LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades, Asunción, Paraguay.
ISSN en línea: 2789-3855, marzo, 2025, Volumen VI

Modelo de Machine Learning para la Gestión de Amenazas con un SIEM de una Institución Financiera

Machine Learning Model for Threat Management with a SIEM of a Financial Institution

Maikel Martin Arbona
Marbona7557@utm.edu.ec
http://orcid.org/0009-0003-1202-8705i
Universidad Técnica de Manabí
Portoviejo – Ecuador

Roberth Abel Alcivar
roberth.alcivar@gmail.com
http://orcid.org/0000-0001-6282-8493
Universidad Técnica de Manabí
Portoviejo – Ecuador

DOI: https://doi.org/10.56712/latam.v6i2.3633

Artículo recibido: 06 de marzo de 2025.
Aceptado para publicación: 20 de marzo de 2025.
Conflictos de Interés: Ninguno que declarar.

Redilat
Red de Investigadores Latinoamericanos

NÚMERO 2

DOI: https://doi.org/10.56712/latam.v6i2.3633

Modelo de Machine Learning para la Gestión de Amenazas con un SIEM de una Institución Financiera

Machine Learning Model for Threat Management with a SIEM of a Financial Institution

Maikel Martin Arbona
Marbona7557@utm.edu.ec
http://orcid.org/0009-0003-1202-8705i
Universidad Técnica de Manabí
Portoviejo – Ecuador

Roberth Abel Alcivar
roberth.alcivar@gmail.com
http://orcid.org/0000-0001-6282-8493
Universidad Técnica de Manabí
Portoviejo – Ecuador

Artículo recibido: 06 de marzo de 2025. Aceptado para publicación: 20 de marzo de 2025.
Conflictos de Interés: Ninguno que declarar.

Resumen

La ciberseguridad es una prioridad para las instituciones financieras, que enfrentan desafíos constantes en la detección y gestión de amenazas. Este estudio evalúa el rendimiento de cinco modelos de Machine Learning (Random Forest, Support Vector Machine, Regresión Logística, K-Nearest Neighbors y Naive Bayes) en la identificación de anomalías en registros similares a los generados por un Sistema de Gestión de Información y Eventos de Seguridad (SIEM). Se adopta un enfoque metodológico basado en CRISP-ML(Q) y Kanban, combinando un marco estructurado de análisis de datos con una gestión ágil del desarrollo. El conjunto de datos utilizado, obtenido de Kaggle, incluye registros de tráfico normal y eventos anómalos. Se aplicaron técnicas de análisis exploratorio, limpieza y selección de características para optimizar el rendimiento de los modelos. La evaluación se realizó mediante métricas como precisión, recall, F1-score y área bajo la curva (AUC), con el objetivo de determinar el modelo más adecuado para la detección de amenazas. Los resultados muestran que la correcta selección de características y el uso de metodologías estructuradas pueden mejorar significativamente la detección de anomalías, reduciendo falsos positivos y optimizando la seguridad informática. Estos hallazgos contribuyen al desarrollo de soluciones más eficientes para la ciberseguridad en instituciones financieras y proporcionan una base para futuras investigaciones en la aplicación de Machine Learning en la detección de amenazas.

Palabras clave: aprendizaje automático, machine learning, seguridad informática, inteligencia artificial, SIEM

Abstract

Cybersecurity is a priority for financial institutions, which face constant challenges in detecting and managing threats. This study evaluates the performance of five Machine Learning models (Random Forest, Support Vector Machine, Logistic Regression, K-Nearest Neighbors and Naive Bayes) in identifying anomalies in logs similar to those generated by a Security Information and Event Management System (SIEM). A methodological approach based on CRISP-ML(Q) and Kanban is

adopted, combining a structured data analysis framework with agile development management. The dataset used, obtained from Kaggle, includes logs of normal traffic and anomalous events. Exploratory analysis, cleaning and feature selection techniques were applied to optimize the performance of the models. The evaluation was carried out using metrics such as precision, recall, F1-score and area under the curve (AUC), with the objective of determining the most appropriate model for threat detection. The results show that the correct selection of features and the use of structured methodologies can significantly improve anomaly detection, reducing false positives and optimizing computer security. These findings contribute to the development of more efficient solutions for cybersecurity in financial institutions and provide a basis for future research in the application of Machine Learning in threat detection.

Keywords: machine learning, machine learning, computer security, artificial intelligence, SIEM

Cómo citar: Arbona, M. M., & Alcivar, R. A. (2025). Modelo de Machine Learning para la Gestión de Amenazas con un SIEM de una Institución Financiera. *LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades 6 (2), 367 – 383. https://doi.org/10.56712/latam.v6i2.3633*

INTRODUCCIÓN

En la actualidad, la seguridad cibernética es una prioridad para las instituciones financieras, las cuales enfrentan desafíos constantes para proteger sus activos digitales (Alimi, Ouahada, & Abu-Mahfouz, 2020). A pesar de contar con áreas especializadas en seguridad de la información, muchas organizaciones se ven sobrepasadas por el volumen excesivo de alertas generadas por sus Sistemas de Gestión de Información y Eventos de Seguridad (SIEM, por sus siglas en inglés), lo que compromete la eficiencia en la respuesta a incidentes (Aminanto, Ban, Isawa, Takahashi, & Inoue, 2020). En este contexto, la integración de tecnologías avanzadas, como el Machine Learning (ML), se presenta como una solución prometedora para mejorar la detección de amenazas y reducir los falsos positivos en la gestión de incidentes de seguridad (Zhong, Lin, Liu, Yen, & Chen, 2018).

Diversos estudios han explorado la aplicación del ML en la seguridad informática, destacando su capacidad para analizar grandes volúmenes de datos y detectar patrones anómalos con mayor precisión que los enfoques tradicionales (Hindy, Brosset, Bellekens, Bayne, & Amar, 2018). En investigaciones previas, modelos como Random Forest, Support Vector Machine (SVM) y Árboles de Decisión han demostrado un desempeño sobresaliente en la detección de amenazas, superando en muchos casos a métodos convencionales (Waskle, Parashar, & Singh, 2020). Sin embargo, existen limitaciones en estos estudios, como la falta de una evaluación comparativa integral de múltiples modelos y el uso de conjuntos de datos limitados o poco representativos. Además, se ha identificado que la correcta selección de características puede mejorar significativamente el desempeño de ciertos algoritmos, como lo demuestra el trabajo de Zhang et al. (2022), en el cual la aplicación de una estrategia de selección de características elevó la precisión de Random Forest.

Para abordar estas carencias, este estudio evalúa de manera exhaustiva el rendimiento de cinco modelos de ML (Random Forest, SVM, Regresión Logística, K-Nearest Neighbors y Naive Bayes) en la detección de amenazas en registros similares a los generados por un SIEM de una institución financiera. A diferencia de estudios previos, se emplea un enfoque metodológico que combina fases de la metodología CRISP-ML en la metodología Kanban. CRISP-ML proporciona un marco estructurado para la preparación y modelado de datos, asegurando la validez del proceso de desarrollo, mientras que Kanban facilita la gestión ágil y adaptativa del proyecto, permitiendo iteraciones continuas en la optimización de los modelos (Castellanos, Cortés, Espitia, & Garzón, 2020).

Para abordar estas carencias, este estudio evalúa de manera exhaustiva el rendimiento de cinco modelos de ML (Random Forest, SVM, Regresión Logística, K-Nearest Neighbors y Naive Bayes) en la detección de amenazas en registros similares a los generados por un SIEM de una institución financiera (Castellanos, Cortés, Espitia, & Garzón, 2020). A diferencia de estudios previos, se emplea un enfoque metodológico que toma de referencia los primeros cuatro pasos de la metodología CRISP-ML con la metodología Kanban. CRISP-ML (Q). La integración con Kanban facilita la gestión ágil y adaptativa del proyecto, permitiendo iteraciones en la optimización de los modelos (Studer et al., 2021).

El objetivo general de esta investigación es desarrollar un modelo de ML para mejorar la gestión de amenazas en un SIEM de una institución financiera, asegurando su capacidad para identificar ataques con mayor eficacia.

Los resultados de este estudio contribuyen a la optimización de la seguridad informática, proporcionando evidencia sobre la efectividad de ciertos modelos y estableciendo una base sólida para su implementación en entornos reales. Además, estos hallazgos sirven como referencia para futuras investigaciones que busquen mejorar la detección de amenazas mediante técnicas avanzadas de ML.

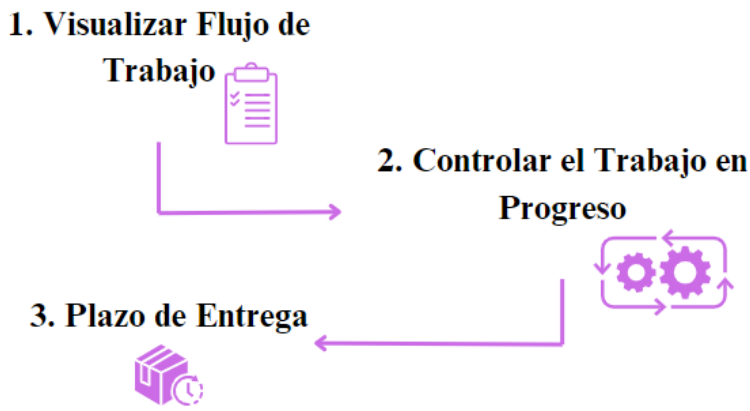
METODOLOGÍA

Se escogió la metodología Kanban por su flexibilidad en desarrollo de proyectos, gracias a que permite listar y clasificar las tareas en pendientes, en proceso y completadas, lo cual es útil en proyectos de Machine Learning que requieren múltiples iteraciones y ajustes continuos. En esta metodología las tareas principales se definen al inicio, pero cuenta con la característica que, de ser necesario se agregan tareas a medida que se avanza en la elaboración del modelo, además proporciona una visión general de las tareas pendientes, las tareas en proceso y las tareas completadas (Gilibets , 2023).

La metodología CRISP-ML(Q) consta de seis fases: comprensión del negocio y de los datos, preparación de los datos, modelado, evaluación, despliegue y monitoreo. En este estudio, se han tomado como referencia las primeras cuatro fases para guiar el desarrollo del modelo de Machine Learning. En la fase de comprensión del negocio y de los datos, se garantiza que el enfoque del proyecto esté alineado con los objetivos comerciales y que los datos disponibles sean adecuados para la tarea de detección de amenazas. Posteriormente, en la fase de preparación de los datos, se aplican técnicas como limpieza, etiquetado y selección de características para mejorar su calidad y maximizar el rendimiento del modelo. En la fase de modelado, se entrenan distintos algoritmos de ML utilizando los datos procesados, optimizando hiperparámetros y aplicando validación cruzada. Finalmente, en la fase de evaluación, se mide el desempeño de los modelos mediante métricas clave, como precisión, recall, F1-score, exactitud y área bajo la curva (AUC), con el objetivo de identificar el modelo más adecuado para la gestión de amenazas dentro del SIEM. En este estudio se omiten las fases de despliegue y monitoreo, ya que el enfoque se centra en la selección y evaluación del modelo óptimo, sin abordar su implementación en un entorno real (Studer et al., 2021).

Figura 1

Kanban en Desarrollo Software



Nota: En esta figura se muestran los pasos a seguir para aplicar correctamente la metodología Kanban en el desarrollo de Software.

Tabla 1

Flujo de Trabajo

Fases	Tareas
Importación de Librerías y Carga de Datos	Preparar Entorno de Trabajo, Seleccionar Lenguaje de Programación, Investigar y Seleccionar Librerías, Importar Librerías en el Entorno, Realizar Conexión con Drive y Cargar el Conjunto de Datos.

Análisis Exploratorio de Datos	Realizar Análisis Exploratorio de los Datos, Generar Gráficos y Visualizaciones para las dimensiones del Dataset, Identificar y Documentar características de los datos (tipo decimal, entero, objeto).
Preparación y limpieza de los datos	Manejo de valores nulos y eliminación de duplicados, Normalizar Registros, Aplicar la Eliminación Recursiva de Características.
Entrenamiento y Evaluación del Modelo	Separar el dataset en conjuntos de entrenamiento y prueba, Seleccionar Modelos, Evaluar el rendimiento de cada modelo utilizando métricas.

La metodología Kanban aplicada al desarrollo de software tiene tres reglas que garantizan un correcto desempeño en su aplicación, las cuales se presentan en la Figura 1. La primera regla consiste en visualizar el flujo de trabajo, manteniendo en mente los entregables y el proceso seguido para realizarlos. La segunda es establecer un límite para el trabajo en progreso (WIP, por sus siglas en inglés), con el fin de evitar cuellos de botella. Finalmente, la tercera regla es controlar el tiempo necesario para completar una actividad (Lead time), lo que evitó retrasos innecesarios en el desarrollo del modelo (Yépez & Armijos, 2020).

Como se observa en la Tabla 1, el desarrollo está dividido en 4 fases, y cada una tiene tareas específicas definidas. Al hacerlo estructurado el objetivo es garantizar un flujo de trabajo ordenado, desde la preparación del entorno hasta la evaluación de los modelos. La planificación de estas tareas permite una gestión adecuada del proceso de modelado y entrenamiento, asegurando que cada etapa contribuya de manera coherente. De esta forma, se asegura un desarrollo escalable y replicable para proyectos futuros de análisis y clasificación de eventos en sistemas de seguridad.

Fase 1: Importación de Librerías y Carga de Datos

Esta fase es importante para el desarrollo del modelo de ML, ya que sienta las bases técnicas, respaldado por la metodología Kanban la cual permitió gestionar el flujo de trabajo.

Tabla 2

Tablero Kanban Fase 1

Tareas	En Proceso	Completadas
Investigar y Seleccionar Librerías	Preparar Entorno de Trabajo	
Importar Librerías en el Entorno	Seleccionar Lenguaje de Programación	
Realizar Conexión con Drive y Cargar el Conjunto de Datos		

Nota: Esta tabla refleja el estado inicial del tablero Kanban. A medida que se completan las tareas, aquellas que están “En proceso” se trasladan a la columna de “Completadas” y es algo presente en todas las fases del desarrollo.

Como se observa en la Tabla 2, las tareas principales que estaban en proceso incluían la preparación del entorno y la selección del lenguaje de programación. El entorno de desarrollo elegido fue Google Colab por su capacidad de acceso a recursos computacionales, como unidades de procesamiento gráfico (GPUs, por sus siglas en inglés) y unidades de procesamiento tensorial (TPUs, por sus siglas en inglés) (thiago-gsantos03, 2020), (Jesús, y otros, 2022), lo que lo hace ideal para el entrenamiento de modelos de ML, el lenguaje de programación que se utilizó es Python debido a la amplia disponibilidad de bibliotecas especializadas en análisis de datos y aprendizaje automático (Rincón, 2021). Para maximizar su potencial, se importaron diversas librerías. Entre ellas, están Scikit-learn

(Sklearn), una biblioteca que ofrece una gran colección de algoritmos y herramientas para modelado, selección de características y métricas de evaluación (Robotko, Topalov, Nekrasov, Zaytsev, & Zaytsev, 2023). La carga de datos se realizó utilizando Pandas, lo que facilitó la importación de archivos de valores separados por comas (CSV, por sus siglas en inglés) que contienen los registros necesarios para el entrenamiento y la prueba del modelo. Para la importación y el acceso a los datos se empleó la biblioteca Drive. Esta herramienta permite acceder y gestionar archivos en Google Drive directamente desde el entorno de trabajo de Google Colab utilizando Python, asegurando la integración con los conjuntos de datos almacenados en la nube.

Uno de los principales desafíos en el desarrollo de modelos de machine learning para la ciberseguridad es obtener conjuntos de datos que reflejen adecuadamente la realidad y estén bien balanceados. Esto significa que los datos deben representar la variedad de situaciones que pueden ocurrir en un entorno real, incluyendo tanto eventos normales como amenazas o ataques. Además, los conjuntos de datos deben estar balanceados en cuanto a la cantidad de ejemplos de cada tipo de evento, evitando un sesgo hacia los eventos más comunes (como el tráfico normal) que podría hacer que el modelo tenga dificultades para identificar situaciones anómalas o ataques menos frecuentes. Los conjuntos de datos disponibles frecuentemente no son homogéneos y, a menudo, no capturan adecuadamente la variedad y complejidad de los ataques cibernéticos que ocurren en el mundo real (Xin, y otros, 2018). Como solución, se ha sugerido la creación de nuevos conjuntos de datos y su disponibilidad pública para mejorar la diversidad de los datos utilizados en el entrenamiento de los modelos (Yavanoglu & Aydos, 2017). Para abordar esta necesidad de datos balanceados y representativos, en este estudio se empleó un conjunto de datos descargado de Kaggle, adaptado para reflejar los registros típicos que el SIEM de una institución financiera recibiría. Este conjunto de datos está diseñado para entrenar modelos de ML e incluye tanto actividades normales como maliciosas, clasificadas en dos categorías dentro de la dimensión “class”: “Normal” y “Anomaly”.

Fase 2: Análisis Exploratorio de Datos

En esta fase, se llevó a cabo una descripción de los datos con el objetivo de comprender mejor su estructura y características antes de proceder con su preparación y limpieza. Este análisis inicial proporciona una visión clara del conjunto de datos, garantizando una adecuada manipulación en las fases posteriores.

Tabla 3

Tablero Kanban Fase 2

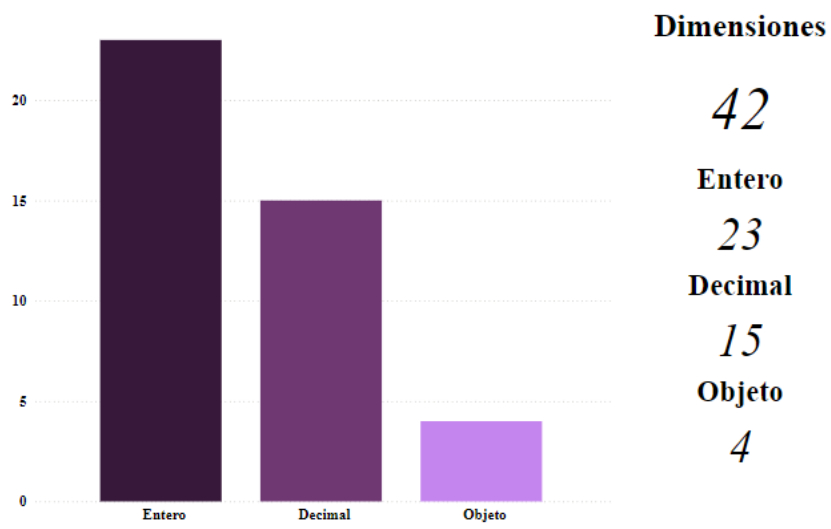
Tareas	En Proceso	Completadas
Identificar y Documentar características de los datos (tipo decimal, entero, objeto).	Realizar Análisis Exploratorio de los Datos	Tareas Fase 1
	Generar Gráficos y Visualizaciones para las dimensiones del Dataset	

Nota: Al pasar a la Fase 2, todas las tareas de la Fase 1 se desplazan a la columna “Completadas”

Como se observa en la Tabla 3, las tareas principales incluyen el Análisis Exploratorio de Datos (EDA, por sus siglas en inglés) y la generación de visualizaciones estadísticas de las dimensiones del conjunto de datos. Este conjunto cuenta con 42 dimensiones, entre las cuales destaca la denominada “class”, que categoriza cada fila. Esta característica es importante, ya que el modelo desarrollado es supervisado y utiliza estos registros para entrenarse en la detección de anomalías.

Gráfico 1

Conteo de Tipos de Datos en las Dimensiones del Conjunto



Como se observa en el gráfico 1, de las 42 dimensiones, 15 son de tipo decimal, indicando que contienen valores numéricos con decimales, comúnmente utilizados para representar datos continuos o medidas precisas. 23 dimensiones son de tipo entero, lo que significa que contienen valores numéricos sin decimales, para datos discretos o conteos. Finalmente, 4 dimensiones son de tipo objeto, lo que generalmente representa variables categóricas.

Tabla 4

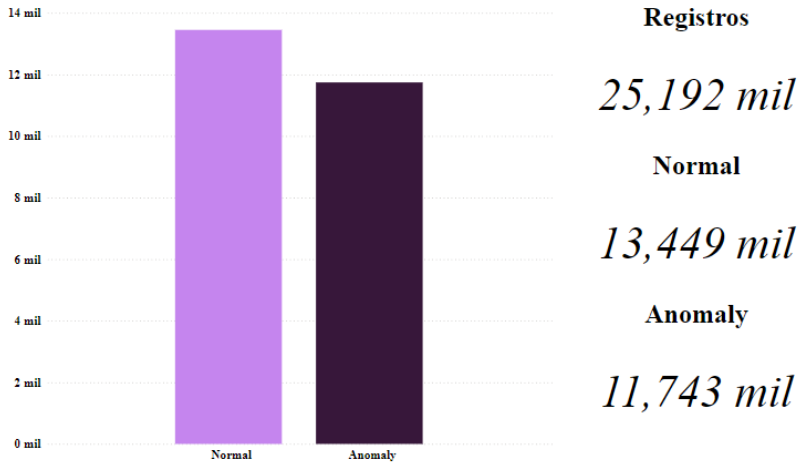
Descripción de Dimensiones Tipo Objeto

Dimensiones	Máximo Valor	Valores Únicos	Frecuencia	Conteo Total
Protocol_type	Tcp	3	20526	25192
Service	http	66	8003	25192
Flag	SF	11	14973	25192
Class	normal	2	13449	25192

Nota: la “Frecuencia” toma en cuenta la cantidad de veces que se repite el “Máximo Valor”.

Gráfico 2

Conteo de la Dimensión Class



La Tabla 4 presenta una descripción de las dimensiones de tipo objeto en el conjunto, que incluye cuatro dimensiones “Protocol_type, Service, Flag y Class” En total, hay 25,192 registros, como se indica en la columna "CONTEO TOTAL." La columna "VALORES ÚNICOS" revela cuántos valores diferentes se encuentran dentro de los registros, y "MÁXIMO VALOR" señala el valor que más veces se repite.

La dimensión "Class," que es la que se encargará de categorizar los registros cuenta con dos Valores Únicos que son “Normal” y “Anomaly” destacando “Normal” apareciendo en 13,449 registros, este valor se encuentra reflejada en la columna "FRECUENCIA," coincidiendo con la visualización del gráfico 3. Por otro lado, el valor único “Anomaly” cuenta con 11,743 registros. Por ende, esta distribución, con un número significativo de registros para cada clase, beneficia el entrenamiento del modelo de ML ya que le permite aprender mejor durante el entrenamiento y también distinguir entre categorías al momento de enfrentarse a nuevos registros.

Fase 3: Preparación y limpieza de los datos

Para garantizar la calidad del modelo de ML, los datos se sometieron a procesos de preparación y limpieza. En primer lugar, se eliminaron los registros con valores nulos y duplicados. Posteriormente, se usaron funciones de la librería Sklearn descrita en la fase 1, las variables categóricas se codificaron mediante 'Label Encoding', un método que transforma las categorías en valores numéricos. Este paso es importante, ya que los modelos de machine learning procesan mejor los datos en formato numérico y sin esta transformación, las variables categóricas no serían adecuadamente interpretadas. Esto asegura que los modelos puedan aprender patrones de las categorías de manera precisa, sin perder información. Después, se aplicó la normalización mediante el método 'Min-Max Scaling' a las variables numéricas para ajustar sus valores al rango [0, 1]. Buscando así mantener todas las características en una escala uniforme, evitando que aquellas con valores más altos afecten de forma desproporcionada el proceso de aprendizaje del modelo. La normalización garantiza así que cada variable numérica contribuya de manera equilibrada a los resultados, esto permitió que el modelo identificara patrones de manera precisa sin que las diferencias de magnitud entre variables interfieran en su desempeño.

Tabla 5

Tablero Kanban Fase 3

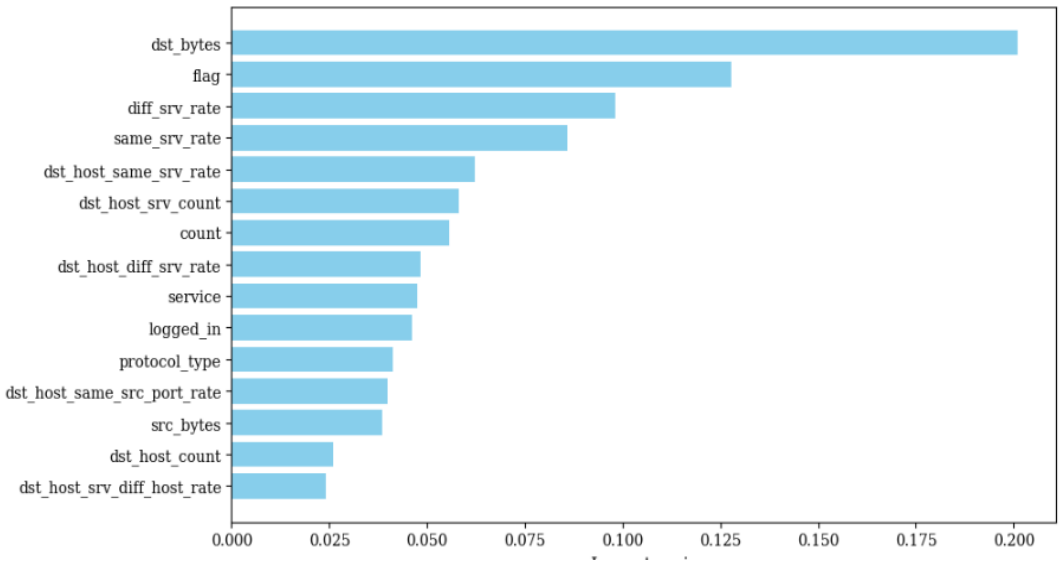
Tareas	En Proceso	Completadas
Normalizar Registros	Eliminar Valores Nulos	Tareas Fase 1
Aplicar la Eliminación Recursiva de Características	Remover Registros Duplicados	Tareas Fase 2

Nota: La columna de Completadas está dividida en dos, las primeras tareas realizadas son las de la columna izquierda y posteriormente se agregaron las de la columna derecha

Como se observa en la Tabla 5, una tarea que también se realizó es la técnica de Eliminación Recursiva de Características (RFE, por sus siglas en inglés) para seleccionar las características más relevantes, utilizando un Clasificador de bosques aleatorio (RFC, por sus siglas en inglés) como estimador (Kornyo, y otros, 2023), lo que permitió reducir el número de características de 42 a 15 y mejoró la precisión del modelo al usar solo las características más importantes, como se indica en el estudio (Barreno, Nelson, Joseph, & Tygar, 2010). Al limitar el modelo a trabajar solo con las características más relevantes, se optimizó el proceso de aprendizaje, mejorando así su capacidad para generalizar sobre datos nuevos. Posteriormente, se aplicó la validación cruzada, un método que permite evaluar de forma confiable el rendimiento del modelo dividiendo el conjunto de datos en varias partes o "subconjuntos". En cada ronda, se entrena el modelo con una parte de los datos y se prueba con otra, repitiendo el proceso según las partes en las que se haya dividido, en este caso se dividió en 5 subconjuntos. Al finalizar, el resultado fue un promedio de precisión del 99%. Este resultado es coherente con las expectativas, dado que el conjunto de datos fue diseñado para entrenar modelos de ML enfocados en detección de amenazas.

Gráfico 3

Importancia de las Dimensiones Seleccionadas por RFE



Como se observa en el grafico 3, las dimensiones seleccionadas por el RFE, ordenadas por su valor de importancia son: la cantidad de bytes enviados al destino (dst_bytes, 0.2009), el estado de la conexión (flag, 0.1278), la tasa de conexiones a servicios distintos (diff_srv_rate, 0.0980), la tasa de conexiones

que utilizan el mismo servicio (same_srv_rate, 0.0859), la tasa de conexiones al mismo servicio en el host de destino (dst_host_same_srv_rate, 0.0621), el número de conexiones realizadas al host de destino (dst_host_srv_count, 0.0581), el número total de conexiones realizadas (count, 0.0558), la tasa de conexiones a servicios distintos en el host de destino (dst_host_diff_srv_rate, 0.0483), el tipo de servicio utilizado (service, 0.0474), un indicador de inicio de sesión del usuario (logged_in, 0.0460), el tipo de protocolo (protocol_type, 0.0411), la tasa de conexiones al mismo puerto en el host de destino (dst_host_same_src_port_rate, 0.0399), la cantidad de bytes enviados desde la fuente (src_bytes, 0.0386), el número de conexiones al host de destino (dst_host_count, 0.0259), y la tasa de conexiones al host de destino con diferentes servicios (dst_host_srv_diff_host_rate, 0.0242). Entre estas dimensiones, la de mayor importancia es “dst_bytes”, debido a su capacidad para identificar actividades inusuales o sospechosas, como los ataques de denegación de servicio (DoS). Este indicador es clave para clasificar distintos tipos de tráfico, permitiendo distinguir entre tráfico normal y tráfico asociado a ataques (Yang, Peng, Zhang, & Lv, 2022).

Tabla 6

Modificaciones en el Dataset

Antes	Ahora
42 dimensiones	15 dimensiones
Variables Objeto sin codificar	Variables Objeto Codificadas
Valores sin Normalizar	Valores Normalizados
Valores Faltantes sin Revisar	Valores Faltantes Revisados y Corregidos
Valores Nulos sin Revisar	Valores Nulos Revisados y Corregidos

Como se observa en la Tabla 6, el conjunto de datos final carece de duplicados y valores nulos. El número de dimensiones se redujo de 42 a 15, y todas las variables numéricas se normalizaron, mientras que las variables categóricas se codificaron. Con estas actualizaciones, el conjunto de datos quedó listo para la siguiente fase.

Fase 4: Entrenamiento y Evaluación del Modelo

Esta es la fase final, en la que el conjunto de datos se puso a prueba con los ajustes realizados. Además, se definieron las métricas y los modelos de ML que se evaluaron para identificar el de mayor rendimiento.

Tabla 7

Tablero Kanban Fase 4

Tareas	En Proceso	Completadas
Evaluar el rendimiento de cada modelo utilizando métricas.	Dividir el dataset en conjuntos de entrenamiento y prueba.	Tareas Fase 1 Tareas Fase 2 Tareas Fase 3
	Seleccionar Modelos	

Nota: Esta tabla muestra la fase final del tablero Kanban, Se aprecia que todas las tareas de las fases anteriores se encuentran Completadas.

Tabla 8

Modelos de ML Seleccionados

Modelo	Descripción
Random Forest	Construye múltiples árboles de decisión para manejar grandes volúmenes de datos
k-Nearest Neighbors	Agrupa datos en función de características similares
Logistic Regression	Modelo estadístico utilizado para clasificar eventos binarios
Support Vector Machine	Busca un límite óptimo para separar categorías en datos complejos
Naïve Bayes	Clasifica eventos usando probabilidades y asume que las características son independientes

La evaluación de los modelos de machine learning abarca desde la preparación y división del conjunto de datos hasta el uso de métricas que miden su eficacia y capacidad para clasificar nuevos registros, como se observa en la Tabla 7, para cumplir una de las primeras tareas se dividió el conjunto de datos en dos partes, siguiendo el procedimiento realizado en el estudio (Hindy, Brosset, Bellekens, Bayne, & Amar, 2018), una de las partes es de entrenamiento que compone el 70% y una de prueba que compone el 30%, permitiendo así evaluar el rendimiento del modelo en datos no conocidos por el modelo y obtener una medida más precisa de su capacidad de clasificar.

Hay modelos que se han implementado para detectar amenazas cibernéticas, demostrando su eficacia en la identificación de patrones anómalos y en la clasificación de eventos de seguridad. Algunos de esos modelos más utilizados son el Bosque Aleatorio (Random Forest) y el Vecinos más Cercanos (k-Nearest Neighbors, k-NN) (Suyal & Goyal, 2022), conocidos por su alta precisión en entornos de seguridad informática. El Bosque Aleatorio, un método basado en la construcción de múltiples árboles de decisión, es eficaz en el manejo de grandes volúmenes de datos con alta dimensionalidad, en estudios ha logrado precisiones superiores al 95% (Hindy, Brosset, Bellekens, Bayne, & Amar, 2018), (Li, Xiong, Chin, & Hu, 2019). El modelo k-NN, por otro lado, se destaca por su capacidad para agrupar datos según características similares, lo cual facilita la identificación de anomalías en grandes volúmenes de información (Alimi, Ouahada, & Abu-Mahfouz, 2020), (Bagaa, Taleb, Bernabe, & Skarmeta, 2020). Además de estos, el estudio (Hindy, Brosset, Bellekens, Bayne, & Amar, 2018) ha evaluado diferentes algoritmos de Aprendizaje Automático con buenos resultados en la detección de amenazas, incluidos la Regresión Logística (Logistic Regression), Naïve Bayes, las Máquinas de Vectores de Soporte (Support Vector Machine, SVM) y los mencionados anteriormente. Estos modelos, empleados en diferentes entornos, mostraron un desempeño confiable al clasificar comportamientos anómalos, lo que resalta su potencial para mejorar los sistemas de detección de amenazas. Un resumen de estos algoritmos se presenta en la Tabla 8.

Tabla 9

Tipos de Predicciones

Predicción	Descripción
Verdadero Positivo (TP)	El modelo predijo correctamente un resultado positivo
Falso Positivo (FP)	El modelo predijo un resultado positivo, pero en realidad era negativo
Verdadero Negativo (TN)	El modelo predijo correctamente un resultado negativo
Falso Negativo (FN)	El modelo predijo un resultado negativo, pero en realidad era positivo

Nota: Es necesario tener en cuenta estas predicciones ya que la mayoría de métricas las utilizan.

En la tabla 9 se observan las predicciones que se usan para calcular las métricas seleccionadas, la primera Métrica que se consideró es la precisión, que indica el porcentaje de predicciones correctas sobre el total de predicciones realizadas, se escogió para evaluar la capacidad del modelo para identificar correctamente las amenazas cibernéticas en relación con todas las instancias que predice como amenazas. Un alto valor de precisión significa un bajo número de falsos positivos, minimizando las falsas alarmas.

$$Precisión = \frac{TP}{TP + FP}$$

La sensibilidad (recall) mide la proporción de verdaderos positivos que son correctamente identificados, crucial para evaluar qué tan bien el modelo puede detectar todas las amenazas reales. Un alto recall significa un bajo número de falsos negativos.

$$Recall = \frac{TP}{TP + FN}$$

La otra métrica utilizada es la medida F1 (F1 Score), la media armónica entre precisión y recall, ofrece una medida más equilibrada del rendimiento del modelo, especialmente en problemas de clasificación desequilibrada. Un alto F1 Score indica una buena precisión y un buen recall, esencial para detectar amenazas con alta fiabilidad.

$$F1\ Score = 2\ x\ \frac{Precisión\ x\ Recall}{Precisión\ + Recall}$$

La Exactitud (Accuracy) mide el porcentaje de predicciones correctas entre todas las predicciones realizadas, ofreciendo una visión general del rendimiento del modelo.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Como última métrica se usó del Área Bajo la Curva (AUC, por sus siglas en inglés) evalúa la capacidad del modelo para distinguir entre clases, proporcionando una medida agregada del rendimiento en todos los umbrales de clasificación posibles. En relación a las categorías utilizadas (Normal, Anomaly), un alto valor de AUC indica una buena discriminación entre actividades normales y maliciosas

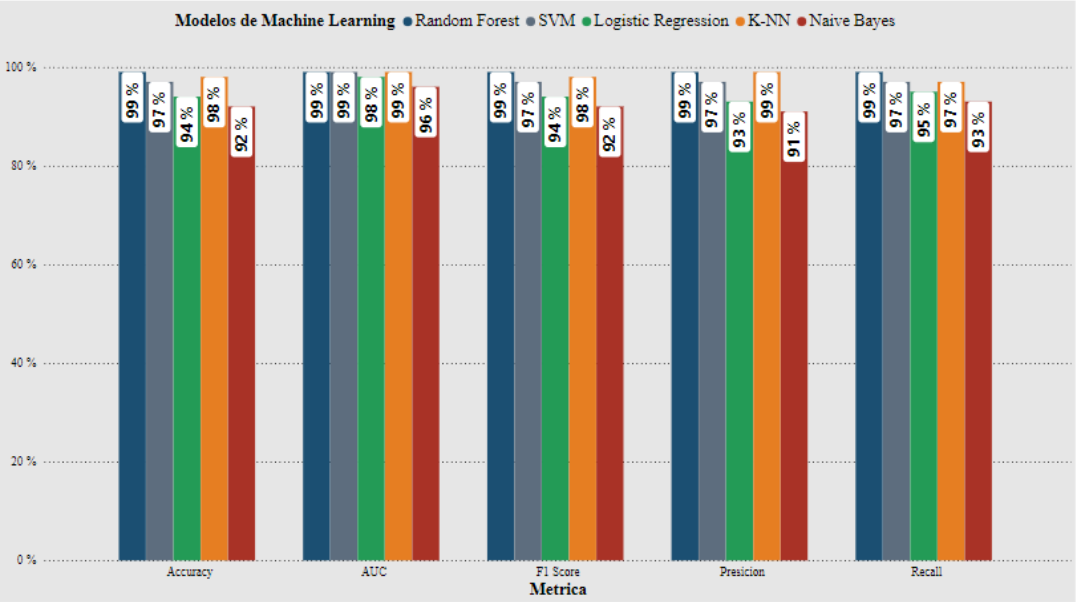
Estas métricas se seleccionaron debido a su relevancia en estudios previos relacionados con la detección de amenazas cibernéticas, como se observa en investigaciones (Bagaa, Taleb, Bernabe, & Skarmeta, 2020), (Xue, Yuan, Wu, Zhang, & Liu, 2020). Su uso permite evaluar el rendimiento del modelo y garantizar una interpretación de su capacidad predictiva. La selección de métricas como precisión, recall, F1 Score, exactitud y AUC permitió identificar fortalezas y debilidades en modelos de Machine Learning, facilitando así su comparación.

RESULTADOS

A continuación, se presentan los resultados obtenidos a partir de la evaluación de los modelos de ML representados dentro de la tabla 8. Este estudio ha evaluado el rendimiento de cinco modelos en la gestión de amenazas a partir de los registros similares a los generados por un SIEM de una institución financiera. Los resultados obtenidos demostraron el potencial de estos modelos para mejorar la seguridad informática.

Gráfico 4

Resultados de Modelos de ML Evaluados



Como se observa en el gráfico, los resultados de los modelos de ML junto a sus métricas evaluadas están cercanos al 100%, lo cual era previsible debido a las pruebas realizadas mediante validación cruzada con un promedio en sus cinco partes de 99%. Sin embargo, un modelo destaca sobre los demás, el cual es Random Forest. Esto es coherente con la capacidad de este algoritmo para realizar divisiones precisas en escenarios de clasificación binaria, en este caso, las clases “Normal” y “Anomaly”. Por otro lado, el modelo con los resultados más bajos en comparación con los demás fue Naive Bayes. A pesar de ello, los demás modelos también mostraron un rendimiento notable, y a continuación se analizarán en mayor detalle los resultados de cada uno.

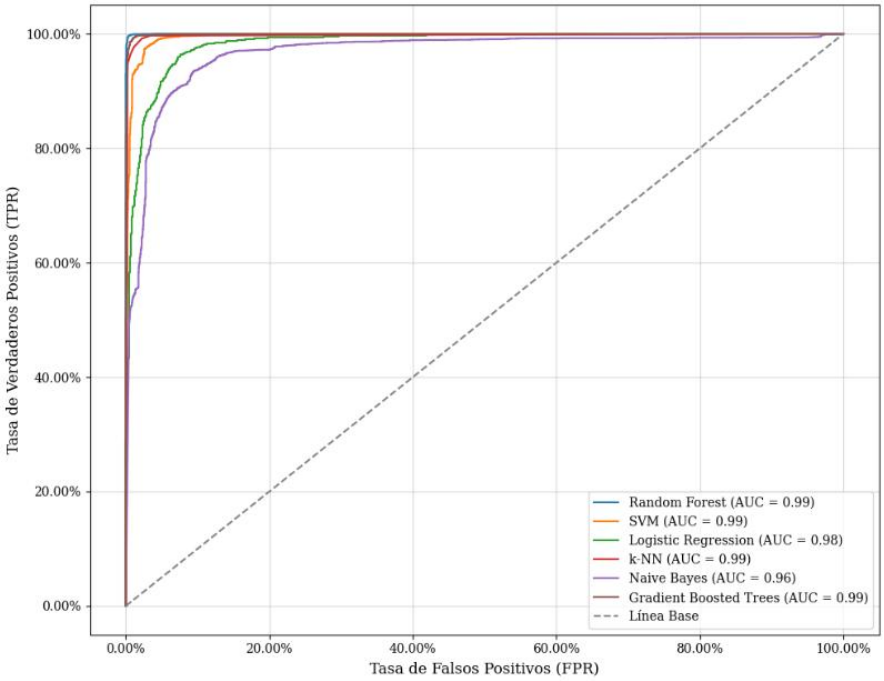
Tabla 10

Modelos de ML Evaluados

Modelo	Precisión	Recall	F1- SCORE	exactitud	AUC
Random Forest	99%	99%	99%	99%	99%
SVM	97%	97%	97%	97%	99%
Logistic Regression	93%	95%	94%	94%	98%
K-NN	99%	97%	98%	98%	99%
Naive Bayes	91%	92%	93%	92%	96%

Gráfico 5

AUC de los Modelos de ML



Como se observa en la Tabla 10, el modelo Random Forest se destacó como el más efectivo, alcanzando un 99% en todas las métricas evaluadas. Estos resultados confirman su sobresaliente desempeño en la clasificación y su capacidad para distinguir con precisión entre clases. El modelo SVM, aunque ligeramente inferior, logró métricas cercanas, con un 97% en precisión, recall, F1 Score y exactitud, y un AUC igualmente alto del 99%, consolidando su capacidad de discriminación entre clases. La Regresión Logística mostró un buen desempeño general, obteniendo un 93% en precisión, 95% en recall, 94% en F1 Score, una exactitud del 94% y un AUC de 98%, lo que refuerza su solidez como modelo de clasificación, aunque con un rendimiento algo menor en comparación con Random Forest y SVM. El modelo k-NN destacó por su balance entre métricas, logrando un 99% en precisión, 97% en recall y 98% en F1 Score. Con una exactitud del 98% y un AUC de 99%, se posiciona como uno de los modelos más competitivos. Por otro lado, el modelo Naive Bayes presentó un desempeño aceptable, aunque inferior al resto, con un 91% en precisión, 93% en recall, 92% en F1 Score, una exactitud del 92% y un AUC de 96%, evidenciando su utilidad como una opción básica en problemas de clasificación. En el gráfico 5 se observa más a detalle el AUC de cada Modelo.

Aunque todos los modelos evaluados mostraron un rendimiento sobresaliente, Random Forest destacó sobre los demás. En (Waskle, Parashar, & Singh, 2020), se compararon tres modelos de Machine Learning: Random Forest, SVM y Árboles de Decisión. En ese estudio, Random Forest superó a los otros dos modelos, alcanzando métricas superiores al 96%. Por otro lado en (Zhang, Wang, Liu, Ren, & Wang, 2022), se utilizó un modelo basado en Random Forest que incorporaba una selección de características previas, considerando la importancia de cada variable. Este proceso es similar al empleado en la preparación de los datos durante la fase 3 de nuestro modelo. En dicha investigación, inicialmente se aplicó Random Forest sin selección de características, logrando métricas satisfactorias (94% en precisión, recall y F1-score). Sin embargo, al aplicar la selección de características, estas métricas aumentaron hasta un 99%, muy cerca del valor ideal. Esto demuestra que Random Forest, con una correcta preparación, es una excelente opción para la detección de amenazas y, por ende, debería considerarse al implementar como apoyo en un SIEM de una institución financiera.

CONCLUSIÓN

Este desarrollo evaluó el rendimiento de cinco modelos de Machine Learning en la gestión de amenazas cibernéticas en un entorno de seguridad informática, empleando un conjunto de datos que simula registros generados por un SIEM de una institución financiera.

La comparación de modelos reveló que Random Forest fue el más efectivo, alcanzando un 99% en todas las métricas evaluadas (precisión, recall, F1-score, exactitud y AUC). Otros modelos, como SVM y k-NN, también mostraron un rendimiento cercano al 99%, demostrando su competitividad. Por otro lado, Naive Bayes, aunque con resultados aceptables, se posicionó como el modelo menos destacado. Estas diferencias reflejan las capacidades de cada algoritmo y destacan la importancia de una preparación adecuada de los datos, que incluyó normalización y selección de características mediante Eliminación Recursiva de Características (RFE).

Estos hallazgos refuerzan la importancia de los modelos de Machine Learning como herramientas para la detección y gestión de amenazas cibernéticas. La implementación de Random Forest en sistemas SIEM puede proporcionar un soporte para identificar patrones anómalos y reducir el impacto de las amenazas, contribuyendo así a un entorno de TI más seguro y confiable. Además, la selección de características demostró ser una estrategia clave para optimizar el rendimiento de los modelos.

El principal aporte de este estudio radica en la demostración de cómo una combinación de preparación adecuada de datos y evaluación comparativa puede identificar el modelo de Machine Learning más efectivo para tareas de seguridad. La validación cruzada y el uso de métricas hicieron que los resultados sean confiables, ofreciendo una guía práctica para la implementación en sistemas de seguridad.

Se recomienda ampliar este trabajo evaluando el desempeño de los modelos en escenarios reales con registros dinámicos, incluyendo datos con ruido y cambios en los patrones de tráfico. Además, se anima a utilizar algoritmos híbridos que combinen Random Forest con métodos de aprendizaje en línea para adaptarse a nuevas amenazas en tiempo real. Por último, se sugiere evaluar la robustez de los modelos frente a ataques adversariales, garantizando su efectividad en entornos altamente hostiles.

Dado el potencial del Random Forest, se insta a las instituciones financieras y otras organizaciones a implementar este modelo como parte de sus estrategias de seguridad informática. Esto no solo mejorará la detección de amenazas, sino que también contribuirá a la protección de datos sensibles y la mitigación de riesgos en entornos digitales.

Disponibilidad de los Datos

El conjunto de datos utilizado en este estudio es gratuito y está disponible públicamente en Internet. Se puede obtener el conjunto de datos través del siguiente enlace: https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection

REFERENCIAS

Alimi, O., Ouahada, K., & Abu-Mahfouz, A. (2020). A Review of Machine Learning Approaches to Power System Security and Stability. IEEE Access, VIII, 113512-113531. doi:10.1109/ACCESS.2020.3003568

Aminanto, M., Ban, T., Isawa, R., Takahashi, T., & Inoue, D. (2020). Threat alert prioritization using isolation forest and stacked auto encoder with day-forward-chaining analysis. IEEE Access, VIII, 217977-217986.

Bagaa, M., Taleb, T., Bernabe, J., & Skarmeta, A. (2020). A machine learning security framework for IoT systems. IEEE Access, VIII, 114066-114077.

Barreno, M., Nelson, B., Joseph, A., & Tygar, J. (2010). The security of machine learning. Machine Learning., LXXXI, 121-148. doi:10.1007/s10994-010-5188-5

Castellanos, B., Cortés, C., Espitia, D., & Garzón, Y. (2020). Redes neuronales artificiales y estado del arte aplicado en la ciberseguridad. Revista Matices Tecnológicos., 12, 58-63.

Gilibets , L. (12 de Enero de 2023). IEBS. Obtenido de https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/

Hindy, H., Brosset, D., Bellekens, X., Bayne, E., & Amar, S. (2018). Improving SIEM for Critical SCADA Water Infrastructures Using Machine Learning. International Workshop on Security and Privacy Requirements Engineering. (págs. 3-19). Springer International Publishing. doi:10.1007/978-3-030-12786-2_1

Jesús, A., Ángela, C.-G., César, D., Manuel, G.-D., Jónathan, H., Adrián, I., . . . Beatriz, P. (2022). GitHub y Google Colaboratory para el desarrollo, comunicación y gestión de prácticas en los laboratorios de informática.

Kornyo, O., Asante, M., Opoku, R., Owusu-Agyemang, K., Partey, B., Baah, E., & Boadu, N. (2023). Botnet attacks classification in AMI networks with recursive feature elimination (RFE) and machine learning algorithms. Computers & Security, 135, 103456.

Li, Y., Xiong, K., Chin, T., & Hu, C. (2019). A machine learning framework for domain generation algorithm-based malware detection. IEEE Access, VII, 32765-32782.

Rincón, K. (2021). Desarrollo de un Prototipo de Software en Python con Técnicas de Machine Learning para el Análisis de Datos Astronómicos de Exoplanetas Recopilados por la NASA.

Robayo, E. (2023). Impacto de la Inteligencia Artificial en la Seguridad de la Información.

Robotko, S., Topalov, A., Nekrasov, S., Zaytsev, V., & Zaytsev, D. (2023). Machine Learning and Modeling of the Impact of Trademark Filings on GDP Growth based on Python. MoMLeT+ DS, (págs. 65-76).

Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021). Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology. Machine Learning and Knowledge Extraction, 3(2), 392-413. doi: https://doi.org/10.3390/make3020020

Suyal, M., & Goyal, P. (2022). A review on analysis of k-nearest neighbor classification machine learning algorithms based on supervised learning. International Journal of Engineering Trends and Technology, 70(7), 43-48.

thiago-gsantos03. (25 de Octubre de 2020). Google Colab: ¿qué es y cómo usarlo? (Alura Latam) Recuperado el 1 de Mayo de 2024, de https://www.aluracursos.com/blog/google-colab-que-es-y-como-usarlo

Waskle, S., Parashar, L., & Singh, U. (July de 2020). Intrusion detection system using PCA with random forest approach. 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)., 803-808.

Xin, Y., Kong, L., Chen, Y., Li, Y., Zhu, H., Gao, M., . . . Wang, C. (2018). Machine learning and deep learning methods for cybersecurity. IEEE Access, VI, 35365-35381. doi:10.1109/ACCESS.2018.2836950

Xue, M., Yuan, C., Wu, H., Zhang, Y., & Liu, W. (2020). Machine Learning Security: Threats, Countermeasures, and Evaluations. IEEE Access, 8, 74720-74742. doi:10.1109/ACCESS.2020.2987435

Yang, X., Peng, G., Zhang, D., & Lv, Y. (2022). An enhanced intrusion detection system for IoT networks based on deep learning and knowledge graph. Security and Communication Networks, 2022(1), 4748528.

Yavanoglu, O., & Aydos, M. (2017). A Review on Cyber Security Datasets for Machine Learning Algorithms. 2017 IEEE International Conference on Big Data (Big Data) (págs. 2186-2193). IEEE. doi:10.1109/BigData.2017.8258167

Yépez, E., & Armijos, K. (2020). Aplicación de la metodología kanban en el desarrollo del software para generación, validación y actualización de reactivos, integrado al sistema informático de control académico UNACH. (Titulación). UNIVERSIDAD NACIONAL DE CHIMBORAZO, Riobamba, Ecuador. Obtenido de http://dspace.unach.edu.ec/handle/51000/6457

Zhang, C., Wang, W., Liu, L., Ren, J., & Wang, L. (2022). Three-branch random forest intrusion detection model. Mathematics, X(23), 4460.

Zhong, C., Lin, T., Liu, P., Yen, J., & Chen, K. (2018). A cyber security data triage operation retrieval system. Computers & Security, LXXVI, 12-31. doi:10.1016/j.cose.2018.02.011