Proceedings of the Third International Conference on Electronics Communication and Aerospace Technology [ICECA 2019]
IEEE Conference Record # 45616; IEEE Xplore ISBN: 978-1-7281-0167-5

A Machine Learning Approach for Detecting Malicious Websites using URL Features

Akshay Sushena Manjeri, Kaushik R, Ajay MNV, Priyanka C. Nair
Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
Email: amanjeri627@gmail.com, rkaushik1997@gmail.com, mnvajay@gmail.com, v_priyanka@blr.amrita.edu

***Abstract*— With the advancement in technology, the internet has become a platform for wide range of illegal activities ranging from spam advertising to financial fraud. Some of these activities are carried out by embedding malware programs in the URLs. Blacklisting services classify URLs but, the constant creation of newer websites poses a challenge. To overcome this challenge Machine Learning approach is used to classify URLs as malicious or benign. The URL dataset, after addressing the issue of class imbalance is fed into several classification models built using a plethora of classification algorithms. Further, feature selection technique is incorporated to reduce the number of features required for classification and used to rank them based on their importance. Also, rule mining algorithms such as Apriori, FP-Growth and Decision Tree Rules is used to generate IF-THEN rules which helps to establish relationship among the features.**

Keywords*— *URL, malicious, Blacklisting services, classification, Apriori, FP-Growth.

I. Introduction

The internet is intended for gaining knowledge and staying connected, but there exist some negative elements which can disrupt this harmony. One such element is malicious programs. Malicious programs are pieces of code that are written with the intention of gaining access to a machine and disrupting its performance. There are many ways by which malicious programs can infiltrate into a machine. [1] focuses on a common way in which malicious programs gain system access i.e. by visiting malicious websites. Many a times users are not aware if the website they are visiting is genuine or bogus as they simply enter the URL of the website and in some cases, once they click enter, malicious code embedded enters the user's system and starts executing itself. Users are unaware that malicious programs have entered their system and started execution and as the systems performance starts deteriorating, they suspect presence of malicious programs but, by then the damage is already done. Identification of these URLs become necessary as they may harm systems or store passwords without user's information. The idea behind using Machine Learning techniques is to automate the process of classifying URLs. Human intervention is removed and overall performance is improved.

This paper, proposes a method to classify an URL as either malicious or benign by handling class imbalance. Feature selection was implemented and based on which features were ranked. Association rule mining to generate rules which establish the relationship among different features was performed. The paper is divided into 5 sections. Section I is Introduction followed by Related Work in Section II which highlights the work done by others in this area of research. Section III explains in detail the working of the proposed system where all the algorithms used are explained in detail. Section IV and V describes the results obtained followed by conclusion and future scope.

II. Related Work

Traditionally, malicious URLs were detected using blacklisting services and plugin or APIs like [2] and [3] which use statistical approaches like TF-IDF, URL weighting systems or manual collection of malicious URLs. The approach and analysis of Blacklisting services is provided in [4]. With the advent of Machine Learning, heuristic based approach for URL detection became more popular. One of the earliest work in this regard was [5] wherein URL lexical features are used in comparison to URL packet features like TTL, Country of Origin and Server - which is used in this paper. Through the course of time, more features were used for URL classification. Identification of phishing websites from their URLs is proposed in [6] whereas, this paper focuses on a broader base of malicious websites. Different features that can be used to efficiently detect URLs is explained in detail by [7]. The features extracted are subjected to various Machine Learning algorithms. Yahoo-Phish tank dataset was used by [8] and considered both lexical features as well as host-based features. The performance of different classifiers using these features is evaluated. [9] proposes an approach which uses the RIPPER [10] algorithm and also created a user interface for the admin who gives URL as input and

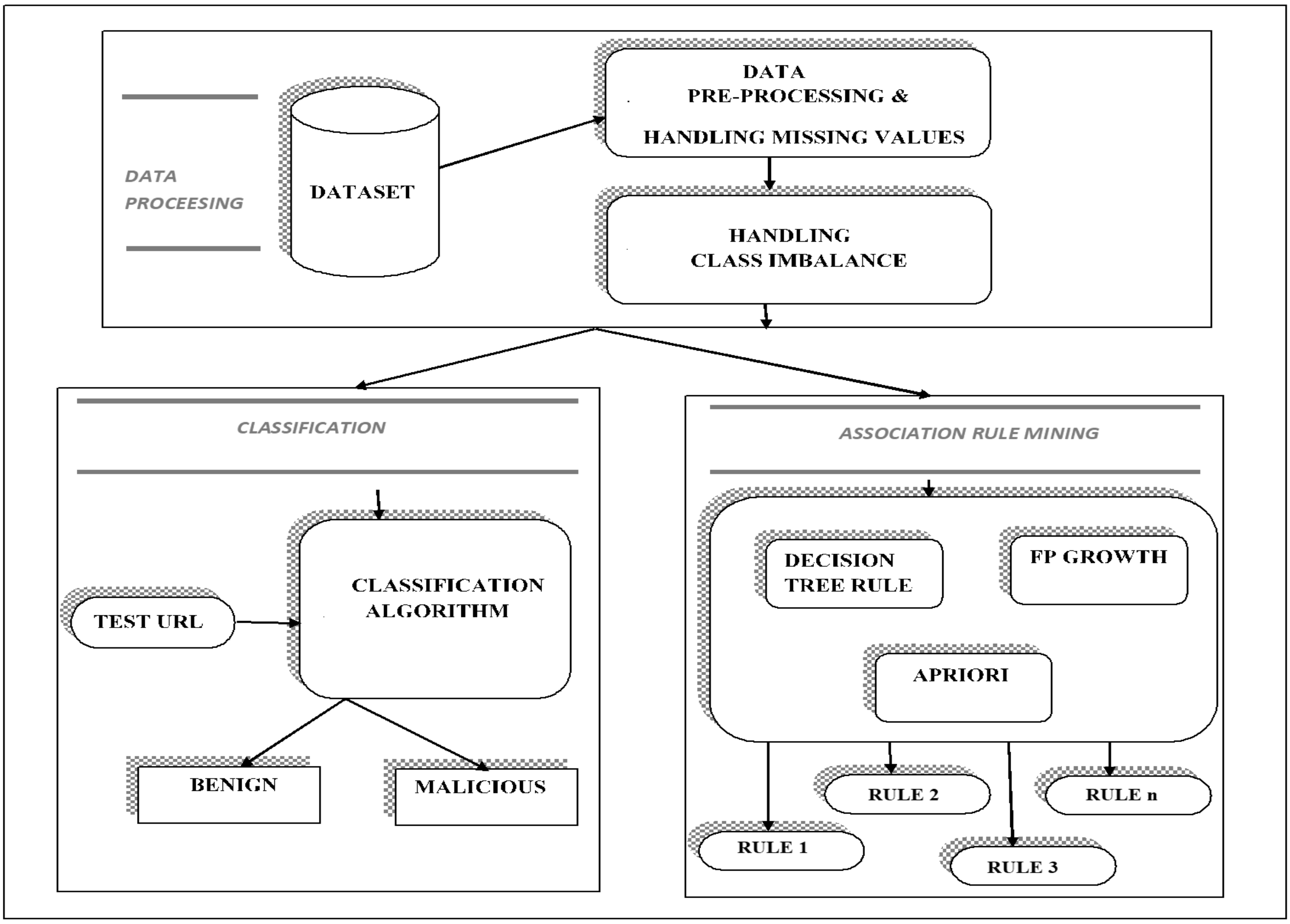

Fig. 1. Proposed Architecture were data is first pre-processed and directed into two separate phases namely classification of URLs and association rule mining

receives output as either fraud or genuine. If the URL is fraud, a notification is sent to the user. In this paper, the approach used for URL classification is most closely related to [11] where a wide range of Machine Learning algorithms (for URL detection) such as SVM, KNN, Random Forest, Decision Tree and Naïve Bayes are used. It is found that Random Forest achieves the highest accuracy. Similar algorithms are incorporated in this paper, but a different dataset is used which consists of newer URLs and different features. In the case of Association Rule Mining, the most direct comparison to this paper comes from [12] whose work includes analyzing phishing URLs and subjecting them to associative rule mining algorithms like Apriori and predictive Apriori algorithms. Although similar in motivation, this paper includes additional algorithms like FP- Growth and Decision Tree Rule making. Moreover, the work proposed in this paper is focused on detecting malicious URLs in comparison to the identification of phishing URLs [12].

With the ever-increasing number of websites, it becomes necessary to include newer URLs to the dataset and incorporate the latest state-of-the-art algorithms. Classifying the website when the dataset had imbalance was not looked into. Ranking the dataset based on their importance was not explored in detail. Rule Mining to find the relations among attributes was not reviewed. This paper focuses on these issues which were not looked into. Feature ranking techniques have been explored along with the addressing of imbalance in the dataset. Feature selection techniques are also discussed to choose features that contribute significantly for the URL to be malicious. This paper talks about a more modern approach for distinguishing between malicious and benign URLs and to find co-occurring factors to identify malicious websites using various Association Rule Mining Algorithms

III. PROPOSED SYSTEM

The acquired dataset has to undergo pre-processing to remove any missing values (if it occurs) and existing values have to be normalized. Class imbalance if present in the dataset has to be handled through various techniques. Then the data has to be subjected to Classification and Association Rule Mining to accurately detect malicious web-sites.

TABLE I. DATASET DESCRIPTION

Feature Name	Type	Description
URL	-- --	A unique Id to identify website
URL Length	Numeric	Count of characters in the URL
Number Of Special Characters	Numeric	Count of special characters recognized like @, $, %.......
Charset	Categorical	Character Encoding Standard
Server	Categorical	Server's Operating System (got from the packet)
Content Length	Numeric	Content size from HTTP Header
Country	Categorical	Country of origin
State	Categorical	State of origin
Registration Date	Categorical	Date on which website was registered
Update Date	Categorical	Date on which website was updated
TCP Conversation Exchange	Numeric	Count of TCP packets exchanged
Dist Remote TCP Port	Numeric	Count of distinct ports detected and different from TCP
Remote IP's	Numeric	Total count of IP's connected
App Bytes	Numeric	Number of bytes exchanged
Remote App Packets	Numeric	Packets received from server
Source App Packets	Numeric	Packets transferred from client and server
App Packets	Numeric	Count of IP packets generated
Source App Bytes	Numeric	Bytes transferred from client and server
Remote App Bytes	Numeric	Bytes received from server
DNS Query Times	Numeric	Count of DNS packets generated
Type	Categorical	Class of websites

A. *Dataset*

Dataset used in this paper is a work done by people as part of their assignment which can be used to detect class of website [13]. The dataset consists of a list of Malicious and Benign URLs. The data was obtained by a process that included different sources of benign and malicious URL, which was verified and used in a low interactive client honeypot in order to get network traffic. Furthermore, some tools were used to get more information, such as the server country with Whois. This stored dataset consists of a total of 1781 URLs accompanied with 20 features. The dataset is divided into 1565 Benign URLs and 216 Malicious URLs.

Initially the dataset contains missing values and also has class imbalance. The description for the features is provided in Table I.

B. *Pre-Processing*

Data consists of missing values and can result in deriving wrong conclusions. To overcome this, handling missing values becomes a crucial step during pre-processing. Another issue that needs to be addressed is range of data values. Some-times, the data consists of values which are spread over a huge range and need more computation. It becomes important to normalize this data and rescale the values within a particular range. Missing values are removed using Replacing with Mode technique i.e. replacing with the most frequently occurring item.

$$Z = (X - X_{\min}) / (X_{\max} - X_{\min}) \tag{1}$$

C. *Class Imbalance*

An issue in Data Mining when the samples of one class (minority) is far less than the samples of the other class (majority). Machine Learning algorithms works at its best when the samples of both are class are approximately equal. When one class is largely greater than the other this problem arises. Dataset used has class imbalance as only 216 samples out of 1781 samples are malicious. Over Sampling was used to achieve the same number of samples of both the class.

1) *SMOTE*

Synthetic Minority Over-Sampling Technique (SMOTE) [14] is used to oversample the data. It synthesizes new minority instances between existing (real) minority instances. SMOTE draws lines between existing minority instances and then thinks from the perspective of these instances and synthesizes new instances at some distance from them towards one of their neighbours.

D. *Classification*

The first algorithm to be used was Random Forest [11]. It is a supervised learning algorithm which forms several decision trees and combines them together to get a better accuracy and clear prediction. It makes use of the same hyper parameters as a decision tree or a bagging classifier. Splitting of a node is done by considering only a subset of features. Trees can be more random, by using random thresholds additionally for each feature rather than searching for the best possible threshold. The next approach is using Decision Trees [11]. It is a decision support methodology which makes use of a tree-like model of decisions and their possible results, including utility, resource costs, and chance event outcomes. The attributes based on which the tree splits into edges are called as condition/internal node. The end of the branch that does not split anymore is the decision/leaf. In our case, whether the URL is Benign or Malicious will be represented in the leaf node.

$$Entropy = \sum_{i=0}^{n} - P_i \log_2 P_i \tag{2}$$

$$Gain(T,X) = Entropy\,(T) - Entropy(T,X) \tag{3}$$

The third algorithm is Logistics Regression [15]. It is an approach to follow when the class types are binary. It is a predictive analysis which is used to report data and to understand the relation between one binary variable and other category variables. Another approach known as Binary Logistic Regression is to classify samples as class variables if it contains only two responses.

$$p = \frac{1}{1+e^{-(b_0+b_1x_1+b_2x_2+\ldots+b_px_p)}} \tag{4}$$

The fourth algorithm is K Nearest Neighbors (KNN) [11]. It follows a non-parametric, lazy learning approach. The motive of it is to use a database in which the sample points are split into many clusters through which it can predict the class of a new test sample. The model does not make any assumptions of the data distributions. It does not use the training samples for any generalizations or form observations from those samples. All the training of samples are done during the testing time. The last algorithm to be used is Support Vector Machines (SVM) which is a discriminative classifier usually defined by a separating hyperplane [16]. It yields an optimal hyperplane which identifies new samples as it is a supervised approach. In 2D space the line dividing a plane in two parts where in each class lay in either side is none other than the hyperplane generated. When used together with random forest or any other machine learning algorithm it provides a very different angle to ensembled model.

E. Feature Selection

This paper uses a feature selection named Recursive Feature Elimination[17] a method that uses a trained model and discards the frail feature(s) that has little impact on classification until the specified number of features are reached. Features are ranked by the model's coefficient or feature importance attributes by recursively eliminating a small number of features per loop. This process is applied until all the features are exhausted. Features are ranked when they are eliminated. Stability of RFE depends on the type of model used for feature ranking.

F. Association Rule Mining

Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. A typical example of using association rule mining for analysis is [18]. Since the URL dataset may consist of numerical attributes having many different values, it becomes difficult to accommodate all the values. Numerical attributes pose many challenges to the generation of rules. There have been many approaches to perform associative rule mining on numerical attributes like [19] which is based on sorting the numeric values and dividing them into groups which can act as nominal values. A bucketing approach is proposed by [20] where numerical attributes of dataset is divided into buckets which optimizes the previous algorithm.

The first algorithm under this section is Apriori. In this approach each part of the dataset is considered and scores are assigned or contrasts it with other data sets in any other ordered way. The scores assigned are used to generate sets that are labelled as frequent appearances in a larger database. Apriori uses a bottom up approach. It uses Breadth First and a hash table to access items from the database. It may be used in conjunction with other algorithms to sort and contrast data. The algorithm was proposed by [21]. The next algorithm explored is FP-Growth. It is a data mining method used for frequent item set mining. It basically tries to identify what item may go with some other item in the dataset. It uses a bottom up approach. FP Growth [22] uses Depth First method to access items of a dataset. It avoids explicit candidate generation of data items. The last approach for generating rules is Decision Tree Rule Making [11]. Here rules are generated sequentially by these algorithms by looking for the best rule that covers a subset of the samples. It looks for rules with best accuracy and not the ones with maximum coverage. Samples covered by the best rule generated by these algorithms are eliminated. This goes on until all the rules that cover the dataset are generated. The output is independent of rules that can be ordered by accuracy.

IV. EXPERIMENTAL RESULTS

A. Classification Results

Initially the dataset had undergone pre-processing to normalize the data and missing values were also handled. The dataset consisted of 816 samples which had missing values and it was replaced using Mode. Min-Max method (1) was used to normalize on App Bytes, Source App Packets, Remote App Packets, Source App Bytes, Remote App Packets and DNS Query Times. Among the 21 features shown in Table I, only 18 features (excluding URL, Registration Date and Update Date) was used for classification and results obtained are shown in Table II. The bar-graph shown in Fig. 2. indicates the presence of class imbalance in the dataset because the difference between the number of benign URLs and number of malicious URLs is large. Various combinations of class balancing was evaluated using SMOTE and equal class option was chosen. After handling class imbalance, the performance of the algorithms using those 18 features is shown in Table III. There is an improvement in the performance of all the algorithms (except SVM) after removing class imbalance.

Feature selection technique was used and the performance of different algorithms was evaluated. The number of was gradually decreased to 8 and performance was evaluated. The results of various algorithms using

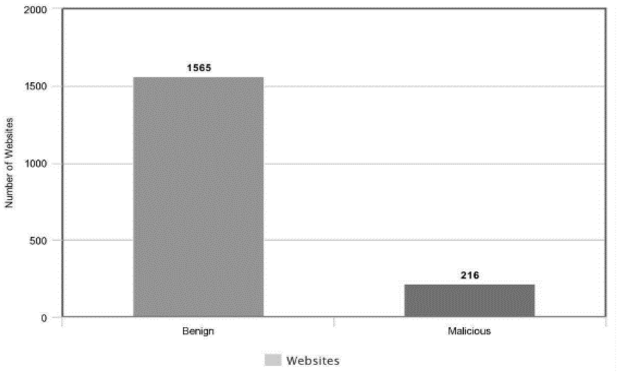

Fig. 2. Bar Graph describing the Imbalance present in the dataset

features ranging from 12 to 8 using Feature Selection Algorithm is shown in Fig. 3. Further decrease in feature lead to a decrease in performance of algorithms and thus it was found that 8 features were enough to efficiently classify the URL. Feature ranking was performed and results are shown in Table IV.

The top 8 features in Table IV are the 8 features that were finally selected. Some features like *URL Length* and *Number Of Special Characters* can easily classify URLs on their own (i.e. longer URL length means malicious URL and more special characters also means malicious URL). The results obtained are evaluated and suitable justification is provided.

After decreasing the number of features to 8, the performance of the algorithms did not deteriorate drastically. Random Forest gives an accuracy of 94% using only 8 features as shown in Fig. 3.

TABLE II. PERFORMANCE WITH CLASS IMBALANCE

Algorithm	Accuracy	Precision	Recall
Random Forest	90.34	94.22	94.82
Decision Tree	83.94	94.56	86.70
Logistic Regression	90.67	92.78	96.93
KNN	91.29	94.39	95.78
SVM	90.34	92.75	96.54

TABLE III. PERFORMANCE WITHOUT CLASS IMBALANCE

Algorithm	Accuracy	Precision	Recall
Random Forest	96.58	98.34	94.76
Decision Tree	92.97	96.15	89.52
Logistic Regression	82.65	84.44	80.06
KNN	93.19	97.74	88.43
SVM	82.68	84.03	80.70

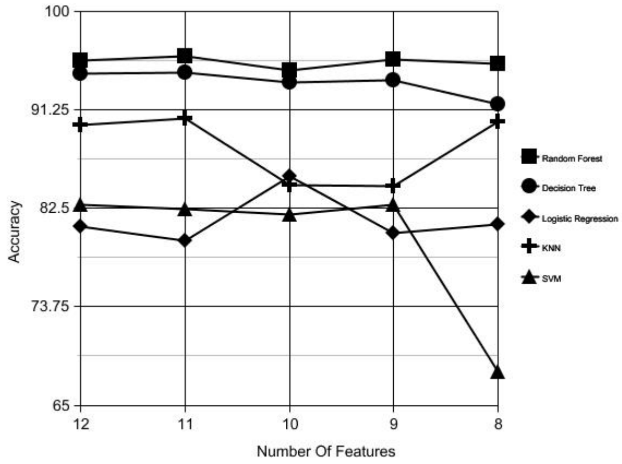

Fig. 3. Performance of algorithms with decrease in the number of features

TABLE IV. FEATURE RANKING

Rank	Feature	Rank	Feature
1	Source App Bytes	10	Source App Packets
2	App Bytes	11	TCP Conversation Exchange
3	State	12	App Packets
4	Number Of Special Characters	13	Remote IP's
5	URL Length	14	Server
6	Dist Remote TCP Port	15	DNS Query Times
7	Remote App Packets	16	Charset
8	Remote App Bytes	17	Content Length
9	Country		

The dataset was provided as input to association rule mining algorithms like Apriori, FP-Growth and Decision tree rule making. Both confidence and support was set to 95%.

B. Association Rule Mining Results

The proposed work uses a similar approach to [18] wherein the average of each numerical attribute is found. This average is set as a threshold value for generating rules. If average of a numerical attribute for negative category is X and positive category is Y rules have been generated based on X and Y values. These values are used as threshold to convert the numerical values to binary.

For categorical values the rules are generated considering frequency of occurrence of labels. The highest occurring label for a negative (positive) category attribute is considered to be the condition value for the attribute to belong to a negative (positive) category. If the highest occurring value for a categorical attribute (which belongs to negative category) is Z. If the value is Z for a particular sample Z is changed to 1 or else it is made 0. Once the rules are general the binary values are reset to the threshold values.

Using this approach rules are generated.

1) Apriori Rules

Rule 1 - IF *Dist Remote TCP Port* > *11* THEN *MALICIOUS*
Rule 2 - IF *Remote App Packets* > *15 THEN Source App Packets* < *7600*
Rule 3 - IF *App Packets* > *15* THEN *MALICIOUS*
Rule 4 - IF *Remote App Bytes* < *1400* THEN *MALICIOUS*
Rule 5 - IF *Dist Remote TCP Port* > *11* AND *Source App Bytes* < 7600 THEN *Source App Packets* > *15*

The rules obtained suggests that when a particular feature(s) have a particular value there is a higher probability that the website is malicious. For example *Rule 2* suggests that if number of IP packets generated is above 15 then the website could be malicious as proposed by Apriori. This is also verified by the fact that malicious URLs lead to downloading unsafe programs that tend to be big files that require more number of *App Packets.*

This was found in the case of URL "http://rozga.fileave.com/e8034335afb724d8fe043166ba57cd23.exe" which is an *.exe* file sent over multiple *App Packets.*

2) FP - Growth Rules

Rule 6 - IF *Source App Packets* > *15* THEN *MALICIOUS*

Rule 7 - IF *Source App Packets* > *15* THEN *Source App Bytes* < *7600*

Rule 8 - IF *Source App Packets* > *15* AND *Source App Bytes* < *7600* THEN *MALICIOUS*

Rule 9 - IF *Remote App Packets* > *15* AND *Source App Packets* > *15* THEN *MALICIOUS*

Rule 10 - IF *Remote App Bytes* < *1400* THEN *Dist Remote TCP Port* > *11*

Similarly for the rule "IF *Source App Packets* > *15* AND *Source App Bytes* < *7600* THEN *MALICIOUS*" when the packets transferred from client to server which is Source App Packets and the data transferred from client to server which is Source App Bytes are greater than 15 and less than 7600 respectively then the website could be malicious which was suggested by FP Growth algorithm. The rest of them follow in the similar way.

3) Decision Tree Rules

Rule 11 - IF *Dist Remote TCP Port* = *0* AND *Remote IP's* = *2* AND *Number Of Special Characters* = *10* THEN *MALICIOUS*

Rule 12 - IF *Server* = *MICROSOFT* AND *Charset* = *UTF-8* THEN *MALICIOUS*

Rule 13 - IF *DNS Query Times* = *2 AND Dist Remote TCP Port* = *0* AND *Server* = *NGINX* THEN *MALICIOUS*

The *Rule 11* is verified by the fact that malicious URLs contain *special characters* which have hidden meanings. If a link has a lot of *special characters* then it is very likely to be malicious. Also malicious URLs tend to re-route the packets, hence they may have more than 1 *Remote IP*. This can be seen with case of URL "http://ad.9tv.co.il/serv4/www/delivery/ajs.php?zoneid%5=37&cb =54350%405237&charset=utf-8" which has a lot of special characters and requires multiple IP re-routing.

V. Conclusion and Future Scope

This paper proposes a model which has been trained using a stored dataset containing close to 1750 URLs and applying different machine learning algorithms ranging from Logistics Regression to Support Vector Machines. After iterations of training and testing, it is found that Random Forest produces the highest accuracy. Random Forest produces an accuracy of 96%. The performance comparison between different algorithms is shown in Table II, Table III and Fig 3. Now, the model can efficiently classify an URL as either benign or malicious. Feature Selection Techniques applied yielded good results as malicious URLs were accurately detected using only 8 features. These features are the top 8 ranked features in Table IV.

The model was also able to produce relationship rules between URL features by subjecting the features to association rule mining. The Association Rule Mining was performed by using algorithms like Apriori, FP Growth and Decision Tree Rule Making. Rules generated by these algorithms were identified and can be used to predict malicious URLs. This paper can pave way for a plug-in to be developed for web browser. In this way the user can be warned about malicious website URLs in real time while browsing the internet. The application can be user friendly and protect the user's system from any threat or attack by notifying the user.

ACKNOWLEDGMENT

We would like to extend our gratitude to the entire staff of Computer Science Department, Amrita School of Engineering - Bengaluru for their support.

REFERENCES

[1] Yoshitha and Sampath, N., “Implementation of data integrity and regenerating data using erasure code”, International Journal of Applied Engineering Research, vol. 10, pp. 37469-37472, 2015

[2] Google Safe Browsing API - Google Code, http://code.google.com/apis/safebrowsing/, accessed on June 12, 2010.

[3] SmartScreen Filter – Microsoft Windows http://windows.microsoft.com/en-US/internetexplorer/products/ie-9/features/smartscreen-filter, 2011.

[4] Luong Anh Tuan Nguyen, Ba Lam To, Huu Khuong Nguyen, Minh Hoang Nguyen. Detecting phishing web sites: A heuristic URL-based approach. 2013 International Conference on Advanced Technologies for Communications (ATC 2013)

[5] Steve Sheng, Brad Wardman, Gary Warner, Lorrie Faith Cranor, Jason Hong, Chengshan Zhang. An Empirical Analysis of Phishing Blacklists. CEAS 2009 - Sixth Conference on Email and Anti-Spam July 16-17, 2009, Mountain View, California USA

[6] Min-Yen Kan and Hoang Oanh Nguyen Thi. Fast Webpage Classification Using URL Features. CIKM’05, October 31-November 5, 2005, Bremen, Germany. ACM 1-59593-140-6/05/0010.

[7] Jin-Lee Lee, Dong-Hyun Kim, Chang-Hoon, Lee. Heuristic-based Approach for Phishing Site Detection Using URL Features. Proc. of the Third Intl. Conf. on Advances in Computing, Electronics and Electrical Technology - CEET 2015

[8] Justin Ma, Lawrence K. Saul, Stefan Savage, Geoffrey M. Voelker. Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs. Department of Computer Science and Engineering University of California, San Diego

[9] Neha Sangal Urvashi Prajapati and Deepti Patole. Fraud Website Detection using Data Mining. International Journal of Computer Applications 141(3):40-44, May 2016

[10] RIPPER William Cohen, Fast Effective Rule Induction, Proceedings of the 12th International Conference on Machine Learning

[11] F. Vanhoenshoven, G. Nápoles, R. Falcon, K. Vanhoof and M. Köppen, "Detecting malicious URLs using machine learning techniques," 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8

[12] S. Carolin Jeeva and Elijah Blessing Rajsingh. Intelligent phishing url detection using association rule mining. Human-centric Computing and Information Sciences, 2016 https://doi.org/10.1186/s13673-016-0064-3

[13] Christian Urcuqui, Andrés Navarro, José Osorio, Melisa García, "Malicious and Benign Websites", IEEE Dataport, 2017.

[14] Shastri Sunil Suresh, Priyanka C. Nair, Deepa Gupta, Ravi C. Nayar, Raghavendra Rao, Amritanshu Ram, Breast Cancer Diagnosis and Prognosis Using Machine Learning Techniques." The International Symposium on Intelligent Systems Technologies and Applica-tions. Springer, Cham, 2017"

[15] McCullagh, Peter, and John A. Nelder. *Generalized linear models*. Vol. 37. CRC press, 1989.

[16] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. Machine Learning 20.3 (1995): 273-297

[17] X. Zeng, Y. Chen, C. Tao and D. v. Alphen, "Feature Selection Using Recursive Feature Elimination for Handwritten Digit Recognition," *2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Kyoto, 2009, pp. 1205-1208.

[18] S. Khare and Dr. Deepa Gupta, “Association rule analysis in cardiovascular disease”, in 2016 Second International Conference on Cognitive Computing and Information Processing (CCIP), 2016

[19] G. Piatetsky-Shapiro, Discovery, analysis, and presentation of strong rules, in “Knowledge Discovery in Databases” pp. 229248, 1991

[20] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita and Takeshi Tokuyama. Mining Optimized Association Rules for Numeric Attributes. Journal of Computer and System Sciences 58, 112 (1999)

[21] Agrawal R, Imielinski T and Swami A (1993) Mining association rules between sets of items in large databases. ACM-SIGMOD 22:207–216

[22] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD’00, Dallas, TX). ACM Press, New York, NY, USA 2000