UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 4

Comparación de métodos de reducción de dimensionalidad para algoritmos de clasificación supervisada aplicados a la detección de intrusiones

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Roberto Antonio Vega Rojas

Josué Hernández Martinez

Tutor: Ing. Darvis Dorvigny Dorvigny

La Habana, noviembre de 2023

"Año 65 de la Revolución"

“Solo sé que no sé nada”.

Sócrates

DELARACIÓN DE AUTORÍA

Los autores del trabajo de diploma con título "**Comparación de métodos de reducción de dimensionalidad para algoritmos de clasificación supervisada aplicados a la detección de intrusiones.**", conceden a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declaran como únicos autores de su contenido. Para que así conste firman la presente a los 29 días del mes de noviembre del año 2023.

Roberto Antonio Vega Rojas

Autor

Josué Hernández Martínez

Autor

Ing. Darvis Dorvigny Dorvigny

Tutor

DATOS DE CONTACTO

Tutor:

Ing. Darvis Dorvigny Dorvigny: Graduado en el año 2008 como Ingeniero en Ciencias Informáticas, en la Universidad de las Ciencias Informáticas. Se encuentra ocupando el cargo de Director General en el área de Dirección General de Tecnología. Correo electrónico: ddorvigny@uci.cu.

DEDICATORIA

A nuestros padres y amigos.

AGRADECIMIENTOS

A nuestro tutor y a todas las personas que de una forma u otra contribuyeron a llegar a este momento.

De parte de Roberto:

A mis padres Yoanka y Miguel, mis abuelos Milagros, Anita, Alexis y Miguel, familiares Pablo, Pablito y Ángel.

De la Uci: Ariel, Nelson, Dannis, Vladimir, Carlos, Enrique, José Carlos, Carlos Daniel, Osmany, Yilber, Addiel, Denis Antony, Lázaro, Alain, Odlanyer, el pollo, Molina, Fidel, Sandy, Richard, Milena, Daniela, Heydi, Yanitza, Ana Nathalí, Beatriz, Aarom y Rosmery.

A mis amistades de Las Tunas Enrique, Norge, Ariel, Alber y Dayan.

A mi compañero de tesis Josué.

De parte de Josué:

Quiero agradecer especialmente a mi madre Mariela por apoyarme en todos estos 5 años de carrera, dando los mejores consejos y guías, a mi abuelo Alfredo por ser mi inspiración a seguir, y mi mayor héroe, a mi tío Alfredo por creer en mí desde niño que algún día lograría ser ingeniero como él y en especial a mi abuela María Ester que a pesar de no poder estar conmigo ahora para verme te llevo en mi corazón cada segundo de mi vida, siempre fuiste la que me impulsó a mejorar y a esforzarme para ser lo que soy ahora. Agradezco a mi tutor Darvis por apoyarnos en todo y siempre guiándonos por el camino de formación hacia un mejor ingeniero, a mis compañeros, que más que eso son mis hermanos por apoyarme en este arduo viaje, en especial a Osmany por brindar una ayuda desinteresada, a Roberto por ser un compañero ejemplar en el completamiento de esta labor, a Jean que siempre veló por que haga las cosas lo mejor posible aunque seas un incordio a veces, a Elier por ser Elier, a Felipe por esos momentos de ocio que tanto necesitaba después de un día de trabajo, a todos los profesores que tuve por sus valiosas enseñanzas y a todos los que ayudaron en mi formación y crecimiento como persona.

Muchas gracias de corazón.

RESUMEN

En la actualidad el crecimiento de las redes de comunicación ha proporcionado un escenario para el desarrollo de actividades maliciosas y ataques informáticos. Los Sistemas de Detección de Intrusiones se encargan de monitorizar el comportamiento de la red, alertando cualquier intento de actividad sospechosa. Con el auge y desarrollo de la Inteligencia Artificial, especialmente los algoritmos de aprendizaje automático, ha evolucionado la forma en que se realiza la detección de intrusiones. Un problema clave es que los métodos de detección de intrusiones basados en aprendizaje automático poseen una alta complejidad en los modelos de clasificación que genera alto costo computacional computacional en determinados escenarios ante problemas de grandes dimensiones. El presente trabajo tuvo como objetivo comparar métodos de selección de características para identificar aquellos que mejoren el rendimiento de algoritmos de clasificación aplicados a la detección de intrusiones. La estrategia a seguir fue de tres fases principales: preprocesamiento de datos, entrenamiento de clasificadores, prueba y evaluación de los clasificadores. El método de selección de características RFE en combinación con KNN obtuvo el mejor desempeño en la clasificación de intrusiones, con 99% de efectividad. Se utilizó el lenguaje de programación Python sobre el ambiente de Google Colaboratory.

Palabras clave: aprendizaje automático, selección de características, métodos de clasificación, detección de intrusiones, reducción de dimensionalidad.

ABSTRACT

At present, the expansion of communication networks has provided a scenario for the development of malicious activities and computer attacks. Intrusion Detection Systems are responsible for monitoring network behavior, alerting any attempt of suspicious activity. With the rise and development of Artificial Intelligence, especially Machine Learning algorithms, the way in which intrusion detection is performed has evolved. A key problem is that intrusion detection methods based on machine learning have a high complexity in the classification models that generates high computational cost in certain scenarios with large problems. The objective of this work was to compare feature selection methods to identify those that improve

the performance of classification algorithms applied to intrusion detection. The strategy followed was three main phases: data preprocessing, classifier training, testing and evaluation of classifiers. The RFE feature selection method in combination with KNN obtained the best performance in intrusion classification, with 99% effectiveness. The Python programming language was used on the Google Colaboratory environment.

Keywords: *machine learning, feature selection, classification methods, intrusion detection, dimensionality reduction.*

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la era digital actual, el crecimiento exponencial de las redes de comunicación ha proporcionado un escenario propicio para el desarrollo de actividades maliciosas y ataques informáticos. Estos ataques representan una amenaza significativa para la integridad, privacidad y disponibilidad de la información en las redes (Dhabliya, 2021).

Los ataques intentan aprovechar las vulnerabilidades del sistema objetivo para ver cuál es la mejor forma de comprometer el sistema y consecuentemente obtener un beneficio. Los ataques más comunes a una red son categorizados en:

- Ataques basados en web (inyección SQL, man in the middle, ataques de diccionario y fuerza bruta, cross-site scripting, denegación de servicio (DoS), phishing.).
- Ataques basados en el sistema (virus, gusanos, troyanos, backdoor, exploits, bots) (Trapero Estepa, 2021).

La ciberseguridad es el área de las ciencias de la computación encargada del desarrollo y la implementación de mecanismos de protección de la información y de la infraestructura tecnológica. Este campo de estudio cobra especial importancia en el mundo interconectado en el que se vive en la actualidad; un ciberataque puede resultar en el robo de datos sensibles u ocasionar la caída de equipos críticos, de ahí la obligación de mantener segura cualquier tipo de infraestructura. Una forma efectiva de hacerlo es mediante la implementación de un sistema de detección de intrusiones (Trapero Estepa, 2021).

Un Sistema de Detección de Intrusiones (IDS) está en la primera línea de defensa en una red informática. En el nivel más alto, los IDS se clasifican en: IDS basados en host (H-IDS) e IDS basados en red (N-IDS). El N-IDS funciona de forma distribuida dentro de un sistema de red, y el H-IDS se ejecuta en una computadora o sistema anfitrión. Además, ambos tipos de IDS funcionan utilizando dos métodos, estos son, la detección basada en anomalías y el reconocimiento basado en firmas (Kasongo & Sun, 2020). El desarrollo de la Inteligencia Artificial como campo de la ciencia de la computación, ha permitido su aplicación en diversos

campos como la ciberseguridad, especialmente en la detección de intrusiones basada en anomalías.

Una limitación de los IDS basados en anomalías es la complejidad de los modelos debido a la alta dimensionalidad de los datos, lo que representa un desafío y dificultad en las decisiones tomadas por el modelo y la identificación de patrones en los datos (Belén Del Río, 2021). Un número mayor de características implica un costo computacional superior, y por tanto, una afectación al rendimiento de los clasificadores que se utilizan en sistemas de detección de intrusiones basados en anomalías. Esto ha motivado el desarrollo de varias técnicas de reducción de dimensionalidad en los datos.

De acuerdo a los elementos expuestos anteriormente, se plantea el siguiente **problema de investigación:** los algoritmos de clasificación supervisada que se aplican en la detección de intrusiones, se ven afectados por la alta dimensionalidad de los datos.

Para dar solución a la problemática antes mencionada se propone como **objetivo general** comparar métodos de reducción de dimensionalidad para identificar aquellos que mejoren el rendimiento de algoritmos de clasificación aplicados a la detección de intrusiones.

Se define como **objeto de estudio** métodos de reducción de dimensionalidad y clasificación supervisada en el aprendizaje automático. El **campo de acción** de esta tesis se centra métodos de reducción de dimensionalidad y clasificación supervisada del aprendizaje automático para la detección de intrusiones.

Para lograr este objetivo se hizo importante trazar una serie de tareas que dieran como resultado la propuesta de solución, las cuales se encuentran desglosadas a continuación:

- Realizar una revisión de la literatura relacionada con detección de intrusiones y de algoritmos aprendizaje automático.
- Recopilar y obtener un conjunto de datos representativos de tráfico de red, que incluyan tanto actividades normales como maliciosas.
- Realizar un análisis de los atributos de tráfico de red y su relevancia para la detección de intrusiones.

- Evaluar la efectividad y precisión de los métodos de clasificación supervisada.

Los métodos de investigación utilizados para la realización de este trabajo fueron:

Métodos teóricos:

- Analítico-Sintético: luego de analizar las teorías y documentos relacionados con la detección de intrusiones se pudo determinar la esencia del problema. Analizando otras propuestas de solución se pudo reconocer los aspectos coincidentes, en cuanto a los métodos planteados.
- Histórico-Lógico: para el estudio crítico de trabajos anteriores y para utilizarlos como punto de referencia.

Métodos empíricos:

- Observación: para identificar los mejores algoritmos empleados en el aprendizaje automático y poder escoger la mejor combinación de métodos de selección de características y clasificación en la detección de intrusiones.
- Medición: para el cálculo de las métricas de evaluación del desempeño de los clasificadores y para comparar sus resultados.
- Experimentación: para validar los resultados derivados de las métricas obtenidas de los clasificadores.

El presente trabajo se compone de tres capítulos de la siguiente forma:

Capítulo 1 Fundamentación Teórica: en este capítulo se estudian los referentes teóricos y el estado del arte de las técnicas de aprendizaje automático aplicado a la detección de intrusiones.

Capítulo 2 Fases del proceso de detección de intrusiones utilizando técnicas de aprendizaje automático: en este capítulo se documentan las diferentes fases del proceso de detección de intrusiones y se expone la estrategia de comparación de algoritmos.

Capítulo 3 Resultados experimentales del estudio aplicado: en este capítulo se muestran y discuten los resultados de la investigación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

A lo largo de este capítulo se expondrán los conceptos necesarios para facilitar la comprensión del tema desarrollado. Además, se determinan y se describen las distintas herramientas y métodos que permiten llegar a la propuesta de solución.

1.1. Conceptos asociados al problema

1.1.1. Detección de intrusiones

La detección de intrusiones es el proceso de monitorizar los eventos que ocurren en un sistema o red, para analizarlos en busca de problemas de seguridad y tienen funciones de vigilancia, alarma y emiten alarmas cuando un determinado suceso tiene lugar (Alazzam et al., 2020). Este proceso se realiza mediante los Sistemas de Detección de Intrusiones.

Los Sistemas de Detección de Intrusiones son programas informáticos utilizados para detectar accesos no autorizados o maliciosos en una computadora o red (Umbarkar & Shukla, 2018). Esto se realiza generalmente mediante el análisis de los paquetes recibidos por una determinada interfaz de red. Cuando el IDS detecta un paquete o conjunto de paquetes que puedan corresponderse a un ataque informático, genera una alarma, que tendrá que ser atendida por un técnico en seguridad (Alazzam et al., 2020).

Lo deseable sería que el programa reaccionase únicamente ante paquetes que corresponden a un ataque, pero esto es imposible. El IDS nunca sabrá a ciencia cierta si un paquete corresponde a un ataque o no; deberá utilizar una serie de técnicas para tratar de determinarlo. En función del tipo de sistema, será deseable una mayor o menor sensibilidad frente a paquetes sospechosos (Mario Aragonés Lozano, 2020).

Los IDS pueden monitorizar información de toda una red o un equipo. Aquellos que monitorizan solo el dispositivo en el que están instalados, se denominan Sistemas de Detección de Intrusiones basados en Host (HIDS, por sus siglas en inglés) (Masdari & Khezri, 2020). Los HIDS pueden monitorizar, además de los paquetes que llegan a una determinada interfaz de red, elementos muy diferentes, como archivos de log o procesos en

ejecución, por lo que son bastante dependientes de la veracidad de esta información (Amrita & Ahmed, 2012).

Por otro lado, los IDS basados en Red (NIDS) analizan información relativa a toda una red (Masdari & Khezri, 2020). Generalmente son instalados en diferentes equipos situados en determinados puntos de una misma red informática para tener una visión global de lo que ocurre en la red (Amrita & Ahmed, 2012).

De modo complementario a la clasificación anterior, los IDS pueden clasificarse en función de las técnicas que utilizar para determinar si un paquete o conjunto de paquetes corresponde a un ataque:

- Los IDS basados en firmas analizan el contenido de los paquetes y lo comparan con una base de datos de firmas conocidas correspondientes a ataques ya documentados. Un paquete o conjunto de paquetes se descompone en una serie de características que en conjunto determinan una firma (Alazzam et al., 2020). Si la firma de parte del tráfico de red se corresponde con alguna de las de la base de datos, este se marcaría como malicioso. El administrador también puede crear sus propias firmas para que salte una alarma cuando ciertas situaciones se produzcan en su red. Este tipo de IDS no sirve para detectar ataques de los que no se tenga constancia previa; aunque, sí se utilizan las firmas adecuadas para cada red, puede tenerse una seguridad bastante alta de que, cuando genera una alarma, se ha detectado un ataque (Ávila Pérez, 2020).
- Los IDS basados en anomalías tratan de modelar el tráfico legítimo y habitual de una red analizándolo. Cuando un paquete o flujo de paquetes difiere de este modelo se marca como malicioso y se genera una alerta. De esta forma un IDS basado en anomalías sí que puede detectar ataques que nunca se hayan visto antes, aunque puede generar alarmas para eventos legítimos o ignorar aquellos que no lo son. Todo depende cuanto se aproxime a la realidad del modelo utilizado para diferenciar los eventos maliciosos de los legítimos (Alazzam et al., 2020). Es aquí donde juegan un papel importante las técnicas de aprendizaje automático, puesto que proporcionan

una herramienta para que el IDS aprenda el comportamiento normal de un equipo o una red e identifique cualquier anomalía que se salga de este.

A continuación, un esquema sobre la taxonomía de los IDS y un esquema sobre los IDS de acuerdo a su estrategia de análisis:

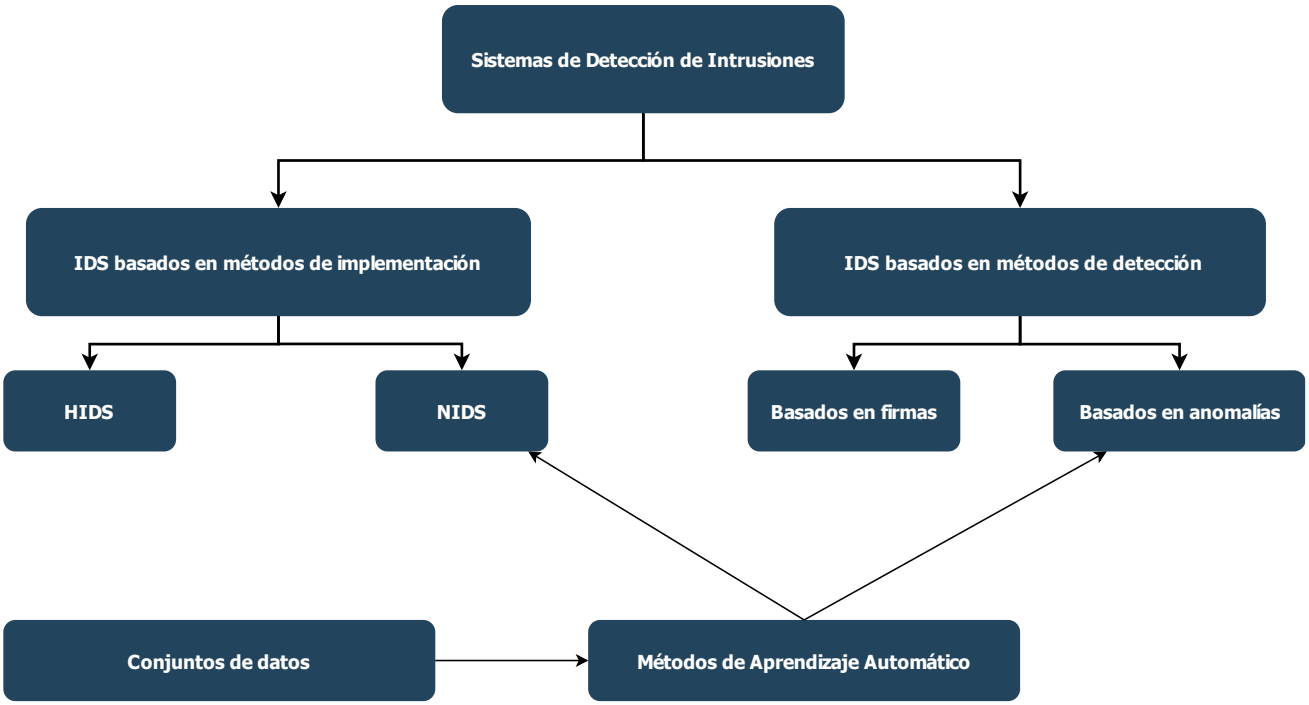

Figura 1: Breve taxonomía de los IDS. Elaboración por los autores.

Los IDS basados en firmas son utilizados con mayor frecuencia porque tienden a ser rápidos en detección, pero tienen el problema fundamental de necesitar actualización constante de sus bases de datos, porque se quedan obsoletas según avanza el tiempo. En cambio, los basados en anomalías superan en alguna medida este problema, dado que generan un modelo clasificador del tráfico de red y todo lo que difiere de este modelo genera una alerta. Es en este tipo de IDS es donde los métodos de aprendizaje automático poseen relevancia, ya que son los que permiten la generación de un modelo efectivo.

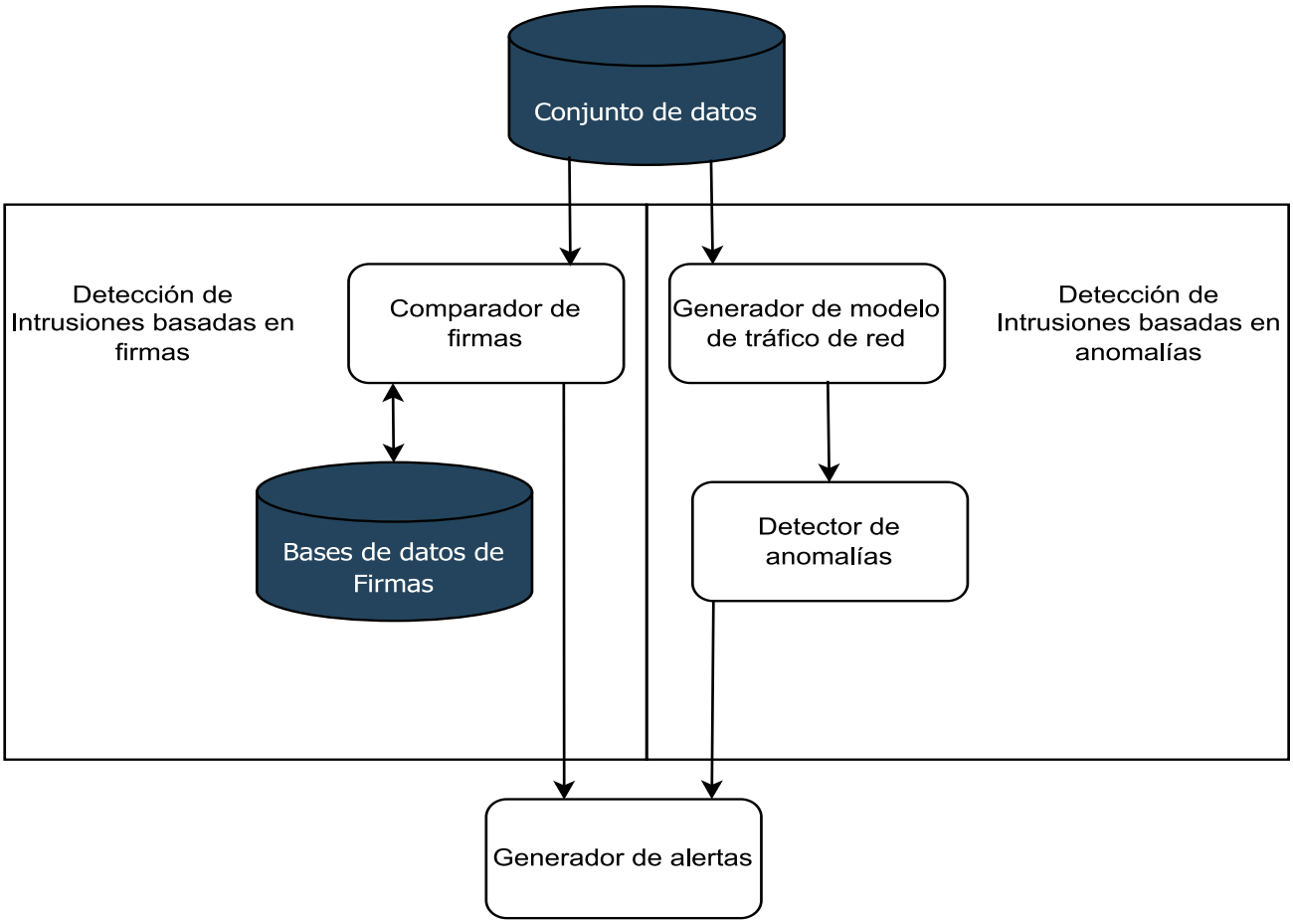

Figura 2: Esquema funcional sobre IDS respecto a su estrategia de análisis. Elaboración por los autores.

Debido al auge de la Inteligencia Artificial (IA) en los últimos años, esta ha abarcado varias aristas en la tecnología, una de ellos es su incorporación a los IDS. La Inteligencia Artificial (IA), en el contexto de las ciencias de la computación, es el estudio y diseño de agentes racionales (Stuart J. Russell & Peter Norvig, 2021).

Dentro de la IA se encuentra la rama conocida como Aprendizaje Automático. El Aprendizaje Automático es la rama de la IA cuyo objetivo es desarrollar técnicas que permitan que un programa informático aprenda. De esta forma, un algoritmo de aprendizaje automático es capaz de aprender a partir de unos datos. “Consideramos que un programa aprende de una experiencia E respecto a una tarea especifica T y a una medida de rendimiento R; cuando su

rendimiento en la tarea T, de acuerdo a la métrica R, aumenta con la experiencia E" (Tari et al., 2020).

Los algoritmos de aprendizaje automático aprenden procesando datos, habitualmente tabulados, que contienen valores para cada una de las variables organizadas como columnas, también llamadas características. El número de estas impacta en la dimensionalidad del problema de aprendizaje, y por tanto en la complejidad del modelo. (Tari et al., 2020). Algunos conjuntos de datos de ciberseguridad utilizados con frecuencia para el desarrollo de sistemas de detección de intrusiones son:

- KDD CUP 1999: basado en el conjunto de datos DARPA 1998/1999, creado por el MIT Lincoln Lab a partir de un entorno de red emulado, en el que se capturaron durante varias semanas tanto tráfico normal como anómalo. El conjunto de datos contiene atributos básicos sobre conexiones tcp y atributos de alto nivel, como el número de inicios de sesión fallidos, pero no direcciones IP (Castellanos Leyva & García Borroto, 2019). KDD CUP 99 contiene datos basados en flujo enriquecidos con datos provenientes de logs de sistema y de los paquetes de red implicados en la comunicación. El Instituto Canadiense de Ciberseguridad (CIC, por sus siglas en inglés), ofrece este conjunto de datos. Consta de un total de 42 columnas, de ellas 41 de características y la restante es la variable objetivo con la clasificación del tráfico de red (Valdezate Álvarez, 2020).
- NSL-KDD: creado en 2009 como una actualización y mejora del KDD CUP 99, eliminando duplicados del conjunto de datos KDD CUP99 (Castellanos Leyva & García Borroto, 2019). Es considerado un buen benchmark para evaluar diferentes métodos de detección de intrusiones y sigue siendo uno de los más utilizados en estudios recientes, por lo que ofrece la ventaja de que permite comparar los estudios realizados sobre él al estar su uso bastante estandarizado. Al igual que KDD CUP 99 tiene datos basados en flujos enriquecidos con información procedente de los hosts y los paquetes implicados en la comunicación. Posee similar estructura al KDD CUP 99 (Valdezate Álvarez, 2020).

- UNSW-NB15 es creado con el uso de la herramienta IXIA Perfect Storm en un pequeño entorno emulado durante 31 horas. La generación de tráfico sintético con comportamiento malicioso incluye nueve tipos de ataques como backdoors, DoS, exploits, worms y fuzzers. Está compuesto por 49 características extraídas con herramientas como Argus, Bro-IDS, entre otras. En el sitio oficial, https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets, los investigadores pueden encontrar 4 archivos de tipo csv que contienen los registros de ataques y comportamiento normal, así como otros archivos con los datos de origen de cada una de las herramientas utilizadas (Castellanos Leyva & García Borroto, 2019).
- CICIDS 2017 se creó en un entorno emulado durante un período de 5 días, contiene tráfico de red en formato de paquetes y los resultados del análisis del tráfico en flujo bidireccional, obtenidos con la herramienta CICFlowMeter y descritos por 85 características. Incluye una amplia gama de tipos de ataque como SSH, brute force, heartbleed, botnet, DoS, DDoS, web y ataques de infiltración. Para generar el tráfico de fondo se implementaron perfiles de comportamiento abstracto de 25 usuarios basados en protocolos como HTTP, HTTPS, FTP, SSH y protocolos de correo electrónico. Este conjunto de datos se puede obtener en https://www.unb.ca/cic/datasets/ids-2017.html (Castellanos Leyva & García Borroto, 2019).
- En el caso de CDMC 2013 es un conjunto de datos creado a partir de un sistema de detección de intrusos real. Al igual que en CDMC 2012 es necesario dividir los datos en 2 conjuntos. Cada instancia está conformada por 7 atributos numéricos, además de del atributo que define la clase (Herrera Semenets, 2019).

Existen varios tipos de algoritmos de aprendizaje automático en función de la forma en la que procesan los conjuntos de datos durante su fase de aprendizaje:

- En el aprendizaje no supervisado se procesa un conjunto de datos para obtener sus propiedades útiles. De esta forma el sistema aprende qué esperar de los datos de

entrada; construye un modelo aproximado de los valores de las diferentes características y las relaciones entre ellos (Tari et al., 2020).

- En el aprendizaje supervisado se procesan datos etiquetados a partir del conocimiento de expertos, expresado en una columna objetivo. Esta característica dependerá del sistema de aprendizaje en cuestión. En un NIDS le mostrará al sistema eventos de la red y se le indicará a que categoría pertenecen (eventos legítimos o maliciosos) (Tari et al., 2020).
- En el aprendizaje por refuerzo, los algoritmos aprenden de la experiencia, ya que es agregado un mecanismo de recompensa que se encarga de premiar o penalizar al sistema en base a la calidad de las decisiones ya sean estas buenas o malas, produciendo que el sistema se actualice continuamente, asimismo. Por lo tanto, el sistema aprende a base de ensayo-error (Alba Vega & Calle Jara, 2020).

1.1.2. Reducción de dimensionalidad

Cuando los algoritmos de aprendizaje automático se aplican a conjuntos de datos de alta dimensión, surge un problema crítico conocido como la “maldición de la dimensionalidad”. Este es el fenómeno de que los datos se vuelven más dispersos en el espacio de alta dimensión, lo que afecta negativamente a los algoritmos diseñados para el espacio de baja dimensión. Además, con un gran número de características, los modelos de aprendizaje tienden a sobre ajustarse, lo que puede causar una degradación del rendimiento en datos no vistos. Los datos de alta dimensionalidad pueden aumentar significativamente los requisitos de almacenamiento en memoria y los costes computacionales para el análisis de datos(Li et al., 2017).

En el contexto del Aprendizaje Automático, una característica es una variable, atributo o propiedad individual medible, como entrada para un algoritmo de aprendizaje automático. Pueden ser numéricas, categóricas o basadas en texto, y representan diferentes aspectos de los datos que son relevantes para el problema en cuestión (Charfaoui, 2020).

Se denomina reducción de la dimensionalidad al proceso de encontrar una representación relacionada para las características de entrada en un espacio de menor dimensión. Este

proceso debe ser capaz de acomodar datos no vistos en tiempo de prueba (ejecución) y generalizar adecuadamente a partir de ahí. Si sus características son dispersas, puede sospechar que sólo una fracción de los posibles vectores de características tienen probabilidades de aparecer. Eso significa que sus datos normalmente caen en un subespacio más pequeño y regular del espacio de características (Duboue, 2020).

La reducción de dimensionalidad puede dividirse en dos componentes principales: extracción y selección de características. La extracción de características proyecta las características originales de alta dimensionalidad a un nuevo espacio de características de baja dimensionalidad. El nuevo espacio de características suele ser una combinación lineal o no lineal de las características originales(Li et al., 2017). Un ejemplo de este método es el Análisis de Componentes Principales.

- El Análisis de Componentes Principales (PCA, por sus siglas en inglés) es una técnica del álgebra lineal que puede utilizarse para reducir la dimensionalidad. Tiene una definición matemática general y un caso de uso específico en el análisis de datos. PCA encuentra el conjunto de direcciones ortogonales que representan la matriz de datos original. PCA funciona mapeando el conjunto de datos original en un nuevo espacio en el que cada uno de los nuevos vectores columna de la matriz son ortogonales (Jain & Singh, 2018). Desde la perspectiva del análisis de datos, el PCA transforma la matriz de covarianza de los datos en vectores columna que pueden explicar determinados porcentajes de la varianza (Avila & Hauck, 2017).

No obstante, el empleo de este tipo de métodos posee como desventaja que al reemplazarse los atributos originales por otros se modifica el conjunto de datos y por tanto las características generadas pierden su legibilidad. Por otra parte esto no sucede con los métodos de selección de características (Herrera Semenets, 2019).

La selección de características se refiere al proceso de obtener un subconjunto de un conjunto original de características de acuerdo con un determinado criterio de selección de características, que selecciona las características relevantes del conjunto de datos (Cai et al., 2018).

Existen varios métodos para la selección de características: los métodos basados en filtros, los métodos de envoltura, y los métodos integrados (Thakkar & Lohiya, 2021). Los métodos basados en filtros son un proceso de selección de características separado de los algoritmos de entrenamiento del modelo (Herrera Semenets, 2019). En general, se evalúa la correlación entre cada característica y el objetivo mediante la puntuación de cada característica, eliminando las características con una correlación menor que cierto umbral y manteniendo las características con una correlación mayor como entrada del modelo (Xiaosong et al., 2020).

La ventaja del método de filtro es que es independiente del entrenamiento del modelo y no afecta la aplicación posterior después del filtrado. Esto facilita su aplicación a datos de alta dimensionalidad con un costo computacional relativamente bajo. Sin embargo, debido a que la correlación de cada característica se calcula por separado, se ignora la correlación entre características, lo que puede llevar a un mal rendimiento del conjunto de características seleccionadas (Sahu et al., 2018). Algunos de estos métodos son:

- Ganancia de información (IG, por sus siglas en inglés): es un método de evaluación de características basado en la entropía, ampliamente utilizado en el campo del Aprendizaje Automático. Como la ganancia de información se utiliza en la selección de características, se define como la cantidad de información proporcionada por las características. La ganancia de información de cada atributo se calcula teniendo en cuenta los valores objetivo para la selección de características y cuánto de un término puede utilizarse para clasificar la información (Jabali et al., 2017).
- Correlación de Pearson's: es una medida de la relación lineal entre dos variables aleatorias numéricas. Devuelve un valor que indica la fuerza de la correlación entre dos variables. Se calcula tomando la covarianza de dos variables y dividiendo por el producto de sus desviaciones estándar. El coeficiente no se ve afectado por los cambios de escala en las dos variables (Montoya Guirado, 2018).

  La correlación de Pearson's varía entre -1 y 1, un valor de correlación de 1 indica una relación lineal positiva perfecta entre las dos variables mientras que un valor de -1

indica una relación lineal negativa perfecta. Un valor de 0 indica ausencia de relación lineal entre las variables (Xiaosong et al., 2020).

- Análisis de Varianza (ANOVA, por sus siglas en inglés): es un método muy simple y eficaz para probar la diferencia en medias entre grupos. Es intuitivo para analizar la interacción de las dos variables y puede utilizarse de manera efectiva incluso cuando el número de observaciones es diferente en cada grupo (Ding et al., 2014).

Los métodos basados en envoltorios utilizan algoritmos de aprendizaje específicos para evaluar la calidad de las características seleccionadas, son dependientes del método de aprendizaje que se emplea (Herrera Semenets, 2019). El rendimiento de la predicción del modelo se utiliza para evaluar el subconjunto de características (Khun & Johnson, 2019). Por lo general, se predefine un proceso de búsqueda en el espacio de posibles subconjuntos de características, y se generan y evalúan varios subconjuntos de características. El proceso general es: seleccionar un subconjunto, evaluar el subconjunto según el rendimiento de la predicción, seleccionar un nuevo subconjunto y continuar evaluando hasta que se alcance la calidad esperada (Cai et al., 2018).

En los métodos basados en envoltorios uno muy utilizado es el de Eliminación Recursiva de Características (RFE, por sus siglas en inglés) que es un algoritmo que busca encontrar el subconjunto de características con mejor rendimiento en un modelo (Khun & Johnson, 2019). Para esto crea repetidamente modelos y deja de lado la peor característica en cada iteración. Construye el siguiente modelo con las características que quedan y repite este procedimiento hasta quedarse sin características. Al final clasifica las características por el orden de eliminación y se queda con las más importantes (Ariza Gacharná, 2022).

Los métodos integrados se diferencian de los métodos filtro y envoltura en que, con los métodos integrados, la parte de aprendizaje y la parte de selección de características no pueden separarse. Los métodos integrados realizan la selección de características en el proceso de aprendizaje, lo que ahorra el tiempo necesario para la inducción en dos pasos como en los métodos de envoltura. La búsqueda del mejor subconjunto de características se incorpora a la construcción del clasificador (Sahu et al., 2018).

Los métodos integrados son más eficientes que los de envoltura a la hora de utilizar mejor los datos disponibles sin necesidad de dividir los datos de entrenamiento en conjuntos de entrenamiento y pruebas. Además, encuentran una solución más rápidamente, ya que no es necesario volver a entrenar un predictor desde cero para cada subconjunto de variables examinado (Sahu et al., 2018).

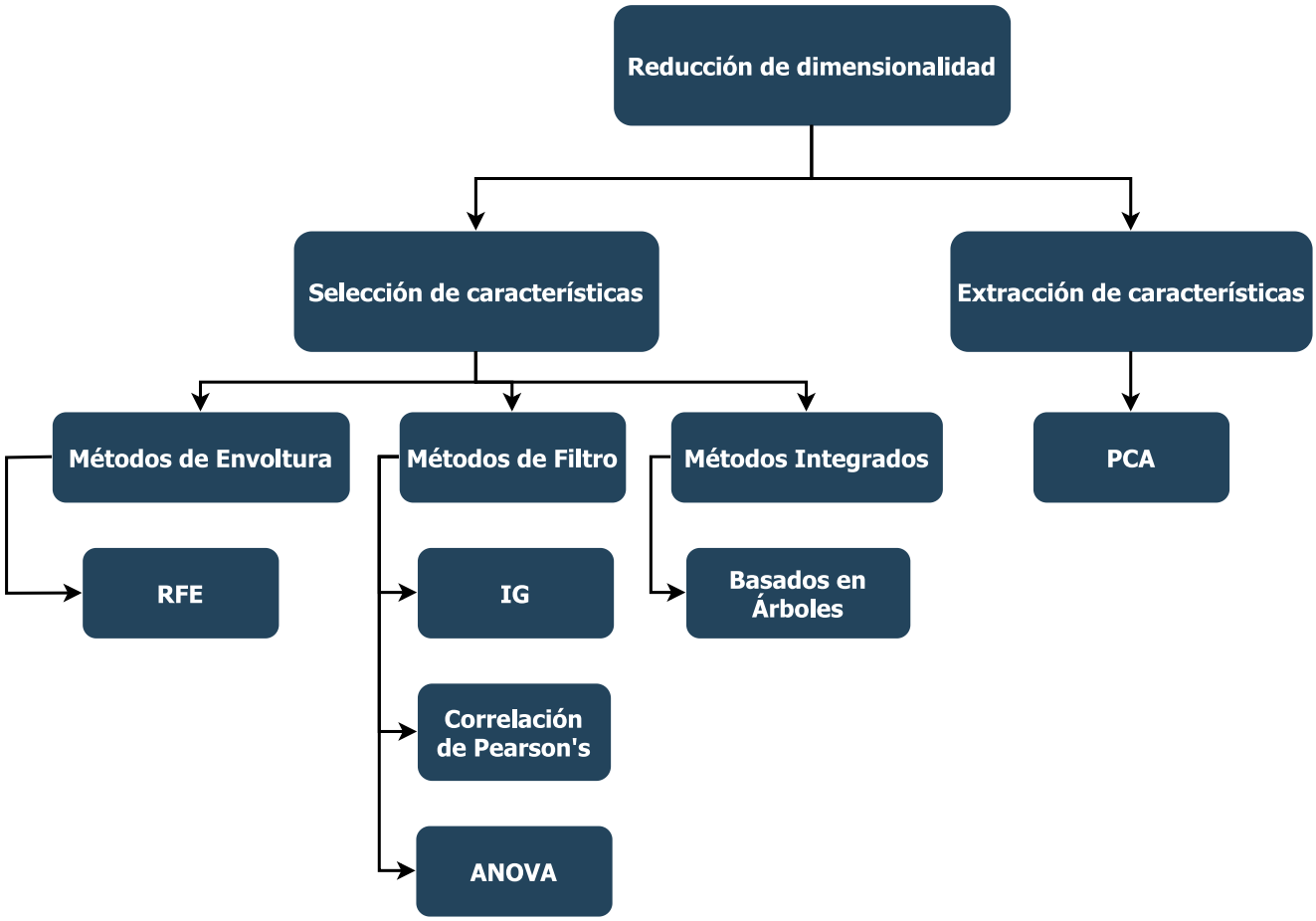

Figura 3: Reducción de dimensionalidad y sus ramas. Modificado de (Herrera Semenets, 2019).

1.1.3. Algoritmos de clasificación supervisada

La clasificación y la regresión son dos problemas de predicción importantes que generalmente se tratan en Aprendizaje Automático (Haipeng Yao, Chunxiao Jiang, Yi Qian, 2019). La regresión es el proceso de encontrar un modelo o función para distinguir los datos

en valores reales continuos, en lugar de utilizar clases o valores discretos. También puede identificar el movimiento de distribución en función de los datos históricos (Calderón Romero & Hurtado Cortes, 2019).

La clasificación es el proceso de encontrar o descubrir un modelo o función que ayude a separar los datos en múltiples clases categóricas, es decir, valores discretos (Martínez Martínez, 2021). En la clasificación, los datos se clasifican bajo diferentes etiquetas de acuerdo con algunos parámetros proporcionados en la entrada y luego se predicen las etiquetas para los datos, en los que los datos se pueden dividir en etiquetas discretas binarias o múltiples (Calderón Romero & Hurtado Cortes, 2019). La problemática de este trabajo pertenece a este tipo de problema.

A continuación, un resumen de cada algoritmo:

La Regresión Logística (RL) es un tipo de algoritmo de clasificación que se utiliza para predecir una variable de salida binaria. Su utilización es común en aplicaciones de aprendizaje automático donde la variable de salida es verdadera o falsa, como en la detección de fraude o el filtrado de spam. En la regresión logística, el algoritmo intenta encontrar una relación lineal entre las características de entrada y la variable de salida (Prateek, 2021). Luego, la variable de salida se transforma utilizando una función logística para producir un valor de probabilidad entre 0 y 1. La regresión logística es un algoritmo de aprendizaje automático supervisado (Alba Vega & Calle Jara, 2020).

Los Árboles de Decisión (DT, por sus siglas en inglés) son una de las herramientas más poderosas de los algoritmos de aprendizaje supervisado que se utilizan tanto para tareas de clasificación como de regresión. Construye una estructura de árbol similar a un diagrama de flujo donde cada nodo interno denota una prueba en un atributo, cada rama representa un resultado de la prueba y cada nodo hoja (nodo terminal) contiene una etiqueta de clase. Se construye dividiendo de forma recursiva los datos de entrenamiento en subconjuntos según los valores de los atributos hasta que se cumpla un criterio de parada, como la profundidad máxima del árbol o el número mínimo de muestras necesarias para dividir un nodo (E. Ramírez & E. Grandón, 2018).

Las Redes Bayesianas (NB, por sus siglas en inglés), red de creencias o modelo acíclico dirigido son un modelo probabilístico que representa una serie de variables de azar y sus independencias condicionales a través de un grafo acíclico dirigido. Una red bayesiana puede representar, por ejemplo, las relaciones probabilísticas entre enfermedades y síntomas. Dados ciertos síntomas, la red puede usarse para calcular las probabilidades de que ciertas enfermedades estén presentes en un organismo. Hay algoritmos eficientes que infieren y aprenden usando este tipo de representación (Kasongo & Sun, 2019).

Las Máquinas de Soporte Vectorial (SVM, por sus siglas en inglés) son una serie de métodos de aprendizaje supervisado usados para clasificación y regresión. Los algoritmos de SVM usan un conjunto de ejemplos de entrenamiento clasificado en dos categorías para construir un modelo que prediga si un nuevo ejemplo pertenece a una u otra de dichas categorías (Keshta, 2018).

K-Vecinos más Cercanos (KNN, por sus siglas en inglés) funciona encontrando k ejemplos de entrenamiento más cercanos a una entrada determinada y luego predice la clase o el valor en función de la clase mayoritaria o el valor promedio de estos vecinos. El rendimiento de KNN puede verse influenciado por la elección de k y la métrica de distancia utilizada para medir la proximidad. Sin embargo, es intuitivo, pero puede ser sensible a datos ruidosos y requiere una selección cuidadosa de k para obtener resultados óptimos. Es un tipo de algoritmo que se utiliza tanto para tareas de clasificación como de regresión (Fossaceca et al., 2015).

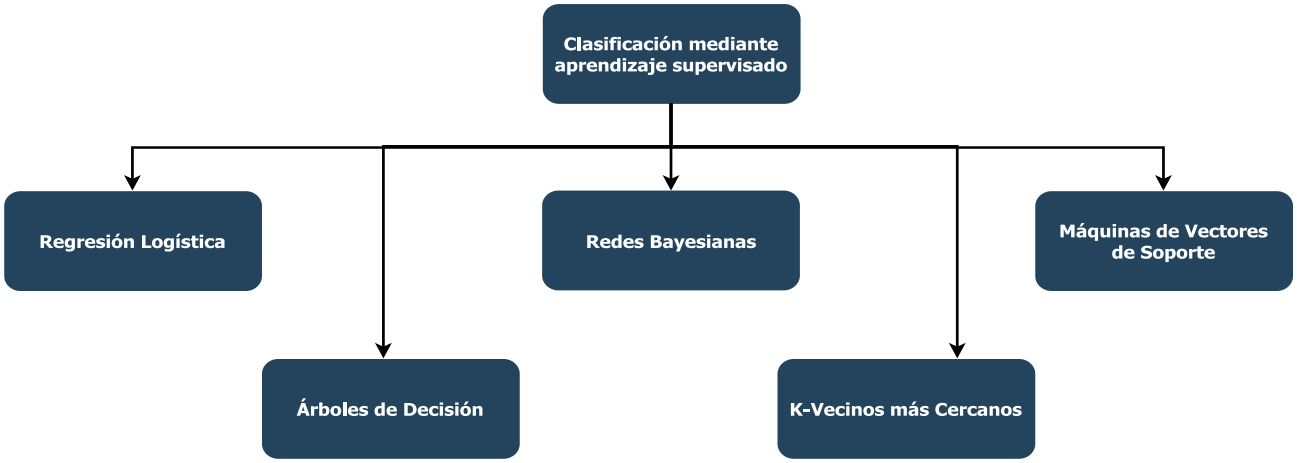

Figura 4. Algoritmos de clasificación mediante aprendizaje supervisado. Elaboración por los autores.

Tras la aplicación de los métodos de clasificación, resulta fundamental evaluar su desempeño mediante métricas, con el fin de establecer una comparativa acerca de qué enfoque de selección de características optimiza los resultados obtenidos por los clasificadores. Estas métricas se definen a continuación:

- Exactitud ("*Accuracy*") que representa la fracción de los paquetes clasificados correctamente frente a los paquetes clasificados en total. También es conocida como efectividad en ocasiones. Es una de las métricas de rendimiento más usadas en un algoritmo de clasificación (Valdezate Álvarez, 2020).

$$ACC = (VP + VN)/(VP + VN + FP + FN)$$

Figura 5. Fórmula para calcular la Exactidud. Tomado de: (Valdezate Álvarez, 2020).

- Precisión ("*Precision*") es la capacidad del clasificador de no etiquetar como positiva una muestra que es negativa. Esto representa el porcentaje de positivos que correspondan a un ataque. Una precisión alta significa pocos falsos positivos (Valdezate Álvarez, 2020).

$$P = VP/(VP + FP)$$

Figura 6. Fórmula para calcular la Precisión. Tomado de: (Valdezate Álvarez, 2020)

- Sensibilidad (*"Recall"*) o recuperación es la capacidad del clasificador de encontrar todas las muestras positivas. Esto representa el porcentaje de ataques que han sido detectados correctamente. Una sensibilidad alta significa pocos falsos negativos (Valdezate Álvarez, 2020).

$$S = VP/(VP + FN)$$

Figura 7. Fórmula para calcular la Sensibilidad. Tomado de: (Valdezate Álvarez, 2020).

- F1-score que es la media armónica de la precisión y la sensibilidad. Al igual que la exactitud, sirve para hacerse una idea general de cómo de bueno es un clasificador. Puede no ser la mejor métrica para la detección de intrusiones, pues asume que precisión y sensibilidad tienen igual importancia (Valdezate Álvarez, 2020).

$$Fscore = 2(P * S)/(P + S)$$

Figura 8. Fórmula para calcular F1-Score. Tomado de: (Valdezate Álvarez, 2020).

En las figuras del número 5 al 8 se encuentran representadas las fórmulas para calcular las métricas argumentadas anteriormente. Las siglas: VP significa verdaderos positivos, VN verdaderos negativos, FP falsos positivos y FN falsos negativos.

Se escoge como métrica fundamental en este trabajo la exactitud, sin dejar de tomar en cuenta las restantes por el estudio realizado en (Ahmad et al., 2021). Se muestran sus resultados en el capítulo 3 y los anexos correspondientes.

1.2. Conclusiones del capítulo

Con la realización de este capítulo se obtienen las siguientes conclusiones:

- El empleo de algoritmos de aprendizaje automático puede impactar en la mejora de la protección de las redes informáticas.

- Las técnicas de aprendizaje automático deben combinarse con el conocimiento de expertos en seguridad de redes para obtener los mejores resultados.
- Los algoritmos de aprendizaje automático supervisados como la clasificación, pueden entrenarse en datos históricos etiquetados para aprender a distinguir entre tráfico normal y anómalo.

CAPÍTULO 2: FASES DEL PROCESO DE DETECCIÓN DE INTRUSIONES UTILIZANDO TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

En este capítulo se documentan las diferentes fases del proceso de detección de intrusiones, identificando los procesos relativos a cada fase, las cuales se basan en diferentes propuestas de investigación que se han tomado como referentes.

2.1. Propuesta de solución

Luego de realizarse el análisis de las necesidades correspondientes a la problemática, se define como estrategia a seguir la comparación de varios métodos de selección con varios métodos de clasificación para buscar que composición de método de selección con clasificadores encuentra los mejores resultados en clasificación, en tres fases con sus respectivas subfases: fase de preprocesamiento de datos, entrenamiento y prueba de los clasificadores y clasificación del tráfico de red. En la figura 5 se muestra esta estrategia.

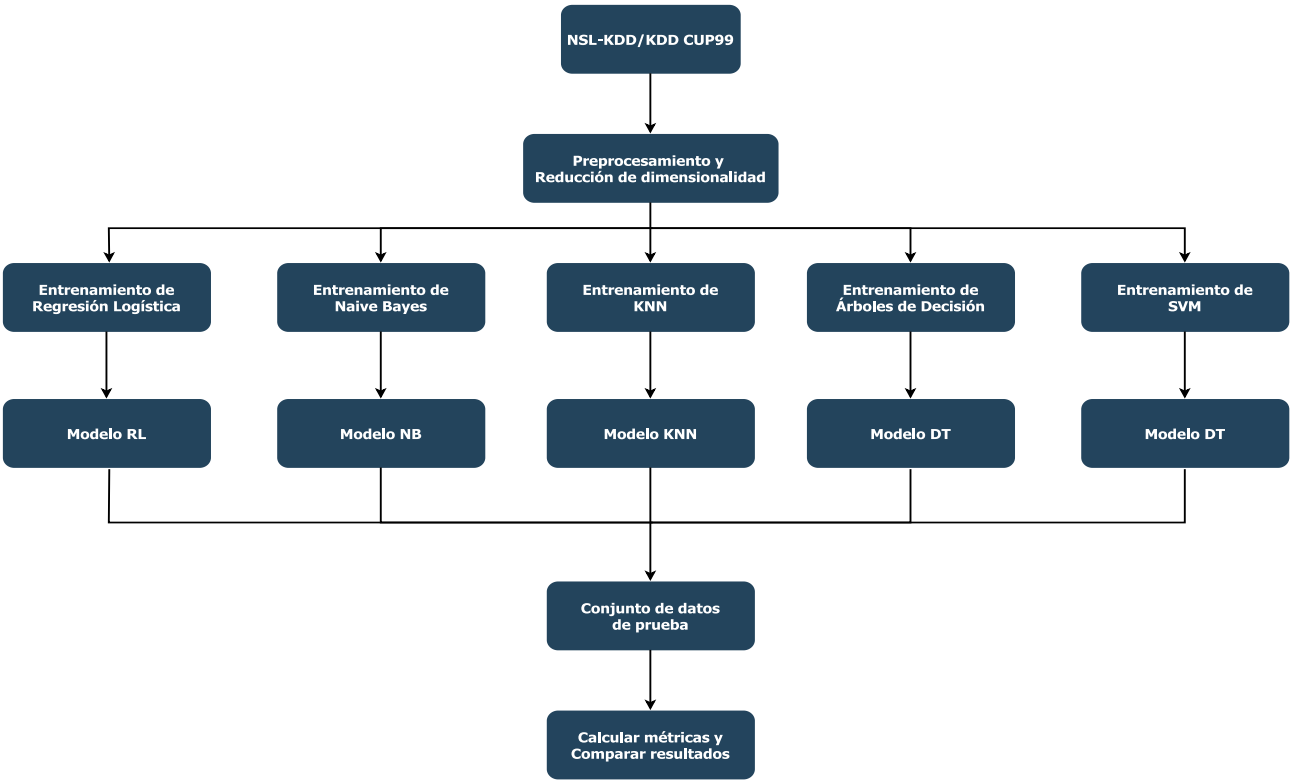

Figura 9. Estrategia de pasos en el proceso de detección de intrusiones. Elaboración por los autores.

2.1.1. Fase de preprocesamiento

El preprocesamiento de datos es una fase crucial en el desarrollo de modelos de aprendizaje automático. La calidad y relevancia de los datos de entrada son factores determinantes en la capacidad de un modelo para producir predicciones precisas y fiables. Sin embargo, los datos en bruto a menudo contienen ruido, valores atípicos, valores nulos y otras anomalías que pueden influir en el rendimiento del modelo. Así, por ejemplo, las máquinas de vector soporte son muy sensibles a la escala de los datos, mientras que los algoritmos basados en árboles de decisión no lo son. Dentro de esta fase, se encuentra la subfase de reducción de dimensionalidad (Fan et al., 2021).

El preprocesamiento de datos hace referencia al análisis de los conjuntos de datos y a la transformación de los datos en bruto en un formato más adecuado y comprensible para el algoritmo, esto se conoce como normalización. Este proceso incluye el tratamiento de valores nulos, la normalización de características y la codificación de variables categóricas. A continuación, la figura 6 muestra este proceso.

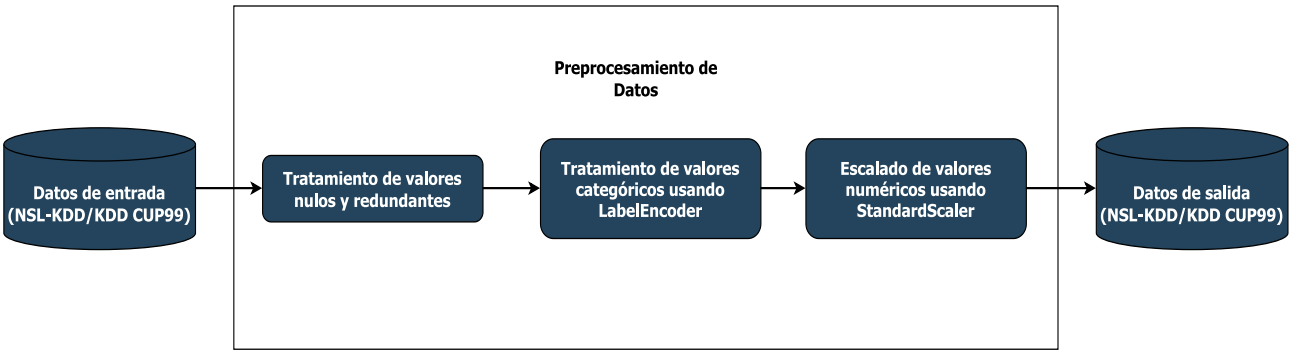

Figura 10. Preprocesamiento de datos. Elaboración por los autores.

En el análisis de datos se cargan los conjuntos de datos, así como un estudio de estos y las cantidades de datos numéricos y categóricos, número de características y variable de salida; en este caso un comportamiento normal o anómalo en la red. Se muestran las tablas 1 y 2 con los resultados de los análisis realizados a los conjuntos KDD CUP99 y NSL-KDD, ambos en versiones reducidas.

Tabla 1. Análisis al conjunto KDD CUP99.

Cantidad de columnas:	42
Cantidad de filas:	25192
Cantidad de columnas con datos numéricos:	38
Cantidad de datos nulos:	0
Cantidad de columnas redundantes:	1
Cantidad de columnas con datos categóricos:	4
Cantidad de filas etiquetadas como tráfico normal:	13449
Cantidad de filas etiquetadas como tráfico anómalo:	11743

Tabla 2. Análisis al conjunto NSL-KDD.

Cantidad de columnas:	42
Cantidad de filas:	125972
Cantidad columnas con datos numéricos:	38
Cantidad de datos nulos:	0
Cantidad de columnas redundantes:	1
Cantidad de columnas con datos categóricos:	4
Cantidad de filas etiquetadas como tráfico normal:	67343
Cantidad de filas etiquetadas como tráfico anómalo:	58630

En las tablas mostradas a continuación se realiza una descripción de varios de atributos que forman los conjuntos de datos seleccionados.

Tabla 3. Atributos básicos que caracterizan conexiones de redes.

Atributos básicos de las conexiones TCP		
Atributo	**Descripción**	**Tipo**

Duration	Longitud (número de segundos) de la conexión.	Continuo
protocol_type	Tipo de protocolo (tcp, icmp, udp).	Discreto
Service	Tipo de servicio de destino (HTTP, Telnet, SMTP...).	Discreto
src_bytes	Número de bytes de datos de fuente a destino.	Continuo
dst_bytes	Número de bytes de datos de destino a la fuente.	Continuo
Flag	Estado de la conexión (SF, SI, REJ...).	Discreto
Land	1 si la conexión corresponde mismo host/puerto; 0 de otro modo.	Discreto
wrong_fragment	Número de fragmentos erróneos.	Continuo
Urgent	Número de paquetes urgentes.	Continuo

Tabla 4. Atributos especiales que caracterizan conexiones de redes.

Atributos Especiales		
Atributo	**Descripción**	**Tipo**
Hot	Número de indicadores “hot”.	Continuo
num_failed_logins	Número de intentos de acceso fallidos.	Continuo
logged_in	1 si acceso exitoso; 0 de otro modo.	Discreto
num_compromised	Número de condiciones “sospechosas”.	Continuo
root_shell	1 si se obtiene superusuario para acceso a root; 0 de otro modo.	Discreto
su_attempted	1 si se intenta el comando “su root”; 0 de otro modo.	Discreto
num_root	Número de accesos a root.	Continuo
num_file_creations	Número de operaciones de creación de ficheros.	Continuo
num_shells	Número de shell prompts.	Continuo
num_access_files	Número de operaciones de control de acceso a ficheros.	Continuo

num_outbound_cmds	Número de comandos de salida en una sesión ftp	Continuo
is_hot_login	1 si el login pertenece a la lista “hot”; 0 de otro modo.	Discreto
is_guest_login	1 si el acceso es un “guest” login; 0 de otro modo.	Discreto

Tabla 5. Atributos categóricos y sus respectivos valores.

Atributos de tipo simbólico y sus respectivos valores	
Atributo	**Valores**
protocol_type	tcp, udp, icmp.
Service	http, mtp, smtp, finger, domain, domain_u, auth, telnet, ftp, eco_i, ntp_u, ecr_i, other, private, pop_3, ftp_data, rje, time, link, remote_job, gopher, ssh, name, whois, login, imap4, daytime, ctf, nntp, shell, IRC, nnsp, http_443, exec, printer, icmp, efs, courier, uucp, klogin, kshell, echo, discard, systat, supdup, iso_tsap, hostnames, csnet_ns, pop_2, sunrpc, uucp_path, netbios_ns, netbios_ssn, netbios_dgm, sql_net, vmnet, bgp, Z39_50, ldap, netstat, urh_i, X11, urp_i, pm_dump, tftp_u, tim_i, red_i .
Flag	SF, S1, REJ, S2, S0, S3, RSTO, RSTR, RSTOS0, OTH, SH.
class	normal, anomaly

Los atributos anteriormente descritos pueden ser de dos tipos: numéricos (real o entero) o simbólicos (valores discretos a partir de una lista especificada).

Los datos procedentes de los conjuntos de datos deberían estar en el rango [0 a 1] o [-1 a 1]. Sin embargo, no lo están, debido a que todas las conexiones en sus 41 características poseen valores continuos, discretos o simbólicos y en diferentes rangos de significación. Con el propósito de estandarizar dichos valores para que puedan ser eficazmente procesados por los algoritmos de aprendizaje automático, se debe hacer un análisis y normalización de los datos contenidos en las conexiones (Fan et al., 2021).

En el tratamiento de valores nulos, se procede a realizar una eliminación de los mismos. Sin embargo, este método puede llevar a la pérdida de información si los registros con valores nulos son significativos, en ese caso se le asigna un valor que puede ser la media o mediana de la columna perteneciente. En el caso de valores redundantes se eliminan porque no aportan información relevante (Ibañez Monteagudo, 2018), (Fan et al., 2021).

Para la conversión de los datos categóricos en formato numérico, un código entero se asigna a cada categoría. Por ejemplo, en el caso de la característica protocol_type, se asigna “0” a TCP, “1” a UDP, y “2” a ICMP. Por otra parte, existen características cuyos valores se extienden en un rango de números enteros muy grande (De la Hoz Franco, 2016).

Así, src_bytes toma valores entre 0 y más de 1 millón igual que dst_bytes. Se aplica entonces una escala a estas características para reducir el rango de estas. Todas las demás características son booleanas, en el rango de [0.0 a 1.0]. Por lo tanto, el escalado no es necesario para estos atributos (De la Hoz Franco, 2016).

En cuanto a la subfase de reducción de dimensionalidad se deben identificar las técnicas de selección o reducción de características a utilizar, debido a que no es conveniente efectuar el entrenamiento y prueba con la totalidad de características dado que ello podría ralentizar considerablemente el procesamiento, sin suponer una mejora significativa en la clasificación. En muchos conjuntos de datos, puede haber cientos o incluso miles de características predictivas, lo que puede ser un desafío para los algoritmos de aprendizaje automático y causar un mal entrenamiento y como resultado una clasificación ineficiente (De la Hoz Franco, 2016).

Con una disminución de la cantidad de características se mejora el coste de cálculo necesario, se reduce la complejidad del modelo y mejora la precisión del modelo (Amrita & Ahmed, 2012). Las técnicas escogidas son: IG, Correlación de Pearson’s, ANOVA, RFE y PCA en base al estudio de publicaciones asociadas al tema (Taher et al., 2019), (Maza & Touahria, 2018), (Thakkar & Lohiya, 2021), (Preyanka Lakshme & Kumar, 2023).

Como ya se ha indicado, la fase de preprocesamiento implica la ejecución de dos etapas la normalización y la selección o extracción de características. Para la implementación de esta

última es necesario seleccionar los métodos de reducción de dimensionalidad que permitan efectuar el proceso de identificación de características, de tal forma que se determine el nivel de relevancia de las mismas, con el propósito de no utilizar la totalidad de las características del conjunto de datos, sino sólo aquellas que incidan directamente en el proceso de clasificación y en consecuencia disminuir la carga computacional que agregaría al proceso, el innecesario uso de la totalidad de las características.

2.1.2. Fase de entrenamiento

En esta fase, en primera instancia se procede a realizar una división de los conjuntos de datos en conjunto de datos de entrenamiento, que es el 70% del conjunto original y conjunto de datos de pruebas, que utiliza el restante 30%. Al dividir los conjuntos de esta manera se garantiza que se ajusten los parámetros del modelo con el conjunto de entrenamiento y con el conjunto de pruebas se verifique y evalúe su funcionamiento porque este conjunto contiene datos no vistos durante el entrenamiento (De la Hoz Franco, 2016). En las tablas siguientes los resultados de realizar las divisiones correspondientes a KDD CUP99 y NSL-KDD:

Tabla 6. Resultado de la división del conjunto KDD CUP99.

División del conjunto KDD CUP99	
Conjunto de entrenamiento	**Cantidad**
Filas etiquetadas como tráfico normal:	9407
Filas etiquetadas como tráfico anómalo:	8227
Conjunto de pruebas	**Cantidad**
Filas etiquetadas como tráfico normal:	4042
Filas etiquetadas como tráfico anómalo:	3516

Tabla 7. Resultado de la división del conjunto NSL-KDD.

División del conjunto NSL-KDD

Conjunto de entrenamiento	**Cantidad**
Filas etiquetadas como tráfico normal:	47620
Filas etiquetadas como tráfico anómalo:	40921
Conjunto de pruebas	**Cantidad**
Filas etiquetadas como tráfico normal:	20083
Filas etiquetadas como tráfico anómalo:	17709

Los algoritmos escogidos para la clasificación son: Regresión Logística, Árboles de Decisión, Redes Bayesianas, Máquinas de Vectores de Soporte y K-Vecinos más Cercanos. Estos métodos fueron escogidos en base a otros estudios y trabajos sobre el tema (Rivero Pérez et al., 2016), (Zhou et al., 2020), (Ahmad et al., 2021), (Liu & Lang, 2019).

Después se procede a entrenar los clasificadores considerados, a partir del tipo de algoritmo de aprendizaje seleccionado y utilizando los conjuntos de datos, los cuales están normalizados y reducidos a las características relevantes para generar un aprendizaje eficiente. El proceso de entrenamiento se realiza con el conjunto de datos de entrenamiento correspondiente, que no es el mismo a utilizar en el proceso de pruebas, producto de ejecuciones iterativas que generarán un modelo de aprendizaje.

2.1.3. Fase de pruebas y clasificación

Una vez el modelo está entrenado, se procede con la fase de clasificación en la que el clasificador determina que tráfico es normal o anómalo, efectuando la subsiguiente clasificación de cada una de las conexiones del conjunto de datos de pruebas. Debido a este paso, se presentará la información de resumen del proceso, de forma estadística, calculando las métricas de desempeño que van a permitir evaluar los clasificadores (De la Hoz Franco, 2016).

2.2. Patrones de diseño

Durante las fases analizadas se ve presente el uso de patrones de diseño específicos en el Aprendizaje Automático, los cuales difieren de los patrones de diseño convencionales de la programación orientada a objetos. Estos patrones de diseño aplicados al Aprendizaje Automático recogen las mejores prácticas y soluciones a problemas habituales. (Valliappa Lakshmanan et al., 2021). Algunos de ellos son:

- El patrón de diseño de incrustaciones (*"Embeddings"*) es una representación de datos que se pueden aprender, que mapea datos de alta cardinalidad, en un espacio de menor dimensión de tal forma que se conserva la información relevante para el problema de aprendizaje. Las incrustaciones son el núcleo del Aprendizaje Automático actual y tienen varias encarnaciones en este campo (Valliappa Lakshmanan et al., 2021). Este patrón se ve representado en el uso del método LabelEncoder.

Los modelos de aprendizaje automático buscan patrones en los datos que capten cómo las propiedades de las características de entrada del modelo se relacionan con la etiqueta de salida. Mientras que el manejo de datos de entrada estructurados y numéricos es bastante sencillo, los datos necesarios para entrenar un modelo de aprendizaje automático pueden presentarse en una miríada de variedades, como características categóricas, texto, imágenes, audio, series temporales y muchos más. Para estas representaciones de datos, se necesita un valor numérico significativo que suministrar a nuestro modelo de aprendizaje automático.

- El patrón de diseño de grupos (*"Ensembles"*) hace referencia a las técnicas de aprendizaje automático que combinan varios modelos de aprendizaje automático y agregan sus resultados para realizar predicciones. Los conjuntos pueden ser un medio eficaz para mejorar el rendimiento y producir predicciones mejores que cualquier modelo individual (Valliappa Lakshmanan et al., 2021). Este patrón se ve representado en el uso del método RandomForestClassifier.
- El patrón de diseño de división repetible (*"Repeatable Splitting"*) se usa para garantizar que el muestreo sea repetible y reproducible, es necesario utilizar una

columna bien distribuida y una función determinista para dividir los datos disponibles en conjuntos de datos de entrenamiento y de prueba (Valliappa Lakshmanan et al., 2021). Este patrón se ve representado en el uso del método train_test_split.

- El patrón de diseño de reequilibrado (*"Rebalancing"*) proporciona varios enfoques para manejar conjuntos de datos que están inherentemente desequilibrados. Esto se refiere a conjuntos de datos en los que una etiqueta constituye la mayor parte del conjunto de datos, dejando muchos menos ejemplos de otras etiquetas (Valliappa Lakshmanan et al., 2021). Este patrón se ve representado en el uso del método train_test_split.

2.3. Tecnologías y herramientas a utilizar

Como lenguaje de programación se empleó Python. Python es un lenguaje de programación de propósito general. Fue diseñado y desarrollado para escribir software para una amplia variedad de disciplinas. Python se ha utilizado para escribir aplicaciones destinadas a resolver problemas en biología, química, análisis financiero, análisis numérico, robótica y muchos otros campos. También se utiliza mucho como lenguaje de scripting para uso de administradores informáticos, que lo emplean para capturar y reproducir secuencias de comandos informáticos (Kalb, 2018).

Como editor de código fuente se empleó Visual Studio Code. Este es un editor de código fuente ligero pero potente que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como C++, Java y Python). Es un editor de código optimizado con soporte para operaciones de desarrollo como depuración, ejecución de tareas y control de versiones (Bin Uzayr, 2022).

Como entorno de ejecución se usó Colaboratory, o "Colab" para abreviar, el cual es un producto de Google Research. Permite a cualquier usuario escribir y ejecutar código arbitrario de Python en el navegador. Es adecuado para tareas de aprendizaje automático, análisis de datos y educación. Desde un punto de vista más técnico, Colab es un servicio de cuaderno alojado de Jupyter que no requiere configuración y que ofrece acceso sin coste adicional a recursos informáticos (Naik et al., 2021). Provee en su versión sin coste un

entorno de ejecución con 12GB de memoria RAM, un procesador Intel(R) Xeon(R) CPU @ 2.20GHz de 2 núcleos y 12 horas de ejecución ininterrumpidas.

Se usó Anaconda, que es una distribución de los lenguajes de programación Python y R para computación científica (ciencia de datos, aplicaciones de aprendizaje automático, procesamiento de datos a gran escala, análisis predictivo, etc.). Tiene como ventaja simplificar la gestión e implementación de paquetes (Yan & Yan, 2018).

2.4. Conclusiones del capítulo

Con la realización de este capítulo se obtienen las siguientes conclusiones:

- La fase de preprocesamiento es crucial ya que permite preparar los datos para su uso en el entrenamiento del modelo.
- La fase de entrenamiento es donde se utiliza el algoritmo de aprendizaje automático para modelar la relación entre los datos de entrada y la variable respuesta.
- La fase de pruebas es esencial para evaluar el rendimiento del modelo y determinar si alcanza los resultados deseados.
- Los patrones de diseño utilizados en Aprendizaje Automático encapsulan las técnicas y enfoques más efectivos, así como las soluciones probadas a desafíos comunes.

CAPÍTULO 3: RESULTADOS EXPERIMENTALES DEL ESTUDIO APLICADO

A lo largo de este capítulo se hará énfasis en los resultados de la última fase del proceso de detección de intrusiones.

3.1. Resultados

En las tablas siguientes se puede apreciar las características más importantes después de aplicar reducción de dimensionalidad.

Tabla 8. Características relevantes en el orden de 9 características usando RFE.

Top 9 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • protocol_type • service • flag

Tabla 9. Características relevantes en el orden de 12 características usando RFE.

Top 12 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • count • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • dst_host_diff_srv_rate • dst_host_same_src_port_rate • protocol_type • service • flag

Tabla 10. Características relevantes en el orden de 15 características usando RFE.

Top 15 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • logged_in • count • srv_count • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • dst_host_diff_srv_rate • dst_host_same_src_port_rate • dst_host_srv_diff_host_rate • protocol_type • service • flag

Los restantes resultados para los grupos de 9, 12 y 15 características seleccionadas de los métodos Correlación de Pearson's, IG, ANOVA se encuentran en los anexos en las tablas desde la 27 hasta la 35 respectivamente.

Luego de aplicar los distintos tipos de métodos de selección o extracción de las características; en el orden de 9, 12 y 15 características escogidas, para el entrenamiento de los algoritmos de aprendizaje automático de clasificación; se obtuvieron los siguientes resultados, usando como métrica de evaluación la efectividad:

Tabla 11. Efectividad de los clasificadores utilizados en NSL-KDD.

Métodos de Reducción de dimensionalidad/Clasificación (l = 9, 12, 15)	**Regresión Logística**	**KNN**	**Árboles de decisión**	**SVM**	**Naive Bayes**
IG	88.55% 89.81% 91.51%	98.88% 99.07% 98.94%	98.76% 98.81% 98.82%	89.28% 89.71% 90.45%	88.11% 88.29% 88.75%
ANOVA	90.22% 93.18%	94.94% 98.53%	93.62% 96.45%	90.37% 93.22%	87.51% 88.69%

	93.60%	99.21%	97.44%	93.74%	88.98%
RFE	92.74% 94.26% 94.28%	98.33% 98.97% 99.28%	98.06% 98.39% 98.39%	93.00% 93.49% 93.96%	87.92% 89.41% 91.33%
PCA	93.43% 93.54% 93.55%	99.54% 99.55% 99.56%	97.34% 97.43% 97.58%	92.27% 93.00% 93.35%	83.23% 87.36% 87.17%
Correlación	89.19% 89.48% 89.99%	98.31% 98.98% 99.24%	96.40% 95.82% 96.40%	87.79% 87.09% 93.42%	88.54% 88.25% 89.49%

Tabla 12. Efectividad de los clasificadores utilizados en KDD CUP99.

Métodos de Reducción de dimensionalidad/Clasificación (l = 9, 12, 15)	**Regresión Logística**	**KNN**	**Árboles de decisión**	**SVM**	**Naive Bayes**
IG	89.94% 89.90% 91.78%	98.27% 98.29% 98.43%	98.50% 98.82% 98.80%	81.95% 87.90% 90.01%	87.80% 88.64% 90.16%
ANOVA	90.34% 93.67% 93.94%	93.87% 97.96% 98.55%	94.11% 96.93% 97.52%	90.46% 93.95% 93.76%	88.07% 89.46% 89.90%
RFE	92.98% 94.32% 94.54%	98.22% 98.65% 98.84%	98.16% 98.27% 98.34%	93.38% 94.33% 94.39%	90.22% 90.84% 91.88%
PCA	93.83% 93.97%	99.06% 99.08%	97.47% 97.30%	93.43% 93.80%	89.30% 91.36%

	94.04%	99.08%	97.63%	93.64%	91.13%
Correlación	90.15% 90.18% 92.88%	97.96% 98.55% 98.80%	95.15% 95.30% 95.91%	89.09% 85.35% 87.43%	89.56% 89.28% 90.43%

Como se puede apreciar en las tablas, con 15 características, de 41, se alcanza buena efectividad, lo que significa que esas características recogen información importante para la clasificación. Por lo que se escoge este grupo. Para escoger el mejor método de reducción de características se calculó la media aritmética que poseen sus clasificadores. Las tablas 13 y 14 muestran los resultados.

Tabla 13. Media aritmética de la efectividad causada por los métodos de clasificación en el conjunto KDD CUP99.

Métodos de reducción de dimensionalidad	**Media aritmética**
IG	93.84%
ANOVA	94.73%
RFE	95.60%
PCA	95.10%
Correlación	93.09%

Tabla 14. Media aritmética de la efectividad causada por los métodos de clasificación en el conjunto NSL-KDD.

Métodos de reducción de características	**Media aritmética**
IG	93.69%
ANOVA	94.59%
RFE	95.45%

PCA	94.24%
Correlación	93.70%

En ambos conjuntos debido a su análoga estructura se aprecian resultados similares. En el conjunto KDD CUP99 el método de mejores resultados es RFE y en el conjunto NSL-KDD el mejor método también es RFE. Para escoger el mejor clasificador se usan las restantes métricas, además de la exactitud. En las tablas 15 y 16 se muestran los clasificadores con sus respectivas métricas.

Tabla 15. Métricas de los clasificadores en el conjunto KDD CUP99 usando el método RFE.

Clasificadores	**Exactitud**	**Precisión**	**Sensibilidad**	**F1-score**
Regresión Logística	94.54%	95.14%	93.03%	94.07%
KNN	98.84%	98.66%	98.86%	98.76%
Árboles de decisión	98.34%	98.65%	97.78%	98.21%
SVM	94.39%	96.84%	90.89%	93.77%
Naive Bayes	91.88%	92.06%	90.35%	91.20%

Tabla 16. Métricas de los clasificadores en el conjunto NSL-KDD usando el método RFE.

Clasificadores	**Exactitud**	**Precisión**	**Sensibilidad**	**F1-score**
Regresión Logística	94.28%	94.96%	92.73%	93.83%
KNN	99.28%	99.20%	99.26%	99.23%
Árboles de Decisión	98.39%	99.41%	97.14%	98.26%
SVM	93.96%	93.85%	93.21%	93.53%
Naive Bayes	91.33%	93.37%	87.72%	90.46%

Los resultados obtenidos del análisis de las tablas muestran que KNN tiene el mejor rendimiento, seguido por DT. Regresión Logística y SVM presentaron un desempeño moderado y Naive Bayes fue el de peores resultados. Los restantes resultados de los clasificadores se muestran en los anexos en las tablas 17, 18, 19, 20, 21 para el conjunto KDD CUP99 y en las tablas 22, 23, 25, 25 y 26 para el conjunto NSL-KDD.

Por lo que, para este estudio, con los conjuntos de datos utilizados, los métodos de reducción de dimensionalidad y clasificación evaluados, la mejor combinación para obtener la mejor eficiencia de los modelos en la detección de intrusiones es: RFE + KNN, porque logró clasificar correctamente el 99% de las observaciones, detectar el 99% de los casos positivos y posee el mejor rendimiento de manera general.

3.2. Conclusiones del capítulo

En este capítulo se ha demostrado que la combinación de técnicas de reducción de dimensionalidad y clasificación del aprendizaje automático pueden ser efectiva para la detección de intrusiones. Los resultados obtenidos sugieren que el uso de 15 características y el método de reducción de características RFE son los más efectivos para la tarea de clasificación. Además, el algoritmo KNN resultó ser el mejor clasificador para el problema planteado.

CONCLUSIONES GENERALES

Al término de este trabajo de investigación se pudo constatar que la aplicación de los métodos de aprendizaje automático a la detección de intrusiones constituye un campo abierto de investigación, con importantes retos a superar. Varios autores han planteado soluciones ingeniosas para mejorar la efectividad de los algoritmos de clasificación en este ámbito de aplicación.

Los métodos de reducción de dimensionalidad mejoran el rendimiento de los clasificadores aplicados sobre los conjuntos de datos de intrusiones que se utilizaron para este estudio. En especial la selección de características mostró resultados alentadores para próximas investigaciones.

Se pudo observar que el método RFE, combinado con el clasificador KNN exhibió los mejores resultados.

RECOMENDACIONES

El alcance del presente trabajo estuvo limitado a métodos discretos de reducción de dimensionalidad, y a una selección arbitraria de algoritmos, motivado fundamentalmente por la potencia de cálculo computacional de la que se pudo disponer. Esto también obligó a seleccionar dos de los conjuntos de datos mejor referenciados en la literatura. Por ello recomendamos para trabajos posteriores:

- Ampliar la muestra de métodos de reducción de dimensionalidad, tanto de selección como de reducción de características.
- Ampliar la muestra de algoritmos de clasificación aplicados a la detección de intrusiones, para evaluar con mayor rigor su eficiencia ante cada resultado emitido por los métodos de reducción.
- Extender el uso de todas las métricas disponibles para evaluar los clasificadores.
- Proponer un método no empírico de determinar las cantidades de características para la ejecución de la reducción de dimensionalidad.
- Proponer un nuevo método compuesto de reducción de dimensionalidad que tenga un impacto mayor en el rendimiento de los clasificadores.

REFERENCIAS BIBLIOGRÁFICAS

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1), Article 1.

Alazzam, H., Sharieh, A., & Sabri, K. E. (2020). A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. *Expert systems with applications*, *148*, 113249.

Alba Vega, D. A., & Calle Jara, J. F. (2020). *Aplicación de Técnicas de Machine Learning basado en información sísmica para profundizar la probabilidad mediante el uso de Regresión Logística y Redes Neuronales.* Universidad de Guayaquil.

Amrita, A., & Ahmed, P. (2012). A study of feature selection methods in intrusion detection system: A survey. *Int. J. Comput. Sci. Eng. Info. Technol. Res.(IJCSEITR)*, *2*(3), Article 3.

Ariza Gacharná, J. A. (2022). *Detección de ataques de reconocimiento en Redes IoT empleando modelos de Machine Learning*. Universidad de los Andes.

Avila, J., & Hauck, T. (2017). *Scikit-Learn Cookbook* (2nd Edition). Packt Publishing.

Ávila Pérez, M. (2020). Implementación de un IDS. *Gestión, Competitividad e Innovación*, *8*(1), 11-23.

Belén Del Río, D. (2021). *Sistema de Detección de Intrusiones y Anomalías en un CPD*. Universidad de Cantabria.

Bin Uzayr, S. (2022). *Mastering Visual Studio Code: A Beginner's Guide* (1st Edition). Mastering Computer Science.

Cai, J., Luo, J., Wang, S., & Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, *300*, 70-79.

Calderón Romero, A., & Hurtado Cortes, H. (2019). Machine learning en la detección de enfermedades en plantas. *Tecnología, Investigación y Academia TIA*, *7*(2), 55-61.

Castellanos Leyva, O., & García Borroto, M. (2019). Análisis y caracterización de conjuntos de datos para la detección de intrusiones. *02/04/2020*, *13*(4), 39-52.

Charfaoui, Y. (2020). *Feature Engineering and Feature Selection with Python—A Practical Guide For Feature Crafting*.

De la Hoz Franco, E. (2016). *Sistemas de Detección de Intrusos con Mapas Autorganizativos Probabilísticos y Optimización Multiobjetivo.* [Tesis Doctoral, Universidad de Granada]. http://hdl.handle.net/10481/43551

Dhabliya, D. (2021). Feature Selection Intrusion Detection System for The Attack Classification with Data Summarization. *Machine Learning Applications in Engineering Education and Management*, *1*(1), Article 1.

Ding, H., Feng, P.-M., Chen, W., & Lin, H. (2014). Identification of bacteriophage virion proteins by the ANOVA feature selection and analysis. *Molecular BioSystems*, *10*, 2229--2235. https://doi.org/10.1039/c4mb00316k

Duboue, P. (2020). *The Art of Feature Engineering: Essentials for Machine Learning* (1st Edition). Cambridge University Press.

E. Ramírez, P., & E. Grandón, E. (2018). Predicción de la Deserción Académica en una Universidad Pública Chilena a través de la Clasificación basada en Árboles de Decisión con Parámetros Optimizados. *Formación Universitaria*, *11*(3), 3-10. http://dx.doi.org/10.4067/S0718-50062018000300003

Fan, C., Meiling, C., Wang, X., Wang, J., & Huang, B. (2021). A Review on Data Preprocessing Techniques Toward Efficient and Reliable Knowledge Discovery From Building Operational Data. *Frontiers in Energy Research*, *9*. https://doi.org/10.3389/fenrg.2021.652801

Fossaceca, J. M., Mazzuchi, T. A., & Sarkani, S. (2015). MARK-ELM: Application of a novel Multiple Kernel Learning framework for improving the robustness of Network Intrusion Detection. *Expert Systems with Applications*, *42*(8), Article 8.

Haipeng Yao, Chunxiao Jiang, Yi Qian. (2019). *Developing Networks using Artificial Intelligence* (1.ª ed.). Springer Cham.

Herrera Semenets, V. (2019). *Algoritmo para la reducción de datos y su aplicación a la detección de intrusos.* Universidad de las Ciencias Informáticas.

Ibañez Monteagudo, P. (2018). *Implementation of Machine Learning techniques and Artificial Intelligence for the intrusion detection in communications networks*. Politechnika Krakowska.

Jabali, V. K., Rahbari, M., & Kashkouli, A. (2017). Taxonomy of feature selection in intrusion detection system. *International Journal of Computer Science and Network Security*, *17*(6), Article 6.

Jain, D., & Singh, V. (2018). An efficient hybrid feature selection model for dimensionality reduction. *Procedia Computer Science*, *132*, 333-341.

Kalb, I. (2018). *Learn to Program with Python 3: A Step-by-Step Guide to Programming* (2nd Edition). Apress.

Kasongo, S. M., & Sun, Y. (2019). A deep learning method with filter based feature engineering for wireless intrusion detection system. *IEEE access*, *7*, 38597-38607.

Kasongo, S. M., & Sun, Y. (2020). A deep learning method with wrapper based feature extraction for wireless intrusion detection system. *Computers & Security*, *92*, 101752.

Keshta, I. M. (2018). Intelligent intrusion detection system based on MLP, RBF and SVM classification algorithms: A comparative study. *International Journal of Computer Science and Information Security*, *16*(5), Article 5.

Khun, M., & Johnson, K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models* (1st Edition). Chapman and Hall/CRC.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM computing surveys (CSUR)*, *50*(6), Article 6.

Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences*, *9*(20), Article 20.

Mario Aragonés Lozano. (2020). *IMPLEMENTACIÓN DE UN SISTEMA DE DETECCIÓN DE INTRUSOS EN REDES IP CON INTELIGENCIA ARTIFICIAL*. Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València.

Martínez Martínez, L. (2021). *Comparativa de Modelos de Machine Learning para la estimación de parámetros de interés empleando datos de la European Soil Database*. Universidad de Valladolid.

Masdari, M., & Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. *Applied Soft Computing*, *92*, 106301. https://doi.org/10.1016/j.asoc.2020.106301

Maza, S., & Touahria, M. (2018). Feature selection algorithms in intrusion detection system: A survey. *KSII Transactions on Internet and Information Systems (TIIS)*, *12*(10), Article 10.

Montoya Guirado, D. (2018). *Machine learning mediante Microsoft Azure: Una aplicación sobre real-state.* Universitat de Barcelona.

Naik, P. G., Naik, G. R., & Patil, M. B. (2021). *Conceptualizing Python in Google COLAB* (1st Edition). Shashwat Publication.

Prateek, G. (2021). *Practical Data Science with Jupyter: Explore Data Cleaning, Pre-processing, Data Wrangling, Feature Engineering and Machine Learning using Python and Jupyter* (2nd Edition). BPB Publications.

Preyanka Lakshme, R., & Kumar, S. G. (2023). *A Review Based on Machine Learning for Feature Selection and Feature Extraction*. 144-157.

Rivero Pérez, J. L., Ribeiro, B., & Ortiz, K. H. (2016). Comparación de algoritmos para detección de intrusos en entornos estacionarios y de flujo de datos. *Universidad y Sociedad*, *8*(4), 32-42.

Sahu, B., Dehuri, S., & Jagadev, A. (2018). A study on the relevance of feature selection methods in microarray data. *The Open Bioinformatics Journal*, *11*(1), Article 1.

Stuart J. Russell & Peter Norvig. (2021). *Artificial Intelligence: A Modern Approach, Global Edition* (4.ª ed.). Pearson.

Taher, K. A., Jisan, B. M. Y., & Rahman, M. M. (2019). *Network intrusion detection using supervised machine learning technique with feature selection*. 643-646.

Tari, Z., Fahad, A., Yi, X., & Almalawi, A. (2020). *Network Classification for Traffic Management: Anomaly Detection, Feature Selection, Clustering and Classification*. Computing and Networks.

Thakkar, A., & Lohiya, R. (2021). Attack classification using feature selection techniques: A comparative study. *Journal of Ambient Intelligence and Humanized Computing*, *12*, 1249-1266.

Trapero Estepa, M. D. (2021). *Clasificación de Ataques a una Red de Telecomunicación con Deep Learning*. Universidad de Sevilla.

Umbarkar, S., & Shukla, S. (2018). *Analysis of heuristic based feature reduction method in intrusion detection system*. 717-720.

Valdezate Álvarez, D. G. (2020). *Sistemas de Detección de Intrusos Basados en Técnicas de Machine Learning.* Universidad de Valladolid.

Valliappa Lakshmanan, Robinson, S., & Munn, M. (2021). *Machine Learning Design Patterns Solutions to Common Challenges in Data Preparation, Model Building, and MLOps* (1st ed.). O'Reilly Media, Inc.

Xiaosong, H., Yunhong, C., Xianke, L., & Simona, O. (2020). Battery health prediction using fusion-based feature selection and machine learning. *IEEE*, *7*(2), 382-398. https://doi.org/10.1109/TTE.2020.3017090

Yan, Y., & Yan, J. (2018). *Hands-On Data Science with Anaconda: Utilize the right mix of tools to create high-performance data science applications* (1st Edition). Packt Publishing.

Zhou, Y., Cheng, G., Jiang, S., & Dai, M. (2020). Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Computer networks*, *174*, 107247.

ANEXOS

A1. Tabla 17 - Reporte de clasificación usando Regresión logística en KDD CUP99.

	precision	recall	f1-score	support
anomaly	0.95	0.93	0.94	3516
normal	0.94	0.96	0.95	4042
accuracy			0.95	7558
macro avg	0.95	0.94	0.95	7558
weighted	0.95	0.95	0.95	7558

A2. Tabla 18 - Reporte de clasificación usando KNN en KDD CUP99.

	precision	recall	f1-score	support
anomaly	0.99	0.99	0.99	3516
normal	0.99	0.99	0.99	4042
accuracy			0.99	7558
macro avg	0.99	0.99	0.99	7558
weighted	0.99	0.99	0.99	7558

A3. Tabla 19 - Reporte de clasificación usando Árboles de decisiones en KDD CUP99.

	precision	recall	f1-score	support
anomaly	0.99	0.98	0.98	3516
normal	0.98	0.99	0.98	4042
accuracy			0.98	7558
macro avg	0.98	0.98	0.98	7558

weighted	0.98	0.98	0.98	7558

A4. Tabla 20 - Reporte de clasificación usando SVM en KDD CUP99.

	precision	recall	f1-score	support
anomaly	0.97	0.91	0.94	3516
normal	0.92	0.97	0.95	4042
accuracy			0.94	7558
macro avg	0.95	0.94	0.94	7558
weighted	0.95	0.94	0.94	7558

A5. Tabla 21 – Reporte de clasificación usando Naive Bayes en KDD CUP99.

	precision	recall	f1-score	support
anomaly	0.92	0.90	0.91	3516
normal	0.92	0.93	0.92	4042
accuracy			0.92	7558
macro avg	0.92	0.92	0.92	7558
weighted	0.92	0.92	0.92	7558

A6. Tabla 22 - Reporte de clasificación usando Regresión logística en NSL-KDD.

	precision	recall	f1-score	support
anomaly	0.95	0.93	0.94	17709
normal	0.94	0.96	0.95	20083
accuracy			0.94	37792

macro avg	0.94	0.94	0.94	37792
weighted	0.94	0.94	0.94	37792

A7. Tabla 23 - Reporte de clasificación usando KNN en NSL-KDD.

	precision	recall	f1-score	support
anomaly	0.99	0.99	0.99	17709
normal				20083
accuracy			0.99	37792
macro avg	0.99	0.99	0.99	37792
weighted	0.99	0.99	0.99	37792

A8. Tabla 24 - Reporte de clasificación usando Árboles de decisiones en NSL-KDD.

	precision	recall	f1-score	support
anomaly	0.99	0.97	0.98	17709
normal	0.98	0.99	0.99	20083
accuracy			0.98	37792
macro avg	0.98	0.98	0.98	37792
weighted	0.98	0.98	0.98	37792

A9. Tabla 25 - Reporte de clasificación usando SVM en NSL-KDD.

	precision	recall	f1-score	support
anomaly	0.94	0.93	0.94	17709
normal	0.94	0.95	0.94	20083

accuracy			0.94	37792
macro avg	0.94	0.94	0.94	37792
weighted	0.94	0.94	0.94	37792

A10. Tabla 26 - Reporte de clasificación usando Naive Bayes en NSL-KDD.

	precision	recall	f1-score	support
anomaly	0.93	0.88	0.90	17709
normal	0.90	0.95	0.92	20083
accuracy			0.91	37792
macro avg	0.92	0.91	0.91	37792
weighted	0.91	0.91	0.91	37792

A11. Tabla 27. Características relevantes en el orden de 9 características usando Correlación de Pearson's.

Top 9 características más relevantes que inciden sobre la variable de salida:	• dst_host_srv_serror_rate • dst_host_serror_rate • serror_rate • srv_serror_rate • count • dst_host_count • service • srv_rerror_rate • dst_host_srv_rerror_rate

A12. Tabla 28. Características relevantes en el orden de 12 características usando Correlación de Pearson's.

Top 12 características más relevantes que inciden sobre la variable de salida:	• dst_host_srv_serror_rate • dst_host_serror_rate • serror_rate • srv_serror_rate

	• count • dst_host_count • service • srv_rerror_rate • dst_host_srv_rerror_rate • rerror_rate • dst_host_rerror_rate • dst_host_diff_srv_rate

A13. Tabla 29. Características relevantes en el orden de 15 características usando Correlación de Pearson's.

Top 15 características más relevantes que inciden sobre la variable de salida:	• dst_host_srv_serror_rate • dst_host_serror_rate • serror_rate • srv_serror_rate • count • dst_host_count • service • srv_rerror_rate • dst_host_srv_rerror_rate • rerror_rate • dst_host_rerror_rate • dst_host_diff_srv_rate • diff_srv_rate • wrong_fragment • dst_host_same_src_port_rate

A14. Tabla 30. Características relevantes en el orden de 9 características usando IG.

Top 9 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • logged_in • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • service • flag

A15. Tabla 31. Características relevantes en el orden de 12 características usando IG.

Top 12 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • logged_in • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • dst_host_diff_srv_rate • dst_host_serror_rate • dst_host_srv_serror_rate • service • flag

A16. Tabla 32. Características relevantes en el orden de 15 características usando IG.

Top 15 características más relevantes que inciden sobre la variable de salida:	• src_bytes • dst_bytes • logged_in • count • serror_rate • srv_serror_rate • same_srv_rate • diff_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • dst_host_diff_srv_rate • dst_host_serror_rate • dst_host_srv_serror_rate • service • flag

A17. Tabla 33. Características relevantes en el orden de 9 características usando ANOVA.

Top 9 características más relevantes que inciden sobre la variable de salida:	• logged_in • serror_rate • srv_serror_rate

	• same_srv_rate • dst_host_srv_count • dst_host_same_srv_rate • dst_host_serror_rate • dst_host_srv_serror_rate • flag

A18. Tabla 34. Características relevantes en el orden de 12 características usando ANOVA.

Top 12 características más relevantes que inciden sobre la variable de salida:	• logged_in • count • serror_rate • srv_serror_rate • same_srv_rate • dst_host_count • dst_host_srv_count • dst_host_same_srv_rate • dst_host_serror_rate • dst_host_srv_serror_rate • protocol_type • flag

A19. Tabla 35. Características relevantes en el orden de 15 características usando ANOVA.

Top 15 características más relevantes que inciden sobre la variable de salida:	• logged_in • count • serror_rate • srv_serror_rate • srv_rerror_rate • same_srv_rate • dst_host_count • dst_host_srv_count • dst_host_same_srv_rate • dst_host_serror_rate • dst_host_srv_serror_rate • dst_host_srv_rerror_rate • protocol_type • service y flag