Inteligencia Artificial 25(69), 107-121
doi: 10.4114/intartif.vol25iss69pp107-121

INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Web architecture for URL-based phishing detection based on Random Forest, Classification Trees, and Support Vector Machine

Javier Julio Martin Lamas Piñeiro [1], Lenis R. Wong Portillo [1]

[1]FISI - Facultad de Ingeniería de Sistemas e Informática, Universidad Nacional Mayor de San Marcos, Lima, Perú.

Resumen Nowadays phishing is as serious a problem as any other, but it has intensified a lot in the current coronavirus pandemic, a time when more than ever we all use the Internet even to make payments daily. In this context, tools have been developed to detect phishing, there are quite complex tools in a computational calculation, and they are not so easy to use for any user. Therefore, in this work, we propose a web architecture based on 3 machine learning models to predict whether a web address has phishing or not based mainly on Random Forest, Classification Trees, and Support Vector Machine. Therefore, 3 different models are developed with each of the indicated techniques and 2 models based on the models, which are applied to web addresses previously processed by a feature retrieval module. All this is deployed in an API that is consumed by a Frontend so that any user can use it and choose which type of model he/she wants to predict with. The results reveal that the best performing model when predicting both results is the Classification Trees model obtaining precision and accuracy of 80%.

Resumen. En la actualidad el phishing es un problema tan serio como cualquier otro, pero se ha intensificado bastante en la actual pandemia del coronavirus, un momento en el que más que nunca todos utilizamos internet hasta para realizar pagos cotidianamente. En este contexto se han desarrollado herramientas para detectar phishing, existen herramientas bastante complejas en calculo computacional y que no son de tan sencilla utilización para cualquier usuario. Por ende, en este trabajo proponemos una arquitectura web basada en 3 modelos de aprendizaje automático para predecir si una dirección web tiene phishing o no, basados principalmente en Random Forest, Classification Trees y Support Vector Machine. Por lo tanto, se desarrollan 3 modelos distintos con cada una de las técnicas indicadas y 2 modelos basados en los anteriormente mencionados modelos, los cuales son aplicados a direcciones web previamente procesadas por un módulo de obtención de características. Todo ello se despliega en un API la cual es consumida por un Frontend para que cualquier usuario lo pueda utilizar y escoger con qué tipo de modelo quiere predecir. Los resultados revelan que el modelo que mejor se comporta al momento de predecir ambos resultados es el modelo de Árboles de clasificación obteniendo una precisión y exactitud de 80%.

Keywords: Phishing detection, Random Forest, Support Vector Machine, Machine Learning, Classification Trees.

Palabras Clave: Detección de phishing, Bosque Aleatorio, Máquina de vectores de soporte, aprendizaje automático, árboles de clasificación.

1 Introduction

Phishing has increased by 59% since the pandemic began. And according to the study State of Cyber Risk in Latin America, 49% of Peruvian companies surveyed perceived an increase in cyber-attacks because of the pandemic, and 21% consider that social engineering (phishing) is the cyber-attack that has increased the most [1].

ISSN: 1137-3601 (print), 1988-3064 (on-line)

We must keep in mind that this problem is not something recent, the It Reseller portal reports that in 2019 55% of companies suffered a successful phishing attack 2019 [2]. And that in a single day half a million attacks have been seen as happened on February 14, 2019, as a Microsoft Security Report tells us [3]. And if we go back to the present, we can see that in Venezuela two out of every ten users suffered identity theft or impersonation attempt via email or Internet portals [4] during the beginning of the pandemic.

There are works where a multitude of techniques are used to detect phishing [5] where up to 7 machine learning techniques are used which causes a high computational cost and in the same work, the high effectiveness of the Random Forest technique is highlighted, as in the work of [6] where the fact that Random Forest is the most appropriate technique for the current fields of application such as the issue of URLs is enhanced. To confirm the veracity of what is stated in the work of [7] it is commented that the Random Forest technique has about 95% accuracy and therefore stands out among other techniques.

Currently, all kinds of studies related to the prevention and/or detection of phishing have already been carried out, such as the so-called studies with Blacklists and others with Machine Learning techniques. Among those that apply Machine Learning, several types of research using multiple Machine Learning methods that perform feature decomposition, obtaining URL resources and text processing, as well as the use of dictionaries to recognize common characters in URLs and all this is correct but represents a fairly large computational load and complexity, Moreover, as proven by multiple studies, the Random Forest method is the one that has the highest accuracy in terms of URLs, so comparing 3 methods applied to the same URL or using them simultaneously with some known logical relationship would be quite effective and efficient to have a greater assurance that the prediction is more correct.

In this paper, we focus on query-driven detection of phishing web pages by decomposing URL features and resources by obtaining lexical features, host-based features, and web content features. We use 1 publicly fetched dataset with which 3 different models based on 3 different techniques such as Random Forest, Classification Trees, and Support Vector Machines are trained to predict under the features fetched by a given feature whether the URL is safe or suspected of being phishing. A comparison will be made between the results of the 3 models and conjunction of these with a logical relationship "AND" and "OR" will be made to compare them.

The article is organized as follows. In section 2 we discuss the related work; as for section 3 the contribution that was made is detailed; consequently, section 4 describes how the tests were performed and the results of this are shown in section 5; and finally, section 6 describes the conclusions and future work on this topic.

2 Related jobs

The systematic literature review is carried out based on the methodology proposed by Wong et al. [8], which has 3 phases: (1) planning of the review, (2) development of the review, and (3) results and analysis.

In the Planning phase of the review, the following research questions were posed: Q1: What solutions have been developed to prevent mass phishing, Q2: What techniques have been developed to control phishing, Q3: What factors influence phishing?

The following keywords were defined for the search: "phishing", "phishing chatbot", "phishing URL Random Forest" which were applied to the titles, abstracts, and keywords. The search for articles is performed in the ScienceDirect, IEEEXplore, and ACM databases.

The following inclusion criteria are established: year of publication between 2018 and 2020, type of sources Journals or Proceedings, and those articles that are related to answering at least one of the three research questions mentioned above.

In the development phase of the review, the search for articles is carried out considering the inclusion and exclusion criteria mentioned above, to finally select the primary studies resulting from the systematic literature review process.

In the results phase of the review, because of the review process, a total of 19 studies were selected by the previously defined inclusion criteria.

The articles obtained are analyzed through a taxonomy, which consists of 3 categories: Solutions (Q1), techniques (Q2), and Factors (Q3), each of them related to the research questions previously posed. Table 1 shows the number of articles for each classification.

Table 1. Classification of the studies found in the systematic review of the literature.

Taxonomy	Source	Quantity
Solutions	[9,10,11,12,13,14,15]	7
Techniques	[16,17,18,5,19,7,6]	7
Factors	[20,21,22,23,24]	5

In the first classification of "Solutions" are various software implemented for mass use as the phishing simulator and email aliases [9], mobile browser extension that uses the phishing discovery method [11], the tool that integrates the cosine sine algorithm [12], the framework for evaluating phishing systems [13], the software with a detailed description of IR tools[14] or the chatbot with artificial intelligence that integrates 5 modules [15], the most remarkable being [10] in which a Web Crawler-based phishing attack detector (WC-PAD) is proposed, which proposes a 3-phase phishing attack detection approach: A DNS blacklisting, a heuristics-based approach, and a web crawler-based approach.

In the second classification of "Techniques," there are several techniques such as fuzzy logic as a classifier [16], neural networks based on decision trees [17], web graph to model the link structure [18], convolution neural networks [19], Random Forest [7] and feature extraction [6] where most of them use various Machine Learning techniques, among which stands out the article of [5] where a phishing detection system is proposed by using 7 Machine Learning algorithms such as Decision Tree, Adaboost, K-star, CNN, Random Forest, SMO, and Naive Bayes and different numbers/types of features based on NLP, a product of the test of techniques we could see that the Random Forest technique is the one that has a higher percentage of effectiveness.

In the third classification of "Factors" as a result of the review, a wide variety of factors are identified, such as an overview of the methods available to detect phishing [20], interaction between the target's personality and the principle of persuasion [21], types of users susceptible to phishing [22], change in the susceptibility of users affected by phishing [23], among which highlights the factors identified in the work of [24] where a model is proposed that addresses the impact of information, privacy risks, information security risks, and information security knowledge on the susceptibility to launch phishing attacks through the construction of moderate trust.

3 The logical architecture of Machine Learning models

The proposal presents a responsive web system that allows the user to enter a URL and choose from a variety of 3 models that predict whether the web address is prone to phishing. This web consumes an API in which a feature extraction module has been deployed which processes the URL and then passes to the chosen model to return the prediction result (see Figure 1).

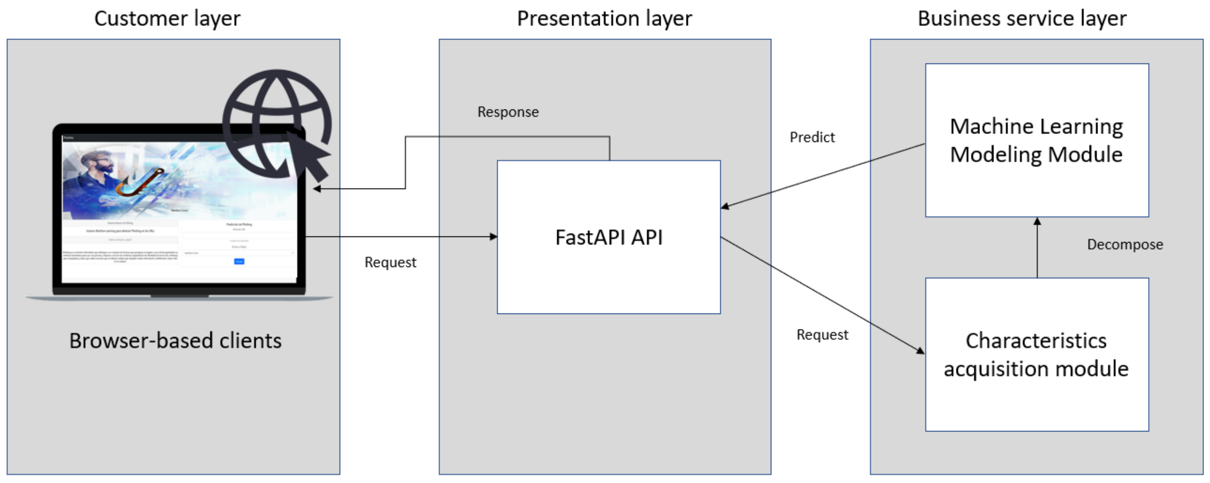


Figure 1. General Architecture of the Proposal.

The core of the proposal is based on synthesizing a method for phishing detection with a low computational cost that compares the effectiveness of 3 models and uses them together, but at the same time is quite accurate. We work with a dataset, which is partitioned for the training of each of the 3 models, performing the tests and finding the ideal accuracy product from which we obtain the ready model. Then, the conjunction models are made thanks to the 3 models already calculated previously. After having the 5 models ready, the URL can be processed in the feature retrieval module, which decomposes the URL into numbers that can be processed by the model that has been chosen, resulting in output as a prediction corresponding to the URL (see Figure 2).

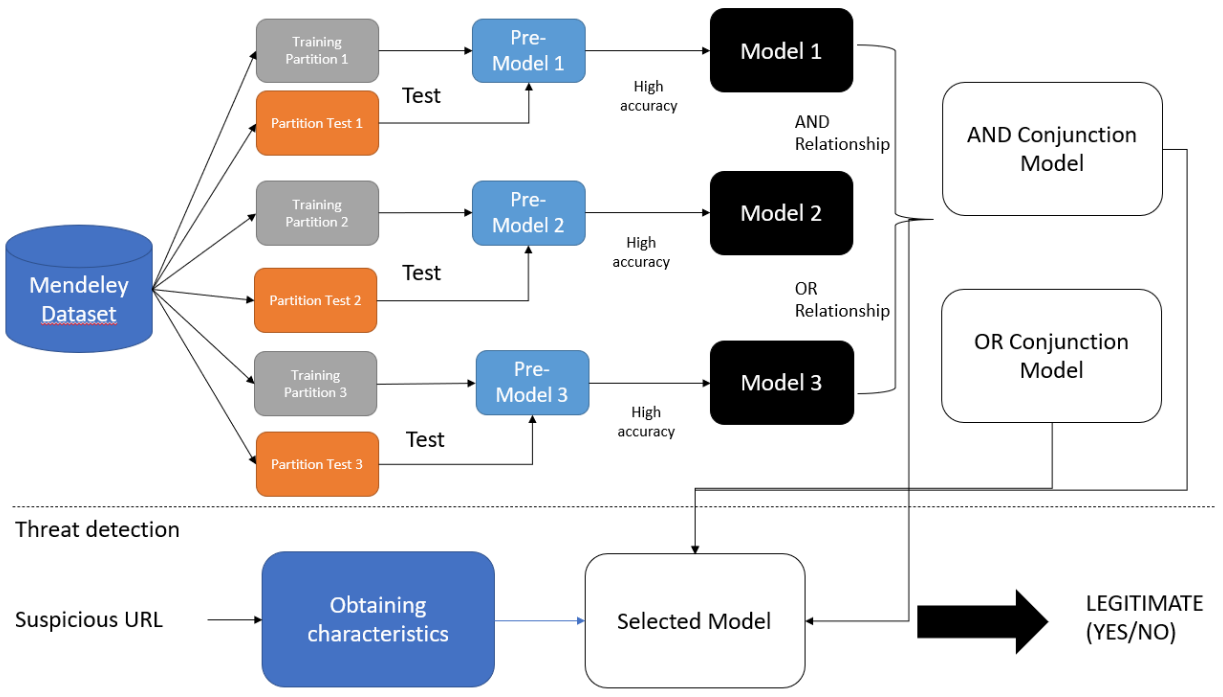


Figure 2. Core architecture of the proposal.

3.1 Obtaining datasets

In this stage, an exhaustive search of processed datasets with a phishing diagnosis is carried out with their corresponding fields, the URL, and the response field if it is phishing or not. Something important to mention is that for reasons of ease and practicality the dataset needed to have the proper documentation to know how to obtain each field to extrapolate it to new URLs not included in the same dataset and make the necessary feature extraction functions. After the long search, the Mendeley Dataset [25] is obtained, which complies with all the required characteristics, in addition to having documentation that explains in detail what has been obtained in each field. Table 2 shows a summary of the most important fields of the chosen Dataset.

Table 2. Description of the datasets to be used.

Field	Description	Type
length_url	Number of characters in the URL	numeric
email_in_url	If there is an email address in the URL	boolean
qty_tld_url	Number of characters of the topmost domain	numeric
domain_length	URL domain length	numeric
domain_in_ip	An IP address exists in the domain of the URL	boolean
tld_present_params	TLDs exist in the parameters	boolean
time_response	URL response time	numeric
url_shortened	This is the short URL	boolean

We obtain a dataset of 88647 records with 112 columns [25] considering the result and 111 columns only obtained by lexical features, host-based features, and web content features. Positive records sum up to 30647 (35%) and negative records sum up to 58000 (65%) which is shown in Figure 3.

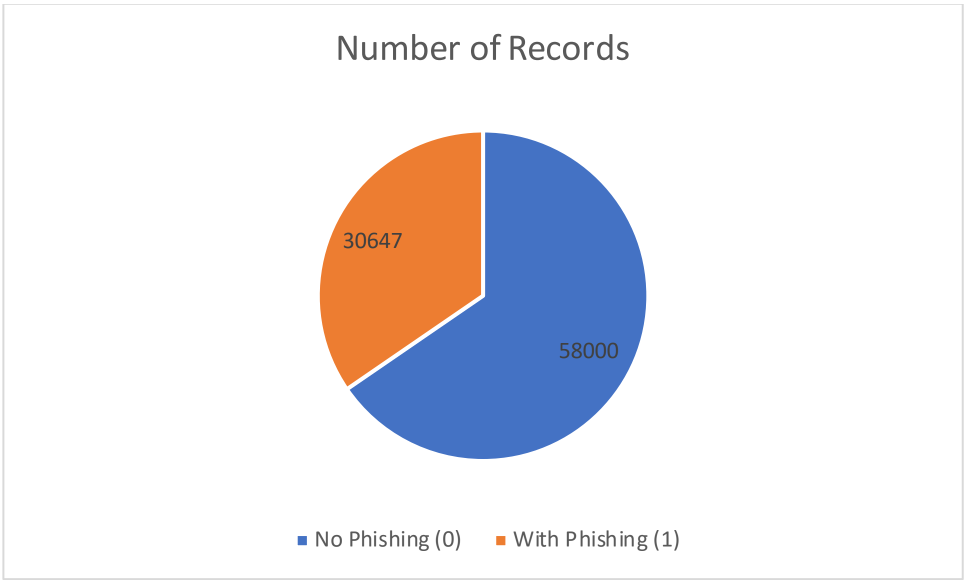


Figure 3. Distribution of positive and negative phishing records in the proposed dataset.

3.2 Analysis of Machine Learning techniques

In this stage, tests are carried out with different Machine Learning techniques with the same dataset, a product of which it is observed that the technique with the highest approval was Random Forest, who is the main one used in this contribution, however, the 3 are used to make the comparison and use them together to enhance that effectiveness. As shown in Table 3.

Table 3. Comparison of accuracies of the different Machine Learning techniques chosen.

Machine Learning Technology	Accuracy of "0"	Accuracy of "1"	Average Accuracy
Support Vectors (gamma='auto')	0.72	0.97	0.75
Classification trees	0.97	0.93	0.95
Random Forest (n_estimators=80)	0.98	0.95	0.97

3.3 Obtaining characteristics

At this stage, we used the Python language and the Mendeley Dataset Phishing documentation [25] to use algorithms to calculate each of the dataset fields when entering a new record or a new URL. However, we encountered some difficulties among them the impossibility of calculating 10 columns of the dataset for different reasons which are specified for each field in Table 4.

Table 4. Justification of the columns was eliminated from the dataset because it was not possible to calculate them.

Number	Field discarded	Reasons
1	server_client_domain	The concept of the character is not well defined
2	domain_spf	Non-existence of a Python library that obtains this information to automate it.
3	asn_ip	Non-existence of a Python library that obtains this information to automate it.
4	time_domain_activation	Non-existence of a Python library that obtains this information to automate it.
5	time_domain_expiration	Non-existence of a Python library that obtains this information to automate it.
6	qty_ip_resolved	The concept of the character is not well defined
7	qty_nameservers	The concept of the character is not well defined
8	qty_mx_servers	Non-existence of a Python library that obtains this information to automate it.

3.4 Dataset preparation

In this stage the model preparation is performed, the model will be based on the Mendeley Dataset [25]. First, it is separated into 2 datasets, one with the lexical features, host-based features, and web content features that has 101 columns in which the elimination of the 10 columns that have not been possible to calculate in the previous stage has already been performed, and the second with the result label corresponding to each record. Next, a *train_test_split* is performed in which the dataset is divided in a ratio of 7 to 3, where 70% of the dataset will be used for training and 30% of it to test the effectiveness of this and with a random state seed of 92.

Table 5. Comparison of the accuracy of the models with the different techniques after eliminating the columns that cannot be calculated.

Machine Learning Technology	Accuracy of "0"	Accuracy of "1"	Average Accuracy
Support Vectors (gamma='auto')	0.94	0.90	0.93
Classification trees	0.94	0.89	0.92
Random Forest (n_estimators=80)	0.96	0.92	0.94

3.5 Training of RF, SVM, and CA models

In the case of the first model with the Random Forest (RF) technique with a single non-predetermined parameter which is, in this case, the number of estimators which is left at 90. By using bagging, what is happening is that different trees see different portions of the data. No single tree sees all the training data. This results in each tree being trained with different data samples for the same problem. Thus, by combining their results, some errors are compensated by others, and we have a better generalizing prediction [26].

In the case of the second model with the Support Vector Machine (SVM) technique with only one non-default parameter which is, in this case, the gamma which is set to automatic. Support Vector Machines allow finding the optimal way to classify among several classes. The optimal classification is performed by maximizing the margin of separation between classes. The vectors that define the edge of this separation are the support vectors. In case the classes are not linearly separable, we can use the kernel trick to add a new dimension where they are [27].

In the case of the third model with the technique of classification trees with all parameters predetermined. A classification or decision tree is a predictive model that divides the predictor space by grouping observations with similar values for the response or dependent variables. To divide the sample space or into subregions, it is necessary to apply a series of rules or decisions so that each subregion contains the largest possible proportion of individuals from one of the populations [28].

Starting with the training, we choose the value of our X which will be all the numerical values on which based on them the corresponding predictions will be made, and the Y value which will be our field to predict which is in this case the value 'phishing', then the training train is performed with the *Sklearn* library [30] and the *model_selection* module with a training size of 30% and with a random state seed of 85 for the RF case and, 92 for MVC and AC.

For the case of RF, we use again the *Sklearn* library implementing the ensemble module which allows us to import the *RandomForestClassifier* which already implements the internal logic and operations to train a model with the Random Forest technique, also the number of estimators is defined in 90. After that, it is possible to train the model with the "fit" method. Within this function, the model is trained, and we have the model ready as the value returned by the training function of the Random Forest model. Figure 4 shows the plotted confusion matrix resulting from the training of the model in question.

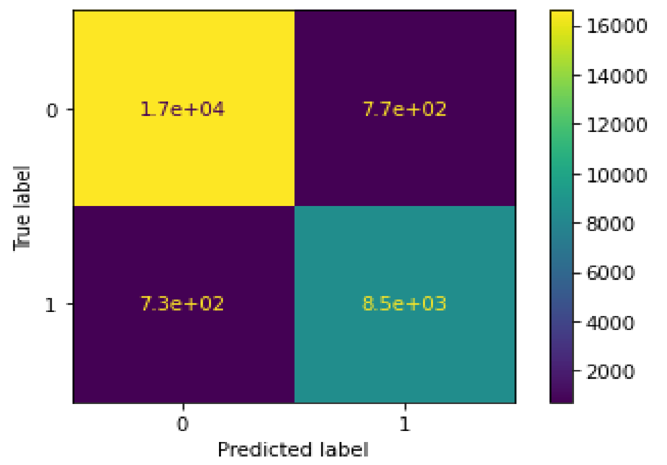


Figure 4. Confusion matrix in plots of the model training with the Random Forest technique.

For the case of SVM, the *Sklearn* library is used again implementing the *SVM* module which allows us to import the SVC which already implements the internal logic and operations to train a model with the Support Vector Machine technique, also the parameter of the gamma function is set to 'auto'. After that, it is possible to train the model with the "fit" method. Within this function, the model is trained, and we obtain the ready model as the value returned by the training function of the Support Vector Machine model. Figure 5 shows the plotted confusion matrix resulting from the training of the model in question.

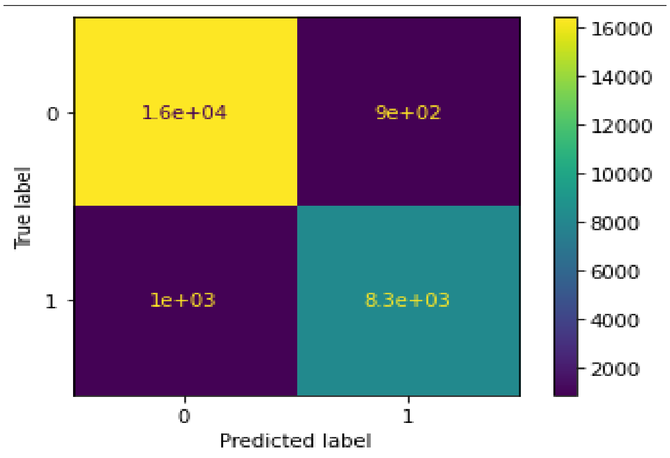


Figure 5. Confusion matrix on model training plots with the Support Vector Machine technique.

For the case of CA, we use again the *Sklearn* library implementing the tree module which allows us to import the *DecisionTreeClassifier* which already implements the internal logic and operations to train a model with the classification trees technique, in this case, no internal parameter is defined in the function, and everything is kept by default. After that, it is possible to train the model with the "fit" method. Within this function, the model is trained, and we have the model ready as the value returned by the training function of the classification trees model. Figure 6 shows the plotted confusion matrix resulting from the training of the model in question.

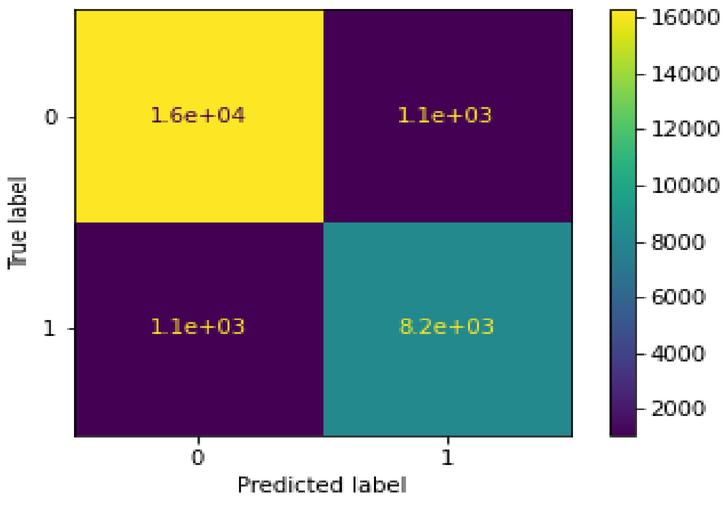


Figure 6. Confusion matrix on model training plots using the classification tree technique.

Then the *Joblib* library [31] is used to persist the model in a file in the cloud and not have to train the model whenever you want to use it and finally the value is cast to a String for the convenience of the API function that will implement it.

3.6 AND (ACM) and OR (OCM) Conjunction Model Training

In the case of the AND and OR conjunction model, only the models that have already been implemented within a function where the logical function as such is performed are reused. This function returns a string with the result, which is used by the API.

3.7 Elaboration of the API Backend

The backend is performed on the same notebook where the model, model training, and feature extraction have been developed. Two Python libraries are installed that allow running and lifting the API in the same notebook, on the one hand, *ColabCode* [32] that allows deploying Machine Learning models within the same *Google Colab* and on the other hand *FastApi* [33] that allows performing the necessary configuration for the API, such as the port, the configuration of the routes and provides an interface to test the endpoints within the same route.

It should be noted that *ColabCode* uses *Ngrok* [34] to use a dynamically generated URL, which points to a web service running locally on our PC. So, some additional CORS security configurations were needed so that our backend that has an external IP can connect to the frontend that has a local IP.

3.8 Frontend development and layout of the web application

The layout uses html5 and css3 with the *Bootstrap* framework [35] for the more agile design issues. On the frontend side as such, *javascript* is used with *Jquery* [36] to simplify the treatment of the endpoints and the data returned by it. To raise a frontend server and do it in a quite fast and practical way an extension of the Visual Studio Code editor is used, the extension is called Live Server [37] which allows us to raise a local server that every time that some change is made in the files it is reloaded automatically. It is important to indicate that the interface of the web application is developed in Spanish because it is the native language of the country where the tool is developed and because it is the language of the target audience where it is to be used.

4 Validation

The validation is performed with 2 case studies, the first one using 5 URLs obtained from *Phishtank* [29] which are already labeled by a large community as phishing URLs, and the second one, using 5 URLs indexed in Google validated by Google itself as safe URLs that do not have any phishing. Each case study has 5 scenarios which are the 5 types of prediction techniques which are: (1) Random Forest, (2) Support Vector Machine, (3) Classification Trees, (4) Conjunction of AND Techniques, and (5) Conjunction of OR Techniques. The process is described in Table 6.

Table 6. Description of the 2 case studies with scenarios.

Machine Learning Technology	Accuracy of "0"	Accuracy of "1"	Average Accuracy
1	URLs obtained by PhishTank	5	Random Forest
			Support Vector Machine
			Classification trees
			The conjunction of AND Techniques
			The conjunction of OR Techniques
2	URLs indexed in Google	5	Random Forest
			Support Vector Machine
			Classification trees
			The conjunction of AND Techniques
			The conjunction of OR Techniques

Each case study is validated with the metrics described below. The validation is performed using a table called confusion matrix that describes the classification results in detail. For the 2 categories, a 2*2 matrix is used which is shown in Table 7.

Table 7. Confusion matrix obtained from the 2 categories with possible outcomes.

	Predicted as positive (PP)	Predicted as negative (PN)
Tagged as positive (TP)	True positive (TP)	False-negative (FN)
Tagged as negative (TN)	False-positive (FP)	True negative (TN)

Based on the values in Table 7, we will obtain 4 metrics based on the numbers obtained from the tests we will perform, which are as follows.

The true positive rate (TPR): also known as sensitivity, is the number of correct classifications of positive samples that represents the proportion of the number of positive samples, which has the following formula:

$$TPR = TP/(TP+FN) \tag{1}$$

False-positive rate (FPR): the number of negative misclassification samples represents the proportion of the total number of negative samples, which has the following formula:

$$FPR = FP/(FP+TN) \tag{2}$$

Exactitude: the number of correct classification of samples represents the proportion of all samples, which has the following formula:

$$Exac = (TP+TN)/(TP+FP+FN+TN) \tag{3}$$

Accuracy: the number of correct classifications of positive samples represents the proportion of all positive samples, which has the following formula:

$$Acc = (TP)/(TP+FP) \tag{4}$$

5 Results and Discussion

In each of the case studies, the 5 techniques proposed in the contribution were considered: Random Forest, Support Vector Machine, Classification Trees, AND conjunction model an OR conjunction model.

We define the URLs to be tested for both case studies, as defined in the validation section the URLs of the first case study are obtained from the *Phishtank* page. While the URLs of the second case study are indexed in the Google search engine. The URLs used are shown in Table 8.

Table 8. Test URLs for each of the scenarios.

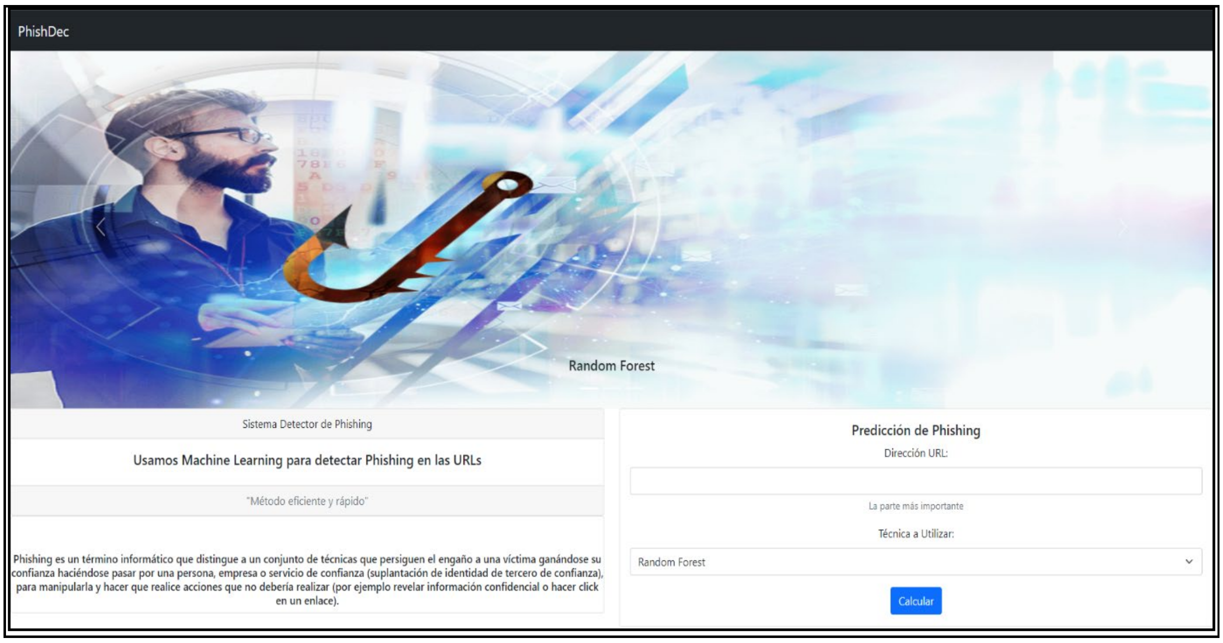
	Predicted as positive (PP)	Predicted as negative (PN)
1	http://3342533cq.000webhostapp.com/step1.php	https://www.twitch.tv/
2	https://dogecoingod.com/homee/load/dhl/?i=i&0=aaaa@example.jp	https://icpna.instructure.com/
3	http://grupwhatspss-ku.duckdns.org/	https://www.facebook.com/
4	https://anoueaom.dshqnn.cn/	https://www.scribbr.es/detector-de-plagio/generador-apa/
5	https://louywd.club/r/LT-867HBRAvaw	https://www.kaggle.com/

After entering the URLs with the 5 different techniques for each of the 2 scenarios, the results shown in Table 9 are as follows.

Table 9. Phishing prediction results.

URL	Technique	Result Scenario 1	Result Scenario 2
1	RF	0	0
	SVM	1	1
	CA	1	1
	ACM	1	1
	OCM	0	0
2	RF	1	0
	MVC	1	1
	AC	0	0
	MCA	1	1
	MCO	0	0
3	RF	0	0
	MVC	1	1
	AC	1	0
	MCA	1	1
	MCO	0	0
4	RF	0	1
	MVC	1	1
	AC	1	0
	MCA	1	1
	MCO	0	0
5	RF	1	0
	MVC	1	1
	AC	1	0
	MCA	1	1
	MCO	1	0

For both case studies, the *PhishDec* web platform is accessed and shows a very attractive interface where you can see at the top images related to phishing and computer security. Then, in the lower section, on the left side, there is a section with important related information and on the right side, both fields for entering and selecting information for the calculation of the prediction, as shown in Figure 7.



Sistema Detector de Phishing

Usamos Machine Learning para detectar Phishing en las URLs

"Método eficiente y rápido"

Phishing es un término informático que distingue a un conjunto de técnicas que persiguen el engaño a una víctima ganándose su confianza haciéndose pasar por una persona, empresa o servicio de confianza (suplantación de identidad de tercero de confianza), para manipularla y hacer que realice acciones que no debería realizar (por ejemplo revelar información confidencial o hacer click en un enlace).

Predicción de Phishing

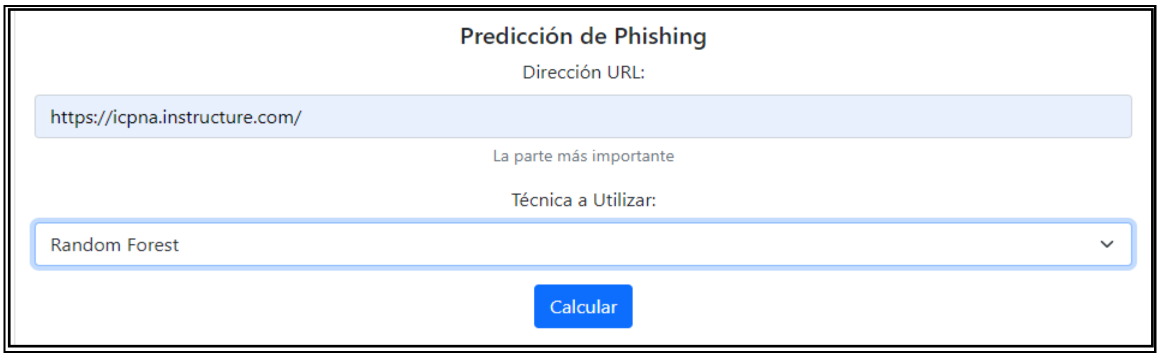
Dirección URL:

La parte más importante

Técnica a Utilizar: Random Forest

Figure 7. PhishDec web platform.

Then enter the URL in the first field and select which of the 5 options to predict with as shown in Figure 8.


Predicción de Phishing

Dirección URL:

https://icpna.instructure.com/

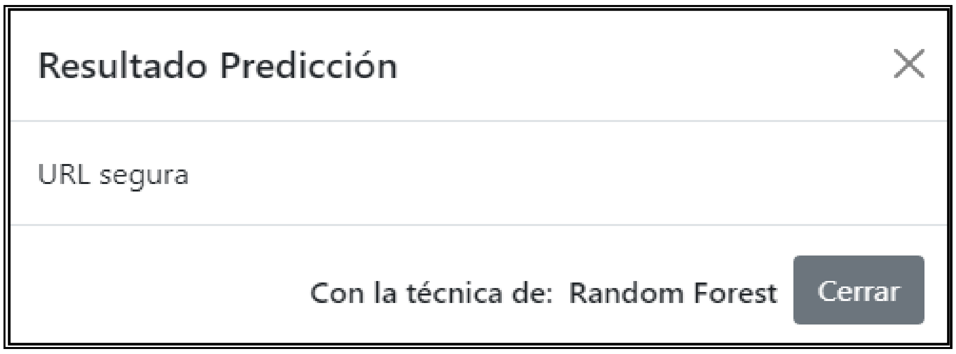
La parte más importante

Técnica a Utilizar:

Random Forest

Figure 8. URL prediction fields.

By clicking on calculate we will be able to see the result of the prediction in a Modal that appears as shown in Figure 9.


Resultado Predicción

URL segura

Con la técnica de: Random Forest

Figure 9. Modal displaying the result of the prediction.

After entering the URLs of both case studies, we can form our joint Confusion Matrix of all the techniques in the 2 scenarios, which is shown in Table 10.

Table 10. Confusion matrix of all the techniques in the 2 scenarios.

Technique	Tagged	PP	PN
RF	TP	2 (TP)	3 (FN)
	TN	1 (FP)	4 (TN)
SVM	TP	5 (TP)	0 (FN)
	TN	5 (FP)	0 (TN)
CA	TP	4 (TP)	1 (FN)
	TN	1 (FP)	4 (TN)
ACM	TP	5 (TP)	0 (FN)
	TN	5 (FP)	0 (TN)
OCM	TP	1 (TP)	4 (FN)
	TN	0 (FP)	5 (TN)

As shown in Table 10, the SVM, CA and ACM techniques are the most effective in predicting positive phishing cases, while the RF, CA and OCM techniques are the most effective in predicting negative phishing cases. We also found something interesting and that is that the CA technique is the one that performs best in both fields, having an 80% effectiveness in both cases in predicting positive and negative cases.

We then proceed to calculate the metrics corresponding to the confusion matrix:

Table 11. Metrics were obtained for each technique used.

Técnica	(1) TPR	(2) FPR	(3) Exac	(4) Acc
RF	0.4	0.2	0.6	0.67
SVM	1	1	0.5	0.5
CA	0.8	0.2	0.8	0.8
ACM	1	1	0.5	0.5
OCM	0.2	0	0.6	1

In Table 11, we can observe that the AC technique is the one that obtains the highest TVP while the OLS technique is the one that obtains the lowest value for this metric. As for the TFP, both MVC and MCA obtain the perfect score for this metric, while both RF and AC obtain the lowest values for this metric. If we look at Accuracy, we see that AC obtains the highest indicator of all techniques while MCV and MCA are tied with accuracy at half. And finally, talking about accuracy, MCO obtains the perfect value, while MVC and MCA are tied again with the lowest values.

6 Conclusions and future work

In this research, a web architecture was developed by applying machine learning to predict whether a URL is safe or not, to enhance the security of people surfing the Internet, and to compare different machine learning techniques well known.

For the research, each of the phases of the proposed methodology was applied: research questions, training of the models, development of the web architecture, and, finally, validation of the proposal.

In the training of the models, a dataset was compiled with sufficient documentation and information to obtain each of its characteristics, then the different techniques were analyzed on the raw dataset. Subsequently, all possible feature extraction functions were performed, eliminating the fields that are not possible to calculate, then the dataset was prepared, and the training of the machine learning models was started, to develop the backend and the frontend of the web architecture. Then, the proposal was validated with 2 case studies, where the first case used URL addresses labeled as positive and the second case used URL addresses labeled as negative.

The following statements were obtained from the validation results: (i) The Classification Trees technique has the highest effectiveness in both predicting that a URL is secure and that it is not secure. (ii) The Support Vector Machine technique is very good at detecting insecure URLs. (iii) Both AND and OR conjunction models are quite effective in each of their cases, e.g., AND conjunction is very good at detecting positive phishing cases, while OR conjunction is quite good at detecting negative phishing cases. There is currently a multitude of phishing solutions, but none that stand out from the crowd, and it is expected that with advances in technology, phishing will become increasingly difficult to detect and anti-phishing solutions will have to make the grade. Future work will aim to refine the model and make a joint model that has near-perfect accuracy for both cases and further refine the platform to make it even simpler to use.

References

[1] Empresa Peruana de Servicios Editoriales S. A. EDITORA PERÚ. (2021, 29 enero). El phishing es el ciberataque que más aumentó en Perú a raíz de la pandemia. Noticias | Agencia Peruana de Noticias Andina.https://andina.pe/agencia/noticia-el-phishing-es-ciberataque-mas-aumento-peru-a-raiz-de-pandemia-831392.aspx

[2] IT Digital Media Group. (2020, 27 enero). El 55% de las empresas sufrieron un ataque de phishing exitoso en 2019. Seguridad | IT Reseller. https://www.itreseller.es/seguridad/2020/01/el-55-de-las-empresas-sufrio-un ataque-de-phishing-exitoso-en-2019

[3] Otero, C. (2019, 19 marzo). Cuántos ataques Phishing suceden al día, las cifras de Microsoft Security. AS.com. https://as.com/meristation/2019/03/19/betech/1553035270_030413.html

[4] Pasquali, M. (2020, 1 septiembre). Los intentos de phishing en tiempos de COVID-19. Statista Infografías. https://es.statista.com/grafico/18427/intentos-de-phishing durante-la-pandemia/

[5] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345–357. https://doi.org/10.1016/j.eswa.2018.09.029

[6] Liu, C., Wang, L., Lang, B., & Zhou, Y. (2018). Finding effective classifier for malicious URL detection. Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences - ICMSS 2018. Published. https://doi.org/10.1145/3180374.3181352

[7] Parekh, S., Parikh, D., Kotak, S., & Sankhe, S. (2018). A New Method for Detection of Phishing Websites: URL Detection. 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). Published. https://doi.org/10.1109/icicct.2018.8473085

[8] L. Wong, D. Mauricio, and G. Rodriguez (Feb. 2017). "A Systematic Literature Review About Software Requirements Elicitation". In Journal of Engineering Science and Technology (JESTEC), pp. 296 – 317.

[9] Higashino, M., Kawato, T., Ohmori, M., & Kawamura, T. (2019). An Anti-phishing Training System for Security Awareness and Education Considering Prevention of Information Leakage. 2019 5th International Conference on Information Management (ICIM). Published. https://doi.org/10.1109/infoman.2019.8714691

[10] Nathezhtha, T., Sangeetha, D., & Vaidehi, V. (2019). WC-PAD: Web Crawling based Phishing Attack Detection. 2019 International Carnahan Conference on Security Technology (ICCST). Published. https://doi.org/10.1109/ccst.2019.8888416

[11] Geng, G. G., Yan, Z. W., Zeng, Y., & Jin, X. B. (2018). RRPhish: Anti-phishing via mining brand resources request. 2018 IEEE International Conference on Consumer Electronics (ICCE). Published. https://doi.org/10.1109/icce.2018.8326085

[12] Moorthy, R. S., & Pabitha, P. (2020). Optimal Detection of Phising Attack using SCA based K-NN. Procedia Computer Science, 171, 1716–1725. https://doi.org/10.1016/j.procs.2020.04.184

[13] Zeng, V., Zhou, X., Baki, S., & Verma, R. M. (2020). PhishBench 2.0: A Versatile and Extendable Benchmarking Framework for Phishing. Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. Published. https://doi.org/10.1145/3372297.3420017

[14] Gori Mohamed, J., & Visumathi, J. (2020). A predictive model of machine learning against phishing attacks and effective defense mechanisms. Materials Today: Proceedings. Published. https://doi.org/10.1016/j.matpr.2020.09.612

[15] Kovalluri, S. S., Ashok, A., Singanamala, H., & P., P. (2018). LSTM Based Self-Defending AI Chatbot Providing Anti-Phishing. Proceedings of the First Workshop on Radical and Experiential Security. Published. https://doi.org/10.1145/3203422.3203431

[16] Chapla, H., Kotak, R., & Joiser, M. (2019b). A Machine Learning Approach for URL Based Web Phishing Using Fuzzy Logic as Classifier. 2019 International Conference on Communication and Electronics Systems (ICCES). Published. https://doi.org/10.1109/icces45898.2019.9002145

[17] Zhu, E., Ju, Y., Chen, Z., Liu, F., & Fang, X. (2020). DTOF-ANN: An Artificial Neural Network phishing detection model based on Decision Tree and Optimal Features. Applied Soft Computing, 95, 106505. https://doi.org/10.1016/j.asoc.2020.106505

[18] Tan, C. L., Chiew, K. L., Yong, K. S., Sze, S. N., Abdullah, J., & Sebastian, Y. (2020). A graph-theoretic approach for the detection of phishing webpages. Computers & Security, 95, 101793. https://doi.org/10.1016/j.cose.2020.101793

[19] Wei, W., Ke, Q., Nowak, J., Korytkowski, M., Scherer, R., & Woźniak, M. (2020). Accurate and fast URL phishing detector: A convolutional neural network approach. Computer Networks, 178, 107275. https://doi.org/10.1016/j.comnet.2020.107275

[20] Patil, S., & Dhage, S. (2019). A Methodical Overview on Phishing Detection along with an Organized Way to Construct an Anti-Phishing Framework. 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS). Published. https://doi.org/10.1109/icaccs.2019.8728356

[21] Lawson, P., Pearson, C. J., Crowson, A., & Mayhorn, C. B. (2020). Email phishing and signal detection: How persuasion principles and personality influence response patterns and accuracy. Applied Ergonomics, 86, 103084. https://doi.org/10.1016/j.apergo.2020.103084

[22] Frauenstein, E. D., & Flowerday, S. (2020). Susceptibility to phishing on social network sites: A personality information processing model. Computers & Security, 94, 101862. https://doi.org/10.1016/j.cose.2020.101862

[23] Chen, R., Gaia, J., & Rao, H. R. (2020). An examination of the effect of recent phishing encounters on phishing susceptibility. Decision Support Systems, 133, 113287. https://doi.org/10.1016/j.dss.2020.113287

[24] Aleroud, A., Abu-Shanab, E., Al-Aiad, A., & Alshboul, Y. (2020). An examination of susceptibility to spear phishing cyber attacks in non-English speaking communities. Journal of Information Security and Applications, 55, 102614. https://doi.org/10.1016/j.jisa.2020.102614

[25] VrbanÄiÄ, G. (2020, 24 septiembre). Phishing Websites Dataset. Mendeley. https://data.mendeley.com/datasets/72ptz43s9v/1

[26] Heras, J. M. (2020, 18 septiembre). Random Forest (Bosque Aleatorio): combinando árboles. IArtificial.net. https://www.iartificial.net/random-forest-bosque-aleatorio/

[27] Heras, J. M. (2019, 28 mayo). Máquinas de Vectores de Soporte (SVM). IArtificial.net. https://www.iartificial.net/maquinas-de-vectores-de-soporte-svm/

[28] Qué son los árboles de decisión y para qué sirven. (2020). Máxima Formación. https://www.maximaformacion.es/blog-dat/que-son-los-arboles-de-decision-y-para-que-sirven/

[29] PhishTank | Join the fight against phishing. (2021). PhishTank. https://phishtank.org/

[30] scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation. (2021). Scikit-Learn. https://scikit-learn.org/stable/

[31] Embarrassingly parallel for loops — joblib 1.1.0.dev0 documentation. (2021). Joblib. https://joblib.readthedocs.io/en/latest/parallel.html

[32] Gupta, K. (2021, 4 febrero). ColabCode: Deploying Machine Learning Models From Google Colab. Medium. https://towardsdatascience.com/colabcode-deploying-machine-learning-models-from-google-colab-54e0d37a7b09

[33] FastAPI. (2021). FastApi. https://fastapi.tiangolo.com/

[34] I. (2021). ngrok - secure introspectable tunnels to localhost. Ngrok. https://ngrok.com/

[35] Otto, M. J. T. (2021). Bootstrap. Boostrap. https://getbootstrap.com/

[36] JS Foundation - js.foundation. (2021). jQuery. Jquery. https://jquery.com/

[37] Live Server - Visual Studio Marketplace. (2021). Visual Studio MarketPlace. https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer