

Análisis y caracterización de datos en la etapa 3 del módulo

*Informe de la Práctica Laboral 2*

**Autores:**

Carlos Raúl Robinson Thompson

Adrian Marqués Gaute

**Tutores:**

Dra.C. Nayma Cepero Pérez

Ing. Daniel Pardo Echevarría

**La Habana, 2024**

**Resumen**

La creciente utilización de Internet en la actualidad ha propiciado un aumento en la frecuencia de ataques cibernéticos. Entre las herramientas más empleadas por los delincuentes digitales para llevar a cabo estas infracciones de seguridad se encuentran los *bots* maliciosos. Estos programas tienen la capacidad de replicar comportamientos humanos, lo que les permite afectar una gran variedad de sitios web.

Con el fin de asegurar la identificación de *bots* maliciosos, se han desarrollado diversas técnicas, destacando especialmente la implementación de múltiples algoritmos de aprendizaje automático. En este contexto, el meta aprendizaje se presenta como una estrategia prometedora para optimizar el rendimiento de estos algoritmos, considerando sus principales limitaciones.

Este trabajo tiene como objetivo reestructurar el proceso de caracterización de datos en un módulo destinado a la detección de *bots* maliciosos, mediante la introducción de un enfoque de clasificación que no dependa de datos etiquetados para su funcionamiento. Asimismo, se modifican las características a evaluar para eliminar la necesidad de datos etiquetados.

Se llevará a cabo un diseño experimental con el propósito de identificar la configuración óptima del enfoque implementado, complementado con un estudio de casos que permita examinar los resultados obtenidos a través del componente de meta aprendizaje.

Palabras Claves: Detección de *bots*, aprendizaje automático, meta aprendizaje, datos etiquetados, características

**Abstract**

The increasing use of the Internet today has led to a rise in the frequency of cyberattacks. Among the tools most commonly used by cybercriminals to carry out these security breaches are malicious bots. These programs can replicate human behaviors, allowing them to affect a wide variety of websites.

To ensure the identification of malicious bots, various techniques have been developed, with the implementation of multiple machine learning algorithms standing out. In this context, meta learning emerges as a promising strategy to optimize the performance of these algorithms, considering their main limitations.

This work aims to restructure the data characterization process in a module intended for malicious bot detection by introducing a classification approach that does not rely on labeled data for its operation. Also, the features to be evaluated are modified to eliminate the need for labeled data.

An experimental design will be carried out to identify the optimal configuration of the approach, complemented by a case study that allows for examining the results obtained through the meta learning component.

Keywords: detecting bots, machine learning, meta learning, labeled data, features

**ANEXO A. PLAN INDIVIDUAL DE TAREAS DEL ESTUDIANTE**

|  |  |  |
| --- | --- | --- |
| **Tareas** | **Fecha de entrega** | **Rol(es) que desarrolla(n) con la tarea** |
| Reunión de inicio de la práctica | 29/4/2024 | - |
| Buscar métricas de caracterización de datos que no dependan de datos etiquetados. | 10/5/2024 | AN |
| Revisar Introducción (Con diseño de la investigación) y entregar al tutor. | 18/5/2024 | EE |
| Elaborar capítulo 1 del estado del arte centrado en la detección de bots malignos y caracterización de datos. | 14/5/2024 | EE |
| Entregar capítulo 1 al tutor. | 25/5/2024 | EE |
| Rectificar señalamientos del capítulo 1 y la introducción. | 28/5/2024 | EE |
| Entregar al tutor capítulo 1 e introducción corregidos. | 30/5/2024 | EE |
| Implementar tres métricas de caracterización consultadas. | 7/6/2024 | PG |
| Proponer un re-diseño de la base de hechos de módulo de detección de bots malignos, a partir de las métricas implementadas y del enfoque que caracterice los datos en bots malignos y humanos. | 26/6/2024 | AR, AS |
| Proponer y desarrollar una nueva forma para simular un cambio en las características de los datos que no sea aleatoria. | 30/6/2024 | PG |
| Efectuar un estudio de casos de la etapa número tres del módulo de detección de bots malignos, a partir de la nueva forma de caracterizar los datos de bots malignos y humanos. | 07/7/2024 | PB, AD |
| Elaborar capítulo con los resultados alcanzados en el estudio de casos. | 10/7/2024 | EE, AD |
| Entregar capítulo al tutor para revisión. | 14/7/2024 | EE |
| Rectificar los señalamientos del capítulo revisado. | 15/7/2024 | EE |
| Elaborar informe de práctica profesional. | 10/7/2024 | EE |
| Entregar informe al tutor para revisión. | 17/7/2024 | EE |
| Rectificar los señalamientos del informe. | 19/7/2024 | EE |
| Entregar informe final de las prácticas profesionales 2. | 02/9/2024 | - |
| Defensa de la práctica profesional. | 4-6/9/2024 | EE |

|  |  |  |
| --- | --- | --- |
| Nayma Cepero Pérez  D:\_____Investigación\___Mi reguero\firmas\firma.pngNombre completo y firma del primer tutor | Daniel Pardo Echevarría  Nombre completo y firma del segundo tutor |  |
| Adrian Marqués Gaute  Nombre completo de los estudiantes | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Firma de los estudiantes |

**ANEXO A. PLAN INDIVIDUAL DE TAREAS DEL ESTUDIANTE**

|  |  |  |
| --- | --- | --- |
| **Tareas** | **Fecha de entrega** | **Rol(es) que desarrolla(n) con la tarea** |
| Reunión de inicio de la práctica | 29/4/2024 | - |
| Buscar métricas de caracterización de datos que no dependan de datos etiquetados. | 10/5/2024 | AN |
| Revisar Introducción (Con diseño de la investigación) y entregar al tutor. | 18/5/2024 | EE |
| Elaborar capítulo 1 del estado del arte centrado en la detección de bots malignos y caracterización de datos. | 14/5/2024 | EE |
| Entregar capítulo 1 al tutor. | 25/5/2024 | EE |
| Rectificar señalamientos del capítulo 1 y la introducción. | 28/5/2024 | EE |
| Entregar al tutor capítulo 1 e introducción corregidos. | 30/5/2024 | EE |
| Implementar dos métricas de caracterización consultadas. | 7/6/2024 | PG |
| Proponer un nuevo enfoque que permita caracterizar a un conjunto de datos en bots malignos y usuarios humanos. | 15/6/2024 | AS |
| Desarrollar el nuevo enfoque de caracterización de datos de bots y humanos propuesto | 22/6/2024 | PG |
| Efectuar un estudio de casos de la etapa número tres del módulo de detección de bots malignos, a partir de la nueva forma de caracterizar los datos de bots malignos y humanos. | 07/7/2024 | PB, AD |
| Elaborar capítulo con los resultados alcanzados en el estudio de casos. | 10/7/2024 | EE, AD |
| Entregar capítulo al tutor para revisión. | 14/7/2024 | EE |
| Rectificar los señalamientos del capítulo revisado. | 15/7/2024 | EE |
| Elaborar informe de práctica profesional. | 10/7/2024 | EE |
| Entregar informe al tutor para revisión. | 17/7/2024 | EE |
| Rectificar los señalamientos del informe. | 19/7/2024 | EE |
| Entregar informe final de las prácticas profesionales 2. | 02/9/2024 | - |
| Defensa de la práctica profesional. | 4-6/9/2024 | EE |

|  |  |  |
| --- | --- | --- |
| Nayma Cepero Pérez  D:\_____Investigación\___Mi reguero\firmas\firma.pngNombre completo y firma del primer tutor | Daniel Pardo Echevarría  Nombre completo y firma del segundo tutor |  |
| Carlos Raúl Robinson Thompson  Nombre completo de los estudiantes | C:\Users\Thompson\Desktop\photo_2023-06-02_19-37-49.jpg\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Firma de los estudiantes |

Índice

[Introducción 1](#_Toc176797672)

[Capítulo 1: Estado del Arte 7](#_Toc176797673)

[1.1-Impacto de los *bots* en la Web 7](#_Toc176797674)

[1.2-Tipos de bots 9](#_Toc176797675)

[1.3-Detección de *bots* 11](#_Toc176797676)

[1.3.1-Aprendizaje automático en la detección de *bots* 13](#_Toc176797677)

[1.4-Algoritmos de aprendizaje supervisado en la detección de bots 14](#_Toc176797678)

[1.4.1-Naive Bayes 15](#_Toc176797679)

[1.4.2-KNN 16](#_Toc176797680)

[1.4.3 -Suport Vector Machine 16](#_Toc176797681)

[1.4.4-Árbol de decisión 16](#_Toc176797682)

[1.4.5-Bosque de decisión 18](#_Toc176797683)

[1.5- Meta aprendizaje 20](#_Toc176797684)

[1.6- Métricas de caracterización de datos 21](#_Toc176797685)

[1.6.1-Rango Intercuartil (IQR) 21](#_Toc176797686)

[1.6.2- Desviación Media Absoluta (MAD) 21](#_Toc176797687)

[1.6.3- Entropía de Shannon 22](#_Toc176797688)

[1.6.4- Distancia de Mahalanobis 22](#_Toc176797689)

[1.6.5- CUSUM 23](#_Toc176797690)

[1.6.6- Distancia de Jensen\_Shannon 24](#_Toc176797691)

[1.7-Módulo de detección de *bots* 24](#_Toc176797692)

[1.8-Conclusiones parciales 27](#_Toc176797693)

[Capítulo 2: Desarrollo del nuevo enfoque de detección 28](#_Toc176797694)

[2.1-Detalles de la etapa 3 del módulo de detección de *bots* malignos 28](#_Toc176797695)

[2.2-Requisitos funcionales 31](#_Toc176797696)

[2.3 - Simulación de cambios no aleatorios 31](#_Toc176797697)

[2.4-Implementación de métricas de caracterización de datos 33](#_Toc176797698)

[2.4.1- Implementación de la distancia de Mahalanobis (MD) 34](#_Toc176797699)

[2.4.2- Implementación de MAD 35](#_Toc176797700)

[2.4.3- Implementación de la entropía de Shannon (SE) 36](#_Toc176797701)

[2.4.4- Implementación de la distancia de Jensen-Shannon (JS) 38](#_Toc176797702)

[2.4.5- Implementación de CUSUM 39](#_Toc176797703)

[2.5-Algoritmo para la asignación de la etiqueta de caracterización, a partir de un enfoque de detección 41](#_Toc176797704)

[2.6- Ejecución de la tercera etapa del módulo 43](#_Toc176797705)

[2.7-Conclusiones parciales 46](#_Toc176797706)

[Capítulo 3: Evaluación del nuevo enfoque de detección 47](#_Toc176797707)

[3.1-Base de datos CTU-13 47](#_Toc176797708)

[3.1.1- Pre procesamiento 48](#_Toc176797709)

[3.1.2- Base de datos central de CTU-13 48](#_Toc176797710)

[3.2-Generación del árbol de reglas para la selección de etiqueta de caracterización 49](#_Toc176797711)

[3.2.1-Elección de la configuración 50](#_Toc176797712)

[3.3-Estudio de casos 57](#_Toc176797713)

[3.3.1-Descripción de los casos identificados 57](#_Toc176797714)

[3.4 – Conclusiones parciales 70](#_Toc176797715)

[Conclusiones generales 71](#_Toc176797716)

[Recomendaciones 72](#_Toc176797717)

[Referencias Bibliográficas 73](#_Toc176797718)

[Anexos 76](#_Toc176797719)

**Figuras**

[Figura 1 Cambios en el flujo de red en el 2023 [18] 9](#_Toc176722673)

[Figura 2 Evolución de la representación de los bots en el flujo de red global [18] 9](#_Toc176722674)

[Figura 3 Tipos de bots malignos [18] 11](#_Toc176722675)

[Figura 4 Modelo de dominio tomado de [9] 15](#_Toc176722676)

[Figura 5 Etapas del módulo [9] 26](#_Toc176722677)

[Figura 6 Ejecución del proceso de la etapa 3 [9] 30](#_Toc176722678)

[Figura 7 Diagrama de flujo de la distancia de Mahalanobis 35](#_Toc176722679)

[Figura 8 Diagrama de flujo de MAD 36](#_Toc176722680)

[Figura 9 Diagrama de flujo de la entropía de Shannon 37](#_Toc176722681)

[Figura 10 Diagrama de flujo de la distancia de Jensen Shannon 39](#_Toc176722682)

[Figura 11 Diagrama de flujo de CUSUM 40](#_Toc176722683)

[Figura 12: Diagrama de flujo del proceso de caracterización de usuarios por meta componente 44](#_Toc176722684)

[Figura 13 Diagrama de secuencia del proceso de la etapa 3 46](#_Toc176722685)

[Figura 14 Flujo de Knime utilizado para generar el conjunto de reglas 50](#_Toc176722686)

[Figura 15: Gráfica de Caja para los tamaños de paquete de 5000, 10000, 20000 y 25000 54](#_Toc176722687)

[Figura 16: Gráfica de Caja para los tamaños de paquete de 5,000, 10,000, 20,000 y 25,000 57](#_Toc176722688)

**Tablas**

[Tabla 1 Seudocódigo de la función simulate\_positives 32](#_Toc176722689)

[Tabla 2: Seudocódigo del algoritmo para la asignación de la etiqueta de clase 41](#_Toc176722690)

[Tabla 3: Resultados del enfoque con un tamaño de entrada de 100,000 51](#_Toc176722691)

[Tabla 4: Resultados del enfoque con un tamaño de entrada de 200,000 52](#_Toc176722692)

[Tabla 5 Diferencias para entrada de 200,000 usuarios 54](#_Toc176722693)

[Tabla 6: Resultados del enfoque con un tamaño de entrada de 300,000 55](#_Toc176722694)

[Tabla 7 Diferencias para entrada de 300,000 usuarios 56](#_Toc176722695)

[Tabla 8 Caso 1 Primera ejecución 59](#_Toc176722696)

[Tabla 9 Caso 1 Segunda ejecución 61](#_Toc176722697)

[Tabla 10 Caso 2 Primera ejecución 63](#_Toc176722698)

[Tabla 11 Caso 2 Segunda ejecución 64](#_Toc176722699)

[Tabla 12 Caso 3 Primera ejecución 67](#_Toc176722700)

[Tabla 13 Caso 3 Segunda ejecución 68](#_Toc176722701)

**Ecuaciones**

[Ecuación 1 Entropía de Shannon 22](#_Toc176722842)

[Ecuación 2 Distancia de Mahalanobis 23](#_Toc176722843)

[Ecuación 3 Distancia de Jensen\_Shannon 24](#_Toc176722844)

[Ecuación 4 Distribución promedio 24](#_Toc176722845)

[Ecuación 5 Divergencia KL 24](#_Toc176722846)

# Introducción

La expansión y creciente adopción de dispositivos conectados a Internet, junto con el aumento constante de datos producidos por estos, han creado nuevas oportunidades para los ataques cibernéticos. Los incidentes más comunes en las redes incluyen: estafas de identidad, robos de información de tarjetas de crédito y débito, fallas en los sistemas y servidores, entre otros tipos de amenazas, pero estos ataques no son realizados solo por humanos, debido al papel cada vez más prominente jugado por software malicioso. En particular, los ataques llevados a cabo por *bots* se han convertido en uno de los principales riesgos de seguridad entre todas las formas de malware presentes en Internet, llegando a representar el 80% de los ataques cibernéticos en el mundo moderno [1]. Los *bots*, abreviatura de "robots", son programas informáticos diseñados para realizar tareas automáticas en la red [2-4]. Estos pueden tener diversas funciones, desde responder automáticamente a consultas simples y realizar tareas repetitivas, hasta realizar acciones más complejas como interactuar con usuarios en redes sociales o realizar transacciones comerciales [2, 3].

Sin embargo, el auge de los *bots* también ha planteado preocupaciones en torno a la privacidad y la seguridad de los datos. Algunos *bots* pueden ser utilizados de manera maliciosa para recopilar información personal de los usuarios sin su consentimiento, lo que plantea riesgos de violación de la privacidad. Además, también pueden ser utilizados para difundir spam, realizar actividades fraudulentas, ataques distribuidos de denegación de servicio (DDoS), entre otros, causando perjuicios a usuarios y clientes en línea [1, 5, 6].

Ante esta doble naturaleza, los *bots* suponen una amenaza significativa para la ciberseguridad, lo que hace imperativa su correcta detección [5]. Para abordar este reto, han surgido numerosas soluciones como las basadas en *honeypots* o análisis de comportamiento (test de Turing), pero en especial se destacan las soluciones de minería de datos aplicando diversos algoritmos de aprendizaje automático [7].

La minería de datos es una disciplina que se encarga de extraer información relevante y patrones ocultos a partir de grandes volúmenes de datos lo que permite identificar comportamientos anómalos que podrían indicar la presencia de *bots* [8]. En diversos trabajos tales como [7, 9, 10] se plantea el uso de la minería de datos para detectar *bots* malignos o maliciosos, a partir de la aplicación y desarrollo de enfoques de aprendizaje automático(ML por sus siglas en ingles). Este se centra en el entrenamiento de algoritmos para analizar y aprender de los datos, lo que facilita la creación de modelos capaces de detectar la actividad de *bots*, especialmente a partir del aprendizaje supervisado [8, 11].

El aprendizaje supervisado es una rama del aprendizaje automático que se basa en el uso de datos etiquetados para entrenar algoritmos [8, 11]. En la detección de *bots*, implica el uso de conjuntos de datos que contienen ejemplos de comportamientos de *bots* malignos previamente etiquetados. Estos datos se utilizan para entrenar un modelo que pueda distinguir entre actividades legítimas y maliciosas [9].

Dentro del aprendizaje supervisado en la detección de *bots* malignos, sobresalen los árboles de decisión, que son estructuras de datos jerárquicos capaces de capturar relaciones no lineales entre las características y las etiquetas. Representan una secuencia de decisiones a partir de un nodo raíz y sus posibles resultados (nodos hojas).Sin embargo, estos tienden a sobre ajustarse a los datos de entrenamiento, lo que puede limitar su capacidad para generalizar a nuevos datos [8].

Otro de los algoritmos utilizados en la detección de *bots* malignos son los bosques de decisión, tales como *Random Forest* (RF) y *Proactive Forest* (PF), que se plantean como una alternativa a las limitaciones de un árbol de decisión individual. Los bosques de decisión son conjuntos de múltiples árboles de decisión que trabajan juntos para mejorar la precisión y la diversidad del modelo [12-14]. Cada árbol en el bosque se entrena con una muestra del conjunto de datos original, lo que ayuda a reducir el sobreajuste y a mejorar la capacidad predictiva del modelo [13]. Esta técnica permite capturar patrones complejos y variaciones en los datos, lo que los hace especialmente efectivos en la detección de *bots* y otros problemas de clasificación.

Como se ha planteado los algoritmos de aprendizaje supervisado aprenden de datos históricos, sin embargo con la rápida evolución de los *bots* y la similitud que tienen los *bots* malignos con los usuarios humanos, hace que su detección en el futuro pueda tener ciertas limitaciones si solo se usa un algoritmo de aprendizaje supervisado [15, 16]. Una posible solución a este problema es la aplicación del meta aprendizaje, que permite a los modelos adaptarse y mejorar continuamente frente a nuevas estrategias de detección de *bots* malignos [16].

El meta aprendizaje es una técnica de aprendizaje automático que permite a un modelo adaptarse a nuevos conjuntos de datos y escenarios en función de las experiencias previas, incluso cuando los comportamientos difieren significativamente de los ejemplos con los que fue entrenado [16]. Uno de los marcos fundamentales de esta técnica lo constituye el uso de métricas para la caracterización de conjuntos de datos. Estas métricas permiten evaluar y describir las propiedades y estructuras de los datos, lo que es ayuda en gran medida a la selección y optimización de modelos de aprendizaje [16]. También es importante destacar que existen métricas que se pueden utilizar independientemente para la detección de usuarios anómalos (*bots*) como es el caso de la entropía de Shannon [17].

A partir del análisis de las potencialidades que ofrece el aprendizaje automático, en la Universidad Tecnológica de La Habana José Antonio Echeverría (Cujae) se desarrolló un módulo de detección de *bots* malignos, basado en ML que consta de tres etapas, sin embargo, este no ha sido completado. En su tercera etapa se realiza un proceso de caracterización que depende de datos etiquetados tanto para la asignación de la etiqueta clase como para la caracterización en sí, lo que puede restringir la precisión en caso de instancias totalmente desconocidas [7, 9]. A partir de esto se produce un tema de investigación o **problema** a resolver: ¿Cómo mejorar la clasificación entre *bots* y humanos?

Para solucionar este problema se propone como **objeto de estudio** el aprendizaje automático en la detección de *bots* y como **campo de acción** el meta aprendizaje en la detección de *bots*.

El **objetivo general** es desarrollar una nueva versión del módulo para detectar *bots* malignos, basado en aprendizaje automático (durante las PP2 lo relacionado al proceso de caracterización de datos de la etapa 3 y en la Tesis lo relacionado a la etapa 2 y la integración). A continuación, se presentan los **objetivos específicos**:

1. Asimilar los elementos teóricos asociados a las métricas de caracterización de datos en la detección de *bots*
   * Analizar la literatura existente sobre detección de *bots* (Adrian y Carlos).
   * Identificar métricas de caracterización de datos que no dependan de datos etiquetados (PP2, Adrian y Carlos).
   * Implementar 6 métricas de caracterización de datos (PP2, Adrian-3 y Carlos-3).
2. Rediseñar el módulo de clasificación de *bots* malignos.
   * + Proponer e implementar un nuevo enfoque de clasificación a partir de las métricas implementadas (PP2, Carlos).
     + Rediseñar la base de hechos del módulo a partir de las métricas empleadas y el enfoque propuesto (PP2, Adrian).
     + Integrar un esquema basado en meta aprendizaje en la segunda etapa, que permita elegir qué clasificador utilizar a partir de los datos (Tesis, Carlos).
   * Proponer y desarrollar una nueva forma para simular un cambio en las características de los datos que no sea aleatoria (PP2, Adrian).
   * Integrar las tres etapas del módulo de detección de *bots* malignos (Tesis, Adrian).
3. Evaluar y comparar los resultados obtenidos por las etapas del módulo de detección de *bots* malignos.

* Efectuar un estudio de casos de la etapa tres del módulo a partir de la nueva forma de caracterizar los datos (PP2, Adrian y Carlos).
* Comparar los resultados alcanzados en la etapa dos al aplicar el nuevo esquema de selección de un algoritmo de clasificación con los resultados obtenidos en trabajos previos (Tesis, Carlos).
* Comparar los resultados del módulo de detección de *bots* con los obtenidos en trabajos previos y otros enfoques propuestos en la literatura (Tesis, Adrian y Carlos).
* Analizar los resultados y arribar a conclusiones (Adrian y Carlos).

El valor teórico del trabajo radica en la asimilación de los principales conceptos, técnicas de aprendizaje automático y métricas de caracterización de datos aplicados a la detección de *bots*. El valor práctico radica en el progreso en un módulo para detectar *bots* malignos, basado en aprendizaje automático, evaluando su rendimiento a partir de un diseño experimental y estudio de casos. Como artefacto de salida se produce un algoritmo.

La estructura del trabajo propuesto para las Prácticas Profesionales 2 se organiza en tres capítulos fundamentales, diseñados para abordar de manera sistemática y práctica el desafío de mejorar la detección de *bots* mediante el uso de métricas de caracterización de datos. Cada capítulo tiene un propósito específico, desde la revisión teórica hasta la implementación práctica y evaluación de los resultados.

Capítulo 1: Análisis Teórico y Contextual

Este capítulo establece las bases teóricas y contextualiza el problema de la detección de *bots* en el panorama actual de seguridad cibernética. Se discuten:

* **Impacto de los *bots***: Se examina cómo los *bots* afectan la seguridad en línea, destacando su papel en ataques comunes como DDoS, *phishing*, y spam.
* **Técnicas de Detección**: Se revisan las técnicas actuales para la detección de *bots*, incluyendo el aprendizaje automático y la minería de datos.
* **Métricas de Caracterización de Datos**: Se introduce el concepto de métricas de caracterización de datos como una alternativa a los métodos tradicionales de detección, enfocándose en aquellos que no requieren datos etiquetados.

Capítulo 2: Desarrollo del nuevo enfoque de detección

Aquí se lleva a cabo la implementación práctica de los avances realizados en el módulo:

* **Implementación de nuevas métricas de caracterización de datos.**
* **Desarrollo de Clasificador**: Se propone e implementa un nuevo enfoque de clasificación basado en las métricas elegidas.
* **Rediseño de la Base de Hechos**: Se proponen y desarrollan modificaciones en la base de hechos del módulo de detección de *bots* para integrar las nuevas métricas.
* **Simulación de Cambios en Datos**: Se crea una nueva forma de simular variaciones en las características de los datos, no aleatorias, para evaluar el nuevo enfoque.

Capítulo 3: Evaluación del nuevo enfoque de detección

Finalmente, este capítulo está dedicado a la evaluación de los resultados obtenidos con el nuevo enfoque de clasificación:

* **Configuración del enfoque: Se realiza un experimento para determinar la configuración a emplear en el enfoque implementado, como la cantidad de paquetes a analizar y las reglas a emplear.**
* **Pruebas de Caso Estudio**: Se realizan estudios de caso para validar el nuevo enfoque de clasificación.

# Capítulo 1: Estado del Arte

En este capítulo se presentan los principales conceptos necesarios para comprender el problema de la detección de *bots*. Se analizan las principales técnicas utilizadas para la detección de *bots*, con un enfoque especial en el aprendizaje automático y las métricas de caracterización de datos.

## 1.1-[Impacto de los *bots* en la Web](file:///C:\Users\Adrian%20y%20Karina\Downloads\Telegram%20Desktop\PP1-Adrian_Carlos-G33-RevDany.docx#_Toc128028)

En el vertiginoso mundo digital actual, donde la información se mueve a una velocidad sin precedentes, la necesidad de optimizar procesos y automatizar tareas se ha vuelto fundamental. Es aquí donde los bots entran en escena como poderosos aliados. Estos agentes que imitan el comportamiento humano, se convierten en un ejército digital capaz de realizar tareas repetitivas a gran velocidad, liberando a los humanos de las tareas tediosas y permitiendo que se concentren en labores más complejas y creativas [9].

Sin embargo, la capacidad de los *bots*, originalmente concebida para mejorar la eficiencia, se ha convertido en un arma de doble filo. Esta potencia, en manos equivocadas, se transforma en una amenaza silenciosa que acecha el mundo digital.

Los *bots* maliciosos no actúan de forma aislada, sino que se organizan bajo una estructura centralizada que les permite operar de forma coordinada. Este sistema de control, conocido como "comando y control" (C&C), es el cerebro detrás de este riesgo digital, el punto de origen de las órdenes y las actualizaciones que los mantienen operativos y evolucionando [1, 5, 6].

Los *bots* maliciosos están constituidos por 5 fases o etapas principales las cuales se explican a continuación [1, 5]:

1. Fase de Exploración: El servidor C&C, como un general estratégico, escanea la red en busca de vulnerabilidades en sistemas y servidores, buscando el punto débil para infiltrarse.

2. Fase de Infección: Mediante código malicioso (malware) descargado desde el servidor C&C, como un virus letal, los *bots* se introducen en los sistemas objetivo, infectando computadoras y dispositivos.

3. Fase de Conexión: El malware se activa, conectando el sistema infectado al servidor C&C, estableciendo un canal de comunicación para recibir órdenes del *botmaster*, el controlador del atacante digital.

4. Fase de Ataque: El *botmaster*, a través del servidor C&C, envía comandos para ejecutar ataques coordinados, como ataques de denegación de servicio (DoS), robo de información, spam o la propagación de malware, causando disrupciones en la red y daños a los sistemas.

5. Fase de Actualización: Los *bots* se actualizan constantemente para evadir la detección y mantenerse operativos, como un ejército en constante evolución, adaptándose a las nuevas medidas de seguridad.

Las estadísticas evidencian que esta amenaza comprende más del 80% de los ataques online [1]. Como se ve en la figura 1, los *bots* representan el 49.6% del tráfico total de internet, en el que los malignos representan el 32%. La cifra es alarmante, considerando que el objetivo principal de los sitios web es atraer a usuarios humanos, que solo constituyen el 50.4, y cuyo porcentaje en la red va disminuyendo a medida que pasan los años, como se evidencia en la figura 2.

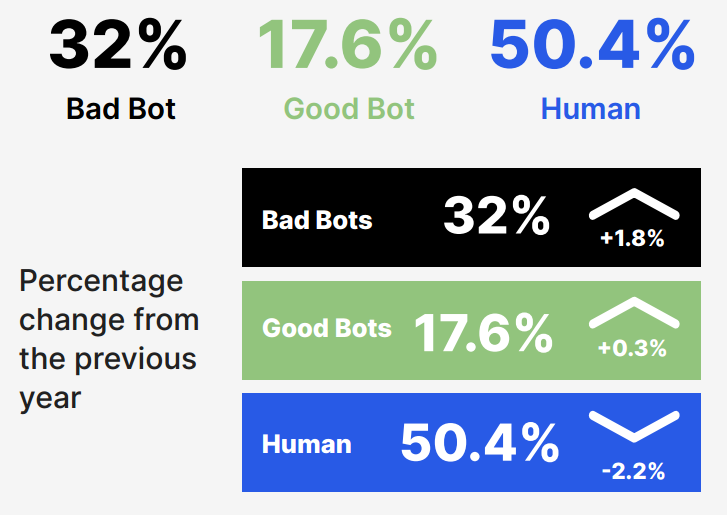


Figura 1 Cambios en el flujo de red en el 2023 [18]

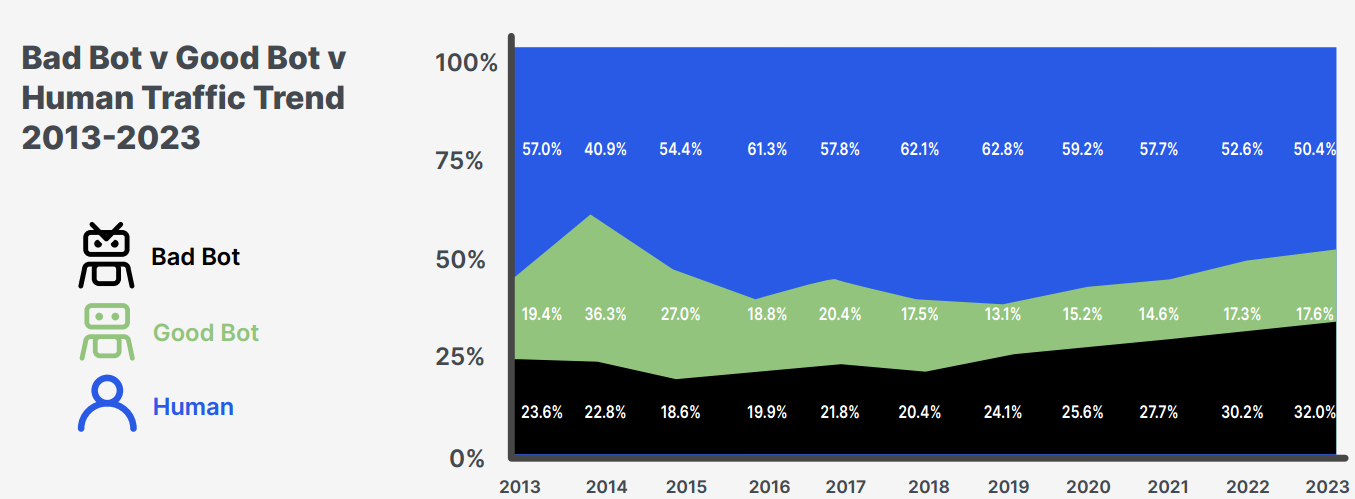


Figura 2 Evolución de la representación de los bots en el flujo de red global [18]

## 1.2-[Tipos de bots](file:///C:\Users\Adrian%20y%20Karina\Downloads\Telegram%20Desktop\PP1-Adrian_Carlos-G33-RevDany.docx#_Toc128029)

Los *bots*, como entidades digitales omnipresentes en la red, se clasifican en diversas categorías, entre las que se destaca una clasificación dual en el comportamiento de los mismos según los motivos de su creación:

*Bots* Benignos: Son herramientas valiosas que agilizan procesos y tareas en internet desde una perspectiva legal. Suelen ser utilizados por motores de búsqueda para indexar información, por *chatbots* para interactuar con clientes, o por asistentes virtuales para automatizar tareas repetitivas. Su objetivo es brindar eficiencia y mejorar la experiencia del usuario [2, 3, 18].

*Bots* Malignos o maliciosos: Diseñados para cometer actos ilícitos, representan una amenaza para la seguridad online. Se programan para perpetrar ataques a sitios web, robar información confidencial, propagar spam o realizar otras actividades dañinas. Se caracterizan por su capacidad para imitar el comportamiento humano, lo que dificulta su detección y rastreo [2, 3, 18]. Según su nivel de sofisticación se clasifican en [18]:

* + - Simples: Se caracterizan por su simplicidad y código básico. Usan una dirección IP fija y no simulan la navegación humana.
    - Moderados: Emplean un software de navegación sin cabeza que imita el comportamiento de un navegador web. Son más complejos y difíciles de detectar que los simples.
    - Avanzados: Son extremadamente complejos y utilizan técnicas avanzadas para imitar el comportamiento humano. Emplean software de automatización de navegadores o malware que se instala en los navegadores reales para acceder a sitios web.
    - Evasivos: Representan una combinación de los niveles moderado y avanzados, moviéndose a través de múltiples direcciones IP a través de *proxies* anónimos, capaces de modificar credenciales de usuario, superar retos CAPTCHA y mantener presencia continua en plataformas objetivo.

La creciente adopción de la IA ha tenido un impacto significativo en los *bots* maliciosos, creando una división entre *bots* avanzados, impulsados por recursos sofisticados, y *bots* simples que utilizan herramientas básicas de IA para generar scripts, como se puede apreciar en la figura 3. Si bien los simples han aumentado su participación en el tráfico de *bots* maliciosos (39.6% en 2023, frente al 26.3% de hace cinco años), la popularidad de los moderadamente sofisticados ha disminuido (12.4% en 2023). Sin embargo, los avanzados siguen representando una parte importante del tráfico de *bots* malignos (48.1% en 2023), lo que indica que las tácticas de evasión siguen siendo altamente sofisticadas y se están desarrollando constantemente [18].

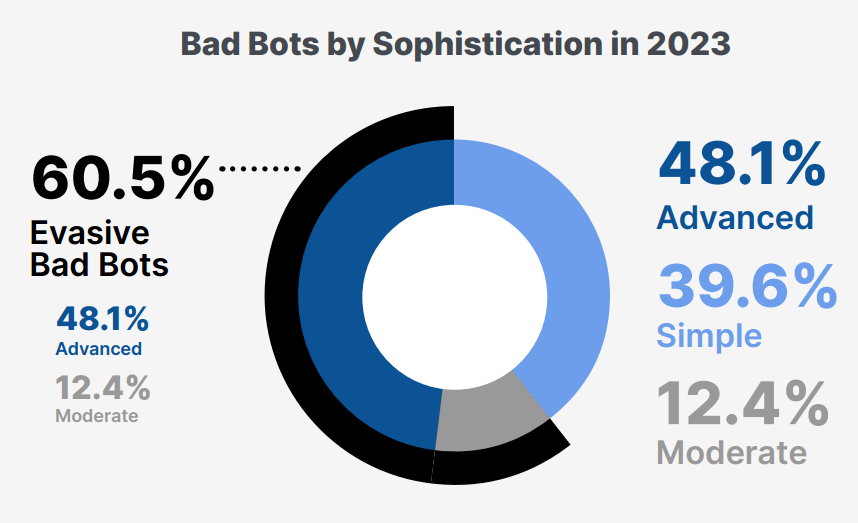


Figura 3 Tipos de bots malignos [18]

## 1.3-Detección de *bots*

A partir de la complejidad y sofisticación de los *bots* maliciosos analizada, es fundamental implementar estrategias efectivas para detectar su presencia y neutralizar su impacto. El proceso de detección se basa en la capacidad de identificar, con precisión, si una solicitud proviene de un humano o un *bot* maligno. Esta tarea implica clasificar las sesiones, es decir, el conjunto de solicitudes realizadas por un usuario durante una visita al sitio web. Esta operación se basa principalmente en técnicas de análisis de los registros de acceso al servidor, pero algunas permiten la detección en tiempo real mientras todavía está navegando en el sitio [19].

La detección en tiempo real permite bloquear el acceso a los recursos del sitio de forma inmediata, pero es crucial que el margen de error sea bajo para evitar la clasificación errónea de usuarios humanos [19]. Debido a la gran variedad de estructuras y técnicas de evasión que los *bots* utilizan, se requiere la implementación de diferentes enfoques para combatirlos. Estos enfoques se pueden agrupar en las siguientes categorías [7, 19]:

* Enfoques Distribuidos: Se basan en la distribución lógica o física de los detectores para reducir la superficie de ataque. Son efectivos para mejorar la seguridad en entornos estáticos y dinámicos y disminuir la resiliencia de los ataques de *bots*.
* *Honeypots*: Son trampas digitales diseñadas para atrapar a los *bots* maliciosos. Se presentan como recursos "vulnerables", como enlaces ocultos o páginas web con código susceptible a ataques, que son invisibles para los usuarios humanos. Al colocar estos recursos en un sitio web, los administradores pueden identificar a los rastreadores que buscan explotar estas vulnerabilidades [20].
* Enfoques basados en pruebas de comportamiento: Utilizan pruebas online para clasificar a los usuarios y detectar *bots*. Evalúan la capacidad de los usuarios para realizar tareas que requieren procesamiento de lenguaje natural, representación del conocimiento, razonamiento y aprendizaje automático. Algunos ejemplos incluyen CAPTCHA y variantes más sofisticadas como reCAPTCHA de Google, BeCAPTCHA y CAPTUTE. Estas pruebas se basan en el concepto de Test de Turing, el cual busca determinar si una máquina puede exhibir un comportamiento inteligente indistinguible del de un humano.
* Enfoque basado en aprendizaje automático: Es el más utilizado en la actualidad, se basa en una variedad de algoritmos de minería de datos. Estos algoritmos permiten a las máquinas aprender a reconocer patrones complejos y tomar decisiones inteligentes basadas en los datos. Las actividades clave dentro de este enfoque incluyen: clasificación, agrupamiento, detección de anomalías, reducción de dimensionalidad, extracción y selección de características. Estas actividades se implementan a través de algoritmos de aprendizaje supervisado, no supervisado, por refuerzo y profundo.
* Enfoque basado en minería de datos complejos: En algunos casos, la detección de actividades de *bots* requiere el uso de técnicas de procesamiento de datos complejos. Los tipos de datos complejos incluyen secuencias, grafos, datos multimedia, datos de texto, datos espaciales, datos web y *streams*. Los algoritmos de última generación se clasifican en función del tipo de datos complejos que procesan, como la minería de secuencias, basada en grafos y otros tipos de datos.

### 1.3.1-Aprendizaje automático en la detección de *bots*

La minería de datos es una de las técnicas para detectar *bots* malignos. Se basa en la extracción de información valiosa a partir de grandes volúmenes de datos [8]. Un aspecto fundamental de la minería de datos es la clasificación, que juega un papel crucial en la detección de *bots*. La clasificación permite identificar si un usuario es un *bot* malicioso o un humano, lo que facilita la detección y el análisis de estos. De estos datos, se obtienen tendencias, relaciones y patrones ocultos a través de algoritmos de aprendizaje automático, que de otro modo serían imperceptibles [8, 21, 22].

El aprendizaje automático, permite a las máquinas aprender de los datos sin necesidad de ser programadas explícitamente [8]. En este campo, la creación de modelos que identifiquen patrones sospechosos en el comportamiento online y permitan diferenciar entre *bots* y humanos se convierte en un arma fundamental contra las amenazas digitales [5, 23]. Entre los tipos de aprendizaje se encuentran [7, 9, 24]:

* Aprendizaje Supervisado: Se basa en el entrenamiento de modelos con conjuntos de datos etiquetados, donde se conoce la clase a la que pertenece cada instancia. Los modelos aprenden a identificar las características que diferencian a los *bots* de los humanos y a predecir la clase de nuevas instancias.
* Aprendizaje No Supervisado: Se aplica a conjuntos de datos sin etiquetas predefinidas. Los algoritmos buscan patrones ocultos en los datos y agrupan las instancias con características similares. Esta técnica permite descubrir comportamientos sospechosos sin necesidad de una clasificación previa.
* Aprendizaje Profundo: Inspirado en la estructura del cerebro humano, utiliza redes neuronales con múltiples capas para extraer información de los datos. El aprendizaje profundo puede procesar información compleja y encontrar patrones que otros métodos no son capaces de identificar.

## 1.4-Algoritmos de aprendizaje supervisado en la detección de bots

Como se destaca en [1, 10], los algoritmos de aprendizaje supervisado sobresalen en la tarea de detección de *bots*, ya que aprenden a partir de datos etiquetados, que se traduce en una alta tasa de aciertos en la identificación de estas entidades. Dentro del amplio abanico de algoritmos de aprendizaje supervisado, existen varios que se han destacado por su efectividad en la detección como *K-Nearest Neighbors* (KNN), *Naive Bayes, Support Vector Machine* (SVM) y en especial los basados en árboles y bosques de decisión como *Random Forest* y *Ada Boost [9]*.

La investigación en este campo ha llevado a la construcción de modelos de dominio, como el presentado en [9] y representado en la figura 4. Este modelo expone la relación entre los conceptos clave relacionados con los *bots* y su impacto actual en diversos entornos, especialmente en el comercio electrónico. Se resalta el rol crucial del aprendizaje automático como una herramienta de la minería de datos para la detección de *bots* maliciosos.

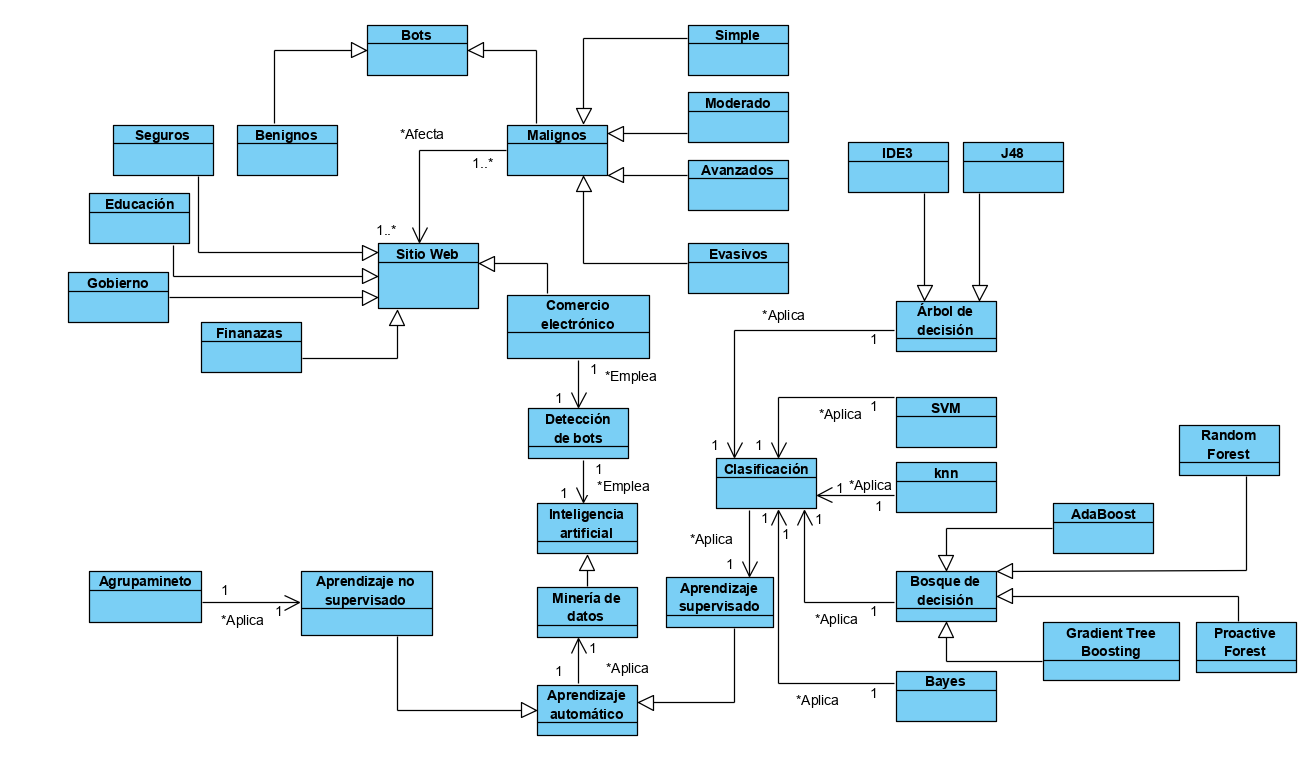


Figura 4 Modelo de dominio tomado de [9]

En el siguiente apartado, se expondrán con mayor detalle algunos de los algoritmos de aprendizaje supervisado utilizados en la detección de *bots*.

### 1.4.1-Naive Bayes

El algoritmo *Naive Bayes*, inspirado en el Teorema de Bayes, constituye una buena herramienta para clasificar datos basados en probabilidades. Este se nutre de un conjunto de datos etiquetados, donde cada instancia representa un evento independiente. Cada evento tiene una probabilidad de ocurrencia que representa su probabilidad previa, independientemente de la ocurrencia de otros [10]. Su esencia reside en la independencia de los eventos, obteniendo buenos resultados con características totalmente independiente y con características funcionalmente dependientes como se demostró en [25].

### 1.4.2-KNN

El algoritmo KNN (*K-Nearest Neighbors*, K vecinos más cercanos) es un algoritmo no paramétrico, utilizado principalmente en problemas de clasificación. Este algoritmo se basa en la idea de que los puntos de datos cercanos en un espacio de características tienen una alta probabilidad de pertenecer a la misma clase [6, 10].

Para clasificar un punto de datos nuevo, el algoritmo KNN calcula la distancia entre este punto y todos los puntos de datos en el conjunto de entrenamiento, seleccionando los "K" vecinos más cercanos. La clasificación se realiza asignando al nuevo punto la etiqueta de la clase que predomina entre sus "K" vecinos más cercanos [6, 10].

### 1.4.3 -Suport Vector Machine

El algoritmo *Support Vector Machine* (SVM) es un clasificador que busca encontrar el hiperplano que mejor separa los datos en el espacio, de modo que las clases queden lo más separadas posible con un margen entre ellas. Este margen está delimitado por dos límites, y la distancia entre estos límites es la longitud de un segmento perpendicular a ellos. Durante el entrenamiento, el algoritmo SVM busca maximizar esta distancia para lograr una separación óptima de las clases [9].

Al clasificar una nueva instancia, el algoritmo determina la clase basándose en el vector que proporciona el mayor soporte para clasificarla, es decir, el vector que se encuentra más cerca del límite de decisión [9].

### 1.4.4-Árbol de decisión

Los árboles de decisión son estructuras de datos jerárquicas que representan una forma intuitiva de modelar relaciones complejas en datos [8, 9, 22]. Utilizados tanto para tareas de clasificación como de regresión, estos árboles se asemejan a un diagrama de flujo, donde cada nodo interno representa una pregunta sobre una característica específica de los datos, y cada rama representa una posible respuesta a esa pregunta. El recorrido a través del árbol, desde la raíz hasta una hoja, conduce a una decisión final: una clase en el caso de la clasificación o un valor predicho en el caso de la regresión [8, 9].

La construcción del árbol de decisión es un proceso recursivo. En cada nodo, se selecciona la característica que mejor divide los datos, utilizando criterios como la ganancia de información para medir qué tan bien un atributo separa los ejemplos de datos o el índice Gini para buscar la impureza máxima de los datos. Este proceso continúa hasta que se alcanza una condición de parada, como la profundidad máxima del árbol, el número mínimo de ejemplos en un nodo o un umbral de impureza [9].

Entre los algoritmos basados en árboles de decisión que obtienen buenos resultados en la detección de *bots* se encuentran [9]:

* ID3: Construye modelos predictivos mediante árboles de decisión de forma recursiva, seleccionando el atributo que mejor separa los datos según la ganancia de información. Se detiene cuando todos los datos pertenecen a la misma clase. Su simplicidad lo hace atractivo para fines educativos, pero presenta desventajas como el sobreajuste y la incapacidad para manejar atributos no categóricos.
* J48: Es una implementación del algoritmo C4.5, desarrollado en Java y parte de la herramienta WEKA. Representa una evolución del ID3, utilizando la proporción de ganancia para seleccionar atributos y permitiendo el manejo de datos continuos. Su condición de parada se basa en un umbral de instancias a dividir. J48 también incluye un proceso de poda pesimista para mejorar la eficacia del modelo al eliminar ramas innecesarias.

La simplicidad y la transparencia de los árboles de decisión son sus mayores ventajas. Su estructura jerárquica y el uso de preguntas fáciles de entender los hacen fácilmente interpretables, incluso para personas sin experiencia en aprendizaje automático. Sin embargo, los árboles de decisión también presentan algunas desventajas. Su tendencia al sobreajuste, es decir, aprender el conjunto de datos de entrenamiento demasiado bien y perder capacidad de generalización, es un problema conocido. Además, las decisiones en los árboles de decisión se basan en una sola característica a la vez, ignorando posibles interacciones entre las características que pueden proporcionar información valiosa para el modelo [8, 9].

### 1.4.5-Bosque de decisión

Los bosques de decisión son una técnica de aprendizaje *ensemble* que aprovecha la sabiduría colectiva de múltiples árboles de decisión para mejorar la precisión de la predicción y superar las limitaciones individuales de sus integrantes [12]. La efectividad de un bosque de decisión depende de dos principios fundamentales: diversidad y poder predictivo. La diversidad se refiere a qué tan diferente debe ser los árboles del bosque. La precisión, por otro lado, implica que cada árbol individual debe ser lo suficientemente preciso como para superar un modelo de predicción aleatoria [13].

Los algoritmos basados en bosques de decisión se han vuelto muy populares en la detección de *bots*, especialmente métodos como *Adaptive* *Boosting* (*AdaBoost*) y *Random Forest* (RF) [9]. Estos algoritmos aprovechan la capacidad del bosque de decisión para combinar la información de múltiples árboles y detectar patrones sutiles que pueden pasar desapercibidos para un modelo individual. Su capacidad de generalización los convierten en una herramienta valiosa para identificar comportamientos anómalos y distinguir entre usuarios humanos y *bots* [1]. A continuación, se explican brevemente los algoritmos basados en bosques de decisión que se encuentran implementados en el módulo:

1. *Random Forest*: Algoritmo de aprendizaje automático que utiliza un conjunto de árboles de decisión, cada uno entrenado con subconjuntos aleatorios de datos y atributos, para realizar predicciones más robustas y precisas. Este enfoque reduce el riesgo de sobreajuste al forzar a los árboles a considerar diferentes combinaciones de características y al combinar sus predicciones mediante un sistema de votación mayoritaria, lo que suaviza variaciones y mejora la estabilidad del modelo [14, 26].
2. *Proactive Forest*: Es un algoritmo que construye árboles de decisión secuencialmente, utilizando la información de los árboles previos para guiar la construcción de los siguientes. Esto permite un control más preciso sobre la diversidad del modelo, al favorecer la selección de atributos menos utilizados y equilibrar la importancia de características relevantes y no utilizadas. La selección del atributo óptimo en cada nodo se basa en su probabilidad de selección, calculada a partir de la impureza de disminución media, que mide la importancia de las características[12, 14, 27].
3. *Ada Boost*: Algoritmo de "Adaptive Boosting" que combina clasificadores débiles, como los "muñones" (árboles de decisión simples), para crear un clasificador fuerte. En cada iteración se enfoca en las instancias mal clasificadas por los clasificadores existentes, incrementando su peso para que sean más relevantes en el entrenamiento del siguiente clasificador. Los clasificadores nuevos se combinan con los existentes, ponderados de acuerdo con su precisión. Este proceso adaptativo, que aprende de los errores de los clasificadores anteriores, permite mejorar continuamente la precisión global del conjunto [28].
4. *Gradient Tree Boosting:* Combina múltiples árboles de decisión débiles para crear un modelo predictivo fuerte. Comienza con un árbol simple y, en cada iteración, crea un nuevo árbol que se enfoca en corregir los errores del modelo actual. Los nuevos árboles aprenden de los residuos, la diferencia entre las predicciones actuales y los valores reales, mejorando la precisión del modelo con cada iteración. Este proceso repetitivo permite al modelo capturar patrones complejos en los datos, ajustándose gradualmente a las áreas donde los árboles anteriores cometieron errores [29].

A pesar de que los algoritmos de aprendizaje automático aprenden el comportamiento de datos de entrada para crear un modelo que prediga el valor de instancias futuras, todavía existen inconvenientes por resolver en ellos. Estos modelos requieren grandes cantidades de datos para funcionar correctamente, debido a que a medida que la cantidad de características aumenta, el espacio de posibles datos se expande exponencialmente. El problema es que, en la práctica, la cantidad de datos disponibles para entrenar un modelo a menudo es limitada. Esto lleva a que los algoritmos aprendan patrones específicos del conjunto de datos de entrenamiento, lo que puede resultar en un rendimiento deficiente cuando se enfrentan a datos nuevos [15].

## 1.5- Meta aprendizaje

En el ámbito del aprendizaje automático, los algoritmos tradicionales a menudo se enfrentan a desafíos como la necesidad de grandes conjuntos de datos, la sobre adaptación y la dificultad para generalizar a nuevas tareas. Para superar estas limitaciones, surge el meta aprendizaje, una rama que se centra en la habilidad de aprender a aprender. Los algoritmos de este tipo extraen experiencias de múltiples escenarios de aprendizaje, a menudo pertenecientes a tareas relacionadas, y a partir de dicha experiencia mejoran el aprendizaje futuro. Es importante resaltar que a diferencia de los enfoques tradicionales , en los que las tareas se resuelven desde cero utilizando un algoritmo de aprendizaje fijo, el meta aprendizaje tiene como objetivo mejorar el algoritmo de aprendizaje en sí [16].

Su núcleo reside en la adquisición de meta conocimiento. Este meta conocimiento no se limita a la información específica de una tarea, sino que engloba la comprensión del proceso de aprendizaje en sí, transmitiendo la información recopilada de una tarea a otra. Por ejemplo, un algoritmo de meta aprendizaje puede aprender a identificar qué tipo de datos son más relevantes para una tarea determinada o qué estrategias de entrenamiento son más efectivas, a partir de tres enfoques principales [30]:

1. Modelos adaptativos: El meta aprendizaje puede optimizar el proceso de aprendizaje al ajustar de manera dinámica los hiperparámetros del algoritmo. Estos ajustes se basan en información obtenida durante el entrenamiento del modelo, permitiendo una mayor eficiencia y precisión. Este proceso se puede realizar a través de una arquitectura interna especializada o mediante la influencia de un modelo de meta aprendizaje externo.

2. Modelos basados en optimización: En lugar de simplemente ajustar los hiperparámetros, busca optimizar el proceso de aprendizaje a través del entrenamiento de un optimizador interno. Este optimizador utiliza información sobre el estado de la optimización para determinar el mejor paso a seguir en cada iteración del aprendizaje, mejorando la eficiencia y la calidad del resultado final.

3. Aprendizaje basado en métricas: Mejora el procesamiento de los datos mediante la creación de representaciones métricas eficientes. Esta técnica busca extraer información clave de los datos de entrada, como características relevantes, para generar una representación más efectiva que facilite el aprendizaje. Este meta conocimiento facilita la comparación de datos y la detección de patrones, mejorando la precisión y eficiencia del proceso de aprendizaje.

## 1.6- Métricas de caracterización de datos

Las métricas son un valor numérico o nominal que se le asigna a determinados atributos o características, para una comparación con datos de referencia previamente analizados, que permita sacar conclusiones sobre el aspecto en cuestión [31]. En las secciones siguientes se describen la implementación de las nuevas métricas para la caracterización

1.6.1-Rango Intercuartil (IQR)

El Rango Intercuartil (IQR) es una medida para comprender la dispersión de datos, especialmente cuando se trata de conjuntos de datos que pueden contener valores atípicos, se enfoca en la dispersión de la mitad central de los datos, ignorando los valores más bajos y más altos. Su cálculo se basa en dividir los datos ordenados en cuatro partes iguales, representadas por los cuartiles. El primer cuartil (Q1) marca el punto donde el 25% de los datos son menores o iguales a él, mientras que el tercer cuartil (Q3) indica el punto donde el 75% de los datos son menores o iguales a él. El IQR, entonces, se calcula como la diferencia entre Q3 y Q1 (IQR = Q3 - Q1) [32].

### 1.6.2- Desviación Media Absoluta (MAD)

La Media Absoluta de Desviaciones (MAD) es una medida para analizar la dispersión de los datos, que no se ve afectada por valores atípicos. A diferencia de la desviación estándar, que se basa en la suma de las diferencias cuadradas, la MAD se calcula como el promedio de las diferencias absolutas entre cada valor y la media del conjunto de datos. Esto significa que no se ve influenciada por valores extremos y le da a cada desviación su lugar proporcional en el resultado, siendo una medida más fiable en conjuntos con valores atípicos [33].

### 1.6.3- Entropía de Shannon

La entropía de Shannon es una medida de la incertidumbre asociada a la información. Se utiliza en teoría de la información y en campos relacionados, como el aprendizaje automático y la estadística, para cuantificar la cantidad de información o desorden en un conjunto de datos. Su capacidad para medir la incertidumbre o la aleatoriedad en los datos permite identificar patrones de comportamiento sospechosos, especialmente cuando se trata de automatización. La fórmula de la entropía de Shannon se define en la ecuación 1 [17]:

Ecuación 1 Entropía de Shannon

Donde:

* H(X) es la entropía de la fuente de información X.
* P(x) es la probabilidad de que ocurra el evento x.
* X es el número total de eventos posibles.

En esencia, esta fórmula calcula la incertidumbre promedio de una variable, tomando en cuenta la probabilidad de cada evento posible. Cuanto mayor sea la incertidumbre (mayor entropía), mayor es la aleatoriedad del comportamiento [17].

### 1.6.4- Distancia de Mahalanobis

La distancia de Mahalanobis es una métrica para analizar datos multivariados, ofreciendo una medida de similitud entre dos variables aleatorias. A diferencia de la distancia euclidiana tradicional, que considera cada variable de forma independiente, la distancia de Mahalanobis considera la relación entre las variables, lo que la convierte en una medida más precisa para la comparación de datos multidimensionales.

En esencia, la distancia de Mahalanobis calcula la distancia de un punto de datos al centroide de un conjunto de datos, teniendo en cuenta la dispersión y la correlación de las variables. El centroide, que representa la media general de los datos multivariados, actúa como un punto de referencia para determinar la distancia de un punto específico. Cuanto mayor sea la distancia de Mahalanobis, más alejado estará el punto de datos del centroide, lo que indica una mayor diferencia en términos de la distribución general del conjunto de datos [34].

La fórmula para calcular la distancia de Mahalanobis entre dos puntos de datos, x e y, se define en la ecuación 2 [34]:

Ecuación 2 Distancia de Mahalanobis

Donde:

• son los vectores de datos multidimensionales.

• Σ es la matriz de covarianza del conjunto de datos, que captura la correlación entre las variables del conjunto de datos.

### 1.6.5- CUSUM

La técnica CUSUM (*Cumulative Sum*) se erige como una buena herramienta en el control estadístico de calidad, dedicada a la vigilancia de posibles modificaciones en los procesos. Su principal aplicación radica en la detección de variaciones en la media del proceso. Se basa en la suma acumulada de las desviaciones entre un conjunto de muestras y un valor de referencia preestablecido. En esencia, se calcula la diferencia entre cada observación y el valor objetivo, y se acumula esta diferencia a lo largo del tiempo [35].

Una de las fortalezas más notables de CUSUM reside en su capacidad para identificar cambios sutiles en la media del proceso, incluso aquellos que son apenas perceptibles, que podrían pasar desapercibido para otros métodos de control de calidad [35].

### 1.6.6- Distancia de Jensen\_Shannon

La distancia de Jensen-Shannon (JSD) es una medida de disimilitud entre dos distribuciones de probabilidad que ofrece una alternativa más fácil de interpretar a la divergencia de Kullback-Leibler (KL), debido a que es simétrica y siempre tiene un valor finito, lo que la convierte en una medida más útil para comparar distribuciones de probabilidad. Cuanto mayor sea la diferencia, más diferentes serán las dos distribuciones [36].

Se define como la mitad de la divergencia KL entre la distribución promedio y cada una de las dos distribuciones originales, como se muestra en la ecuación 3 [36]:

Ecuación 3 Distancia de Jensen\_Shannon

Donde M es la distribución promedio (ecuación 4) y KL (P, Q) es la divergencia KL entre las distribuciones P y Q (ecuación 5) [36]:

Ecuación 4 Distribución promedio

Ecuación 5 Divergencia KL

## 1.7-Módulo de detección de *bots*

El módulo de detección de *bots* malignos es presentado en [9], como un enfoque para detectar *bots* malignos basado en aprendizaje automático que consta de 3 etapas, dentro de las cuales se desarrollan diferentes enfoques y procesos para la detección de *bots* con el objetivo de formar un enfoque integral. Como se puede apreciar en la figura 5 las etapas actuales son [9]:

* Etapa 1: Detección de potenciales usuarios *bots* malignos: En esta etapa se extraen las características de aquellos usuarios que acceden a un sitio web. Para detectar si en un conjunto determinado hay usuarios potenciales amenazas o no, a partir de un proceso de meta aprendizaje. Los usuarios considerados como potenciales amenazas son analizados en la próxima etapa. Comprende los procesos de captura de características de usuario y clasificación por meta componente.
* Etapa 2: Detección de instancias de usuarios *bots* malignos: Esta etapa se centra en clasificar individualmente cada instancia de los usuarios potenciales *bots* malignos recibidos en *bot* maligno o humano. Empleando para ello un modelo de clasificación, entrenado sobre una base de datos de ataques de *bots* malignos. Esta etapa comprende los procesos de entrenar un algoritmo de aprendizaje automático y clasificar datos de potenciales *bots*.
* Etapa 3: Caracterización de datos de usuarios *bots* malignos y humanos: En esta etapa se caracterizan las instancias de usuarios que interactúan con la plataforma web, estando dentro de la misma. A partir de todos los datos ya clasificados en la anterior etapa, empleando además diversas medidas de complejidad. Almacenando los resultados de esta caracterización en una base de hechos. Esta etapa incluye el proceso de caracterizar usuarios por meta componente. .

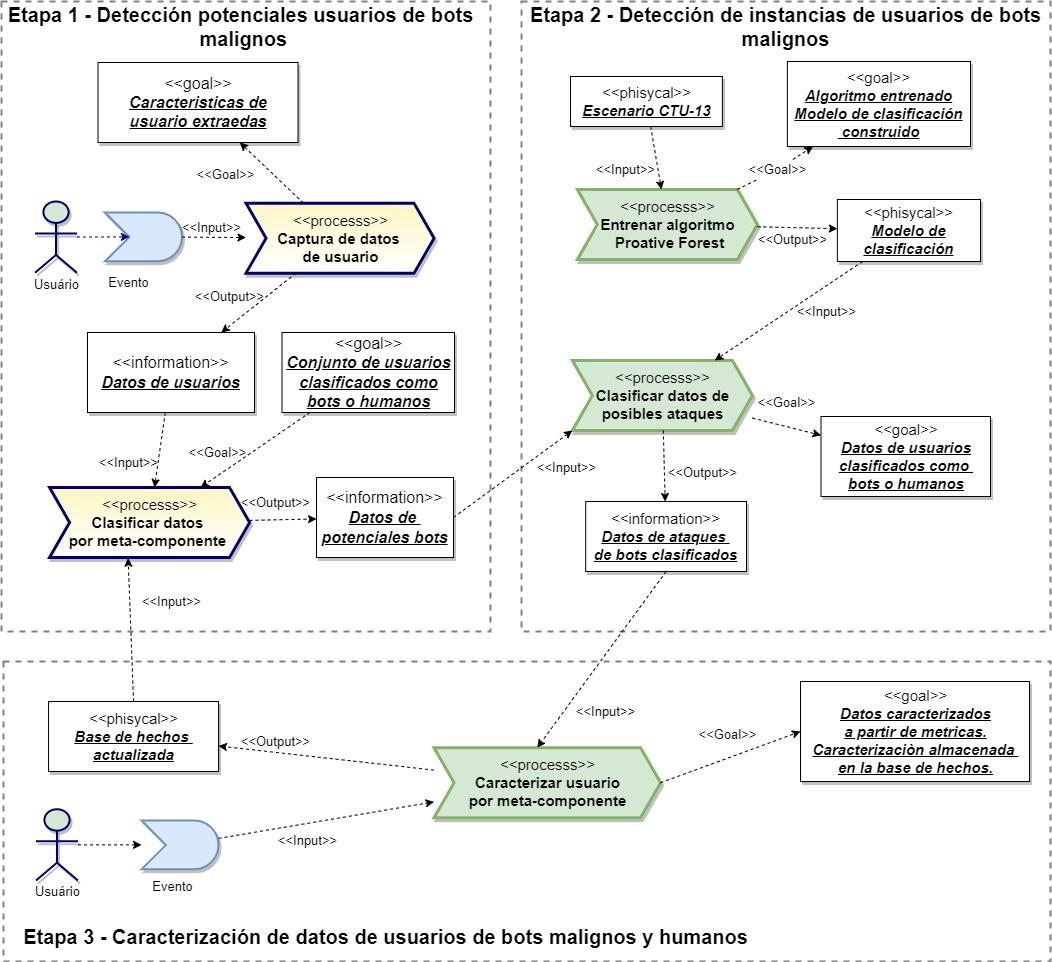


Figura 5 Etapas del módulo [9]

A pesar de los avances logrados en el módulo, se identifican deficiencias que han sido señaladas en trabajos previos, como las tesis de: Hélder Joao Chissingui [7], Jorge Alberto Pacheco Senard y Dariel Luis Rubio Oviedo [10], y Daniel Pardo Echevarría [9] . En particular, la etapa 3 del proceso, que se centra en la caracterización de datos de usuarios *bots* malignos y humanos, presenta limitaciones significativas.

Una de las principales desventajas radica en la dependencia de datos etiquetados para la caracterización y clasificación. Aunque el uso de datos etiquetados suele ofrecer buenos resultados, esta práctica conlleva ciertas restricciones. La base de hechos utilizada en esta etapa se basa en métricas que requieren datos previamente etiquetados. Esta dependencia puede resultar en una falta de flexibilidad y en una disminución de la precisión en la detección de instancias previamente desconocidas, por lo que en [7] se recomienda cambiar las métricas empleadas por otras que sean independientes de los datos etiquetados.

## 1.8-[Conclusiones parciales](#_Toc128048)

Al realizar un análisis del marco teórico, se puede concluir que:

* Los modelos de aprendizaje supervisado son una herramienta para la detección de *bots*, pero tienen limitaciones.
* El meta aprendizaje resuelve las limitaciones del aprendizaje automático.
* El empleo de un enfoque de meta aprendizaje, favorece el rendimiento de técnicas de aprendizaje automático convencionales, permitiendo que junto al análisis de métricas de caracterización de datos pueda ofrecer un enfoque eficaz para las tareas de detección.
* Las métricas de caracterización de datos permiten evaluar la distribución de los datos y detectar patrones inusuales que podrían ser indicadores de actividad anómala.
* La dependencia de datos etiquetados para el proceso de clasificación que realiza el módulo, puede provocar una disminución de la precisión en casos previamente desconocidos.

# Capítulo 2: Desarrollo del nuevo enfoque de detección

Este capítulo se adentra en la implementación de nuevas técnicas dentro de la etapa 3 del módulo, con el propósito de explorar nuevas opciones para mejorar la precisión en la distinción entre humanos y *bots*. Además alguno de estos nuevos procedimientos, darán cumplimiento a recomendaciones surgidas del trabajo del equipo de investigación anterior.

## 2.1-Detalles de la etapa 3 del módulo de detección de *bots* malignos

La tercera etapa del módulo de detección de *bots* malignos está compuesta por un meta componente que se encarga de llevar a cabo el proceso de caracterización de usuarios. El proceso se describe en la figura 6, donde los elementos de color verde serán modificados para tratar de superar las deficiencias de la etapa. El flujo comienza una vez que se ha completado el proceso de clasificación llevado a cabo por un modelo de aprendizaje supervisado en la etapa anterior y está dividido en varias fases [9]:

1. Los datos resultantes de la clasificación se almacenan en un fichero que contiene toda la información relevante. Este registro proporciona una base para futuras referencias y análisis. Posteriormente, estos datos clasificados se cargan desde el fichero.
2. Se extrae un subconjunto de los datos que han sido clasificados como humano. Para enriquecer el análisis, se simulan cambios aleatorios en algunas características de estos datos. Con el subconjunto de datos humanos listo, se procede a construir un árbol de decisión mediante el entrenamiento del algoritmo con las instancias cargadas. Utilizando el árbol de decisión construido, se reclasifica el subconjunto de datos que corresponde a los usuarios humanos para evaluar la efectividad del modelo en la identificación correcta de los usuarios.
3. Se calculan diversas métricas que caracterizan los datos reclasificados. Estas medidas permiten comprender la complejidad y las características del conjunto de datos en cuestión. Los valores obtenidos de estas métricas se almacenan en una lista destinada a caracterizar el conjunto de datos.
4. Se realiza una nueva clasificación del subconjunto de datos humanos utilizando el modelo obtenido. Este paso asegura que la clasificación sea coherente y esté basada en el modelo más actualizado. Tras esta reclasificación, se revisan los resultados y se asigna una etiqueta correspondiente a la descripción del conjunto de datos. Si alguna instancia es clasificada como *bot* maligno, se le asigna el valor de clase "usuarios *bots*". En caso contrario, si todos los usuarios continúan siendo clasificados como humanos, se les asigna el valor "usuarios humanos".
5. Se guarda la caracterización en una base de hechos, actualizando la clase en aquellas filas que presenten las mismas métricas que la descripción actual, siempre que el valor de clase sea diferente. Este último paso es vital para mantener la integridad y actualidad de la base de datos frente a nuevas interacciones y cambios en el comportamiento del usuario.

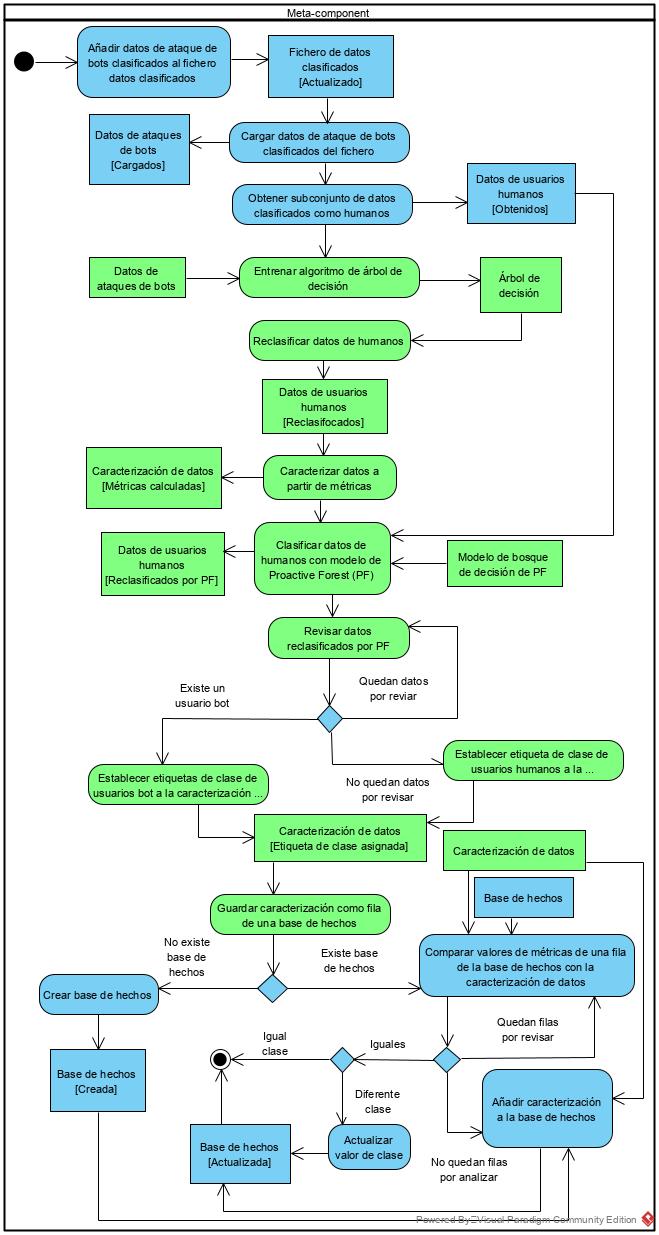


Figura 6 Ejecución del proceso de la etapa 3 [9]

## 2.2-Requisitos funcionales

A partir del proceso de caracterización de datos de la tercera etapa se presentan los siguientes requisitos funcionales del sistema:

* El sistema debe recibir un conjunto de datos clasificados como humanos.
* El sistema debe seleccionar un subconjunto de datos a modificar.
* El sistema debe simular cambios no aleatorios en las características del subconjunto.
* El sistema debe calcular métricas de caracterización de datos para el conjunto.
* El sistema debe asignar etiquetas de clase al conjunto tras los cambios.
* El sistema debe guardar la caracterización del conjunto en una base de hechos. Para esto debe comparar las métricas nuevas con las existentes en la base de hechos, si se encuentran coincidentes, debe actualizarlas y si no guardar la caracterización.

## 2.3 - Simulación de cambios no aleatorios

Debido a la imposibilidad de desplegar el módulo en un entorno real en esta etapa se implementa una función específica que tiene como objetivo simular cambios en los datos. Actualmente, estos cambios se llevan a cabo de manera aleatoria, lo que puede introducir variabilidad y falta de control en los resultados. Para abordar esta limitación, se ha decidido seguir las recomendaciones establecidas en [9], que sugieren el desarrollo de un método más estructurado y determinista. Este enfoque permitirá simular los cambios de manera coherente y reproducible, lo cual es fundamental para obtener resultados confiables.

El método propuesto se basa en el seudocódigo descrito en la Tabla 1, que detalla los pasos necesarios para llevar a cabo la simulación de cambios en los datos. Sin embargo, es importante destacar que, si se aplica Análisis de Componentes Principales (PCA) a los datos, se presenta un desafío significativo: no es posible identificar métodos que modifiquen características específicas de los datos originales. Esto se debe a que las características originales han sido transformadas y reducidas a un nuevo espacio dimensional, donde solo se conservan las dimensiones resultantes del PCA.

Tabla 1 Seudocódigo de la función simulate\_positives

|  |  |  |  |
| --- | --- | --- | --- |
| **Seudocódigo** | | | |
|  | Función simulate\_positives(x, porcent\_cant\_min, porcent\_cant\_max): | | |
| 1 |  | cant = longitud(x). | |
| **2** |  | cant\_atributes = cantidad\_de\_columnas(x) | |
| **3** |  | cant\_modif = convertir\_a\_entero(porcent\_cant\_min \* cant) | |
| **4** |  | max\_value = máximo(self.x\_instances) | |
| **5** |  | min\_value = mínimo(self.x\_instances) | |
| **6** |  | range\_value = max\_value - min\_value | |
| **7** |  | para index desde 0 hasta cant\_modif - 1 hacer: | |
| **8** |  |  | colum = index MOD cant\_atributes: |
| **9** |  |  | new\_value = min\_value + (index / cant\_modif) \* range\_value |
| **10** |  |  | x[index][colum] = new\_value |
| **11** |  | fin del ciclo | |
| **12** |  | retornar x | |
| **13** | fin de la función | | |

Como se puede ver en la Tabla 1 primero se evalúa cuántas instancias hay en el conjunto, así como la cantidad de atributos o columnas que hay en él. A partir de ahí, la función calcula cuántas instancias necesita modificar. Esto se hace multiplicando el porcentaje mínimo que se le pasa como argumento (porcent\_cant\_min) por la cantidad total de instancias, y luego convierte ese resultado a un número entero. Así, se establece cuántas instancias serán objeto de modificación.

Luego, para poder realizar los cambios de manera efectiva, se determina los valores máximos y mínimos presentes en el conjunto de datos. Con esta información, se calcula el rango total restando el valor mínimo del máximo.

Con todo esto preparado, la función entra en un ciclo que se repetirá tantas veces como instancias haya decidido modificar. En cada iteración del ciclo, se elige una columna específica para modificar utilizando una operación simple que asegura que las modificaciones se distribuyan entre todas las columnas disponibles.

Después, para cada instancia que va a ser modificada, la función genera un nuevo valor. Este nuevo valor no es aleatorio, sino que se calcula de manera que esté dentro del rango definido anteriormente. Comienza desde el valor mínimo y avanza hacia el máximo, creando una progresión suave y equilibrada.

Finalmente, este nuevo valor se asigna a la posición correspondiente en el conjunto de datos. Una vez que todas las modificaciones han sido realizadas y todas las instancias han sido ajustadas, la función devuelve el conjunto de datos ya modificado.

## 2.4-Implementación de métricas de caracterización de datos

En [9] el autor plantea la fase de aplicación de métricas de caracterización, fundamentada en un conjunto de métricas que miden la complejidad de los datos a partir de un conjunto previamente etiquetado. Esta dependencia en datos etiquetados, aunque útil, presenta varios inconvenientes que pueden comprometer la validez y aplicabilidad de los resultados obtenidos. En este sentido, siguiendo las recomendaciones planteadas en [7], se ha decidido reestructurar la base de hechos para implementar únicamente métricas de caracterización que no dependan de la etiqueta de clase. Esta transición hacia un enfoque más independiente favorecerá que las métricas sean aplicables en una variedad más amplia de contextos y no estén limitadas por la disponibilidad o la calidad de los datos etiquetados.

Para llevar a cabo esta reestructuración, se han seleccionado y explicado las nuevas métricas en la [sección 1.6](#sección1_7) . A continuación, se detalla la implementación de estas métricas, empleadas para caracterizar y guardar los datos de los usuarios potenciales amenazas en la base de hechos. Para esto se ha utilizado la biblioteca de Python conocida como *Scipy* [37], reconocida por su eficiencia en la implementación de algoritmos matemáticos, lo que permite realizar cálculos complejos de manera optimizada. Entre las métricas implementadas se incluyen la distancia de Mahalanobis, la distancia de Jensen-Shannon, la entropía de Shannon y el Rango Intercuartil (IQR), de esta última no fue necesaria la implementación puesto que la librería ya contaba con una versión funcional que no necesitaba adaptaciones. Además, para la Desviación Absoluta Media y el CUSUM, se desarrollaron fórmulas codificadas específicas que permiten su cálculo. Es importante destacar que en el caso de que se realice PCA, se calcularía la Desviación Media Absoluta (MAD) y el Rango Intercuartil (IQR) específicamente para el primer componente principal (PC1) obtenido.

Para el cálculo de algunas métricas es necesario la comparación de la distribución de los datos entrantes con la distribución de tráfico de red humano. Para facilitar este análisis, se creó una base de datos centralizada que recopila información de varios conjuntos de datos, garantizando así que se disponga de un conjunto representativo y consistente para las comparaciones necesarias. Esta base de datos se explica a profundidad en la [sección 3.1](#base_centralizada). Este conjunto de datos sirvió como referencia para calcular la distancia de Mahalanobis, la distancia de Jensen-Shannon y el valor base para el CUSUM. Además, para contextualizar mejor los resultados obtenidos de la entropía de Shannon, la distancia de Jensen-Shannon y CUSUM se aplicó un promedio ponderado utilizando como pesos el *ratio* de varianza explicada por cada componente, lo que permite reflejar adecuadamente la importancia relativa de cada uno en el análisis.

### 2.4.1- Implementación de la distancia de Mahalanobis (MD)

Se ha implementado una función para calcular la distancia de Mahalanobis (MD), que permite evaluar la cercanía de cada instancia en un conjunto de datos a un centroide definido por los datos de referencia, lo que permite identificar puntos que se desvían significativamente del comportamiento esperado.

Como se puede ver en el flujo descrito en la figura 7, primero se calcula el vector de medias y la matriz de covarianza del conjunto de datos normalizados. La inversión de esta matriz permite ajustar las distancias en función de la variabilidad y correlación entre las variables. Posteriormente se itera sobre cada fila del conjunto de datos entrante, calculando la distancia de Mahalanobis entre cada instancia y el centroide. Las distancias calculadas se almacenan en una lista, que posteriormente se convierte en un arreglo NumPy, que permitirá calcular la media.

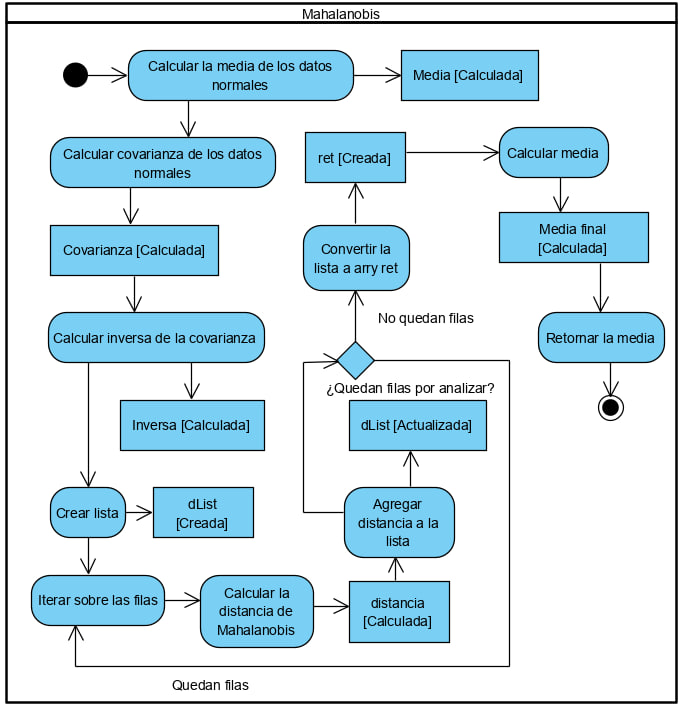


Figura 7 Diagrama de flujo de la distancia de Mahalanobis

### 2.4.2- Implementación de MAD

La MAD fue implementada a partir del flujo descrito en figura 8. Este procedimiento comienza con el cálculo de la media del conjunto de datos. A continuación, se determina la desviación absoluta de cada instancia respecto a esta media, lo cual permite evaluar cuánto se alejan los datos del promedio. Finalmente, se calcula la media de estas desviaciones absolutas, resultando en la Desviación Absoluta Media que proporciona una medida de la dispersión de los datos.

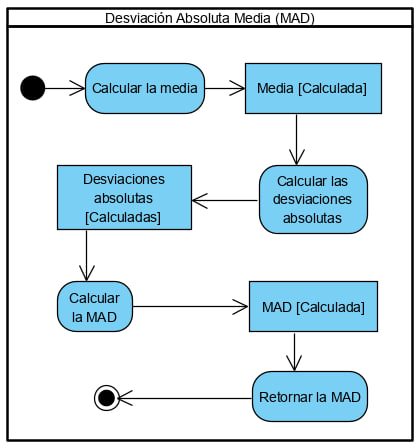


Figura 8 Diagrama de flujo de MAD

### 2.4.3- Implementación de la entropía de Shannon (SE)

Para el cálculo de la entropía de Shannon (SE) en conjuntos de datos continuos se lleva a cabo, un proceso denominado *binning* o agrupamiento. Este método convierte los datos continuos en categorías discretas, lo que simplifica el análisis y reduce significativamente el número de puntos a evaluar. Para determinar el número óptimo de *bins* a utilizar en este proceso, se evaluaron varios métodos establecidos, tales como el Método de Sturges [38], el método de Scott [39], el método de Freedman-Diaconis [40] y el método de Doane [41]. Tras una cuidadosa consideración, se optó por el método de Doane debido a su eficacia particular en el tratamiento de datos que no siguen una distribución normal. Este método es una extensión del Método de Sturges y tiene en cuenta factores como la asimetría y la curtosis en los datos, ajustando así el número de *bins* necesarios para reflejar con precisión las características inherentes del conjunto analizado.

Para calcular la Entropía de Shannon se sigue el flujo descrito en la figura 9, comenzando con la inicialización de una lista vacía llamada entropyArray, que almacenará los valores de entropía calculados para cada columna del conjunto de datos. Se itera sobre cada columna del conjunto para extraer los datos correspondientes y se generar un histograma utilizando np.histogram, especificando que el número de *bins* debe ser determinado por el método de Doane. Esto asegura que la discretización de los datos se realice de manera adecuada.

Una vez generado el histograma, se calcula la distribución de probabilidad dividiendo cada valor del histograma por la suma total del mismo. Esta normalización permite obtener una representación precisa de la probabilidad asociada a cada *bin*.

Con las probabilidades calculadas, se determina la entropía de Shannon correspondiente a esas probabilidades. Este valor se añade a la lista entropyArray, lo que permite acumular las entropías calculadas para cada columna. Finalmente, después de procesar todas las columnas, se procede a calcular la media ponderada de las entropías, como medida de la incertidumbre presente en los datos.

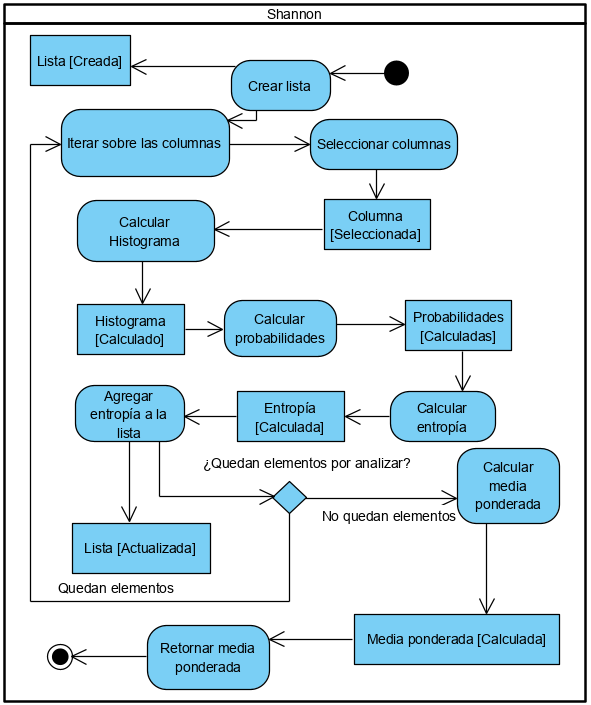


Figura 9 Diagrama de flujo de la entropía de Shannon

### 2.4.4- Implementación de la distancia de Jensen-Shannon (JS)

La distancia de Jensen-Shannon se utilizó para evaluar cómo se distribuyen los datos en comparación con una referencia normal. Como se ve en el flujo descrito en la figura 10, el proceso comienza con la inicialización de una lista vacía llamada distArray, que almacenará las distancias calculadas para cada columna de los datos. Se itera sobre cada columna del conjunto de datos, extrayendo tanto los datos originales como los datos normales correspondientes.

A continuación, se generan histogramas para ambos tipos de datos utilizando el método np.histogram, con el número de intervalos determinado por el método de Doane. Esto asegura que el número de *bins* se ajuste a la distribución de los datos. Con los histogramas listos, se crean distribuciones de probabilidad P y Q para los datos originales y los normales, respectivamente. Estas distribuciones se obtienen normalizando los histogramas, dividiendo cada valor por la suma total del histograma correspondiente.

Una vez calculadas las distribuciones, se procede a calcular la distancia entre las distribuciones P y Q, y este valor se añade a la lista. Finalmente, después de procesar todas las columnas, se procede al cálculo de la media ponderada para reflejar la importancia de cada componente.

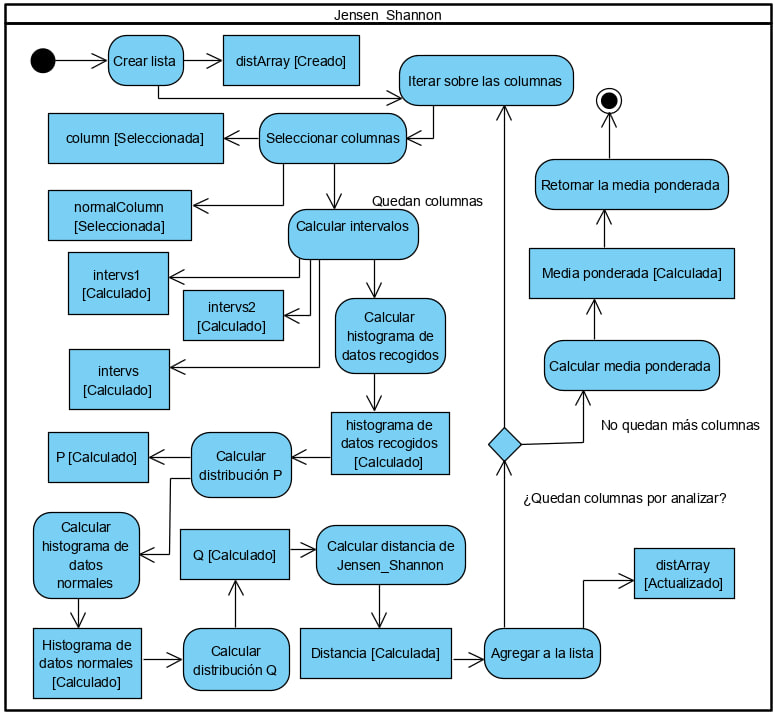


Figura 10 Diagrama de flujo de la distancia de Jensen Shannon

### 2.4.5- Implementación de CUSUM

Para el cálculo de CUSUM como se aprecia en el flujo de la figura 11, se procede a calcular tanto el CUSUM como el N\_CUSUM para cada columna del conjunto de datos. Al principio se inicializan dos listas vacías: cusumArray y nCusumArray, que almacenarán los valores de incremento y decremento calculados para cada columna, respectivamente. A continuación, se inicia un bucle que itera sobre cada columna del conjunto de datos. En cada iteración del ciclo se inicializan las variables cusum y nCusum en cero, y se extraen los datos correspondientes de la columna actual en los datos de referencia y los datos analizados. Se calcula la media de la columna normal utilizando np.mean, que sirve como referencia para determinar si los valores acumulados están por encima o por debajo de lo esperado.

El siguiente paso consiste en calcular el CUSUM. Se utiliza otro bucle que itera sobre cada elemento de la columna actual. En cada iteración, se actualiza el valor de cusum sumando la diferencia entre el valor actual de la columna y la media calculada. Si este valor acumulado se vuelve negativo, se reinicia a cero, asegurando así que solo se consideren incrementos positivos. De manera similar, se calcula el N\_CUSUM, que acumula las diferencias negativas, utilizando la función min para mantener un registro de los decrementos.

Una vez completado el cálculo para cada columna, los resultados finales de CUSUM y N\_CUSUM se añaden a sus respectivas listas y finalmente, se procede a calcular la media ponderada de los valores acumulados.

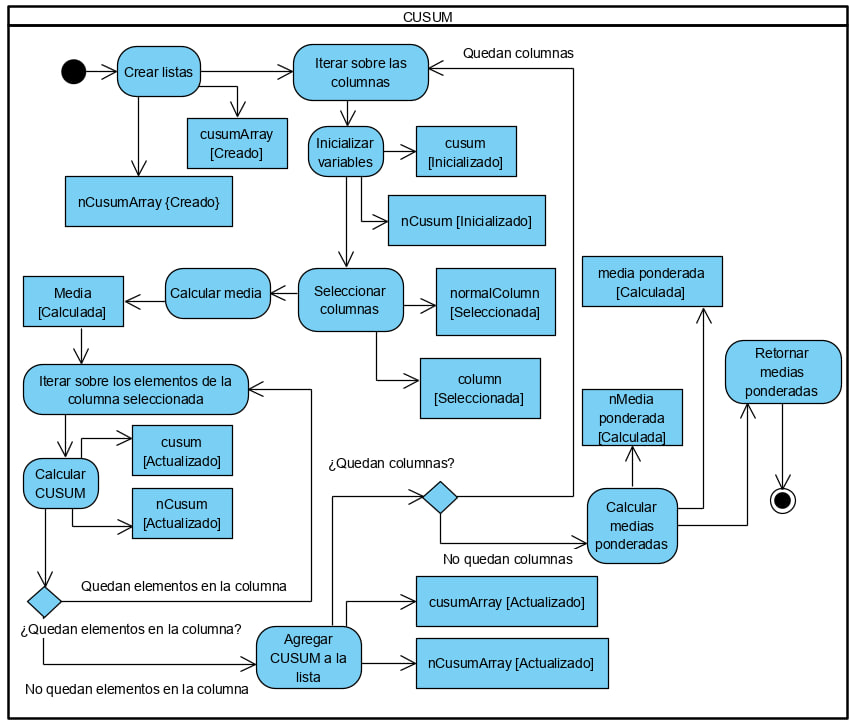


Figura 11 Diagrama de flujo de CUSUM

## 2.5-Algoritmo para la asignación de la etiqueta de caracterización, a partir de un enfoque de detección

En el proceso de la etapa 3 es necesario asignar una etiqueta de clase al conjunto de métricas de descripción calculadas para el conjunto de datos que se está analizando. Para abordar esta tarea, se ha desarrollado un nuevo enfoque de detección de amenazas basado en la [Entropía de Sha](#sección1_6_3)nnon, aplicado específicamente a datos de tráfico de red. Este algoritmo se describe en el seudocódigo de la tabla 2.

Tabla 2: Seudocódigo del algoritmo para la asignación de la etiqueta de clase

|  |  |  |  |
| --- | --- | --- | --- |
| **Seudocódigo** | | | |
|  | **Entrada**: Llegan un conjunto de X instancias de datos de trafico de red. | | |
|  | **Salida:** Valor de verdadero o falso en dependencia de si existen amenazas o no. | | |
| **1** | Se le aplica el pre procesamiento al conjunto de datos entrantes. | | |
| **2** | Se dividen los datos de en paquetes de Y instancias. | | |
| **3** | **Para cada paquete:** | | |
| **4** |  | **Para cada columna:** | |
| **5** |  |  | Se calculan la cantidad de intervalos (bins) usando el método de **Doane**. |
| **6** |  |  | Se calcula la distribución de probabilidad del conjunto de datos. |
| **7** |  |  | Se calcula la entropía |
| **8** |  | **End** | |
| **9** |  | Se guardan los valores en una lista. | |
| **10** | **End** | | |
| **11** | Se conforma la matriz de entropías con los valores calculados. | | |
| **12** | Se calcula la media por cada columna de la matriz de entropía. | | |
| **13** | Se le asigna una etiqueta a partir de la comparación con un conjunto de reglas definidas en un árbol de reglas previamente entrenado | | |

A partir del pseudocódigo mostrado en la tabla 2, se puede observar que se inicia con un conjunto de instancias que representan el tráfico de red; sin embargo, antes de proceder con el análisis, es necesario realizar un pre procesamiento. Este paso implica limpiar los datos para eliminar cualquier ruido o información irrelevante que podría distorsionar los resultados. Al normalizar los datos, se asegura que el análisis posterior sea de mayor calidad.

Una vez que los datos están limpios, el siguiente paso es dividirlos en paquetes. Esta segmentación en grupos más pequeños permite un análisis más manejable y detallado. Al trabajar con paquetes de tamaño Y, se facilita la identificación de patrones dentro de cada segmento, lo que favorece la detección de anomalías.

Con los paquetes listos, se procede a un análisis completo. Para cada columna dentro de un paquete, se calcula la cantidad de intervalos utilizando el método de Doane. A continuación, se evalúa la distribución de probabilidad de cada conjunto de datos en los paquetes. Comprender cómo se distribuyen los valores es importante debido a que si hay una concentración inusual en ciertos rangos, puede ser un indicativo de actividad sospechosa.

A continuación, se calcula la entropía, para medir la incertidumbre en la información contenida en cada columna y los resultados obtenidos se almacenan en una lista, para luego organizarlos en una matriz de entropías. Esta matriz proporciona una vista generalizada de cómo varían las entropías a través de todos los paquetes analizados, permitiendo observar tendencias o cambios significativos en el tráfico.

Posteriormente se procede a calcular la media de las entropías para cada columna. Este valor promedio actúa como un punto de referencia que ayuda a entender el comportamiento general del tráfico. Una vez calculada la media para cada columna, se procede a asignar la etiqueta del conjunto de datos. Para ello se toman una serie de reglas obtenidas al efectuar un análisis en el capítulo 3 (Utilizando un árbol de reglas), de cómo se comporta el valor de entropía por cada columna para diversos conjuntos de datos.  
  
En base a las reglas obtenidas por el árbol y a partir de los valores de entropía de cada columna, el algoritmo concluirá que hay una posible amenaza presente en el tráfico analizado y devolverá un valor verdadero; si no, indicará que no hay amenazas detectadas o lo que es lo mismo devolverá un valor falso.

## 2.6- Ejecución de la tercera etapa del módulo

La tercera etapa del módulo se divide en varios procesos como se vio en la [sección 2.1](#seccion2_1), sin embargo, a partir de los cambios realizados a lo largo de este capítulo se eliminó lo referente al proceso de reclasificación como se puede ver en figura 12 . Este cambio se debe a que ya no hay una dependencia de datos etiquetados para la caracterización de los usuarios que antes se hacía con medidas de complejidad.

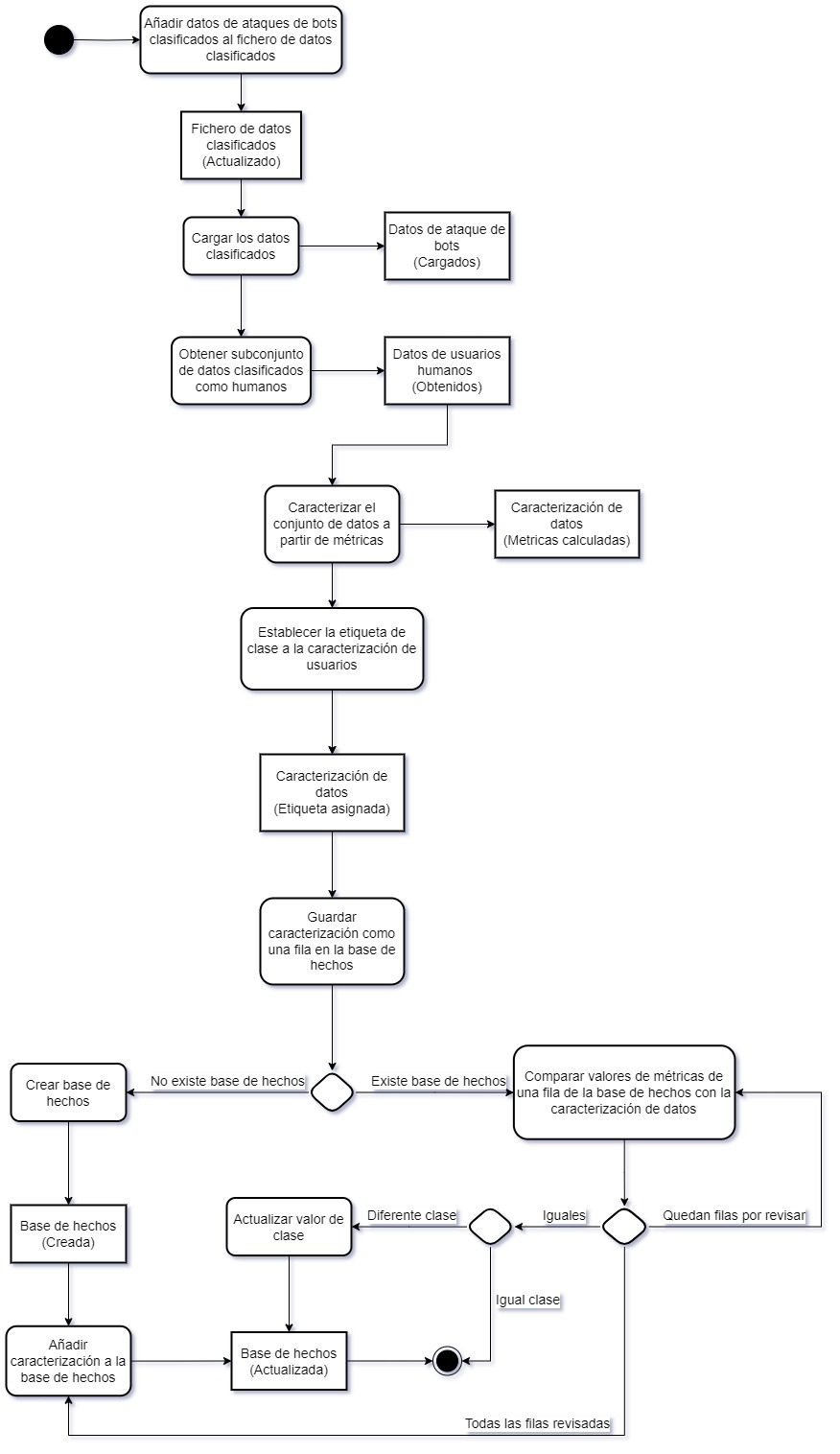


Figura 12: Diagrama de flujo del proceso de caracterización de usuarios por meta componente

Como se ilustra en la figura 13, con la implementación actual de la etapa, al finalizar la clasificación de los datos en la fase 2 mediante el método start\_component (x,y), se genera una instancia del metacomponente basada en el modelo desarrollado y los datos de bots que han sido clasificados.

A continuación, se utiliza add\_data (x,y) para guardar los datos clasificados en un archivo, y con load\_file\_instances() se recuperan todos los datos que se encuentran en dicho archivo.

El resultado de este proceso es la devolución del componente junto con los datos clasificados que han sido cargados. Al ejecutar el método run\_characterization (x,y) con los datos de ataque de bots ya clasificados, se inicia el proceso de caracterización del usuario por medio del metacomponente.

Luego se aplica el método set\_positives (x,y) con el objetivo de extraer los datos que han sido identificados como humanos y modificar una porción de los mismo a traves de la función simulate\_positives. A partir de estos datos, se calculan las métricas necesarias para caracterizar la información utilizando calculate\_metrics(). Luego, se ejecuta el método run\_metrics (xr, yr), empleando la clase correspondiente para ello, y se obtiene la caracterización del conjunto de datos junto con las métricas calculadas.

Posteriormente, se asigna una etiqueta de clase a la caracterización obtenida mediante set\_characterization\_label(). Para esto se utiliza el enfoque de clasificación basado en entropía implementado en [sección 2.5](#enfoque) para revisar la etiqueta de clase de los datos identificados como humanos.

Finalmente, con save\_data\_characterization(), se almacena la caracterización de los datos en una base de hechos, actualizando su valor si se trata de un conocimiento previamente adquirido. Así, se logra obtener una base de hechos actualizada.

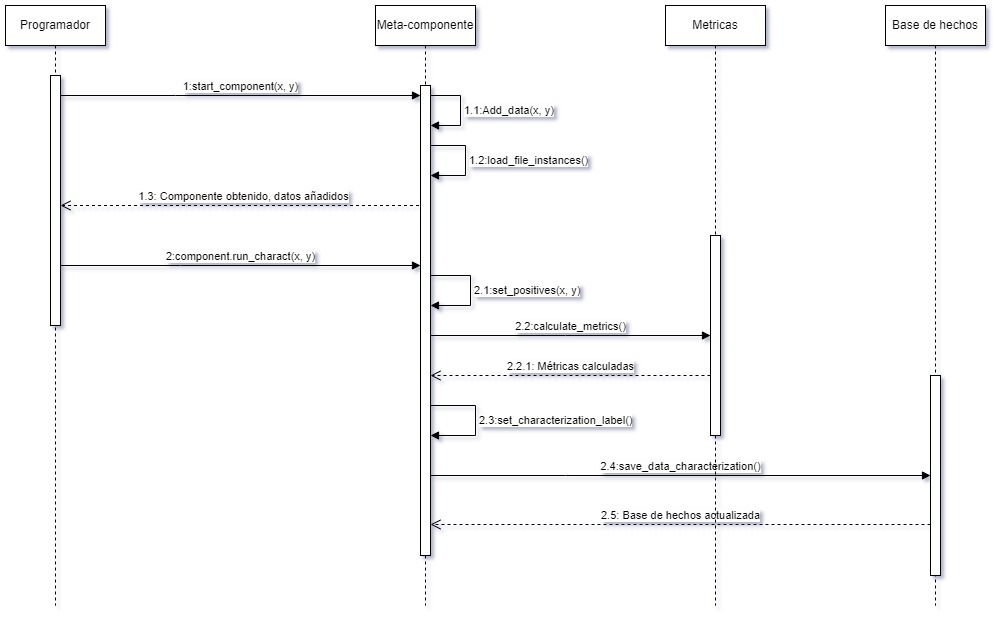


Figura 13 Diagrama de secuencia del proceso de la etapa 3

## 2.7-Conclusiones parciales

Al culminar el presente capítulo se arriban a las siguientes conclusiones parciales:

* Al cambiar el enfoque de métricas de complejidad a métricas de caracterización, se amplía la capacidad del sistema para trabajar con datos no etiquetados.
* La creación de una función para simular cambios en los datos sin elementos aleatorios ayuda a mantener la consistencia en el análisis de reclasificación.
* El nuevo enfoque de clasificación requiere el establecimiento de un conjunto de reglas para la clasificación y la definición de una configuración, que consiste en un número de entrada y un número de paquetes a analizar, determinados a partir de un proceso de experimentación.

# Capítulo 3: Evaluación del nuevo enfoque de detección

En este capítulo, como parte del proceso experimental, se lleva a cabo un estudio para determinar el conjunto de reglas necesario para garantizar el correcto funcionamiento del algoritmo para la asignación de la etiqueta de clase desarrollado en el capítulo anterior. Además, para validar los cambios realizados en la etapa 3, se realiza un análisis de casos. Este análisis incluye la identificación de diferentes casos que permiten evaluar el funcionamiento del meta componente, así como el análisis de los resultados obtenidos a partir de las fases previamente definidas.

## 3.1-Base de datos CTU-13

Para la validación y correcto funcionamiento de los nuevos cambios a realizados en el módulo, se ha elegido la base de datos CTU-13, un conjunto de datos públicos que se ha convertido en un recurso fundamental para la investigación en ciberseguridad y análisis de tráfico de red. Esta base de datos, desarrollada en un entorno de laboratorio controlado, ofrece una valiosa colección de trazas de red que reflejan una variedad de escenarios de ataque. En total, CTU-13 abarca 13 escenarios distintos, cada uno diseñado para simular diferentes tipos de ataques cibernéticos, tales como la propagación de malware, ataques de fuerza bruta, ataques de denegación de servicio (DoS) y otros métodos comunes utilizados por los atacantes en el mundo real [1].

Las trazas de red contenidas en esta base de datos son ricas en información y proporcionan un panorama detallado del tráfico generado durante los ataques. Estas trazas incluyen datos sobre paquetes TCP, UDP e ICMP, así como información relevante sobre los sistemas involucrados en cada ataque, incluyendo sus direcciones IP y otros metadatos que pueden ser cruciales para el análisis posterior [1].

El Anexo 1 evidencia los porcentajes de los flujos de datos para cada uno de los 13 escenarios de CTU-13. Se observan 3 flujos de datos diferentes, los flujos de fondo, los flujos de *bots* y los flujos normales. El flujo normal está compuesto por el tráfico que generan los usuarios humanos al navegar por la web, el flujo de *bots* está determinado por el tráfico de los bots que navegan visibles y finalmente el flujo de fondo se genera para ocultar la presencia de los bots.

Cada escenario de CTU-13, comprende un total de 15 columnas: *StartTime*, *Dur*, *Proto*, *SrcAddr*, *Sport*, *Dir*, *DstAddr*, *Dport*, *State*, *sTos*, *dTos*, *TotPkts*, TotBytes, *SrcBytes* y *Label*. Donde las 14 primeras comprenden las variables independientes de la base de datos y la última la variable independiente. El Anexo 2 muestra la descripción asociada a cada columna.

### 3.1.1- Pre procesamiento

Para el pre procesamiento de los datos se tomó el desarrollado en [9]. En el cual destaca la conversión de todos los datos a tipo numérico. Incluyendo las etiquetas de clase, donde las instancias de clase *background* y normal, se trataron como usuarios humanos, asignando valor 0. Mientras que las de clase *botnet*, fueron tratadas como usuarios *bots* malignos, asignando valor 1. Además, se eliminaron los datos vacíos y se normalizó la muestra de datos, estableciendo los valores en una escala de 0 a 1.

### 3.1.2- Base de datos central de CTU-13

Es importante señalar que la diversidad y la complejidad de los escenarios presentados en la base de datos CTU-13 pueden llevar a desafíos significativos. Cada escenario contiene diferentes tipos y volúmenes de información, y algunos no cuentan con una cantidad representativa de *bots*. Esto puede dificultar el análisis comparativo y la generalización de los resultados obtenidos a partir de los experimentos realizados. Consciente de estas limitaciones, en [42] el autor desarrolló un método para crear una base de datos centralizada que abarca y unifica los datos provenientes de los escenarios deseados de CTU-13.

Este enfoque centralizado permite consolidar la información dispersa en los diferentes escenarios, facilitando así un análisis más exhaustivo y coherente. La creación de bases de datos integradas no solo optimiza el acceso a los datos, sino que también mejora la calidad del análisis al proporcionar un conjunto más completo y representativo sobre el cual trabajar [42].

En el contexto específico de este trabajo, se trabaja con métricas, como es el caso de la entropía, por lo que se ha decidido no implementar un balance en los datos en el pre procesamiento. El balanceo es una técnica comúnmente utilizada en el pre procesamiento de datos para mitigar el sobreajuste en algoritmos de aprendizaje automático. Sin embargo, dado que el objetivo principal en este trabajo es calcular métricas específicas y no entrenar modelos predictivos, se optó por omitir la creación de datos sintéticos que normalmente se emplearía para equilibrar las clases en la base de datos central.

Esto permite centrarse en las características intrínsecas del tráfico de red sin distorsionar los datos originales con información artificial. En lugar de ello, se llevó a cabo una reducción dimensional (PCA) que transforma el conjunto de datos original a solo siete columnas. Esta reducción es esencial para facilitar el análisis posterior al eliminar redundancias y concentrarse en las variables más significativas que impactan directamente en las métricas a evaluar.

## 3.2-Generación del árbol de reglas para la selección de etiqueta de caracterización

Se empleó un árbol de reglas para determinar la etiqueta de caracterización correspondiente al lote analizado. Este árbol de reglas se generó a partir de un árbol de decisión, que fue entrenado con datos de la base de datos CTU-13 mediante un proceso de muestreo. Se decidió excluir los escenarios 5, 7 y 11 debido a la insuficiencia de datos para la realización del experimento. Cada muestra consistía en un conjunto de datos que contenía 100,000 instancias, 200,000 instancias o 300,000 instancias.

En total, se generaron 200 muestras por cada escenario analizado, lo que resultó en un conjunto acumulado de 2,000 muestras. Estas se distribuyeron equitativamente entre dos categorías: una mitad contenía exclusivamente instancias de usuarios humanos, mientras que la otra mitad incluía una mezcla de instancias de usuarios humanos y *bots*.

Se ejecutó el algoritmo del enfoque con cada muestra y se guardaron los valores de media por columna como una nueva fila en un fichero .csv. Se crearon de esta manera varios ficheros .csv, uno por cada combinación de tamaño de entrada analizado (100,000; 200,000; 300,000) y tamaño de paquete analizado (5,000; 10,000; 20,000; 25,000), es decir, uno por cada configuración del algoritmo. En dependencia si la muestra contenía instancias de *bots* se añadió como una nueva columna la etiqueta de clase “bh” en caso verdadero y “h” en caso contrario. Posteriormente, en la plataforma analítica *Knime* se utilizó un árbol de decisión para generar el conjunto de reglas por cada fichero .csv. Este árbol de reglas permite clasificar los datos de cada configuración del algoritmo analizada, como amenaza o no amenaza. En la figura 14 se puede ver el flujo de *Knime* utilizado para generar las reglas del árbol de decisión.

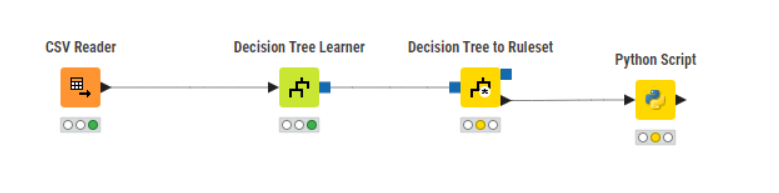


Figura 14 Flujo de Knime utilizado para generar el conjunto de reglas

El árbol creado está conformado por una gran cantidad de reglas y presenta peor rendimiento si el tamaño del conjunto de datos difiere mucho del tamaño de las muestras escogidas para su creación. Esto trae como limitación que es necesario que el conjunto de datos que se le pasa al algoritmo para asignar la etiqueta sea cercano a uno de los tamaños predefinidos (100,000; 200,000 o 300,000 datos).

### 3.2.1-Elección de la configuración

Para seleccionar el número de instancias que debería tener cada paquete al ejecutar el algoritmo se llevó a cabo una comparación del rendimiento en base a la eficacia de las reglas generadas para entradas de 100,000, 200,000 y 300,000 datos y con paquetes de 5,000 instancias, 10,000 instancias, 20,000 instancias y 25,000 instancias por cada tamaño de entrada. Para probar las reglas y por consecuencia, cada configuración del algoritmo, se diseñó un pequeño experimento. Este consiste en ejecutar el algoritmo 100 veces con cada configuración, utilizando diferentes muestras extraídas de CTU-13, para luego guardar el porcentaje de etiquetas correctamente asignadas (rendimiento en base a eficacia) por el algoritmo. Este experimento se repitió 10 veces para generar resultado que serán analizados en secciones posteriores.

#### 3.2.1.1-Entrada de 100,000 datos

Como se puede ver en la tabla 3, con una entrada de 100,000 datos los mejores resultados se obtuvieron con paquetes de 25,000 instancias con una eficacia promedio de 86.1%. Los paquetes de 10,000 instancias registran el mayor valor de precisión con un 92%.

Tabla 3: Resultados del enfoque con un tamaño de entrada de 100,000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tamaño del paquete | | | |
| Experimento | **5 000** | **10 000** | **20 000** | **25 000** |
| 0 | 87 | 86 | 82 | 87 |
| 1 | 87 | 87 | 80 | 85 |
| 2 | 86 | 86 | 81 | 83 |
| 3 | 86 | 85 | 85 | 86 |
| 4 | 78 | 85 | 77 | 89 |
| 5 | 89 | 84 | **88** | **91** |
| 6 | 84 | **92** | 86 | 87 |
| 7 | 77 | 82 | 83 | 81 |
| 8 | **90** | 79 | 80 | 90 |
| 9 | 84 | 87 | 85 | 82 |
| Promedio | 84.8 | 85.3 | 82.7 | 86.1 |

Con los valores representados en la tabla 3 se llevó a cabo una prueba de hipótesis de Kruskall-Wallis para ver si existen diferencias significativas entre el rendimiento del enfoque con cada tamaño de paquete. La prueba devolvió un Valor P de 0.198 que como es mayor que el nivel de significancia = 0.05 se puede afirmar que no hay diferencias significativas en los resultados para las hipótesis siguientes:

H0: µ5 000 = µ10 000 = µ20 000 = µ25 000

H1: µ5 000 ≠ µ10 000 ≠ µ20 000 ≠ µ25 000

Con:

µ5 000: mediana de valores del enfoque con paquetes de 5,000 instancias

µ10 000: mediana de valores del enfoque con paquetes de 10,000 instancias

µ20 000: mediana de valores del enfoque con paquetes de 20,000 instancias

µ25 000: mediana de valores del enfoque con paquetes de 25,000 instancias

α = 0.05

|  |  |
| --- | --- |
| Valor H | Valor P |
| 4.67 | 0.198 |

#### 

#### 3.2.1.2-Entrada de 200,000 datos

Como se puede ver en la tabla 4, con una entrada de 200,000 datos los mejores resultados se obtuvieron con paquetes de 5,000 instancias con una precisión promedio de 86.1%. Los paquetes de 5,000 instancias registran el mayor valor de precisión con un 91%.

Tabla 4: Resultados del enfoque con un tamaño de entrada de 200,000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tamaño del paquete | | | |
| Experimento | **5 000** | **10 000** | **20 000** | **25 000** |
| 0 | **91** | 79 | 79 | 79 |
| 1 | 87 | 83 | 82 | 82 |
| 2 | 81 | 85 | 82 | 80 |
| 3 | 89 | 85 | 85 | 81 |
| 4 | 84 | 83 | 82 | 81 |
| 5 | 87 | 81 | **86** | 80 |
| 6 | 88 | 86 | 82 | **86** |
| 7 | 88 | 82 | 83 | 78 |
| 8 | 83 | 82 | 80 | 80 |
| 9 | 83 | **87** | 85 | 79 |
| Promedio | 86.1 | 83.3 | 82.6 | 80.6 |

Con los valores representados en la tabla 4 se llevó a cabo una prueba de hipótesis de Kruskall-Wallis para ver si existen diferencias significativas entre el rendimiento del enfoque con cada tamaño de paquete. En esta ocasión la prueba devolvió un Valor P de 0.001 que es menor que el nivel de significancia = 0.05. Entonces se puede afirmar que existen diferencias significativas en los resultados con un tamaño de entrada de 200,000 para las hipótesis siguientes:

H0: µ5 000 = µ10 000 = µ20 000 = µ25 000

H1: µ5 000 ≠ µ10 000 ≠ µ20 000 ≠ µ25 000

Con:

µ5 000: mediana de valores del enfoque con paquetes de 5,000 instancias

µ10 000: mediana de valores del enfoque con paquetes de 10,000 instancias

µ20 000: mediana de valores del enfoque con paquetes de 20,000 instancias

µ25 000: mediana de valores del enfoque con paquetes de 25,000 instancias

α = 0.05

|  |  |
| --- | --- |
| Valor H | Valor P |
| 15.66 | 0.001 |

Se llevó a cabo un análisis post-hoc para determinar entre que grupos existen diferencias específicamente, determinándose diferencias significativas entre los grupos de la tabla 5.

Tabla 5 Diferencias para entrada de 200,000 usuarios

|  |  |
| --- | --- |
| Grupos | Valor-P |
| 5,000 vs 25,000 | 0.0001 |
| 5,000 vs 20,000 | 0.0250 |
| 10,000 vs 25,000 | 0.0250 |

Como se puede ver en la tabla 5 los grupos que presentan diferencias entre sí son los tamaños de paquete de 5,000 vs 20,000, 5,000 vs 25,000 y 10,000 vs 25,000. Los grupos de 5,000 vs 10,000, 10,000 vs 20,000 y 10,000 vs 25,000 presentan rendimiento similar. En la figura 15 se muestra el grafico de caja para cada configuración del algoritmo del que se puede concluir que para una entrada de 200 000 datos las reglas generadas con un tamaño de paquete de 5,000 datos obtienen los mejores resultados.

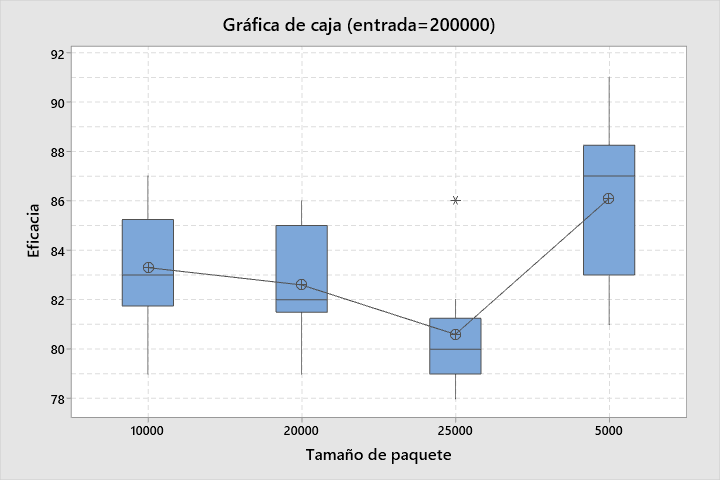


Figura 15: Gráfica de Caja para los tamaños de paquete de 5000, 10000, 20000 y 25000

#### 3.2.1.3-Entrada de 300,000 datos

Como se puede ver en la tabla 6, con una entrada de 300,000 datos los mejores resultados se obtuvieron con paquetes de 25,000 instancias con una precisión promedio de 85.2%. Los paquetes de 25,000 instancias registran el mayor valor de precisión con un 90%.

Tabla 6: Resultados del enfoque con un tamaño de entrada de 300,000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tamaño del paquete | | | |
| **Experimento** | **5 000** | **10 000** | **20 000** | **25 000** |
| **0** | 80 | **84** | **83** | 87 |
| **1** | 76 | 79 | 74 | 85 |
| **2** | 83 | 77 | 81 | 81 |
| **3** | **88** | 74 | 82 | 84 |
| **4** | 81 | 78 | 82 | 83 |
| **5** | 82 | 72 | 81 | 87 |
| **6** | 85 | 82 | 81 | 86 |
| **7** | 87 | 80 | 81 | 80 |
| **8** | 87 | 78 | 81 | 89 |
| **9** | 83 | 82 | 81 | **90** |
| Promedio | 83.2 | 78.6 | 80.7 | 85.2 |

Con los valores representados en la tabla 6 se llevó a cabo una prueba de hipótesis de Kruskall-Wallis para ver si existen diferencias significativas entre el rendimiento del enfoque con cada tamaño de paquete. Nuevamente la prueba devolvió un Valor P = 0.002 que es menor que el nivel de significancia = 0.05. Entonces se puede afirmar que existen diferencias significativas en los resultados con un tamaño de entrada de 300,000 para las hipótesis siguientes:

H0: µ5 000 = µ10 000 = µ20 000 = µ25 000

H1: µ5 000 ≠ µ10 000 ≠ µ20 000 ≠ µ25 000

Con:

µ5 000: mediana de valores del enfoque con paquetes de 5,000 instancias

µ10 000: mediana de valores del enfoque con paquetes de 10,000 instancias

µ20 000: mediana de valores del enfoque con paquetes de 20,000 instancias

µ25 000: mediana de valores del enfoque con paquetes de 25,000 instancias

α = 0.05

|  |  |
| --- | --- |
| Valor H | Valor P |
| 14.54 | 0.002 |

Se llevó a cabo un análisis post-hoc para determinar entre que grupos existen diferencias específicamente, determinándose diferencias significativas entre los grupos de la tabla 7.

Tabla 7 Diferencias para entrada de 300,000 usuarios

|  |  |
| --- | --- |
| Grupos | Valor-P |
| 10,000 vs 25,000 | 0.0004 |
| 5,000 vs 10,000 | 0.0111 |
| 20,000 vs 25,000 | 0.0123 |

Como se puede ver en la tabla 7 los grupos que presentan diferencias entre sí son los tamaños de paquete de 5,000 vs 10,000, 20,000 vs 25,000 y 10,000 vs 25,000. Los grupos de 5,000 vs 20,000, 5,000 vs 25,000 y 10,000 vs 20,000 presentan rendimiento similar. En la figura 16 se muestra el grafico de caja para cada configuración del algoritmo del que se puede concluir que para una entrada de 300 000 datos, las reglas generadas con un tamaño de paquete de 25,000 datos obtienen los mejores resultados por un ligero margen.

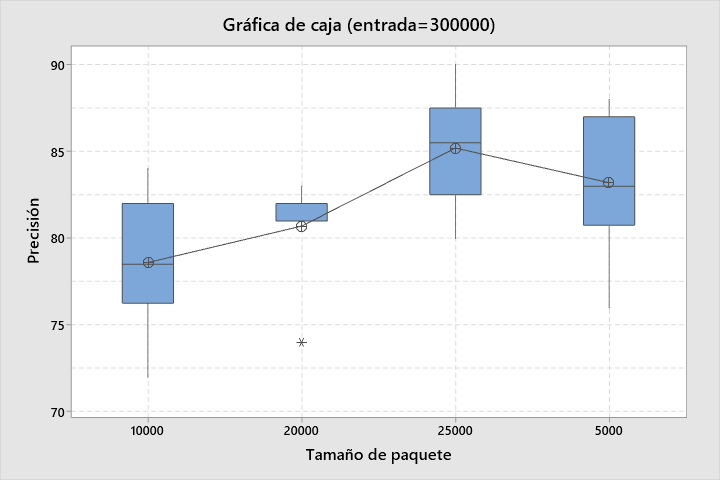


Figura 16: Gráfica de Caja para los tamaños de paquete de 5,000, 10,000, 20,000 y 25,000

## 3.3-Estudio de casos

En este segmento, se presenta un análisis de casos de estudio como parte del proceso de validación llevado a cabo. Este análisis tiene como objetivo evaluar el procedimiento de caracterización de usuarios mediante meta componente, el cual corresponde a la tercera etapa del módulo diseñado para la detección de *bots* maliciosos. Para lograr esto, se identifican diversos casos que facilitan la comprensión del funcionamiento del meta componente, así como la evaluación de los resultados obtenidos en función de las fases establecidas.

### 3.3.1-Descripción de los casos identificados

Los casos están estructurados para evaluar el rendimiento del algoritmo *Proactive Forest* en diferentes escenarios, lo que permitirá obtener una comprensión más profunda en cada uno de los tres escenarios seleccionados. Los 2 primeros casos consisten en variaciones de los definidos en [9]:

* Escenario pequeño: El primer caso de estudio involucra la ejecución del meta componente utilizando un modelo construido sobre un escenario pequeño. Este proceso se inicia con la creación de un bosque de decisiones que ha sido entrenado específicamente con datos de un entorno reducido utilizando el algoritmo *Proactive Forest*. La selección del escenario 12, que se caracteriza por tener la menor cantidad de instancias de los analizados en la [sección 3.2](#umbral), sigue la lógica de evaluar cómo responde el modelo cuando se encuentra con un conjunto de datos más limitado.
* Escenario Grande: En el segundo caso, se repite la estructura del análisis anterior, pero esta vez se trabaja con un escenario grande. Al igual que en el caso anterior, se crea un bosque de decisiones mediante el algoritmo *Proactive Forest*; sin embargo, el entrenamiento se realiza con datos del escenario 3, que presenta la mayor cantidad de instancias de los escenarios analizados en la [sección 3.2](#umbral). Este caso permite observar cómo el modelo se comporta cuando tiene a su disposición un volumen considerable de información.
* Escenario centralizado: El tercer y último caso de análisis emplea un modelo basado en una [base de datos centralizada](#base_centralizada). En este caso, se vuelve a implementar un bosque de decisión, entrenado con datos obtenidos de la base de datos centralizada, utilizando el mismo algoritmo *Proactive Forest*. Este enfoque busca discernir si un conjunto de datos más íntegro y consolidado puede influir positivamente en la precisión del modelo. Al combinar información de diversas fuentes, se espera que el algoritmo tenga acceso a un espectro más amplio de características que podrían ser útiles para la correcta identificación de instancias de *bot*.

Cada uno de los casos identificados será ejecutado un total de dos veces, con variaciones en la cantidad de datos analizados en cada ejecución. En la primera ejecución, se analizarán 100,000 datos humanos y 5,000 datos de *bots* (lo que representa el 5% de la cantidad de datos humanos). En la segunda ejecución, la cantidad de datos aumentará a 200,000 datos humanos y 10,000 datos de *bots*, manteniendo la misma proporción. Al finalizar cada fase del proceso de caracterización de datos de usuario, se presentarán y analizarán los resultados obtenidos.

#### 3.3.2.1- Ejecución del meta componente empleando un modelo construido sobre el escenario pequeño

En el primer caso identificado, se evalúa el rendimiento del meta componente utilizando datos clasificados que se obtienen de un bosque de decisión, el cual ha sido entrenado con el algoritmo *Proactive Forest* sobre un escenario pequeño. Este análisis tiene como objetivo verificar si los usuarios inicialmente clasificados como humanos son, de hecho, falsos positivos. Para ello, se llevan a cabo modificaciones no aleatorias en sus características. En la tabla 8 se presentan los resultados obtenidos en una ejecución que analizó un total de 105,000 datos.

Tabla 8 Caso 1 Primera ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 9,368 | 95,632 | 38,252 | CUSUM: 25,850.719 | *Bots* | Fila añadida |
|  |  |  | nCUSUM: -6,262.960 |  |  |
| SE: 2.707 |
| JS: 0.568 |
| MD: 134.315 |
| IQR: 0.058 |
| MAD: 0.068 |

En el proceso de clasificación inicial de la tabla 8, se intentó distinguir entre 5,000 *bots* y 100,000 humanos. Sin embargo, la eficacia de la clasificación no fue óptima, se identificaron un total de 9,368 *bots* y 95,632 humanos. Para un análisis más detallado, se seleccionaron 38,252 instancias de las clasificadas como humanos, a las que se realizó una modificación en busca de falsos positivos.

Al evaluar estas instancias, se obtuvieron varias métricas. El CUSUM alcanzó un valor de 25,850.719, lo que indica un cambio notable en el comportamiento observado en comparación con el de los humanos, lo cual podría indicar variabilidad significativa o incluso anomalías en el patrón de tráfico de red. Por otro lado, el CUSUM negativo mostró un valor de -6,262.960. Esto podría interpretarse como un indicativo de comportamientos que se sitúan por debajo de la media esperada, sugiriendo la presencia de anomalías o patrones inusuales en las instancias analizadas.

La entropía de Shannon fue de 2.707, lo que sugiere que los datos tienen un cierto grado de orden o predictibilidad. Un valor así implica que hay cierta variabilidad entre los datos, lo que sugiere que no son homogéneos. En cuanto a la distancia Jensen-Shannon, se obtuvo un valor de 0.568, lo que indica una similitud moderada entre las distribuciones de comportamiento de las instancias. La distancia de Mahalanobis, que alcanzó un valor de 134.315, revela que varias instancias están bastante alejadas del comportamiento típico esperado.

Por último, tanto el rango intercuartílico (IQR) como la desviación absoluta media (MAD) mostraron valores bajos: 0.058 y 0.068 respectivamente. Esto sugiere que la mayoría de las instancias están agrupadas cerca de la media, indicando una baja variabilidad en los datos.

Posteriormente a través del nuevo enfoque implementado en la [sección 2.3](#enfoque) se le asignó un valor de clase “*bots*” para la caracterización realizada por las métricas. Después, esta caracterización se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

La tabla 9 muestra los resultados de la segunda ejecución del escenario pequeño, esta vez analizando 210,000 datos.

Tabla 9 Caso 1 Segunda ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 48,674 | 161,326 | 64,530 | CUSUM: 24,875.279 | No *bots* | Fila añadida |
|  |  |  | nCUSUM: -20,842.038 |  |  |
| SE: 2.906 |
| JS: 0.564 |
| MD: 170.013 |
| IQR: 0.0601 |
| MAD: 0.367 |

En esta ocasión se duplicaron las instancias a analizar para un total de 10,000 *bots* y 200,000 humanos reales. A pesar del aumento en la cantidad de instancias, la clasificación también se vio sujeta a errores, identificando un total de 48,674 *bots* y 161,326 humanos. Para un análisis más detallado, se seleccionaron 38,252 instancias de las clasificadas como humanos, a las que se realizó una modificación en busca de falsos positivos.

En la segunda ejecución del caso con el doble de datos el valor de CUSUM disminuye ligeramente de 25,850.719 a 24,875.279 lo cual todavía es considerablemente alto. Esto sugiere que todavía hay una gran diferencia en los datos con respecto a la media de referencia.

El CUSUM negativo, por su parte, mostró un valor negativo de -20,842.038. Esto sugiere que muchas de las instancias analizadas están por debajo del comportamiento esperado.

La entropía aumenta de 2.707 a 2.906. Este aumento indica un incremento en la incertidumbre o desorden dentro de los datos, sugiriendo que el segundo conjunto de datos puede tener más incertidumbre o menos patrones predecibles en comparación con el primer conjunto. La distancia de Jensen-Shannon es casi idéntica. Esto puede sugerir una similitud sustancial de la distribución de probabilidad del conjunto de datos de la primera ejecución y la distribución de probabilidad del conjunto de datos de la segunda ejecución. Esto puede indicar que comparten características similares en cuanto a la presencia de elementos comunes y diferencias notables.

La Distancia de Mahalanobis aumenta de 134.315 a 170.013. Este aumento sugiere una mayor presencia de *outliers* en comparación con el primer conjunto. El IQR aumenta ligeramente de 0.058 a 0.0601. Esta leve diferencia sugiere una ligeramente mayor dispersión en el segundo conjunto de datos, pero ambos conjuntos mantienen una estructura de datos relativamente concentrada entre el primer y el tercer cuartil. La MAD, sin embargo, aumenta significativamente de 0.068 a 0.367. La MAD, a diferencia del IQR, no solo mide la dispersión en el 50% central de los datos (tercer cuartil – primer cuartil) por lo que este cambio tan marcado puede sugerir la presencia de valores atípicos fuera del 50% central de los datos lo que explica también el aumento de la Distancia de Mahalanobis.

Posteriormente a través del nuevo enfoque implementado se le asignó un valor de clase “no *bots*” para la caracterización realizada por las métricas. Este hallazgo se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

#### 3.3.2.2-Análisis de los resultados

Mientras que ambos conjuntos de datos muestran patrones consistentes y cierta similitud, el segundo conjunto parece presentar una mayor variabilidad y dispersión, lo que podría indicar diferencias en la estabilidad o consistencia del tráfico de red. Algo que llama la atención, es que en la segunda ejecución del caso a pesar de que un mayor grupo de humanos fue modificado el enfoque no determinó la presencia de *bots* en el conjunto resultante en contra posición a la ejecución anterior.

#### 3.3.3.1- Ejecución del meta componente empleando un modelo construido sobre el escenario grande

En el segundo escenario analizado, se lleva a cabo una evaluación del meta componente utilizando un modelo desarrollado a partir de grandes conjuntos de datos. Este modelo se basa en un bosque de decisiones que ha sido entrenado mediante el algoritmo *Proactive Forest*. Los resultados de la primera ejecución, realizada con 5,000 instancias de *bots* y 100,000 de humanos, se detallan en la tabla 10.

Tabla 10 Caso 2 Primera ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 5,064 | 99,936 | 39,974 | CUSUM: 33.891 | *bots* | Fila añadida |
| nCUSUM: -8,806.054 |
| SE: 2.795 |
| JS: 0.557 |
| MD: 119.275 |
| IQR: 0.325 |
| MAD: 0.257 |

Como se evidencia en la tabla 10 se obtuvo una clasificación inicial que identificó 5,064 *bots* y 99,936 humanos. Sin embargo, ante la posibilidad de falsos positivos, se procedió a modificar 39,974 instancias clasificadas como humanos para someterlas a otro análisis. Posterior al cambio se aplicó una serie de métricas para caracterizar estas instancias previamente clasificadas como humanos.

La prueba CUSUM arrojó un valor de 33.891, indicando la presencia de cambios sutiles con respecto a la muestra de valores humanos con la que se está comparando. Además, el análisis del CUSUM negativo mostró un decrecimiento de valor -8,806.054, lo que sugiere una desviación considerable hacia abajo en relación a la media esperada. Este resultado puede indicar la existencia de patrones de comportamiento anómalo o inusual en las instancias analizadas. Por otra parte la entropía de Shannon, con un valor de 2.795, refleja un nivel moderado de incertidumbre en las instancias modificadas.

La distancia Jensen-Shannon con un valor de 0.557, sugiere una similitud moderada entre las distribuciones de comportamiento, sin embargo, la distancia de Mahalanobis con 119.275 indica que hay instancias significativamente alejadas del centroide, lo cual podría ser indicativo de comportamientos anómalos. Por último el IQR y MAD con valores de 0.325 y 0.257 respectivamente, reflejan una cierta variabilidad en los comportamientos capturados.

Posteriormente a través del nuevo enfoque implementado se le asignó un valor de clase “*bots*” para la caracterización realizada por las métricas. Este hallazgo se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

La tabla 11 muestra los resultados de la segunda ejecución del escenario grande esta vez analizando 10,000 *bots* y 200,000.

Tabla 11 Caso 2 Segunda ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 10,611 | 199,389 | 79,755 | CUSUM: 411.391 | *bots* | Fila añadida |
|  |  |  | nCUSUM: -14,171.884 |  |  |
| SE: 2.834 |
| JS: 0.592 |
| MD: 8.006 |
| IQR: 0.339 |
| MAD: 0.267 |

Como se observa en la tabla 11, la clasificación inicial identificó 10,611 *bots* y 199,389 humanos de un conjunto de 10,000 *bots* y 200,000 humanos. Ante la posibilidad de falsos positivos, se procedió a modificar 79,755 instancias clasificadas como humanos para someterlas a un análisis adicional. Esto permitió aplicar una serie de métricas para caracterizar estas instancias modificadas.

La prueba CUSUM arrojó un valor de 411.391, indicando cambios significativos en comparación con la muestra de valores humanos. Este resultado sugiere que hay variaciones notables en el comportamiento de las instancias analizadas en relación con las expectativas iniciales.

Además, el análisis del CUSUM negativo mostró un valor de -14,171.884, lo que sugiere la presencia de patrones de comportamiento que se desvían notablemente hacia abajo en relación a la media esperada.

La entropía de Shannon, con un valor de 2.834, refleja un nivel moderado de incertidumbre en las instancias modificadas, sugiriendo una diversidad razonable en los patrones de comportamiento. Esto puede implicar que las instancias no son homogéneas y presentan variaciones en sus características.

La distancia Jensen-Shannon, con un valor de 0.592, indica una similitud moderada entre las distribuciones de comportamiento, lo que sugiere que hay ciertos patrones comunes entre las instancias analizadas, aunque también pueden existir diferencias notables.

Por otro lado, la distancia Mahalanobis, con un valor de 8.006, señala que hay pocas instancias alejadas del centroide de referencia, lo cual podría ser indicativo de que casi no hay comportamientos anómalos. Finalmente, el IQR y MAD, con valores de 0.339 y 0.267 respectivamente, muestran una baja dispersión en los comportamientos capturados.

Con base en el nuevo enfoque implementado, se asignó un valor de clase “bots” para la caracterización realizada por las métricas. Este hallazgo se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

#### 3.3.3.2-Análisis de los resultados

A partir de las métricas de la primera ejecución parece que los datos presentan cierta variabilidad con respecto a la media de referencia (indicado por el valor de CUSUM) y la presencia de valores anómalos (valor de la Distancia de Mahalanobis). Sin embargo, la diversidad en los patrones de tráfico es moderada (Entropía de Shannon y Distancia de Jensen-Shannon), y el conjunto de datos presenta una dispersión estadística relativamente baja (bajos valores de IQR y MAD). Esto podría sugerir que, aunque hay variabilidad y posibles anomalías, los datos muestran una consistencia general en su comportamiento.

Con los valores de la segunda ejecución, se puede inferir que el conjunto de datos ha experimentado cambios significativos en términos de variabilidad y diversidad, probablemente debido a que se duplica la cantidad de datos. Hay un aumento en la variabilidad (reflejado por los incrementos en IQR y MAD) y una ligera disminución en la similitud entre los patrones de tráfico (indicado por el aumento en la distancia de Jensen-Shannon). Además, el valor de CUSUM es más alto, lo que sugiere que los cambios en la media del tráfico de red persisten y además son más pronunciados, pero quizás con una tendencia diferente. Por último, la reducción en la distancia de Mahalanobis indica que los datos se han vuelto más homogéneos, con menos presencia de *outliers* extremos. En esta ocasión los cambios realizados en el conjunto de humanos fueron identificados por el enfoque como un cambio en la naturaleza de los sujetos.

#### 3.3.4.1- Ejecución del meta componente empleando un modelo construido sobre el escenario central

En este estudio se utilizó un escenario centralizado que encapsula la diversidad y amplitud de los tipos de datos presentes en todos los escenarios analizados. Este enfoque integral permite abordar la detección y caracterización de *bots* desde una perspectiva integral, aprovechando la riqueza de los datos recopilados para desarrollar un modelo más generalizado. Este modelo se basa en un bosque de decisiones que ha sido entrenado mediante el algoritmo *Proactive Forest*. Los resultados de la primera ejecución, realizada con 5,000 instancias de *bots* y 100,000 de humanos, se detallan en la tabla 12.

Tabla 12 Caso 3 Primera ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 4,999 | 100,001 | 40,000 | CUSUM: 1,768.969 | *bots* | Fila añadida |
|  |  |  | nCUSUM: -434.568 |  |  |
| SE: 3.263 |
| JS: 0.481 |
| MD: 27.258 |
| IQR: 0.575 |
| MAD: 0.280 |

Como se observa en la tabla 12, la clasificación inicial identificó 4,999 *bots* y 100,001 humanos. Ante la posibilidad de falsos positivos, se procedió a modificar 40,000 instancias clasificadas como humanos para someterlas a un análisis adicional. Esto permitió aplicar una serie de métricas para caracterizar estas instancias modificadas.

La prueba CUSUM arrojó un valor de 1,768.969, indicando cambios en comparación con la muestra de valores humanos. Además, el análisis del CUSUM negativo mostró un valor de -434.568, lo que sugiere la presencia de patrones de comportamiento que se desvían hacia abajo en relación a la media esperada.

La entropía de Shannon, con un valor de 3.263, el cual es mayor que en todos los demás casos analizados, refleja un nivel elevado de incertidumbre en las instancias modificadas, lo que sugiere una gran diversidad en los patrones de comportamiento. Un valor bajo como 0.481 para la Distancia Jensen-Shannon podría indicar que el conjunto de datos está relativamente cerca al conjunto de datos normales en términos de su distribución, lo que sugiere similitudes en el comportamiento. El valor de la Distancia Mahalanobis es similar al valor presentado en la primera ejecución del primer caso y podría indicar presencia de *outliers* o grupos distintos dentro de los datos de tráfico de red. Finalmente, el IQR y MAD, con valores de 0.575 y 0.280 respectivamente, muestran una variabilidad considerable en los comportamientos capturados.

Posteriormente, mediante el nuevo enfoque implementado, se asignó un valor de clase “*bots*” para la caracterización realizada por las métricas. Este hallazgo se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

La tabla 13 muestra los resultados de la segunda ejecución del escenario centralizado, esta vez analizando 10,000 *bots* y 200,000 humanos.

Tabla 13 Caso 3 Segunda ejecución

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Clasificación** | | | **Caracterización** | | **Base de hechos** |
| **Cantidad de bots** | **Cantidad de humanos** | **Cantidad de instancias modificadas** | **Resultado de las métricas** | **Valor de clase asignado** | **Resultado** |
| 11,017 | 198,983 | 79,593 | CUSUM: 3,596.691 | No *bots* | Fila añadida |
|  |  |  | nCUSUM: -1653.702 |  |  |
| SE: 3.454 |
| JS: 0.502 |
| MD: 6.930 |
| IQR: 0.562 |
| MAD: 0.273 |

Como se observa en la tabla 13, en esta ocasión la clasificación inicial identificó 11,017 *bots* y 198,983 humanos. Dada la posibilidad de falsos positivos, se procedió a modificar 79,593 instancias clasificadas como humanos para someterlas a un análisis adicional. Esto permitió aplicar una serie de métricas para caracterizar estas instancias modificadas.

La prueba CUSUM arrojó un valor de 3,596.691, indicando cambios significativos en comparación con la muestra de valores humanos. Este valor indica una acumulación aún mayor de cambios o diferencias en comparación con el anterior. Además, el análisis del CUSUM negativo mostró un valor de -1,653.702, lo que sugiere que existen patrones de comportamiento que se desvían notablemente hacia abajo en relación a la media esperada, indicando posibles anomalías o comportamientos inusuales en las instancias analizadas.

La Entropía de Shannon presenta un ligero incremento en relación con el valor anterior, lo que significa una ligeramente mayor diversidad o aleatoriedad en los datos. Al igual que la entropía la Distancia Jensen-Shannon presenta un pequeño incremento en relación con el valor anterior, lo que indica una ligeramente mayor diferencia entre las distribuciones de probabilidad de los datos. Este cambio sutil sugiere que las distribuciones de probabilidad o los patrones de tráfico de red podrían estar comenzando a divergir ligeramente. La distancia de Mahalanobis presenta una notable disminución en comparación con el valor anterior, lo que indica una menor diferencia entre los puntos de datos en términos de sus características. Este descenso sugiere que los datos de tráfico de red podrían estar mostrando una mayor cohesión o homogeneidad lo cual puede ser indicativo de la reducción de valores anómalos. Finalmente, el IQR y MAD, con valores de 0.562 y 0.273 respectivamente, presentan un ligero decremento en relación con la ejecución anterior, por lo que todavía se puede afirmar que hay una variabilidad considerable en los comportamientos capturados.

Posteriormente, mediante el nuevo enfoque implementado, se asignó un valor de clase “No *bots*” para la caracterización realizada por las métricas. Este hallazgo se añade a la base de hechos, enriqueciendo el entendimiento sobre las características y comportamientos asociados con los *bots* para futuras interacciones.

#### 3.3.4.2-Análisis de los resultados

Basándose en estas métricas, se puede inferir que el conjunto de datos de tráfico de red presenta una combinación de estabilidad y variabilidad. Hay evidencia de cambios significativos (CUSUM), diversidad y complejidad (Entropía de Shannon), similitudes en la distribución de los datos (Distancia Jensen-Shannon), diferencias notables entre los puntos de datos (Distancia Mahalanobis), y una relativa concentración de los datos alrededor de la media (IQR y MAD). En esta ocasión, al igual que en el primer caso, el enfoque no determinó un cambio de naturaleza al modificar los datos clasificados como humanos, en contraposición con la primera ejecución del caso.

## 3.4 – Conclusiones parciales

Al realizar un proceso de experimentación y validación de la solución propuesta, se puede concluir que:

* En el enfoque basado en entropía para una entrada de 200,000 datos, los mejores resultados se obtienen con tamaños de paquete de 5,000 instancias, sin embargo, para entradas de 100,000 y 300,000 estos valores se obtienen con paquetes de 25,000 instancias.
* A partir del estudio de casos se ve que, en un alto porcentaje de los casos con una mayor cantidad de instancias humanas modificadas, no se aprecia cambio en la naturaleza de los mismos, a diferencia de las ejecuciones con menos instancias a procesar.
* La caracterización de datos siempre es añadida a la base de hechos.
* Debido a que la base de hechos cuenta con pocos datos, no se encontró una instancia a modificar.

# Conclusiones generales

Al término del presente trabajo, se llegan a las siguientes conclusiones:

* El meta aprendizaje facilita la caracterización de datos, al adoptar un enfoque de aprendizaje métrico. Las métricas utilizadas permiten identificar y clasificar datos de *bots*, contribuyendo así a la detección.
* La transición de métricas de complejidad a métricas de caracterización amplía la capacidad del módulo
* para manejar datos no etiquetados, mejorando su flexibilidad y efectividad en el análisis.
* La implementación de una función que simula cambios en los datos sin introducir elementos aleatorios contribuye a mantener la consistencia en el análisis.
* En el enfoque basado en entropía, al procesar una entrada de 200,000 datos, se obtienen los mejores resultados utilizando paquetes de 5,000 instancias. Sin embargo, para entradas de 100,000 y 300,000 datos, los resultados óptimos se logran con paquetes de 25,000 instancias.
* El estudio de casos revela que, en un alto porcentaje de las ejecuciones con una mayor cantidad de instancias humanas modificadas, no se observa un cambio en la naturaleza de estas. Esto contrasta con las ejecuciones que procesan un menor número de instancias, donde se aprecian variaciones más significativas.

# Recomendaciones

Al concluir el presente trabajo se tiene como recomendaciones:

* Expandir la cantidad de tamaños de entrada admisibles por el algoritmo de selección de etiqueta para la caracterización de datos, a través de un proceso de generación de reglas más exhaustivo.
* Experimentar con el uso de otros tipos de entropía además de la Entropía de Shannon.
* Analizar el comportamiento del meta-componente cuando se ejecuta usando un modelo diferente a Proactive Forest.

# Referencias Bibliográficas

[1] A. R. Vishwakarma, "Network Traffic Based Botnet Detection Using Machine Learning," 2020.

[2] S. Rovetta, G. Suchacka, and F. Masulli, "Bot recognition in a Web store: An approach based on unsupervised learning," *Journal of Network and Computer Applications,* vol. 157, p. 102577, 2020.

[3] H. Xu *et al.*, "Detecting and characterizing web bot traffic in a large e-commerce marketplace," in *Computer Security: 23rd European Symposium on Research in Computer Security, ESORICS 2018, Barcelona, Spain, September 3-7, 2018, Proceedings, Part II 23*, 2018: Springer, pp. 143-163.

[4] D. Cerqueira, "INTRODUCCIÓN AL CONCEPTO DE BOTS," p. 4, June 2023 2023.

[5] A. A. Ahmed, W. A. Jabbar, A. S. Sadiq, and H. Patel, "Deep learning-based classification model for botnet attack detection," *Journal of Ambient Intelligence and Humanized Computing,* vol. 13, no. 7, pp. 3457-3466, 2022.

[6] A. Niranjan, K. Akshobhya, P. D. Shenoy, and K. Venugopal, "EKNIS: Ensemble of KNN, Naïve Bayes Kernel and ID3 for Efficient Botnet Classification Using Stacking," in *2018 International Conference on Data Science and Engineering (ICDSE)*, 2018: IEEE, pp. 1-6.

[7] H. J. a. Chissingui, "Mecanismo de detección de bots basado en caracterización de datos," 2020.

[8] J. Hernández Orallo, "Introducción a la Minería de Datos," 2004.

[9] D. P. Echevarría, "Módulo de detección de bots malignos basado en aprendizaje automático," Grado, Universidad Tecnológica de La Habana José Antonio Echeverría CUJAE, 2022.

[10] J. A. P. S. y. D. L. R. Oviedo, "Algoritmos de detección de Bots para el comercio electrónico," Tesis para optar por el título de ingeniero informático, Universidad Tecnológica de La Habana “José Antonio Echeverría” -CUJAE, 2021.

[11] M. Mohammed, M. B. Khan, and E. B. M. Bashier, *Machine learning: algorithms and applications*. Crc Press, 2016.

[12] N. Cepero-Pérez, L. A. Denis-Miranda, R. Hernández-Palacio, M. Moreno-Espino, and M. García-Borroto, "Proactive forest for supervised classification," in *International Workshop on Artificial Intelligence and Pattern Recognition*, 2018: Springer, pp. 255-262.

[13] L. Rokach, "Decision forest: Twenty years of research," *Information Fusion,* vol. 27, pp. 111-125, 2016.

[14] L. A. D. Miranda and R. H. P. Palacio, "Construcción proactiva de bosques de decisión," Universidad Tecnológica de La Habana

“José Antonio Echeverría”(Cujae), 2018.

[15] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM,* vol. 55, no. 10, pp. 78-87, 2012.

[16] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE transactions on pattern analysis and machine intelligence,* vol. 44, no. 9, pp. 5149-5169, 2021.

[17] J. Castro Turga, "Evaluación de la entropía como métrica para detección y clasificación de ataques en base al tráfico de red," 2020.

[18] Imperva, "Bad Bot Report," Imperva, California, USA Technical report 2024.

[19] D. Doran and S. S. Gokhale, "Web robot detection techniques: overview and limitations," *Data Mining and Knowledge Discovery,* vol. 22, pp. 183-210, 2011.

[20] N. Nassar and G. Miller, "Method for a two dimensional honeypot to deter web bots in commerce systems," in *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, 2012: IEEE, pp. 250-256.

[21] H. A. Madni, Z. Anwar, and M. A. Shah, "Data mining techniques and applications—A decade review," in *2017 23rd international conference on automation and computing (ICAC)*, 2017: IEEE, pp. 1-7.

[22] W. Ertel, *Introduction to artificial intelligence*. Springer, 2018.

[23] R. S. S. Moorthy and N. Nathiya, "Botnet detection using artificial intelligence," *Procedia Computer Science,* vol. 218, pp. 1405-1413, 2023.

[24] M. G. Gómez-Zermeño, "Libro de Investigación: Inteligencia Artificial: Conceptos clave y experiencias en prácticas educativas," *Portal de Investigación,* 2023.

[25] I. Rish, "An empirical study of the naive Bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, 2001, vol. 3, no. 22: Seattle, WA, USA;, pp. 41-46.

[26] L. Breiman, "Random forests," *Machine learning,* vol. 45, pp. 5-32, 2001.

[27] D. P. Echevarría, N. C. Pérez, and H. D. Pando, "Una aplicación del algoritmo proactive Forest para la detección de bots malignos," *Revista Cubana de Transformación Digital,* vol. 4, no. 2, pp. 199: 1-14, 2023.

[28] C.-X. Zhang and J.-S. Zhang, "RotBoost: A technique for combining Rotation Forest and AdaBoost," *Pattern recognition letters,* vol. 29, no. 10, pp. 1524-1536, 2008.

[29] S. Touzani, J. Granderson, and S. Fernandes, "Gradient boosting machine for modeling the energy consumption of commercial buildings," *Energy and Buildings,* vol. 158, pp. 1533-1543, 2018.

[30] F. G. Mohammadi, M. H. Amini, and H. R. Arabnia, "An introduction to advanced machine learning: Meta-learning algorithms, applications, and promises," *Optimization, Learning, and Control for Interdependent Complex Networks,* pp. 129-144, 2020.

[31] Y. M. Herrera, "Conjunto de Métricas para la evaluación de Bases de Datos Relacionales," Universidad de las Ciencias Informáticas, 2009.

[32] R. Adrogué, "Precios de Transferencias. Rango Intercuartil," *Estudio Adrogué. Buenos Aires,* p. 5, 2010.

[33] S. Gorard, "Introducing the mean absolute deviation ‘effect’size," *International Journal of Research & Method in Education,* vol. 38, no. 2, pp. 105-114, 2015.

[34] M. T. E. Portillo and J. A. S. Plata, "P. CH. Mahalanobis y las aplicaciones de su distancia estadística," *CULCyT: Cultura Científica y Tecnológica,* vol. 5, no. 27, pp. 13-20, 2008.

[35] D. C. Montgomery, *Introduction to statistical quality control*. John wiley & sons, 2019.

[36] O. Salem, F. Naït-Abdesselam, and A. Mehaoua, "Anomaly detection in network traffic using Jensen-Shannon divergence," in *2012 IEEE International Conference on Communications (ICC)*, 2012: IEEE, pp. 5200-5204.

[37] P. Virtanen *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature methods,* vol. 17, no. 3, pp. 261-272, 2020.

[38] H. A. Sturges, "The choice of a class interval," *Journal of the american statistical association,* vol. 21, no. 153, pp. 65-66, 1926.

[39] D. W. Scott, "On optimal and data-based histograms," *Biometrika,* vol. 66, no. 3, pp. 605-610, 1979.

[40] D. Freedman and P. Diaconis, "On the histogram as a density estimator: L 2 theory," *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete,* vol. 57, no. 4, pp. 453-476, 1981.

[41] M. L. Hickman Fulp, "Dynamic Reduction of Scientific Data Through Spatiotemporal Properties," 2021.

[42] A. P. Pérez, "BASE DE DATOS CENTRALIZADA PARA ENTRENAR ALGORITMOS DE DETECCIÓN DE BOTS," p. 40, 2024.

# Anexos

Anexo 1: Flujo de datos para cada escenario, tomado de [9]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Escenario | Flujos de fondo (%) | Flujos de bots (%) | Flujos normales (%) | Flujos Totales |
| 1 | 97.47 | 1.41 | 1.07 | 2824636 |
| 2 | 98.33 | 1.15 | 0.5 | 1808122 |
| 3 | 96.94 | 0.561 | 2.48 | 4710638 |
| 4 | 97.58 | 0.154 | 2.25 | 1121076 |
| 5 | 95.7 | 1.68 | 3.6 | 129832 |
| 6 | 97.83 | 0.82 | 1.34 | 558919 |
| 7 | 98.47 | 1.5 | 1.47 | 114077 |
| 8 | 97.32 | 2.57 | 2.46 | 2954230 |
| 9 | 91.7 | 6.68 | 1.57 | 2753884 |
| 10 | 90.67 | 8.112 | 1.2 | 1309791 |
| 11 | 89.85 | 7.602 | 2.53 | 107251 |
| 12 | 96.99 | 0.657 | 2.34 | 325471 |
| 13 | 96.26 | 2.07 | 1.65 | 1925149 |

Anexo 2: Descripción de las columnas de CTU-13, tomado de [9]

