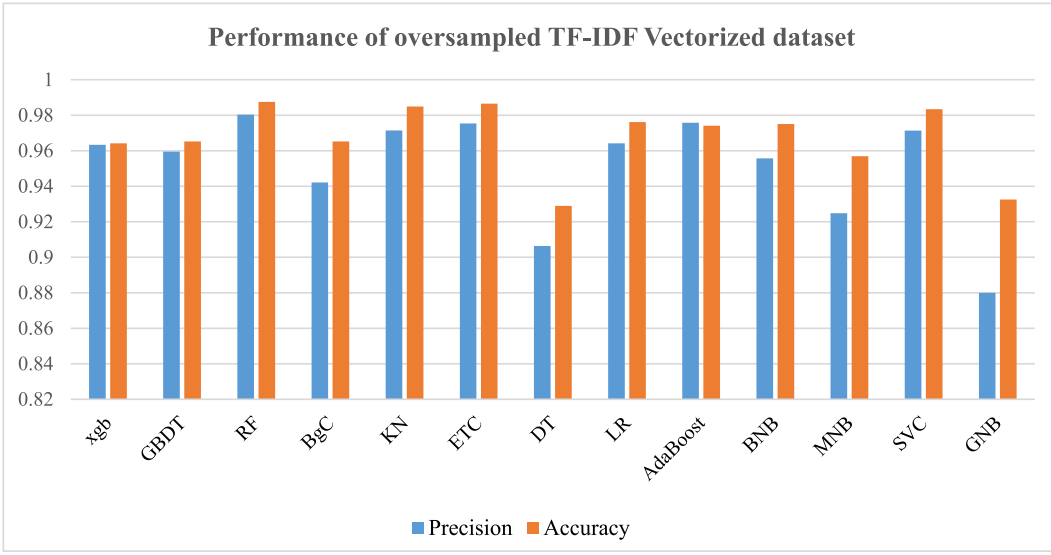

FIGURE 14 Accuracy and Precision of oversampled TF-IDF vectorized data set with maximum 3000 features [Color figure can be viewed at wileyonlinelibrary.com]

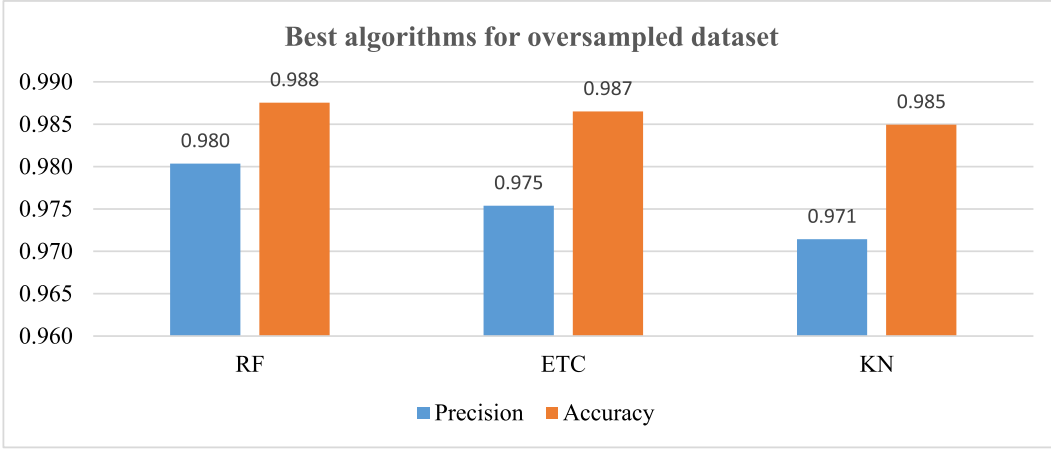

FIGURE 15 Best prediction model for oversampled TF-IDF vectorizer set for maximum 3000 features [Color figure can be viewed at wileyonlinelibrary.com]

98.94% precision. Therefore, our proposed approach is giving 99.03% of accuracy and 98.94% of precision.

5.5 | Comparison with existing smishing detection approaches

Table 5 represents comparison of the proposed model with other proposed systems. The proposed approach is using high-quality data set including the almeida spam data set manually filtered, smishing messages collected from online surveys, and messages extracted from screenshots on internet sources. The model also contains a URL classifier and also handles

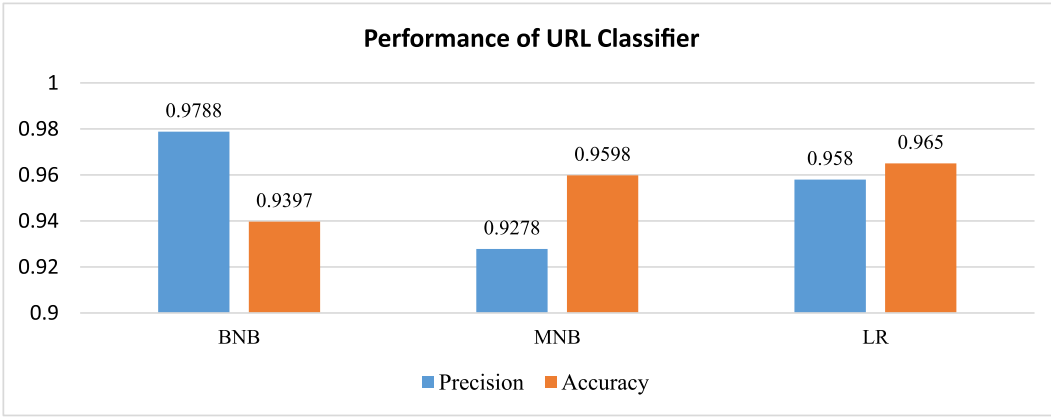

FIGURE 16 Accuracy and Precision obtained from different models for URL phishing classifier [Color figure can be viewed at wileyonlinelibrary.com]

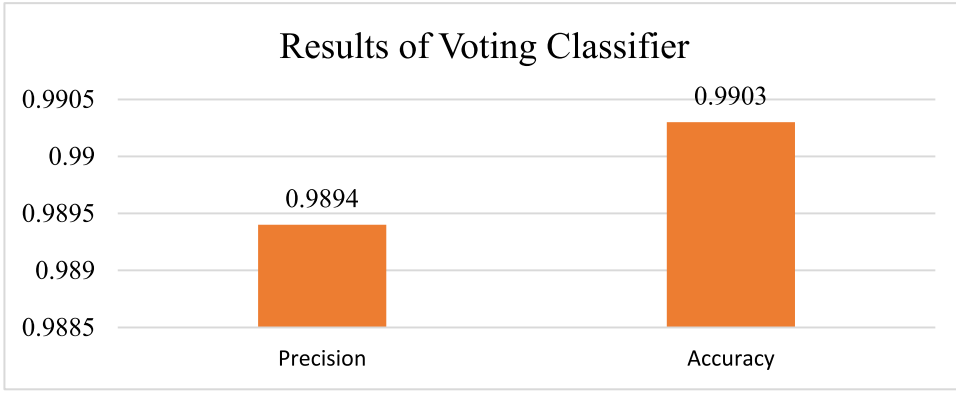

FIGURE 17 Voting classifier integrating KNN, RF, and ETC [Color figure can be viewed at wileyonlinelibrary.com]

imbalanced datasets. Due to all these factors, the proposed approach is giving an accuracy of 99.03% and a precision of 98.94%.

6 | CONCLUSION

In this paper, we have proposed a smishing detector, which is a merged model of text classifier as well as URL classifier. The model scanned the message and if a URL is presented then it will be forwarded to URL classifier. Detection of URL for further processing has increased the accuracy rate of detecting smishing messages. In our model, the balanced data set has been used to reduce the biased results because most of the data set contains less ratio of smishing content. Moreover, we have scaled the data set to improve the working of machine learning algorithms. The proposed approach uses voting classifier to classify the message as smishing or nonsmishing. Voting classifier integrates the result of different models. The proposed approach

is produces 99.03% of detection accuracy and 98.94% of precision rate, which is the best among all the existing models. In future work, we can integrate third-party application to check whether the URL is spam or ham. This process will be the first phase of classification a message as smishing or nonsmishing. As this uses a third-party application so this is very fast phase to classify the message. Moreover, we will make a mobile application that looks for incoming message and warns user if it is smishing or not.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

Data will be made available on the request.

REFERENCES

1. Jain AK, Gupta BB. Detection of phishing attacks in financial and e-banking websites using link and visual similarity relation. *Int J Inform Comput Security*. 2018;10(4):398-417.
2. Gupta BB, Arachchilage NA, Psannis KE. Defending against phishing attacks: taxonomy of methods, current issues and future directions. *Telecommun Syst*. 2018;67(2):247-267.
3. Njuguna D, Kamau J, Kaburu D. Model for mitigating smishing attacks on mobile platforms. 2021 International Conference on Electrical, Computer and Energy Technologies (ICECET). IEEE; 2021:1-6.
4. Mishra S, Soni D. SMS phishing and mitigation approaches. 2019 Twelfth International Conference on Contemporary Computing (IC3); 2019:1-5. doi:10.1109/IC3.2019.8844920
5. Almomani A, Alauthman M, Shatnawi MT, et al. Phishing website detection with semantic features based on machine learning classifiers: a comparative study. *Int J Semantic Web Inform Syst*. 2022;18(1):1-24.
6. Kang A, Dong Lee J, Kang WM, Barolli L, Park JH. Security considerations for smart phone smishing attacks. *Advances in Computer Science and its Applications*. Springer; 2014:467-473.
7. Smishing attacks to watch for in 2022|Technology advice. Accessed April 26, 2022. https://technologyadvice.com/blog/information-technology/smishing-attacks-to-watch-for/
8. Customers warned about scammers posing as HMRC. Accessed April 12, 2022. https://www.gov.uk/government/news/self-assessment-customers-warned-about-scammers-posing-as-hmrc
9. Jain AK, Gupta BB. A survey of phishing attack techniques, defence mechanisms and open research challenges. *Enterprise Inform Syst*. 2022;16(4):527-565.
10. IC3. Internet Crime Report; 2020. Accessed April 12, 2022. https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf
11. Jain AK, Gupta BB. Two-level authentication approach to protect from phishing attacks in real time. *J Ambient Intell Human Comput*. 2018;9(6):1783-1796.
12. State of the phish. Accessed April 12, 2022. https://www.proofpoint.com/sites/default/files/gtd-pfpt-us-tr-state-of-the-phish-2020.pdf
13. Mobile phishing scams. Accessed April 2, 2022. https://www.wandera.com/mobile-phishing-report/
14. Security brief: mobile phishing increases more than 300% as 2020 chaos continues. Accessed April 12, 2022. https://www.proofpoint.com/us/blog/threat-protection/mobile-phishing-increases-more-300-2020-chaos-continues
15. Covid-19 tax scams. Accessed April 8, 2022. https://www.fcc.gov/covid-19-text-scams
16. COVID-19 is creating a boom in internet fraud. Accessed April 10, 2022. https://www.revelock.com/en/blog/covid-19-is-creating-a-boom-in-internet-fraud-but-why
17. Fiorini RA. Computational intelligence from autonomous system to super-smart society and beyond. *Int J Softw Sci Computat Intell*. 2020;12(3):1-13.
18. Dana LP, Salamzadeh A, Hadizadeh M, Heydari G, Shamsoddin S. Urban entrepreneurship and sustainable businesses in smart cities: exploring the role of digital technologies. *Sustain Technol Entrepreneurship*. 2022;1:100016.
19. Most common smishing scams are parcel delivery frauds. https://www.bullguard.com/blog/2021/09/most-common-smishing-scams-are-parcel-delivery-frauds

20. Patel A, Meehan K. Fake news detection on reddit utilising countvectorizer and term frequency-inverse document frequency with logistic regression, multinominalnb and support vector machine. 2021 32nd Irish Signals and Systems Conference (ISSC); 2021:1-6. doi:10.1109/ISSC52156.2021.9467842
21. Joo JW, Moon SY, Singh S, Park JH. S-Detector: an enhanced security model for detecting smishing attack for mobile computing. *Telecommun Syst*. 2017;66(1):29-38.
22. Jain AK, Gupta BB. Rule-based framework for detection of smishing messages in mobile environment. *Proc Comput Sci*. 2018;125:617-623.
23. Almeida TA, Hidalgo JMG, Yamakami A. Contributions to the study of SMS spam filtering: New collection and results. 11th ACM Symposium on Document Engineering; 2011:259-262.
24. Sonowal G, Kuppusamy KS. Smidca: an anti-smishing model with machine learning approach. *Computer J*. 2018;61(8):1143-1157.
25. Jain AK, Gupta BB. Feature based approach for detection of smishing messages in the mobile environment. *J Inform Technol Res*. 2019;12(2):17-35.
26. Mishra S, Soni D. Smishing detector: a security model to detect smishing through SMS content analysis and URL behavior analysis. *Future Generation Comput Syst*. 2020;108:803-815.
27. Mishra S, Soni D. Implementation of 'smishing detector': an efficient model for smishing detection using neural network. *SN Comput Sci*. 2022;3(3):1-13.
28. Chui KT, Gupta BB, Alhalabi W, Alzahrani FS. An MRI scans-based alzheimer's disease detection via convolutional neural network and transfer learning. *Diagnostics*. 2022;12(7):1531.
29. Ali AM, Shamsuddin SM, Eassa FE, et al. Towards an intelligent framework for cloud service discovery. *Int J Cloud Appl Comput*. 2021;11(3):33-57. doi:10.4018/IJCAC.2021070103
30. Ye H, Liu J, Wang W, Li P, Li T, Li J. Secure and efficient outsourcing differential privacy data release scheme in cyber–physical system. *Future Generation Comput Syst*. 2020;108:1314-1323.
31. Sathiyamoorthi V, Suresh P, Jayapandian N, Kanmani P, Janakiraman S. An intelligent web caching system for improving the performance of a web-based information retrieval system. *Int J Seman Web Inform Syst*. 2020;16(4):26-44.
32. Castillo-Zúñiga I, Luna-Rosas FJ, Rodríguez-Martínez LC, Muñoz-Arteaga J, López-Veyna JI, Rodríguez-Díaz MA. Internet data analysis methodology for cyberterrorism vocabulary detection, combining techniques of big data analytics, NLP and semantic web. *Int J Seman Web Inform Syst*. 2020;16(1):69-86.
33. Chen Y, Luo F, Li T, Xiang T, Liu Z, Li J. A training-integrity privacy-preserving federated learning scheme with trusted execution environment. *Inf Sci*. 2020;522:69-79.
34. Gupta BB, Yadav K, Razzak I, Psannis K, Castiglione A, Chang X. A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Comput Commun*. 2021;175:47-57.
35. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. *J Artif Intell Res*. 2002;16:321-357.
36. Gupta BB, Jain AK. Phishing attack detection using a search engine and heuristics-based technique. *J Inform Technol Res*. 2020;13(2):94-109.
37. Li T, Li J, Chen X, Liu Z, Lou W, Hou YT. NPMML: a framework for non-interactive privacy-preserving multi-party machine learning. *IEEE Trans Depend Secure Comput*. 2020;18(6):2969-2982.
38. Wang X, Li J, Kuang X, Tan YA, Li J. The security of machine learning in an adversarial setting: a survey. *J Parallel Distrib Comput*. 2019;130:12-23.

How to cite this article: Jain AK, Gupta BB, Kaur K, Bhutani P, Alhalabi W, Almomani A. A Content and URL analysis based efficient approach to detect smishing SMS in intelligent systems. *Int J Intell Syst*. 2022;1-25. doi:10.1002/int.23035