UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

Técnicas y Algoritmos de Aprendizaje Automático Aplicados a la Detección de Intrusos en Redes de Comunicaciones

Gabriel Sebastian Lara Hidalgo

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero en Ciencias de la Computación

Quito, 17 de mayo de 2024

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

Técnicas y Algoritmos de Aprendizaje Automático Aplicados a la Detección de Intrusos en Redes de Comunicaciones

Gabriel Sebastian Lara Hidalgo

Nombre del profesor, Título académico **Ricardo Flores Moyano, PhD**

Quito, 17 de mayo de 2024

© DERECHOS DE AUTOR

Nombres y apellidos: Gabriel Sebastian Lara Hidalgo

Código: 00215784

Cédula de identidad: 1751984756

Lugar y fecha: Quito, 17 de mayo de 2024

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al.. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

La seguridad en las redes de comunicaciones es un tema de gran importancia en la era digital, especialmente con el aumento de la frecuencia de los ciberataques. Este trabajo se enfoca en el uso de aprendizaje automático para de mejorar la identificación y clasificación de amenazas en las redes de comunicaciones. A través de un estudio exhaustivo del estado del arte, se identificaron áreas de mejora y se propuso un enfoque que integra un sistema basado en reglas, y por otro lado el entrenamiento de modelos de Random Forest, XGBoost, SVM, KNN, los cuales se evaluaron de manera independiente utilizando validación cruzada. Se optó por esta combinación de algoritmos debido a su versatilidad y los resultados diversos que han demostrado ser eficientes. Esta variedad de herramientas permite una adaptación flexible a una amplia gama de situaciones y requisitos. El conjunto de datos UNSW_NB15 fue preparado para el entrenamiento y prueba de los modelos, con el objetivo de evaluar su efectividad mediante métricas como precision, accuracy, F1-Score y AUC. Se compara los resultados obtenidos con estudios previos de diversos autores. Esta comparación destaca la relevancia de la validación cruzada, para fortalecer la fiabilidad de los resultados. De la misma manera se implementa una prueba de Wilcoxon para verificar significancias estadísticas confirmando que las variaciones en el rendimiento no son producto del azar en el rendimiento entre modelos.

Palabras clave: Aprendizaje Automático, Evaluación de Modelos, Seguridad en Redes, Random Forest, XGBoost, SVM, KNN, UNSW_NB15, Prueba de Wilcoxon.

ABSTRACT

Security in communication networks is a topic of great importance in the digital age, especially with the increase in the frequency of cyberattacks. This work focuses on the use of machine learning to improve the identification and classification of threats in communication networks. Through an exhaustive study of the state of the art, areas for improvement were identified and an approach was proposed that integrates a rule-based system, and on the other hand the training of Random Forest, XGBoost, SVM, KNN models, which were evaluated independently using cross-validation. This combination of algorithms was chosen due to their versatility and the diverse results they have proven to be efficient. This variety of tools allows for flexible adaptation to a wide range of situations and requirements. The UNSW_NB15 dataset was prepared for the training and testing of the models, with the aim of evaluating their effectiveness using metrics such as precision, accuracy, F1-Score, and AUC. The results obtained are compared with previous studies by various authors. This comparison highlights the relevance of cross-validation, to strengthen the reliability of the results. In the same way, a Wilcoxon test is implemented to verify statistical significances confirming that variations in performance are not the product of chance in performance between models.

Keywords: Machine Learning, Model Evaluation, Network Security, Random Forest, XGBoost, SVM, KNN, UNSW_NB15, Wilcoxon Test.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la era digital, la seguridad en las redes de comunicaciones es un tema de gran importancia. Con el uso masivo de servicios telemáticos, que se basan en la infraestructura de internet, los ciberataques también han aumentado. Para abordar este desafío, se requieren soluciones que puedan identificar de la manera más eficaz intrusos o ataques y permitir clasificarlos de manera correcta. En este contexto, este proyecto plantea la aplicación de técnicas de aprendizaje automático para crear un sistema capaz de detectar amenazas en un dataset ampliamente conocido en el mundo de la ciberseguridad. Esto con el objetivo de tener un primer acercamiento a lo que es un sistema de detección de intrusos (IDS).

El conjunto de datos UNSW_NB15[11-15], utilizado en este estudio, fue seleccionado por su relevancia y representatividad en el ámbito de la seguridad en redes. Este dataset fue creado en el Australian Centre for Cyber Security (ACCS)[1] y simula tráfico de red moderno que incluye una variedad de intrusos y ataques, lo que lo convierte en un recurso ideal para el entrenamiento y evaluación de modelos de aprendizaje automático. El proyecto combina una investigación exhaustiva del estado del arte con la experimentación de algoritmos como Random Forest, XGBoost, SVM y KNN. Estos algoritmos se evaluaron de manera independiente y a través de técnicas de validación cruzada, con el objetivo de establecer un estándar de efectividad. Además, la implementación de la prueba de Wilcoxon asegura que las mejoras en el rendimiento de los modelos son estadísticamente significativas, destacando que las variaciones en la eficacia no son producto del azar, sino de mejoras reales en las técnicas de detección y clasificación.

[1] https://www.asd.gov.au/about/what-we-do/cyber-security

ANTECEDENTES Y CONTEXTO

La detección de intrusiones en redes de comunicaciones ha avanzado considerablemente con la integración de técnicas de aprendizaje automático, las cuales han facilitado el desarrollo de sistemas de detección más sofisticados y eficaces. Una ventaja clave de estas técnicas es su habilidad para adaptarse y aprender de manera continua, permitiendo a los sistemas basados en aprendizaje automático identificar ataques nuevos y prever futuras intrusiones. Esto es crucial en un entorno donde las amenazas evolucionan rápidamente.

No obstante, la implementación de estas tecnologías enfrenta desafíos significativos. La interpretación de los resultados puede ser compleja, ya que, aunque los algoritmos pueden identificar y clasificar amenazas eficazmente, comprender el proceso detrás de estas decisiones es difícil. Adicionalmente, estos algoritmos requieren acceso a conjuntos de datos extensos y de alta calidad para entrenarse adecuadamente, siendo la calidad de estos datos fundamental para el rendimiento del sistema. Otro desafío es lograr un equilibrio entre la detección efectiva de ataques y la minimización de falsos positivos, ya que un sistema que genera muchas alertas erróneas puede ser tan problemático como uno que no detecta ataques reales con un porcentaje adecuado.

En este contexto, se propone realizar un estudio exhaustivo de la literatura reciente en este ámbito, analizando cómo diversos autores han utilizado algoritmos de aprendizaje automático para la detección de intrusos. Este análisis ayudará a identificar los desafíos actuales y explorar oportunidades para aumentar la eficacia de los sistemas de detección de intrusiones. Posteriormente se aplicará un enfoque de detección de intrusos basado en reglas, y se realizará la experimentación de algoritmos de Random Forest, XGBoost, SVM y KNN. Se evaluará el rendimiento de los algoritmos de forma independiente y a través de técnicas de validación cruzada. Además, se realizará una prueba Estadística de Wilcoxon para asegurar que el rendimiento del modelo sea estadísticamente significativo, y corroborar que la eficiencia y los resultados no son producto del azar.

ESTADO DEL ARTE

En el trabajo de los autores, Talukder et al.[1] usan modelos híbridos de Machine Learning y Deep Learning, mientras que Ripan, R.C. et al.[2] utilizan Isolation Forest Learning para tráfico anómalo. Yang, K. et al. [3] mejora la eficiencia en IoT con One-Class SVM y Athul, K. y John, A. [4] aplican transformadores de NLP para precisión con baja complejidad.

Song, Y. et al. [5] exploran Autoencoders, enfatizando la importancia del tamaño latente, y Alshammari, A. y Aldribi, A. [6] detectan tráfico malicioso en la nube con técnicas de aprendizaje automático. Hagar, A. y Gawali, B. [7] analizan algoritmos para seguridad de red, mientras Sarhan, M. et al. [8] investigan la extracción de características en IoT. Tait, K.-A. et al. [9] evalúan técnicas para clasificación multiclase y Leevy, J.L. y Khoshgoftaar, T.M [10] discuten modelos de detección de intrusiones destacando el desequilibrio de clases. A continuación, se presenta una tabla comparativa que permite identificar objetivos, metodologías, resultados y limitaciones de las diferentes propuestas analizadas.

Tabla 1. Resumen de los Estudios del Estado de Arte en IDS

Autores	Objetivos	Metodología	Resultados	Limitaciones
Talukder et al. [1]	Modelo híbrido de ML y DL para detectar intrusiones.	ML y DL combinados.	Precisión del 99.99% en detección de intrusiones.	Restricciones en datos y complejidad computacional.
Ripan, R.C. et al. [2]	Mejorar precisión en detección con Isolation Forest.	Empleo de Isolation Forest.	Mejora en precisión.	Falta de mejora en SVM.
Yang, K., Kpotufe, S., y Feamster, N. [3]	Crear un modelo One-Class SVM para detectar anomalías en IoT.	Utilización de One-Class SVM.	Detección rápida y precisa de anomalías.	Variabilidad en rendimiento según parámetros.
Athul, K. y John, A. [4]	Transformers mejorar detección de intrusiones con alta precisión.	Empleo de algoritmos BERT y GPT-2 del NLP.	Alta precisión con reducción de dimensiones.	Restricciones en escalabilidad y necesidad de explorar más.
Song, Y., Hyun, S., y Cheong, Y.-G. [5]	Uso de Autoencoders	Experimentos con Autoencoders.	Variabilidad significativa en rendimiento.	Necesidad de explorar más parámetros.
Alshammari, A. y Aldribi, A. [6]	ML para detectar intrusiones en Cloud Computing.	Crear un modelo con el aprendizaje supervisado.	Identificación de características clave.	Necesidad de mejorar calidad de datos.

Hagar, A. y Gawali, B. [7]	Análisis de algoritmos para la detección de intrusiones.	Selección de características y modelos RF y MLP.	Altos valores de precisión.	Falta limpieza de datos, desequilibrio de clases.
Sarhan, M. et al. [8]	Selección de características para detectar intrusiones	Modelos de ML y evaluación de algoritmos.	Mayor precisión con Random Forest.	Necesidad de explorar más parámetros.
Tait, K.-A. et al. [9]	Comparar técnicas de ML para la detección de intrusiones.	Experimentos con diferentes algoritmos de ML.	Mayor consistencia en precisión binaria.	Falta de análisis estadístico y desequilibrio de clases.
Leevy, J.L. y Khoshgoftaar, T.M [10]	Análisis de modelos de detección de intrusiones.	Empleo de métodos de análisis comparativo.	Altos puntajes en rendimiento.	Necesidad de mejorar calidad de datos y realizar análisis estadístico.

ANÁLISIS E IDENTIFICACIÓN DE ÁREAS DE MEJORA

Tabla 2. Áreas de Investigación en Sistemas de Detección de Intrusos

Tema	**Descripción**
Integración de Técnicas Avanzadas NLP y Aprendizaje Profundo	Efectividad de modelos NLP como BERT y GPT-2 en la detección de intrusiones.
Optimización de Hiperparámetros en Modelos de Autoencoder	Optimización de hiperparámetros y mejorar precisión y la eficiencia computacional.
Desarrollo de Modelos Adaptables y Escalables	Adaptabilidad continua con Machine Learning para una detección en tiempo real.
Exploración de Nuevas Características	Innovar en la selección de características puede revelar patrones de ataque ocultos.
Evaluación en Entornos de Red Complejos	Validación en condiciones reales para ajustar los modelos y mejorar su rendimiento.
Reducción de Falsos Positivos y Mejora de Interpretación	Estrategias para equilibrar precisión y minimización de alertas erróneas.
Desarrollo de Técnicas para Manejar el Desequilibrio de Clases	Estrategias de balanceo de clases mejoran el rendimiento en clases desequilibradas.
Evaluación de Técnicas de Reducción de Dimensionalidad	Evaluar técnicas para encontrar equilibrio óptimo entre preservación de información y rendimiento computacional.
Investigación de Técnicas Basadas en Secuencias Temporales	Modelos RNN o Transformers para mejorar la detección de intrusiones.

Tras el estudio realizado, determinamos que existen avances significativos, aplicando aprendizaje automático y profundo. Sin embargo, aún existen desafíos importantes como la adaptabilidad de los sistemas ante nuevas amenazas, de la misma manera que se busca la reducción de falsos positivos y una efectiva interpretación de los resultados.

Tecnologías como reducción de dimensionalidad con Autoencoders, redes neuronales convolucionales y el procesamiento de lenguaje natural han mejorado en términos de precisión la detección y clasificación de ataques cibernéticos. La diferenciación entre tráfico normal y tráfico de ataque aún requiere de optimización para evitar un exceso de falsas alertas. La escalabilidad y adaptabilidad también son muy importantes, considerando un entorno de red que está creciendo, lo que requiere que estos sistemas y modelos de detección sean capaces de ajustarse a volúmenes variables de datos como nuevos tipos de ataques.

DESCRIPCIÓN DE LA PROPUESTA

Este proyecto desarrollará un sistema de detección de intrusiones centrado en el análisis de tráfico de red utilizando un enfoque basado en reglas y evaluación de varios algoritmos de detección de anomalías. Iniciaremos con análisis exhaustivo del dataset UNSW_NB15[11-15], donde se implementarán procesos de limpieza de datos, manejo de valores nulos, eliminación de columnas altamente correlacionadas y normalización. Utilizaremos árboles de decisión para la derivación de reglas y optimización de hiperparámetros con GridSearchCV, enfocándonos en mejorar la métrica de Recall.

Las reglas extraídas serán aplicadas al entrenamiento de un modelo de Random Forest y obtener un subconjunto de datos, optimizando así el entrenamiento al enfocarse en patrones más representativos o críticos. Además, exploraremos algoritmos como XGBoost, SVM, K-nearest neighbors y Random Forest mediante validación cruzada para evaluar su eficacia. Las métricas de rendimiento como Precision, Recall, F1-Score y AUC se utilizarán para evaluar cada enfoque. También compararemos nuestros resultados con los de estudios anteriores en el campo de la detección de intrusiones para contextualizar nuestros hallazgos y establecer la relevancia y contribución de nuestro trabajo al estado del arte.

Concluiremos con un análisis detallado de los resultados, utilizando la prueba de Wilcoxon para determinar que las diferencias observadas en los resultados entre los modelos no son producto del azar.

Flujo de desarrollo del trabajo

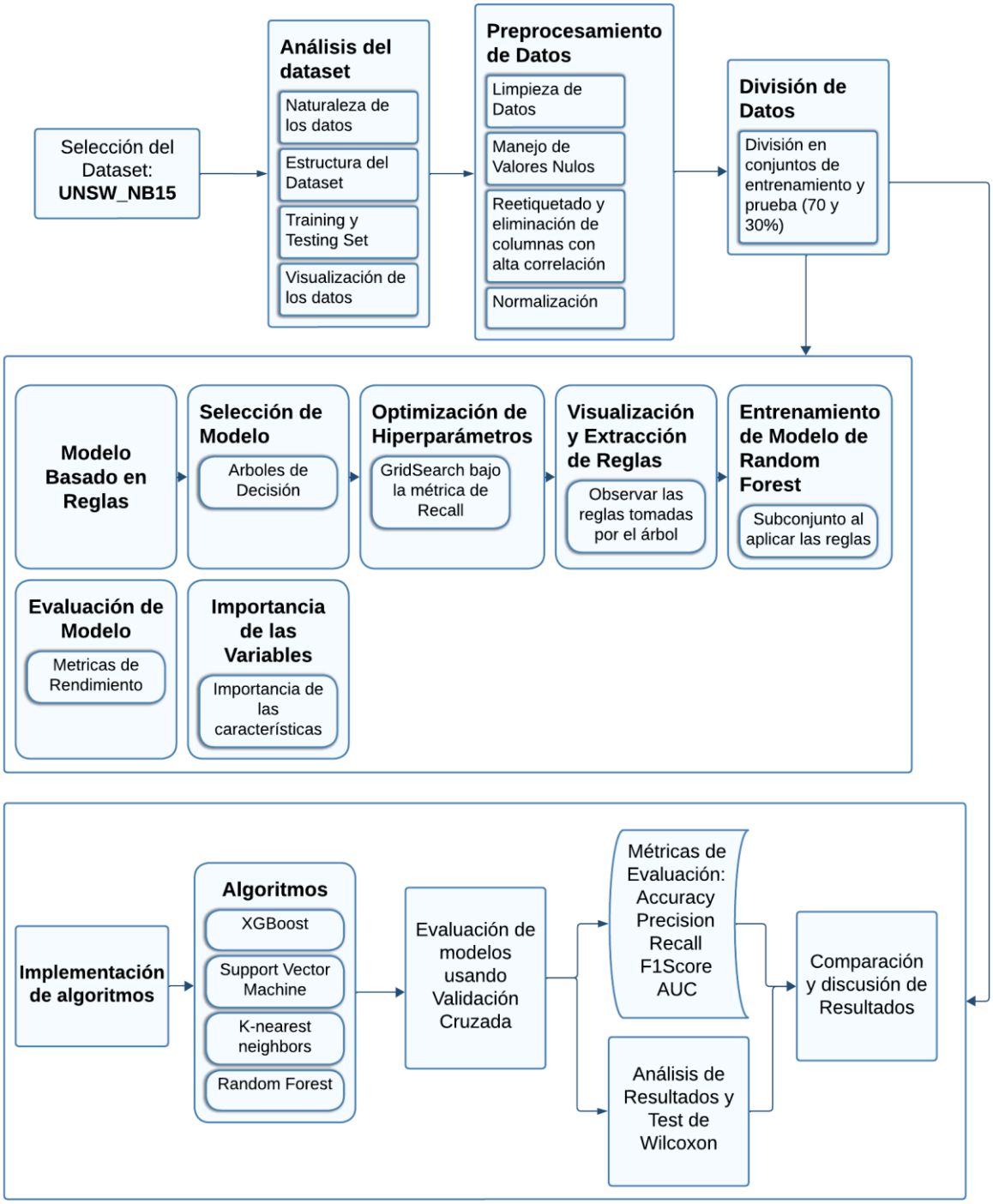

Figura 1. Flujo de trabajo propuesto para el proyecto

DESARROLLO DEL TEMA

Para el desarrollo del proyecto se utiliza la plataforma Google Colaboratory, facilitando la ejecución de código. De igual manera se creó a un repositorio de GitHub para gestionar y controlar las versiones durante el desarrollo del proyecto. El dataset utilizado es UNSW-NB15[11 – 15], y se cargó al repositorio de GitHub[2] para acceder en Google Colaboratory.

Dataset UNSW_NB15

Creado en 2015 por Nour Moustafa y Jill Slay, el dataset UNSW_NB15 [11-15] fue creado para abordar las limitaciones de los conjuntos de datos de referencia existentes como KDD98, KDDCUP99 y NSLKDD, que no reflejan de manera inclusiva el tráfico de red moderno y los ataques de bajo impacto. Los autores utilizaron la herramienta IXIA PerfectStorm en el Cyber Range Lab de Australia. El conjunto de datos UNSW-NB15 se generó utilizando un entorno de especialmente configurado, detallado en la Figura 2. El núcleo de este entorno es el generador de tráfico IXIA que recibía información del sitio web CVE para la obtención de trafico de tipo ataque, la arquitectura compuesta por tres servidores virtuales, donde los servidores 1 y 3 estaban configurados para distribuir tráfico normal, mientras que el servidor 2 se utilizó para generar actividades maliciosas dentro del tráfico de la red. Para capturar el tráfico de red tanto público como privado y permitir la intercomunicación entre los servidores, se configuraron dos interfaces virtuales con direcciones IP. Estos servidores se conectaron a los hosts a través de dos Routers. Todos los Routers estaban conectados a un dispositivo de firewall configurado para permitir el paso de todo el tráfico, tanto normal como anormal. La herramienta tcpdump se instaló en el Router 1 para capturar los archivos Pcap durante el tiempo de simulación de 16 horas generando 100gb de datos.

[2] Enlace al repositorio: https://github.com/GaboLara998/ProyectoIntegrador

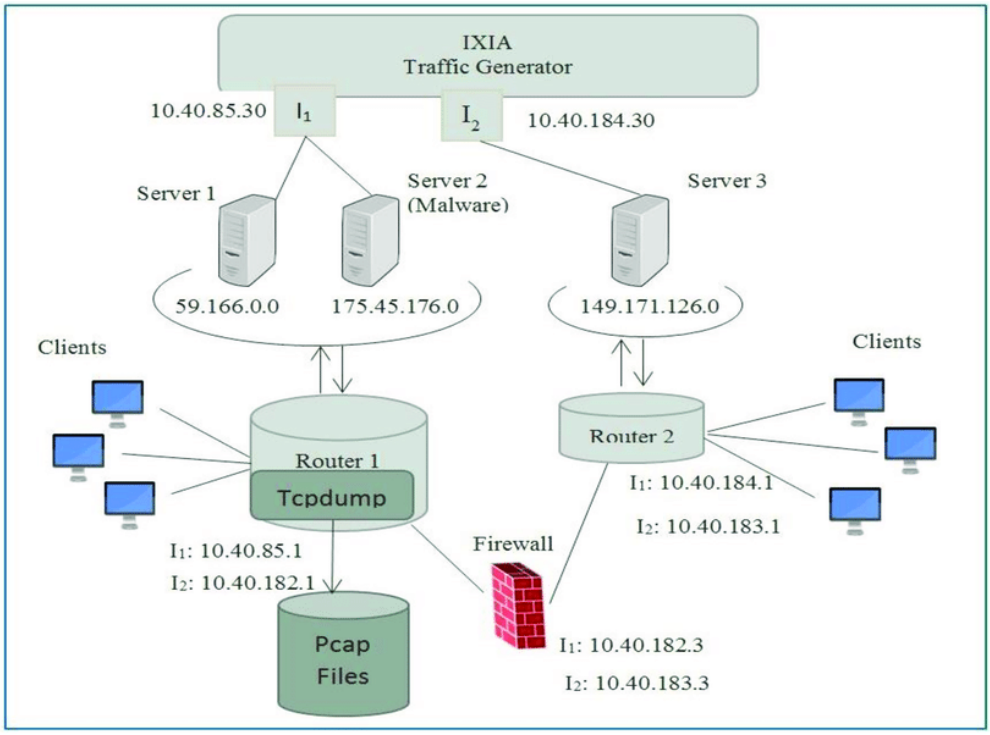

Figura 2. Configuración del banco de pruebas con servidores y clientes de UNSW_NB15

Algoritmos y resultados de los estudios relacionados

Como se mencionó en la propuesta de proyecto, se tomará como línea base de referencia, los algoritmos y los resultados obtenidos en distintos estudios enfocados en la detección de intrusos. A continuación, se detalla estos algoritmos a utilizar, junto con los autores y los resultados obtenidos aplicados a varios datasets, donde se midió Accuracy(AC)[20], Precision(PR) [21], Recall(RC) [22], F1-Score (F1) [23].

Tabla 3. Resultados de algoritmos de machine Learning de varios estudios enfocados en IDS

Algoritmos	**Autores**	**DataSet**	**Resultados**			
Random Forest	Abdullah Al-Obaidi et al. [11]	UNSW-NB15	AC 98.5	PR 98.8	RC 98.8	F1 98.8
KNN	Kochet & Kumar[12]	UNSW-NB15	AC 98.28	PR 99	RC 99	F1 99
XGBoost	Neha Sharma et al. [13]	UNSW-NB15	AC 84.9	PR 97	RC 68	F1 80
SVM	Shweta More et al. [14]	UNSW-NB15	AC 98.7	PR 98.6	RC 99.9	F1 99.3

El algoritmo de Random Forest, Abdullah Al-Obaidi et al.[11] lo implementan y entrenan usando el dataset UNSW_NB15 [15 - 19] en un contexto de clasificación binaria. En cuanto al algoritmo de KNN, propuesto por Kochet & Kumar[12], realizaron una comparación de rendimiento de varios clasificadores de aprendizaje automático, donde obtuvieron los valores anteriormente mencionados para KNN. Por otro lado, Neha Sharma et al.[13] implementaron un modelo de XGBoost para la misma clasificación binaria del dataset. Con el algoritmo de SVM, Shweta More et al.[14] también obtuvieron resultados positivos en términos de clasificación binaria. En todos los casos, los autores resaltan la importancia del preprocesamiento de datos, transformación para evitar sesgos, normalización, y la reducción de datos para enfocarse en características con alta correlación.

Proceso de análisis de dataset

El dataset UNSW-NB15 [15 – 19], incluye registros de varios tipos de ataques cibernéticos. Se analizaron los diferentes tipos de archivos que lo componen, como PCAP (Packet Capture), reportes, archivos BRO-IDS, y archivos ARGUS, así como archivos .csv con datos procesados. Para el análisis, se usó dos conjuntos de datos proporcionados por los autores: UNSW_NB15_testing.csv y UNSW_NB15_training.csv, ambos etiquetados y procesados.

El conjunto de entrenamiento incluye 175,341 entradas, y el de prueba 82,332. Se seleccionaron estos archivos porque contienen información de tráfico de red con características extraídas y etiquetas de clase, facilitando el uso en el entrenamiento y prueba de modelos de aprendizaje automático. A diferencia de otros archivos, no requieren un preprocesamiento que podría llevar a errores en la extracción de características y etiquetado. El conjunto tiene 43 características incluyendo un Label binario identifica si cada registro es un ataque o tráfico normal, y una columna que clasifica los distintos tipos de ataques.

Preprocesamiento de los datos

Se analiza la distribución de tráfico normal y de ataque en el conjunto de datos, notando que la proporción de tráfico normal a tráfico de ataque es de 0.56. Esto indica que, por cada registro de tráfico de ataque, hay aproximadamente 0.56 registros de tráfico normal.

Igualmente, se investiga la cantidad de registros de diferentes tipos de ataques presentes en el conjunto de datos, se cuenta cuántos ataques de cada categoría existen en los datos. Se presenta a continuación la distribución de tráfico normal y de ataque (A) y la distribución de tipos de ataque (B) en la siguiente figura.

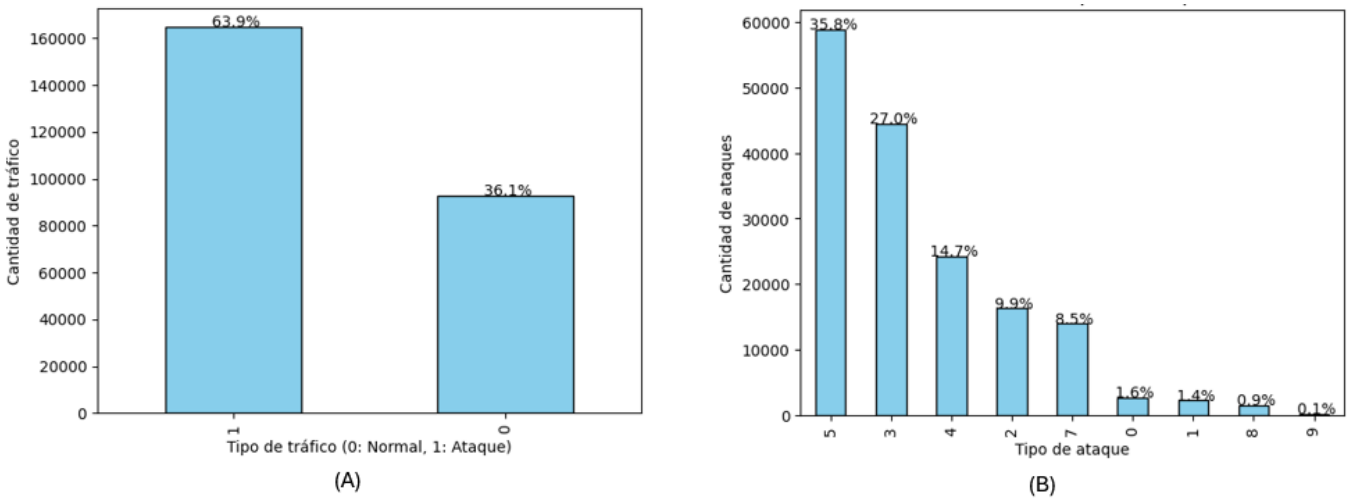

Figura 3. Gráfico de la Distribución de tráfico de red y los tipos de ataque del Dataset

Preparación de datos para el modelado

Se combina los datasets de entrenamiento y prueba. Posteriormente, se eliminan columnas como 'service', 'proto', y 'state' que contienen texto y suelen tener valores nulos. Las categorías de ataques se transforman en códigos numéricos para optimizar el procesamiento algorítmico. También se normalizan las características para estandarizar las escalas y se eliminan aquellas con alta correlación para evitar redundancias y fomentar una mejor generalización del modelo. Finalmente, se segregan las características de las etiquetas y los datos se dividen en conjuntos de entrenamiento y prueba.

Detección de intrusos basado en reglas

Se utiliza un modelo de árbol de decisión para definir un conjunto de criterios que facilitan la detección de ataques cibernéticos. El objetivo es alcanzar una alta tasa de recuperación

(recall), una métrica que evalúa la capacidad del modelo para identificar correctamente los casos reales de ataque. Por tanto, se optimiza el modelo para maximizar este parámetro.

Se define un conjunto de parámetros para el modelo de árbol de decisión, que incluye el criterio (gini o entropía), la profundidad máxima del árbol, el número mínimo de muestras para dividir un nodo interno y el número mínimo de muestras necesarias para un nodo hoja.

Se crea un clasificador de árbol de decisión y se utiliza GridSearchCV para optimizar los parámetros. Este método prueba diversas combinaciones de parámetros y selecciona la mejor basada en la puntuación de recall. Después se identifican los parámetros que maximiza el recall. Con esa información construimos reglas específicas que luego aplicamos al conjunto de datos de prueba, con el objetivo de evaluar el rendimiento del modelo en datos no vistos. Este enfoque ha llevado al modelo a clasificar el 23% del tráfico de red como atípico, indicando valores sospechosos en variables clave que sugieren actividad maliciosa.

El árbol de decisión comienza su clasificación utilizando 'sttl' (Time to Live del paquete) como nodo raíz, con un umbral de sttl <= 0.239 para la primera división de los datos. Este umbral se basa en el análisis estadístico de los datos de entrenamiento, que vinculan los valores bajos de 'sttl' con actividades maliciosas, sugiriendo que los paquetes están diseñados para no viajar lejos antes de expirar, una táctica común en ciertos ataques para evitar el rastreo. El proceso de optimización del modelo selecciona el punto de corte de 0.239, que maximiza la separación entre las clases "Normal" y "Ataque", usando el índice Gini para garantizar la pureza de los nodos y optimizar la precisión en la detección de ataques. A medida que el árbol se expande, integra 'ct_dst_sport_ltm' y 'synack', elegidas por su relevancia estadística y capacidad para ofrecer más detalles en la diferenciación entre comportamientos normales y maliciosos. El umbral de <= 0.033 para 'ct_dst_sport_ltm, utilizado para identificar tráfico potencialmente anómalo como en los ataques DDoS, permite al modelo detectar patrones de paquetes rápidos y atípicos, dependiendo de otras condiciones evaluadas. Este umbral se determina evaluando múltiples puntos de corte durante el entrenamiento, buscando aquel que mejore significativamente la clasificación de las actividades de red. Por otro lado, 'synack' con un umbral de <= 0.012, es crucial para identificar respuestas automatizadas o preprogramadas en escaneos de puertos y otros ataques

automatizados. Este valor se selecciona siguiendo un análisis detallado que busca maximizar la eficacia en la separación de tráfico malicioso del normal, haciendo del modelo una herramienta robusta y precisa para la detección de intrusiones.

Cada nodo en el árbol muestra 'samples', que indica el número de muestras que pasan por ese nodo, y 'value', que muestra la distribución de estas muestras entre las clases de "Normal" y "Ataque". Por ejemplo, un nodo con 'samples = 180371' y 'value = [65310, 115240]' implica que 180,371 muestras fueron evaluadas en ese nodo, de las cuales 65,310 fueron clasificadas como "Normal" y 115,240 como "Ataque". En el caso de 'synack', aunque ambas bifurcaciones resultantes se clasifican como "Ataque", el algoritmo puede haber encontrado que hacer una bifurcación adicional reduce la impureza interna dentro del grupo "Ataque", separando diferentes intensidades o tácticas de ataques, lo que ayuda en la gestión y respuesta a estos ataques de manera más efectiva.

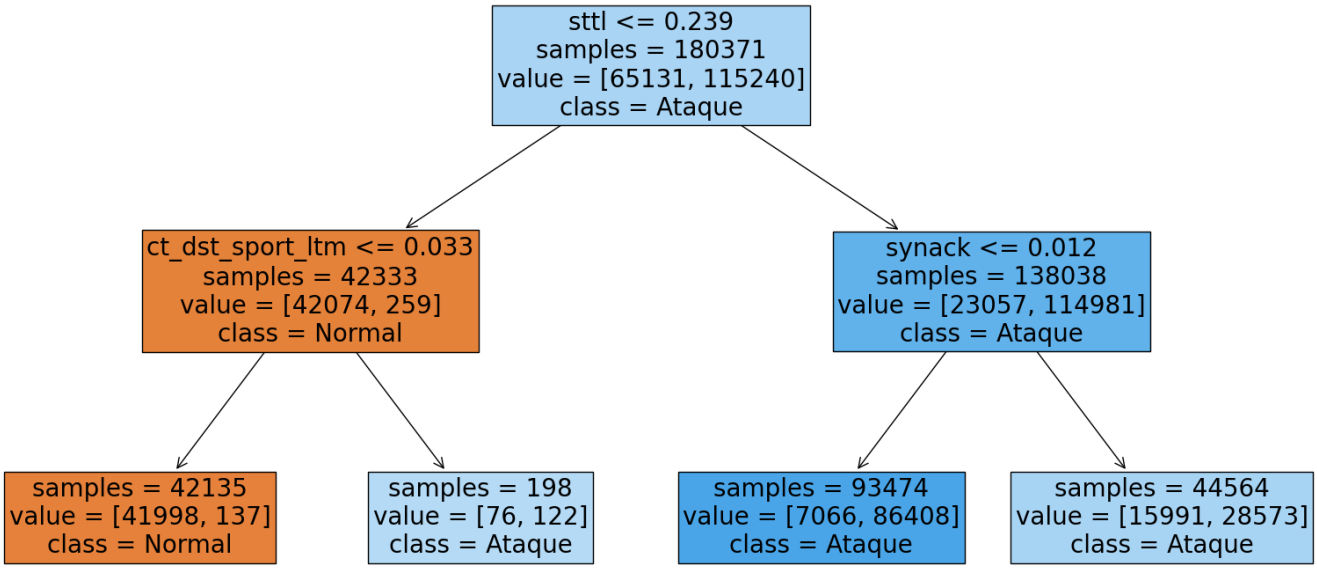

Figura 4. Visualización de la Clasificación de Clases en el Árbol de Decisión

Modelo de Random Forest para detección de intrusos basado en reglas

Utilizando el algoritmo de Random Forest se hace las predicciones en los datos filtrados bajo las reglas establecidas las cuales y evaluamos su rendimiento, donde se obtuvo lo siguiente:

Tabla 4. Métricas de Evaluación del Modelo de Random Forest Basado en Reglas.

Métrica	Valor (%)
Recall	95.9
Precision	96.4
Accuracy	95.1
F1 Score	96.2
AUC	89.17

El modelo tiene un buen rendimiento en la detección de intrusiones, con un recall del 96%, indica que puede identificar correctamente de los ataques reales, y que las identificaciones de ataques realizadas son correctas. Con un Accuracy del 95%, el modelo tiene un alto porcentaje de predicciones correctas, y un AUC de 0.947 muestra su capacidad para distinguir entre el tráfico normal y el tráfico de ataque. La matriz de confusión revela que el modelo identificó correctamente 47,436 ataques y 25,022 casos normales. No obstante, se registraron 1,732 falsos positivos y 1,997 falsos negativos.

Identificamos las características más influyentes en el modelo, evaluamos su importancia basada en la mejora del rendimiento y la reducción de impureza que aportan a los árboles del bosque. Las características más significativas son sttl, que indica "Tiempo de vida de origen a destino", y otras relevantes como ct_state_ttl y ct_dst_src_ltm, que están relacionadas con las conexiones TCP. Además, dload, sload y rate son importantes para analizar la cantidad y velocidad del tráfico, reflejando su utilidad para identificar tipos de ataques.

	Nombre	Importancia
0	sttl	0.145638
1	ct_state_ttl	0.106553
2	dload	0.050570
3	sload	0.049528
4	rate	0.048202
5	smean	0.040595
6	sbytes	0.040532
7	ct_dst_src_ltm	0.038633
8	dttl	0.038604
9	ct_srv_dst	0.035995

Figura 5. Las 10 características más importantes según su valor de importancia

A continuación, se muestran los algoritmos considerados para el desarrollo de este proyecto, junto con sus resultados y la interpretación de estos.

Algoritmo de SVM:

El SVM es un modelo de aprendizaje supervisado que busca el mejor hiperplano para diferenciar dos clases, siendo ideal para conjuntos de datos complejos. Su eficacia incluye la capacidad para gestionar datos no lineales mediante el "kernel", lo que lo hace valioso para la clasificación binaria, como discernir entre datos normales y ataques.

Algoritmo de Random Forest:

A diferencia del modelo previamente descrito, que usaba el conjunto de datos filtrados por el conjunto de reglas establecidas, este nuevo enfoque de Random Forest usa los datos de entrenamiento.

Algoritmo de K-Nearest Neighbors (KNN):

Clasifica instancias en función de las etiquetas de sus k vecinos más cercanos. Por lo que se entrenó nuestro modelo con nuestros datos de entrenamiento utilizando un valor de 3 para el número de vecinos.

Algoritmo XGBoost

Es un algoritmo de boosting secuencial que combina múltiples arboles de decisión para mejorar la precisión del modelo, donde cada nuevo modelo se enfoca en corregir los errores del modelo anterior.

Metricas de Evaluación:

Las métricas de evaluación para determinar el rendimiento de los modelos son Accuracy, Precision, Recall y F1-score.

- Accuracy: Mide cuan bien predice correctamente observaciones positivas o negativas.
- Precision: Mide la proporción de predicciones positivas que son realmente correctas.
- Recall: Capacidad del modelo para identificar todas las instancias reales positivas.
- F1-Score: Combina precision y el recall en una métrica que busca un balance.
- TP (True Positives): Casos en los que el modelo correctamente identifica un ataque real.
- TN (True Negatives): Casos en los que el modelo identifica que no hay ataque.

- FP (False Positives): Casos en él se indica erróneamente de un ataque, donde no lo hay.
- FN (False Negatives): Cuando el modelo no detectar un ataque que está ocurriendo.

Resumen de Resultados de las Métricas de Evaluación para cada Algoritmo:

Se incluye el cálculo del Accuracy y las métricas de Precision, Recall y F1-score para dos categorías: Clase 0 (tráfico normal) y Clase 1 (tráfico de ataque). Además de métricas agregadas como el Macro Average, que calcula la media aritmética de las métricas para ambas clases, y el Weighted Average, que proporciona promedios ponderados basados en la frecuencia de cada clase. Tomando en cuenta tanto el rendimiento por clase individual como el general ajustado por el tamaño de cada clase. La fila "Support" indica el número total de casos reales en el conjunto de datos para cada clase sobre el cual se calculó cada métrica.

Tabla 5. Resumen de los resultados de Evaluación de Algoritmos en Detección de intrusiones

Métrica/Modelo	**SVM**	**Random Forest**	**K-Nearest Neighbors**	**XGBoost**
Accuracy	0.8936	0.9512	0.9151	0.9467
		Clase 0		
Precision	0.99	0.93	0.88	0.92
Recall	0.71	0.94	0.89	0.93
F1-Score	0.83	0.93	0.88	0.93
Support	27,869	27,869	27,869	27,869
		Clase 1		
Precision	0.86	0.96	0.94	0.96
Recall	1.00	0.96	0.93	0.95
F1-Score	0.92	0.96	0.93	0.96
Support	49,433	49,433	49,433	49,433
		Macro Average		
Precision	0.93	0.95	0.91	0.94
Recall	0.85	0.95	0.91	0.94
F1-Score	0.88	0.95	0.91	0.94
		Weighted Average		
Precision	0.91	0.95	0.92	0.95
Recall	0.89	0.95	0.92	0.95
F1-Score	0.89	0.95	0.92	0.95

Los resultados muestran que el modelo de Random Forest tiene el mejor desempeño en términos de accuracy, seguido por XGBoost, ambos manteniendo un buen equilibrio entre Precision y Recall para ambas clases. En contraste, el modelo SVM, a pesar de su alta Precision para la Clase 0 con un 0.99, tiene un recall de solo 0.71, indicando que, aunque sus predicciones para la clase normal son muy precisas, falla en identificar una proporción significativa de casos normales. Por otro lado, K-Nearest Neighbors y Label Spreading también muestran un desempeño competente con accuracies superiores al 91%. En cuanto a la evaluación ponderada, que ajusta las métricas según el soporte de cada clase, Random Forest y XGBoost destacan con puntuaciones cercanas al 0.95 en precision, recall y F1-Score, demostrando su capacidad para manejar eficientemente el desequilibrio de clases.

Análisis y comparación de los modelos y resultados de diversos estudios.

Tabla 6. Comparación entre los resultados de modelos desarrollados y los modelos de varios autores.

Algoritmos	**Nuestros Resultados**				**Resultados de Autores**			
Random Forest [11]	AC	PR	RC	F1	AC	PR	RC	F1
	95.12	95	95	95	98.5	98	98.8	98.8
KNN [12]	AC	PR	RC	F1	AC	PR	RC	F1
	91.36	93.44	93.00	93.22	98.28	99	99	99
XGBoost [13]	AC	PR	RC	F1	AC	PR	RC	F1
	95	95	95	95	84	97	68	80
SVM [14]	AC	PR	RC	F1	AC	PR	RC	F1
	89	93	89	89	98	98	99	99

Como se observa, existen variaciones significativas en el rendimiento de los modelos. Después de un análisis de las metodologías propuestas por los distintos autores, existen diferencias en el preprocesamiento de los datos, la selección de características y la configuración de los hiperparámetros. Sin embargo, se obtuvo buenos resultados en términos del algoritmo de XGBoost, con valores más altos que las métricas obtenidas por los autores.

Resumen de Resultados de las Matrices de Confusión para cada Algoritmo:

Para cada algoritmo evaluado se generó la matriz de confusión, donde la matriz (A) corresponde al modelo de Random Forest. La matriz (B) corresponde al modelo de XGBoost. La matriz (C) corresponde al modelo de KNN y la matriz (D) es del modelo de SVM.

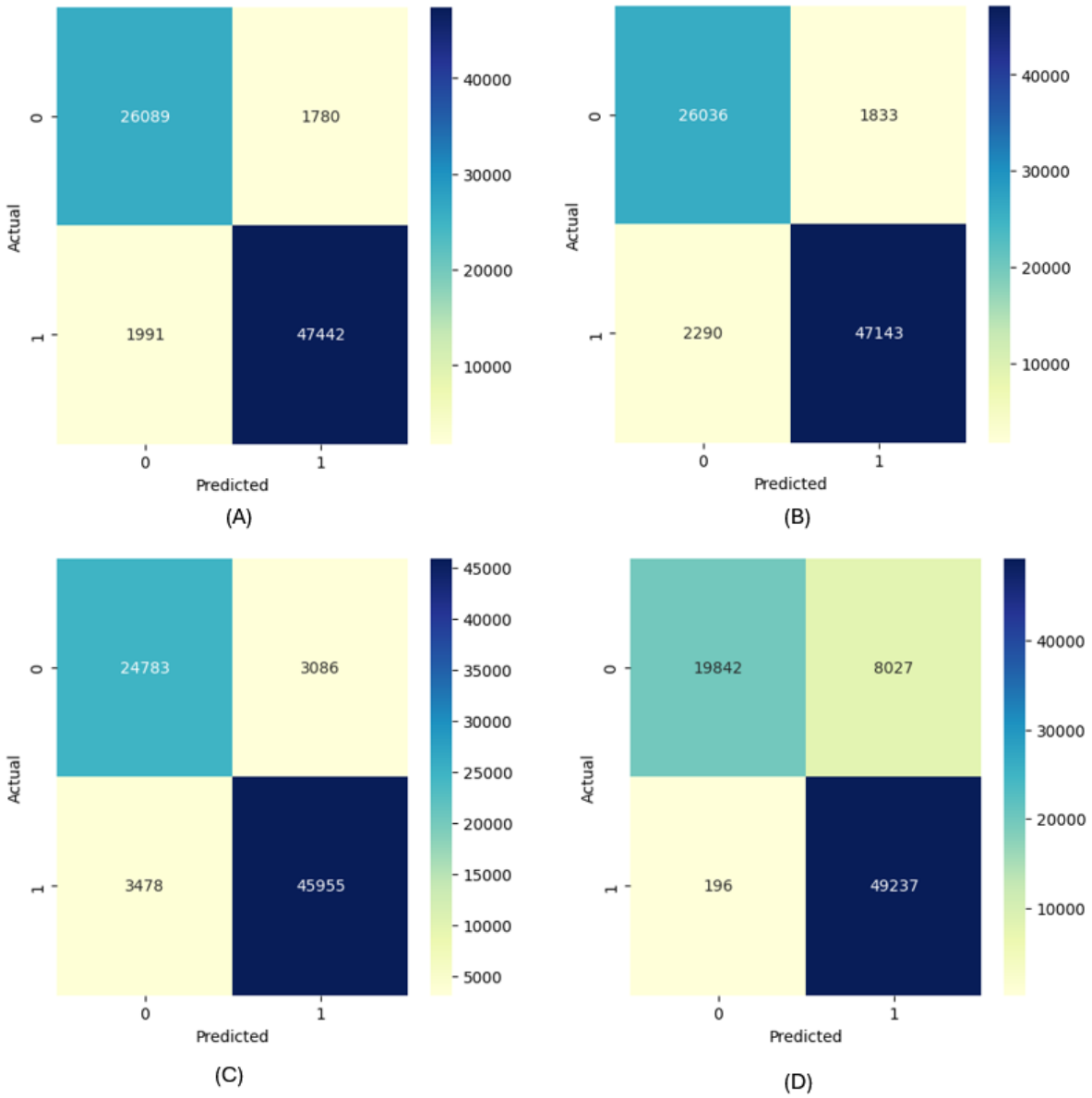

Figura 6. Resultados de matrices de confusión para los distintos algoritmos

Los modelos de Random Forest y XGBoost muestran un alto número de verdaderos positivos y negativos, esto es un buen indicador de la capacidad para identificar correctamente tanto los casos positivos como los negativos. A diferencia del modelo K-Nearest Neighbors, aunque efectivos en clasificar correctamente, registran más errores, indicando una menor precisión. Por otro lado, el modelo SVM, con una alta cantidad de verdaderos negativos y el menor

número de falsos negativos, tiende a predecir la clase negativa con alta confianza. Sin embargo, este enfoque resulta en una cantidad significativamente alta de falsos positivos.

Evaluación de Modelos Mediante Validación Cruzada Estratificada

La validación cruzada estratificada es una técnica útil en el aprendizaje automático que nos permite obtener una estimación más precisa del rendimiento de un modelo. Al dividir el conjunto de datos en múltiples subconjuntos o pliegues, podemos entrenar y probar nuestro modelo en diferentes combinaciones de estos subconjuntos. Esto nos ayuda a entender cómo nuestro modelo se comportará con datos no vistos. Y de igual manera, compararemos los resultados obtenidos para las métricas de AUC obtenidas por los distintos autores. Para el caso de los modelos de KNN y de SVM, los autores Ripan, R.C. et at. [2] evaluaron el rendimiento del modelo de detección de anomalías basado en aprendizaje y eliminación de outliders usando Isolation Forest, mientras que los autores More, S et al.[24], evaluaron a sus modelos de Random Forest y XGBoost en términos de AUC. Para nuestro estudio, utilizamos StratifiedKFold de la librería Sklearn de Python, se configuró para dividir el conjunto de datos en 10 pliegues y evaluar a cada modelo bajo las métricas de Precision, Accuracy, Recall, F1-Score y AUC, a continuación, se detallan los resultados obtenidos para AUC, la cual es una metica para evaluar la calidad y rendimiento en clasificación binaria, junto con la desviación estándar respectiva y los resultados obtenidos por varios autores.

Tabla 7. Resultados de validación cruzada para los algoritmos con mejores resultados

Modelo	AUC	Desviación Estándar	Resultados de Autores
Random Forest [24]	99.16%	0.00076	99.41%
KNN [2]	98.75%	0.0023	99.8%
XGBoost [24]	99.11%	0.0008	98.73%
SVM [2]	97.07%	0.001	99.9%

Los resultados de la validación cruzada para los algoritmos muestran que el modelo Random Forest tiene una AUC promedio del 99.16%, lo cual es bastante bueno, y se acerca bastante a los resultados obtenidos por los autores More, S et al.[24]. El modelo XGBoost tiene un AUC promedio del 99.11% mostrando una eficacia consistente en la clasificación, similarmente a

los resultados obtenidos por los autores More, S et al.[24]. Por otro lado, el modelo K-Nearest Neighbors (KNN) logró una AUC media del 98.75% refleja un rendimiento muy bueno, sin embargo, fue menor a los resultados obtenidos por los autores Ripan, R.C. et at. [2], esta diferencia puede ser debido a una selección diferente de hiperparametros, o la falta de optimización. En cambio, el modelo SVM, aunque mostraron un rendimiento muy bueno, fueron inferiores a los otros modelos evaluados y al de los obtenidos por los autores Ripan, R.C. et at.[2]. Los resultados obtenidos son muy significativos, ya que la métrica AUC proporciona una medida de la capacidad del modelo para clasificar correctamente las instancias a lo largo de todos los posibles umbrales. Por lo que un valor alto de AUC indica que el modelo tiene una buena capacidad de diferenciar entre las clases positivas y las negativas.

Test estadístico aplicados a los modelos

Prueba de Wilcoxon: Es una prueba no paramétrica que evalúa si dos muestras independientes provienen de la misma distribución. Utilizamos esta prueba para comparar los resultados de los modelos en pares a lo largo de 10 pliegues utilizando validación cruzada, lo que nos permite evaluar si existen diferencias significativas entre las distribuciones de los rendimientos de dos modelos.

Tabla 8. Resultados de la prueba de Wilcoxon para cada resultado de algoritmo

Comparación	Valor p
Random Forest vs XGBoost	0.00195
Random Forest vs KNN	0.00195
XGBoost vs KNN	0.00195
SVM vs XGBoost	0.00195
SVM vs KNN	0.00195
SVM vs Random Forest	0.00195

En los resultados de la prueba de Wilcoxon, observamos valores p uniformemente bajos, A pesar de que estos valores son idénticos en todas las comparaciones, la significancia estadística radica en que todos los valores son inferiores al nivel de significancia estándar de 0.05, lo cual ayuda a rechazar la hipótesis nula.

La hipótesis nula para esta prueba asumimos que no hay diferencias en el rendimiento entre los pares de modelos comparados. Los bajos valores p indican que podemos rechazar esta hipótesis nula con un alto grado de confianza y que es muy poco probable que la similitud en los rendimientos observados entre modelos sea debido al azar. Por lo tanto, estos valores p consistentemente bajos confirman que las diferencias entre los modelos no solo son estadísticamente significativas, sino también consistentes a través de diversas comparaciones entre pares de modelos.

Este patrón uniforme de valores p bajos refuerza la idea de que las diferencias observadas no son aleatorias. Más bien, sugieren que las características intrínsecas y los métodos de aprendizaje de cada modelo afectan su rendimiento de manera significativa y reproducible. Es decir, cada modelo procesa y aprende de los datos de manera diferente, lo que se refleja consistentemente en sus resultados a través de las pruebas.

CONCLUSIONES

Este proyecto me ha permitido obtener una comprensión amplia de varios algoritmos de aprendizaje automático aplicados en la detección de anomalías en redes de comunicación, explorando su funcionamiento y lógica operativa. Se evaluaron métodos de clasificación usando modelos como Random Forest, SVM, KNN y XGBoost, centrándonos en su precisión y capacidad para distinguir entre clases en problemas binarios.

El enfoque basado en reglas fue eficaz para nuestro conjunto de datos, demostrando cómo el árbol de decisión clasifica entre tráfico normal y de ataque. Sin embargo, este método enfrenta limitaciones, pues las reglas preestablecidas no se adaptan a nuevas formas de ataque.

Los resultados indicaron que Random Forest y XGBoost fueron especialmente efectivos, superando a menudo los resultados de estudios previos, mientras que SVM y KNN también dieron buenos resultados, pero no alcanzaron las métricas más altas observadas en otras investigaciones. Estas diferencias se atribuyen principalmente a variaciones en el preprocesamiento de datos y ajustes de hiperparámetros. En términos de la métrica de AUC, se obtuvieron muy buenos resultados, con rendimiento poco variado. El modelo Random Forest, con un AUC de 99.16%, se acerca al valor reportado de 99.41%, indicando un desempeño robusto. Sin embargo, los modelos KNN y SVM, con AUCs de 98.75% y 97.07% respectivamente, quedan por debajo de sus referencias, destacando la necesidad de mejorar en el preprocesamiento y la selección de características. Notablemente, XGBoost supera su referencia con un AUC de 99.11%, demostrando eficacia en nuestro estudio utilizando el dataset UNSW_NB15 [11-15].

Este dataset fue crucial para el desarrollo de este proyecto, debido a su diversidad y calidad, que incluye tanto trafico normal como ataques de red modernos, con una amplia variedad de características relevantes relacionadas al tráfico de red. Este dataset proporciona una base sólida para el entrenamiento y evaluación de modelos de aprendizaje automático, mostrando su relevancia en la detección de intrusiones. Sin dejar de lado lo importante que es el preprocesamiento del dataset, eliminando valores nulos, la exclusión de columnas no

relevantes en términos del entrenamiento de los modelos, la normalización de los datos y escalado para que todas las variables contribuyan equitativamente al rendimiento del modelo, y evitar que algunas características tengan valores en rangos muy grandes en comparación con otras, provocando que el modelo lleve a resultados sesgados.

Recomendaciones y Trabajo Futuro

Se recomienda utilizar la plataforma de Google Colaboratory[3] u otra alternativa como Kaggle Notebooks[4] para ejecutar el código, sea para analizar el dataset o para el entrenamiento de los modelos. Ya que ambas plataformas ofrecen acceso gratuito a recursos computaciones bastante robustos, facilitando el manejo de grandes volúmenes de datos y una ejecución eficiente de algoritmos complejos. Garantizando que las ejecuciones sean visibles y reproducibles para mantener la integridad del análisis.

En términos de trabajo futuro, se sugiere implementar y experimentar con métodos más avanzados para el preprocesamiento, combinando y transformando características para extraer nuevos indicadores que capturen mejor las interacciones en los datos. Además, sería beneficioso realizar una selección de características más sofisticada para optimizar los modelos eliminando datos redundantes y enfocándose en las características más relevantes. También sería importante experimentar con arquitecturas más robustas, como el aprendizaje profundo, para evaluar su eficacia en la detección de ataques y en la realización de una clasificación multiclase.

[3] https://colab.research.google.com/
[4] https://www.kaggle.com/docs/notebooks

REFERENCIAS BIBLIOGRÁFICAS

[1] Md. A. Talukder et al.., "A dependable hybrid machine learning model for network intrusion detection," Journal of Information Security and Applications, vol. 72, p. 103405, Feb. 2023, doi: 10.1016/j.jisa.2022.103405.

[2] Ripan, R.C. et at. (2021). An Isolation Forest Learning Based Outlier Detection Approach for Effectively Classifying Cyber Anomalies. In: Abraham, A., Hanne, T., Castillo, O., Gandhi, N., Nogueira Rios, T., Hong, TP. (eds) Hybrid Intelligent Systems. HIS 2020. Advances in Intelligent Systems and Computing, vol 1375. Springer, Cham. https://doi.org/10.1007/978-3-030-73050-5_27.

[3] K. Yang, S. Kpotufe, and N. Feamster, An Efficient One-Class SVM for Anomaly Detection in the Internet of Things. 2021, dio: https://arxiv.org/abs/2104.11146v1.

[4] K, Athul and John, Anita, A Deep Learning based Intrusion Detection System using Transformers (May 31, 2022). Proceedings of the International Conference on Systems, Energy and Environment 2022 (ICSEE 2022), Available at SSRN: https://ssrn.com/abstract=4294593 or http://dx.doi.org/10.2139/ssrn.4294593.

[5] Y. Song, S. Hyun, and Y.-G. Cheong, "Analysis of Autoencoders for Network Intrusion Detection," Sensors, vol. 21, no. 13, 2021, doi: 10.3390/s21134294.

[6] Alshammari, A., Aldribi, A. Apply machine learning techniques to detect malicious network traffic in cloud computing. J Big Data 8, 90 (2021). https://doi.org/10.1186/s40537-021-00475-1.

[7] Hagar and Dr. B. Gawali, "Implementation of Machine and Deep Learning Algorithms for Intrusion Detection System," 2023, pp. 1–20. doi: 10.1007/978-981-19-1844-5_1.

[8] M. Sarhan, S. Layeghy, N. Moustafa, M. Gallagher, and M. Portmann, "Feature extraction for machine learning-based intrusion detection in IoT networks," Digital Communications and Networks, Sep. 2022, doi: 10.1016/j.dcan.2022.08.012.

[9] K.-A. Tait et al.., Intrusion Detection using Machine Learning Techniques: An Experimental Comparison. 2021. https://doi.org/10.48550/arXiv.2105.13435.

[10] Leevy, J.L., Khoshgoftaar, T.M. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. J Big Data 7, 104 (2020). https://doi.org/10.1186/s40537-020-00382-x.

[11] Al-Obaidi, A., Ibrahim, A.A., Khaleel, A.M. The Effectiveness of Deploying Machine Learning Techniques in Information Security to Detect Nine Attacks: UNSW-NB15 Dataset as a Case Study. Journal of Information Security and Applications, vol. 72, p. 103405, Feb. 2023. https://doi.org/10.1016/j.jisa.2022.103405.

[12] Kocher, Geeta & Kumar Ahuja, Dr. Gulshan. (2021). Analysis of Machine Learning Algorithms with Feature Selection for Intrusion Detection using UNSW-NB15 Dataset. International Journal of Network Security & Its Applications. 13. 21-31. 10.5121/ijnsa.2021.13102.

[13] Sharma, N., Yadav, N.S., Sharma, S. Classification of UNSW-NB15 dataset using Exploratory Data Analysis using Ensemble Learning. EUDL, 2021. 10.4108/eai.13-10-2021.171319

[14] More, S., Idrissi, M., Mahmoud, H., Asyhari, A.T. Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis. Journal of Big Data, vol. 8, p. 90, 2021. https://doi.org/10.1186/s40537-021-00475-1.

[15] Moustafa, N., & Slay, J. (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). Military Communications and Information Systems Conference (MilCIS), 2015. IEEE.

[16] Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset. Information Security Journal: A Global Perspective, 1-14.

[17] Moustafa, N., et al. (2017). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. IEEE Transactions on Big Data.

[18] Moustafa, N., et al. (2017). Big data analytics for intrusion detection system: statistical decision-making using finite dirichlet mixture models. Data Analytics and Decision Support for Cybersecurity. Springer, Cham, 127-156.

[19] Sarhan, M., Layeghy, S., Moustafa, N., & Portmann, M. (2020). NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings (p. 117).

[20] Buczak, A.L., Guven, E. (2015), A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications surveys & tutorials.

[21] Liu, F.T., Ting, K.M., Zhou, Z.H. (2008), Isolation Forest. In: 2008 Eighth IEEE International Conference on Data Mining. IEEE

[22] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita (2013), “Network anomaly detection: methods, systems and tools,” IEEE communications surveys & tutorials, vol. 16, pp. 303–336.

[23] Lizama-Perez, L.A.; López R, J.M.; others. Non-Invertible Public Key Certificates. Entropy 2021, 23, 226.

[24] More, S.; Idrissi, M.; Mahmoud, H.; Asyhari, A.T. Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis. Algorithms 2024, 17, 64. https://doi.org/10.3390/a17020064

ANEXO A: ACCESO AL CODIGO FUENTE

Para acceder al código fuente del proyecto se lo puede realizar a través del siguiente enlace:

https://github.com/GaboLara998/ProyectoIntegrador