Traitement du Signal
Vol. 37, No. 4, August, 2020, pp. 563-569
Journal homepage: http://iieta.org/journals/ts

Phishing Website Detection Using Machine Learning Classifiers Optimized by Feature Selection

Dželila Mehanović*, Jasmin Kevrić

Department of Information Technologies, International Burch University, Sarajevo 71000, Bosnia and Herzegovina

Corresponding Author Email: dzelila.mehanovic@ibu.edu.ba

https://doi.org/10.18280/ts.370403

Received: 13 May 2020
Accepted: 10 August 2020

Keywords:
classification, Decision Tree, feature selection, K-Nearest Neighbors, phishing website detection, Random Forest

ABSTRACT

Security is one of the most actual topics in the online world. Lists of security threats are constantly updated. One of those threats are phishing websites. In this work, we address the problem of phishing websites classification. Three classifiers were used: K-Nearest Neighbor, Decision Tree and Random Forest with the feature selection methods from Weka. Achieved accuracy was 100% and number of features was decreased to seven. Moreover, when we decreased the number of features, we decreased time to build models too. Time for Random Forest was decreased from the initial 2.88s and 3.05s for percentage split and 10-fold cross validation to 0.02s and 0.16s respectively.

1. INTRODUCTION

The Internet is widely used among people and it has become an inseparable part of our life. Therefore, huge amounts of data are exchanged. Those users could be more or less experienced using the web. But, nevertheless, nobody is safe from the huge threat that is available there outside. Those threats are phishing websites that are hard to differentiate from the original ones. These websites are used to collect personal and confidential user data that usually should be protected. Later, information is misused and people are experiencing consequences. Some of the consequences could be identity loss or financial debts.

Statistics for 2019 states that 15% of those who were successfully attacked will be attacked at least one more time within a year. Number of phishing attacks increased by 65% in respect to 2018 and around 1.5 million of phishing websites were created each month [1]. Almost one third of all data breaches in 2017 were due to phishing attacks. Approximately 55% of phishing websites in 2019 used SSL certificates [2]. Research also shows that 33% of people closed their business after a phishing attack [3].

The problem with phishing attacks is not only that they are increasing, but also, they are improving and becoming more sophisticated. Due to that, it is necessary to develop systems that will help in detection of these phishing websites to prevent negative outcomes. Therefore, in this work we want to develop an intelligent system that will be used to detect phishing websites. We are going to use machine learning algorithms for classification such as K-Nearest Neighbor (KNN), Decision Tree and Random Forest (RF).

The rest of the work is organized as follows: in section two, we are giving overview of several works related to the phishing websites detection. Section three and four provide details about the database and methodologies, respectively. Results are presented in section five. We conclude our work with section six.

2. LITERATURE REVIEW

Web phishing is a serious security threat that is present in the Internet. Sensitive financial and personal information are taken from the users thanks to the phishing websites. These websites look like legitimate websites and they are used to gather private data. Therefore, phishing attacks use weaknesses of the user and it is hard to reduce those, but it is important to work on detection techniques improvement. In this section we present several works related to detection of phishing websites.

Salihovic et al. [4] applied Artificial Neural Networks, Logistic Regression, Random Forest, Support Vector Machine, k-Nearest Neighbor and Naive Bayes on UCIs phishing websites dataset. In the first experiment they used the original dataset which had 31 attributes. Therefore, in the second experiment, authors applied feature selection using BestFirst+CfsSubsEvaluation and Ranker+Principal Components feature selection optimizers. In the first experiment, Random Forest achieved the highest accuracy equal to 97.33%. First optimizer reduced the number of attributes in the phishing dataset to 10 and accuracy was decreased by 1.53% on average. Second optimizer, Ranker reduced the phishing dataset for only one attribute. Accuracy has been increased for Random Forest and Support Vector Machine, while it was decreased by 0.09% for other algorithms. Also, they applied the same method with spam emails dataset and Random Forest had the best performance. So, they proved that it is possible to use the same algorithm for both datasets.

Yi et al. [5] proposes a method which detects phishing websites. Detection model is based on a deep belief network (DBN). Two types of features are used: original and interaction features. Original features are those directly related to the websites, while interactive features include features related to the interaction between websites such as in-degree and out-degree of URL. To test DBN real IP flows data are used. Dataset includes traffic flow for 40 minutes and 24 hours.

Features like IP address, access time, URL, request page source, user agent and user cookie are extracted. Information about each node is collected and connected to the graph. Contrastive Divergence algorithm is used as a training algorithm. In the final experiment there are three parameters that were changed to find the best combination. Those are the number of layers, number of iterations per layer and number of hidden layers. Approximate true positive rate is approximately 90%. The highest detection rate is achieved with two layers, as the number of layers increases, accuracy decreases. Optimal number of iterations is 250 and the number of hidden layers is between 20 and 40.

Mahajan and Siddavatam [6] present a method for improvement of phishing websites detection. Dataset contains URLs of legitimate and phishing websites. Legitimate URLs are collected from www.alexa.com and phishing URLs are collected from www.phishtank.com. Python program is used to extract features from these URLs. Extracted features are presence of IP address in the URL, presence of @ in the URL, number of dots in hostname, prefix or suffix separated by - to domain, URL redirection, HTTPS token in URL, length of host name, number of slash in URL, presence of unicode in URL, age of SSL certificate, URL of anchor, iframe and website rank. Applied algorithms are decision trees, random forests and support vector machines. Dataset is divided into training and testing dataset in ratio 50:50, 70:30 and 90:10. For implementation of the experiment, the authors used the Scikit-learn tool. The highest accuracy 97.14% is achieved using Random Forest. Also, accuracy increased by increasing the number of instances in the training dataset. For the future work, authors are planning to implement hybrid solutions which will combine machine learning algorithms and blacklist methods.

Ali [7] used a wrapper features selection method to detect phishing websites. Author used a dataset from UCI Machine Learning Repository which has 4898 phishing websites which is 44% and 6157 legitimate websites which is 56% of all data. There are 30 features that are recognized as key features and they are grouped in address bar-based features, abnormal based features, HTML and Javascript based features and domain-based features. As features selection method, author used wrapper features selection method which finds the best set of features for given machine learning classifier. Classifiers as Naive Bayes, Support Vector Machine, C4.5, k Nearest Neighbor and Random Forest are tested before and after features selection. For the feature selection, the author used wrapper methods, Principal Component Analysis and Information Gain methods. The 5 cross validation technique was used and results from the experiments were compared. The highest true positive and true negative rates are achieved when using wrapper features selection method. On the other side, Principal Components Analysis resulted in the worst performance. Highest true value rates are achieved by Random Forest 97.3% and k Nearest Neighbor 97.1%. True negative values 97% are achieved by these two algorithms too.

Kevric et al. [8] combined NBTree, C4.5 and Random Forest to build an effective classifier for network intrusion detection. NSL-KDD dataset with 41 features was used. Random Tree outperformed other individual algorithms with accuracy 88.46% and Random Forest and NBTree achieved highest accuracy 89.24% when applied together.

Random Forest, C4.5, REP Tree, Decision Stump, Hoeffding Tree, Rotation Forest and MLP are applied in the study [9] to compare results for phishing websites classification. Authors used phishing websites dataset available at UCI Machine Learning Repository. Rotation Forest with REP Tree had highest accuracy 89.1% before feature selection. After feature selection was applied using Correlation Attribute Evaluation, 12 attributes were selected and tests for all algorithms are applied again. Accuracy of MLP increased from 85.5% to 89% which was the best for reduced dataset, while accuracy of Rotation Forest decreased to 87.1%.

3. DATABASE

Table 1. Features in phishing websites database

Name	Description
ID	a unique id for the website
alexa rank	for websites parsed from the top 1000 of alexa.com this is the rank of the websites, otherwise null
isPhish	is this webpage url from a phishing list (1) or non-fraudulent (0)
Parent	what is the parent website for this website (for phishes this contains the verified original website) otherwise null
Parent Count	a counter how many parents have been found for this website
url	the url that was originally provided for the scan
urlHash	an md5 hash of this url for quicker finding of identical urls
urlBasedomain	the base domain of this url (this usually means the top-level domain plus the domain part in front of it e.g. "google.com" for some special domain names this may include some more e.g. "google.co.uk")
final Url	the url that was finally parsed after following all redirect requests
finalUrlBasedomain	same as urlBaseDomain for finalUrl
name	a name identifier that is sometimes assigned to websites (otherwise null)
quality	reserved for future use (always null)
scanned	a UNIX-Timestamp when this url was visited
rescan	used internally for rescanning already scanned website (always 0)
statusCode	the statusCode that was returned for this website. Only websites with 200 were manually checked and assigned to states
htmlContent	the HTML content of the page that was loaded under finalUrl
loadTime	the time it took querying the content and taking all screenshots
phishTank_XYZ	fields parsed from the phishtank.com list of fraudulent websites
state	*the state of the phishing website (explained in Table 2)*
duplicatedForm	In some case websites have been assigned more quickly by assigning them as duplicated to other phishing websites. This was stored here. This field does NOT denote that those websites are all duplicated that can be found for that website neither does it denote that the websites look 100% similar
created	UNIX-Timestamp when this website was created and stored in the Table (usually prior to scanned)

Table 2. State of the website

Value	Short name	Message
1	Disabled by Hoster	This website has been disabled by the hoster or it redirects to a site of the hoster that is clearly no phishing website
2	Phishonly	The website is a phishing website that tries to steal user data but it has not real parent. This can be websites just asking for an arbitrary email-address and password or websites that use multiple logos and cannot be assigned to one specific parent.
3	Phishing Website	This is a real phishing website with a parent that has been assigned.
5	Still Loading/No content	It seems that this website has been captured while still loading or at least it does not yet contain any sensemaking content. Mostly those are completely white websites.
6	Dead Link	Although the initial website could load (because the statusCode showed 200) no proper website was loaded in the end. Either the browser did not reach any page at all, or the page reached can be distinguished as a 404 or similar error page. This could have been because of a meta-reload-tag pointing to an illegal location.
7	Original Back	The Original Website seems to be back online. A server was hacked and the phishing website was placed on this usually non-fraudulent domain. But now the original website is back and has been captured instead of the original phish.
8	Weird Content/Weird language	The content of the site is not the original content, neither dead, nor a phishing attack. In this category there are also pages that are written in foreign languages.
9	Coding Error	A script error occurred. Part of the website code was executed but threw an error. Mostly the website just shows some PHP-Warnings and no real content.
10	No Image	The capturing engine did not capture a proper image for the website content. This error occurred for some websites that produced an error when rendering on the screenshot canvas of Firefox. It is possible that another screenshot type may still contain website content.
11	Domainparking	The website shows a series of links and is disabled or parked.

For the purpose of this research we used a phishing websites database available at the link [10]. Database contains 11 215 records and 21 features. From the total number of samples there are 1 185 non-fraudulent, while 10 030 of them are categorized as phishing websites. Description of 21 features is provided in Table 1.

Attributes quality which is always null and rescan which is always 0 are removed at the beginning together with created and scan attributes. Table 2 lists all possible values for the state attribute described in Table 1.

4. METHODOLOGY

4.1 Feature selection

To select features, we used the Weka tool and its algorithms for feature selection. Applied algorithms are: Gain Ratio Attribute Evaluator (GainRatioAttributeEval), Info Gain Ratio Attribute Evaluator (InfoGainAttributeEval), One R Attribute Evaluator (OneRAttributeEval), Relief Attribute Evaluator (ReliefAttributeEval) and Symmetric Uncertainty Attribute Evaluator (SymmetricUncertAttributeEval). For all of these algorithms we used Ranker search method.

Gain Ratio Attribute Evaluator [11] calculates value of feature by calculating gain ratio of feature with respect to the class. Gain ratio is calculated by the following equation:

$$\text{GainR (Class, Feature)} = (\text{H(Class)}-\text{H(Class} \mid \text{Feature)}) / \text{H(Feature)} \quad (1)$$

Info Gain Ratio Attribute Evaluator [12] calculates value of feature by calculating info gain ratio of the feature with respect to the class. Info gain ratio is calculated by the following equation:

$$\text{InfoGain(Class, Feature)} = \text{H(Class} \mid \text{Feature)} \quad (2)$$

One R Attribute Evaluator [13] finds the value of features by performing OneR classifier. This classifier makes a rule for each predictor and selects the one rule which has the smallest error. Rule is made by counting the number of occurrences of each class for attribute and assigning the attribute to the class with the highest occurrence. Then, error is calculated for each attribute. Attribute with the smallest error rate is selected [14].

Relief Attribute Evaluator [15] calculates the value of features such that performs sampling of the instance and compares the value of that instance with that feature value for the nearest instance. Weight vector of instance is calculated by the Eq. (3) where nearHit is the nearest instances that has the same class as given instance and nearMiss is nearest instance with different class.

$$W_i=W_i-(x_i-\text{nearHit}_i)^2+(x_i-\text{nearMiss}_i)^2 \quad (3)$$

Symmetric Uncertainty Attribute Evaluator [16] calculates value of feature by calculating symmetrical uncertainty of the feature with respect to the class. Symmetrical uncertainty is calculated by the following equation:

$$\text{SymmU(Class, Feature)} = 2 * (\text{H(Class)} - \text{H(Class} \mid \text{Feature)}) / \text{H(Class)} + \text{H(Feature)} \quad (4)$$

The H(Class) and H(Feature) denote entropy [17] of the class and feature respectively. The H(Class | Feature) is conditional entropy of class with respect to the given feature.

4.2 Machine learning algorithms

Machine learning becomes popular in different areas. It presents the use of algorithms to build models which will make predictions based on input data which is called training data without need to explicitly program solutions for the task [18, 19].

Classification is supervised machine learning technique, used to determine to which group of output labels new observation belongs to Ref. [20]. Model is trained using part of the entire data set which is called a training set. Later, performance of the model is tested using the remaining part of the data set which is called test set. In the testing phase, the model should be able to discover what is the output label for the provided input data.

To perform phishing websites detection, in this work we applied K-Nearest Neighbor (KNN), Decision Tree and Random Forest classifiers. Below, an overview for each of them is provided.

KNN is an algorithm that could be used for both regression and classification, but mostly it is used for classification problems [21]. Algorithm does not make any assumptions about data and it is simple to interpret, but on the other side it requires a lot of memory and prediction could be slow [22]. The Figure 1 shows an example of classification using KNN. We have two possible classes red and green and we need to decide to which class the new yellow instance belongs. Distance between new instance and neighbors is calculated. Based on the majority of nearest neighbor prediction has been made.

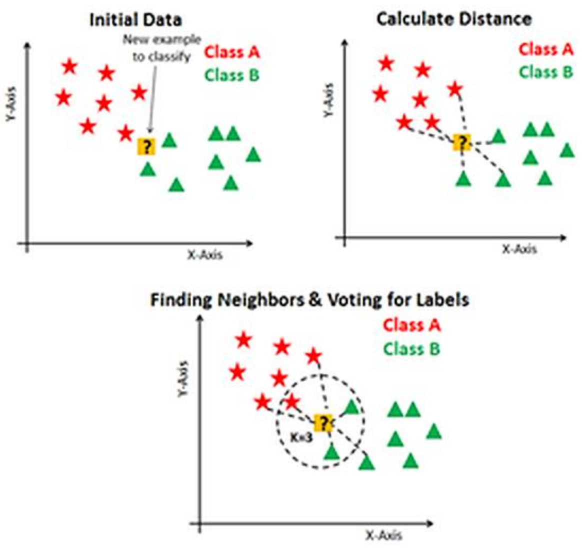

Figure 1. K-Nearest Neighbor [23]

Decision tree is a tree -like model used for classification. It is built using nodes, branches and leaves. Each branch represents decisions made depending on the value of the attributes. Leaves represent class labels (possible outcome classes) [22, 24]. Sometimes trees can be too long due to large numbers of features in the data set.

This makes the tree complex and can lead to the problem of overfitting. In this situation, we can set maximum depth which is the longest path from the root to a leaf. Also, we can set a minimum number of inputs for each leaf. One more term faced while creating decision trees is pruning. It means to remove branches with low priority features. Decision trees can be used with both numerical and categorical data [25]. Figure 2 below shows an example of a decision tree that is built to make decisions about job acceptance. Features that are taken as beneficial for decision are salary, time to commute and whether there is free coffee or not. At the bottom of each branch is a leaf that represents decisions which could be accepted or declined.

Random forests in classification algorithms create multiple trees and combine their results to obtain final prediction. It overcomes the overfitting problem by selecting multiple random subsets of features and uses those subsets as data inputs for different trees. Random forest provides good implementation of feature importance calculation [27]. Also, this algorithm is a very fast and the final decision is made using majority voting.

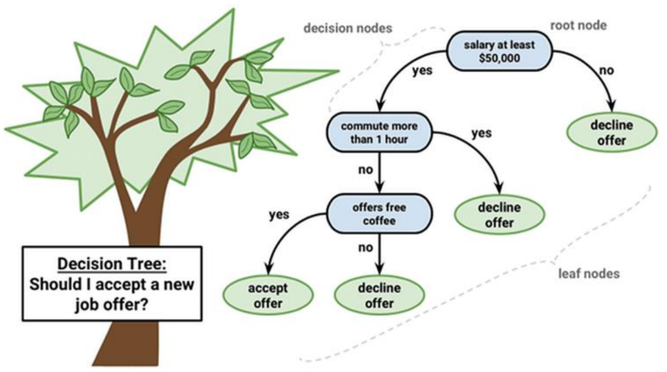

Figure 2. Decision tree [26]

Figure 3 shows a random forest classifier. The first step is to select random subset and create trees from those subsets. After that, prediction is made for each tree separately and voting is performed. Prediction with the highest number of votes is selected as final decision.

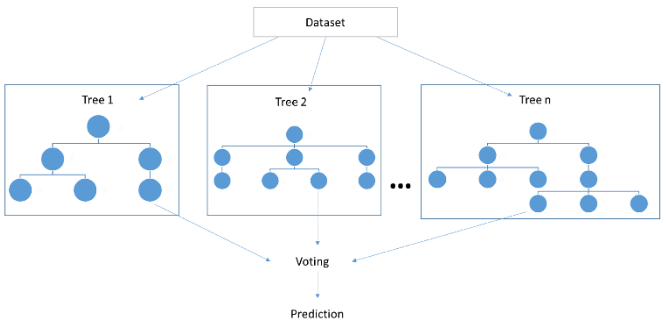

Figure 3. Random forest [28]

In order to decide the maximum number of trees one can run an algorithm with several values to analyse performance. In the obtained results, it should be possible to detect the point at which performance is in the sleep mode or starts decreasing.

Furthermore, when analyzing results, it should be detected what is tree depth when overfitting starts, that is the value at which the tree performs well on training data, but poor on test data.

To find maximum number of the features one can tune the number of features and investigate results to detect point when overfitting starts. To select features, random forest calculates the importance of each feature, that is the amount for which accuracy decreases when the feature is removed. The higher decrease means higher importance of the feature.

5. RESULTS

In this section we present results obtained by the feature selection and machine learning methods described in the previous section. Several feature selection methods were applied, and their results were compared to find the attributes with highest impact to the result. Furthermore, classification algorithms, such as KNN, Decision Tree and Random Forest were applied to initial and reduced dataset.

Table 3. Feature selection filters and selected features

Filter type	Selected features
GainRatioAttributeEval	0 phishtank_verified 0 phishtank_isonline 0.4738 parentCount 0.2250 name 0.1939 scanned 0.1567 phishtank_targetname 0.0911 state 0.0423 urlBasedomain 0.0417 finalUrlBasedomain 0.0406 phishtank_verifiedtime 0.0394 phishtank_submissiontime 0.0393 phishtank_id 0.0393 phishtank_detailurl 0.0366 finalUrl 0.0362 url 0.0362 urlHash 0.0274 parent 0.0219 duplicatedFrom
InfoGainAttributeEval	0.4867 phishtank_verified 0.4867 phishtank_verifiedtime 0.4867 phishtank_id 0.4867 phishtank_isonline 0.4867 phishtank_detailurl 0.4867 phishtank_targetname 0.4867 phishtank_submissiontime 0.4863 url 0.4863 urlHash 0.4671 scanned 0.4659 urlBasedomain 0.4540 finalUrl 0.4304 finalUrlBasedomain 0.2395 state 0.1032 duplicatedFrom 0.0658 parent 0.0655 parentCount 0.0534 name
OneRAttributeEval	100 phishtank_isonline 100 phishtank_verified 99.8841 phishtank_targetname 99.8038 scanned 91.3776 parentCount 91.0834 name 89.4338 state 88.4262 parent 87.2581 duplicatedFrom 71.4935 finalUrlBasedomain 69.0771 urlBasedomain 36.7187 phishtank_verifiedtime 29.835 finalUrl 13.0183 phishtank_submissiontime 10.5662 phishtank_id 10.5662 phishtank_detailurl 10.5484 urlHash 10.5484 url
ReliefAttributeEval	1.000000000 phishtank verified 1.000000000 phishtank_isonline 0.896691930 phishtank_targetname 0.722960320 state 0.303174320 duplicatedFrom 0.299786000 urlBasedomain 0.290441373 parent 0.274926437 finalUrlBasedomain 0.128283548 phishtank_verifiedtime 0.107935800 phishtank_submissiontime 0.105662059 phishtank_id 0.105662059 phishtank_detailurl 0.052340615 finalUrl 0.004641927 name 0.000463936 scanned 0.000090257 parentCount 0.000017833 url 0.000017833 urlHash
SymmetricUncertAttributeEval	1 phishtank_verified 1 phishtank_isonline 0.3226 scanned 0.271 phishtank_targetname 0.2096 parentCount 0.1538 state 0.1476 name 0.0811 urlBasedomain 0.0797 finalUrlBasedomain 0.078 phishtank_verifiedtime 0.0758 phishtank_submissiontime 0.0757 phishtank_id 0.0757 phishtank_detailurl 0.0704 finalUrl 0.0698 url 0.0698 urlHash 0.0456 parent 0.0398 duplicatedFrom

Table 4. Comparison of the classification performances with different researches in Phishing Websites Detection

Research	Method	Accuracy (%)
Salihovic et al. [4]	Random Forest	97.33
Yi et al. [5]	Contrastive Divergence	90
Mahajan & Siddavatam, [6]	Random Forest	97.14
Ali [7]	Random Forest	97.3
Kevric et al. [8]	Random Forest and NBTree	89.24
[9]	MLP	89
Our Research	Random Forest	100

Table 3 shows attribute evaluators from Weka that we used and a list of attributes that are selected for each of them. From those lists we can see that some attributes are highly positioned by one method, but for another they are not in the top few attributes. Also, we have attributes that are at the top of the lists produced by majority of filters that were used.

When it comes to classification, firstly, we used all attributes from the feature selection phase. Also, we used two test options: percentage split and 10-fold cross validation. For percentage split we used 66% of data for the training set and 34% of data for the test set. To evaluate results, we measured accuracy and the obtained accuracy for these algorithms was 100%.

Since we had high accuracy, we wanted to check if it is possible to keep the same or similar accuracy, but with less attributes. That is why in the next phase of model building, we used only attributes that are highly positioned by all feature selection methods. Those attributes are parentCount, scanned, phishtank_verified, phishtank_isonline, phishtank_targetname, state and name. Now, after these updates of features, we built models again. Accuracy was the same, 100%, but time needed to build the model was significantly decreased. In the first experiment, Random Forest needed 2.88s and 3.05s to build the model using percentage split and 10-fold cross validation respectively. Later, with modified number of features this time was reduced to the 0.02s and 0.16s. Also, KNN needed 2.64s for the initial model, and 1.14s for the second model.

Table 4 summarizes results of researches present in section two. It is noticeable that Random Forest outperforms other classifiers with high accuracy values. Salihovic et al. [4], Ali [7], Hodžić et al. [9] used a data set from UCI Machine Learning Repository [29] which contains 11 055 samples from which 4898 are phishing websites and 6157 are legitimate. This data set is often used in research related to phishing websites detection. In our research we used a dataset that we did not find that others used in their works, because we wanted to compare results when using different dataset. This dataset contains 11 215 samples from which 1185 are legitimate and 10 030 are phishing websites and it seems to give satisfactory results.

Number of attributes in the dataset [29] is 30, while our dataset contains 21 attributes. After we applied feature selection, we ended with a dataset that contained 7 features obtained combining results of all feature selectors that were applied. Authors in [9] used Correlation Attribute Evaluation and reduced dataset to 12 features. Using the same dataset, Salihovic et al. [4] reported to have 10 features after they applied BestFirst + CfsSubsEvaluation.

Research done in the study [8] was not related to phishing websites detection, but network intrusion detection. They used a dataset with 41 attributes and compared classifiers that were used in other mentioned works. As a result, they proved that Random Forest is the best algorithm to use for this kind of detection which is considered as an important part of the security sphere.

6. CONCLUSIONS

Phishing websites are a serious threat that exists there in the online world. Damage caused by owners of these websites could be huge for users. Because of that, we wanted to see if it is possible to use machine learning classification algorithms to prevent or decrease the number of harm due to these websites. In this work, we applied feature selection methods from the Weka and tested three classification algorithms: KNN, decision tree and RF.

Also, we found a database, which is not often used in similar works and tested if it is suitable for this kind of application. As a result, we achieved accuracy of 100%. This accuracy seems to outperforms results presented in the section two.

In our approach, to find most valuable features we used multiple feature selection filters. The outputs of these filters are analyzed and features that are proposed as most important by majority of the filters are selected to use in the classification phase.

Moreover, we noticed that it is possible to reduce the number of features and keep the same accuracy. Random Forest needed 2.88s and 3.05s before feature selection and 0.02s and 0.16s after feature selection is applied. This is important, because with a decrease in the number of features, we decreased time needed to build a model which is valuable as performance achievement and main contribution of this work.

REFERENCES

[1] Retruster. https://retruster.com, accessed on Mar. 25, 2020.

[2] Crane, C. (2019). 20 Phishing Statistics to Keep You from Getting Hooked in 2019 - Hashed Out by The SSL Store^{TM}. Hashed Out by The SSL Store^{TM}, Jul. 24, 2019. https://www.thesslstore.com/blog/20-phishing-statistics-to-keep-you-from-getting-hooked-in-2019/, accessed on Mar. 25, 2020.

[3] Crane, C. (2020). Phishing Statistics 2020: 15 Phishing Stats to Help You Avoid Getting Reeled In|InfoSec Insights. InfoSec Insights, Jan. 22, 2020. https://sectigostore.com/blog/phishing-statistics-phishing-stats-to-help-avoid-getting-reeled-in/, accessed Mar. 25, 2020.

[4] Salihovic, I., Serdarevic, H., Kevric, J. (2018). The role of feature selection in machine learning for detection of spam and phishing attacks. Advanced Technologies, Systems, and Applications III, 476-483. https://doi.org/10.1007/978-3-030-02577-9_47

[5] Yi, P., Guan, Y., Zou, F., Yao, Y., Wang, W., Zhu, T. (2018). Web phishing detection using a deep learning framework. Wireless Communications and Mobile Computing, 2018: 1-9. https://doi.org/10.1155/2018/4678746

[6] Mahajan, R., Siddavatam, I. (2018). Phishing website detection using machine learning algorithms. International Journal of Computer Applications, 181(23): 45-47. https://doi.org/10.5120/ijca2018918026

[7] Ali, W. (2017). Phishing website detection based on supervised machine learning with wrapper features selection. International Journal of Advanced Computer Science and Applications, 8(9). https://doi.org/10.14569/ijacsa.2017.080910

[8] Kevric, J., Jukic, S., Subasi, A. (2017). An effective combining classifier approach using tree algorithms for network intrusion detection. Neural Computing and Applications, 28(S1): 1051-1058. https://doi.org/10.1007/s00521-016-2418-1

[9] Hodžić, A., Kevrić, J., Karadag, A. (2016). Comparison of machine learning techniques in phishing website classification. International Conference on Economic and Social Studies (ICESoS'16), Apr. 2016, accessed on May 10, 2020.

[10] Phishload. http://www.medien.ifi.lmu.de/team/max.maurer/files/phishload/index.html, accessed on Dec. 22, 2019.

[11] GainRatioAttributeEval, Dec. 20, 2019. http://weka.sourceforge.net/doc.dev/weka/attributeSelection/GainRatioAttributeEval.html, accessed on Mar. 30, 2020.

[12] Class InfoGainAttributeEval. http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html, accessed on Mar. 30, 2020.

[13] Class OneRAttributeEval. http://weka.sourceforge.net/doc.dev/weka/attributeSelection/OneRAttributeEval.html, accessed on Mar. 30, 2020.
[14] Singh, J., Singh, G., Singh, R. (2017). Optimization of sentiment analysis using machine learning classifiers. Human-centric Computing and Information Sciences, 7(1): 32. https://doi.org/10.1186/s13673-017-0116-3
[15] Class ReliefFAttributeEval. http://weka.sourceforge.net/doc.dev/weka/attributeSelection/ReliefFAttributeEval.html, accessed on Mar. 30, 2020.
[16] Class SymmetricalUncertAttributeEval. http://weka.sourceforge.net/doc.dev/weka/attributeSelection/SymmetricalUncertAttributeEval.html, accessed on Mar. 30, 2020.
[17] Basurto-Flores, R., Guzmán-Vargas, L., Velasco, S. Medina, A., Calvo Hernandez, A. (2018). On entropy research analysis: cross-disciplinary knowledge transfer. Scientometrics, 117(1): 123-139. https://doi.org/10.1007/s11192-018-2860-1
[18] Machine learning-Wikipedia. https://en.wikipedia.org/wiki/Machine_learning, accessed Nov. 26, 2018.
[19] Contributors to Wikimedia projects, Machine learning – Wikipedia. Wikimedia Foundation, Inc., May 25, 2003. https://en.wikipedia.org/wiki/Machine_learning, accessed on Mar. 31, 2020.
[20] Ouchtati, S., Chergui, A., Mavromatis, S., Aissa, B., Rafik, D., Sequeira J. (2019). Novel method for brain tumor classification based on use of image entropy and seven Hu's invariant moments. Traitement du Signal, 36(6): 483-491. https://doi.org/10.18280/ts.360602
[21] Srivastava, T. (2018). Introduction to k-Nearest Neighbors: A powerful Machine Learning Algorithm (with implementation in Python & R). Analytics Vidhya, Mar. 26, 2018. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/, accessed on Mar. 31, 2020.
[22] Bronshtein, A. (2017). A Quick Introduction to K-Nearest Neighbors Algorithm. Medium, Apr. 11, 2017. https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7, accessed on Nov. 26, 2018.
[23] Dey, S. (2020). Introduction to k-Nearest Neighbours. Medium, Aug. 26, 2020. https://medium.com/@sdey2658/introduction-to-k-nearest-neighbours-6aa9b86eb876, accessed on Sep. 12, 2020.
[24] Contributors to Wikimedia projects, Decision tree learning – Wikipedia. Wikimedia Foundation, Inc., Apr. 05, 2004. https://en.wikipedia.org/wiki/Decision_tree_learning, accessed on Mar. 31, 2020.
[25] Gupta, P. (2018). Decision Trees in Machine Learning – Towards Data Science. Towards Data Science, May 17, 2017. https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052, accessed on Nov. 28, 2018.
[26] Asiri, S. (2018). Machine Learning Classifiers. Medium, Jun. 11, 2018. https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623, accessed on May 3, 2020.
[27] Donges, N. (2018). The Random Forest Algorithm – Towards Data Science. Towards Data Science, Feb. 22, 2018. https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd, accessed on Dec. 1, 2018.
[28] Kannan, S. (2020). Predicting NBA Rookie Stats with Machine Learning. Medium, Jun. 29, 2019. https://towardsdatascience.com/predicting-nba-rookie-stats-with-machine-learning-28621e49b8a4, accessed on May 3, 2020.
[29] UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/index.php, accessed on May 11, 2020.

NOMENCLATURE

KNN	K-Nearest Neighbor
RF	Random Forest
URL	Uniform Resource Locator
HTML	Hypertext Markup Language
HTTPS	Hypertext Transfer Protocol Secure
SSL	Security Socket Layer