# No Bot Allowed: Detection of Automated Traffic on Modern Booking Platforms. Methods, Insights, and Current Challenges.

Umberto Fontana[1,2], Elisa Chiapponi[2], Claudio Costanza[2], Vincent Rigal[2], Olivier Thonnard[2], Martynas Buozis[2], and Hervé Debar[1]

[1] SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, France
[2] Amadeus IT Group, France / Germany
{umberto.fontana, herve.debar}@telecom-sudparis.eu
{elisa.chiapponi, claudio.costanza, vincent.rigal, olivier.thonnard, martynas.buozis}@amadeus.com

**Abstract.** Web robots are autonomous agents that in recent years have been increasingly employed for conducting fraudulent activities across various industries. This research addresses the challenge of detecting scraping activities with operational data collected from various e-commerce booking platforms. The study employs a supervised learning approach, designing a set of general and domain-specific features and characteristics for each user session, which could potentially adapt to other scenarios. To overcome the challenge related to the limited ground truth, we developed an heuristic method to automatically detect user sessions suspected of automation, by identifying unusual traffic sources. The labeling method is carried out with particular care for data quality, using the Vendi score metric to reduce potential mislabeling errors. Moreover, this work presents new insights into modern evasive automation targeting API endpoints and reflects on current challenges in existing academic literature and industry regarding existing bot detection methodologies.

**Keywords:** Web Scraping, Data Labeling, Bot Detection, Detection Evasion

## 1 Introduction

Google's popularity since its launch in 1996 can be attributed to two core elements: the PageRank algorithm and *bots*. The bots deployed by Google, known as crawlers, play the crucial role of systematically indexing web pages to create a connected map of the Internet. Today, automated (bot-generated) Internet traffic exceeds human activity, with malicious bots outnumbering benign ones[1].

With the proliferation of Bot-as-a-Service providers and the developments of generative AI technology, crafting bots has become accessible even to users without technical expertise. This increasing trend grows concerns over the emergence of new functional abuse frauds targeting modern applications, which are causing significant financial losses in the industry[8].

Web scraping involves the usage of bots to automatically extract data from Internet websites, sometimes driven by illicit purposes such as illegal data resale or reverse engineering of applications[32]. The acceptability of scraping and its legal status are rather unclear. Specifically, in the travel and hospitality sectors, scrapers significantly distort key performance indexes such as the look-to-book ratio, which is a critical metric measuring how effectively websites convert visitors into paying customers.

The evolution of the architecture of modern web applications consequently changed bot behavior patterns. With web development increasingly relying on Application Programming Interfaces (APIs) to facilitate interoperability between services, threat actors have adapted by strategically exploiting API endpoints for data exfiltration.

This work focuses on the study of scrapers specifically targeting the travel industry, leveraging data collected from different booking platforms targeted by bots to extract pricing and availability information about airlines' destinations. To address the challenge of limited labeled data, we extend the available ground truth using a methodology that relies on heuristic analysis of bot IP address patterns. Next, we apply machine learning techniques for the detection of bots, while providing useful insights into modern evasive scrapers. Our contributions can be summarized as follows:

- We develop an adaptable scraping detection pipeline suitable for API-dependent applications, incorporating both domain specific features and known measures from existing literature.
- We define a heuristic-based approach for automated labeling of bot traffic and evaluate its impact on dataset quality through a diversity metric, with the goal of reducing mislabeling errors.
- We present novel findings about bot traffic patterns in e-commerce websites, along with analysis of potential limitations in current detection methodologies.

The remainder of this paper is organized as follows. Section 2 introduces the fundamental concepts required by the context, such as the application environment, diversity, and key challenges in bot detection. Section 3 reviews the relevant literature and related research work. Section 4 presents the characteristics of the raw dataset. Section 5 describes our methodology, providing a detailed description of session-based features used for the classification and the extended labeling approach. Section 6 describes the experiments and analyses the results. Finally, in Section 7, we synthesize our findings and we outline potential directions for future work.

## 2 Background and Motivations

### 2.1 Environment

The environments under consideration are e-commerce websites designed according to the principles of the Representational State Transfer (REST) architecture.

These principles include uniform representation of resources, statelessness, strong hierarchy, and data cacheability [11]. Communication between client and server takes place via the HTTP protocol, where the client issues a request to the server, which processes the request and returns a response.

The main purpose of these websites is to book flight trips. Following the principle of uniform representation, the websites' frontend communicates with the backend via a set of APIs that abstract the hidden layers, enhancing the application's flexibility and portability. This standardized architecture design ensures that in all platforms, a user follows a consistent sequence of actions to complete the reservations. In a first step, users start the navigation by searching for available options, then proceed to enter passenger details and select additional services through cart management functionalities. Finally, they review and modify their order before processing payment and confirming the purchase. These high-level activities are carried on through multiple interconnected API operations, many of which depend on one-another. For instance, when a user submits a search query specifying departure location, time, destination, and trip type, the system under study executes two coordinated API calls: the `air-bounds` API retrieves all available offers for the specified date, while simultaneously the `air-calendars` API retrieve optimal pricing information for surrounding dates to provide users with flexible booking options. APIs are characterized by a unique path and an HTTP method, i.e. GET, POST, DELETE, and PATCH. Different websites share the same APIs set, that will target different backend resources depending on the specific application.

The RESTful architecture requires that every request to the server should contain all information to allow the server to process and complete them. This design principle ensures that the server maintains no memory of previous interactions (statelessness), requiring alternative methods for managing user session data across multiple requests. When a user initiates a new session on the website, our environment automatically generates a unique hash string serving as the session identifier. This enables precise tracking of user actions, ensuring with high confidence that all the activities are associated with the same individual.

Every web application under consideration operates behind a security firewall that detects and mitigates potential threats. This defensive infrastructure also incorporates an anti-bot module that evaluates incoming traffic using proprietary methods, effectively filtering requests before they reach the API Gateway that directs the call to the right service. The environment schema is explained in Figure 1.

### 2.2 Diversity

Diversity is a strong concept in machine learning, often set as a goal when creating datasets that reflect real-world scenarios [24], [28], [12]. We can define diversity as the variety and heterogeneity of the data along the different dimensions.

Recently, researchers proposed a domain-agnostic metric that is capable of measuring data diversity inspired from ecology and quantum mechanics, named

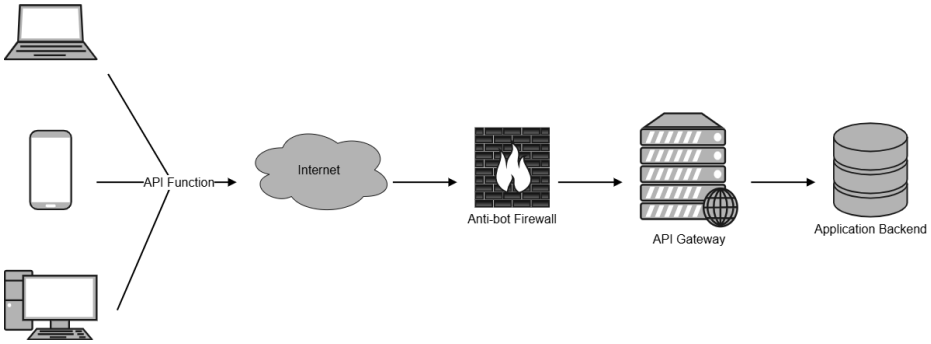



Fig. 1: Diagram of the environment under study

the Vendi score [13], [26]. The score serves as a quantitative measure of the number of dissimilar elements within a data set or sample. The metric produces higher values when the elements in the sample exhibit higher variation across the different dimensions.

The Vendi score is computed by taking the exponential of the Shannon entropy derived from eigenvalues of a matrix of which elements are pairwise similarity scores between the data samples. Specifically, given a set of $n$ samples, given a positive semi-definite similarity matrix $K \in \mathbb{R}^{n \times n}$, being $\lambda_1, ..., \lambda_n$ the eigenvalues of the $K/n$ matrix, the Vendi score is computed as:

$$VS(K) = exp\left(\frac{1}{1-q} log \sum_{i=1}^{n} (\lambda_i)^q\right) \tag{1}$$

The parameter $q$ acts as a sensitivity factor that addresses imbalances in the dataset. When $q \in \{0, 1, \infty\}$, the score is obtained by taking the limit of Eq. 1 as $q$ approaches the respective value. The similarity matrix $K$ is determined by the chosen similarity function. Common options are cosine similarity and the Radial Basis Function (RBF) kernel. In [25], authors proved that when using infinite-dimensional kernels, the Vendi score calculated from a limited sample batch fails to accurately represent the entire population. Therefore, from this point on, all Vendi score calculations will utilize cosine similarity for constructing the elements of the matrix $K$.

In section 6.1, we use the Vendi score to continuously evaluate the dataset's quality as new labels are introduced. The fundamental assumption in behavior-based bot detection is that automated agents exhibit similar behavioral patterns. It is logical at this point that the diversity score for the bot class should remain stable when new labeled entries are added. For this purpose, diversity serves as an effective measure of class purity and can be used to assess eventual labels mistake.

### 2.3 Goals and Challenges

In this work, we collaborate with a leading IT provider company for the travel industry, from which we collect real traffic API log data from different airline booking websites. The development of a bot detection solution faces several challenges deriving from the evasive nature of bots, that also poses some limits in the data labeling. We provide a list of the main difficulties encountered in the study of bot traffic.

**Partial labeling issue** The first requirement of supervised learning is to have a labeled dataset. Problems arise when the data volume increases and the labeling process becomes more expensive and time-consuming. Every day, there are a large number of requests that target airline websites, and manual labeling is not feasible for the environment considered. Some previous work used IP blacklists, or known bot browsers as labeling criteria, but, as also remarked in [31], evasive bots are aware of these blacklists and they try to avoid well-known traffic origin.

**Evasive actions from sophisticated bots** A second challenge comes from the evasive nature of automation. As mentioned in [6], [20], [34], bot operators often detect the parameters used to identify their bots, and modify their settings to bypass defenses and resume their activities. For this reason, systems that heavily rely on static rules (e.g. rules on fingerprinting features, or IP blacklisting) encounter difficulties in adapting to the evolving and dynamic nature characteristics of malicious traffic.

**Heterogenous bot behaviors** The emergence of Bot-as-a-Service platforms and Generative AI tools has made bot creation increasingly accessible. Although bots generated from the same script typically exhibit comparable browsing behaviors, the growing diversity of bot types presents a substantial obstacle for supervised machine learning approaches. In the supervised framework, a model trained to detect one category of bots may fail to identify bots created with different behavioral patterns, posing a limitation for the development of a generalizable solution.

Keeping these challenges in mind, the main goal of this research is to provide some insights about real malicious scraping traffic targeting the travel industry and develop a comprehensive framework for labeling scraping behavior using diversity as a quality metric. This is an alternative method of labeling data that do not rely on existing blacklists or fingerprinting features, and study the impact of data quality measures on the classification of supervised classifiers.

## 3 Related Work

The challenge of bot detection focuses on identifying automated agents through the analysis of the traffic inside the application to protect. Many commercial anti-bot solutions employ signature-based detection methods [3], which employs static rules that rely on device fingerprints or IP address reputation. These

approaches have proven to be weak against evasive bots capable to adapt their parameters once they identified the blocking criteria. To do so, they modify their fingerprint by mimicking characteristics typical of legitimate browsers [18]. Prior research in [10], [29], [31] has also investigated in the direction of detecting bots through additional, absent, or contradictory values in the fingerprint, but these solutions are usually complex and difficult to maintain up-to-date with the evolution of inconsistencies and fingerprinting techniques.

Another widely used detection strategy is through CAPTCHAs, which propose challenges designed for humans only. However, recent developments in AI have enabled bots to automatically solve these puzzles including both visual puzzles [16] and more sophisticated ones, like in the case of Google reCAPTCHA v3 [2]. Additionally, modern bots frequently use CAPTCHA solving services that employ cheap human labor to manually complete challenges for monetary reward, effectively bypassing the system[23], [6].

An alternative approach to bot detection uses machine learning algorithm to identify automated behavior by analyzing navigational patterns. The typical methodology employed for the automated detection of bot sessions using web logs involves the extraction of HTTP request logs from web server. Next, a session is extracted from the IP address, eventually combined with the User Agent, and split after an inactivity timeout of 30 minutes. For each session is then generated a feature vector to train machine learning models. As previously discussed (Section 2.3), machine learning methods are powerful but they often need labeled data for supervised learning training and model performance evaluation. Previous research relied on indicators such as whether or not a bot accessed trap file, or requested the `robot.txt` file within the session for the labeling [35], [21]. Others have employed IP blacklists or identified bot user agents as classification criteria using external tools, as in the case of [4], [5], [27]. While these labeling strategies are useful alternatives when other tools are unavailable, bots that request access to the `robot.txt` file or are recognizable from the user agent string probably don't put an effort in hiding their identity, and consequently also their automated behavior, unlike evasive bots. Moreover, many bot operators are aware of blacklisted IPs and they can circumvent this protection by using services from Residential IP Proxy (RESIP) providers [22], [7]. In [19], the authors labeled the data by combining firewall rules and CAPTCHA challenge feedback on IP sessions. To address the persistent issue of insufficient labeled data, they also developed an architecture designed to generate data, with a strong focus on outliers to balance the dataset, obtaining a labeled dataset for each IP session. While they used a generative model to sample new data, the study did not focus on the initial quality of the dataset. Instead, they opted to generate new entries rather than find a method to accurately label potential false negatives.

## 4 Dataset

The source data consists of operational logs collected from different travel booking platforms implemented using RESTful principles, as described in Section 2.1.

Log entries are collected from three different platforms, each of which constitutes a domain (domain name). All domains are affected by scraping traffic. Logs information include details about the issued API path, unique to each API, along with the HTTP method used, the associated set of HTTP header parameters, and, when applicable, the user-provided input data in JSON format.

Traffic across all three domains passes through an initial anti-bot protection system that uses static rules and fingerprinting techniques to identify automation. When one of these protections is activated, the traffic that triggers it can be tagged rather than blocked, allowing us to establish reliable ground truth labels by looking up these tags in the header parameters. If any request within a session, identifiable by a unique hash string across multiple logs, receives a bot tag, we classify the entire session as anomalous (positive label). The anti-bot detection rules are intentionally strict to minimize false positives cases, where legitimate users are incorrectly flagged as bots. However, this strict approach results in a high amount of false negatives, where actual bot traffic goes undetected, which negatively impact the overall dataset quality. Moreover, this labeling capability is available for only two of the three domains, raising some difficulties for the evaluation step.

Behavior-based approaches require establishing a minimum threshold for the number of actions needed to create reliable user profiles. This threshold varies depending on the data source and how user sessions are defined. For instance, in [19], authors used solely the IP addresses to identify sessions and required at least 30 logged actions per session. The work [30], instead, used the more common approach of combining User Agent strings, IP addresses, and 30-minute timeouts to define sessions, filtering out any with fewer than 7 actions. Our methodology differs from the other works in several ways. We use session IDs provided by the application, that provide a more precise session identification than the methods described above. Additionally, our logs capture only API calls rather than detailed interactions, excluding UI-related requests like static content retrieval. Given these characteristics of our dataset, we empirically determined that sessions with fewer than $N = 4$ actions should be excluded from analysis. While this threshold requirement might appear to be a limitation of behavioral-based methods, we argue that it actually serves our scope of scraping detection. If a bot attempts to collect the same amount of information while making fewer requests per session, it must compensate by creating a larger amount of sessions. This behavioral shift moves the detection challenge away from scraper identification toward abnormal traffic pattern recognition, similar to approaches used for Denial-of-Service (DoS) attack detection.

Table 1 contains summary information about the dataset. For each of the three domains, we collect data on consecutive days, reaching a total of more than 4.4 million logs, aggregated in ∼480k sessions, of which less than 12k are labeled as malicious by firewall rules.

The high amount of mislabeled sessions presents a significant challenge for evaluating model results and analyzing scraper behavior. In the following section,

| Dataset | #Days | #Logs | #Unique Sessions | #Positive Labels | %Positive Labels |
|---|---|---|---|---|---|
| D1 | 2 | 399,829 | 44,449 | 7,058 | 15.88% |
| D2 | 1 | 159,7209 | 109,233 | 4,118 | 3.77% |
| D3 | 2 | 249,8044 | 332,253 | - | - |
| Total | 5 | 4,495,082 | 485,935 | 11,176 | 2.30% |

Table 1: Dataset Summary

we will describe a new labeling approach we developed to enhance dataset quality and reduce mislabeled entries.

## 5 Method

Figure 2 describes the processing steps of our method. The *Log Preprocessing* step acquires raw API logs and filters out short sessions based on the criteria outlined in Section 4. The *Features Extraction* step (Section 2) creates a feature set for the classification task. These features are measurable statistics created to capture user behavior patterns within the booking platforms, described in Table 2. The *Session Labelling* step (Section 5.2) uses external information to add labels to the data. The *Model Training* and *Classification* steps are then applied to obtain the results shown in Section 6.

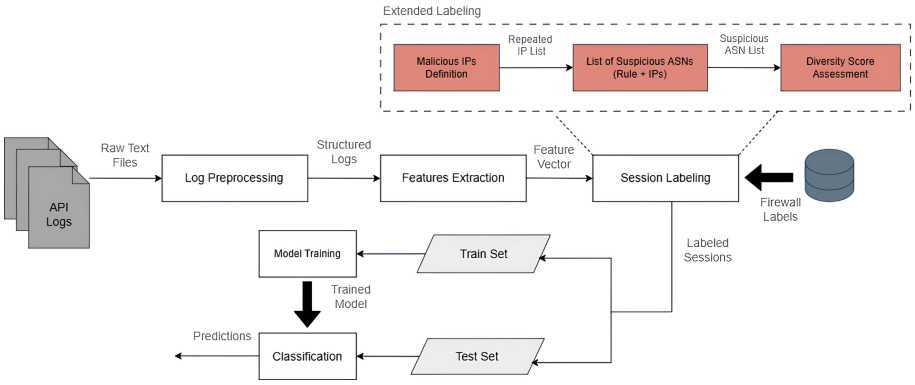



Fig. 2: Flowchart of the proposed method

### 5.1 Features Creation

Inspired by established research on web bot detection using web logs [14], [17], [27], we developed a set of 8 features, detailed in Table 2, with a short description and the feature category they belong to. Features are segmented in 3 categories,

| Feature Category | Feature Name | Feature Description |
|---|---|---|
| Session Volume and Timing | Session volume | Total number of API calls issued by the client |
| | Browsing speed | The ratio of the total number of API calls over time (in seconds) |
| | Inference time min | Minimum time between successive requests |
| | Inference time std | Standard deviation of time between successive requests |
| Content-based features | Bound-calendar ratio | The number of `air-bounds` calls over the total number of call for `air-calendars` |
| | Total search APIs | Total number of API calls of search type |
| | Total cart management APIs | Total number of API calls of cart management type |
| Navigational Patterns | Repeated APIs std | Standard deviation of the requests per APIs in the session |

Table 2: Features created for each session

*volume and timing* reflecting the amount of information retrieved, *content* reflecting the semantics of interactions, and *navigational patterns* reflecting the profile of the session.

Unlike previous studies that use HTTP log data, API interactions with the backend servers do not log certain details about user navigation, such as the set of static files accessed. For this reason, we could not create the exact same set of features proposed in state-of-the-art research, but we develop features that share similar meanings.

An example is the key feature HTML-to-image ratio, which compares HTML content requests to image requests. Scrapers typically focus on information-rich portions of websites, ignoring auxiliary content. This feature, combined with the percentage of requests for CSS and JavaScript files within a request, proves to be an effective measure for scraping detection. In our environment, this information is abstracted within the API calls and is not available. However, the scrapers are visible when they request the `air-bound` API endpoint (mentioned in section 2.1) without the associated `air-calendar`. Our bound-calendar ratio serves a similar purpose to the HTML-to-image ratio, measuring how directly a user access core website data while ignoring non-essential content.

### 5.2 Bot Analysis and Extended Labeling

To gain a clearer understanding of the method used to extend the labeling, we first share some early findings about scraping traffic upon examining the data traffic from the three available domains.

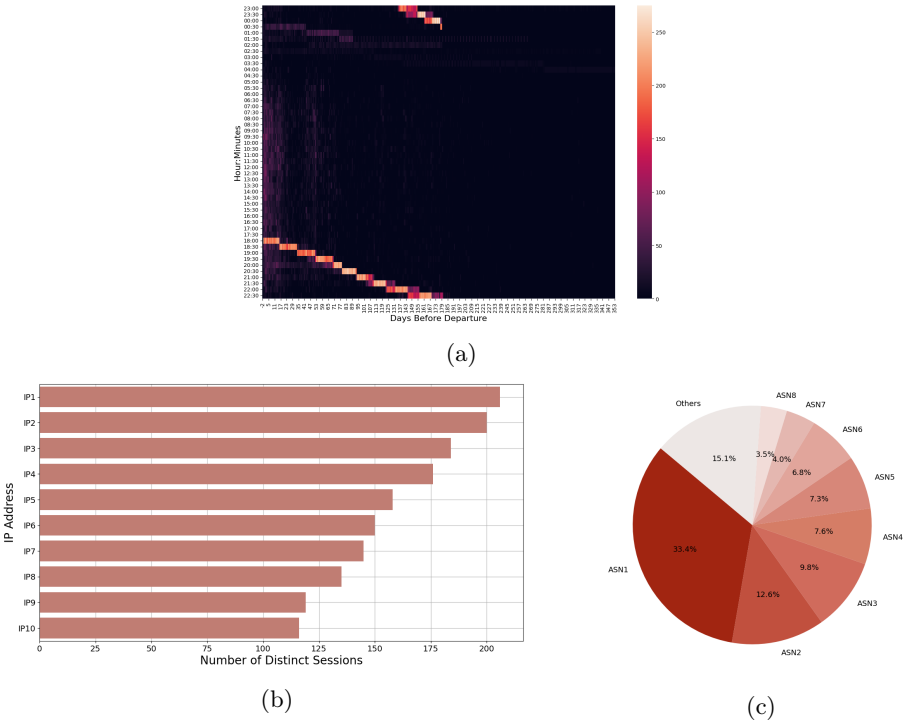



Fig. 3: (a) Heatmap of the traffic accessing offers for flight departing in a number of days (x-axis) in a specific time frame (y-axis). (b) Plot of the top-10 IPs addresses (y-axis) that are repeated across distinct sessions, ordered by frequency of appearence (x-axis). (c) Pie chart illustrating the distribution of malicious traffic across different ASNs. The first eight slices represent traffic from the top 8 ASNs, while the remaining slice collectively account for traffic from nearly 40 other ASNs.

**Bots collect data in an orderly manner** . Automated traffic is recognizable by looking at certain navigational patterns that repeat themselves across multiple sessions (Figure 3). One of these pattern is related with the content retrieval pattern. After extracting the query content from the log payload, it was possible to see that the data collection of scrapers is done with a specific criteria. In Image 3a, we display the amount of traffic in a representative day for flight searches, with brighter areas indicating higher search volume in 30-minutes intervals (y-axis) for flights departing within the next x-days (x-axis). To optimize information retrieval, scrapers perform systematic queries, which is evident in the night hours. This organized pattern contrasts with the more "noisy" behavior of human users during daytime hours. Sometimes, bots tend to generate a high volume of traffic in particular pages (brighter color), while other times their activity shows less noticeable patterns. One

possible explanation for this phenomenon is that scrapers often verify data consistency across different time frames through a fast check on previous scraped pages.

**Bots rotate IPs from a finite set of addresses** . Analysis of session IP addresses revealed that some bots repeatedly use a limited set of IPs across multiple sessions within short timeframes. Figure 3b displays a frequency distribution of IP addresses (y-axis) that appear most frequently across multiple sessions (x-axis) for traffic labeled as positive. Upon examining the Autonomous System Number (ASN) associated with this traffic, we found an even greater concentration, where most of malicious traffic originated from a very small set of AS, as shown in Figure 3c, with several ASNs serving as sources exclusively for malicious activity.

Having these characteristics in mind, we leveraged them to extend the labeling of the dataset. If we consider human traffic as background noise in Figure 3a, we can theoretically determine the minimum frequency threshold for an IP to be labeled as suspicious. By isolating traffic from frequent IPs and calculating the proportion of human traffic as background noise, calculated as in [9], we identify the optimal filtering threshold. The point where noise levels reach their minimum indicates that most human traffic has been filtered out, leaving predominantly automated traffic. Specifically, by selecting only the traffic generated by IP addresses that appear in at least $k$ different sessions within a day, where $k \in 1, 2, ..., M$ and $M \in \mathbb{N}$, we derive a noise function $S(k)$ that varies with the parameter $k$. An example of this noise evolution is shown in Figure 4. In this example, the noise function's decline has an inflection point around $k = 15$ on the x-axis. However, we can consider $k \approx 20$ as a reliable threshold to identify suspicious IPs and minimize potential false positives.

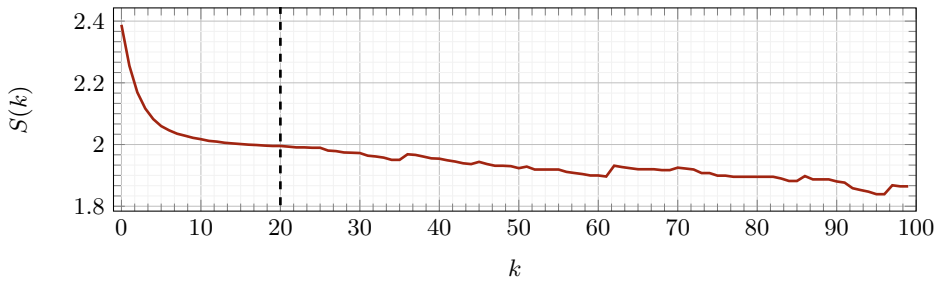



Fig. 4: Evolution of the image noise by removing all the sessions of which IP is repeated less than a certain values indicated in the x-axis.

In the second step, we list all the ASNs associated with malicious behavior. This classification is based on the two criteria already mentioned: either the session was flagged as bot activity by firewall rules, or the IP address appeared

in an abnormal number of different sessions according to the threshold parameter defined above.

Before blacklisting an ASN as malicious, we assess the effect of incorporating its associated traffic into the positive labels while removing it from the negative labels. The underlying assumption is that bot traffic features defined in section 5.1 should exhibit similar values with each other. By adding new positive labels, we expect the diversity score calculated using equation 1 to remain stable and any significant score increase would suggest either the presence of mislabeled legitimate users or the discovery of a new bot variant. This approach enables the labeling of a dataset without depending on external detection tools, such as honeypots as in [17].

## 6 Experiments and Results

This section analyzes the results of the methodology introduced in previous sections. First, in Section 6.1, we study how the new labeling approach affects data diversity score, and how diversity can be used to select which ASNs to blacklist and the limitations of doing so. In Section 6.2 we assess the general performance of the machine learning pipeline, with an attentive analysis of the features and of labeling impact on the classification. In Section 6.3 we discuss how the different bot types across the different domains are classified according to their common and unique characteristics.

### 6.1 Diversity Study

Diversity is computed using the Vendi score (Equation 1), where the elements of the matrix $K$ are pairwise samples cosine similarity score. The parameter $q$ is set to one, and the score is computed as the limit of Eq. 1 as $q \to 1$, yielding:

$$VS(K) = exp\left(-\sum_{i=1}^{n} \lambda_i log \lambda_i\right) \tag{2}$$

The choice of the value of $q$ was led by our uncertainty about the true data imbalance, leading us to maintain the standard equation form to prevent affecting the problem definition. Figure 5 represents the variation in the Vendi score for both positive and negative classes when negative labels are shifted into the positive set by introducing new ASNs into the blacklist for the three domains. The ASNs are chosen in decreasing order of malicious traffic volume in terms of IP repetition and firewall labeling.

**First Domain Diversity.** Under initial conditions with firewall rules in place, Dataset D1 exhibits a high proportion of misclassified entries. Analysis of the labeled traffic reveals that the website was likely targeted by a single bot variant, with automated traffic displaying a consistent behavior across sessions. Shifting negative labels into the positive set had minimal impact on the diversity score of the positive class, as illustrated in Fig. 5a, while

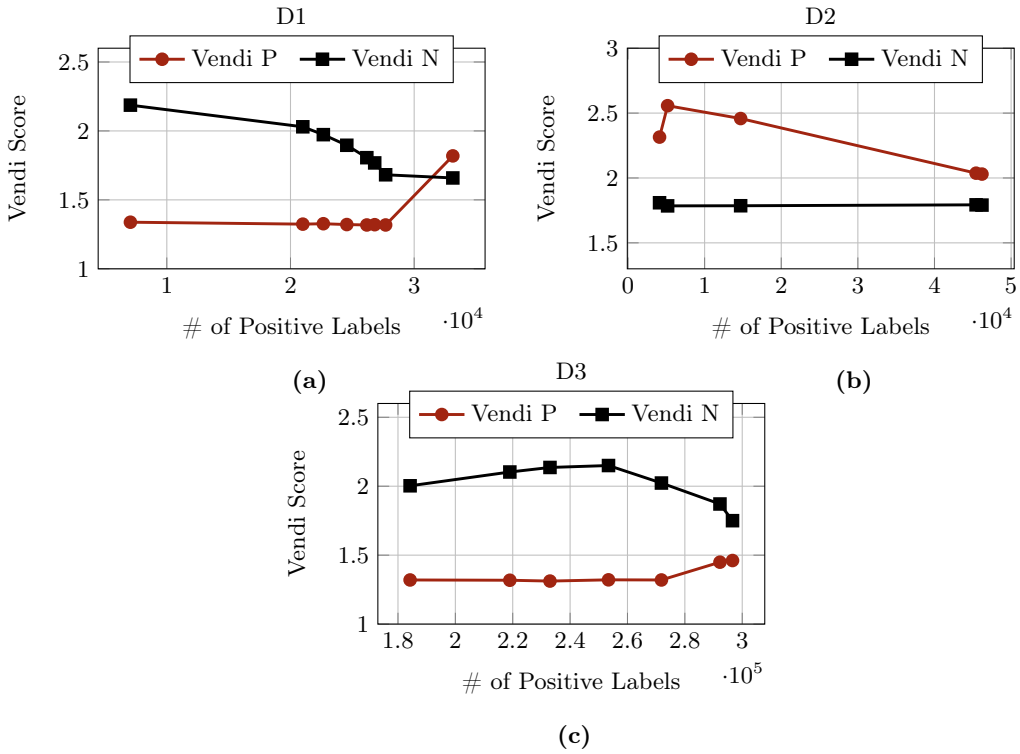



Fig. 5: Diversity score evolution of the three domains by introducing new positive labels shifted from the negative class.

the negative class score actually decrease. The inflection point at the end of the plot results from the high amount of mislabeled data introduced in the positive set. This contamination causes a dramatic spike of the positive class diversity score while leaving negative class diversity almost unchanged. We could therefore identify the initial group of ASNs added to the blacklist before this inflection point as origin sources of malicious traffic.

**Second Domain Diversity.** Analysis on the second domain reveals that the website was targeted by multiple scraper variants. For this reason, using firewall labels alone initially produces a very high diversity score, as shown in Fig. 5b. The diversity score of the positive class changes significantly with the addition of new ASNs to the blacklist, while the negative class score remains stable. This pattern suggests that the ASNs-based labeling introduces many false positives. After further study, we discovered that the malicious traffic primarly originated from IPs associated with the Great Firewall of China (GFW), a sophisticated filtering system that controls the online information flow within China. Since this system functions as a proxy for all the traffic from China [15], [33], our methodology fails to detect malicious sessions

without flagging legitimate traffic as bot activity. Given these findings, we decide to use exclusively anti-bot rule labeling for domain D2.

**Third Domain Diversity.** As mentioned in section 4, we do not have ground truth data from the anti-bot rules on the third domain, and we solely rely on the ASN-based labeling. In Figure 5c, the initial conditions were set by the first blacklisted ASN. Traffic analysis reveals that bot behavior on this domain closely resembles the one observed in D1. In this case, the introduction of new positive labels from the negative class maintains stable diversity score for the positive class up to the inflection point where they start to manifest mislabeling effects. The negative class initially shows increased diversity, reaching a mid-chart peak before declining again. This is probably due to the initial imbalance in favor of bot traffic, which is dominant in the class. As false negative labels are progressively removed, human traffic becomes more prominent, culminating at the peak where approximate balance between the two classes emerges within the negative set. As for D1, we create a list of malicious ASNs for traffic labeling with all the Autonomous systems that did not affect the positive class diversity score.

### 6.2 General Performance and Discussion

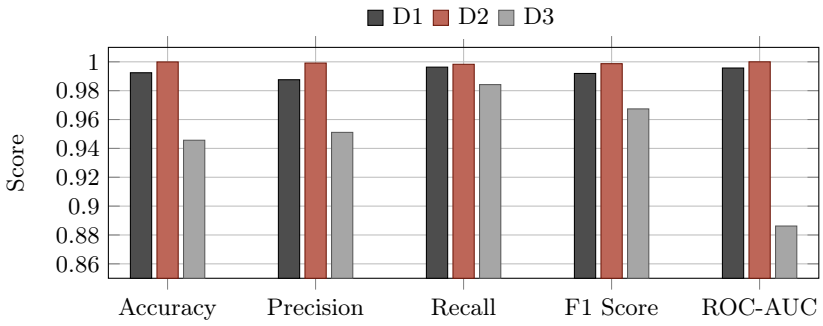



Fig. 6: Performance evaluation of the XGBoost model on the three dataset with the different scores (x-axis).

The classification task was conducted using a set of well-known algorithms often used in literature, i.e. Support Vector Machine (SVM), Random Forest, Multi-layer Perceptron (MLP), Logistic Regression, and Extreme Gradient Boosting (XGBoost). For each model, we select parameters based on the grid search cross-validation method. The evaluation metrics include Accuracy, Precision, Recall, F1-Score, and ROC-AUC Score. Overall, all the models demonstrate similar performance across the three domains, with the MLP showing slightly better results. Since the objectives of this paper do not include machine learning models comparison for the bot detection task, we proceed with the discussion using the XGBoost algorithm as classifier, given its comparable performance to MLP and its explainability features suitable for the task.

Initially, without the extended labeling method, the model performance on domain D1 was deeply affected by data imbalance and mislabeling problems, not reaching a 0.6 in terms of F1 score. The performance increases with the introduction of the new ground truth as showed in Figure 6.

In domain D2 the data are imbalanced, with the majority of session belonging to the negative class. Some of the bots in D2 directly retrieve information using direct calls to the `air-bound` API, regardless of its relationship with `air-calendars`. Moreover, these simple bots also have a high volume of requests within a short time frame, suggesting that their sophistication level is lower than bots preferring a "low and slow" approach to mimic human behavior.

In the third domain, D3, the model's performance is the lowest across all the evaluation metrics. The F1 score, in particular, is significantly impacted by a low precision score, indicating a high number of false positives. Although this can be seen as a model mistake, with attentive analysis we discovered that most of these false positives are actually entries labeled as humans but showing evidence of automation, as we will discuss in Section 6.3. Unlike the other domains, D3 misses an initial ground truth given by the first line anti-bot protection system, which would have enabled more precise identification of sessions undetectable by our ASN-based labeling method.

### 6.3 Features Analysis and Bot Variants

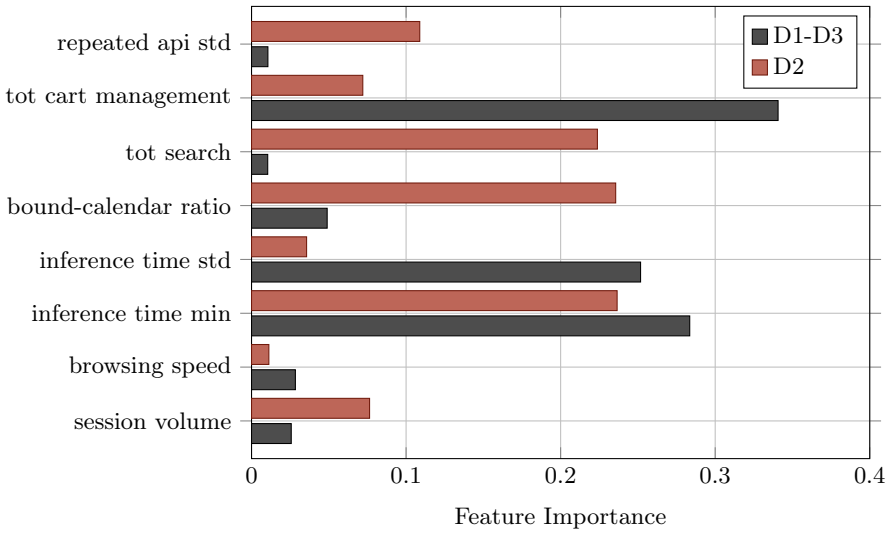



Fig. 7: Feature importance score with XGBoost model trained on the different bot varieties present on the three domains.

All three domains share a bot variant that tries to mimic human behavior by introducing delays in the navigation. One of the most impacting features

in the classification for all three domains is the minimum inference time. The reason behind this high score is the fact that the delay introduced by bots is also present between API calls that should be called together, noticeably increasing the minimum time between subsequent requests within a session. While domain D1 and D3 are targeted exclusively by this bot type, domain D2 includes more variants, changing the importance that the models gives for the different features for each domain (Figure 7). Additionally, all bots consistently avoid accessing cart management pages, giving a hint on their primary objective.

While the model trained in domain D1 and D3 mainly rely on inference time and cart management-related features for the detection, the model trained on the second bot variant in domain D2 appears to increase the importance of other features, such as the standard deviation of API repetition and the volume of API calls of type search. Moreover, on the second domain there is a reduced impact of features highly important to the first model, like the standard deviation of the inference time or the cart management feature.

The bound-calendar feature plays a crucial role in the model trained on D2, while it has lower importance on D1-D3 model. This difference is due from the distinct behavior patterns observed in D2, where numerous bots directly retrieve information from the bound endpoint, regardless of calendars' information, obtaining for these sessions the correspondent feature set to the value of -1. Additionally, these bot type also triggered frequent search operations, which inflates the corresponding feature values beyond those seen in negative samples. For a more detailed discussion, please refer to Appendix A.

## 7 Conclusion and Future Work

In this paper we present a real-world case study of scraper detection on modern e-commerce platforms built on REST API architecture. We develop a heuristic for automated data labeling using IP address repetition across sessions and firewall flagging as anchor points. However, this approach has limitations in scenarios like traffic proxied from behind the Great Firewall of China where there is higher risk of false positives. Next, we extract session-based statistical features for the bot detection task that play different roles depending on the specific bot variants.

This system specifically targets advanced scraping but cannot detect other fraudulent behaviors like Denial-of-Inventory attacks (a.k.a Inventory Hoarding), which require more complex session modeling. Future work could enhance the system by examining lower-level user actions and sequential operation patterns within applications. By combining robust user identification with more sophisticated user modeling, we could extend beyond scraping detection to provide stronger protection against the continuously evolving bot threats impacting the e-retail industry.

### Acknowledgment

*RESCUE Project funded by the European Union – NextGenerationEU. The views and opinions expressed are solely those of the author(s) and do not necessarily reflect the views of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.*

## References


1. 2025 Bad Bot Report. Tech. rep., Imperva a Thales company (2025)
2. Akrout, I., Feriani, A., Akrout, M.: Hacking google recaptcha v3 using reinforcement learning (2019), https://arxiv.org/abs/1903.01003
3. Amin Azad, B., Starov, O., Laperdrix, P., Nikiforakis, N.: Web runner 2049: Evaluating third-party anti-bot services. In: Maurice, C., Bilge, L., Stringhini, G., Neves, N. (eds.) Detection of Intrusions and Malware, and Vulnerability Assessment. pp. 135–159. Springer International Publishing
4. Balla, A., Stassopoulou, A., Dikaiakos, M.D.: Real-time web crawler detection. In: 2011 18th International Conference on Telecommunications. pp. 428–432 (2011). https://doi.org/10.1109/CTS.2011.5898963
5. Cabri, A., Suchacka, G., Rovetta, S., Masulli, F.: Online web bot detection using a sequential classification approach. In: 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). pp. 1536–1540 (2018). https://doi.org/10.1109/HPCC/SmartCity/DSS.2018.00252
6. Chiapponi, E., Dacier, M., Thonnard, O., Fangar, M., Mattsson, M., Rigal, V.: An industrial perspective on web scraping characteristics and open issues. In: 2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S). pp. 5–8 (2022). https://doi.org/10.1109/DSN-S54099.2022.00012
7. Chiapponi, E., Dacier, M., Thonnard, O., Fangar, M., Rigal, V.: BADPASS: Bots taking ADvantage of proxy as a service. In: Su, C., Gritzalis, D., Piuri, V. (eds.) Information Security Practice and Experience. pp. 327–344. Springer International Publishing
8. Chiapponi, E., Fontana, U., Boulila, E., Costanza, C., Rigal, V., Thonnard, O.: When features gets exploited: Functional abuse and the future of industrial fraud prevention. In: 2025 55nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S) (2025)
9. Donoho, D.L., Johnstone, I.M.: Ideal spatial adaptation by wavelet shrinkage **81**(3), 425–455. https://doi.org/10.1093/biomet/81.3.425, https://doi.org/10.1093/biomet/81.3.425
10. Eckersley, P.: How unique is your web browser? In: Atallah, M.J., Hopper, N.J. (eds.) Privacy Enhancing Technologies. pp. 1–18. Springer Berlin Heidelberg
11. Fielding, R.T., Taylor, R.N.: Architectural styles and the design of network-based software architectures. Ph.D. thesis (2000), aAI9980887
12. Flood, R., Engelen, G., Aspinall, D., Desmet, L.: Bad Design Smells in Benchmark NIDS Datasets . In: 2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P). pp. 658–675. IEEE Computer Society, Los Alamitos, CA, USA (Jul 2024). https://doi.org/10.1109/EuroSP60621.2024.00042
13. Friedman, D., Bousso Dieng, A.: The Vendi Score: A Diversity Evaluation Metric for Machine Learning. arXiv e-prints arXiv:2210.02410 (Oct 2022). https://doi.org/10.48550/arXiv.2210.02410
14. Hamidzadeh, J., Zabihimayvan, M., Sadeghi, R.: Detection of web site visitors based on fuzzy rough sets **22**(7), 2175–2188. https://doi.org/10.1007/s00500-016-2476-4, https://doi.org/10.1007/s00500-016-2476-4
15. Hoang, N.P., Niaki, A.A., Dalek, J., Knockel, J., Lin, P., Marczak, B., Crete-Nishihata, M., Gill, P., Polychronakis, M.: How Great is the Great Firewall?

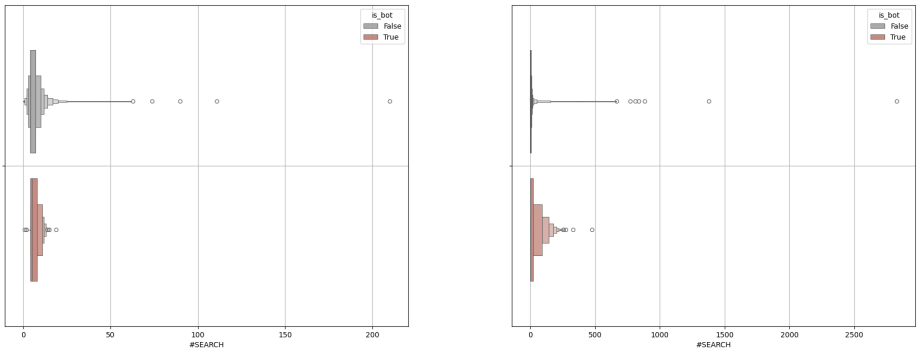
Measuring China's DNS Censorship. arXiv e-prints arXiv:2106.02167 (Jun 2021). https://doi.org/10.48550/arXiv.2106.02167
16. Hossen, M.I., Hei, X.: A low-cost attack against the hcaptcha system. In: 2021 IEEE Security and Privacy Workshops (SPW). pp. 422–431 (2021). https://doi.org/10.1109/SPW53761.2021.00061
17. Iliou, C., Kostoulas, T., Tsikrika, T., Katos, V., Vrochidis, S., Kompatsiaris, Y.: Towards a framework for detecting advanced web bots. In: Proceedings of the 14th International Conference on Availability, Reliability and Security. ARES '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3339252.3339267
18. Iqbal, U., Englehardt, S., Shafiq, Z.: Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. In: 2021 IEEE Symposium on Security and Privacy (SP). pp. 1143–1161 (2021). https://doi.org/10.1109/SP40001.2021.00017
19. Jan, S.T., Hao, Q., Hu, T., Pu, J., Oswal, S., Wang, G., Viswanath, B.: Throwing darts in the dark? detecting bots with limited data using neural data augmentation. In: 2020 IEEE Symposium on Security and Privacy (SP). pp. 1190–1206 (2020). https://doi.org/10.1109/SP40000.2020.00079
20. Khan, E., Chiapponi, E., Verkleij, M., Sperotto, A., Van Rijswijk-Deij, R., Van Der Ham-De Vos, J.: A First Look at User-Installed Residential Proxies From a Network Operator's Perspective. In: 2024 20th International Conference on Network and Service Management (CNSM). pp. 1–9 (2024). https://doi.org/10.23919/CNSM62983.2024.10814519
21. Lagopoulos, A., Tsoumakas, G.: Content-aware web robot detection **50**(11), 4017–4028. https://doi.org/10.1007/s10489-020-01754-9, https://doi.org/10.1007/s10489-020-01754-9
22. Mi, X., Feng, X., Liao, X., Liu, B., Wang, X., Qian, F., Li, Z., Alrwais, S., Sun, L., Liu, Y.: Resident evil: Understanding residential ip proxy as a dark service. In: 2019 IEEE Symposium on Security and Privacy (SP). pp. 1185–1201 (2019). https://doi.org/10.1109/SP.2019.00011
23. Motoyama, M., Levchenko, K., Kanich, C., McCoy, D., Voelker, G., Savage, S.: Re: Captchas – understanding captcha-solving services in an economic context. In: Proceedings of the 19th USENIX Security Symposium. pp. 435–452. Proceedings of the 19th USENIX Security Symposium, USENIX Association (2010)
24. Orr, W., Crawford, K.: Building Better Datasets: Seven Recommendations for Responsible Design from Dataset Creators. arXiv e-prints arXiv:2409.00252 (Aug 2024). https://doi.org/10.48550/arXiv.2409.00252
25. Ospanov, A., Farnia, F.: On the Statistical Complexity of Estimating Vendi Scores from Empirical Data. arXiv e-prints arXiv:2410.21719 (Oct 2024). https://doi.org/10.48550/arXiv.2410.21719
26. Pasarkar, A.P., Bousso Dieng, A.: Cousins Of The Vendi Score: A Family Of Similarity-Based Diversity Metrics For Science And Machine Learning. arXiv e-prints arXiv:2310.12952 (Oct 2023). https://doi.org/10.48550/arXiv.2310.12952
27. Rovetta, S., Suchacka, G., Masulli, F.: Bot recognition in a web store: An approach based on unsupervised learning. J. Netw. Comput. Appl. **157**(C) (May 2020). https://doi.org/10.1016/j.jnca.2020.102577, https://doi.org/10.1016/j.jnca.2020.102577
28. Scheuerman, M.K., Denton, E., Hanna, A.: Do Datasets Have Politics? Disciplinary Values in Computer Vision Dataset Development. arXiv e-prints arXiv:2108.04308 (Aug 2021). https://doi.org/10.48550/arXiv.2108.04308

29. Schwarz, M., Lackner, F., Gruss, D.: Javascript template attacks: Automatically inferring host information for targeted exploits. In: NDSS (Feb 2019), https://www.ndss-symposium.org/ndss2018, network and Distributed System Security Symposium 2018, NDSS'18 ; Conference date: 18-02-2018 Through 21-02-2018
30. Tan, P.N., Kumar, V.: Discovery of web robot sessions based on their navigational patterns **6**(1), 9–35 (2002). https://doi.org/10.1023/A:1013228602957, https://doi.org/10.1023/A:1013228602957
31. Venugopalan, H., Munir, S., Ahmed, S., Wang, T., King, S.T., Shafiq, Z.: FP-Inconsistent: Detecting Evasive Bots using Browser Fingerprint Inconsistencies. arXiv e-prints arXiv:2406.07647 (Jun 2024). https://doi.org/10.48550/arXiv.2406.07647
32. Watson, C., Zaw, T.: OWASP Automated Threat Handbook Web Applications Version 1.2. Tech. rep., OWASP Foundation (2018)
33. Wu, M., Sippe, J., Sivakumar, D., Burg, J., Anderson, P., Wang, X., Bock, K., Houmansadr, A., Levin, D., Wustrow, E.: How the great firewall of china detects and blocks fully encrypted traffic. In: Proceedings of the 32nd USENIX Conference on Security Symposium. SEC '23, USENIX Association, USA (2023)
34. Wu, S., Sun, P., Zhao, Y., Cao, Y.: Him of Many Faces: Characterizing Billion-scale Adversarial and Benign Browser Fingerprints on Commercial Websites. In: NDSS (2023)
35. Zabihi, M., Jahan, M.V., Hamidzadeh, J.: A density based clustering approach for web robot detection. In: 2014 4th International Conference on Computer and Knowledge Engineering (ICCKE). pp. 23–28 (2014). https://doi.org/10.1109/ICCKE.2014.6993362

## A Bot variants behavior across features

We conducted a detailed analysis of the features contributing to bot classification across the three domains. The feature that represents the volume of API calls of type search was not given high importance in the model trained on the first bot variant (on D1 and D3) compared to the second variant on D2, as displayed in Figure 7. In fact, as shown in Figure 8a, bots in the first (and third) domain tend to have a more stealthy approach, performing small searches per session, in a comparable way as humans. However, in the second domain, automated session prefer a more effective approach, retrieving high volume of data, consequently issuing many requests, in a single session. Additionally, Figure 8b reveals that some outliers in the negative cluster exceed 2,000 searches in a single session, suggesting these outliers are false negative labeled entries.

With regards to some other features, the model trained on D2 assigns low importance to the `inference time std` feature. When plotting the empirical cumulative distribution function for the two domains D1 and D2, as in Figure 9, we observe a clear discrepancy in the behavior of the negative and positive classes. Human sessions are chaotic, with high variance in the time between subsequent actions, whereas bots in D1 have a waiting time not exceeding the 10 seconds. In contrast, bots in D2 exhibits higher variance in the waiting time, narrowing the gap between the positive and negative class curves. For this reason,




(a) (b)

Fig. 8: Boxplot of the volume of API calls of type search within sessions for domains D1 in (a), and D2 in (b)

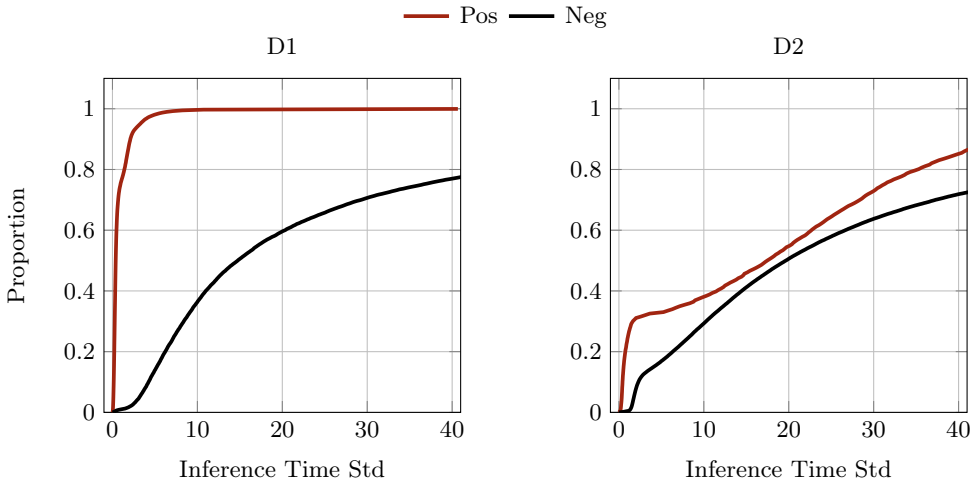
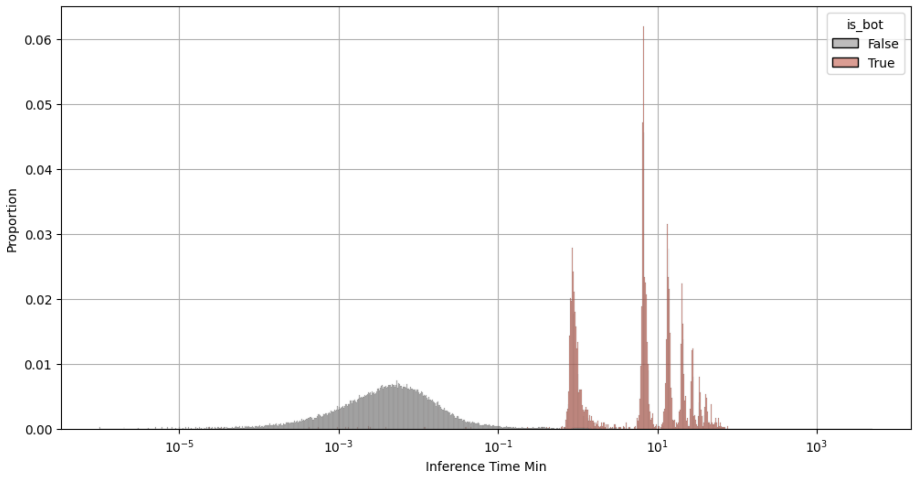


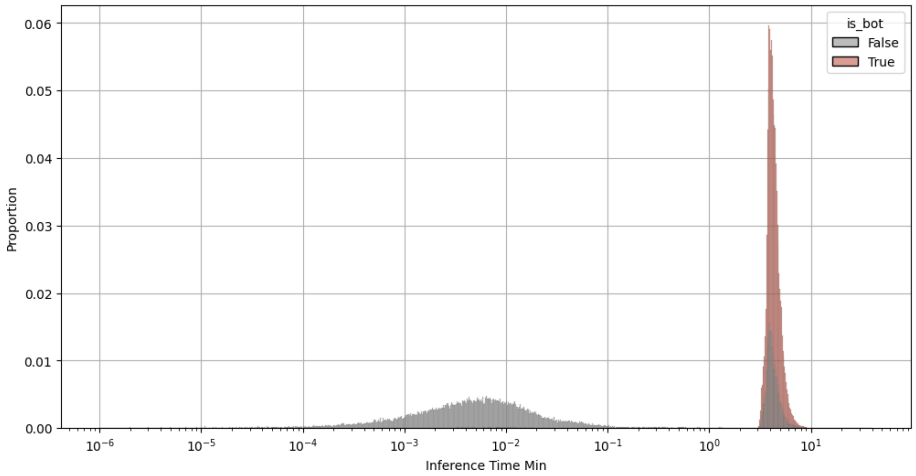

Fig. 9: Empirical cumulative distribution of the inference time std features for the two bot variants.

the model on trained on the second domain prioritize other features for the detection.

On the other hand, the minimum inference time within a session is crucial across all domains. Figure 10a shows that in domain D2, the positive class has more peaks compared to D3 (and D1), inFigure 10b, indicating the probable presence of different bot varieties in D2. These peaks are distinct from the negative class values distribution, which generally has lower minimum inference time within a session, reflecting only the latency between dependent requests rather than artificially crafted waiting time. In D3, many negative labels also have high




(a)




(b)

Fig. 10: Histogram distribution of minimum inference time for domains D2 in (a), and D3 in (b)

values for this feature, likely due to false negative labeled data undetected by the labeling heuristic. In fact, domain D3 only relied on malicious ASN-based labeling, based on IP repetition rule, which alone may not be enough to capture all highly suspicious traffic. To establish a more accurate ground truth, this method should be integrated with additional techniques, such as fingerprinting rules.