UNIVERSIDAD TECNOLÓGICA ISRAEL

ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN SEGURIDAD INFORMÁTICA

Resolución: **RPC-SO-02-No.053-2021**

PROYECTO DE TITULACIÓN EN OPCIÓN AL GRADO DE MAGÍSTER

Título del proyecto:
Evaluación del desempeño de algoritmos de machine learning dentro de la IA para uso en la búsqueda de patrones de ciberataques y mitigación de su impacto
LÍNEA DE INVESTIGACIÓN:
Ciencias de la ingeniería aplicadas a la producción, sociedad y desarrollo sustentable
Campo amplio de conocimiento:
Tecnologías de la información y la comunicación (TIC)
Autora:
Wendy Viviana Obregón Martínez
Tutor:
Mg. Renato Toasa PhD Maryory Urdaneta

Quito – Ecuador

2024

APROBACIÓN DEL TUTOR

Yo, **Renato Mauricio Toasa Guachi**, con C.I: **1804724167** en mi calidad de Tutor del proyecto de investigación titulado: EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO.

Elaborado por: **Wendy Viviana Obregón Martínez**, de C.I: **1310422538**, estudiante de la Maestría: Seguridad Informática, de la **UNIVERSIDAD TECNOLÓGICA ISRAEL (UISRAEL)**, como parte de los requisitos sustanciales con fines de obtener el Título de Magister, me permito declarar que luego de haber orientado, analizado y revisado el trabajo de titulación, lo apruebo en todas sus partes.

Quito D.M., marzo de 2024

Firmado electrónicamente por:
RENATO MAURICIO
TOASA GUACHI

Mg. Toasa Guachi Renato Mauricio
ORCID: 0000-0002-2138-300X

APROBACIÓN DEL TUTOR

Yo, **Urdaneta Herrera Maryory**, con C.I: **1759316126** en mi calidad de Tutor del proyecto de investigación titulado: EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO.

Elaborado por: **Wendy Viviana Obregón Martínez**, de C.I: **1310422538**, estudiante de la Maestría: Seguridad Informática, de la **UNIVERSIDAD TECNOLÓGICA ISRAEL (UISRAEL)**, como parte de los requisitos sustanciales con fines de obtener el Título de Magister, me permito declarar que luego de haber orientado, analizado y revisado el trabajo de titulación, lo apruebo en todas sus partes.

Quito D.M., marzo de 2024

Firmado electrónicamente por:
MARYORY URDANETA HERRERA

PhD. Urdaneta Herrera Maryory
ORCID: 0000-0001-8773-5349

DECLARACIÓN DE AUTORIZACIÓN POR PARTE DEL ESTUDIANTE

Yo, Wendy Viviana Obregón Martínez con C.I: 1310422538, autor/a del proyecto de titulación denominado: EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO.

Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar el respectivo trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

1. Manifiesto mi voluntad de ceder a la Universidad Tecnológica Israel los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor@ del trabajo de titulación, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital como parte del acervo bibliográfico de la Universidad Tecnológica Israel.
2. Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de prosperidad intelectual vigentes.

Quito D.M., marzo 2024

Firmado electrónicamente por:
WENDY VIVIANA
OBREGON MARTINEZ

Firma

Orcid: 0000-0003-2402-6670

Tabla de contenidos

Índice de tablas

Índice de figuras

INFORMACIÓN GENERAL

En las últimas décadas, ha habido un notorio avance en la inteligencia artificial (IA), la cual ha probado su eficacia en diversas áreas, y su potencial para contribuir en la detección de ataques cibernéticos es notable y como ejemplos tenemos los motores de búsqueda, sistemas de recomendación, análisis de datos, diagnósticos médicos, predicción del clima, juegos, clonación de tarjetas, ingeniería social y más.

Contextualización del tema

Actualmente los ciberataques representan una amenaza crítica que requiere abordarse de manera efectiva. La ciberdelincuencia presenta riesgos tangibles y cuantificables. Por otro lado, la seguridad informática está en constante cambio y adaptación, y la incorporación de técnicas de IA se ha vuelto esencial para identificar y neutralizar riegos a los que se enfrentan las entidades. La evolución continúa de las tecnologías de la información y comunicación hace indispensable adoptar medidas de ciberseguridad con el fin de proteger la privacidad, autenticidad y disponibilidad de la información. Puesto que es necesario desarrollar habilidades para detectar y controlar oportunamente las nuevas formas de amenazas que emergen constantemente (Flores, 2020).

La optimización de software evoluciona; la IA abarca algoritmos que simulan inteligencia humana, incluyendo aprendizaje automático y toma de decisiones, mejorando los resultados. Su aplicación en optimización de software automatiza mejoras y aborda problemas complejos, también se usa para detectar cuellos de botella, elige algoritmos eficientes y adapta soluciones dinámicamente. En definitiva, la IA potencia la eficiencia y efectividad de aplicaciones (Luzuriaga et al., 2023, p. 51).

El aprendizaje automático fusiona conceptos de varias disciplinas y ha impulsado la IA. En ciberseguridad, su uso es eficaz al abordar problemas con precisión y aprovechar grandes conjuntos de datos. Ha contribuido al avance de la seguridad informática al permitir un estudio detallado y ataques más precisos (Pinilla, 2020, p. 14).

Problema de investigación

La ciberseguridad enfrenta desafíos como la identificación de intrusiones, protección de la privacidad y enfoque preventivo ante amenazas cambiantes y sofisticadas. Se exploran métodos basados en IA para el análisis y las decisiones inmediatas, agilizando la detección y respuesta a los ciberataques. La inteligencia artificial se emplea para desarrollar sistemas que pueden adaptarse automáticamente y para automatizar las respuestas a las amenazas cibernéticas (Ayerbe, 2020, p. 2).

Esto ocasiona que la naturaleza de los ciberataques sea cada vez más astuta lo que hace que la identificación de intrusiones sea un desafío constante. Los atacantes utilizan técnicas avanzadas para eludir las defensas tradicionales, lo que complica la tarea de detectar actividades maliciosas.

Por esta razón este trabajo propone realizar una evaluación de desempeño de algoritmos de machine learning en la Inteligencia artificial (IA), con el propósito de fortalecer la defensa de las redes y sistemas informáticos para mejorar la detección de amenazas en los ciberataques.

¿Cómo los algoritmos de la machine learning en IA permiten identificar patrones de comportamiento de amenazas y responder de manera más efectiva a los ataques?

Objetivo general

Evaluar el desempeño de algoritmos y modelos de machine learning capaces de encontrar patrones para fortalecer las defensas y disminuir el impacto de los ciberataques.

Objetivos específicos

- Contextualizar fundamentos teóricos aplicados en los modelos de machine learning y ciberseguridad.
- Determinar los patrones y comportamientos de ataques sobre aplicaciones y dispositivos conectados a una red.
- Evidenciar la efectividad que el uso de algoritmos de machine learning en IA para la identificación, protección y mitigación del impacto de las amenazas.
- Validar la presente investigación con el criterio de un especialista.

Vinculación con la sociedad y beneficiarios directos:

Esta investigación contribuye y aporta a diversos Objetivos de Desarrollo Sostenible (ODS), como el objetivo 17, dado que la utilización de estas tecnologías ayuda en la prevención de ataques cibernéticos y puede estimular colaboraciones estratégicas en favor del desarrollo sostenible entre naciones. Además, se encuentra en sintonía con el objetivo 9, ya que la introducción de tecnologías como la IA y el machine learning promueve el avance e innovación de la infraestructura tecnológica en el ámbito de la seguridad informática.

Este trabajo tiene como finalidad determinar algoritmos de machine learning utilizados por las tecnologías con IA, con el fin de mejorar la capacidad de defensa de las redes y sistemas informáticos contra los ciberataques. La finalidad es proporcionar una referencia útil para organizaciones y grupos de interés, con el propósito de prevenir posibles impactos peligrosos, como la pérdida de datos, amenazas recurrentes, interrupciones operativas y la

exposición de información confidencial. La intención es poner a disposición un método que agilice la identificación de las causas principales asociadas con estos riesgos. De este modo se pretende ofrecer una perspectiva más integral para la detección de ciberataques y mejorar la capacidad de defensa de redes y sistemas informáticos.

Adicionalmente puede aportar otros métodos de defensa a los profesionales informáticos que luchan contra los delitos cibernéticos, ya que aportaría como una solución a utilizarse en las estructuras de los apoyos en las cooperaciones internacionales sobre estos delitos.

CAPÍTULO I: DESCRIPCIÓN DEL PROYECTO

Los algoritmos de aprendizaje de machine learning en IA para la ciberseguridad ofrecen una capacidad mejorada de detección de amenazas, respuesta rápida, adaptabilidad a nuevos ataques, reducción de falsos positivos y automatización de tareas repetitivas.

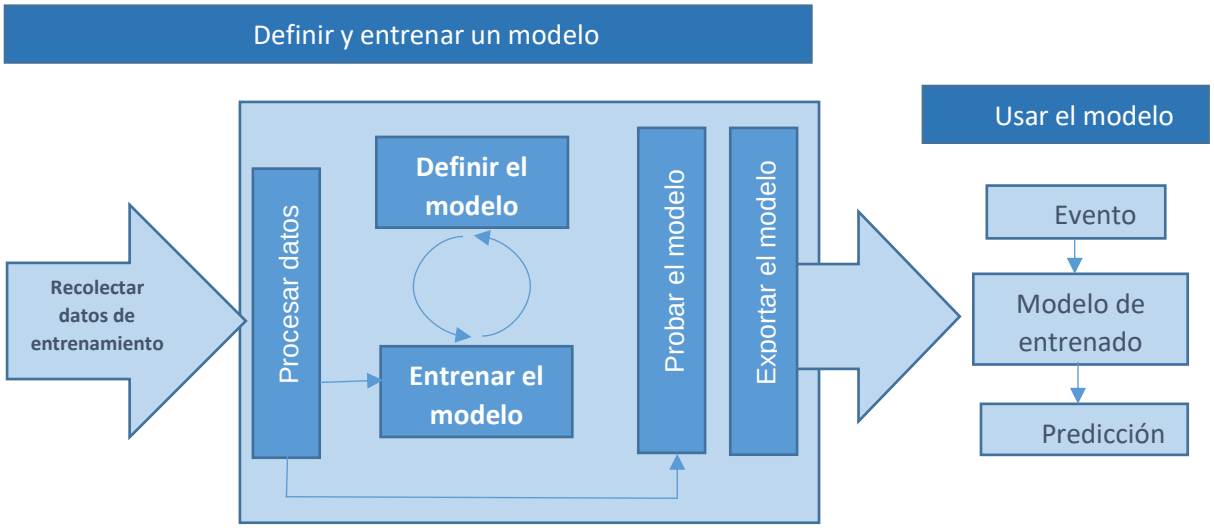
1.1. Contextualización general del estado del arte

Con el progreso tecnológico, la IA sigue expandiendo sus horizontes y presenta un potencial transformador en varios sectores y facetas de la vida cotidiana. En el panorama actual la ciberseguridad es una preocupación creciente debido al aumento de las amenazas y ataques cibernéticos. Las organizaciones, gobiernos y personas particulares necesitan adoptar enfoques preventivos para resguardar sus sistemas e información y mantenerse al tanto de las últimas tendencias y soluciones en seguridad cibernética para adaptarse a un entorno en constante evolución.

La inteligencia artificial se presenta como una herramienta valiosa para apoyar a profesionales especializados en lidiar con la creciente complejidad de las plataformas de tecnología de la información (IT) en uso en la actualidad, la industria 4.0 y la infraestructura del Internet de las Cosas (IoT). Además, puede ayudar a gestionar la considerable cantidad de datos generados por estos sistemas, con el objetivo de adelantarse a los posibles ciberataques. La seguridad cibernética enfrenta diversos desafíos, incluyendo la detección de intrusiones, la salvaguardia de la privacidad, la preparación proactiva, la identificación de patrones anómalos y el reconocimiento de amenazas avanzadas. No obstante, el principal desafío radica en adaptarse a las amenazas cambiantes que emergen constantemente (Ayerbe, 2020, p. 2).

El continuo desarrollo en el ámbito de mejora de programas informáticos ha visto la emergencia de la inteligencia artificial (IA) como una herramienta fundamental para lograr resultados más efectivos; la IA se centra en desarrollar algoritmos y sistemas que pueden ejecutar funciones que tradicionalmente demandarían habilidades humanas, como el aprendizaje automático y la toma de decisiones, sistemas capaces de aprender, razonar, planificar y reconoces patrones (Luzuriaga et al., 2023, p. 49). Ver figura 1.

Figura 1.
Flujo de procesos de aprendizaje automático.



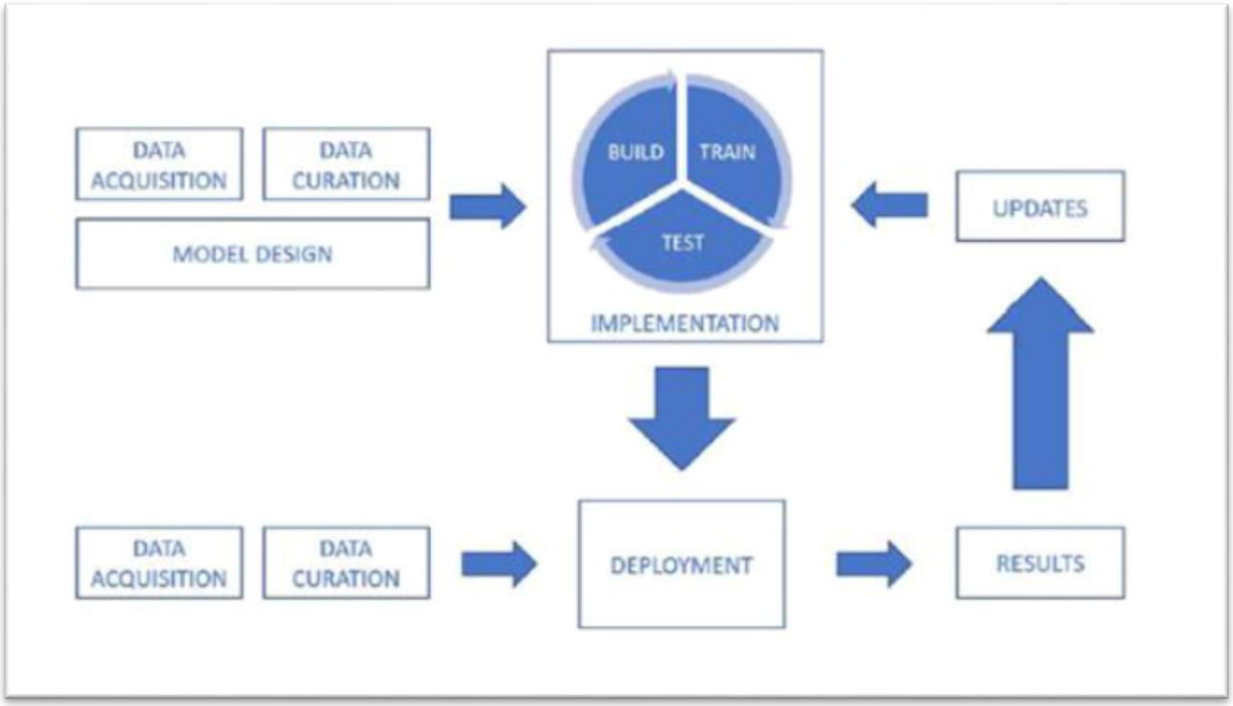
Nota: Diseño basado en "De Nils Ackermann Licenciado bajo Creative Commons CC BY-ND 4.0".

La ciberseguridad nace por la necesidad de salvaguardar la información frente a intrusiones de software malicioso que roban datos privados y exponen proyectos en desarrollo. Se han desarrollado estrategias, como la inteligencia artificial con las redes neuronales autónomas (RNA), que utiliza aprendizaje autónomo para contrarrestar ataques y mejorar la seguridad informática. En los años 80, se resaltó la relevancia de la ciberseguridad, y desde entonces, la IA ha sido aplicada para mitigar vulnerabilidades en sistemas y páginas web. En 1998, se seleccionaron algoritmos de aprendizaje para RNA dentro del ámbito de Seguridad de la Información (SI), entrenándolos y evaluándolos para el reconocimiento de intrusos en redes informáticas y preservar la privacidad de la información. (Castellanos et al., 2020, p. 60).

Detectar ataques no registrados, dirigidos a vulnerabilidades aún no identificadas en el sistema operativo o software (conocidos como Zero Day), presenta un desafío tecnológico significativo. Aquellos que desarrollan malware están familiarizados con las debilidades de los sistemas de seguridad y se esfuerzan por superar las defensas del mercado. Con el propósito de evitar e identificar posibles ataques, se utilizan sistemas asociación o correlación que integran métodos de Inteligencia Artificial, especialmente Machine Learning (ML). No obstante, si el malware logra eludir las defensas perimetrales, firewalls, filtros como de aplicaciones web y las políticas Zero Trust, la última barrera de protección se encuentra en el agente protector integrado al sistema operativo del dispositivo. La interrogante crucial es: ¿amigo o enemigo? En la actualidad, una cámara de vigilancia puede autónomamente reconocer rostros humanos, y un robot puede identificar defectos en una línea de ensamblaje. ¿Es viable aplicar el mismo principio para reconocer código malicioso antes de ser

almacenado en la memoria, eliminando la necesidad de recurrir a un sistema remoto? Algunos creadores afirman que sí, mediante el entrenamiento de redes neuronales (NN) utilizando tecnologías de Deep Learning (DL) (Portela, 2022, p. 5). Ver figura 2.

Figura *2.*
Proceso de machine learning.



Nota: Basado en “ETSI, Securing Artificial Intelligence (SAI); AI Threat Ontology 9”.

1.2. Proceso investigativo metodológico

Se describen los métodos de estudio empleados:

Investigación documental

La investigación documental, una metodología cualitativa de investigación, se centra en recopilar y elegir información a partir de fuentes como libros, revistas, grabaciones y periódicos, entre otros. Incluye la observación y examinar la información, reconocer y elegir elementos, y relacionarlos con el tema de investigación. También se denomina investigación bibliográfica y se basa en información derivada de fuentes segundarias. Su objetivo es unir datos preexistentes de diversas fuentes y proporcionar una visión sistemática de una cuestión específica que está dispersa en múltiples fuentes (Reyes, 2020).

Para esta investigación fue crucial utilizar criterios de búsqueda identificando palabras claves y conceptos fundamentales relacionados con la investigación propuesta, en bases de datos especializadas con datos académicos (libros, artículos académicos, revistas científicas, informes), bibliotecas digitales, en documentos que abordaban el tema y que estuviera considerada su credibilidad y la relevancia de sus fuentes.

Investigación descriptiva

La investigación descriptiva se centra en destacar los atributos de la población estudiada. Según Mario Tamayo en el año 1994, implica el proceso de registrar, analizar e interpretar la forma y estructura actuales de los fenómenos, centrándose en conclusiones predominantes o en cómo personas, grupos o cosas se comportan en el presente. En el año 1992 Carlos Sabino, en "El proceso de investigación", La caracteriza como una forma de investigación que puede describir aspectos esenciales de grupos uniformes de fenómenos. Utiliza criterios sistemáticos para comprender la forma o la conducta de los acontecimientos en análisis, generando información sistemática y comparativa con otras fuentes (Guevara et al., 2020).

Esta investigación tiene como contexto comprender en detalle la funcionalidad y eficacia de los algoritmos de machine learning aplicados a la ciberseguridad, para proporcionar una visión clara de su desempeño y su aplicación en situaciones prácticas.

Describir e identificar sus características principales, capacidades y limitaciones, así como también seleccionar una muestra representativa de datos de ciberataques y eventos relacionados para realizar la medición del desempeño para poder mostrar los resultados en un compara y contrastar resaltando sus fortalezas y debilidades.

1.3. Análisis de resultados

Para validar la propuesta de investigación sobre el desempeño de los algoritmos de machine learning dentro de la IA para uso en la búsqueda de patrones de ciberataques y mitigación de su impacto, se han evaluado las siguientes aristas:

Según Flores, (2020 p. 1) en la actualidad, los ciberataques representan una amenaza crítica que requiere abordarse de manera efectiva. La ciberdelincuencia presenta riesgos tangibles y cuantificables. La seguridad informática está en constante cambio y adaptación, y la incorporación de técnicas de inteligencia artificial se ha vuelto esencial para identificar y contrarrestar desafíos que confrontan las organizaciones. En una situación donde se producen progresos continuos en las TICs, resulta esencial implementar medidas de seguridad cibernética con el propósito de salvaguardar la privacidad, autenticad y accesibilidad de los datos, además es necesario desarrollar habilidades para detectar y controlar oportunamente las nuevas formas de amenazas que emergen constantemente.

Durante los años recientes, se han llevado a cabo iniciativas para crear soluciones basadas en inteligencia artificial (IA) en diversas aplicaciones de seguridad cibernética, debido al aumento en la comprensión de las organizaciones sobre la importancia de la IA en la mitigación de amenazas cibernéticas. Ejemplos de IA en la clasificación no lineal han demostrado ser útiles para la identificación de amenazas. Los avances en la capacidad de

cómputo, evidenciados por la reducción en el tiempo de entrenamiento de sistemas de clasificación de imágenes en la nube, están impulsando el interés en soluciones basadas en IA. Las capacidades informáticas para enfoques basados en IA se duplican cada tres meses, superando la Ley de Moore, lo que puede mejorar el rendimiento de las soluciones de seguridad cibernética (Zhimin et al., 2021, p. 2).

Según lo mencionado por (Flores, 2020) y (Zhimin et al., 2021) la ciberseguridad y las soluciones basadas en IA son temas que siempre están en constante evolución por esta razón se realiza esta investigación para plantear que el uso de algoritmos de machine learning en la detección de malware y de posibles atacantes en nuestra red, realizando pruebas con datos de una muestra representativa y así poder tener una idea más clara sobre la efectividad y su posible uso.

CAPÍTULO II: PROPUESTA

El presente proyecto tiene como finalidad proveer una visión más amplia sobre el rendimiento de los algoritmos de machine learning apoyados también en la inteligencia artificial para el aprendizaje de patrones de comportamiento para detección de ciberataques y mejorar la capacidad de defensa de las redes y sistemas informáticos.

2.1. Fundamentos teóricos aplicados

La ciberseguridad

La ciberseguridad abarca medidas, métodos y recursos para asegurar que los sistemas estén protegidos contra amenazas y vulnerabilidades, brindando servicios eficientes a los usuarios. Esto incluye la protección contra amenazas externas e internas, con un enfoque en la seguridad de red. El objetivo es minimizar el impacto de estas amenazas en la operación de los sistemas, mediante la detección anticipada, la gestión durante un incidente y la recuperación posterior a dicho incidente (Zhimin Z. H., 2021).

Conjunto de estrategias, técnicas y precauciones diseñadas para proteger los sistemas de información, redes, dispositivos electrónicos y datos digitales de posibles ataques, accesos no autorizados, robo de información, daños o interrupciones. La principal meta de la ciberseguridad consiste en asegurar la privacidad, integridad y accesibilidad de los activos digitales, resguardándolos de amenazas cibernéticas como virus, malware, ransomware, phishing, ataques de denegación de servicio (DDoS), y otras posibles vulnerabilidades La ciberseguridad involucra la implementación de medidas preventivas, detección temprana de intrusiones y reacción rápida ante incidentes de seguridad para mantener la protección y mantener un entorno digital seguro (https://www.ibm.com/topics/cybersecurity, s.f.).

La seguridad informática incluye una diversidad de campos y estrategias con el fin de asegurar la salvaguardia de los recursos digitales. Algunos de los elementos fundamentales de la ciberseguridad abarcan:

Seguridad de la red: comprende las actividades destinadas a preservar el control de acceso, la integridad y el adecuado funcionamiento de la red. Esto se logra a través de la aplicación de acciones, como la instalación de programas antivirus, la configuración de cortafuegos y la utilización de redes privadas (VPN) (Pérez, 2022).

Seguridad de los sistemas informáticos: Involucra proteger los sistemas operativos, aplicaciones y servidores contra vulnerabilidades y ataques maliciosos, como malware, ransomware y exploits (Jiménez y Ramírez, 2022).

Gestión de identidad y accesos: Se refiere a la administración y control de acceso a usuarios y los permisos a recursos digitales para garantizar que únicamente individuos autorizados tengan la capacidad de entrar a los datos (Jiménez-Almeira y Enrique López, 2023, p. 27).

Encriptación: La encriptación de datos o cifrado de archivos es un método de seguridad que garantiza la protección de la información (Cabanillas y Nizama, 2019).

Concienciación y entrenamiento: La formación de usuarios y empleados es esencial para mejorar la concientización acerca de los peligros informáticos y fomentar protocolos de seguridad en el área de la ciberseguridad, como la implementación de contraseñas robustas y la identificación de mensajes de correo electrónico fraudulentos (Jiménez-Almeira y Enrique López, 2023, p. 25).

Respaldo y recuperación de datos: La instalación de sistemas de respaldo y restauración aseguran que la información vital esté resguardada y pueda ser recuperada en situaciones de pérdida o daño (Jiménez-Almeira y Enrique López, 2023, p. 27).

Seguridad en la nube: La ciberseguridad también incluye la protección de los datos y servicios alojados en plataformas en la nube, lo que implica medidas adicionales debido a la naturaleza distribuida y compartida de estos entornos (Tenelema et al., 2020).

Patrones de comportamiento: Son secuencias o tendencias recurrentes de acciones, actividades o conductas que se observan en individuos, sistemas o grupos durante un período de tiempo. Estos patrones pueden ser identificados a través del análisis de datos y observación de comportamientos repetitivos y consistentes en diferentes situaciones (Pinilla, 2020, p. 10).

En la IA para ciberseguridad, los patrones de comportamiento son secuencias identificadas a partir del análisis de datos que representan actividades normales en sistemas y redes. Algoritmos de machine learning pueden reconocer estos patrones y detectar comportamientos anómalos que podrían indicar ataques o amenazas cibernéticas, permitiendo una rápida detección y respuesta ante posibles incidentes de seguridad.

Inteligencia Artificial

La IA simula la mente humana en sistemas informáticos para pensar, aprender y resolver problemas como humanos según Russell y Norvig. Se basa en representar el conocimiento y buscar soluciones según Nilsson. La IA capacita a máquinas para imitar el pensamiento humano y resolver problemas en medicina, manufactura y más, mejorando

eficiencia y precisión. Con el progreso tecnológico, la IA podría impactar la vida diaria, aunque plantea cuestiones éticas y de privacidad (Luzuriaga et al., 2023, p. 51)

Se trata de un ámbito de la informática que se concentra en la creación de sistemas y aplicaciones con la capacidad de llevar a cabo labores que generalmente demandan inteligencia humana. Estos sistemas buscan simular, emular o mejorar la capacidad de aprendizaje, razonamiento, percepción y toma de elecciones por parte de las personas. La inteligencia artificial se fundamenta en modelos y algoritmos que posibilitan a los equipos a realizar acciones similares a las ejecutadas por seres humanos, procesar datos, aprender de ellos y realizar acciones o tomar decisiones en función de esa información. La IA involucra una diversidad de usos que se extiende desde asistentes virtuales hasta sistemas de diagnóstico médico, autos autónomos, análisis de datos, juegos y muchas otras áreas donde se pueden aplicar capacidades inteligentes a las máquinas (Strand, 2019).

Existen diferentes tipos de aprendizaje dentro de la inteligencia artificial, algunos de los cuales incluyen: Ver figura 3.

Figura 3.
Ciberseguridad con inteligencia artificial y algoritmos de aprendizaje.

		Detección de Intrusos		Análisis de Malvare	Detección de SPAM	Phishing	Usuarios maliciosos	Datos anómalos
		Redes	Botnet					
Deep Learning	Supervisado	Redes neuronales profundas recurrentes (RNN)	Redes neuronales profundas recurrentes (RNN). Multi-capa Perceotron (MLP).	Redes neuronales profundas totalmente conectadas (FNN). Red neuronal convolucional (CNN). Redes neuronales profundas recurrentes (RNN). Multi-capa Perceptron (MLP).		Red Neuronal convolucional (CNN).	Red Neuronal convolucional (CNN) Red neuronal recurrente de memoria a corto plazo.	Aprendizaje profundo por transferencia (DTL).
	No supervisado	Redes neuronales profundas (DBN). Autocodificadores apilados (SAE).		Redes neuronales profundas (DBN). Autocodificadores apilados (SAE).	Redes neuronales profundas (DBN). Autocodificadores apilados (SAE).			

Nota: Elaboración propia basada en “A review of deep learning applied to cybersecurity”.

Aprendizaje Automático Supervisado: Los algoritmos de este tipo de aprendizaje se emplean para crear modelos de detección de intrusos que se enfocan en firmas, donde los datos de entrenamiento (aprendizaje) consisten en firmas de ataques previamente identificados. Los métodos comunes como soporte de máquinas vectoriales, linealización de regresión, regresión logarítmica, método de Naive Bayes, análisis lineal discriminante y estructuras de elección y redes neuronales (especialmente la Percepción multicapa) son ampliamente utilizados para construir mecanismos de identificación de intrusos basados en firmas. Estos enfoques de aprendizaje supervisado se ajustan según las firmas conocidas, también llamadas etiquetas o clasificadores de eventos, para lograr la detección. La

efectividad del sistema de detección de intrusiones fundamentado en firmas depende de la calidad de la selección de datos utilizados para el entrenamiento y de los parámetros empleados en los algoritmos de aprendizaje supervisado (Aljamal et al., 2019, p. 89).

Aprendizaje Automático No Supervisado: En los enfoques no supervisados del aprendizaje automático, los parámetros necesarios se obtienen de información sin categorización y luego se utilizan en datos o situaciones nuevas. Ambas categorías de enfoques tienen la capacidad de emplearse en situaciones de clasificación, donde se organizan elementos en grupos específicos, o en problemas de regresión, donde se realiza la predicción de propiedades numéricas de una observación (Pinilla, 2020, p. 16).

Redes Neuronales Artificiales: Una red neuronal artificial consiste en nodos conectados que interactúan, procesan y transmiten información en un ciclo computacional iterativo, generando salidas aproximadas a partir de datos reales. Sin embargo, el éxito de esta red depende de su entrenamiento, el cual está influenciado por diversos factores como la estructura de la red, el tamaño del conjunto de datos de entrenamiento y la función empleada en el proceso de formación. Para este propósito, se emplean diversos algoritmos de entrenamiento, entre ellos, backpropagation y feedforward, que se distinguen por su capacidad para proporcionar precisión en la generación de predicciones en situaciones reales (Castellanos et al., 2020).

Redes Generativas Antagónicas (GANs): Un ejemplo destacado de adiestramiento semisupervisado es una variante especial de estructura neuronal denominada GAN (Generative Adversarial Network, en inglés), en la cual se emplean datos de entrenamiento que cuentan con etiquetas para generar información adicional mediante la red generadora. Estos nuevos datos son luego evaluados por otra red neuronal llamada discriminativa. Las redes generativas y discriminativas mejoran a través de un proceso de retroalimentación positiva en un ciclo continuo. Esta red mejora al generar información falsa que sean convincentes, mientras que el discriminador se perfecciona al diferenciar entre los datos falsos y los datos reales de aprendizaje (Pinilla, 2020, p. 16).

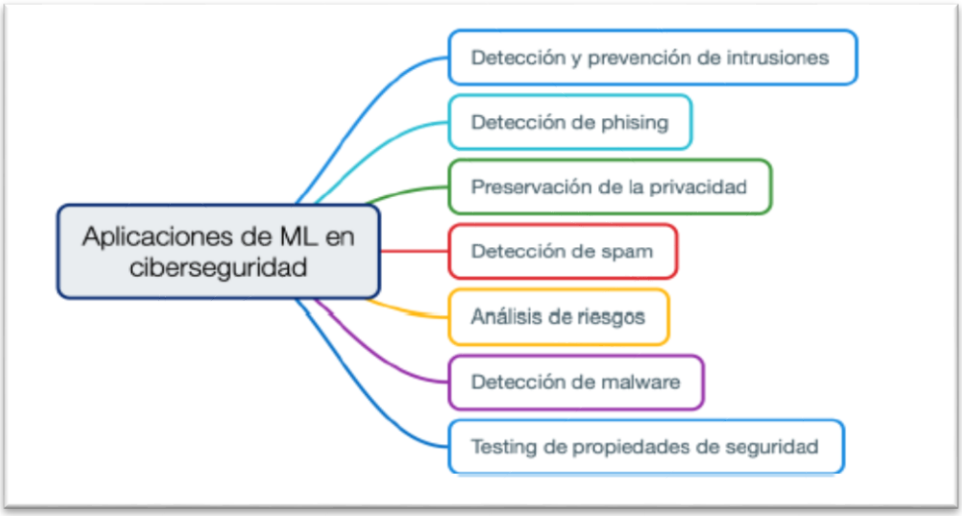
Machine Learning: Aprendizaje automático (Machine Learning) son un conjunto de métodos estadísticos que identifican patrones en datos automáticamente. Ha desempeñado un papel fundamental para el avance de la IA y se utiliza en seguridad informática para reconocimiento de patrones y detección de anomalías. Los modelos aprenden por experiencia y se adaptan óptimamente a los datos de entrenamiento. Hay tres enfoques en el campo aprendizaje automático: aprendizaje supervisado y no supervisado y reforzado (Pinilla, 2020, pp. 14-15).

En el aprendizaje supervisado, se utilizan firmas de ataques conocidos como datos de entrenamiento para crear modelos de detección de intrusos. Algunos algoritmos comunes son máquinas de soporte vectorial, regresión lineal, regresión logística, Naive Bayes, análisis discriminante lineal, árboles de decisión y redes neuronales (como la Percepción multicapa). Las técnicas de aprendizaje automático no supervisado construyen modelos basados en la relación, consistencia o continuidad entre eventos. Debido a la considerable cantidad de información generada por las redes de computación en la nube hace que el etiquetado sea costoso, el enfoque de aprendizaje no supervisado se considera una solución más adecuada en este caso (Aljamal et al., 2019, p. 85).

El aprendizaje reforzado se sustenta en la premisa de que recompensas positivas otorgadas por acciones beneficiosas pueden establecer comportamientos óptimos. En su esencia, implica un agente interactuando con su entorno, cuyo comportamiento se ve influenciado por la retroalimentación positiva o negativa de sus interacciones en situaciones específicas. Esta retroalimentación refuerza acciones deseables. Además, el aprendizaje profundo, a menudo confundido con el aprendizaje automático, es otro término ampliamente reconocido (Pinilla, 2020, p. 17).

Las principales utilidades del aprendizaje automático en el campo de la seguridad en los sistemas de información incluyen la identificación y prevención de intrusiones en sistemas, la detección de intentos de phishing, la salvaguarda de la privacidad del usuario, el reconocimiento de mensajes no deseados, la evaluación de riesgos, la detección de software malicioso y la prueba de las características de seguridad de un sistema (Dueñas, 2020). Ver figura 4.

Figura 4.
Aplicaciones de Machine Learning a la ciberseguridad.



Nota: Basada en aprendizaje supervisado para la detección de amenazas.

2.2. Descripción de la propuesta

El concepto de ciberseguridad incorpora un conjunto de estrategias, técnicas y medidas diseñadas para resguardar los sistemas de información, redes, dispositivos electrónicos y datos digitales de posibles incidentes maliciosos, accesos no autorizados, robo de información, daños entre otras vulnerabilidades.

El objetivo principal es asegurar la privacidad, autenticidad y accesibilidad de los activos digitales y protegerlos de amenazas cibernéticas, esto también implica la adopción de acciones preventivas, identificación temprana de intrusiones y reacción rápida ante tales incidentes. Con base en esto se busca analizar herramientas que utilicen algoritmos de machine learning con IA para potenciar la capacidad de reconocimiento de ciberataques.

Los algoritmos de machine learning pueden desempeñar un papel importante en búsqueda de patrones de seguridad para poder mitigar el impacto de los ciberataques, puesto que ayudarían en procesos de detección de anomalías como son los comportamientos normales de los sistemas y redes para identificar patrones extraños que podrían significar ataques en curso, por ejemplo detectar desviaciones inusuales en el tráfico de la red, el uso de los recursos entre otros indicadores claves.

Estos algoritmos pueden ser entrenados con un conjunto de datos etiquetados que contienen ejemplos de ciberataques y actividades legítimas. A medida que se alimenta al algoritmo con más datos, pueden aprender a reconocer patrones y características específicas asociadas con diferentes tipos de ataques.

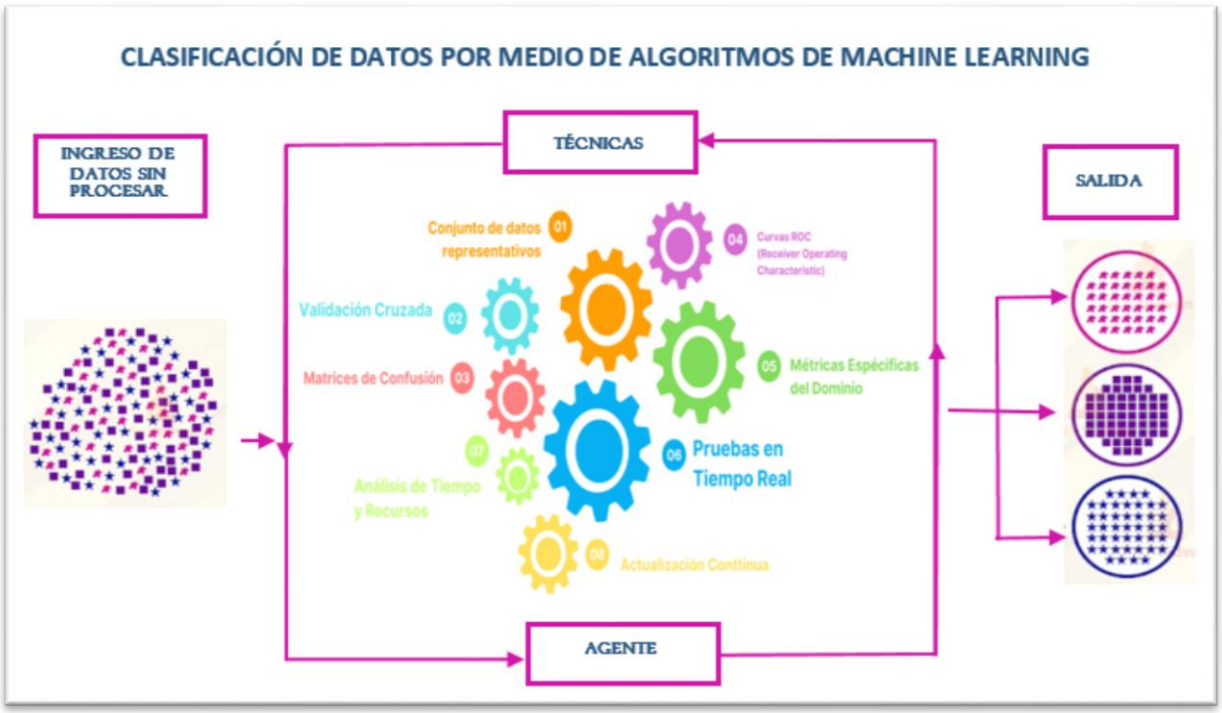
También se pueden utilizar redes neuronales profundas y otros enfoques de aprendizaje profundo que son efectivos para analizar datos no estructurados y extraer características complejas. Estos consiguen detectar amenazas incipientes al procesar grandes cantidades de registros de actividades y tráfico de red.

Poder analizar modelos de comportamiento para usuarios, dispositivos o sistemas; Identificado patrones típicos y alertar cuando se desvían de ellos, y de esta manera detectar actividades sospechosas y no autorizadas.

a. Estructura general

Se presenta una representación varias técnicas de evaluación de los datos aplicados para abordar a un problema de seguridad cibernética. En la figura 5.

Figura 5.
Técnicas de clasificación por algoritmos de Machine Learning.



b. Explicación del aporte

Las herramientas existentes para detección, prevención de intrusiones y de administración de eventos y de la seguridad de la información se pueden fácilmente adaptar a la propuesta de modelos de aprendizaje automático y ejecución de procesos sobre mitigación de la ciberseguridad, a razón de buscar generar más confianza en el uso de las tecnologías, disminuir el impacto de las pérdidas e incrementar la transparencia en los procesos de gestión.

Los algoritmos de machine learning en la IA pueden desempeñar un papel crucial en la detección de patrones de ciberataques y la mitigación de su impacto. Aquí hay algunas maneras en que estos algoritmos son útiles en este contexto:

- **Detección de anomalías:** utilizan modelos de aprendizaje no supervisado para identificar comportamientos inusuales en los datos de red. Esto permite detectar actividades sospechosas que podrían indicar un ciberataque en curso.
- **Análisis de Comportamiento:** Los algoritmos de aprendizaje automático pueden analizar el comportamiento normal de usuarios y sistemas para identificar desviaciones significativas. Esto facilita la detección de actividades maliciosas que podrían pasar desapercibidas mediante métodos tradicionales.
- **Modelos Predictivos:** Mediante el uso de algoritmos predictivos, la IA puede anticipar posibles amenazas al analizar patrones históricos y tendencias. Esto ayuda en la

identificación temprana de posibles ciberataques y permite la toma de medidas preventivas

- **Detección de Amenazas Avanzadas Persistentes (ATP):** Los algoritmos de inteligencia artificial son eficaces para identificar patrones asociados con amenazas avanzadas persistentes. Estas amenazas suelen ser más complejas y persistentes, y los modelos de IA pueden ser entrenados para reconocer sus características distintivas.
- **Respuesta Automatizada:** Algoritmos de IA pueden automatizar la respuesta a incidentes de seguridad, permitiendo respuestas más rápidas y precisas ante amenazas. Esto implica la habilidad de separar sistemas comprometidos, impedir el acceso de direcciones IP perjudiciales y emprender acciones correctivas de forma autónoma.
- **Identificación de Firmas Maliciosas:** Los algoritmos pueden analizar patrones de código malicioso para identificar firmas específicas de malware. Esto facilita la detección de amenazas conocidas y la aplicación de medidas de seguridad específicas.
- **Mejora continua:** La IA puede adaptarse y mejorar continuamente a medida que se enfrenta a nuevas amenazas. Los modelos de machine learning pueden actualizarse con nuevos datos para mantenerse al día con las tácticas cambiantes de los ciberdelincuentes.

c. Técnicas

Es un trabajo de investigación de tecnología aplicada; combinando técnicas de análisis documental y de contenido sobre algoritmos de IA y su comportamiento en la ciberseguridad. Aquí hay algunas técnicas comunes utilizadas para evaluar algoritmos de IA en el contexto de la búsqueda de patrones de ciberataques y la mitigación de su impacto:

Conjunto de datos representativos: utilizan conjuntos de datos bien curados y representativos que contengan ejemplos de amenazas y comportamientos normales. Estos conjuntos deben abarcar diversas tácticas de ciberataques y reflejar la complejidad del entorno de seguridad.

Validación Cruzada: aplican técnicas de validación cruzada para evaluar la generalización del algoritmo. Esto implica fragmentar el conjunto de datos en varias secciones y entrenar/probar el modelo en diversas combinaciones, lo que ayuda a evaluar su rendimiento en diversas condiciones.

Matrices de Confusión: utilizan matrices de confusión para evaluar la capacidad del algoritmo para clasificar correctamente las instancias. Esto incluye la identificación de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, proporcionando una visión detallada del rendimiento.

Curvas ROC (Receiver Operating Characteristic): analizan las curvas ROC y calcular el área bajo la curva (AUC) para medir la capacidad de discriminación del modelo entre las clases de interés. Una AUC más alta indica un mejor rendimiento en la clasificación.

Métricas Específicas del Dominio: definen métricas específicas del dominio para evaluar el impacto real de los resultados del algoritmo en términos de seguridad. Esto podría incluir la eficacia para detectar amenazas críticas y la capacidad de mitigar el impacto de los ciberataques.

Pruebas en Tiempo Real: realizan pruebas en entornos de producción simulados o en tiempo real para evaluar el rendimiento del algoritmo en condiciones del mundo real. Esto permite identificar posibles problemas que pueden no ser evidentes en entornos controlados.

Resistencia a Ataques Adversarios: evalúan la resistencia del algoritmo a ataques adversarios. Los ciberdelincuentes pueden intentar engañar al sistema, y evaluar la robustez del algoritmo contra tales intentos es esencial.

Análisis de Tiempo y Recursos: evalúan el tiempo de ejecución y los recursos computacionales requeridos por el algoritmo. Es crucial que los algoritmos sean eficientes y prácticos para su implementación en entornos de ciberseguridad.

Actualización Continua: evalúan la capacidad del algoritmo para adaptarse a nuevas amenazas y actualizaciones en el tiempo. Los algoritmos de IA deben ser capaces de aprender y mejorar con el tiempo a medida que evolucionan las tácticas de los ciberdelincuentes.

La combinación de estas técnicas proporciona una evaluación integral de los algoritmos de inteligencia artificial en ciberseguridad, garantizando su efectividad y robustez en la detección y mitigación de ciberataques.

2.3. Validación de la propuesta

La evaluación y estudio de algoritmos de machine learning en la IA en ciberseguridad requieren enfoques rigurosos para asegurar su eficacia y confiabilidad. La fusión de elementos de aprendizaje automático e inteligencia artificial aplicados al ámbito de ciberseguridad, son una nueva tendencia innovadora que tienen herramientas con mayor capacidad para ajustarse a nuevas amenazas y ayudar en diversos tipos de aplicaciones.

Para la selección de algún algoritmo se debe de tomar en cuenta factores como la exactitud, la precisión, la interpretabilidad, la complejidad, la escalabilidad, el tiempo en construir, probar, entrenar y realizar predicciones.

Con base en la validación realizada por dos expertos, cuyos informes se encuentran detallados en los anexos 2 y 3, se indica lo siguiente:

Los algoritmos podrían haber llegado a analizar más información para ayudar a aumentar y mejorar el entrenamiento; pero eso se entiende que debe ser parte de la continuación del trabajo.

El estudio puede y debe ser complementado con otros trabajos a razón de dar continuidad a la propuesta, es necesario que se exploren más alternativas y evalúen los resultados, para lograr un entrenamiento más refinado de los algoritmos y modelos propuestos.

Se recomienda seguir realizando investigación sobre esta temática, ya que es un tema bastante interesante y amplio con grandes aportaciones a la seguridad informática del país y las empresas.

El aporte del trabajo es bastante amplio y podría tener varias aportaciones o publicaciones a posterior en base a las simulaciones de los algoritmos realizadas.

A continuación se realizan unos cuadros comparativos fortalezas y debilidades de la investigación realizada:

Red Neuronal Artificial

En el contexto de la ciberseguridad, el funcionamiento de las RNA implica la utilización de patrones de clasificación o un conjunto de datos representativos, los cuales tienen la capacidad de reconocer cada ataque previamente registrado en los datos. Esto se logra mediante una fase de entrenamiento que tiene como objetivo generar predicciones precisas frente a posibles vulnerabilidades y acciones alarmantes. Posteriormente, como parte de esta secuencia lógica, la red debe ser evaluada con datos desconocidos, buscando mejorar su capacidad predictiva. En última instancia, la interfaz que la red presentará es similar a un programa informático que proporciona instrucciones para su correcta utilización (Castellanos et al., 2020, p. 60). Tabla 1.

Tabla 1.
Cuadro comparativo de Foda – Red Neuronal Artificial

Debilidades	Fortalezas
• Discriminación por perfiles que cumplen ciertos perfiles.	• Análisis estadístico • Análisis dinámico Empleo de técnicas de esteganografía y esteganografía.
Amenazas	**Oportunidades**
• No caracterización de elementos diferenciales de perfiles online.	• Detección de amenazas y ataques cibernéticos.

Nota: Elaboración propia.

Aprendizaje profundo por transferencia (DTL Deep Transfer Learning) o Deep TL

Este enfoque posibilita la solución de problemas fundamentales asociados con conjuntos de datos de entrenamiento inapropiados. En consecuencia, se prescinde de la obligación de entrenar modelos de inteligencia artificial (IA), lo que facilita la capacitación de redes neuronales con volúmenes de datos relativamente bajos. En el campo de la ciencia de datos, su uso es frecuente en la actualidad, ya que en la mayoría de los casos del mundo real, no se dispone de grandes conjuntos de datos etiquetados para entrenar modelos altamente complejos. La transferencia de aprendizaje profundo (DTL) se puede categorizar en tres configuraciones diferentes (Quirumbay et al., 2022). Tabla 2

Tabla 2.
Cuadro comparativo de Foda - Aprendizaje Deep TL.

Debilidades	Fortalezas
• Investigación de métodos de recopilación de datos.	• Aprendizaje por transferencia inductiva. • Aprendizaje por transferencia transductiva. Aprendizaje por transferencia no supervisada.
Amenazas	**Oportunidades**
• Recolectar valores ambiguos, perdidos atípicos y datos sin sentido.	• Ahorro de tiempo de entrenamiento. • Mejora de precisión de resultados. • Necesidad de menos datos de entrenamiento.

Nota: Elaboración propia.

Protección del endpoint con aprendizaje profundo

Los sistemas de detección y respuesta en el punto final (EDR, XDR) equipos en un continuo proceso de examinación y evaluación. Normalmente, estos sistemas están integrados con un sistema SIEM (Security Information and Event Management) que correlaciona eventos y evalúa posibles amenazas, teniendo la capacidad de reaccionar al bloquear la ejecución y

propagación del software malicioso. Es relevante señalar que este enfoque implica la utilización de recursos de comunicación y cómputo, lo cual conlleva costos constantes las 24 horas del día, los 7 días de la semana, y requiere actualizaciones periódicas para asegurar su eficacia. (Portela, 2022, p. 4). Tabla 3.

Tabla 3.
Cuadro comparativo de Foda – Protección endpoint.

Debilidades	Fortalezas
• Falsos positivos.	• Gestionar grandes cantidades de datos en tiempo real. • Identificar ataques que no han sido previamente clasificados.
Amenazas	**Oportunidades**
• Falsos positivos.	• Mejora continua.

Nota: Elaboración propia.

En el anexo 1 se detalla una evaluación de muestras e interpretación de resultados incluida y a continuación se resume lo siguiente:

El conjunto de datos utilizado proviene de las muestras tomadas por (Agustín Parmisano, 2020), mismos que fueron reinterpretados con base en los siguientes elementos o etiquetas para la evaluación de las muestras y posterior análisis de resultados:

- **Ataque:** Sirve para determinar la ocurrencia de un ataque originado de un dispositivo infectado dirigido hacia otro equipo, se incluyen varios tipos tales como intentos de inicio de sesión en Telnet por ataque de fuerza bruta, inyecciones GET.
- **Benignidad:** Indica las conexiones no sospechosas o actividades maliciosas no detectadas.
- **Comando y Control:** Indica que el equipo infectado has establecido una conexión con un servidor de control y comandos.
- **DDoS (Denegación de servicio distribuida):** Asignada cuando un dispositivo infectado está activamente envuelto en un ataque de DDoS, identificado por el volumen de flujos dirigidos hacia una sola dirección IP.
- **FileDownload:** Significa que un archivo ha sido descargado al dispositivo infectado.
- **Mirai:** Etiqueta asignada a equipos que muestran características del botnet Mirai.
- **PartOfAHorizontalPortScan:** Utilizada cuando las conexiones están envueltas en un escaneo horizontal de puertos usados para una posterior recolección de información para ataques potenciales.

Para este estudio fue necesario hacer uso de librerías relevantes para Python para el análisis de datos; el uso de registros o archivos de servidores que contienen información sobre el tráfico de red sobre dispositivos IoT, además archivos o bases de datos que también contienen información acerca del tráfico de información en una red. Para esto se empleó una muestra aleatoria de 100000 registros.

Figura 6.
Muestra Aleatoria de registros.

Generación de una muestra aleatoria de 100000 datos

```
random_sample_df = original_df.sample(n=100000, random_state=42)

# Creación de una copia de la muestra del DataFrame aleatorio para evitar mod
ificar los datos originales
df = random_sample_df.copy()

file_path = '/content/CTU-IoT-Malware-Capture-1-1conn.log.labeled.csv'

original_df = pd.read_csv(file_path, delimiter='|')
df = original_df.copy()
```

Nota: Elaboración propia tomada de los anexos.

En el ejemplo se clasifica dos campos para revisar su equivalencia y luego se realiza una correlación entre esos campos en este caso se escogieron la duración y el tipo de ataque para sacar datos para las matrices y poder mostrar los resultados de los registros de información, esta evaluación de datos van creando un conjunto de datos para ir generando patrones, para tener un procesamiento de información de manera general utilizando estas herramientas de machine learning.

Figura 7.
Procesamiento de datos.

Preprocesamiento de los datos.

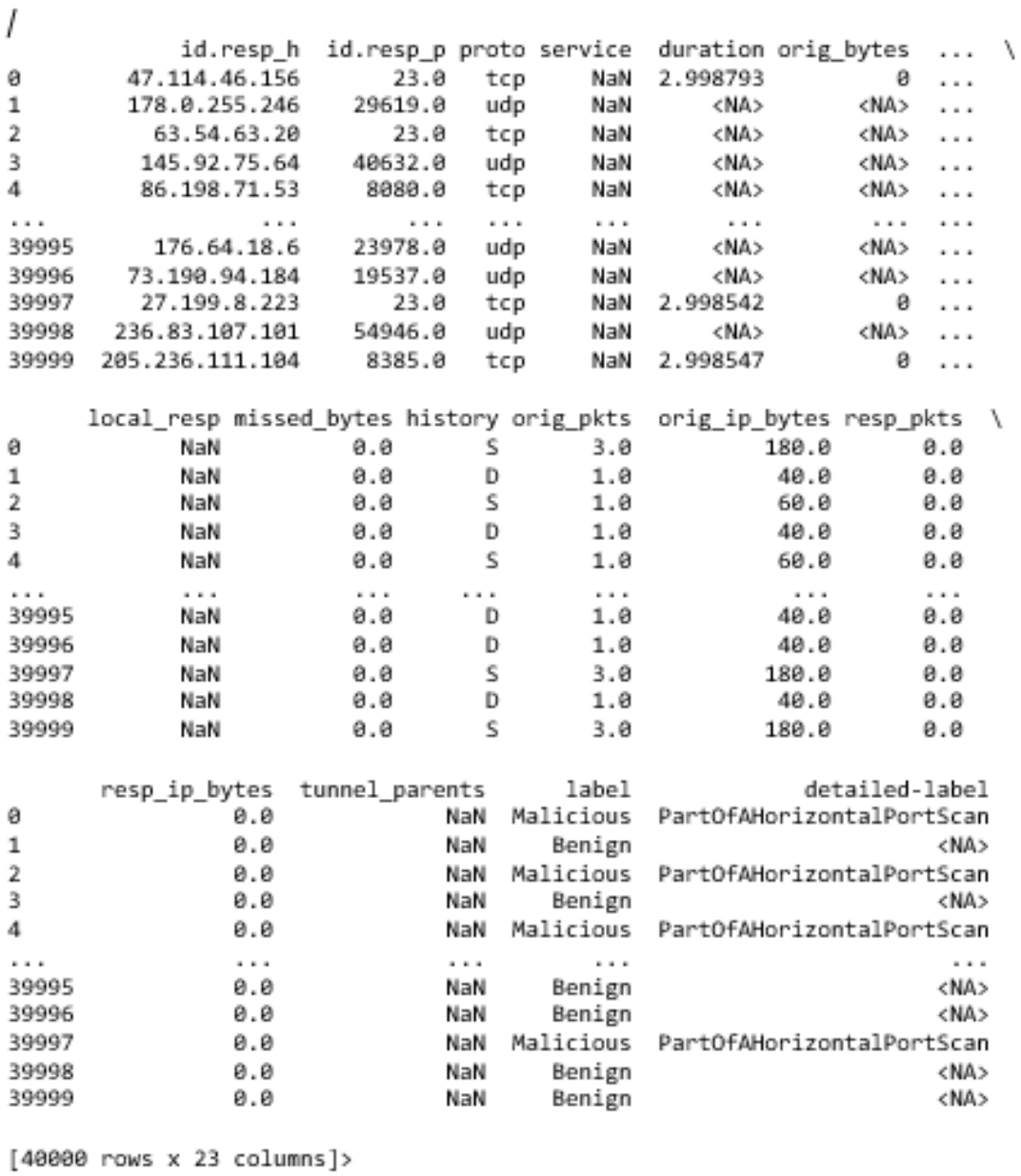
```
print(df.head)

<bound method NDFrame.head of                    ts                 uid
id.orig_h  id.orig_p  \
0      1.525931e+09   CKJGqN4hUpmdgFK3b  192.168.100.103    39447.0
1      1.525903e+09  C3SlOV1zQ6XDih4LCe  192.168.100.103    43763.0
2      1.525932e+09  CH0rP02O7BdjPsnpKj  192.168.100.103    36407.0
3      1.525885e+09  CPI1KZ19rAP0r7niji  192.168.100.103    43763.0
4      1.525901e+09  CddRN62AYK3bKhw6Z7  192.168.100.103    36618.0
...             ...                 ...                ...        ...
39995  1.525886e+09  CHgsnB40ZBCp0IUV05  192.168.100.103    43763.0
39996  1.525901e+09  CUkoyw4FMxqboVMZJg  192.168.100.103    43763.0
39997  1.525893e+09  Chkv0g40xqVqE2GPBa  192.168.100.103    43304.0
39998  1.525919e+09  C8peMs3J4HYI81deR7  192.168.100.103    43763.0
39999  1.525889e+09  CxyZzo2noUr5lun7sk  192.168.100.103    38185.0

             id.resp_h  id.resp_p proto service  duration orig_bytes  ...  \
0        47.114.46.156       23.0   tcp       -  2.998793          0  ...
1        178.0.255.246    29619.0   udp       -         -          -  ...
2          63.54.63.20       23.0   tcp       -         -          -  ...
3         145.92.75.64    40632.0   udp       -         -          -  ...
4         86.198.71.53     8080.0   tcp       -         -          -  ...
...                ...        ...   ...     ...       ...        ...  ...
39995      176.64.18.6    23978.0   udp       -         -          -  ...
39996    73.190.94.184    19537.0   udp       -         -          -  ...
39997     27.199.8.223       23.0   tcp       -  2.998542          0  ...
39998   236.83.107.101    54946.0   udp       -         -          -  ...
39999  205.236.111.104     8385.0   tcp       -  2.998547          0  ...
```

Nota: Elaboración propia tomada de los anexos.

La identificación de malware en el tráfico de red mediante el uso de Support Vector Machines (SVM) y Naive Bayes (NB) es un enfoque común en la ciberseguridad. Aquí se explica brevemente en qué se basa cada uno de estos métodos:

Fundamento: SVM un algoritmo de aprendizaje supervisado, se emplea en la clasificación y regresión, siendo particularmente relevante en el ámbito de la detección de malware, SVM busca encontrar un hiperplano óptimo que pueda separar de manera eficiente las instancias de malware de las instancias no maliciosas en un espacio multidimensional.

Figura 8.
Clasificación de manera eficiente.



```
/
            id.resp_h  id.resp_p proto service  duration orig_bytes  ...  \
0        47.114.46.156       23.0   tcp     NaN  2.998793          0  ...
1        178.0.255.246    29619.0   udp     NaN      <NA>       <NA>  ...
2          63.54.63.20       23.0   tcp     NaN      <NA>       <NA>  ...
3         145.92.75.64    40632.0   udp     NaN      <NA>       <NA>  ...
4         86.198.71.53     8080.0   tcp     NaN      <NA>       <NA>  ...
...                ...        ...   ...     ...       ...        ...  ...
39995      176.64.18.6    23978.0   udp     NaN      <NA>       <NA>  ...
39996    73.190.94.184    19537.0   udp     NaN      <NA>       <NA>  ...
39997     27.199.8.223       23.0   tcp     NaN  2.998542          0  ...
39998   236.83.107.101    54946.0   udp     NaN      <NA>       <NA>  ...
39999  205.236.111.104     8385.0   tcp     NaN  2.998547          0  ...

      local_resp missed_bytes history orig_pkts  orig_ip_bytes resp_pkts  \
0            NaN          0.0       S       3.0          180.0       0.0
1            NaN          0.0       D       1.0           40.0       0.0
2            NaN          0.0       S       1.0           60.0       0.0
3            NaN          0.0       D       1.0           40.0       0.0
4            NaN          0.0       S       1.0           60.0       0.0
...          ...          ...     ...       ...            ...       ...
39995        NaN          0.0       D       1.0           40.0       0.0
39996        NaN          0.0       D       1.0           40.0       0.0
39997        NaN          0.0       S       3.0          180.0       0.0
39998        NaN          0.0       D       1.0           40.0       0.0
39999        NaN          0.0       S       3.0          180.0       0.0

       resp_ip_bytes  tunnel_parents      label             detailed-label
0                0.0             NaN  Malicious  PartOfAHorizontalPortScan
1                0.0             NaN     Benign                       <NA>
2                0.0             NaN  Malicious  PartOfAHorizontalPortScan
3                0.0             NaN     Benign                       <NA>
4                0.0             NaN  Malicious  PartOfAHorizontalPortScan
...              ...             ...        ...                        ...
39995            0.0             NaN     Benign                       <NA>
39996            0.0             NaN     Benign                       <NA>
39997            0.0             NaN  Malicious  PartOfAHorizontalPortScan
39998            0.0             NaN     Benign                       <NA>
39999            0.0             NaN     Benign                       <NA>

[40000 rows x 23 columns]>
```

Nota: Elaboración propia tomada de los anexos.

Ayuda a encontrar un conjunto reducido de patrones comunes entre los dispositivos, el tráfico de red generado y acumulado en los registros a través de los cuales se determinó la ocurrencia de los ataques.

Figura 9.
Búsqueda de patrones comunes.

Manejando los valores nulos

```
null_values = df.isnull().sum()
print(null_values)
```

```
ts                    0
uid                   0
id.orig_h             0
id.orig_p             0
id.resp_h             0
id.resp_p             0
proto                 0
service           39837
duration          31683
orig_bytes        31683
resp_bytes        31683
conn_state            0
local_orig        40000
local_resp        40000
missed_bytes          0
history            1098
orig_pkts             0
orig_ip_bytes         0
resp_pkts             0
resp_ip_bytes         0
tunnel_parents    40000
label                 0
detailed-label    20000
dtype: int64
```

Nota: Elaboración propia tomada de los anexos.

Kernel Trick: SVM utiliza a menudo el "kernel trick" para mapear los elementos a un espacio de atributos de mayor dimensión, facilitando la identificación de un hiperplano que pueda separar las clases de manera más efectiva.

Figura 10.
Mapeo de datos.

Entrenamiento

Entrada:

```
svm_model = SVC(kernel='linear', C=0.001, gamma='scale')
svm_model.fit(X_train_scaled,y_train)
```

Salida:

```
SVC(C=0.001, kernel='linear')
```

Evaluando el modelo

Entrada:

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
/
print(f"Precisión (Accuracy): {accuracy}")
print(f"Reporte de Clasificación:\n{report}")
```

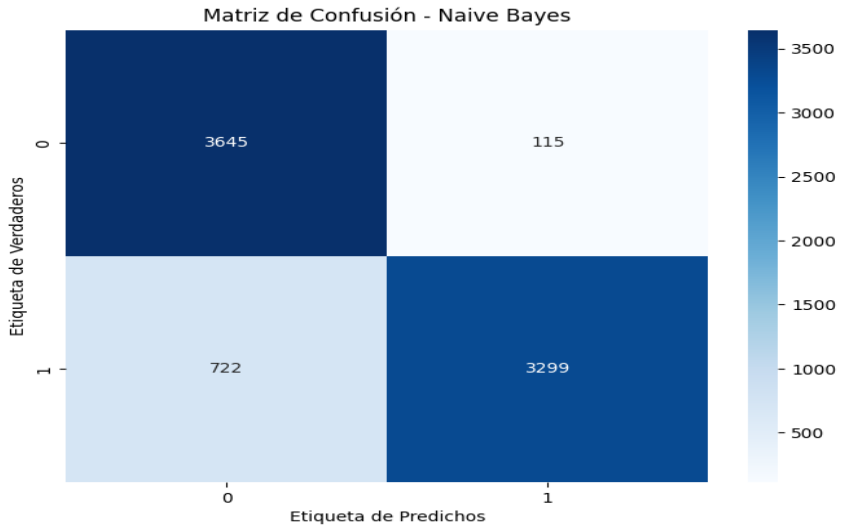
```
Precisión (Accuracy): 0.5167716231846806
Reporte de Clasificación:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      3760
           1       0.52      1.00      0.68      4021

    accuracy                           0.52      7781
   macro avg       0.26      0.50      0.34      7781
weighted avg       0.27      0.52      0.35      7781
```

Nota: Elaboración propia tomada de los anexos.

Por ejemplo se manejan los valores nulos en cuanto al campo de duración y se revisa la correlación.

Se empleó el algoritmo de clasificación probabilística basado en el teorema de Bayes, se lo utilizó para calcular las probabilidades condicionales de pertenencia a una clase dada con los campos seleccionados. En el caso de la detección de malware se buscó en que intervalo de tiempo han ocurrido esas amenazas y se evalúan las probabilidades que ciertas características del tráfico de red indiquen la presencia de malware, dentro de sus características es fácil de implementar y funciona bien incluso con conjuntos de datos de dimensiones considerables, como se puede observar en la figura 11.

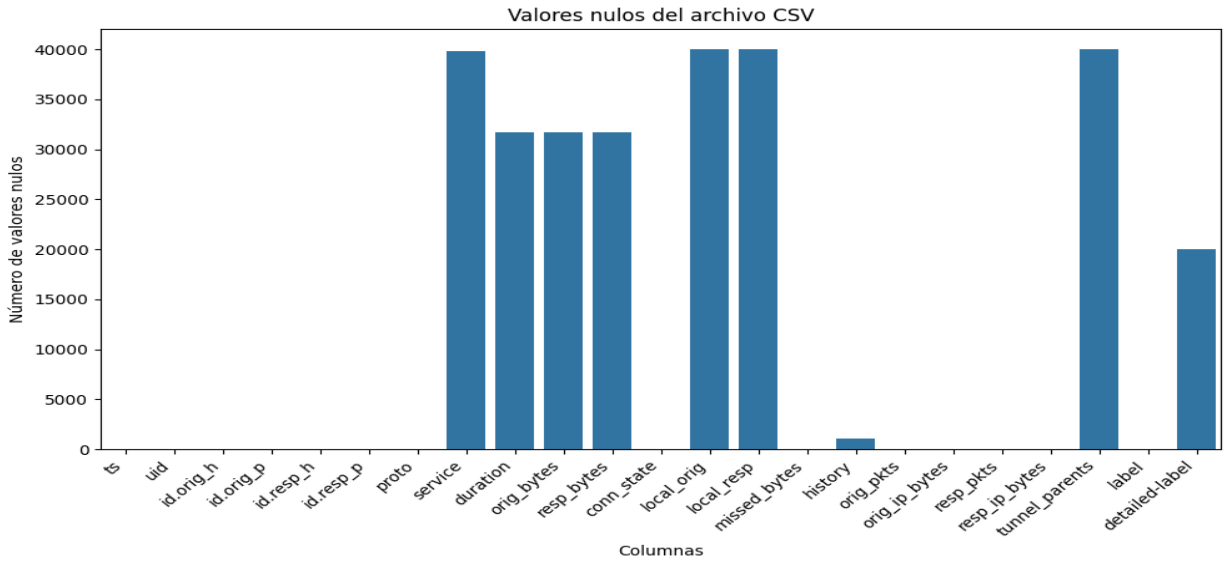
Figura 11.
Resultado de Matriz de confusión Naive Bayes.



Nota: Elaboración propia tomada de los anexos.

Con los resultados derivados de la implementación de este algoritmo se ha empleado herramientas de visualización como el mapa de calor que se pueden revisar en las figuras 13, 14, 16, 17 y el código de barras en la figura 12, para representación visual de los datos, para evidenciar y resaltar los patrones y tendencias de datos obtenidos y de la manera que están relacionadas las diferentes variables para tener un mejor entendimiento y evaluación de la información referente a su funcionamiento y eficacia con respecto a la detección de patrones de amenazas cibernéticas.

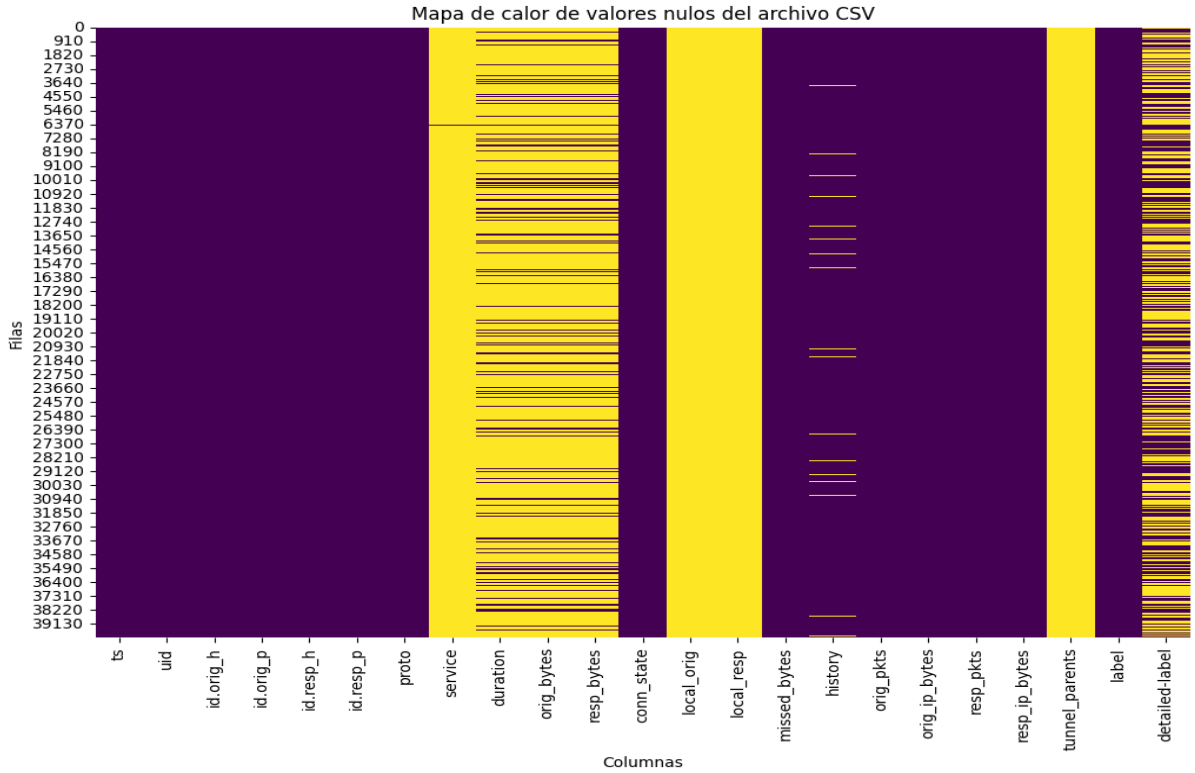
Figura 12.
Resultado en un gráfico de barras de búsqueda de valores nulos.



Nota: Elaboración propia tomada de los anexos.

Figura 13.

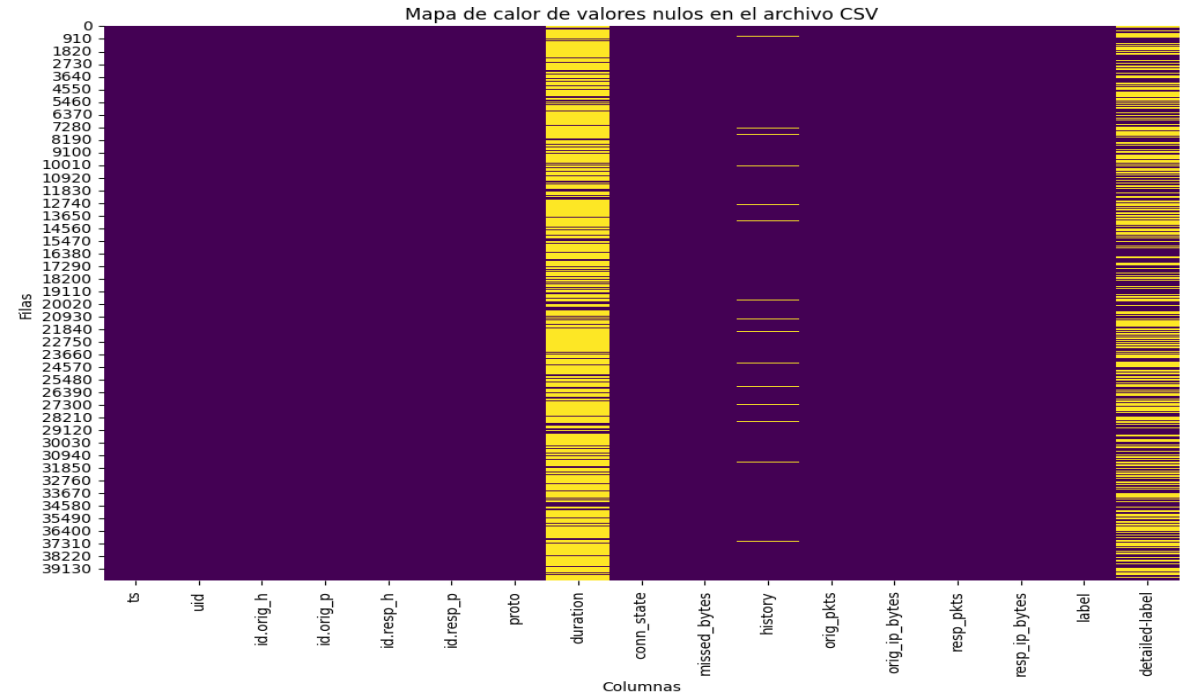
Resultado en un mapa de calor de búsqueda de valores nulos.



Nota: Elaboración propia tomada de los anexos.

Figura 14.

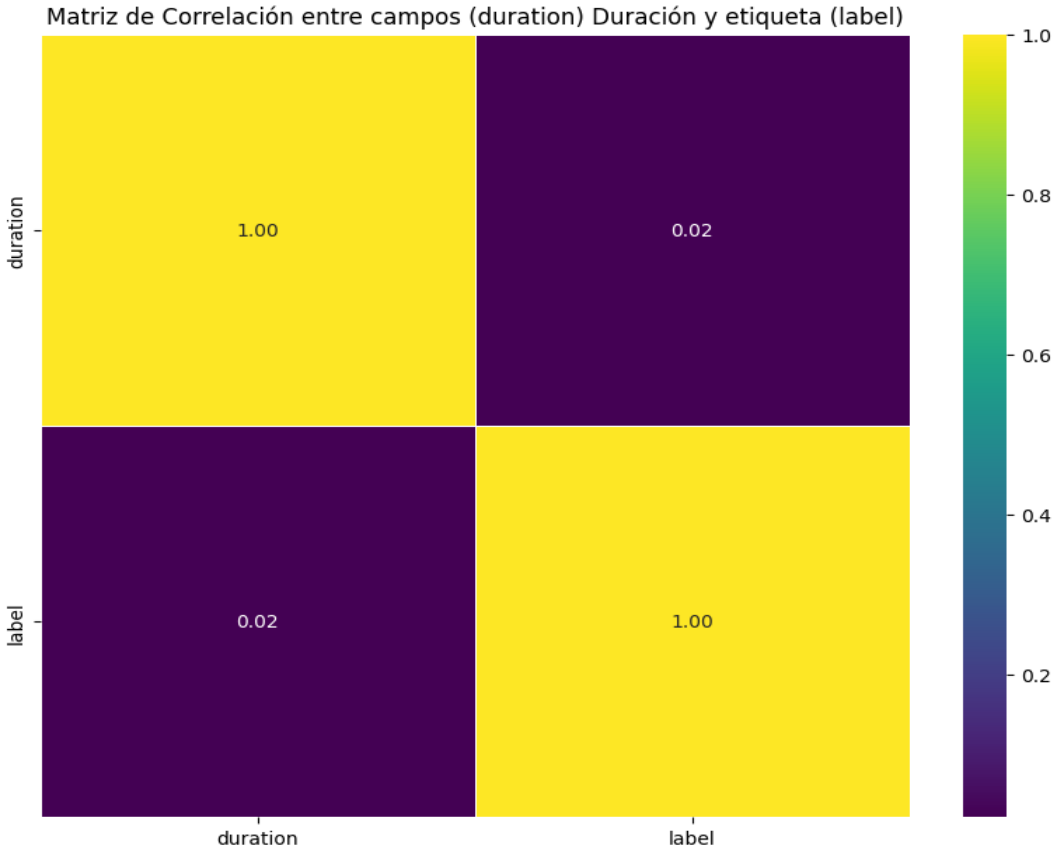
Resultado en un mapa de calor de búsqueda de valores nulos.



Nota: Elaboración propia tomada de los anexos.

Figura 15.

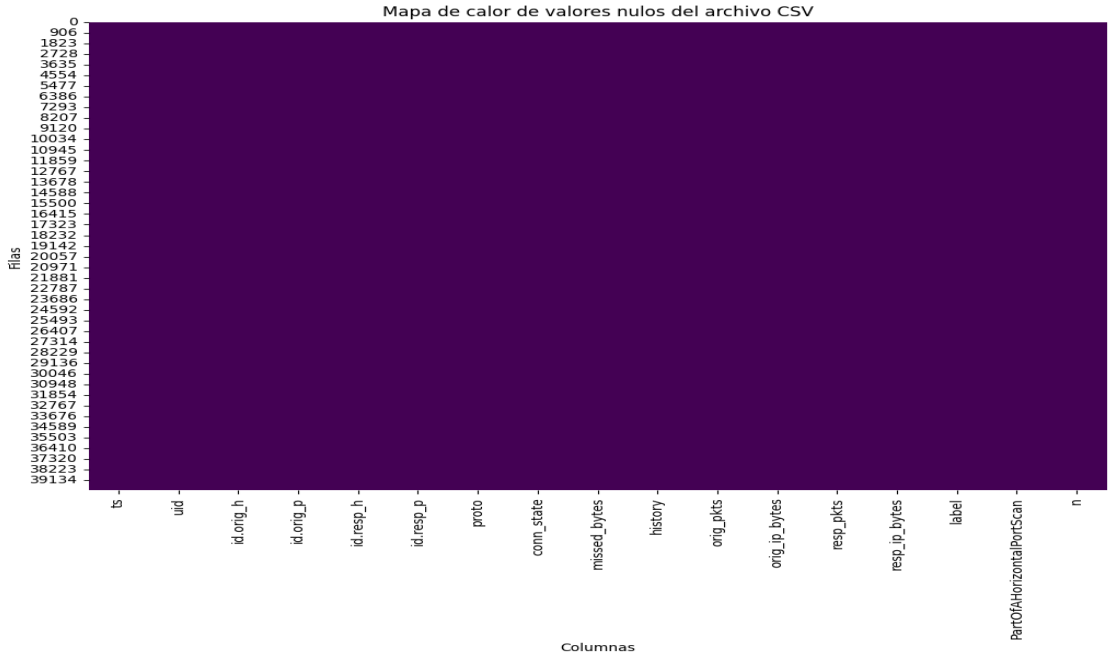
Resultado de Matriz de Correlación entre campos duración y etiqueta.



Nota: Elaboración propia tomada de los anexos.

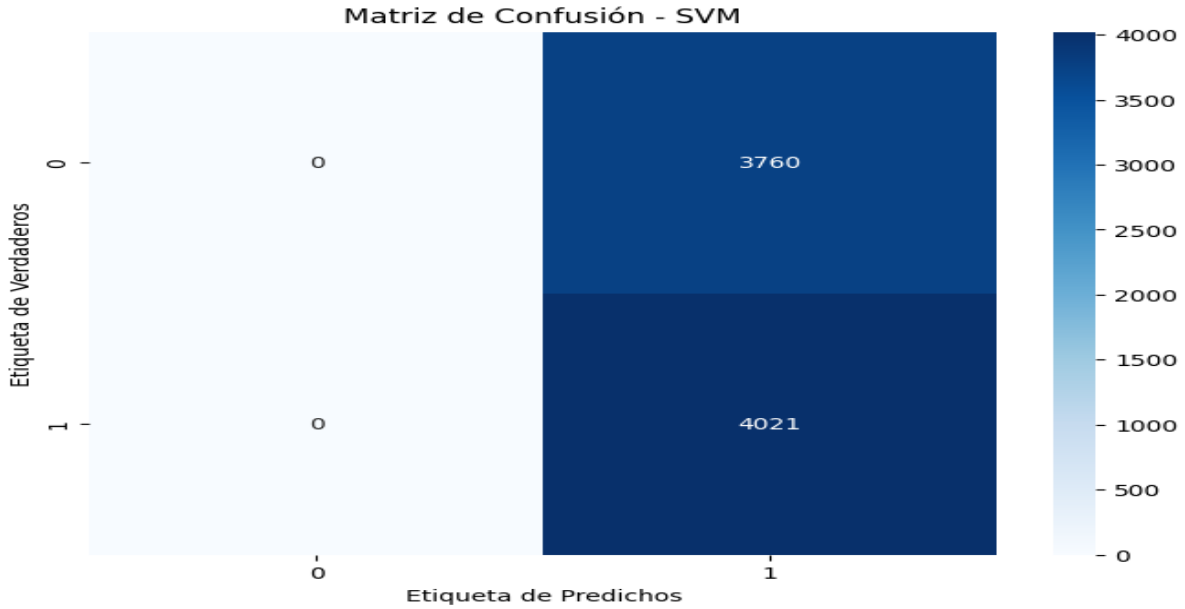
Figura 16.

Resultado de mapa de calor de valores nulos.



Nota: Elaboración propia tomada de los anexos.

Figura 17.
Resultado Matriz de confusión SVM.



Nota: Elaboración propia tomada de los anexos.

2.4. Matriz de articulación de la propuesta

Tabla *4.*
Matriz de articulación.

EJES O PARTES PRINCIPALES	SUSTENTO TEÓRICO	SUSTENTO METODOLÓGICO	ESTRATEGIAS / TÉCNICAS	DESCRIPCIÓN DE RESULTADOS	INSTRUMENTOS APLICADOS
Red neuronal	Neurobiología y modelos matemáticos inspirados en el funcionamiento del cerebro (Pinilla, 2020).	Recopilación y preparación de datos.	Arquitectura de red (número de capas y neuronas). Selección de funciones de activación. Implementación de algoritmos.	Evaluación del rendimiento del modelo en términos de métricas específicas, precisión, sensibilidad, especificidad y curva ROC.	Software y bibliotecas especializadas como TensorFlow o PyTorch.
Aprendizaje profundo	Conceptos de propagación hacia adelante, retro propagación del error y optimización. Redes neuronales profundas o multicapa (Flores, 2020).	Elección y configuración de arquitecturas de red. Técnicas de regulación como dropout.	Transferencia de aprendizaje. Implementación de arquitecturas. Redes neuronales recurrentes (RNN).	Capacidad del modelo para aprender representaciones. Evaluar métricas. Analizar rendimientos.	Bibliotecas y herramientas específicas para el procesamiento de datos. Marcos de trabajo (frameworks) como TensorFlow o PyTorch.

Herramientas de software	Revisión de documentación y material bibliográfico.	Experimentación	Experimentación directa	Identificación de los algoritmos con mejores resultados en los análisis.	Aplicación de herramientas que incorporan algoritmos de aprendizaje e inteligencia artificial.
Librerías	Revisión de documentación y material bibliográfico.	Experimentación y revisión literaria.	Librerías de aprendizaje automático, de aprendizaje profundo, análisis de datos, matrices y tablas, entre otros.	Pruebas con las librerías, para analizar sus resultados.	Librerías: Cython3, Gfortran, MesonPy, Libfreetype, Libpng, Libqhull, Libagg2. Librerias de Python: Scipy, Scikit-learn, Numpy, H5py, Beautifulsoup4, Pandas, Requests, Scrapy, Tensorflow

Nota: Elaboración propia

CONCLUSIONES

De lo analizado durante el trabajo; se ha encontrado que, existen pocas herramientas que incorporan IA y ML para realizar la recopilación, análisis e identificación de patrones y entrega de resultados, se recurrió a la recopilación de información de los logs o registros de los servidores, muestras tomadas por herramientas de monitoreo de red, por tanto:

- El presente trabajo permitió evaluar las herramientas existentes que ya incorporan IA y ML y que en conjunto con los algoritmos sobre todo (RNA) permiten tener una perspectiva más amplia acerca de la realidad actual de la IA en el campo de la ciberseguridad. Con el uso de las librerías de aprendizaje profundo permiten construir y entrenar modelos de redes neuronales, se podrían aplicar en la detección de malware, análisis de patrones de tráfico de red y otros escenarios de seguridad.
- Se determinaron modelos basados en aprendizaje de máquina en los cuales se recurre a diferentes algoritmos para el análisis de información y previo entrenamiento de los mismos, lo que permitió obtener patrones típicos de los ciberataques y determinar la prevalencia y probabilidad de ocurrencia de los mismos.
- En base a las simulaciones realizadas se encontraron un conjunto reducido de patrones comunes entre los dispositivos, el tráfico de red generado y acumulado en los registros a través de los cuales se determinó la ocurrencia de los ataques.
- Con los antecedentes previos, se realizaron los ajustes en los algoritmos y uno de los modelos analizados a razón de poder inferir y estimar el alcance de las amenazas y vulnerabilidades lo que permitió alcanzar un nivel óptimo de mitigación del impacto de los ciberataques.
- Se ha validado por unos expertos que la fusión de elementos de aprendizaje automático e inteligencia artificial aplicados al ámbito de ciberseguridad, son una nueva tendencia innovadora que tienen herramientas con mayor capacidad para ajustarse a nuevas amenazas y ayudar en diversos tipos de aplicaciones, por ejemplo un SVM puede capturar patrones no lineales, mientras que NB puede manejar eficientemente conjuntos de datos grandes con características independientes.

- Debido al continúo avance de la tecnología y en particular al crecimiento de la IA y los algoritmos de ML, los estudios realizados no son concluyentes en su totalidad. En este contexto se realizado una aproximación al uso de modelos en el estado todavía inicial en el que se encuentran actualmente las herramientas, por tal razón esta investigación puede ser un gran aporte y contribución para trabajos de experimentación en el campo de la seguridad informática y la ciberseguridad.

RECOMENDACIONES

Con base en lo realizado hasta ahora; es necesario tener en cuenta lo siguiente:

- Continuar la exploración y experimentando con algoritmos y modelos para mejorar el análisis de los patrones, para evaluar el rendimiento, manteniendo un enfoque ético y transparente en todo momento.
- Se recomienda seguir realizando la escritura o propuesta de modelos y algoritmos que permitan ir determinando el uso para los patrones de comportamiento de los vectores de ataque en ciberseguridad.
- Se requiere la incorporación de más herramientas para análisis y administración de resultados como OpenVAS y otras relacionadas con las bases de datos de vulnerabilidades para en conjunto con los algoritmos y modelos y cerrando los ciclos originales de análisis.
- Será necesario tener más ambientes de pruebas en particular para los nuevos y mejorados mecanismos de ciberataques, a razón de seguir entrenando a los algoritmos y generar nuevos modelos.

BIBLIOGRAFÍA

Agustín Parmisano, S. G. (2020). *https://www.stratosphereips.org/datasets-iot23*.

Aljamal, I., Tekeoglu, A., Bekirog, K., y Sengupta, S. (2019). Hybrid Intrusion Detection System Using Machine Learning Techniques in Cloud Computing Environments. *IEEE Computer Society*. https://doi.org/978-1-7281-0798-1/19/$31.00 ©2019 IEEE

Ayerbe, A. (2020). *La ciberseguridad y su relación con la inteligencia artificial.* Real Instituto Elcano.

Cabanillas, H., y Nizama, J. (2019). *ANÁLISIS DE ALGORITMOS DE ENCRIPTACIÓN DE DATOS DE TEXTO, UNA REVISIÓN DE LA LITERATURA CIENTÍFICA.* Universidad Privada del Norte.

Castellanos, B. S., Cortés Rodríguez, C. U., Espitia Osorio, D. J., y Garzón Bello, Y. T. (2020). Redes neuronales artificiales y estado del arte aplicado en la ciberseguridad. *Revista Matices Tecnológicos Edición 12. UNISANGIL. ISSN 2027-4408*. https://doi.org/UNISANGIL. ISSN 2027-4408

Dueñas, J. (2020). *Aplicación de técnicas de machine learning a la ciberseguridad: Aprendizaje supervisado para la detección de amenazas web mediante clasificación basada en árboles de decisión.* Universidad Oberta de Catalunya, Máster Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones.

Flores, C. (2020). Inteligencia Artificial, Machine Learning, Deep Learning aplicados a la Ciberseguridad. *Revista PGI. Investigación, Ciencia y Tecnología en Informática*(7).

Guevara, G. P., Verdesoto, A. E., y Castro, N. E. (2020). Metodologías de investigación educativa (descriptivas,experimentales, participativas, y de investigación-acción). *Revista Científica Mundo de la Investigación y el Conocimiento*. https://doi.org/2588-073X

http://www.lab.inf.uc3m.es/~a0080630/redes-de-neuronas/perceptron-multicapa.html. (Julio de 2014).

https://www.ibm.com/topics/cybersecurity. (s.f.).

Jiménez, C., y Ramírez, Ó. (2022). *Propuesta de buenas prácticas de ciberseguridad para el uso de chatbots en el sector privado costarricense.* Universidad Cenfotec.

Jiménez-Almeira, G. A., y Enrique López, D. (2023). Ciberseguridad y Seguridad Integral: un análisis reflexivo sobre el avance normativo en Colombia. *Iberian Journal of Information Systems and Technologies*, 16-31.

Luzuriaga, E., Fernando, A., Secaira, R., Marcelo, F., Galarza Sánchez, P. C., y Boné Andrade, M. F. (Enero-Marzo de 2023). La inteligencia artificial aplicada a la optimización de programas informáticos. *Journal of Economic and Social Science Research, 3*(1). https://doi.org/ISSN: 2953-6790

Martín, C. (05 de Septiembre de 2022). *Estándares y normas ISO para mejorar la ciberseguridad*. https://www.globalsuitesolutions.com/: https://www.globalsuitesolutions.com/es/normas-iso-para-mejorar-la-ciberseguridad/

Pérez, M. P. (2022). *Diseño e implementación de la estrategia de concienciación digital de ciberseguridad en el área de TI para la compañía colombiana de comercio.* http://hdl.handle.net/11371/4902.

Pinilla, I. A. (2020). Técnicas de inteligencia artificial usadas en seguridad informática. *Universidad de Buenos Aires*.

Portela, S. (2022). PANORAMA DE LA INTELIGENCIA ARTIFICIAL EN EL DOMINIO DE LA CIBERSEGURIDAD. *RUIDERAe: Revista de Unidades de Información. Número 19 , ISSN 2254-7177*, 35.

Quirumbay, D., Castillo, C., y Coronel, I. (2022). A review of deep learning applied to cybersecurity. *Revista Científica Y Tecnológica UPSE, 9(1). http://dx.doi.org/10.26423/rctu.v9i1.671*, 57-65.

Reyes, L. &. (2020). La investigación documental para la comprensión ontológica del objeto de estudio.

Ruiz Martinez, W. (2021). Análisis de técnicas de Machine Learning aplicadas a la ciberseguridad informática para mejorar la detección de intrusiones y comportamientos anómalos en la Web.

Satpatía, S. (29 de Abril de 2022). *https://betterprogramming.pub/introducing-flowmeter-97e0507862b6*. https://betterprogramming.pub/introducing-flowmeter-97e0507862b6.

Strand, K. (25 de febrero de 2019). *https://blogs.iadb.org/conocimiento-abierto/es/inteligencia-artificial/Inteligencia*. https://blogs.iadb.org/conocimiento-abierto/es/inteligencia-artificial/Inteligencia

Takaesu, I. (2023). *https://github.com/13o-bbr-bbq/machine_learning_security*. https://twitter.com/bbr_bbq

Tenelema, E. N., Méndez, P. M., Villa, H. M., y Caiza, D. G. (2020). Modelo de seguridad para mitigación de. *Espacios, Vol. 41 (Nº 17)*(ISSN 0798 1015).

Zhimin, Z. H. (2021). Artificial intelligence in cyber security: research advances, challenges, and opportunities. *Artificial Intelligence Review*.

Zhimin, Z., Huansheng, N., Feifei, S., Fadi, F., Yang, X., Jiabo, X., . . . Kim-Kwang, R. C. (2021). Artificial intelligence in cyber security: research advances,challenges, and opportunities. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-021-09976-0

ANEXOS

ANEXO 1

Evaluación de la propuesta con datos:

Detección de Malware en el tráfico de Red.

Wendy Viviana Obregón Martínez

Base o fuente del trabajo:

- https://www.kaggle.com/code/chandujayathilake/malware-detection-in-network-traffic-svm-and-nb/notebook

Los datos han sido reinterpretados y con base en otros parámetros se realizaron los análisis de las muestras.

- Malware en una red de dispositivos IoT

Inicializando el conjunto de datos (dataset)

Importando librerías relevantes

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

Cargando los datos del archivo.

- El archivo utilizado ha sido adaptado y contiene el resumen de los registros (logs) de un servidor que recibe la información de dispositivos IoT.

```
file_path = '/content/CTU-IoT-Malware-Capture-1-1conn.log.labeled.csv'
original_df = pd.read_csv(file_path, delimiter='|') #delimitador | para separ
ar las columnas.
```

Generación de una muestra aleatoria de 100000 datos

```
random_sample_df = original_df.sample(n=100000, random_state=42)

# Creación de una copia de la muestra del DataFrame aleatorio para evitar mod
ificar los datos originales
df = random_sample_df.copy()

file_path = '/content/CTU-IoT-Malware-Capture-1-1conn.log.labeled.csv'

original_df = pd.read_csv(file_path, delimiter='|')
df = original_df.copy()
```

Creación del nuevo conjunto de datos.

```
df_benign = df[df['label'] == 'Benign']
df_malicious = df[df['label'] == 'Malicious']
#
# # Sample 50% of data from each class
frac_benign = df_benign.sample(n=20000, random_state=42)
frac_malicious = df_malicious.sample(n=20000, random_state=42)
#
# # Concatenate the two DataFrames
df_balanced = pd.concat([frac_benign, frac_malicious])
#
# # Shuffle the rows of the resulting DataFrame
df_balanced = df_balanced.sample(frac=1, random_state=42).reset_index(drop=Tr
ue)
#
# # Print the counts of each class in the balanced DataFrame
df = df_balanced
print(df['label'].value_counts())
#
#
```

```
Malicious    20000
Benign       20000
Name: label, dtype: int64
```

Preprocesamiento de los datos.

```
print(df.head)
```

```
<bound method NDFrame.head of                  ts                 uid
id.orig_h  id.orig_p  \
0      1.525931e+09   CKJGqN4hUpmdgFK3b  192.168.100.103    39447.0
1      1.525903e+09  C3SlOV1zQ6XDih4LCe  192.168.100.103    43763.0
2      1.525932e+09  CH0rP02O7BdjPsnpKj  192.168.100.103    36407.0
3      1.525885e+09  CPI1KZ19rAP0r7niji  192.168.100.103    43763.0
4      1.525901e+09  CddRN62AYK3bKhw6Z7  192.168.100.103    36618.0
...             ...                 ...                ...        ...
39995  1.525886e+09  CHgsnB40ZBCp0IUV05  192.168.100.103    43763.0
39996  1.525901e+09  CUkoyw4FMxqboVMZJg  192.168.100.103    43763.0
39997  1.525893e+09  Chkv0g40xqVqE2GPBa  192.168.100.103    43304.0
39998  1.525919e+09  C8peMs3J4HYI81deR7  192.168.100.103    43763.0
39999  1.525889e+09  CxyZzo2noUr5lun7sk  192.168.100.103    38185.0

             id.resp_h  id.resp_p proto service  duration orig_bytes  ...  \
0        47.114.46.156       23.0   tcp       -  2.998793          0  ...
1        178.0.255.246    29619.0   udp       -         -          -  ...
2          63.54.63.20       23.0   tcp       -         -          -  ...
3         145.92.75.64    40632.0   udp       -         -          -  ...
4         86.198.71.53     8080.0   tcp       -         -          -  ...
...                ...        ...   ...     ...       ...        ...  ...
39995      176.64.18.6    23978.0   udp       -         -          -  ...
39996    73.190.94.184    19537.0   udp       -         -          -  ...
39997     27.199.8.223       23.0   tcp       -  2.998542          0  ...
39998   236.83.107.101    54946.0   udp       -         -          -  ...
39999  205.236.111.104     8385.0   tcp       -  2.998547          0  ...

      local_resp missed_bytes history orig_pkts  orig_ip_bytes resp_pkts  \
0              -          0.0       S       3.0          180.0       0.0
```

```
1              -          0.0       D       1.0           40.0       0.0
2              -          0.0       S       1.0           60.0       0.0
3              -          0.0       D       1.0           40.0       0.0
4              -          0.0       S       1.0           60.0       0.0
...          ...          ...     ...       ...            ...       ...
39995          -          0.0       D       1.0           40.0       0.0
39996          -          0.0       D       1.0           40.0       0.0
39997          -          0.0       S       3.0          180.0       0.0
39998          -          0.0       D       1.0           40.0       0.0
39999          -          0.0       S       3.0          180.0       0.0

       resp_ip_bytes  tunnel_parents      label             detailed-label
0                0.0               -  Malicious  PartOfAHorizontalPortScan
1                0.0               -     Benign                          -
2                0.0               -  Malicious  PartOfAHorizontalPortScan
3                0.0               -     Benign                          -
4                0.0               -  Malicious  PartOfAHorizontalPortScan
...              ...             ...        ...                        ...
39995            0.0               -     Benign                          -
39996            0.0               -     Benign                          -
39997            0.0               -  Malicious  PartOfAHorizontalPortScan
39998            0.0               -     Benign                          -
39999            0.0               -     Benign                          -

[40000 rows x 23 columns]>

# Reemplazo del '-' con pandas a valores NA para un mejor manejo de los datos
no existentes o no definidos.
df.replace('-', pd.NA, inplace=True)

print(df.head)

<bound method NDFrame.head of                  ts                 uid
id.orig_h  id.orig_p  \
0      1.525931e+09   CKJGqN4hUpmdgFK3b  192.168.100.103    39447.0
1      1.525903e+09  C3SlOV1zQ6XDih4LCe  192.168.100.103    43763.0
2      1.525932e+09  CH0rP02O7BdjPsnpKj  192.168.100.103    36407.0
3      1.525885e+09  CPI1KZ19rAP0r7niji  192.168.100.103    43763.0
4      1.525901e+09  CddRN62AYK3bKhw6Z7  192.168.100.103    36618.0
...             ...                 ...              ...        ...
39995  1.525886e+09  CHgsnB40ZBCp0IUV05  192.168.100.103    43763.0
39996  1.525901e+09  CUkoyw4FMxqboVMZJg  192.168.100.103    43763.0
39997  1.525893e+09  Chkv0g40xqVqE2GPBa  192.168.100.103    43304.0
39998  1.525919e+09  C8peMs3J4HYI81deR7  192.168.100.103    43763.0
39999  1.525889e+09  CxyZzo2noUr5lun7sk  192.168.100.103    38185.0
```

```
            id.resp_h  id.resp_p proto service  duration orig_bytes  ...  \
0        47.114.46.156       23.0   tcp     NaN  2.998793          0  ...
1        178.0.255.246    29619.0   udp     NaN      <NA>       <NA>  ...
2          63.54.63.20       23.0   tcp     NaN      <NA>       <NA>  ...
3         145.92.75.64    40632.0   udp     NaN      <NA>       <NA>  ...
4         86.198.71.53     8080.0   tcp     NaN      <NA>       <NA>  ...
...                ...        ...   ...     ...       ...        ...  ...
39995      176.64.18.6    23978.0   udp     NaN      <NA>       <NA>  ...
39996    73.190.94.184    19537.0   udp     NaN      <NA>       <NA>  ...
39997     27.199.8.223       23.0   tcp     NaN  2.998542          0  ...
39998   236.83.107.101    54946.0   udp     NaN      <NA>       <NA>  ...
39999  205.236.111.104     8385.0   tcp     NaN  2.998547          0  ...

      local_resp missed_bytes history orig_pkts  orig_ip_bytes resp_pkts  \
0            NaN          0.0       S       3.0          180.0       0.0
1            NaN          0.0       D       1.0           40.0       0.0
2            NaN          0.0       S       1.0           60.0       0.0
3            NaN          0.0       D       1.0           40.0       0.0
4            NaN          0.0       S       1.0           60.0       0.0
...          ...          ...     ...       ...            ...       ...
39995        NaN          0.0       D       1.0           40.0       0.0
39996        NaN          0.0       D       1.0           40.0       0.0
39997        NaN          0.0       S       3.0          180.0       0.0
39998        NaN          0.0       D       1.0           40.0       0.0
39999        NaN          0.0       S       3.0          180.0       0.0

       resp_ip_bytes  tunnel_parents      label             detailed-label
0                0.0             NaN  Malicious  PartOfAHorizontalPortScan
1                0.0             NaN     Benign                       <NA>
2                0.0             NaN  Malicious  PartOfAHorizontalPortScan
3                0.0             NaN     Benign                       <NA>
4                0.0             NaN  Malicious  PartOfAHorizontalPortScan
...              ...             ...        ...                        ...
39995            0.0             NaN     Benign                       <NA>
39996            0.0             NaN     Benign                       <NA>
39997            0.0             NaN  Malicious  PartOfAHorizontalPortScan
39998            0.0             NaN     Benign                       <NA>
39999            0.0             NaN     Benign                       <NA>

[40000 rows x 23 columns]>
```

Manejando los valores nulos

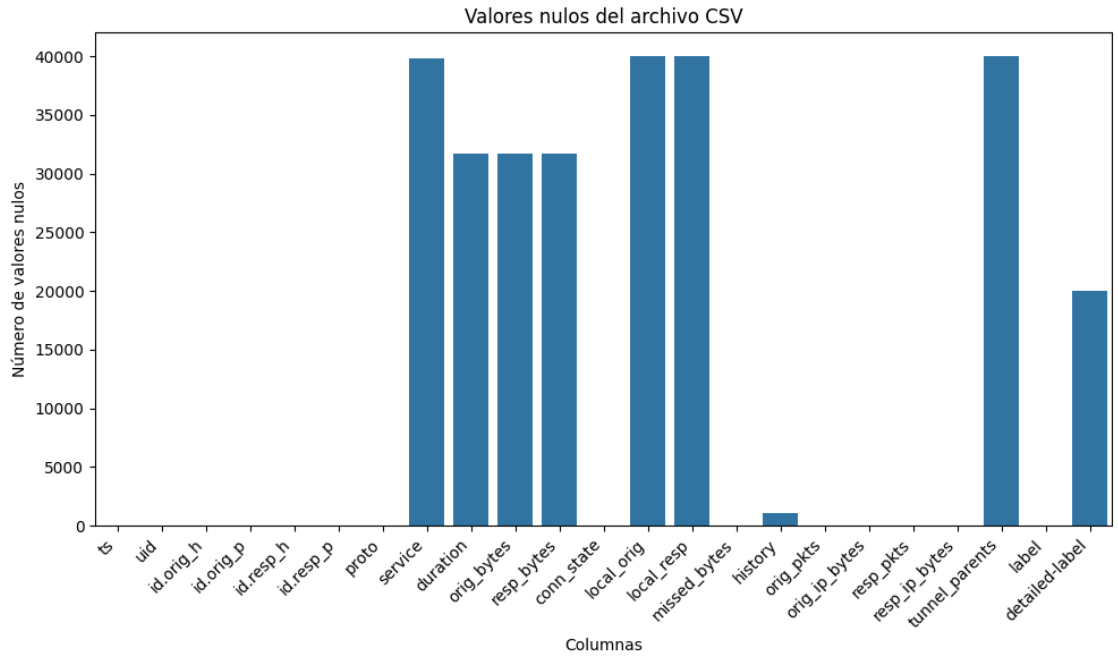
```
null_values = df.isnull().sum()
print(null_values)
```

```
ts                    0
uid                   0
id.orig_h             0
id.orig_p             0
id.resp_h             0
id.resp_p             0
proto                 0
service           39837
duration          31683
orig_bytes        31683
resp_bytes        31683
conn_state            0
```

```
local_orig        40000
local_resp        40000
missed_bytes          0
history            1098
orig_pkts             0
orig_ip_bytes         0
resp_pkts             0
resp_ip_bytes         0
tunnel_parents    40000
label                 0
detailed-label    20000
dtype: int64
```

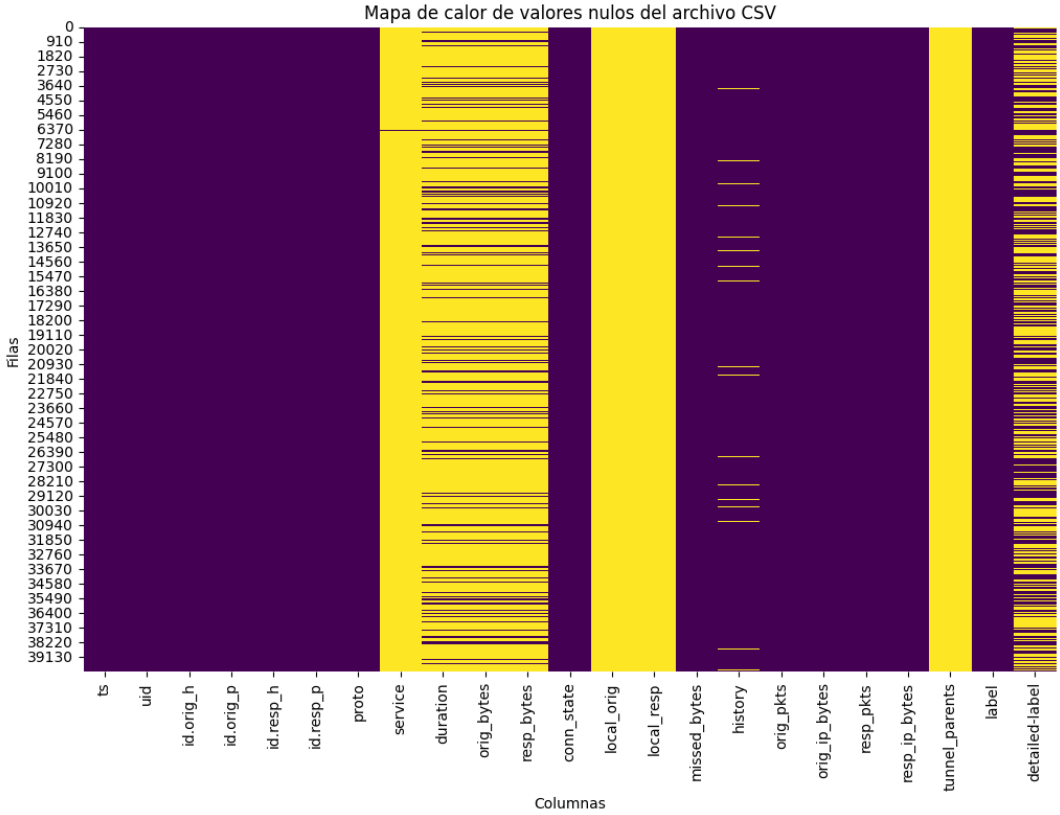
Gráfico de barras

```
# Plotting null values using seaborn
plt.figure(figsize=(10, 6))
sns.barplot(x=null_values.index, y=null_values)
plt.xticks(rotation=45, ha='right')
plt.xlabel('Columnas')
plt.ylabel('Número de valores nulos')
plt.title('Valores nulos del archivo CSV')
plt.tight_layout()
plt.show()
```



Mapa de calor

```
# Generación de un mapa de calor de valores nulos
plt.figure(figsize=(12, 8))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False)
plt.xlabel('Columnas')
plt.ylabel('Filas')
plt.title('Mapa de calor de valores nulos del archivo CSV')
plt.show()
```



Calculando los valores nulos como porcentaje

```
null_values = df.isnull().sum()

null_percentage = (null_values / len(df)) * 100

columns_with_null = null_percentage[null_percentage > 0]

print("Columnas con valores nulos (Porcentaje):")
print(columns_with_null)
```

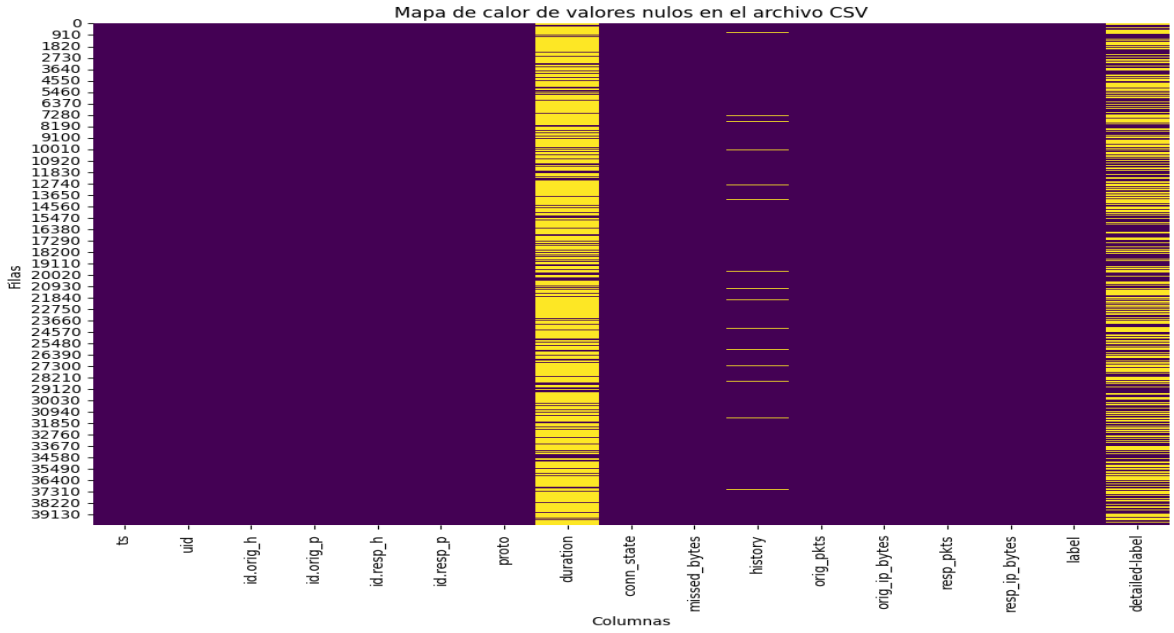
```
Columnas con valores nulos (Porcentaje):
service            99.5925
duration           79.2075
orig_bytes         79.2075
resp_bytes         79.2075
local_orig        100.0000
local_resp        100.0000
history             2.7450
tunnel_parents    100.0000
detailed-label     50.0000
dtype: float64
```

Eliminando columnas con valores nulos que no pueden ser reemplazadas

```
# Eliminación de columnas con valores nulos
df.drop(['service','orig_bytes','resp_bytes','local_orig','local_resp','tunne
l_parents'],axis=1,inplace=True)
```

Después de eliminar las columnas.

```
#Generación de un mapa de calor de valores nulos
plt.figure(figsize=(12, 8))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False)
plt.xlabel('Columnas')
plt.ylabel('Filas')
plt.title('Mapa de calor de valores nulos en el archivo CSV')
plt.show()
```



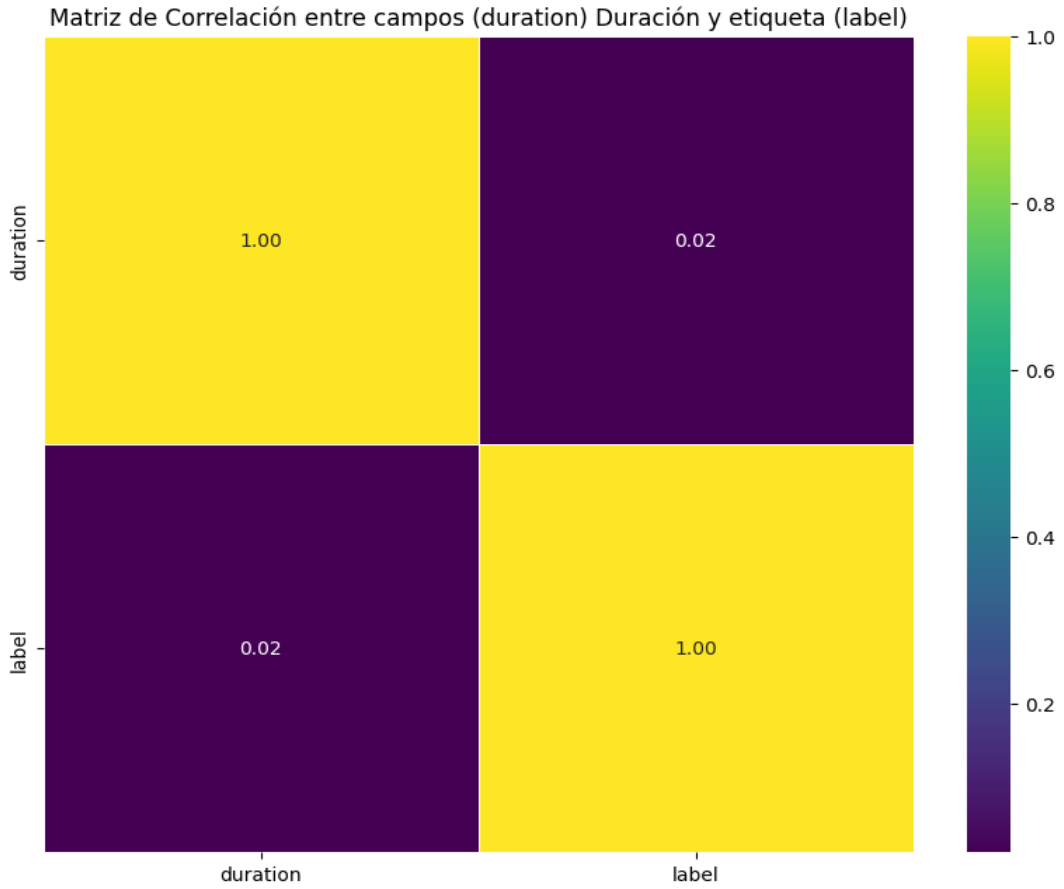
Manejando los valores nulos en cuanto al campo de duración:

```
label_encoder = preprocessing.LabelEncoder()
#label encode label
df['label']= label_encoder.fit_transform(df['label'])
df.head()
df['duration'] = pd.to_numeric(df['duration'])
```

Revisando la correlación.

```
selected_columns = ['duration', 'label']

# Create a DataFrame with only the selected columns
selected_df = df[selected_columns]

# Calculate the correlation matrix
correlation_matrix = selected_df.corr()

plt.figure(figsize=(10, 8))

# Plot a heatmap of the correlation matrix
sns.heatmap(correlation_matrix, annot=True, cmap='viridis', fmt=".2f", linewi
dths=0.5)

plt.title('Matriz de Correlación entre campos (duration) Duración y etiqueta
(label)')
```

```
plt.show()
```



Eliminación de la columna (duration)

```
df.drop(['duration'],axis=1,inplace=True)
```

Analizando el historial

```
df['history'].value_counts()
```

```
S             21211
D             16470
Dd              412
Sr              263
ShAdDaFf        134
ShAdDafF        122
ShADadfF         82
ShAdDfFr         40
ShADdfFa         36
ShAdDaFr         36
ShADafr          33
ShADafF          24
R                 8
ShAdDatFr         5
ShADadfR          4
ShAdtDaFrR        3
ShAfFa            3
ShAF              2
```

```
ShAdDatFrR        2
ShAdDaTFf         2
ShADFr            1
ShAr              1
ShAdDafFr         1
ShA               1
^r                1
ShAdDaTF          1
ShADadRf          1
ShAdDaFrR         1
ShAdtDaFr         1
ShAdaFr           1
Name: history, dtype: int64

#Eliminando los valores nulos
df.dropna(subset=['history'], inplace=True)

label_encoder = preprocessing.LabelEncoder()
df['history']= label_encoder.fit_transform(df['history'])
df['history'].unique()

array([ 3,  0, 11, 22,  1, 28, 13, 18,  9,  7, 14, 20, 10, 29, 17,  2, 25,
       21, 24, 26, 15, 12,  6,  8, 16,  5,  4, 19, 27, 23])
```

Análisis de la columna (detailed-labels)

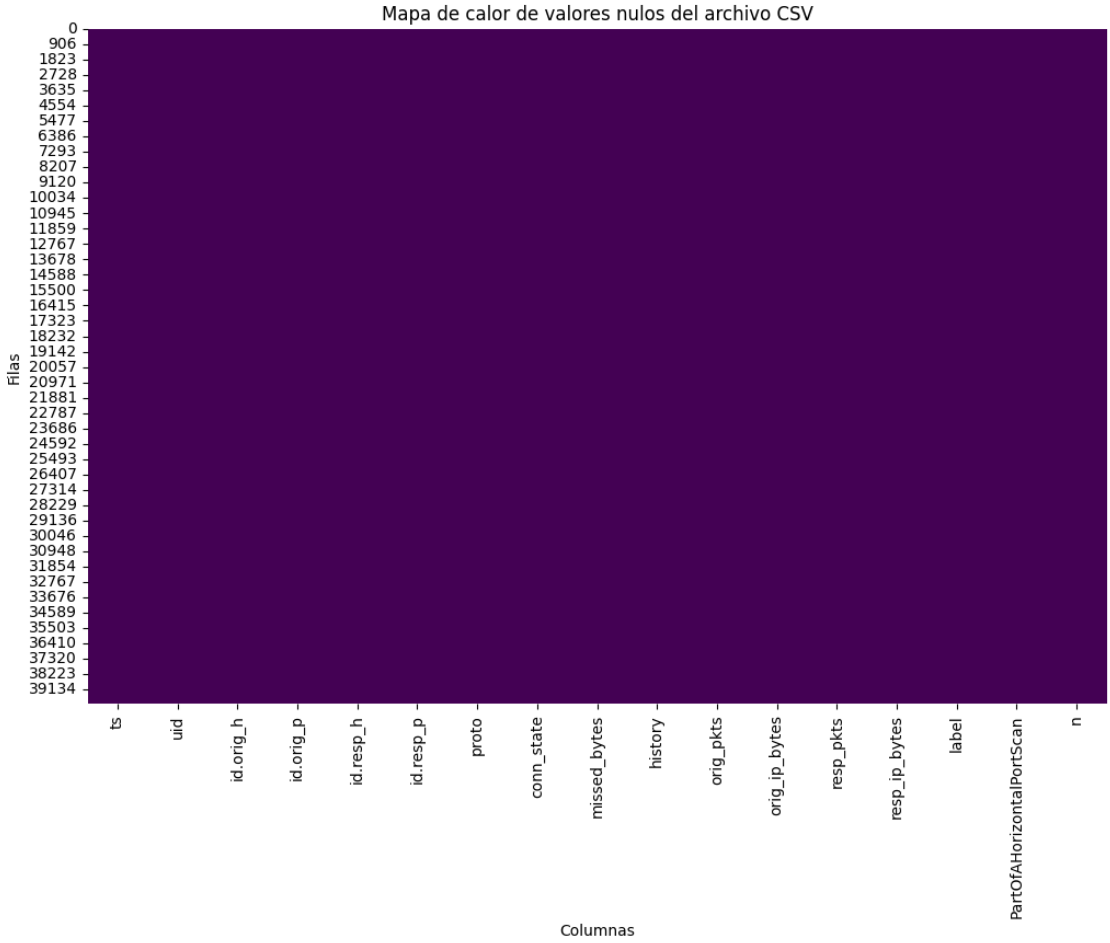
```
df['detailed-label'].value_counts()

PartOfAHorizontalPortScan    20000
Name: detailed-label, dtype: int64

df.drop(df[df['detailed-label'] == 'C&C'].index, inplace = True)
df['detailed-label'].value_counts()

PartOfAHorizontalPortScan    20000
Name: detailed-label, dtype: int64

df['detailed-label'].fillna('n', inplace=True)

#onehot encode
onehot = pd.get_dummies(df['detailed-label'])
df = df.join(onehot)
df.head()
df.drop(['detailed-label'],axis=1,inplace=True)
df.head()
```

ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	conn_state	missed_bytes	history	orig_pkts	orig_ip_bytes	resp_pkts	resp_ip_bytes	label	PartOfAHorizontalPortScan

0	1.525931e+09	CKJGqN4hUpmdgFK3b	192.168.100.103	39447.0	47.1144.656	23.0	tcp	S0	0.0	3	3.0	180.0	0.0	0.0	1	1	0
1	1.525905e+09	C3SlOV1zQ6XDih4LCe	192.168.100.103	43763.0	178.0.255.246	2961	udp	S0	0.0	0	1.0	40.0	0.0	0.0	0	0	1
2	1.525932e+09	CH0rP02O7BdjPsnpKj	192.168.100.103	36407.0	63.54.63.20	23.0	tcp	S0	0.0	3	1.0	60.0	0.0	0.0	1	1	0
3	1.525885e+09	CPI1KZ19rAP0r7niji	192.168.100.103	43763.0	145.92.754.64	4063	udp	S0	0.0	0	1.0	40.0	0.0	0.0	0	0	1
4	1.525901e+09	CddRN62AYK3bKhw6Z7	192.168.100.103	36618.0	86.19187.153	808.0	tcp	S0	0.0	3	1.0	60.0	0.0	0.0	1	1	0

Revisión de valores nulos

```
plt.figure(figsize=(12, 8))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False)
plt.xlabel('Columnas')
plt.ylabel('Filas')
plt.title('Mapa de calor de valores nulos del archivo CSV')
plt.show()
```



Conversión de la marca de tiempo (timestamp) a un tipo numérico

```
#Conversión de timestamp a numeric
df['ts'] = pd.to_numeric(df['ts'])
df['ts'].mean()
```

```
1525908461.0238504
```

Columna Uid

```
# types of timestamp
df['uid'].value_counts()
```

```
CKJGqN4hUpmdgFK3b     1
CDM1eurm1ZcPqqJs1     1
CqGf2v9XTP7nMCyF      1
CAWIVYTXlwTXx86R3     1
CKVNqS19FG4uf3yqBe    1
                     ..
CH7efDrrF7AFIIHW9     1
CzmwR84N8rLf49a0ke    1
Ccihq12ngMCNVpSsu5    1
CzoEQe3W1lJcRjR9D4    1
CxyZzo2noUr5lun7sk    1
Name: uid, Length: 38902, dtype: int64
```

```
df['uid']= label_encoder.fit_transform(df['uid'])
# types of timestamp
df['uid'].value_counts()
```

```
12955    1
8456     1
32895    1
6599     1
13074    1
        ..
10880    1
38775    1
24530    1
38794    1
37685    1
Name: uid, Length: 38902, dtype: int64
```

Análisis del campo ID host origen - Id.orig_h

```
df['id.orig_h'].value_counts()
```

```
192.168.100.103    38894
68.240.229.148         1
173.153.39.7           1
70.6.52.195            1
174.154.68.156         1
173.115.37.65          1
70.4.229.255           1
68.26.69.87            1
173.126.78.110         1
Name: id.orig_h, dtype: int64
```

Entrada:

```
df['id.orig_h']= label_encoder.fit_transform(df['id.orig_h'])
# types of timestamp
df['id.orig_h'].value_counts()
```

Salida:

```
4    38894
5        1
2        1
8        1
3        1
0        1
7        1
6        1
1        1
Name: id.orig_h, dtype: int64
```

Análisis del campo ID respuesta host - id.resp_h

```
df['id.resp_h'].value_counts()
```

```
147.231.100.5      259
37.187.104.44       66
213.239.154.12      63
89.221.214.130      50
61.101.126.35       41
```

```
                ...
126.34.122.168       1
140.181.206.185      1
205.86.217.151       1
126.228.15.149       1
205.236.111.104      1
Name: id.resp_h, Length: 33228, dtype: int64
```

```
df['id.resp_h']= label_encoder.fit_transform(df['id.resp_h'])
df['id.resp_h'].value_counts()
```

```
7457     259
23494     66
18102     63
31529     50
27251     41
         ...
4286       1
6378       1
16895      1
4259       1
16865      1
Name: id.resp_h, Length: 33228, dtype: int64
```

Análisis del campo puerto de respuesta - id.resp_p

Entrada:

```
df['id.resp_p'].value_counts()
```

Salida:

```
23.0       10122
8080.0      4917
2323.0      3328
9527.0      1633
123.0        438
           ...
10729.0        1
1822.0         1
50427.0        1
21646.0        1
8385.0         1
Name: id.resp_p, Length: 15598, dtype: int64
```

Entrada:

```
df['id.resp_p']= label_encoder.fit_transform(df['id.resp_p'])
df['id.resp_p'].value_counts()
```

Salida:

```
4        10122
1964      4917
567       3328
2311      1633
19         438
          ...
2634         1
440          1
```

```
11929        1
5219         1
2042         1
Name: id.resp_p, Length: 15598, dtype: int64
```

Análisis del campo puerto de origen - id.orig_p

Entrada:

```
df['id.orig_p'].value_counts()
```

Salida:

```
43763.0    16444
123.0        438
45057.0       10
40004.0        7
59756.0        7
           ...
51873.0        1
57868.0        1
44000.0        1
48804.0        1
38185.0        1
Name: id.orig_p, Length: 12966, dtype: int64
```

Entrada:

```
df['id.orig_p']= label_encoder.fit_transform(df['id.orig_p'])
df['id.orig_p'].value_counts()
```

Salida:

```
5051     16444
1          438
5622        10
3371         7
12351        7
         ...
8783         1
11494        1
5166         1
7324         1
2516         1
Name: id.orig_p, Length: 12966, dtype: int64

tcp    22020
udp    16882
Name: proto, dtype: int64
```

Ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	conn_state	missed_bytes	history	orig_pkts	orig_ip_bytes	resp_pkts	resp_ip_bytes	label	PartOfAHorizontalPortScan	tunnel_parents	proto

0	1.525931e+09	129	4	3124	5092	4	S0	0.0	3	3.0	180.0	0.0	0.0	1	1	0	1	0
1	1.525903e+09	2187	4	5051	12363	705	S0	0.0	0	1.0	40.0	0.0	0.0	0	0	1	0	1
2	1.525932e+09	108	4	1707	2764	7	S0	0.0	3	1.0	60.0	0.0	0.0	1	1	0	1	0
3	1.525885e+09	1612	4	5051	72033	965	S0	0.0	0	1.0	40.0	0.0	0.0	0	0	1	0	1
4	1.525901e+09	2510	4	1807	31056	194	S0	0.0	3	1.0	60.0	0.0	0.0	1	1	0	1	0

Análisis del campo conn_state

Entrada:

```
df['conn_state'].value_counts()
```

Salida:

```
S0         37681
SF           856
REJ          263
RSTR          84
RSTOS0         8
RSTO           5
S2             3
RSTRH          1
S1             1
Name: conn_state, dtype: int64
```

Entrada:

```
#label encode conn_state
df['conn_state']= label_encoder.fit_transform(df['conn_state'])
df['conn_state'].value_counts()
```

Salida:

```
5    37681
8      856
0      263
3       84
2        8
1        5
7        3
4        1
6        1
Name: conn_state, dtype: int64
```

Análisis del campo orig_pkts

Entrada:

```
df['orig_pkts'].value_counts()
```

Salida:

```
1.0     31319
3.0      7034
5.0       163
2.0        48
15.0       37
28.0       30
12.0       29
4.0        20
45.0       19
27.0       18
25.0       18
21.0       15
26.0       14
44.0       11
9.0        11
11.0       10
20.0        8
6.0         6
39.0        6
7.0         6
38.0        5
37.0        5
18.0        5
10.0        5
22.0        5
19.0        5
29.0        4
43.0        4
46.0        4
34.0        4
24.0        3
35.0        3
```

```
14.0        3
31.0        3
16.0        3
23.0        3
50.0        2
30.0        2
13.0        2
41.0        2
36.0        2
17.0        1
48.0        1
0.0         1
40.0        1
52.0        1
42.0        1
Name: orig_pkts, dtype: int64
```

resp_pkts

Entrada:

```
df['resp_pkts'] = pd.to_numeric(df['resp_pkts'])
```

```
1    20000
0    18902
Name: label, dtype: int64
```

	Ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	conn_state	history	orig_pkts	orig_ip_bytes	resp_pkts	resp_ip_bytes	label	PartOfAHorizontalPortScan	n	tcp	udp
0	1.525931e+09	129555	4	3124	2509 2	4	5	3	3.0	180.0	0.0	0.0	1	1	0	1	0
1	1.525903e+09	218 87	4	5051	12363	7075	5	0	1.0	40.0	0.0	0.0	0	0	1	0	1
2	1.525932e+09	108 211	4	1707	27647	4	5	3	1.0	60.0	0.0	0.0	1	1	0	1	0

3	1.5258 85e+09	16125	4	5051	7203	9653	5	0	1.0	40.0	0.0	0.0	0	0	1	0	1
4	1.5259 01e+09	25100	4	1807	3105 6	1964	5	3	1.0	60.0	0.0	0.0	1	1	0	1	0

Entrenando el modelo

Dividir para entrenar y probar

Entrada:

```
from sklearn.model_selection import train_test_split

X = df.drop('label', axis=1)
y = df['label']

# Split the DataFrame into X and y
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rand
om_state=42)
y_train.head()
y_train.value_counts()
```

Salida:

```
1    15979
0    15142
Name: label, dtype: int64
```

Entrenamiento del Modelo

Modelo de Soporte de Máquinas Vectoriales (SVM)

Entrada:

```
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
```

Ajuste de hiperparámetros (Hyperparameter Tunning)

Entrada:

```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
# # Assuming you have X_train_scaled, y_train defined

# # Define the hyperparameter grid
param_grid = {'C': [0.001,0.01,0.1, 1, 10], 'kernel': ['linear']}
```

```
# # Create the GridSearchCV object
grid_search = GridSearchCV(SVC(gamma='scale'), param_grid, cv=5)
grid_search.fit(X_train_scaled, y_train)

# # Get the best hyperparameters
best_params = grid_search.best_params_

# # Create a new model with the best hyperparameters
svm_model = SVC(kernel=best_params['kernel'], C=best_params['C'], gamma='scal
e')
svm_model.fit(X_train_scaled, y_train)
```

Salida:

```
SVC(C=0.001, kernel='linear')
```

Entrenamiento

Entrada:

```
svm_model = SVC(kernel='linear', C=0.001, gamma='scale')
svm_model.fit(X_train_scaled,y_train)
```

Salida:

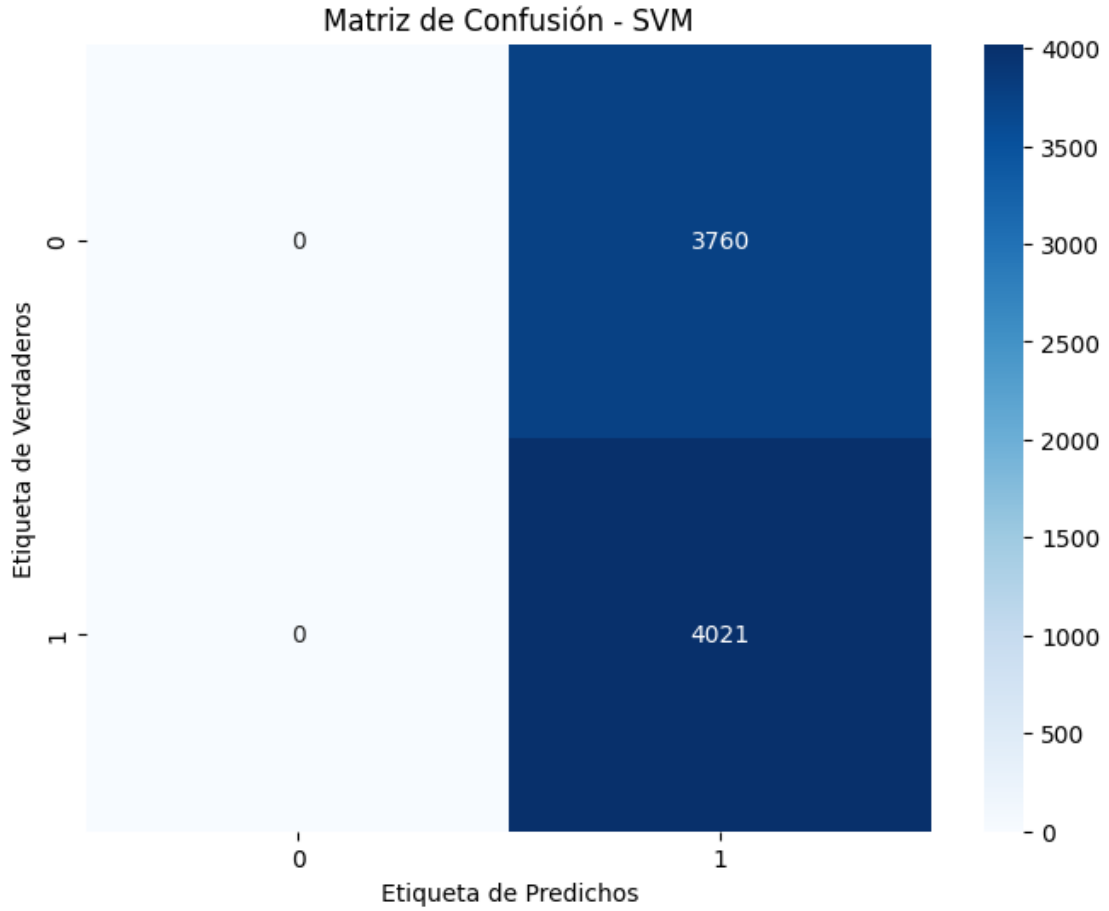
```
SVC(C=0.001, kernel='linear')
```

Evaluando el modelo

Entrada:

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print(f"Precisión (Accuracy): {accuracy}")
print(f"Reporte de Clasificación:\n{report}")
```

```
Precisión (Accuracy): 0.5167716231846806
Reporte de Clasificación:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      3760
           1       0.52      1.00      0.68      4021

    accuracy                           0.52      7781
   macro avg       0.26      0.50      0.34      7781
weighted avg       0.27      0.52      0.35      7781
```



Entrenamiento

```
nb_model.fit(X_train, y_train)
```

Haciendo predicciones

```
y_pred_nb = nb_model.predict(X_test)
```

Evaluando el modelo

```
accuracy_nb = accuracy_score(y_test, y_pred_nb)
report_nb = classification_report(y_test, y_pred_nb)

print(f"Precisión (Naive Bayes): {accuracy_nb}")
print(f"Reporte de Clasificación (Naive Bayes):\n{report_nb}")
```

```
Precisión (Naive Bayes): 0.8924302788844621
Reporte de Clasificación (Naive Bayes):
              precision    recall  f1-score   support

           0       0.83      0.97      0.90      3760
           1       0.97      0.82      0.89      4021

    accuracy                           0.89      7781
   macro avg       0.90      0.89      0.89      7781
weighted avg       0.90      0.89      0.89      7781
```

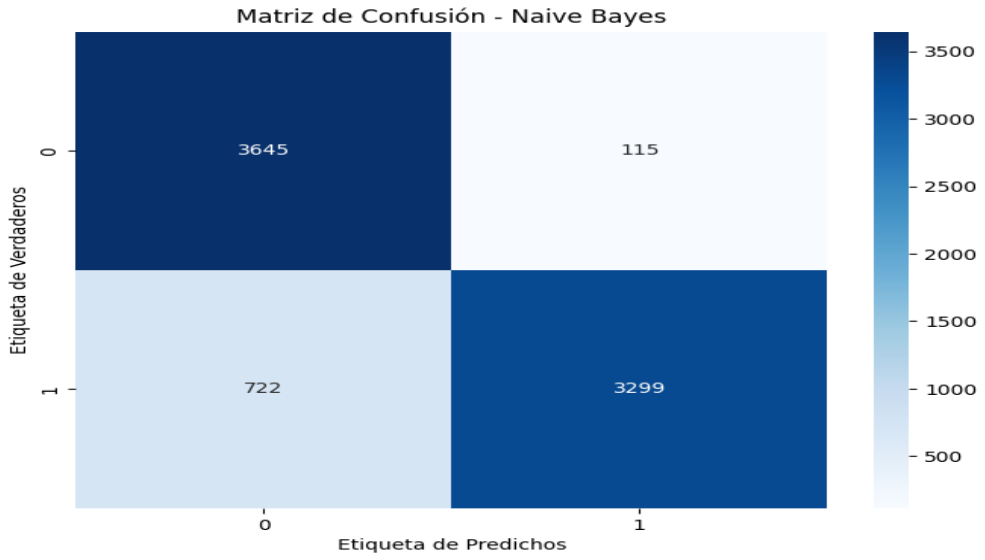
Entrada:

```
from sklearn.model_selection import cross_val_score, KFold


# number of folds for cross-validation
k_folds = 10

kf = KFold(n_splits=k_folds, shuffle=True, random_state=42)

clf_nb =nb_model

# performing k-fold cross-validation
cross_val_results = cross_val_score(clf_nb, X_train_scaled, y_train, cv=kf, s
coring='accuracy')

# results
print(f'Resultados de la validación cruzada: {cross_val_results}')
print(f'Precisión media: {cross_val_results.mean()}')
```

```
Resultados de la validación cruzada: [0.98490202 0.98264781 0.98296915 0.9855
3985 0.98329049 0.98071979
 0.98071979 0.9816838  0.98264781 0.9816838 ]
Precisión media: 0.9826804337395959
```

In [64]:

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Compute the confusion matrix
conf_matrix_nb = confusion_matrix(y_test, y_pred_nb)

# Display the confusion matrix using a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_nb, annot=True, fmt='d', cmap='Blues', xticklabels=['
0', '1'], yticklabels=['0', '1'])
plt.xlabel('Etiqueta de Predichos')
plt.ylabel('Etiqueta de Verdaderos')
plt.title('Matriz de Confusión - Naive Bayes')
plt.show()
```



ANEXO 2

Validación de la propuesta

INSTRUMENTO DE VALIDACIÓN

UNIVERSIDAD TECNOLÓGICA ISRAEL

ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN SEGURIDAD INFORMÁTICA

INSTRUMENTO PARA VALIDACIÓN DE LA PROPUESTA

Estimado colega:

Se solicita su valiosa cooperación para evaluar la siguiente propuesta del proyecto de titulación: **EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO.**

Sus criterios son de suma importancia para la realización de este trabajo, por lo que se le pide brinde su cooperación contestando las preguntas que se realizan a continuación.

Datos informativos

Validado por: Gabriel Eduardo Morejón López

Magíster en Gerencia de Sistemas, cursando estudios Doctorales en el área de Computación.

Título obtenido
Magíster en Gerencia de Sistemas
Cédula de Identidad
1309540779
E- mail
gabriel.morejon@utm.edu.ec
Institución de Trabajo
Universidad Técnica de Manabí
Cargo
Docente
Años de experiencia en el área
19

1

Instructivo:

- Responda cada criterio con la máxima sincera del caso;
- Revisar, observar y analizar la propuesta del proyecto de titulación; y,
- Coloque **una X** en cada indicador, tomando en cuenta que Muy adecuado equivale a 5, Bastante Adecuado equivale a 4, Adecuado equivale a 3, Poco Adecuado equivale a 2 e Inadecuado equivale a 1.

Tema: EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO.

Indicador	*Descripción*	Muy adecuado	Bastante Adecuado	Adecuado	Poco adecuado	Inadecuado
Impacto	*El alcance que tendrá la propuesta y su representatividad en la generación de valor*	X				
Aplicabilidad	*La capacidad de implementación de la propuesta considerando que los contenidos sean aplicables*	X				
Conceptualización	*La base de conceptos y teorías propias de la propuesta de manera sistémica y articulado*	X				
Actualidad	*Los procedimientos actuales y los cambios científicos y tecnológicos considerados en la propuesta*	X				
Calidad Técnica	*Los atributos cualitativos del contenido de la propuesta para satisfacer las expectativas de sus beneficiarios*	X				
Factibilidad	*El nivel de utilización de la propuesta por parte de la organización acorde a los recursos disponibles*	X				
Pertinencia	*La contundencia y conveniencia de la propuesta para solucionar el problema planteado.*	X				
Total		X				

Observaciones: Los algoritmos podrían haber llegado a analizar más información para ayudar a aumentar y mejorar el entrenamiento; pero eso se entiende que debe ser parte de la continuación del trabajo.

Recomendaciones: El estudio puede y debe ser complementado con otros trabajos a razón de dar continuidad a la propuesta, es necesario que se exploren más alternativas y evalúen los resultados, para lograr un entrenamiento más refinado de los algoritmos y modelos propuestos.

Lugar, fecha de validación: Portoviejo, 7 de marzo de 2024.

Firma del especialista

2

ANEXO 3

Validación de la propuesta

INSTRUMENTO DE VALIDACIÓN

UNIVERSIDAD TECNOLÓGICA ISRAEL

ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN SEGURIDAD INFORMÁTICA

INSTRUMENTO PARA VALIDACIÓN DE LA PROPUESTA

Estimado colega:

Se solicita su valiosa cooperación para evaluar la siguiente propuesta del proyecto de titulación: **EVALUACIÓN DEL DESEMPEÑO DE ALGORITMOS DE MACHINE LEARNING DENTRO DE LA IA PARA USO EN LA BÚSQUEDA DE PATRONES DE CIBERATAQUES Y MITIGACIÓN DE SU IMPACTO**

Sus criterios son de suma importancia para la realización de este trabajo, por lo que se le pide brinde su cooperación contestando las preguntas que se realizan a continuación.

Datos informativos

Validado por: LEONARDO JAVIER CHANCAY GARCÍA

Título obtenido
Doctor en Informática
Cédula de Identidad
1309566832
E- mail
leonardo.chancay@utm.edu.ec
Institución de Trabajo
Universidad Técnica de Manabí
Cargo
Docente Investigador
Años de experiencia en el área
16 años

1

Instructivo:

- Responda cada criterio con la máxima sincera del caso;
- Revisar, observar y analizar la propuesta del proyecto de titulación; y,
- Coloque **una X** en cada indicador, tomando en cuenta que Muy adecuado equivale a 5, Bastante Adecuado equivale a 4, Adecuado equivale a 3, Poco Adecuado equivale a 2 e Inadecuado equivale a 1.

Tema: Evaluación del desempeño de algoritmos de machine learning dentro de la IA para uso en la búsqueda de patrones de Ciberataques y mitigación de su impacto

Indicador	*Descripción*	Muy adecuado	Bastante Adecuado	Adecuado	Poco adecuado	Inadecuado
Impacto	*El alcance que tendrá la propuesta y su representatividad en la generación de valor*	X				
Aplicabilidad	*La capacidad de implementación de la propuesta considerando que los contenidos sean aplicables*	X				
Conceptualización	*La base de conceptos y teorías propias de la propuesta de manera sistémica y articulada*	X				
Actualidad	*Los procedimientos actuales y los cambios científicos y tecnológicos considerados en la propuesta*	X				
Calidad Técnica	*Los atributos cualitativos del contenido de la propuesta para satisfacer las expectativas de sus beneficiarios*	X				
Factibilidad	*El nivel de utilización de la propuesta por parte de la organización acorde a los recursos disponibles*	X				
Pertinencia	*La contundencia y conveniencia de la propuesta para solucionar el problema planteado.*	X				
Total		7				

Observaciones:

Se puede observar un trabajo estructurado, con una metodología práctica y fácil de seguir. Se han construido bases sólidas respecto al tema a través de la introducción y el marco teórico. Se presenta una propuesta y unos resultados acorde a los objetivos planteados. Se termina con unas conclusiones y recomendaciones, además de su bibliografía.

Recomendaciones

- El trabajo es bastante excelente por lo que se recomienda seguir realizando investigación sobre esta temática, ya que es un tema bastante interesante y amplio con grandes aportaciones a la seguridad informática del país y las empresas.

2

- El aporte del trabajo es bastante amplio y podría tener varias aportaciones o publicaciones a posterior en base a las simulaciones de los algoritmos realizadas.

Lugar, fecha de validación: Portoviejo, 8 de marzo del 2024

Firma del especialista