Hindawi
Computational and Mathematical Methods
Volume 2023, Article ID 5780357, 8 pages
https://doi.org/10.1155/2023/5780357

Research Article

Towards Supercomputing Categorizing the Maliciousness upon Cybersecurity Blacklists with Concept Drift

M. V. Carriegos,[1] **N. DeCastro-García**,[1] **and D. Escudero**[2]

[1] *Departamento de Matemáticas, Universidad de León, León, Spain*
[2] *RIASC, Instituto de Ciberseguridad, Universidad de León, León, Spain*

Correspondence should be addressed to M. V. Carriegos; miguel.carriegos@unileon.es

Received 13 December 2022; Revised 18 April 2023; Accepted 25 April 2023; Published 20 May 2023

Academic Editor: Pedro Alonso

In this article, we have carried out a case study to optimize the classification of the maliciousness of cybersecurity events by IP addresses using machine learning techniques. The optimization is studied focusing on time complexity. Firstly, we have used the extreme gradient boosting model, and secondly, we have parallelized the machine learning algorithm to study the effect of using a different number of cores for the problem. We have classified the cybersecurity events' maliciousness in a biclass and a multiclass scenario. All the experiments have been carried out with a well-known optimal set of features: the geolocation information of the IP address. However, the geolocation features of an IP address can change over time. Also, the relation between the IP address and its label of maliciousness can be modified if we test the address several times. Then, the models' performance could degrade because the information acquired from training on past samples may not generalize well to new samples. This situation is known as concept drift. For this reason, it is necessary to study if the optimization proposed works in a concept drift scenario. The results show that the concept drift does not degrade the models. Also, boosting algorithms achieving competitive or better performance compared to similar research works for the biclass scenario and an effective categorization for the multiclass case. The best efficient setting is reached using five nodes regarding high-performance computation resources.

1. Introduction

Data science has become essential for companies and organizations to extract actionable knowledge. This can be a competitive edge whose value is directly related to the quality of the used datasets and the efficiency of the models and their implementations. The case of computer security incident response teams (CSIRTs) and the managing of cybersecurity databases is one of the best-known examples of the scenario described. A cybersecurity database is a database with reports of cybersecurity events. A cybersecurity report contains data about a cybersecurity incident that is considered malicious. The information included is, for example, its geolocation, time stamp, type of event, and confidentiality. Then, a cybersecurity database contains a lot of unstructured and correlated information. Several sources of information provide streams of data, and they are enriched by human agents, usually by requesting external platforms of blacklists or malware platforms. The information is updated daily, weekly, or online depending on the source and the type of event that is reported. Data flow is constant and dynamic, generating large volumes of data. In this scenario, knowing the severity of a cybersecurity event of potentially malicious activity is essential to determine an appropriate response.

In this article, we present a study case in which we optimize the application of supervised machine learning (ML) models to classify cybersecurity data streams of IP addresses in terms of the level of the maliciousness of the associated cybersecurity incident. In particular, we have applied the extreme gradient boosting algorithm and used geolocation features [1]. The study has been carried out on 99720 IP addresses provided by the Spanish National Cybersecurity Institute (INCIBE). We have conducted the experiments in two scenarios: biclass and multiclass. Also, data distribution can change over time, yielding a concept drift scenario and increasing the possible error associated with the models [2–4]. Then, detecting concept drift, or the absence, is crucial to evaluate the suitability of the models and the possible effect of this on the classification of the maliciousness [5]. For this reason, we have extracted the data from the experiments at two different time points, and we have analyzed the degree of concept drift and the possible effect on the accuracy of the results.

Since the usual cybersecurity databases are huge, we have created the ML models using a different number of cores to optimize the procedure's time complexity. Then, we highlight the necessary high-performance computing (HPC) resources, considering the validity of incoming data and resulting measures.

The concrete results of our experiments are three-fold; it is shown that there is no significant concept drift among the proposed databases; it is evaluated the degradation of geolocation features and, finally, the suitability of HPC to the creation of ML models among those cybersecurity databases. Regarding the latter question, although HPC does not improve the ML algorithms' accuracy/sensitivity/specificity performance, the optimum number of cores [6] is reached with 5, where our algorithms gain 50% of execution time.

The article is organized as follows: In Section 2, we develop the related work. In Section 3, experimental details are explained. The results are included and discussed in Section 4. Finally, the conclusions and the references are given.

2. Related Work

A cybersecurity event is a cybersecurity change that may have an impact on organizational operations (including capabilities or reputation). (https://csrc.nist.gov/glossary/term/cybersecurity_event). The severity of a cybersecurity event is a measure that determines its risk or maliciousness. The assessment of this characteristic is crucial to ensure that the countermeasures that are taken are appropriate. For this reason, an increasing body of literature is trying to solve this task from several approaches. The perspective depends mainly on the type of cybersecurity event with which we deal and the resources that we have available. There are tools based on different methodologies and standards (Microsoft Security Bulletin Vulnerability Rating [7], Common Vulnerability Scoring System (CVSS) [8], Open Web Application Security Project (OWASP) Risk Rating Methodology [9], and Cyber Incident Scoring System [10], among others) or other approaches based on data science and ML models [11]. In all cases, we need a tool that not only determines the maliciousness of a cybersecurity event as closely as possible but also attaches importance to identifying false negatives. These cases may become difficult situations for citizens, institutions, and companies.

This work focuses on the maliciousness assigned to an IP address. Then, we use registers of the IPs of the different several cybersecurity events as any occurrence of an adverse nature in a public or private sphere within a country's information and communication networks. In particular, an IP address's severity is considered a measure of its reputation. We deal with cybersecurity databases with all IP addresses associated with threats. It is not a question of determining whether an address is malicious. We know that all the registers are "threats." The point is to assess the level of maliciousness to provide an adequate response.

Measuring the maliciousness of the reputation of an IP address has been studied from several perspectives. The first approach is using blacklists to create alerts. These works apply techniques such as time series forecasting, clustering, or ML models based on data in the blacklists reaching maximum accuracy rates of 0.776 and predicting if an IP can be considered malicious or not. One of the disadvantages of this approach is the vast volume of black or whitelisted IPs to create the models [12–15]. The second approach takes advantage of contextual information about the IP address, such as geolocation, DNS registers, hosts, and the proper address. This information is easy to extract and does not require a large volume of data. In this case, the models that are created are based on computations about the frequencies at which contextual information appears, or again, clustering techniques [16–19]. A global accuracy of 0.77 is reached to classify an IP address as malicious or not. Another perspective is analyzing the dynamical behavior of the IP address from logs or intrusion detection systems [20–22]: the number of alerts that are generated, requests, access, etc. Although this approach reaches the best accuracies, 0.91-0.93, it implies additional resource costs because it requires monitoring and extracting the features online.

As we mentioned, one of the most used approaches to categorizing the maliciousness of an IP address is applying ML models. However, although we find relevant features such as geolocation variables, the values of these variables, or the blacklists are expected to change over time, leading to a concept drift scenario. A concept drift scenario is that in which there is a change in the data distribution ([3]). If X denotes the feature vector space in a data sample and Y is the X label space, then the concept drift happens if $P_t(y|X) \neq P_{t+\Delta t}(y|X)$ and/or $P_t(X) \neq P_{t+\Delta t}(X)$, where $P_t(X)$ is the marginal distribution of data in an instant t. Analogously, $P_t(y|X)$. The drift is real, virtual, or a combination of both if the differences appear on one or the other—or both, probabilities [2–4]. Ough there are studies in which the concept drift is involved, usually, something other than this is the focus of the research.

Recently, in [1], an optimal feature set to categorize an IP address's maliciousness was configured with Autosklearn [23] by analyzing contextual variables joint with temporal information extracted by blacklists. Although the feature set has been optimized, the question now is whether the implementation can be optimized in terms of efficiency and scalability. For this reason, in this work, we have conducted a case study with the optimal configuration set of [1], the geolocation features, but changing the ML algorithm and adding a possible parallelization by HPC resources. Also, we have conducted the experiments in a biclass and a multiclass scenario to compare our results with other research works.

3. Materials and Methods

In this section, the experimental details are described.

3.1. Features Extraction. The selected features that are used are related to the geolocation and the time stamp of the IP address. After analyzing the existing studies on IP classification [1], we keep with the geolocation of the IP address without feature selection. In all, we have extracted five features:

(i) Latitude and longitude: these are measured as decimal numbers (F_1 and F_2)

(ii) The country code: this is categorized as an integer number taking values from 0 to the total of countries represented by the ISO 3166-1 (F_3)

(iii) The IP is transformed into a numerical integer value in the following way (F_4)

(iv) $A.B.C.D \longrightarrow A * 256^3 + B * 256^2 + C * 256^1 + D * 256^0$

(v) The time stamp is transformed in UNIX time (number of elapsed seconds from 01/01/1970 at 00 : 00) (F_5)

3.2. Datasets. In the experimentation, several datasets have been used and constructed:

(1) *BL*, the reputation list or blacklist of IPs (INCIBE has provided this dataset under a confidentiality agreement). This sample is from May 2021. It contains 99720 IPs. We have the IP address, time stamp, and associated severity from each IP. An expert of INCIBE assigns the severity value. It takes values 1, 3, 6, and 9, ordered from less severity to high severity. We have analyzed two scenarios: biclass and multiclass

(2) Then, we constructed another blacklist from *BL* by transforming the labels 1 and 3 into 0 and 6 and 9 into 1. Then, we clustered the samples with very low and low severity on the one hand and grouped the samples with high and very high severity. This experiment is denoted by *B*. Also, we have repeated the experiment but transforming the label 1 into 0, and grouping the labels 3, 6, and 9 into 1. This experiment is denoted by B'

(3) The second dataset, D_1, is a subset of *BL* with 55728 IPs. In this dataset, we have extracted the set of the features latitude, longitude, and country code by querying to Maxmind (https://www.maxmind.com/en/home) in May 2021

(4) The third dataset, D_2, is D_1, but the geolocation features were extracted in April 2022 to analyze the presence of concept drift

The reason to take a subset of 55728 IPs from *BL* and not all the IP addresses is that these present changes in the geolocation features.

Table 1: Frequency of each class of the severity in the datasets D_1, D_2, and BL.

Dataset	Severity	Frequency	Proportion
BL	1	8402	8.4255%
	3	24943	25.0130%
	6	54437	54.5898%
	9	11938	11.9715%
D_1 and D_2	1	3618	6.4922%
	3	14126	25.3481%
	6	31966	57.3607%
	9	6018	10.7988%
B	0	33345	33.4386%
	1	66375	66.5614%
B'	0	8402	8.4256%
	1	91318	91.5744%

Table 2: Results of applying ML-constructed models with D_t predicting over $D_{t+\Delta t}$. B, B' = bi − class, M = multiclass.

Scenario	Response variable	Average (B)	Average (M)	Average (B')
Biclass	MCC	.7622	.7586	.8762
	Accuracy	.8982	.8590	.9852
	Sensitivity	.8982	.8590	.9852
	Specificity	.8982	.9530	.9852

Table 3: Confusion matrix for the biclass scenario B. In parenthesis, the rate of success by row.

True label	Predicted label		Total
	0	1	
0	3603 (81.80%)	802 (18.20%)	4405
1	616 (6.47%)	8911 (93.53%)	9527
Total	4219	9713	13932

Table 4: Confusion matrix for the biclass scenario B'. In parenthesis, the rate of success by row.

True label	Predicted label		Total
	0	1	
0	775 (86.08%)	130 (13.92%)	905
1	87 (0.60%)	12940 (99.4%)	13027
Total	858	13074	13932

The datasets are in https://github.com/amunc/IP_datasets, but by the confidentiality agreement, the variable with the IP is transformed to a numerical value for anonymizing it. The proportion of each severity class in the datasets is included in Table 1.

3.3. Research Questions. The research questions are described below:

Table 5: Confusion matrix for the multiclass scenario. In parenthesis, the rate of success by row.

True label	Predicted label				Total
	1	3	6	9	
1	846 (92.97%)	18 (1.98%)	39 (4.29%)	7 (0.77%)	910
3	23 (0.66%)	2885 (82.55%)	429 (12.27%)	158 (4.52%)	3495
6	44 (0.55%)	420 (5.24%)	7376 (92.1%)	169 (2.11%)	8009
9	15 (0.99%)	273 (17.98%)	369 (24.3%)	861 (56.72%)	1518
Total	928	3596	8213	1195	13932

R1: Is there any concept drift in the geolocation features of IP addresses between D_1 and D_2?

R2: Do the results obtained from applying the ML models show any degradation when geolocation features change?

R3: Is HPC a suitable tool to face the computation of concept drift?

3.4. Analyses. To analyze the presence of concept drift in the geolocation features, we have computed $\sigma_{t,t+\Delta t}(Z)$ and $\sigma_{t,t+\Delta t}^{Y|X}$ where

$$\sigma_{t,t+\Delta t}(Z) = \frac{1}{2} \sum_{\bar{z} \in Dom(Z)} |P_t(\bar{z}) - P_{t+\Delta t}(\bar{z})| y, \quad (1)$$

$$\sigma_{t,t+\Delta t}^{Y|X} = \sum \left[\frac{P_t(\bar{x}) + P_{t+\Delta t}(\bar{x})}{2} \frac{1}{2} \sum |P_t(y|\bar{x}) - P_{t+\Delta t}(y|\bar{x})| \right], \quad (2)$$

following the approach given in [24]. Both take values between 0 and 1. Here, we take $D_1 = D_t$, and $D_2 = D_{t+\Delta t}$.

The ML models have been created by extreme gradient boosting, XGB, [25], whose implementation is [26]. The hyperparameters that have been optimized are the depth and the number of trees. 70% of the data is used for hyperparameter optimization, and the other 30% is used to evaluate the suitability of the optimized models. For all experiments, we have performed 10-fold cross-validation. Since the datasets are unbalanced, the response variables that have been analyzed are the accuracy and Matthews' coefficient, $\text{MCC} = \text{TP} \cdot \text{TN} - \text{FP} \cdot \text{FN} / \sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}$ where TP, TN, FP, and FN denote the true positives, true negatives, false positives, and false negatives, [27] Also, we have computed the recall (or sensitivity) and the selectivity (or specificity) of the models for each class.

Also, to evaluate the computational cost, we have collected the total time in seconds of the procedure. This includes the time model construction, the time features selection, the time feature construction, the time preprocessing, and the time data load. We highlight that, in our case, the time feature selection is 0.

Finally, to decide if HPC is a suitable tool to face this problem, we have carried out all the above experiments with different cores parallelizing the XGB algorithm. We have performed the analyses with 1, 2, 3, 5, 10, and 16 cores. The experiments have been carried out with Python 3.8.

Table 6: Results of applying ML-constructed models with $D_{t+\Delta t}$ predicting over $D_{t+\Delta t}$. B, B' = biclass, M = multiclass.

Scenario	Response variable	Average	Median
Biclass B	MCC	.7388	.7393
	Accuracy	.8877	.8880
	Sensitivity	.8877	.8880
	Specificity	.8877	.8880
Biclass B'	MCC	.8692	.8692
	Accuracy	.9844	.9844
	Sensitivity	.9844	.9844
	Specificity	.9844	.9844
Multiclass	MCC	.7122	.7159
	Accuracy	.8342	.8320
	Sensitivity	.8342	.8320
	Specificity	.9447	.9401

Table 7: Confusion matrix for the biclass scenario B. In parenthesis, the rate of success by row.

True label	Predicted label		Total
	0	1	
0	3563 (80.89%)	842 (19.11%)	4405
1	721 (7.57%)	8806 (92.43%)	9527
Total	4284	9648	13932

Table 8: Confusion matrix for the biclass scenario B'. In parenthesis, the rate of success by row.

True label	Predicted label		Total
	0	1	
0	775 (85.64%)	130 (14.36%)	905
1	87 (0.67%)	12940 (99.33%)	13027
Total	862	13070	13932

4. Results and Discussion

This section is organized according to the research questions proposed.

Table 9: Confusion matrix for the multiclass scenario. In parenthesis, the rate of success by row.

True label	Predicted label				Total
	1	3	6	9	
1	789 (86.70%)	50 (5.49%)	63 (6.92%)	8 (0.88%)	910
3	35 (1%)	2796 (80%)	460 (13.16%)	204 (5.87%)	3495
6	55 (0.69%)	491 (6.13%)	7231 (90.29%)	232 (2.90%)	8009
9	7 (0.46%)	290 (19.1%)	445 (29.31%)	776 (51.11%)	1518
Total	886	3627	8199	1220	13932

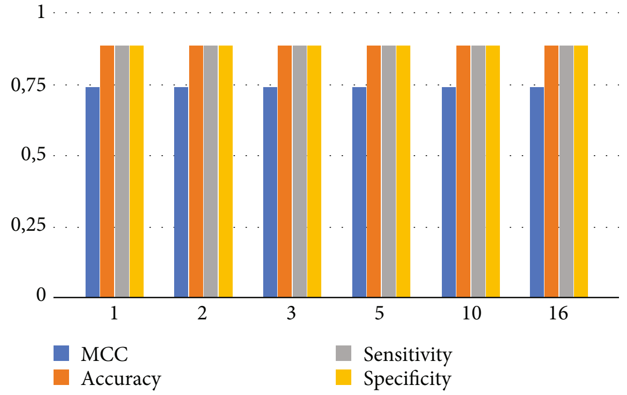

Figure 1: Average results of the response variables with different cores in the biclass scenario B. Axis X: number of cores. Axis Y: values of the response variable.

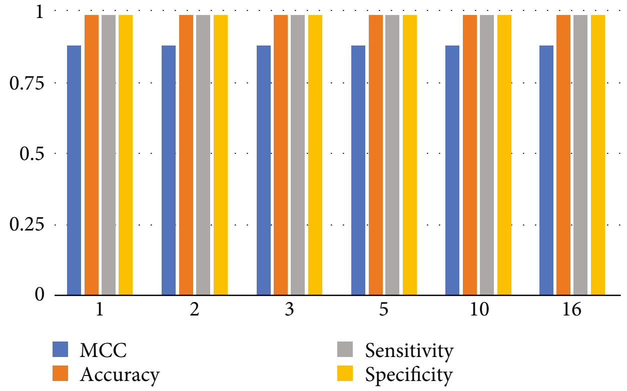

Figure 2: Average results of the response variables with different cores in the biclass scenario B'. Axis X: number of cores. Axis Y: values of the response variable.

4.1. *RQ1: Study of the Concept Drift.*

Recall that we set $D_t = D_1$ and $D_{t+\Delta t} = D_2$ and compute the concept drift measures $\sigma_{t,t+\Delta t}$ and $\sigma^{Y|F_i}_{t,t+\Delta t}$. The average results are $\sigma_{t,t+\Delta t} = 0.3178 \pm 0.0427$ and $\sigma^{Y|F_i}_{t,t+\Delta t} = 0.1589 \pm 0.0213$. Although the values are not high, we need to study whether this drift affects the ML models' performance to predict the severity. That is, what is the result if we remain the ML model obtained with D_t and apply it over D_{t+1}.

4.2. *RQ2: Do the Results Obtained from Applying the ML Models Show any Degradation when Geolocation Features Change?*

In Table 2, we have included the results of the ML models constructed when they are applied over t and $t+\Delta t$.

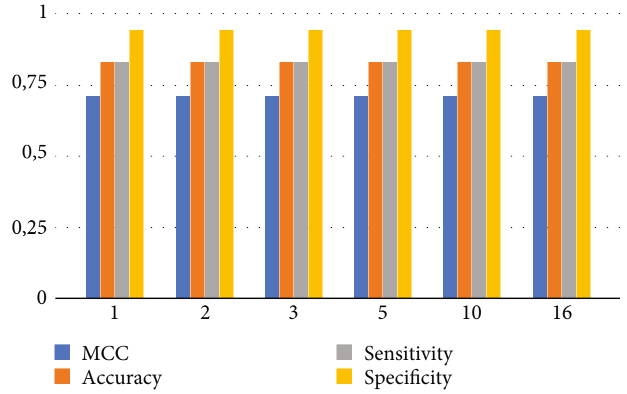

Figure 3: Average results of the response variables with different cores in the multiclass scenario. Axis X: number of cores. Axis Y: values of the response variable.

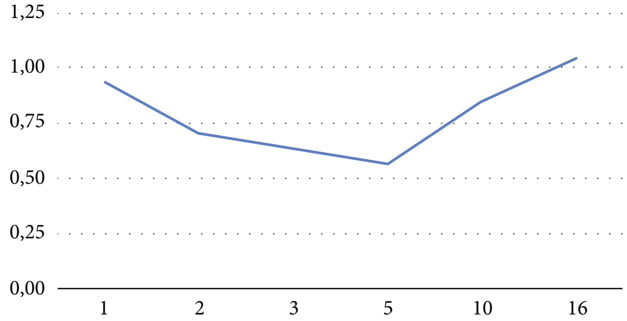

Figure 4: Average total time depending on the cores used in the biclass scenario B. Axis X: number of cores. Axis Y: time in seconds.

Regarding the confusion matrices, they are included in Tables 3, 4, and 5.

We can see in Table 6 the average and the median results of applying ML-constructed models with $D_{t+\Delta t}$ predicting over $D_{t+\Delta t}$.

Regarding the confusion matrices, they are included in Tables 7, 8, and 9.

4.3. *RQ3: Is HPC a Suitable Tool to Face the Creation and Application of ML Models over Cybersecurity Datasets with Concept Drift?*

If we analyze the above results but performed with a different number of cores with the parallelization in the algorithm XGB, see Figures 1, 2, and 3, as it was to be expected, the results about the MCC, accuracy, sensitivity,

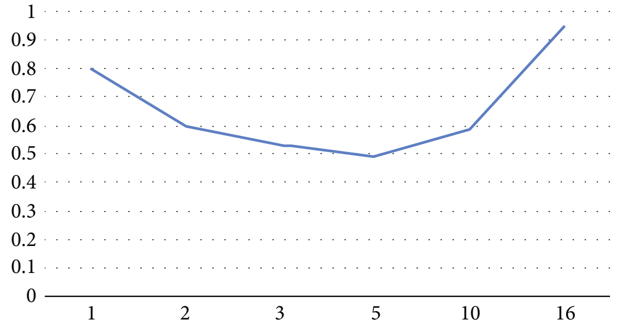

Figure 5: Average total time depending on the cores used in the biclass scenario B'. Axis X: number of cores. Axis Y: time in seconds.

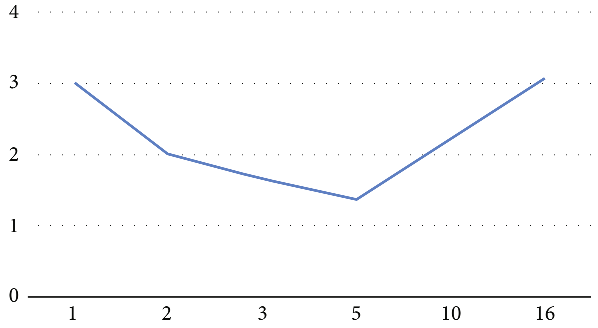

Figure 6: Average total time depending on the cores used in the multiclass scenario. Axis X: number of cores. Axis Y: time in seconds.

and specificity are similar. They are better for the biclass scenario. As expected, the settings obtained with a different number of cores are very similar. So, it seems logical to study the temporal complexity of the process. Thus, we will be able to analyze whether the increase in the number of cores and the use of HPC resources provides us with a considerable reduction necessary to work in conceptual drift scenarios.

The overall running time (in seconds) of the biclass and multiclass scenarios is included in Figures 4, 5, and 6. We can observe that introducing a greater number of cores provides less consumed time. However, the gain is limited to 5 cores. From this, the asymptotic behavior of the parallelization process begins to lose time. The boosting algorithm depends on the results of past iterations, so the parallelization model used in XGB does not create several trees in parallel but produces several different candidate splits that are integrated into a single tree in each iteration. Synchronizing the splits incurs additional costs, so adding too many parallel processes increases the time spent in synchronization relative to the computation of the tree model. Using five cores instead of 1 gives us a gain of 39.50% in the binarized case and 53.82% in the multiclass scenario. Then, parallelizing the construction model, we reduce the time in half.

In Figures 7, 8, and 9, we have included the distribution of the running time (in seconds) of the biclass and multiclass scenarios. Still parallelized, what is taking the most time is the construction of the model.

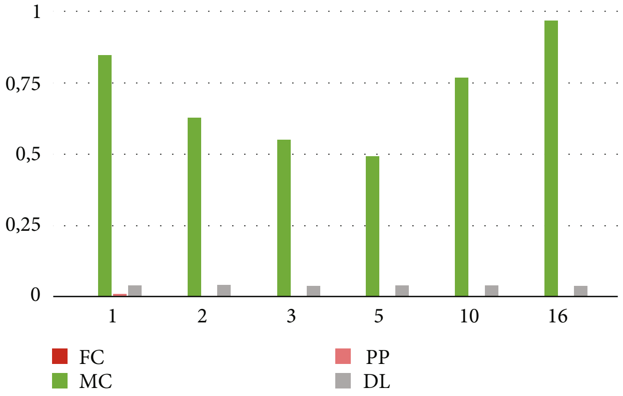

Figure 7: Distribution time depending on the cores used in the biclass scenario B. Axis X: number of cores. Axis Y: time in seconds. FC: feature construction; MC: model construction; PP: data preprocessing; DL: data load.

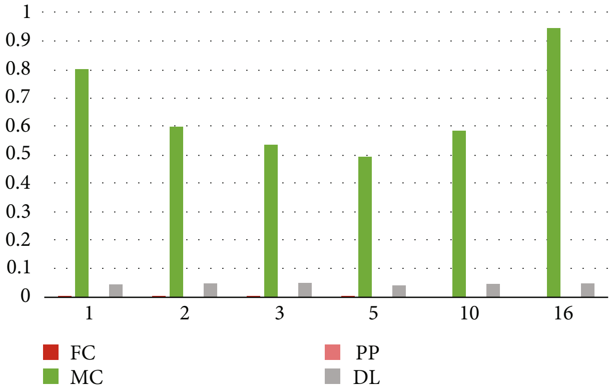

Figure 8: Distribution time depending on the cores used in the biclass scenario B'. Axis X: number of cores. Axis Y: time in seconds. FC: feature construction; MC: model construction; PP: data preprocessing; DL: data load.

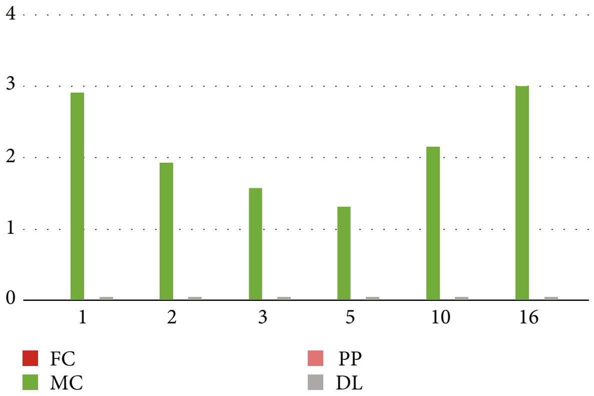

Figure 9: Distribution time depending on the cores used in the multiclass scenario. Axis X: number of cores. Axis Y: time in seconds. FC: feature construction; MC: model construction; PP: data preprocessing; DL: data load.

5. Conclusions

We propose concrete experiments involving dynamic cybersecurity datasets to optimize the categorization of the maliciousness of an IP address by ML models and geolocation information. Also, we study whether concept drift degrades the obtained models. Furthermore, we want to know if HPC would improve our results and performances since cybersecurity datasets are always massive. Accurate boosting ML models are studied, showing that the optimum number of cores is around 5 for the analyzed dataset.

In future work, we plan to relate this optimum to the adequate size of datasets and the type of ML models.

Data Availability

The datasets generated during and/or analyzed during the current study are available in a Github repository, https: 208//github.com/amunc/IP_datasets.

Conflicts of Interest

The authors have no competing interests to declare that are relevant to the content of this article.

Acknowledgments

Partial support was received from the Spanish National Cybersecurity Institute (INCIBE) under the contract art (83, 203 key: X54)

References

[1] N. DeCastro-García, D. E. García, and M. V. Carriegos, "A mathematical analysis about the geo-temporal characterization of the multi-class maliciousness of an IP address," *Wireless Networks*, vol. 2022, 2022.

[2] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964–994, 2016.

[3] J. Gama, I. Žliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, 2014.

[4] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: a review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019.

[5] D. E. García, N. DeCastro-García, and A. L. M. Castañeda, "An effectiveness analysis of transfer learning for the concept drift problem in malware detection," *Expert Systems with Applications*, vol. 212, article 118724, 2023.

[6] C. M. Pancake, "What computational scientists and engineers should know about parallelism and performance," *Computer Applications in Engineering Education*, vol. 4, no. 2, pp. 145–160, 1996.

[7] Microsoft, "Security update severity rating system," 2007, https://www.microsoft.com/en-us/msrc/security-updateseverity-rating-system.

[8] Forum of Incident Response and Security Teams (FIRST), "Common vulnerability scoring system," https://www.first.org/cvss/calculator/3.0.

[9] OWASP Foundation, "Owasp testing guide v4: Owasp risk rating methodology," https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology.

[10] Cybersecurity and Infrastructure Security Agency (CISA), "Nciss cyber incident scoring system," https://www.us-cert.gov/NCCIC-Cyber-Incident-Scoring-System.

[11] N. DeCastro-García, Á. L. Muñoz Castañeda, and M. Fernández-Rodríguez, "Machine learning for automatic assignment of the severity of cybersecurity events," *Computational and Mathematical Methods*, vol. 2, no. 1, article e1072, 2020.

[12] "Firehol - linux firewalling and traffic shaping for humans," https://firehol.org/.

[13] Y. Liu, J. Zhang, A. Sarabi, M. Liu, M. Karir, and M. Bailey, "Predicting cyber security incidents using feature-based characterization of network-level malicious activities," in *Proceedings of the 2015 ACM international workshop on international workshop on security and privacy analytics, IWSPA '15*, pp. 3–9, New York, NY, USA, 2015.

[14] D. Likhomanov and V. Poliukh, "Predicting malicious hosts by blacklisted ipv 4 address density estimation," in *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, pp. 102–109, Kyiv, Ukraine, 2020.

[15] B. Coskun, "(un) wisdom of crowds: accurately spotting malicious ip clusters using not-so-accurate ip blacklists," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1406–1417, 2017.

[16] "IPQualityScore," https://www.ipqualityscore.com/.

[17] J. L. Lewis, G. F. Tambaliuc, H. S. Narman, and W. S. Yoo, "Ip reputation analysis of public databases and machine learning techniques," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, pp. 181–186, Big Island, HI, USA, 2020.

[18] A. Renjan, K. P. Joshi, S. N. Narayanan, and A. Joshi, "Dabr: dynamic attribute based reputation scoring for malicious ip address detection," in *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pp. 64–69, Miami, FL, USA, 2018.

[19] H. Sainani, J. M. Namayanja, G. Sharma, V. Misal, and V. P. Janeja, "IP reputation scoring with geo-contextual feature augmentation," *Information Systems*, vol. 11, no. 4, pp. 1–29, 2020, [Online]. Available:.

[20] Y. Huang, J. Negrete, A. Wosotowsky et al., "Detect malicious ip addresses using cross protocol analysis," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 664–672, Xiamen, China, 2019.

[21] N. Usman, S. Usman, F. Khan et al., "Intelligent dynamic malware detection using machine learning in ip reputation for forensics data analytics," *Future Generation Computer Systems*, vol. 118, pp. 124–141, 2021.

[22] D. Jeon and B. Tak, "Blackeye: automatic ip blacklisting using machine learning from security logs," *Wireless Networks*, vol. 28, no. 2, pp. 937–948, 2022.

[23] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc, 2015, https://proceedings.neurips.cc/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf.

[24] G. I. Webb, L. K. Lee, B. Goethals, and F. Petitjean, "Analyzing concept drift and shift from sample data," *Data Mining and Knowledge Discovery*, vol. 32, no. 5, pp. 1179–1199, 2018.

[25] T. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 785–794, New York, NY, USA, 2016.

[26] XGBoost contributors, "XGBoost: extreme gradient boosting," 2022, https://github.com/dmlc/xgboost.

[27] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.