Trabajo Fin de Grado de Ingeniería de Tecnologías y Servicios de la Telecomunicación

Detección de ataques en redes de sistemas de control industrial mediante técnicas de aprendizaje máquina

Javier Morón Borraz

Director: Ricardo J. Rodríguez Fernández

Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza

Febrero de 2023
Curso 2022/2023

Agradecimientos

A mi familia y amigos.
A Ricardo J. Rodríguez.

Detección de ataques en redes de sistemas de control industrial mediante técnicas de aprendizaje máquina

RESUMEN

Los sistemas de control industrial (ICS) han sufrido un cambio de paradigma que ha llevado de entornos originalmente aislados a entornos altamente conectados, como parte de la evolución de la llamada *Industria 4.0*. Diseñar modelos que se adapten a la detección de ataques en redes de ICS es un desafío para los sistemas de detección de intrusos en red, que tratan de detectar anomalías en el tráfico que representen un riesgo potencial.

El trabajo realizado en este trabajo se divide en dos partes. En primer lugar, se realiza un estudio relacionado con sistemas de control industrial y sistemas de detección de intrusos basados en red. Dicho estudio inicial sirve de base para estructurar el trabajo a realizar. En segundo lugar, se evalúan diferentes algoritmos de lenguaje máquina (supervisados y no supervisados) mediante el uso de distintos conjuntos de datos (concretamente, NSL_KDD, UNSW_NB15 y CIC_IDS2017). Para usar estos conjuntos de datos, se realiza previamente un *análisis exploratorio de datos* con objeto de eliminar problemas de sobreajuste, desequilibro en los datos, y facilitar su procesamiento. Por último, se comprueba la eficacia de los modelos construidos cuando evalúan datos de prueba nunca vistos y se discute si la solución estudiada puede ser válida dentro de un entorno de control industrial. Adicionalmente, se comentan posibles soluciones a implementar en estos sistemas que puedan proteger frente a este tipo de ataques.

La información recopilada en este trabajo permite concluir si un sistema de detección de intrusos basado en lenguaje máquina es capaz de realizar una detección correcta en un entorno de control industrial. Además, también permite concluir otros aspectos como el rendimiento de los algoritmos supervisados frente a los no supervisados o la diferencia entre usar un conjunto de datos distinto para entrenar un modelo en el contexto de entornos de control industrial.

Attack detection in industrial control systems networks through machine learning techniques

ABSTRACT

Industrial Control Systems (ICS) have suffered a paradigm shift that led originally isolated environments to highly connected environments, as part of the evolution of the so-called *Industry 4.0*. Designing models that are suitable for attack detection in ICS networks is a challenge for network-based intrusion detection systems, which try to detect anomalies in traffic that represents a potential risk.

The work done in this Bachelor's Final Degree Project is divided into two parts. First of all, we have conducted a study related to industrial control systems and network-based intrusion detection systems. This study serves as the initial basis for structuring the work to be done. Second, we have evaluated different machine learning algorithms (supervised and unsupervised) using different datasets (specifically, NSL_KDD, UNSW_NB15 and CIC_IDS2017). To use these datasets, we previously performed an *exploratory data analysis* in order to eliminate problems of overfitting, imbalance in the data and to facilitate its processing. Lastly, we measure the efficiency of the models when they predict never-seen before test data, and discuss whether the studied solution can be valid in an industrial control environment. Furthermore, we discuss possible solutions to implement in these systems that can protect against this kind of attacks.

The information collected in this work allows us to conclude if an intrusion detection system based on machine learning is capable of performing a correct detection in an industrial control environment. Besides, it also allows us to conclude other aspects such as the performance of supervised versus unsupervised algorithms or the difference between using a different dataset to train a model in the context of industrial control environments.

Índice

Índice de figuras

Índice de tablas

Lista de acrónimos

AI *Artificial Intelligence*

AUC *Area Under the Curve*

CSV *Comma-Separated Values*

DDoS *Distributed Denial of Service*

DT *Decision Tree*

DoS *Denial of Service*

EDA *Exploratory Data Analysis*

FN *False Negative*

FP *False Positive*

FPR *False Positive Rate*

FTP *File Transfer Protocol*

GBDT *Gradient Boosted Decision Tree*

HTTP *Hypertext Transfer Protocol*

HULK *HTTP Unbearable Load King*

ICS *Industrial Control System*

IDS *Intrusion Detection System*

IF *Isolation Forest*

IMAP *Internet Message Access Protocol*

IoT *Internet of Things*

IP *Internet Protocol*

KM *K-Means Clustering*

KNN *K-Nearest Neighbors*

LR *Logistic Regresion*

MCC *Matthews Correlation Coefficient*

ML *Machine Learning*

NIDS *Network-based Intrusion Detection System*

PCA *Principal Component Analysis*

PCAP *Packet Capture*

RF *Random Forest*

RFE *Recursive Feature Elimination*

ROC *Receiver Operating Characteristic*

R2L *Remote To Local*

SNMP *Simple Network Management Protocol*

SQL *Structured Query Language*

SSH *Secure Shell Protocol*

SVM *Support Vector Machines*

TCP *Transmission Control Protocol*

TFG Trabajo Fin de Grado

TN *True Negative*

TP *True Positive*

TPR *True Positive Rate*

UDP *User Datagram Protocol*

U2R *User To Root*

XML *Extensible Markup Language*

XSS *Cross-site Scripting*

Capítulo 1

Introducción

1.1. Planteamiento del problema

Con la llegada de la Cuarta Revolución Industrial, también conocida como *Industria 4.0* [1], las tecnologías y los entornos implementados por los negocios han sufrido un cambio drástico. Mediante la integración de las tecnologías de la información en el sector industrial, tecnologías como el Internet de las Cosas (IoT) o la Inteligencia Artificial (AI) se encuentran presentes en procesos como la fabricación del producto [2]. Debido a ello, los *sistemas de control industrial* (ICS) han evolucionado de entornos aislados a entornos altamente conectados.

Sin embargo, la alteración provocada por la Industria 4.0 convierte al sector industrial en un sector vulnerable. En la actualidad, las redes de control industrial son objetivos atractivos para el terrorismo y la guerra cibernética [3]. Al estar conectados a la red, los ICS deben ser protegidos de posibles amenazas que afecten a los procesos industriales. En este contexto, se diferencian dos tipos de amenazas principales:

- **Amenazas internas.** Este tipo de amenazas son realizadas de forma física, ya sea por empleados o por atacantes que posean credenciales de acceso remoto. Son ataques como el Ataque Ucraniano o la Ejecución Remota de Código, descritos en [4].

- **Amenazas externas.** Este tipo de amenazas son realizadas de forma telemática, fuera del entorno industrial. Son ataques de red como la *denegación de servicios* (DoS) [5], que pueden ser detectados por herramientas de detección de intrusos. Un ataque DoS trata de hacer que un recurso informático (como un servicio web, por ejemplo) no esté disponible para los usuarios previstos. Esto suele conseguirse inundando una red o el servidor con grandes cantidades de solicitudes y datos.

Las consecuencias de los ataques van desde un apagado del sistema hasta el robo de datos sensibles y daño a los equipos y empleados, en función del objetivo del atacante y del conocimiento sobre el sistema industrial. Es por ello que se deben tomar medidas de seguridad adicionales para minimizar riesgos y evitar pérdidas, tanto económicas como humanas.

1.2. Objetivos

El objetivo del presente trabajo es el análisis, diseño y evaluación de un grupo de modelos basados en algoritmos de aprendizaje máquina para posteriormente realizar una segunda evaluación con datos relacionados con control industrial. Dichos modelos realizan un análisis en el tráfico recibido basado en criterios y parámetros sobre los que se han entrenado, clasificándolo como tráfico maligno o benigno.

Se estudiarán dos tipos de soluciones, una mediante algoritmos supervisados y otra mediante algoritmos no supervisados, evaluando el rendimiento de ambas soluciones frente a datos que no han visto en su entrenamiento.

1.3. Metodología

La metodología seguida en el trabajo para obtener los resultados ha sido la siguiente: primero, se ha profundizado en el conocimiento de los sistemas de control industrial, de los sistemas de detección de intrusos y del aprendizaje máquina; después, se ha realizado una búsqueda de conjuntos de datos de detección de intrusos y de control industrial; más adelante, se han estudiado, elegido e implementado los algoritmos en los modelos de detección de intrusos para cada conjunto de datos. Por último, se han probado los mejores modelos obtenidos con un conjunto de datos de control industrial nunca visto por los modelos, y se han discutido y analizado los resultados, extrayendo conclusiones sobre el correcto o incorrecto funcionamiento de los modelos, así como posibles mejoras a implementar en trabajos futuros.

1.4. Estructura del documento

El Capítulo 2 contiene una presentación teórica sobre los *sistemas de detección de intrusos en red* (NIDS), así como una explicación breve del aprendizaje máquina y del formato de datos usado en el trabajo. En el Capítulo 3 se exponen las características individuales de cada conjunto de datos, además de un breve análisis exploratorio. A continuación, en el Capítulo 4 se muestran los algoritmos elegidos para construir los modelos y una descripción de su funcionamiento teórico. Más adelante, en el Capítulo 5 se estudian los resultados obtenidos, tanto en la fase de entrenamiento de modelos como en la fase de evaluación con tráfico real. Por último, en el Capítulo 6 se expondrán las conclusiones obtenidas, posibles soluciones IDS a implementar con estos modelos y los trabajos futuros identificados.

En cuanto a los apéndices, en el Apéndice A se presenta una tabla con la planificación temporal y las horas asignadas, acompañada por un diagrama de Gantt. Por otra parte, en el Apéndice B se exponen las gráficas de variables con mayor correlación, agrupadas para cada conjunto de datos. Por último, en el Apéndice C se detallan los tipos de ataque presentes en cada grupo de ataques de los conjuntos de datos.

Capítulo 2

Conceptos previos

En este capítulo se realiza una descripción general de los NIDS. Más adelante, se explica de forma breve el propósito de usar aprendizaje máquina en clasificación de tráfico. Por último, se explica el concepto de los conjuntos de datos y se discute el formato de datos usado en el trabajo.

2.1. Sistemas de detección de intrusos en red

Los sistemas de detección de intrusos en red (del inglés, *Network-based Intrusion Detection Systems*, o NIDS) son herramientas cuyo propósito principal es detectar de forma anticipada anomalías o patrones sospechosos que puedan estar relacionadas con un ataque de red. Se encargan de vigilar tanto el tráfico entrante como el saliente de forma pasiva, esto es, sin interrumpir el flujo de datos. Para que resulten efectivos, los NIDS deben ser actualizados conforme se descubran otros patrones de ataque desconocidos. Algunos ejemplos de este tipo de herramienta son **Snort [6]**, **Zeek [7]** y **Suricata [8]**.

2.2. Aprendizaje máquina

El aprendizaje máquina es una rama de la inteligencia artificial [9], cuyo propósito consiste en desarrollar técnicas para entrenar computadoras. Una vez entrenadas, las computadoras son capaces de crear reglas para automatizar la tarea que se les ha asignado. A pesar de estar ligeramente relacionado con la estadística matemática, el aprendizaje máquina necesita una teoría matemática menor y se encuentra orientado a la ingeniería. Esto se debe a que los datos que se deben procesar (sobre todo en aprendizaje profundo) constan de conjuntos de datos elevadamente grandes e inadecuados para realizar un análisis estadístico clásico.

Existen tres tipos diferentes de aprendizaje máquina: supervisado, no supervisado y reforzado. En este trabajo se usarán algoritmos de los dos primeros tipos mencionados anteriormente, tal y como se detalla en el Capítulo 4. En concreto, estos tipos se definen como:

- **Aprendizaje supervisado.** Consiste en etiquetar datos de salida y entrada para entrenar y probar modelos, que permiten más adelante clasificar la categoría a la que pertenece una entrada. Se le denomina *supervisado* porque este tipo de práctica requiere supervisión humana para etiquetar los datos con precisión.
- **Aprendizaje no supervisado.** El objetivo de este tipo es el entrenamiento de modelos sin la necesidad de etiquetar o procesar los datos previamente. Ya que no es necesaria la intervención humana más que para establecer los parámetros del modelo, se le atribuye el nombre de *no supervisado*.
- **Aprendizaje reforzado.** Este tipo de aprendizaje se caracteriza por hacer aprender a la máquina mediante un esquema de "premios y castigos". En estos modelos no se tiene una etiqueta de salida, sino que el modelo aprende por sí mismo y, a diferencia de los métodos anteriores, prioriza maximizar la recompensa frente a minimizar la función de coste.

Los algoritmos mencionados se enfrentan en este trabajo a problemas de clasificación de tráfico. Existen dos escenarios posibles en un análisis de clasificación: binario y multiclase. Mientras que los sistemas de clasificación binaria dividen el tráfico entre dos clases (por ejemplo, tráfico benigno o tráfico de ataque), los sistemas de clasificación multiclase dividen el tráfico en más de dos clases (por ejemplo, tráfico normal, DoS, backdoor, etc). En este trabajo se ha decidido usar el método de clasificación binaria, aunque algunos de los algoritmos evaluados se pueden orientar a clasificación multiclase.

2.3. Conjuntos de datos

Los conjuntos de datos usados en el trabajo son representaciones de colecciones de datos. En el caso de conjuntos de datos con formato de tabla, cada columna representa una variable o característica, mientras que cada fila representa un registro o muestra del conjunto de datos. Los detalles y características individuales de cada conjunto de datos utilizado en este trabajo se encuentran recogidos en el Capítulo 3.

Existen diferentes formatos de los conjuntos de datos usados en aprendizaje máquina. Los dos tipos más predominantes en este sector son las capturas de paquete (formato PCAP) o los valores separados por comas (formato CSV), siendo este último el más común. Por lo general, los modelos de aprendizaje máquina basados en capturas de paquete preprocesan los datos antes de trabajar con ellos. También existen otras alternativas, como convertir capturas de paquetes a archivos de lenguaje de marcado extensible (formato XML), pero este tipo de prácticas es poco frecuente.

En este trabajo, se ha escogido como formato CSV. La razón principal de esta decisión es el almacenamiento: una captura de paquete tiene un tamaño muy superior a un archivo de valores separados por comas, lo cual hace su procesado más lento y ralentiza la ejecución de los algoritmos de aprendizaje máquina. Además, existe software que permite procesar muy fácilmente la información contenida en un archivo con formato CSV.

Capítulo 3

Selección de conjuntos de datos

En este capítulo se realiza una introducción a los conjuntos de datos que se han usado para construir los modelos. En primer lugar, se realiza una breve descripción del conjunto de datos escogido, así como de las características que poseen. Se indican también todas las operaciones realizadas en el conjunto de datos en la fase de preprocesado. En segundo lugar, se muestra un *análisis exploratorio de datos* (EDA) de dichas características para juzgar si son necesarias o relevantes a la hora de procesarlas con los modelos.

3.1. Conjunto de datos NSL_KDD

3.1.1. Descripción

NSL_KDD [10] es la evolución del conjunto de datos KDD'99 y fue concebido para solventar alguno de sus problemas, como la gran cantidad de registros duplicados. A pesar de ello, sigue sufriendo de algunos inconvenientes y puede no ajustarse a una representación real de una red. En la actualidad, se usa de referencia para comparar métodos de detección de intrusos, tal y como se indica en [10].

Este conjunto de datos se caracteriza por no tener muestras redundantes en el conjunto de entrenamiento y no tener muestras duplicadas en el conjunto de prueba, garantizando que los modelos no den mayor peso a parámetros duplicados. Cuenta con un total de 43 características, mostradas en la Tabla 3.1.

Conjunto de datos NSL_KDD					
#	**Nombre**	**Tipo**	**#**	**Nombre**	**Tipo**
1	duration	Integer	23	count	Integer
2	protocol_type	Nominal	24	srv_count	Integer
3	service	Nominal	25	serror_rate	Float
4	flag	Nominal	26	srv_serror_rate	Float
5	src_bytes	Integer	27	rerror_rate	Float
6	dst_bytes	Integer	28	srv_rerror_rate	Float
7	land	Binary	29	same_srv_rate	Float
8	wrong_fragment	Integer	30	diff_srv_rate	Float
9	urgent	Integer	31	srv_diff_host_rate	Float
10	hot	Integer	32	dst_host_count	Integer
11	num_failed_logins	Integer	33	dst_host_srv_count	Integer
12	logged_in	Binary	34	dst_host_same_srv_rate	Float
13	num_compromised	Integer	35	dst_host_diff_srv_rate	Float
14	root_shell	Binary	36	dst_host_same_src_port_rate	Float
15	su_attempted	Binary	37	dst_host_srv_diff_host_rate	Float
16	num_root	Integer	38	dst_host_serror_rate	Float
17	num_file_creations	Integer	39	dst_host_srv_serror_rate	Float
18	num_shells	Integer	40	dst_host_rerror_rate	Float
19	num_access_files	Integer	41	dst_host_srv_rerror_rate	Float
20	num_outbound_cmds	Integer	42	class	Nominal
21	is_host_login	Binary	43	difficulty	Integer
22	is_guest_login	Binary			

Tabla 3.1: Datos y sus tipos contenidos en el conjunto de datos NSL_KDD

Como se observa en la tabla, el conjunto de datos NSL_KDD consta de un total de 4 características categóricas y 39 numéricas, de las cuales 6 son binarias. La descripción detallada cada característica se encuentra en [11]. Además, la columna *difficulty* indica la dificultad de detección del ataque, por lo que no es una característica sobre la que se quiera entrenar el modelo.

Para realizar la clasificación binaria correctamente, se elimina la columna *difficulty* y se sustituye la columna categórica *class* (que indica el tipo de ataque) por una columna binaria *label*, en la que el valor '0' indica tráfico benigno y el valor '1' indica que pertenece a muestras asociadas con ataques.

3.1.2. EDA

Al realizar un análisis exploratorio general en el conjunto de datos NSL_KDD se puede observar la ausencia de valores nulos en las características. También destaca que los valores obtenidos en cada columna son acordes al tipo de datos y no presentan anomalías. Por último, se puede comprobar que el conjunto de datos de entrenamiento está equilibrado, es decir, que presenta un número similar de tráfico normal frente a tráfico de ataques. Esto es un factor deseable en un conjunto de datos a la hora de usarse para entrenar los modelos.

De cara a optimizar la obtención de resultados (especialmente en modelos supervisados), interesa eliminar características con un coeficiente de correlación alto, ya que no aportan información nueva al clasificador. Para ello, se ha calculado la matriz de correlación para las columnas del conjunto de este conjunto de datos, mostrada en la Figura 3.1.

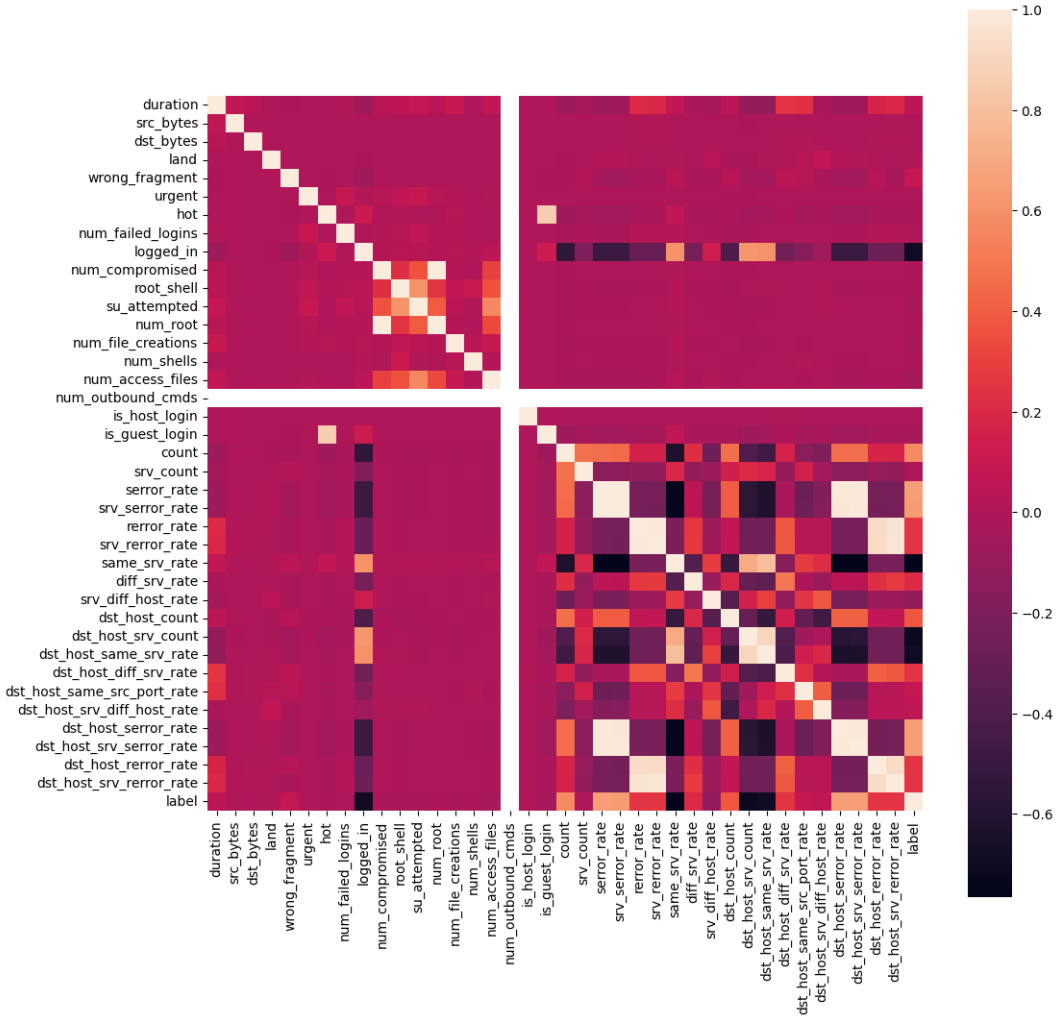

Figura 3.1: Matriz de correlación del conjunto de datos NSL_KDD

Como se observa en la figura, existen tres grupos que se encuentran altamente correlados:

- En primer lugar, las columnas *num_compromised* y *num_root* son las columnas más correladas de todo el conjunto de datos, llegando a un coeficiente de correlación de 0.9988.
- Por otra parte, las columnas *serror_rate*, *srv_serror_rate*, *dst_host_serror_rate* y *dst_host_srv_serror_rate* se encuentran altamente correladas entre ellas, con coeficientes de correlación entre 0.9776 y 0.9932.
- Por último, las columnas *rerror_rate*, *srv_rerror_rate*, *dst_host_rerror_rate* y *dst_host_srv_rerror_rate* se encuentran altamente correladas de forma similar al grupo anterior, pero con coeficientes de correlación entre 0.9178 y 0.97.

Una vez analizada la correlación entre las características mencionadas, se decide conservar las variables *num_compromised*, *serror_rate* y *rerror_rate*. El resto de variables mencionadas son desechadas y no serán consideradas en el procesado de datos. En el Apéndice B.1 se muestran las gráficas que relacionan las columnas con gran coeficiente de correlación.

Conjunto de datos NSL_KDD		
Tipo	**Train**	**Test**
Normal	67343	9711
DoS	45927	7167
Probe	11656	2421
R2L	995	3178
U2R	52	67

Tabla 3.2: Tipos de ataque en el conjunto de datos NSL_KDD

En la Tabla 3.2 se muestran agrupados los tipos de ataque presentes en el conjunto de datos, tanto para el conjunto de entrenamiento como para el de prueba. Existen ataques de denegación de servicio, de *probing*, de tipo R2L y de tipo U2R. Un ataque de *probing* trata de escanear una red o una máquina en busca de información y de posibles vulnerabilidades. Por otro lado, un ataque R2L intenta explotar vulnerabilidades de un sistema de forma remota para tratar de obtener privilegios de un usuario local. Por último, en los ataques U2R el atacante ya posee acceso a un recurso en un sistema y trata de explotar alguna vulnerabilidad a nivel de aplicación o de sistema operativo para obtener privilegios más elevados.

Se puede observar que la mayoría predominante de ataques en el conjunto de entrenamiento pertenecen a la clase de DoS y *probing*. Sin embargo, en el conjunto de prueba se incrementa el número de ataques de tipo remoto a local (R2L) y usuario a raíz (U2R). En el Apéndice C.1 se detallan los tipos de ataque presentes en cada uno de los cuatro grupos mencionados en la tabla.

3.2. Conjunto de datos UNSW_NB15

3.2.1. Descripción

El conjunto de datos UNSW_NB15 es un conjunto de datos de detección de intrusos en red desarrollado en 2015 por Nour Moustafa y Jill Slay [12]. Es uno de los conjuntos de datos más usados en detección de intrusos y contiene 9 tipos de ataques diferentes.

A pesar de que el conjunto de datos ya cuenta con un conjunto de entrenamiento y un conjunto de prueba separados, también está disponible repartido en 4 ficheros distintos, de modo que el usuario tenga la oportunidad de crear un conjunto de entrenamiento y de prueba propio. Cuenta con un total de 49 características distintas, ilustradas en la Tabla 3.3.

Conjunto de datos UNSW_NB15					
#	**Nombre**	**Tipo**	**#**	**Nombre**	**Tipo**
1	srcip	Nominal	26	res_bdy_len	Integer
2	sport	Integer	27	sjit	Float
3	dstip	Nominal	28	djit	Float
4	dsport	Integer	29	stime	Time
5	proto	Nominal	30	ltime	Time
6	state	Nominal	31	sintpkt	Float
7	dur	Float	32	dintpkt	Float
8	sbytes	Integer	33	tcprtt	Float
9	dbytes	Integer	34	synack	Float
10	sttl	Integer	35	ackdat	Float
11	dttl	Integer	36	is_sm_ips_ports	Binary
12	sloss	Integer	37	ct_state_tt	Integer
13	dloss	Integer	38	ct_flw_http_mthd	Integer
14	service	Nominal	39	is_ftp_login	Binary
15	sload	Float	40	ct_ftp_cmd	Integer
16	dload	Float	41	ct_srv_src	Integer
17	spkts	Integer	42	ct_srv_dst	Integer
18	dpkts	Integer	43	ct_dst_ltm	Integer
19	swin	Integer	44	ct_src_ltm	Integer
20	dwin	Integer	45	ct_src_dport_ltm	Integer
21	stcpb	Integer	46	ct_dst_sport_ltm	Integer
22	dtcpb	Integer	47	ct_dst_src_ltm	Integer
23	smeanz	Integer	48	attack_cat	Nominal
24	dmeanz	Integer	49	label	Binary
25	trans_depth	Integer			

Tabla 3.3: Datos y sus tipos contenidos en el conjunto de datos UNSW_NB15

Como se observa en la tabla, el conjunto de datos UNSW_NB15 contiene un total de 6 características categóricas y 43 numéricas, de las cuales 3 son binarias. A diferencia del conjunto de datos NSL_KDD, este conjunto sí presenta valores nulos en algunas columnas. La descripción de cada característica se encuentra en [13]. Cabe destacar también que los valores y tipos de datos de ciertas columnas no son coherentes con la respectiva descripción presentada en el trabajo de investigación citado anteriormente, por lo que deben ser modificados para un correcto procesado de los datos.

Antes de dicho procesado, se deben eliminar las 4 primeras columnas (*srcip, sport, dstip y dsport*) ya que corresponden a variables relacionadas con puertos y direcciones IP y no resultan útiles en la clasificación de tráfico. Por otro lado, dado que se afronta un problema de clasificación binaria, es necesario eliminar la columna *attack_cat*, correspondiente a los tipos de ataque presentes en el conjunto de datos.

Al calcular el porcentaje de ataques respecto al tráfico total, se descubre que los ataques solo corresponden a un 12,65 %. Esto indica que el conjunto de datos se encuentra altamente desequilibrado, algo que no ocurría en el caso del conjunto de datos anterior. Este factor es sumamente importante en la interpretación de las métricas descritas en la Sección 5.2. Además, a diferencia del conjunto de datos NSL_KDD, en este no se utiliza el conjunto de entrenamiento y el conjunto de prueba ya definido, sino que el conjunto de datos en su totalidad es dividido en una proporción de 70 % para entrenamiento y 30 % para prueba. La razón de esta decisión se refleja más adelante en el análisis de resultados, en la Sección 5.3.

3.2.2. EDA

Como se ha mencionado anteriormente, el conjunto de datos UNSW_NB15 presenta valores nulos en algunas columnas. En concreto, se deben modificar las columnas *attack_cat, is_ftp_login y ct_flw_http_mthd*, sustituyendo los valores nulos por '0' o por 'normal' (en el caso de la columna *attack_cat*). Además, la columna *ct_ftp_cmd* se interpreta como variable categórica al cargar los datos, mostrando espacios en blanco como posible valor. Se debe modificar la columna para sustituir dichos espacios por '0' y convertirla en numérica. Por último, la columna *service* no posee valores nulos, pero la falta de servicio se indica mediante el signo '-'. En orden de ser coherente con los datos, se modifica dicho valor por 'none'.

Tal y como se ha descrito para el conjunto de datos anterior, se calcula la matriz de correlación para obtener los coeficientes de correlación que comparten las características del conjunto de datos entre ellas. Dicha matriz de correlación se muestra en la Figura 3.2.

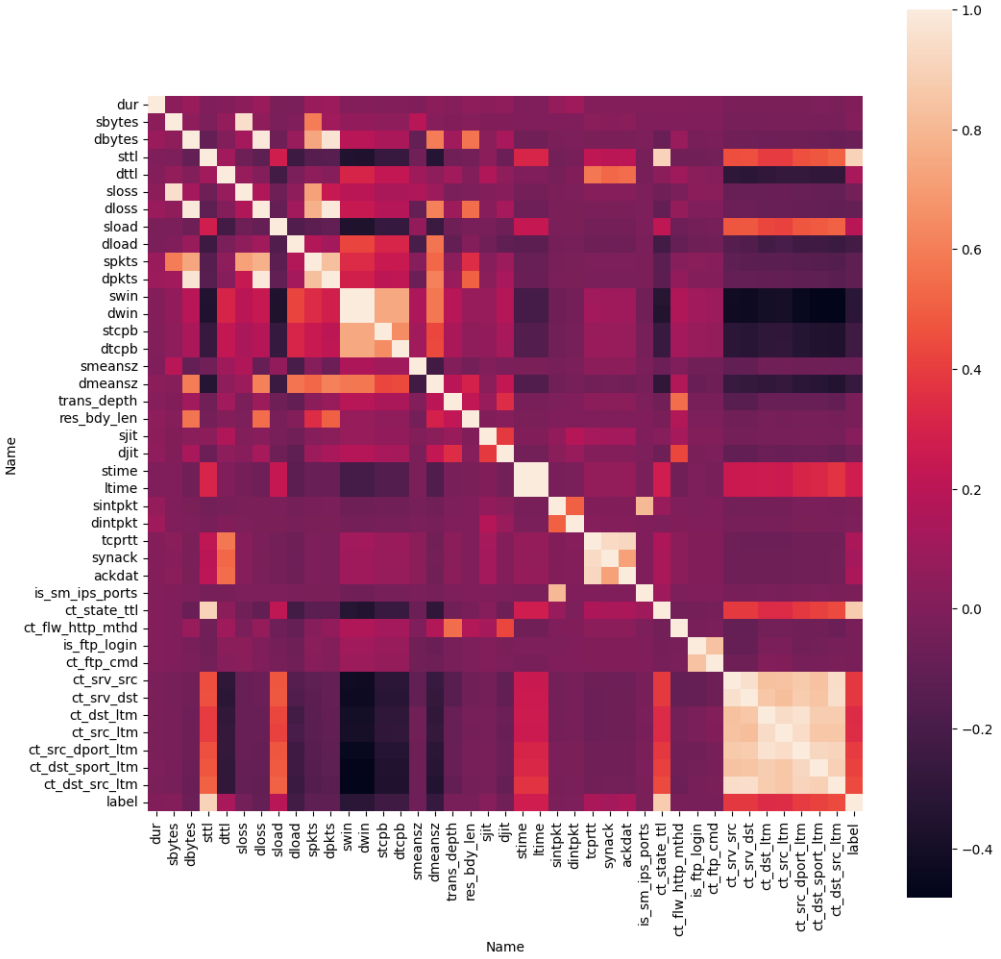

Figura 3.2: Matriz de correlación del conjunto de datos UNSW_NB15

Como se observa en la figura, existen siete grupos que se encuentran altamente correlados:

- Las columnas *sbytes* y *sloss* se encuentran correladas de forma notable, obteniendo un coeficiente de correlación de 0.9515.

- Las columnas *dpkts*, *dbytes* y *dloss* también se encuentran altamente correladas, con coeficientes de correlación entre 0.9705 y 0.9912.

- las columnas *sttl* y *ct_state_ttl* se encuentran correladas con un coeficiente de correlación de 0.9058. Además, se destaca que la columna *sttl* se encuentra correlada en gran medida con la etiqueta *label*, con un coeficiente de correlación de 0.9043. Esto indica que la columna *sttl* puede jugar un papel importante en la clasificación del tráfico en este conjunto de datos.

- Las columnas *swin* y *dwin*, que hacen referencia a las ventanas de recepción del protocolo TCP en fuente y destino respectivamente, se encuentran enormemente correladas con un coeficiente de correlación de 0.9971.
- Las columnas *stime* y *ltime* son las columnas más correladas de todo el conjunto de datos, alcanzando un coeficiente de correlación de 0.9999.
- Las columnas *tcprtt*, *synack* y *ackdat* se encuentran bastante correladas, con coeficientes de correlación entre 0.9202 y 0.9332.
- Las columnas *ct_srv_src*, *ct_srv_dst* y *ct_dst_src_ltm* se encuentran muy relacionadas, obteniendo coeficientes de correlación entre 0.9421 y 0.9567.
- Las columnas *ct_dst_ltm*, *ct_src_ltm*, *ct_src_dport_ltm* y *ct_dst_sport_ltm* también se encuentran asociadas entre ellas, con coeficientes de correlación entre 0.9109 y 0.9601.

Tras analizar las correlaciones más elevadas entre las características, se deciden conservar las variables *sbytes*, *dbytes*, *sttl*, *ct_state_ttl*, *swin*, *stime*, *ct_srv_src*, *ct_dst_ltm*, *ct_src_ltm* y *ct_dst_sport_ltm*. El resto de variables mencionadas anteriormente (a excepción de *label*) son eliminadas de nuestro conjunto de datos. En el Apéndice B.2 se muestran las gráficas que relacionan las columnas con gran coeficiente de correlación.

Conjunto de datos UNSW_NB15	
Tipo	**Nº Ataques**
Normal	2218764
DoS	16353
Rastreo	232145
Vulnerabilidades	72785

Tabla 3.4: Tipos de ataque en el conjunto de datos UNSW_NB15

En la Tabla 3.4 se muestran agrupados los tipos de ataque presentes en el conjunto de datos. Existen ataques de denegación de servicio, de rastreo y de vulnerabilidades. Los ataques de rastreo están relacionados con la recopilación de información de los equipos de una red mediante el uso de ataques de escaneo de puertos, spam y penetración de sistemas, entre otros. Por otro lado, los ataques relacionados con vulnerabilidades aprovechan fallas de seguridad en el sistema para infiltrarse o infectar dicho sistema con software malicioso.

Se puede observar que los ataques más comunes en el conjunto de datos UNSW_NB15 son los ataques de rastreo, seguido de los ataques que aprovechan vulnerabilidades. En este caso, no se observan tantos registros de DoS como en el anterior conjunto de datos. Los 9 diferentes tipos de ataque presentes en el conjunto de datos se encuentran en el Apéndice C.2, divididos en las 3 secciones mostradas en la tabla.

3.3. Conjunto de datos CIC_IDS2017

3.3.1. Descripción

El conjunto de datos CIC_IDS2017 es un conjunto de datos de detección de intrusos desarrollado en 2017 por el Instituto Canadiense de Ciberseguridad [14]. Este conjunto de datos cuenta con un total de 14 ataques distintos, correspondientes a los ataques más comunes en su momento de desarrollo, con la finalidad de reflejar tráfico de una red real.

Este conjunto de datos es el más nuevo de los conjuntos de datos usados, y se caracteriza por contener flujos de datos bidireccionales, lo que significa que cada uno contiene información sobre ambos lados de la comunicación, tanto fuente como destino [15]. Cuenta con un total de 79 columnas, ilustradas en la Tabla 3.5.

Además, el conjunto de datos CIC_IDS2017 está formado por 8 ficheros distintos, repartidos de la siguiente forma:

- **Lunes.** Un fichero, correspondiente al tráfico de la mañana. Está formado por tráfico benigno.
- **Martes.** Un fichero, correspondiente a la mañana. Contiene tráfico normal y tráfico de ataques de fuerza bruta. Los ataques de fuerza bruta se caracterizan por usar el método de prueba y error múltiples veces para descifrar credenciales o claves de cifrado.
- **Miércoles.** Un fichero, correspondiente a la mañana. Contiene tráfico normal y ataques DoS.
- **Jueves.** Dos ficheros, correspondientes al tráfico capturado por la mañana y por la tarde. Por la mañana se tiene tráfico normal con ataques web y de fuerza bruta. Los ataques web están destinados a una aplicación cliente y se origina desde un lugar en la Web. Por la tarde se tiene tráfico normal mientras se producen ataques de explotación de vulnerabilidades.
- **Viernes.** Tres ficheros, uno correspondiente a la mañana y dos correspondientes a la tarde. El fichero de la mañana consta de tráfico normal y un ataque de DoS. Un fichero de la tarde contiene ataques de rastreo, mientras que el otro fichero contiene otro ataque DoS.

Conjunto de datos CIC_IDS2017					
#	**Nombre**	**Tipo**	**#**	**Nombre**	**Tipo**
1	Destination Port	Integer	41	Packet Length Mean	Float
2	Flow Duration	Integer	42	Packet Length Std	Float
3	Total Fwd Packets	Integer	43	Packet Length Variance	Float
4	Total Backward Packets	Integer	44	FIN Flag Count	Binary
5	Total Length of Fwd Packets	Integer	45	SYN Flag Count	Binary
6	Total Length of Bwd Packets	Integer	46	RST Flag Count	Binary
7	Fwd Packet Length Max	Integer	47	PSH Flag Count	Binary
8	Fwd Packet Length Min	Integer	48	ACK Flag Count	Binary
9	Fwd Packet Length Mean	Float	49	URG Flag Count	Binary
10	Fwd Packet Length Std	Float	50	CWE Flag Count	Binary
11	Bwd Packet Length Max	Integer	51	ECE Flag Count	Binary
12	Bwd Packet Length Min	Integer	52	Down/Up Ratio	Integer
13	Bwd Packet Length Mean	Float	53	Average Packet Size	Float
14	Bwd Packet Length Std	Float	54	Avg Fwd Segment Size	Float
15	Flow Bytes/s	Float	55	Avg Bwd Segment Size	Float
16	Flow Packets/s	Float	56	Fwd Header Length.1	Integer
17	Flow IAT Mean	Float	57	Fwd Avg Bytes/Bulk	Integer
18	Flow IAT Std	Float	58	Fwd Avg Packets/Bulk	Integer
19	Flow IAT Max	Integer	59	Fwd Avg Bulk Rate	Integer
20	Flow IAT Min	Integer	60	Bwd Avg Bytes/Bulk	Integer
21	Fwd IAT Total	Integer	61	Bwd Avg Packets/Bulk	Integer
22	Fwd IAT Mean	Float	62	Bwd Avg Bulk Rate	Integer
23	Fwd IAT Std	Float	63	Subflow Fwd Packets	Integer
24	Fwd IAT Max	Integer	64	Subflow Fwd Bytes	Integer
25	Fwd IAT Min	Integer	65	Subflow Bwd Packets	Integer
26	Bwd IAT Total	Integer	66	Subflow Bwd Bytes	Integer
27	Bwd IAT Mean	Float	67	Init_Win_bytes_forward	Integer
28	Bwd IAT Std	Float	68	Init_Win_bytes_backward	Integer
29	Bwd IAT Max	Integer	69	act_data_pkt_fwd	Integer
30	Bwd IAT Min	Integer	70	min_seg_size_forward	Integer
31	Fwd PSH Flags	Binary	71	Active Mean	Float
32	Bwd PSH Flags	Binary	72	Active Std	Float
33	Fwd URG Flags	Binary	73	Active Max	Integer
34	Bwd URG Flags	Binary	74	Active Min	Integer
35	Fwd Header Length	Integer	75	Idle Mean	Float
36	Bwd Header Length	Integer	76	Idle Std	Float
37	Fwd Packets/s	Float	77	Idle Max	Integer
38	Bwd Packets/s	Float	78	Idle Min	Integer
39	Min Packet Length	Integer	79	Label	Nominal
40	Max Packet Length	Integer			

Tabla 3.5: Datos y sus tipos contenidos en el conjunto de datos CIC_IDS2017

Como se observa en la tabla, el conjunto de datos consta de una sola característica categórica (*label*) y 78 numéricas, de las cuales 12 son binarias. Una ventaja de este conjunto de datos reside en que dispone de un gran número de registros y no se tiene ninguno nulo. No obstante, existen valores excesivamente elevados que afectan al procesado de datos, por lo que dichos valores deben ser descartados.

En este caso, solo es necesaria la eliminación de la columna *Destination Port* correspondiente al puerto destino ya que, tal y como se ha descrito para el conjunto de datos UNSW_NB15, las variables relacionadas con puertos y direcciones IP no resultan útiles en la tarea de clasificación. Al no usar clasificación multiclase en este trabajo, la columna *label* no es relevante y es eliminada también del conjunto de datos. En su lugar, se establece una columna binaria con el mismo nombre, indicando el valor '0' en caso de tráfico normal y el valor '1' en caso de tráfico relacionado con ataques.

Al calcular el porcentaje de ataques respecto al tráfico total, se calcula que los ataques corresponden a un 19,68 % del tráfico total. Tal y como sucedía en el conjunto de datos UNSW_NB15, este conjunto de datos se encuentra altamente desequilibrado. De igual forma, al no tener un conjunto de entrenamiento y prueba definidos previamente, se decide dividir el conjunto de datos mediante la proporción 70-30 indicada en la Sección 3.2.1.

3.3.2. EDA

Tal y como se ha procedido con los anteriores conjuntos de datos, se calcula la matriz de correlación para obtener los coeficientes de correlación, mostrada en la Figura 3.3. Cabe destacar que este conjunto de datos es con diferencia el que más variables correladas tiene. Gracias al EDA realizado, se descubre que se pueden eliminar un total de 23 columnas que se encuentran altamente correladas.

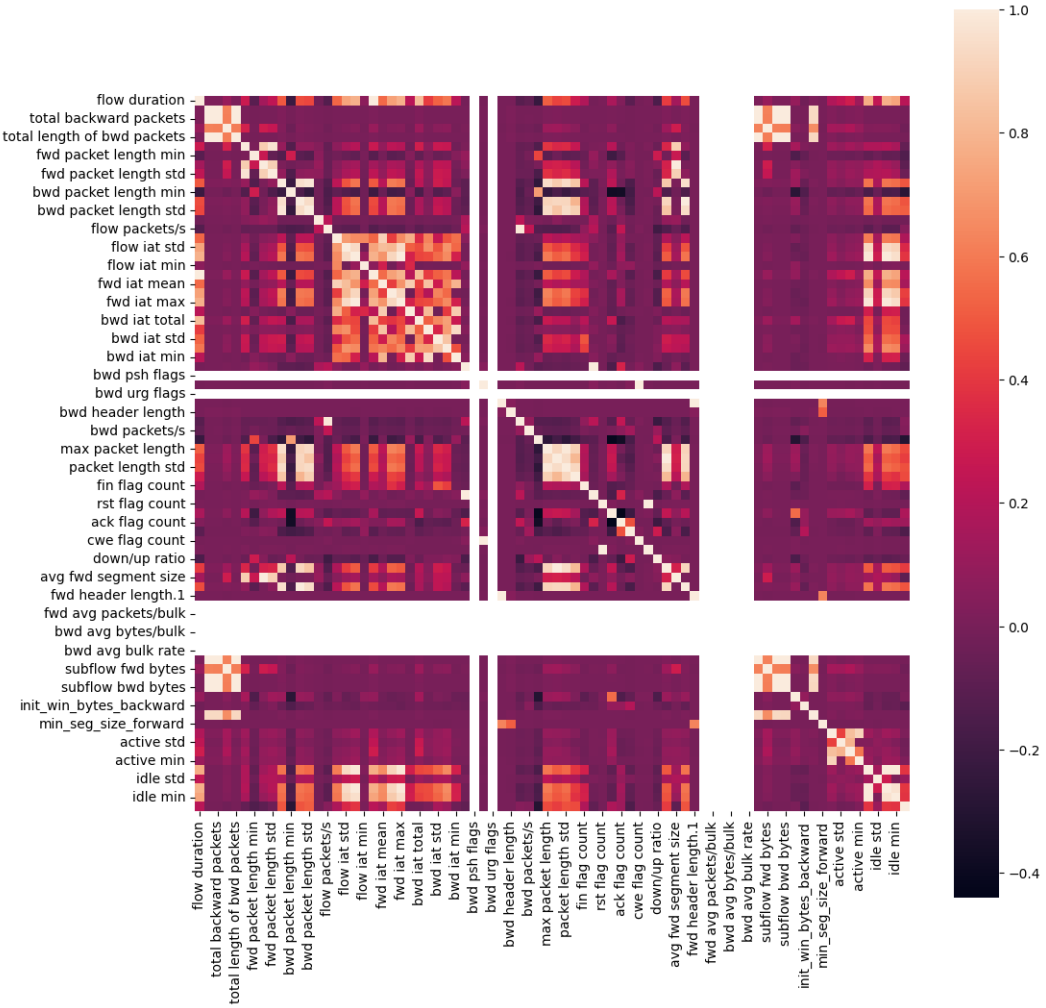

Figura 3.3: Matriz de correlación del conjunto de datos CIC_IDS2017

En el conjunto de datos CIC_IDS2017 existen un total de 16 grupos que se encuentran altamente correlados:

- Las columnas *flow duration* y *fwd iat total* tienen un coeficiente de correlación de 0.9985.
- Las columnas *total fwd packets*, *total backward packets*, *total length of bwd packets*, *act_data_pkt_fwd*, *subflow fwd packets*, *subflow bwd packets*, *subflow fwd bytes* y *subflow bwd bytes* se encuentran altamente correladas, con valores del coeficiente de correlación entre 0.9282 y 1.
- Las columnas *fwd packet length max* y *fwd packet length std* alcanzan un coeficiente de correlación de 0.9683.

- Las columnas *fwd packet length mean* y *avg fwd segment size* obtienen una correlación positiva perfecta, es decir, un coeficiente de correlación de 1.
- Las columnas *bwd packet length max*, *bwd packet length mean*, *bwd packet length std*, *max packet length*, *packet length std* y *avg bwd segment size* obtienen valores del coeficiente de correlación entre 0.9405 y 0.9828.
- Las columnas *flow packets/s* y *fwd packets/s* consiguen un coeficiente de correlación de 0.9875.
- Las columnas *flow iat mean* y *fwd iat mean* registran un coeficiente de correlación de 0.9001.
- Las columnas *flow iat std*, *flow iat max*, *fwd iat max*, *idle mean*, *idle max* e *idle min* obtienen coeficientes de correlación entre 0.9090 y 0.9981.
- Las columnas *bwd iat mean* y *bwd iat min* alcanzan un coeficiente de correlación de 0.9327.
- Las columnas *fwd psh flags* y *syn flag count* también obtienen una correlación positiva perfecta, con un coeficiente de correlación de 1.
- Las columnas *fwd urg flags* y *cwe flag count*, de igual forma, consiguen una correlación positiva perfecta.
- Las columna *fwd header length.1* es una réplica de la columna *fwd header length*, obteniendo un coeficiente de correlación de 1. Por lo tanto, la réplica debe eliminarse.
- Las columnas *max packet length*, *packet length mean*, *packet length std*, *average packet size* y *avg bwd segment size* obtiene valores del coeficiente de correlación entre 0.9058 y 0.9977.
- Las columnas *rst flag count* y *ece flag count* comparten un coeficiente de correlación de 0.9969.
- Las columnas *active mean* y *active min* consiguen un coeficiente de correlación de 0.90681.

Como se ha podido comprobar, existe un gran número de variables correladas, lo que reduce drásticamente el tamaño del conjunto de datos a utilizar. Tras analizar las correlaciones más elevadas entre las características, se deciden conservar las siguientes variables: *flow duration*, *total fwd packets*, *act_data_pkt_fwd*, *fwd packet length max*, *fwd packet length mean*, *bwd packet length max*, *max packet length*, *flow packets/s*, *flow iat mean*, *fwd iat mean*, *flow iat std*, *flow iat max*, *bwd iat mean*, *bwd iat min*, *fwd psh flags*, *fwd urg flags*, *fwd header length*, *max packet length*, *rst flag count*, *active mean* y *active min*.

El resto de variables mencionadas anteriormente se eliminan antes de realizar el preprocesado de datos, reduciendo el número de columnas a un total de 55. Por lo tanto, se han eliminado 24 características redundantes o sin valor para la clasificación, como era el caso de la columna *Destination Port*. En el Apéndice B.3 se muestran las gráficas de las distintas variables correladas agrupadas con las características que comparten similitud.

Conjunto de datos CIC_IDS2017	
Tipo	**Nº Ataques**
Normal	2271320
DoS	381693
Rastreo	158804
Fuerza Bruta	13832
Ataques Web	2180
Vulnerabilidades	47

Tabla 3.6: Tipos de ataque en el conjunto de datos CIC_IDS2017

En la Tabla 3.6 se muestran agrupados los tipos de ataque presentes en el conjunto de datos, siendo estos ataques de denegación de servicios, rastreo, fuerza bruta, ataques web y explotación de vulnerabilidades. Se puede observar que los ataques más comunes son los ataques DoS, de forma similar al conjunto de datos NSL_KDD. No obstante, también se registra un gran número de ataques de rastreo, mientras que el resto de ataques representan un porcentaje del tráfico muy bajo. Los 14 diferentes tipos de ataque se muestran en el Apéndice C.3, agrupados de la misma forma que en la tabla.

Capítulo 4

Selección de algoritmos

Este capítulo describe los algoritmos usados en la clasificación de detección de intrusos. Primero se presentan los algoritmos supervisados, y después se describen los algoritmos no supervisados.

Además de los algoritmos usados, se estudiaron otras posibilidades a la hora de obtener un mejor modelo de clasificación. Un ejemplo de ello es el algoritmo supervisado *K-Nearest Neighbors* [16]. Sin embargo, dicha solución fue desechada, ya que el algoritmo no trabaja bien con conjuntos de datos grandes. Esto se debe a que el algoritmo necesita calcular la distancia entre un punto y el resto de puntos del conjunto de datos. Esta tarea, sin embargo, resultaba muy costosa y suponía una degradación notable del rendimiento del modelo.

4.1. Algoritmos supervisados

4.1.1. Regresión Logística

El algoritmo de Regresión Logística [17] es un método de clasificación binaria utilizado en problemas de aprendizaje máquina. Es uno de los métodos más simples y más utilizados en clasificación binaria, estimando la relación entre una variable binaria dependiente (ataques) y las variables independientes (las características).

Concretamente, es un modelo estadístico utilizado para determinar la probabilidad de que ocurra un evento. En este tipo de análisis se asume que las características son esencialmente independientes entre sí, presentando poca o nula multicolinealidad. La razón de elección de la regresión logística en lugar de una regresión lineal es debida a que una regresión lineal no sirve para predecir clases binarias [18].

4.1.2. Support Vector Machines

El método de Máquinas de Vector Soporte, conocido comúnmente como *Support Vector Machines* (SVM) [19], es un método de clasificación y regresión, desarrollado originalmente para clasificación binaria. Se considera un referente dentro del aprendizaje estadístico y aprendizaje máquina al ser uno de los mejores clasificadores para un abanico de situaciones [20].

Las SVM se basan en el clasificador de máximo margen, cuya idea principal es clasificar los datos mediante una frontera lineal que separa los datos de una categoría de los de otra. Dicho de otra forma, el clasificador de máximo margen representa un hiperplano óptimo en el caso de que dos clases sean linealmente separables. Para una descripción detallada del algoritmo, se recomienda consultar [21].

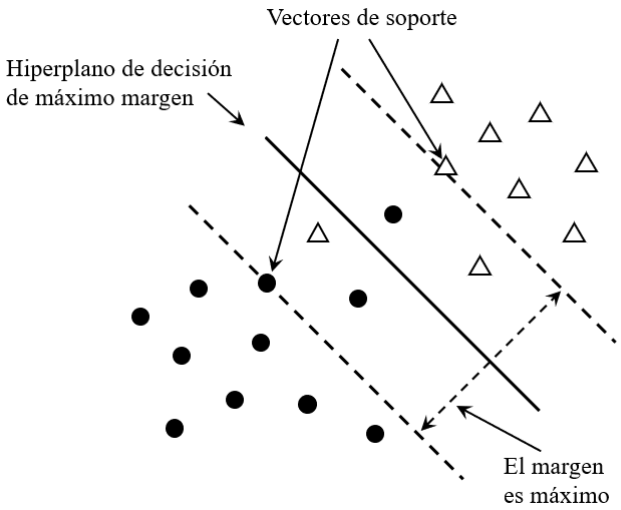

Figura 4.1: Esquema del algoritmo *Support Vector Machines*

A modo de ejemplo, en la Figura 4.1 se muestra la clasificación de muestras pertenecientes a dos clases distintas. La línea continua representa el hiperplano y las líneas discontinuas el margen a cada lado. Se puede observar que, a pesar de estar dentro de su respectivo margen, las muestras pueden ser clasificadas correctamente si se encuentran en el lado asociado a su clase del hiperplano. Por lo tanto, las muestras que se encuentran en el lado opuesto de su clase son clasificadas de forma incorrecta.

4.1.3. Decision Tree

El algoritmo del aprendizaje del árbol de decisión, mejor conocido como *Decision Tree* (DT) [22], es un algoritmo de aprendizaje supervisado basado en tomas de decisiones representadas por árboles. Es muy usado en estadística, en minado de datos y en aprendizaje de máquinas, donde resulta muy popular dada su simpleza y facilidad de comprensión [23].

Existen dos tipos principales de árboles de decisión: los árboles de clasificación y los árboles de regresión. Para los problemas de clasificación, como es el caso del presente trabajo, las hojas de los árboles representan las etiquetas de cada clase. Por otro lado, las ramas representan decisiones que se toman en función de las características de los datos, y que a su vez llevan a las etiquetas de clase mencionadas anteriormente.

4.1.4. Random Forest

El algoritmo del bosque aleatorio, conocido comúnmente como *Random Forest* (RF) [24], es un método supervisado de aprendizaje conjunto basado en la construcción de varios árboles de decisión en el entrenamiento. A su vez, cada árbol de decisión es un DT, comentado anteriormente (véase la Sección 4.1.3). Dichos árboles son usados como modelos predictivos, extrayendo conclusiones de un conjunto de datos propuesto.

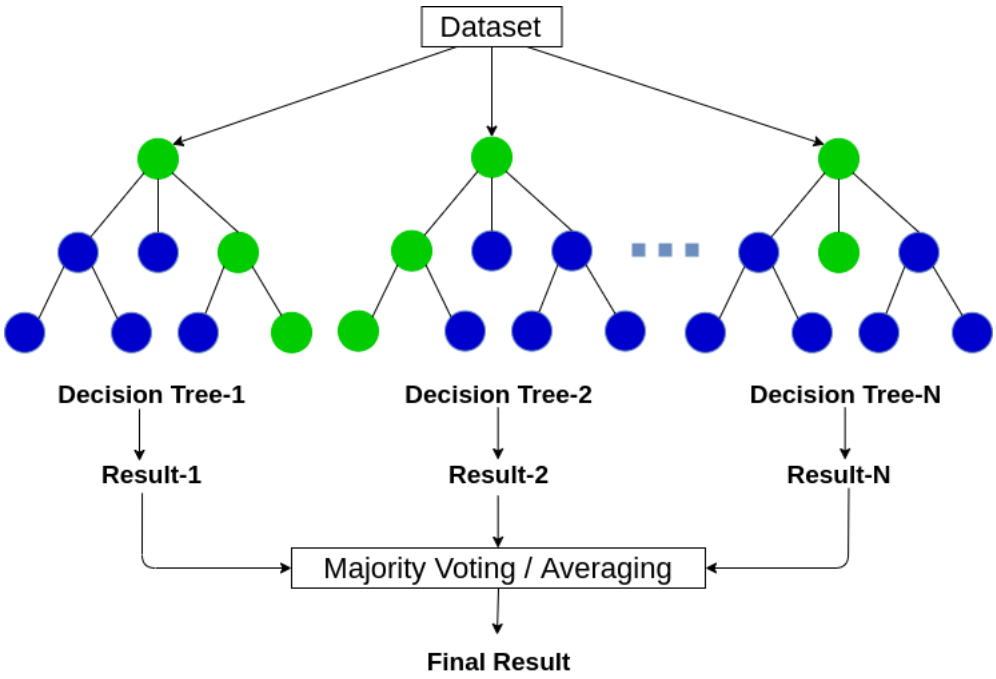

Figura 4.2: Esquema del algoritmo *Random Forest* (obtenido de [25])

En la Figura 4.2 se muestra un esquema simple del funcionamiento del bosque aleatorio: Los datos usados para entrenar el modelo son procesados por un número finito N de árboles de decisión y cada uno devuelve un resultado. Posteriormente, los resultados se ponen en común entre los N árboles de decisión y la salida se corresponde con la clase seleccionada por voto mayoritario. Por lo general, el algoritmo del bosque aleatorio presenta mejores resultados que los árboles de decisión ya que al estar formado por N árboles de decisión posee un mayor criterio para clasificar correctamente los datos.

4.1.5. Gradient Boosted Decision Tree

La técnica del impulso de gradiente, mejor conocida como *Gradient Boosting* [26], es una técnica de aprendizaje máquina usada en tareas relacionadas con regresión y clasificación. Proporciona un modelo de predicción fuerte mediante un conjunto de modelos de predicciones débiles. Si los modelos débiles usados son árboles de decisión, el algoritmo usado se denomina árbol de decisión con impulso de gradiente (*Gradient Boosted Decision Tree*).

En problemas relacionados con la clasificación, el algoritmo busca minimizar la divergencia *Kullback-Leibler*, que indica la divergencia direccionada entre dos distribuciones. Para cada muestra de datos, se busca que la probabilidad de distribución predicha sea lo más cercana a la verdadera probabilidad de distribución. Esto se consigue ajustando los modelos débiles que conforman el modelo fuerte de predicción [27].

4.2. Algoritmos no supervisados

Por norma general, los algoritmos no supervisados no son usados para problemas de clasificación, ya que no se puede obtener información precisa con respecto a la clasificación de datos. Es más, las etiquetas que asocian los algoritmos no supervisados a los datos no siempre se corresponden con las etiquetas de clase empleadas en la clasificación, lo que ofrece una menor precisión de los resultados con respecto a los algoritmos supervisados [28].

Sin embargo, en este trabajo se han estudiado modelos basados en algoritmos no supervisados. El objetivo de ello es verificar lo expuesto en el párrafo anterior: si los algoritmos supervisados obtienen mejores resultados y son más aptos para problemas de clasificación en el contexto de entornos de control industrial. Para este trabajo, se han escogido dos algoritmos no supervisados.

4.2.1. Isolation Forest

El algoritmo del bosque de aislamiento, conocido como *Isolation Forest* (IF) [29], es un método no supervisado utilizado en detección de anomalías cuando los datos no se encuentran etiquetados en clases. Su funcionamiento está inspirado en el algoritmo *Random Forest*, expuesto en la Sección 4.1.4.

Un modelo del bosque de aislamiento está formado por la combinación de múltiples árboles de aislamiento. Las observaciones de entrenamiento se van separando de forma recursiva, creando las ramas del árbol hasta que cada observación queda aislada en un nodo terminal. Sin embargo, la decisión de los puntos de división se hace de forma aleatoria, a diferencia de los árboles de decisión. Para una descripción detallada del algoritmo, se recomienda consultar [30].

4.2.2. K-Means Clustering

El algoritmo de las k-medias de agrupamiento, mejor conocido como *K-Means Clustering* [31], es un algoritmo no supervisado que se basa en identificar grupos en los datos. Estos grupos, también denominados clústeres, engloban a un conjunto de datos con características análogas.

El objetivo de este algoritmo es agrupar observaciones similares para descubrir patrones en los datos. Para ello, el algoritmo busca un número k de clústeres en el conjunto de datos. Para el problema de clasificación binaria planteado en el trabajo, se fija el valor de k en 2, de modo que se tenga un clúster asociado a tráfico benigno y otro asociado a tráfico de ataques.

Capítulo 5

Experimentación y discusión de resultados

En este capítulo se exponen los resultados obtenidos, a la par que se extraen conclusiones en cada conjunto de datos. Para ello, primero se explica el procesado de los datos llevado a cabo antes de ser evaluados por los modelos. En segundo lugar, se describen las métricas usadas en la evaluación. Por último, se discuten los resultados de cada conjunto de datos, comparando sus métricas y calificando los modelos según su capacidad para clasificar datos de forma correcta.

5.1. Limpieza y preprocesamiento de datos

Como se indica en el Capítulo 3, ciertas variables o características necesitan eliminarse de los conjuntos de datos, ya sea porque no aportan información útil al clasificador o porque no son parámetros adecuados que el modelo deba tener en cuenta. Un ejemplo de ello son las características relacionadas con direcciones IP y puertos presentes en el conjunto de datos UNSW_NB15, correspondientes a las 5 primeras columnas. Estas operaciones realizadas en los conjuntos de datos pertenecen a la fase de **limpieza de datos**.

Sin embargo, existen otras medidas que se deben aplicar a los datos para garantizar que el modelo sea entrenado eficientemente. Este grupo de operaciones se encuentra dentro del llamado **preprocesamiento de datos**, que engloba todas las operaciones relacionadas con la limpieza, integración, transformación y reducción de datos, de cara a la fase de minado de datos [32]. En este trabajo se han acudido a las prácticas de preprocesado de datos más comunes. Tras eliminar las características altamente correladas en los conjuntos de datos, se ha procedido de la siguiente manera:

- Se ha agregado la característica *network_bytes*, resultado de sumar los bytes de datos de fuente y destino. Este paso ha sido realizado en los conjuntos de datos NSL_KDD y UNSW_NB15, explicados en las Secciones 3.1 y 3.2, respectivamente.

- Para columnas con un gran número de valores únicos, se ha calculado su coeficiente de correlación con la variable *label*, que indica si la trama procede de un ataque o es tráfico benigno. Se ha calculado la correlación estándar y la correlación resultante al aplicar la transformación del logaritmo de $1 + x$, conocida como la transformación *log1p*. Aquellas columnas que, al ser transformadas, muestran mejor correlación con la variable *label* se conservan y en su lugar la columna sin transformar es eliminada.

- Para características numéricas, se han estandarizado sus valores mediante el uso de un escalador estándar. El principal objetivo es normalizar las características, estableciendo media nula y desviación típica con valor unidad.

- Para características categóricas, se ha aplicado codificación *One Hot* [33]. Este tipo de codificador asigna una columna binaria para cada valor distinto que exista en la característica, donde un valor '0' indica que dicho valor no se encuentra presente y un valor '1' lo contrario. Con ello, se consigue transformar las variables categóricas en numéricas.

- Por último, se construye para el conjunto de datos de entrenamiento y para el conjunto de datos de prueba una matriz dispersa. Este tipo de matrices resuelve algunos problemas de memoria y computación a la hora de trabajar con un modelo, y son muy útiles en conjuntos de datos grandes como los usados en el trabajo.

Una vez realizadas todas las operaciones mencionadas anteriormente, los datos están listos para ser procesados por los modelos elegidos. Antes de obtener los resultados, se debe realizar un estudio sobre los hiperparámetros de cada modelo. Los parámetros establecidos para cada modelo difieren entre un conjunto de datos u otro, ya que los tres conjuntos de datos del trabajo no poseen las mismas características y no se pueden modelar de la misma forma.

5.2. Métricas de evaluación de modelos

A continuación se detallan las métricas utilizadas en el trabajo para la evaluación de los modelos, tanto supervisados como no supervisados. Cabe destacar que algunas de las métricas propuestas se deben interpretar parcialmente para los algoritmos no supervisados por las razones expuestas en la Sección 4.2.

Antes de presentar las métricas, es necesario introducir el concepto de matriz de confusión [34]. Una matriz de confusión es una representación matricial de los resultados de las predicciones, y es frecuentemente utilizada para describir el rendimiento del clasificador sobre un conjunto de datos de prueba cuyos valores reales se desconocen. En el caso de matrices de confusión con tráfico normal y tráfico de ataques, se tienen 4 parámetros con los que se pueden describir la mayoría de métricas usadas:

- **Verdadero Positivo (TP):** Tráfico de ataque predicho como un ataque.
- **Verdadero Negativo (TN):** Tráfico normal predicho como tráfico normal.
- **Falso Positivo (FP):** Tráfico normal predicho como ataque.
- **Falso Negativo (FN):** Tráfico de ataque predicho como tráfico normal.

5.2.1. Exactitud

La *exactitud* [35] indica la precisión del modelo a la hora de clasificar los datos, y es una métrica adecuada si las clases son aproximadamente iguales en tamaño; es decir, cuando el conjunto de datos se encuentra balanceado. Representa el número de predicciones correctas como una proporción de todas las predicciones, tal y como se describe en la Ecuación (5.1).

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

5.2.2. Tasa de verdaderos positivos

La *tasa de verdaderos positivos* (TPR), también conocida como *exhaustividad* o *sensibilidad* [36], es una medida de la probabilidad de que una instancia positiva real se clasifique como positiva. En este trabajo, indica la probabilidad de que un ataque se clasifique correctamente. El cálculo de la TPR se indica en la Ecuación (5.2).

$$TPR = \frac{TP}{TP + FN} \tag{5.2}$$

5.2.3. Tasa de falsos positivos

La *tasa de falsos positivos* (FPR) [36] es una medida de la frecuencia con la que una instancia negativa real se clasifica como positiva, fenómeno conocido como "falsa alarma". De forma análoga a la métrica anterior, esta indica la probabilidad de que el tráfico normal sea catalogado como ataque. El cálculo de dicha probabilidad se define en la Ecuación (5.3).

$$FPR = \frac{FP}{FP + TN} \tag{5.3}$$

No se debe confundir la FPR con la métrica de *precisión* [35], descrita en la Ecuación (5.4).

$$Precision = \frac{TP}{TP + FP} \tag{5.4}$$

Las métricas de exhaustividad y precisión son usadas cuando las clases del conjunto de datos están balanceadas. Sin embargo, las métricas de FPR y TPR son más usadas en el caso contrario. Este factor se tendrá en cuenta en el análisis de resultados mostrado en la Sección 5.3.

5.2.4. Área bajo la curva

El *área bajo la curva* (del inglés, *Area Under the Curve*, o AUC) [36], es una métrica que indica el rendimiento de un clasificador binario. Se fundamenta en medir el área bajo la curva ROC de un clasificador. Para entender mejor el concepto de la curva ROC, se muestra en la Figura 5.1 una representación gráfica de los valores de la curva ROC en función de la tasa de valores positivos (TPR) y la tasa de valores negativos (FPR).

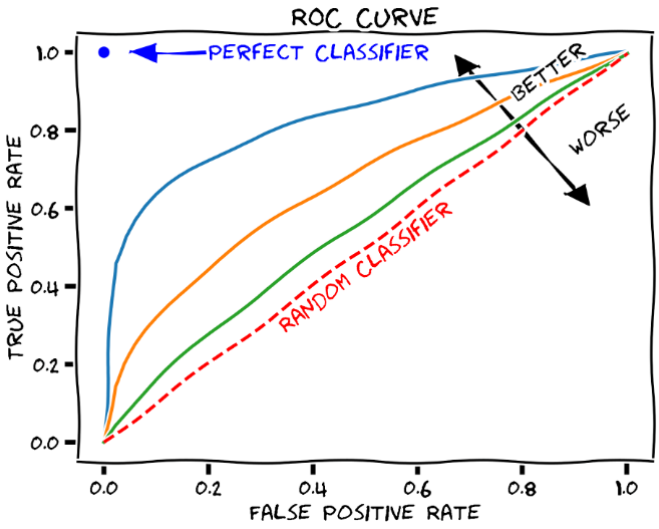

Figura 5.1: Gráfica de la curva ROC (obtenida de [36])

Cuanto mayor es la puntuación AUC, mejor es el rendimiento del modelo. El peor caso se obtiene con un valor de AUC igual a 0.5, ya que esta puntuación indica que el modelo realiza predicciones de forma aleatoria. Por otro lado, valores inferiores a 0.5 indican que el modelo está realizando predicciones invertidas, es decir, tendería a predecir la clase negativa como positiva y viceversa.

5.2.5. F1-Score

La *puntuación F1* [35] corresponde a la media armónica entre las medidas de precisión y exhaustividad de un modelo, combinándolas en una sola métrica. Esta métrica asume que tanto la precisión como la exhaustividad tienen la misma importancia en el modelo. La fórmula se muestra a continuación en la Ecuación (5.5).

$$F1 = 2 \times \frac{precision \times exhaustividad}{precision + exhaustividad} \tag{5.5}$$

5.2.6. Coeficiente kappa de Cohen

El *coeficiente Kappa de Cohen* [37] es una medida de concordancia utilizada como complemento de la exactitud en casos de clasificación multiclase o conjuntos de datos no equilibrados. Sirve para indicar cómo de mejor es el modelo en comparación a un clasificador que realiza estimaciones de forma aleatoria, según la frecuencia de cada clase. La Ecuación (5.6) muestra su definición, donde p_o es la concordancia observada y p_e es la concordancia esperada.

$$K = \frac{p_o - p_e}{1 - p_e} \tag{5.6}$$

5.2.7. Coeficiente de correlación de Matthews

El *coeficiente de correlación de Matthews* (MCC) [38] es una métrica utilizada en clasificación binaria en conjuntos de datos con clases no balanceadas, en las que una medida de la exactitud no es suficiente. El MCC toma valores entre -1 y 1, obteniendo una clasificación perfecta si el valor es igual a la unidad. La Ecuación (5.7) describe su cálculo, a través de los coeficientes expuestos en la matriz de confusión.

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{5.7}$$

5.3. Resultados

En esta sección se presentan los datos obtenidos para los conjuntos de datos usados en el trabajo. Se describe el comportamiento de los modelos en la tarea de clasificación, interpretando las métricas y extrayendo conclusiones sobre los mejores modelos en cada conjunto de datos. Para cada métrica y conjunto de datos, se han destacado en la tabla correspondiente los peores y los mejores resultados en rojo y en verde, respectivamente.

Cabe destacar que, en el caso de NSL_KDD, el conjunto de datos ya se encuentra dividido en un conjunto de entrenamiento y en un conjunto de prueba. Sin embargo, esto no se cumple para los conjuntos de datos UNSW_NB15 y CIC_IDS2017. Para ello, antes de evaluar los modelos se dividen los conjuntos de datos mediante una proporción de 70 % para entrenamiento y 30 % para prueba. Esta proporción es frecuentemente usada, al ser de las relaciones más consideradas para obtener el mejor rendimiento de los modelos de aprendizaje máquina [39].

5.3.1. Conjunto de datos NSL_KDD

Los resultados obtenidos en el conjunto de entrenamiento se muestran en la Tabla 5.1 coinciden con lo esperado, ya que existe una gran diferencia en la clasificación entre los modelos supervisados y los no supervisados. Al observar los resultados, se comprueba que el algoritmo que peores prestaciones ofrece es el algoritmo *Isolation Forest*, con tan sólo una exactitud de 53 % y una puntuación de AUC de 0.5, lo que indica que el algoritmo está realizando predicciones de forma aleatoria. Esto se puede contrastar con el resto de métricas, que resultan excesivamente bajas (por ejemplo, un MCC de 0.00681). Por otro lado, el algoritmo *K-Means Clustering* ha conseguido clasificar el tráfico con una exactitud de 89 %. Este resultado depende en gran medida de que el algoritmo asigne el tráfico normal y los ataques a su clúster correspondiente, pudiendo obtener una exactitud inversa (11 %) si no se cumple el requisito anterior.

Conjunto de datos NSL_KDD - Train							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.97855	0.97831	0.97484	0.01822	0.97690	0.95688	0.95689
SVM	0.98115	0.98068	0.97392	0.01254	0.97963	0.96209	0.96215
DT	0.99843	0.99841	0.99807	0.00124	0.99831	0.99685	0.99685
RF	0.99996	0.99996	0.99997	5.56e-05	0.99995	0.99992	0.99992
XGB	0.99994	0.99994	1	1.11e-04	0.99993	0.99988	0.99988
IF	0.53469	0.50014	5.79e-04	2.96e-04	0.00115	0.00030	0.00681
KM	0.89042	0.88637	0.82781	0.05506	0.87550	0.77824	0.78237

Tabla 5.1: Resultados del conjunto de datos NSL_KDD en el conjunto de entrenamiento

Respecto a los algoritmos supervisados, se obtienen excelentes resultados. Las tasas de verdaderos positivos más bajas pertenecen a los algoritmos de Regresión Logística y *Support Vector Machines*, lo que indica que dichos algoritmos detectan los ataques con menor frecuencia que el resto, tal y como indican los coeficientes Kappa y MCC.

Las mejores prestaciones a nivel general las ofrece el algoritmo *Random Forest*, que posee mayor exactitud, mayor área bajo la curva, menor tasa de falsos positivos, etc. Sin embargo, el algoritmo *Gradient Boosted Decision Tree* alcanza una tasa de verdaderos positivos perfecta, consiguiendo clasificar todos los ataques de forma correcta.

Conjunto de datos NSL_KDD - Test							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.75168	0.77233	0.62323	0.07857	0.74076	0.51823	0.55332
SVM	0.75616	0.77663	0.62884	0.07558	0.74594	0.52668	0.56155
DT	0.83361	0.84832	0.74207	0.04541	0.83546	0.67249	0.69385
RF	0.77688	0.80081	0.62799	0.02636	0.76215	0.56940	0.61745
XGB	0.79555	0.81695	0.66243	0.02852	0.78672	0.60326	0.64375
IF	0.43164	0.50071	0.00194	5.14e-04	0.00388	0.00123	0.01946
KM	0.43124	0.50042	8.57e-04	0	0.00171	0.00073	0.01922

Tabla 5.2: Resultados del conjunto de datos NSL_KDD en el conjunto de prueba

No obstante, los resultados obtenidos en el proceso de evaluación difieren con los obtenidos en el entrenamiento. En la Tabla 5.2 se refleja la incapacidad de los algoritmos no supervisados para clasificar el tráfico de prueba: tanto el algoritmo *Isolation Forest* como el algoritmo *K-Means Clustering* tienen una exactitud de 43 % y un AUC de 0.5. Este hecho, sumado al hecho de que tanto las tasas TPR y FPR como los coeficientes son excesivamente bajos, permite afirmar que los algoritmos no supervisados realizan predicciones aleatorias, sin usar un criterio adecuado en el contexto evaluado.

Por otro lado, el algoritmo *Decision Tree* obtiene los mejores resultados de forma general, con una exactitud de 83 % y un AUC de aproximadamente 0.85, a pesar de que la tasa de falsos positivos más baja la sigue teniendo el algoritmo *Random Forest*. Tanto las prestaciones del anterior algoritmo como del *Gradient Boosted Decision Tree* disminuyen considerablemente respecto al conjunto de entrenamiento, especialmente en el coeficiente Kappa de RF. Los algoritmos de Regresión Logística y *Support Vector Machines* obtienen unos resultados inferiores a los demás algoritmos, pero cumplen la tarea de clasificación con mayor precisión que los algoritmos no supervisados. Por lo tanto, se puede confirmar que los modelos supervisados poseen una clara ventaja a la hora de clasificar el trafico en este conjunto de datos en particular.

Para explicar la diferencia de prestaciones al evaluar el conjunto de entrenamiento y el de prueba en el conjunto de datos NSL_KDD, es recomendable visualizar la distribución de ataques reflejada en el Apartado C.1, en la que se muestra que el conjunto de prueba contiene mayor número de tipos de ataque. Como se mencionaba en la Tabla 3.2, se puede comprobar que el conjunto de entrenamiento tiene un gran porcentaje de ataques de denegación de servicios y de *probing*. Sin embargo, en el conjunto de prueba se incluyen numerosos ataques de tipo remoto a local. Por ende, el problema principal reside en que el conjunto de entrenamiento no se encuentra balanceado, es decir, no contiene una cantidad similar de ataques en los cuatro tipos de grupo. Este factor puede favorecer la detección de los ataques más frecuentes y a su vez dificultar la detección de los ataques menos conocidos, resultando en un sobreajuste [40] del modelo a los datos de entrenamiento.

5.3.2. Conjunto de datos UNSW_NB15

Los resultados obtenidos para los algoritmos no supervisados en el conjunto de entrenamiento de UNSW_NB15 resultan similares a los mostrados en el conjunto de datos NSL_KDD. Tal como muestra la Tabla 5.3, el algoritmo *Isolation Forest* obtiene de nuevo un área bajo la curva aproximado de 0.5 y unos coeficientes bajos de F1, Kappa y MCC, aunque mejora su exactitud respecto al caso anterior. Por otro lado, el algoritmo *K-Means Clustering* obtiene resultados poco relevantes, con una FPR muy elevada y peores prestaciones respecto al conjunto de datos NSL_KDD.

Conjunto de datos UNSW_NB15 - Train							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.98862	0.98599	0.98247	0.01048	0.95628	0.94975	0.95019
SVM	0.98779	0.97746	0.96363	0.00870	0.95235	0.94535	0.94544
DT	0.99054	0.98796	0.98450	0.00857	0.96348	0.95805	0.95834
RF	0.99739	0.99271	0.98644	0.00101	0.98969	0.98820	0.98820
XGB	0.99807	0.99560	0.99229	0.00108	0.99239	0.99129	0.99129
IF	0.85677	0.54967	0.13840	0.03905	0.19662	0.13302	0.14929
KM	0.88958	0.85616	0.81140	0.09907	0.65050	0.58797	0.60473

Tabla 5.3: Resultados del conjunto de datos UNSW_NB15 en el conjunto de entrenamiento

Por lo general, todos los algoritmos supervisados mantienen resultados similares en entrenamiento respecto a los obtenidos en el apartado anterior. En el caso de la Regresión Logística y *Support Vector Machines*, ambos algoritmos disminuyen sus coeficientes F1, Kappa y MCC, pero obtienen mayor exactitud y menor tasa de falsos positivos. El resto de algoritmos (DT, RF y XGB) sufren una ligera pérdida de prestaciones, pero conservan resultados prometedores como una exactitud y un porcentaje de AUC aproximado de 99 %.

Sin embargo, la diferencia con el conjunto de datos previo se resalta en el conjunto de prueba. Al consultar la Tabla 5.4 se observa de nuevo que el algoritmo *Isolation Forest*, a pesar de obtener una mayor AUC, es incapaz de clasificar los ataques como anomalías de forma consistente, tal y como muestra la TPR y los tres últimos coeficientes. Por otro lado, el algoritmo *K-Means Clustering* es capaz de distinguir razonablemente el tráfico normal de los ataques, algo que no ocurría en el caso anterior. Sin embargo, también es el algoritmo con mayor tasa de falsos positivos, clara desventaja en clasificación frente a los algoritmos supervisados.

Conjunto de datos UNSW_NB15 - Test							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.98856	0.98571	0.98189	0.01046	0.95588	0.94932	0.94976
SVM	0.98759	0.97682	0.96242	0.00877	0.95139	0.94429	0.94437
DT	0.99042	0.98773	0.98415	0.00867	0.96284	0.95735	0.95764
RF	0.99417	0.98603	0.97515	0.00308	0.97685	0.97351	0.97351
XGB	0.99407	0.98595	0.97509	0.00318	0.97648	0.97310	0.97310
IF	0.85786	0.57766	0.20292	0.04760	0.26478	0.19413	0.20596
KM	0.88931	0.85560	0.81051	0.09931	0.64878	0.58617	0.60307

Tabla 5.4: Resultados del conjunto de datos UNSW_NB15 en el conjunto de prueba

Dentro de los algoritmos supervisados, tanto el algoritmo *Support Vector Machines* como el algoritmo de Regresión Lógica muestran una sorprendente mejora, obteniendo aproximadamente las mismas prestaciones que en el conjunto de entrenamiento. Por otro lado, el algoritmo *Decision Tree* obtiene la mayor puntuación de AUC (0.9877) y la mayor TPR entre todos los algoritmos supervisados. No obstante, las mejores prestaciones de forma general las ofrecen los algoritmos *Gradient Boosted Decision Tree* y *Random Forest*, alcanzando este último la mayor exactitud, la menor tasa de falsos positivos y los mejores coeficientes de F1, Kappa y MCC.

Al dividir el conjunto de datos UNSW_NB15 en su totalidad, mediante una proporción del 70 % para entrenamiento y 30 % para evaluación, se asegura en el entrenamiento que los modelos propuestos ajusten su criterio en base a los ataques que deben detectar. Al haber un menor número de tipos de ataque respecto al anterior conjunto de datos, se ofrece una mejor respuesta a la hora de clasificar los ataques. Tal vez se podría obtener una mejora en la precisión del modelo si, en vez de usar un modelo que clasifique todos los tipos de ataque, se diseñase un modelo principal construido por otros modelos, cada uno con la función de detectar un tipo de ataque en concreto. Esta idea planteada deriva del llamado *aprendizaje conjunto* (del inglés, *Ensemble Learning*) [41], una técnica usada para mejorar la precisión en la clasificación.

5.3.3. Conjunto de datos CIC_IDS2017

Por último, los resultados obtenidos en el conjunto de datos CIC_IDS2017 resultan satisfactorios a la par que sorprendentes. Ya que el conjunto de datos se encuentra desequilibrado, métricas como la exactitud no son igual de aptas para comparar modelos como lo son los coeficientes Kappa y el MCC. Tal como muestra la Tabla 5.5, el algoritmo *Isolation Forest* sigue siendo el algoritmo con peor rendimiento en el entrenamiento para todas las métricas. Por otro lado, el algoritmo *K-Means Clustering* no realiza el agrupamiento de forma consistente, tal y como indica el coeficiente de Kappa. Se comprueba de nuevo que los algoritmos no supervisados no son aptos para tareas de clasificación en este contexto.

Conjunto de datos CIC_IDS2017 - Train							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.94845	0.92158	0.87726	0.03409	0.87015	0.83799	0.83804
SVM	0.95337	0.90960	0.83740	0.01819	0.87612	0.84748	0.84885
DT	0.99686	0.99481	0.99144	0.00180	0.99203	0.99007	0.99007
RF	0.99900	0.99856	0.99784	7.12e-04	0.99746	0.99684	0.99684
XGB	0.99926	0.99920	0.99908	6.87e-04	0.99814	0.99768	0.99768
IF	0.78456	0.54354	0.14596	0.05887	0.21061	0.11335	0.13067
KM	0.83008	0.63852	0.32251	0.04547	0.42774	0.34022	0.36719

Tabla 5.5: Resultados del conjunto de datos CIC_IDS2017 en el conjunto de entrenamiento

Respecto a los algoritmos supervisados, se destacan las altas prestaciones que ofrece el algoritmo *Gradient Boosted Decision Tree*, superando al algoritmo *Random Forest* incluso en la tasa de falsos positivos. En este caso, se prueba la teoría de que el rendimiento del algoritmo XGB puede ser superior al de RF si es usado en conjuntos de datos no balanceados, tal y como se tiene en el tráfico en tiempo real [42]. No obstante, las prestaciones de los algoritmos *Random Forest* y *Decision Tree* son muy próximas a las obtenidas en el algoritmo XGB, siendo DT el peor de los tres casos. Por otro lado, los algoritmos de Regresión Logística y *Support Vector Machines* alcanzan un rendimiento mucho menor que el obtenido en el conjunto de datos UNSW_NB15, tal como indican los coeficientes de F1, Kappa y MCC (aproximadamente 87 %) frente a los mostrados en la Tabla 5.3 (en torno a 95 %).

Conjunto de datos CIC_IDS2017 - Test							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
LR	0.94870	0.92212	0.87832	0.03407	0.87068	0.83869	0.83873
SVM	0.95389	0.91091	0.84007	0.01825	0.87752	0.84919	0.85047
DT	0.99671	0.99461	0.99115	0.00191	0.99165	0.98961	0.98961
RF	0.99861	0.99785	0.99660	8.90e-04	0.99648	0.99562	0.99562
XGB	0.99901	0.99885	0.99857	8.75e-04	0.99750	0.99689	0.99689
IF	0.79020	0.54071	0.12953	0.04811	0.19536	0.10921	0.13210
KM	0.83140	0.64409	0.33539	0.04720	0.43891	0.35065	0.37542

Tabla 5.6: Resultados del conjunto de datos CIC_IDS2017 en el conjunto de prueba

Si se observa la Tabla 5.6, se puede comprobar para todos los algoritmos que los resultados obtenidos apenas varían respecto al conjunto de entrenamiento. En el caso de los modelos no supervisados, se sigue sin alcanzar resultados positivos en los coeficientes Kappa y MCC, tanto para el algoritmo IF como para KM. Sin embargo, los modelos supervisados se han ajustado a los datos de entrenamiento y clasifican el tráfico de prueba con una gran precisión. El algoritmo XGB sigue manteniendo las mejores prestaciones, seguido de RF y de DT. Los coeficientes Kappa y las puntuaciones AUC de los tres anteriores algoritmos demuestran que son capaces de realizar estimaciones correctas en datos nunca vistos. Los algoritmos LR y SVM mejoran ligeramente sus resultados respecto al conjunto de entrenamiento, pero en este caso siguen sin poder competir con los algoritmos basados en árboles de decisión.

Los resultados obtenidos se asemejan a los mostrados para el conjunto de datos UNSW_NB15. Ambos conjuntos comparten la proporción de tráfico en entrenamiento y prueba, además de ser conjuntos no balanceados en los que el tráfico normal representa la mayoría. Por ello, es lógico que los modelos hayan obtenido resultados similares. Tal vez una forma de mejorar los resultados obtenidos en el conjunto NSL_KDD sea agrupar el conjunto de entrenamiento y prueba en un solo conjunto, para después dividirlo siguiendo la proporción propuesta para los otros dos conjuntos de datos. De esta forma, los modelos son entrenados para un mayor número de tipos de ataque y pueden abarcar una región de la curva ROC mayor.

5.4. Resultados con características importantes

Una mejora que se puede realizar a los modelos es la comentada en la Sección 5.3.2: establecer un modelo de aprendizaje conjunto que incorpore los mejores modelos propuestos (en este caso, *Decision Tree*, *Random Forest* y *Gradient Boosted Decision Tree*) mediante un modelo de sistema de voto mayoritario (de ahí el nombre de *Voting Classifier*).

Además de establecer dicho modelo, se propone otra mejora en los algoritmos DT y RF, en la que el modelo solo procesa una cantidad fija de características, en función de la importancia de dichas características para la clasificación. Estos modelos mejorados se han denominado DT_FI y RF_FI. Las gráficas que muestran las características más relevantes para la clasificación en cada conjunto de datos se encuentran en el Apéndice D.

5.4.1. Conjunto de datos NSL_KDD

En el caso del conjunto de datos NSL_KDD, se aprecia en la Tabla 5.7 que el modelo *Random Forest* ofrece mejores prestaciones que el algoritmo *Decision Tree* y el modelo de aprendizaje conjunto en la fase de entrenamiento. Respecto a los resultados obtenidos anteriormente, el algoritmo DT sufre una pérdida minúscula de prestaciones, mientras que el algoritmo RF obtiene los mismos resultados que los descritos en la Sección 5.3.1. Respecto al modelo *Voting Classifier* se obtienen resultados bastante prometedores, superando las prestaciones alcanzadas por DT en entrenamiento.

Conjunto de datos NSL_KDD - Train con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.99804	0.99799	0.99728	0.00129	0.99789	0.99607	0.99607
RF_FI	0.99996	0.99996	0.99997	5.56e-05	0.99995	0.99992	0.99992
Voting	0.99986	0.99986	0.99985	1.29e-04	0.99985	0.99972	0.99972

Tabla 5.7: Resultados del conjunto de datos NSL_KDD en el conjunto de entrenamiento con parámetros importantes

Sin embargo, en la Tabla 5.8 se muestra que en el conjunto de prueba no se mantiene la misma jerarquía. En su lugar, DT alcanza los mejores resultados en exactitud, TPR, puntuación F1 y coeficiente Kappa. Por otro lado, el sistema de voto consigue la mejor puntuación AUC y el mejor MCC. Finalmente, el algoritmo RF, que había logrado los mejores resultados en el conjunto de entrenamiento, solo consigue obtener la mejor FPR entre los tres modelos, obteniendo las peores prestaciones en las demás métricas.

Conjunto de datos NSL_KDD - Test con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.81959	0.83219	0.74121	0.07682	0.82387	0.64363	0.66067
RF_FI	0.77413	0.79828	0.62386	0.02728	0.75871	0.56432	0.61294
Voting	0.81422	0.83338	0.69508	0.02831	0.80987	0.63770	0.67184

Tabla 5.8: Resultados del conjunto de datos NSL_KDD en el conjunto de prueba con parámetros importantes

Los cambios realizados en los modelos no parecen ser de gran ayuda a la hora de mejorar la precisión. De hecho, DT y RF han obtenido por lo general peores resultados en este conjunto de datos en particular. La razón de ello puede ser el hecho de no haber conseguido que los modelos se adapten al conjunto de datos con tanta precisión como se ha conseguido en el resto de conjuntos. Este factor se puede achacar a un mal diseño de la estructura de evaluación de los modelos, aunque también puede cobrar relevancia el desequilibrio entre los ataques estudiados para entrenamiento y para prueba.

5.4.2. Conjunto de datos UNSW_NB15

Por otra parte, para el conjunto de datos UNSW_NB15 se ha conseguido mejorar el rendimiento del algoritmo *Random Forest* en el entrenamiento, tal y como indican los coeficientes Kappa y MCC en la Tabla 5.9. El algoritmo *Decision Tree* ha sufrido una ligera pérdida de prestaciones, de forma análoga al apartado anterior. Se destaca que el modelo *Voting Classifier* ha obtenido la mejor puntuación AUC y la mejor TPR, siendo superado en el resto de métricas por RF. El rendimiento del sistema de voto es superior al del algoritmo DT, debido a que la combinación de los algoritmos de RF y XGB aportan un mejor enfoque en la clasificación que DT.

Conjunto de datos UNSW_NB15 - Train con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.99049	0.98771	0.98399	0.00856	0.96326	0.95781	0.95809
RF_FI	0.99800	0.99413	0.98895	6.81e-04	0.99210	0.99096	0.99097
Voting	0.99780	0.99472	0.99060	0.00115	0.99132	0.99006	0.99006

Tabla 5.9: Resultados del conjunto de datos UNSW_NB15 en el conjunto de entrenamiento con parámetros importantes

Los resultados obtenidos para el conjunto de prueba se muestran en la Tabla 5.10. En este caso, tanto el algoritmo DT como RF han mejorado respecto a los resultados obtenidos en la Sección 5.3.2, incrementando su puntuación AUC y obteniendo una menor FPR. Por otro lado el sistema de voto mantiene sus números respecto al conjunto de entrenamiento, si bien se aprecia una pérdida del 2 % en los coeficientes Kappa y MCC. Pese a eso, el *Voting Classifier* se percibe como una opción viable para adjuntar a un sistema de detección de intrusos.

Conjunto de datos UNSW_NB15 - Test con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.99031	0.98732	0.98333	0.00867	0.96242	0.95687	0.95715
RF_FI	0.99450	0.98638	0.97552	0.00275	0.97817	0.97503	0.97503
Voting	0.99443	0.98697	0.97700	0.00305	0.97791	0.97473	0.97473

Tabla 5.10: Resultados del conjunto de datos UNSW_NB15 en el conjunto de prueba con parámetros importantes

En este caso, se ha conseguido mejorar el rendimiento de los algoritmos DT y RF mediante el uso de características importantes. A pesar de que no son grandes cambios, se ha de tener en cuenta que es difícil aumentar la precisión del modelo cuando los números que ofrece ya son lo suficientemente altos. Además, el sistema de voto no ofrece las mejores métricas entre los tres modelos, pero ofrece un modelo flexible al estar conformado por varios algoritmos. Esto se puede verificar tanto en los resultados de este conjunto de datos como en los del anterior, ya que no ha obtenido la peor métrica en ninguno de los cuatro casos mencionados anteriormente.

5.4.3. Conjunto de datos CIC_IDS2017

Por último, los resultados obtenidos para el conjunto de datos CIC_IDS2017 resultan verdaderamente sorprendentes. Al echar un vistazo a la Tabla 5.11, se puede observar que el modelo con mejores métricas en la fase de entrenamiento es el *Voting Classifier*, que obtiene los mejores resultados a excepción de la FPR, en la que le supera el algoritmo *Random Forest*. En este caso, el algoritmo *Decision Tree* vuelve a sufrir una pérdida de prestaciones muy ligera a nivel general, siendo el peor modelo de los 3 expuestos. No obstante, el algoritmo RF sufre un cambio positivo de la misma magnitud, obteniendo mejores coeficientes Kappa y MCC al costo de un ligero incremento en la FPR.

Conjunto de datos CIC_IDS2017 - Train con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.99685	0.99479	0.99140	0.00181	0.99199	0.99004	0.99004
RF_FI	0.99901	0.99859	0.99789	7.17e-04	0.99748	0.99687	0.99687
Voting	0.99914	0.99902	0.99881	7.73e-04	0.99783	0.99729	0.99730

Tabla 5.11: Resultados del conjunto de datos CIC_IDS2017 en el conjunto de entrenamiento con parámetros importantes

Por otra parte, en la Tabla 5.12 se comprueba que en la fase de predicción de datos de prueba todos los algoritmos han sufrido un decremento en sus métricas, como es habitual. Respecto a DT, el algoritmo obtiene unas métricas inferiores a las obtenidas en la Sección 5.3.3. El algoritmo RF mantiene los mismos números en el conjunto de prueba, lo cual resulta interesante dado que se han eliminado ciertas características y se sigue manteniendo el mismo resultado. Además, el modelo conjunto vuelve a obtener las mejores métricas en el conjunto de prueba, pero no supera los resultados obtenidos en el conjunto de datos de prueba por el algoritmo *Gradient Boosted Decision Tree*.

Conjunto de datos CIC_IDS2017 - Test con parámetros importantes							
Modelo	**Exactitud**	**AUC**	**TPR**	**FPR**	**F1-Score**	**Kappa**	**MCC**
DT_FI	0.99671	0.99459	0.99109	0.00191	0.99163	0.98959	0.98959
RF_FI	0.99861	0.99788	0.99667	9.06e-04	0.99648	0.99562	0.99562
Voting	0.99889	0.99863	0.99819	9.34e-04	0.99719	0.99650	0.99650

Tabla 5.12: Resultados del conjunto de datos CIC_IDS2017 en el conjunto de prueba con parámetros importantes

Una vez comprobados todos los conjuntos de datos, se aprecia un patrón general con los cambios realizados. Por un lado, las métricas del algoritmo DT no han mejorado al eliminar las características no importantes, pero esto no se aplica al algoritmo RF. Por lo tanto, es posible que se pueda lograr un avance mayor en la clasificación usando el algoritmo RF, especialmente para el caso del conjunto de datos NSL_KDD. Por otro lado, el modelo de voto mayoritario solo es válido en el caso de que los modelos que lo conforman sean entrenados adecuadamente, tal y como muestran los resultados obtenidos para los conjuntos UNSW_NB15 y CIC_IDS2017.

Capítulo 6

Conclusiones y trabajo futuro

Este capítulo recopila las conclusiones obtenidas en la elaboración de este trabajo. Tras mostrar dichas conclusiones, se describen líneas de trabajo futuro a las que se puede orientar la investigación realizada en este trabajo, en orden de mejorar la calidad de la clasificación de ataques en sistemas de control industrial.

6.1. Conclusiones

Tras la evaluación de los modelos supervisados y no supervisados en los tres conjuntos de datos propuestos, se puede concluir que un sistema de detección de intrusos en red basado en aprendizaje máquina de tipo supervisado tiene un rendimiento superior frente a las técnicas no supervisadas en detección de ataques de red. No obstante, se considera el uso de algoritmos de detección de anomalías como complemento en un IDS, ya que puede aportar un enfoque distinto de los algoritmos supervisados y puede resultar útil a la hora de detectar patrones desconocidos.

Dado que en el conjunto de datos NSL_KDD no se han obtenido resultados tan encomiables como en el resto de los conjuntos de datos, se requiere una investigación más profunda para optimizar los algoritmos evaluados o la estructura modelo. Por otro lado, los algoritmos basados en el árbol de decisión, ya sea *Decision tree*, *Random Forest* o *Gradient Boosted Decision Tree*, muestran un comportamiento superior en la clasificación de tráfico benigno y maligno al resto de algoritmos usados en este trabajo. Sin embargo, el algoritmo *Random Forest* ha demostrado ser el candidato más apto en la detección de ataques conocidos, presentando los mejores coeficientes y registrando la menor tasa de falsos positivos en la mayoría de los casos.

También cabe destacar que, a pesar de evaluar modelos a la hora de incorporarlos en una red de control industrial, se ha decidido usar finalmente conjuntos de datos basados en detección de intrusos. Esto se debe a la falta de acceso a datos de redes de control industrial, lo cual dificulta la tarea de evaluar modelos con su respectivo tráfico. Además, para predecir correctamente los datos de una red de control industrial, estos deben ser procesados con las mismas características con las que se entrenó el modelo. Este factor dificulta enormemente la tarea de comparar o unir conjuntos de datos de control industrial.

6.2. Trabajo futuro

Como se ha mencionado en la Sección 5.3.2, una de las líneas de futuro más interesantes para este trabajo sería el diseño de un modelo de aprendizaje conjunto más completo que el estudiado en el trabajo. Con ello, se podría elevar la precisión en la detección de ataques y construir un modelo robusto y flexible. Otra línea de futuro relacionada con los modelos sería el diseño de un modelo basado en redes neuronales [43].

Además, se propone una optimización en el preprocesamiento de datos mediante el uso de técnicas como la eliminación recursiva de características [44] y el análisis de componentes principales [45].

Como última línea de futuro propuesta, se sugiere la construcción de un conjunto de datos de control industrial. La falta de conjuntos de datos de control industrial públicos es uno de los mayores problemas que ha tenido este trabajo. Es por ello que el diseño de un nuevo conjunto de datos de control industrial, que sirva de base para comparar modelos, sería un trabajo futuro de gran relevancia para la comunidad en aras de mejorar los mecanismos de detección de intrusos en ICS.

Apéndice A

Planificación temporal y esfuerzo

En este apéndice se muestra la planificación temporal y el esfuerzo llevado a cabo en este trabajo. Primero, se muestra una tabla con el reparto de horas en cada actividad. Después, se adjunta un diagrama de Gantt que refleja la planificación de tiempo que se ha seguido en el trabajo.

A.1. Horas dedicadas al trabajo

Tareas	Horas
Reuniones con el tutor	12
Estudio de sistemas de control industrial (ICS)	32
Estudio de sistemas de detección de intrusos (IDS)	49
Estudio de aprendizaje máquina (ML)	56
Búsqueda de conjuntos de datos de ICS e IDS	24
Elección de algoritmos	47
Experimentación y optimización	110
Documentación de la memoria	183
Total	**513**

Tabla A.1: Horas dedicadas al trabajo.

A.2. Diagrama de Gantt

El diagrama de Gantt del trabajo muestra el periodo total de la elaboración y se encuentra dividido en dos figuras. Las tareas de estudio previo y búsqueda de conjuntos de datos se encuentran en la Figura A.1, mientras que las tareas de elección de algoritmos, experimentación y redacción de la memoria se detallan en la Figura A.2.

Figura A.1: Diagrama de Gantt de la primera parte del trabajo

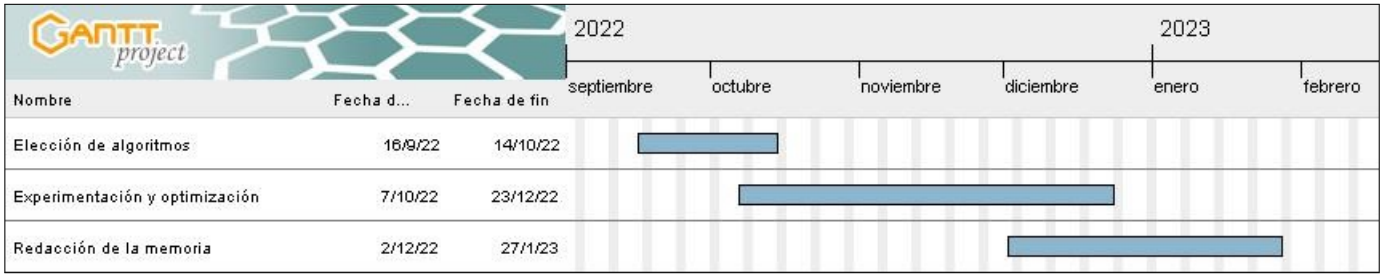

Figura A.2: Diagrama de Gantt de la segunda parte del trabajo

Apéndice B

Gráficas de correlación

B.1. Conjunto de datos NSL_KDD

B.1.1. Variables num_compromised y num_root

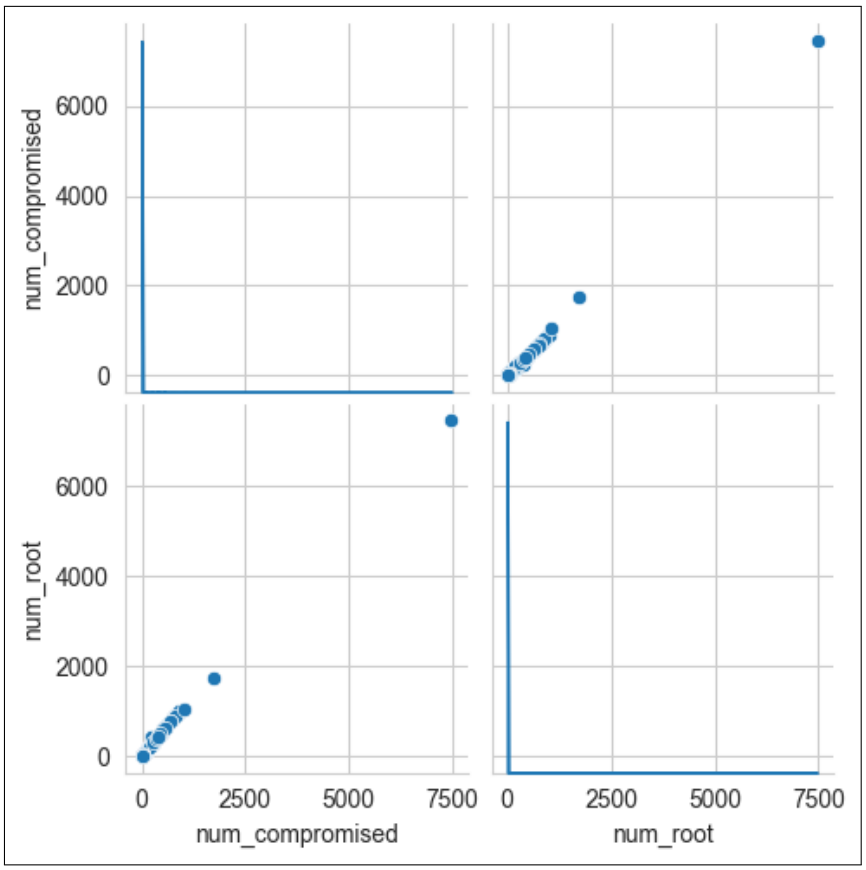

Figura B.1: Correlación entre *num_compromised* y *num_root*

B.1.2. Variables de grupo serror

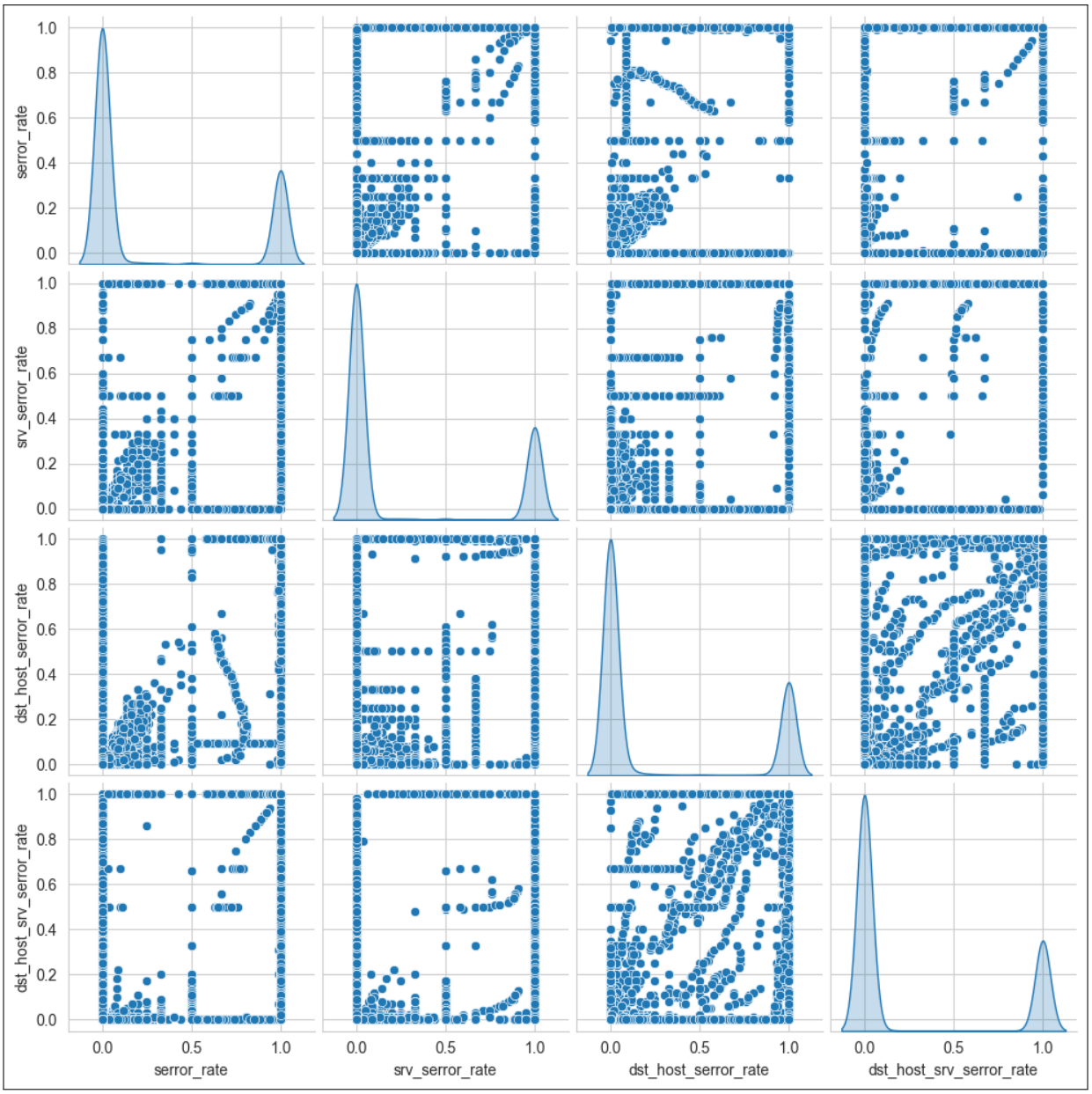

Figura B.2: Correlación entre columnas *serror*

B.1.3. Variables de grupo rerror

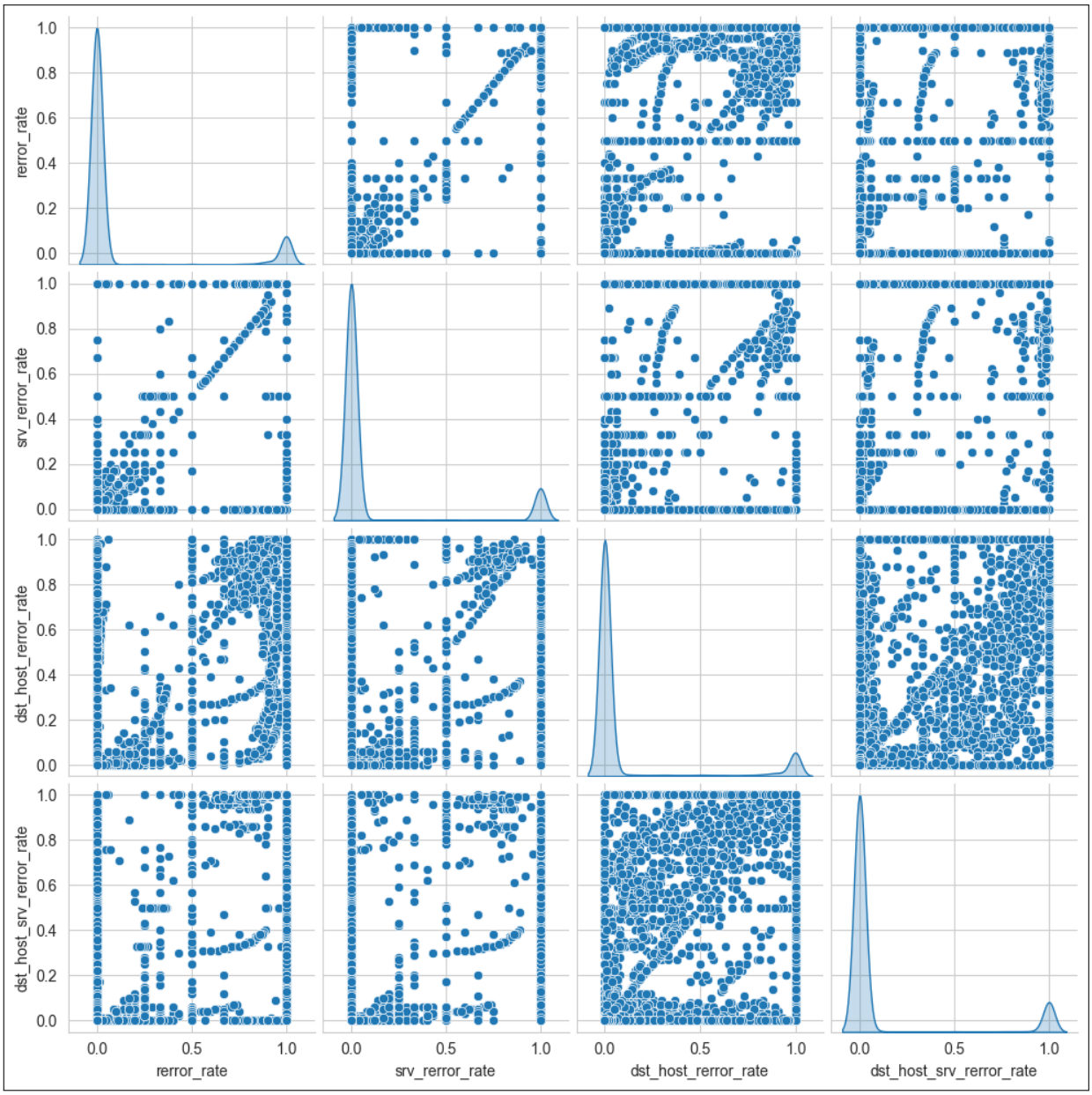

Figura B.3: Correlación entre columnas *rerror*

B.2. Conjunto de datos UNSW_NB15

B.2.1. Variables sbytes y sloss

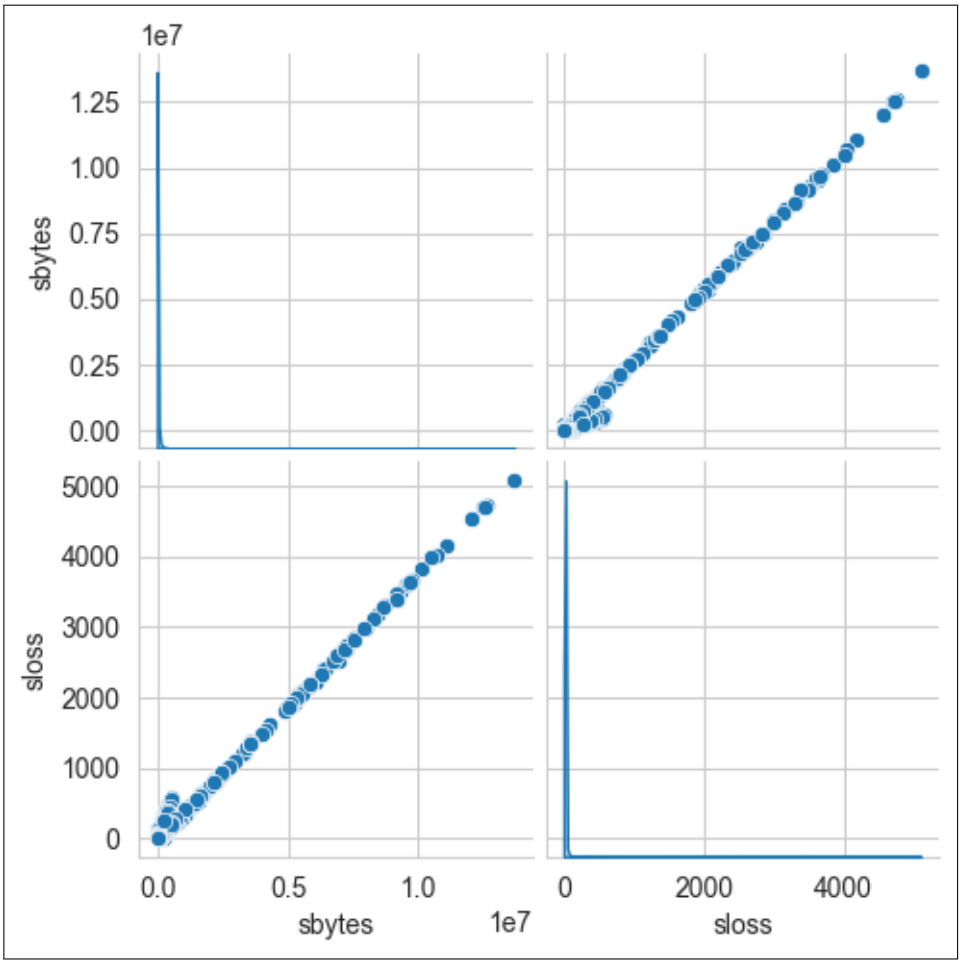

Figura B.4: Correlación entre columnas *sbytes* y *sloss*

B.2.2. Variables dpkts, dbytes y dloss

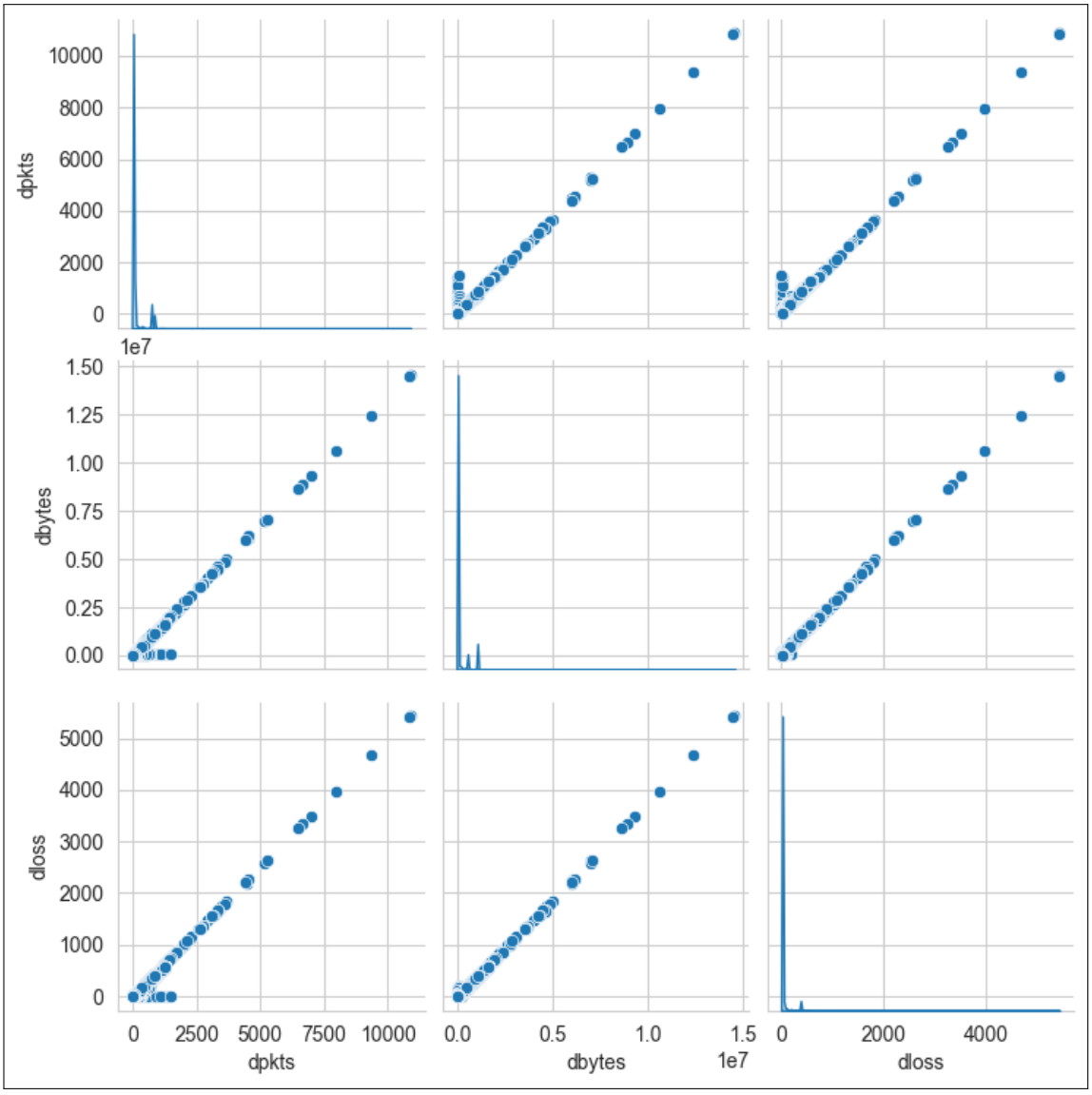

Figura B.5: Correlación entre columnas *dpkts*, *dbytes* y *dloss*.

B.2.3. Variables sttl, ct_state_ttl y label

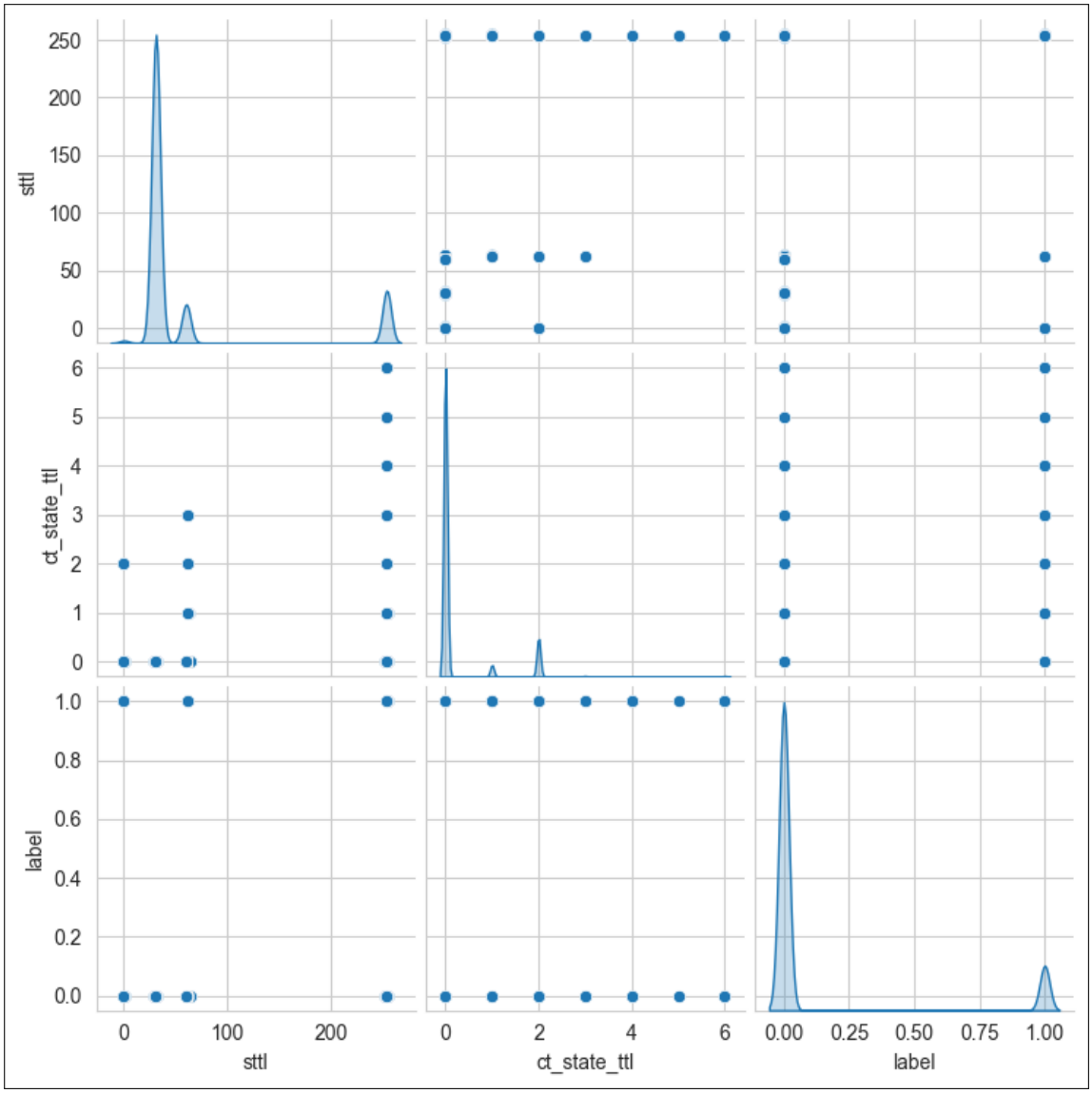

Figura B.6: Correlación entre columnas *sttl*, *ct_state_ttl* y *label*.

B.2.4. Variables swin y dwin

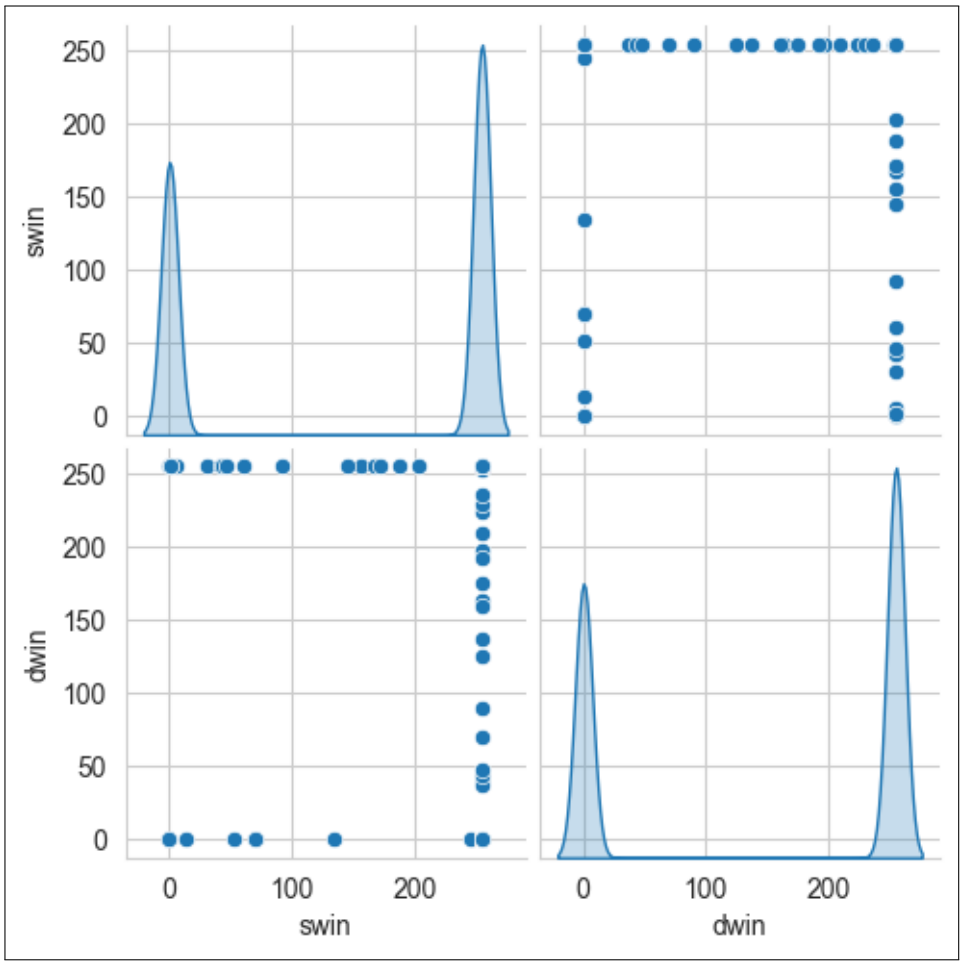

Figura B.7: Correlación entre columnas *swin* y *dwin*.

B.2.5. Variables stime y ltime

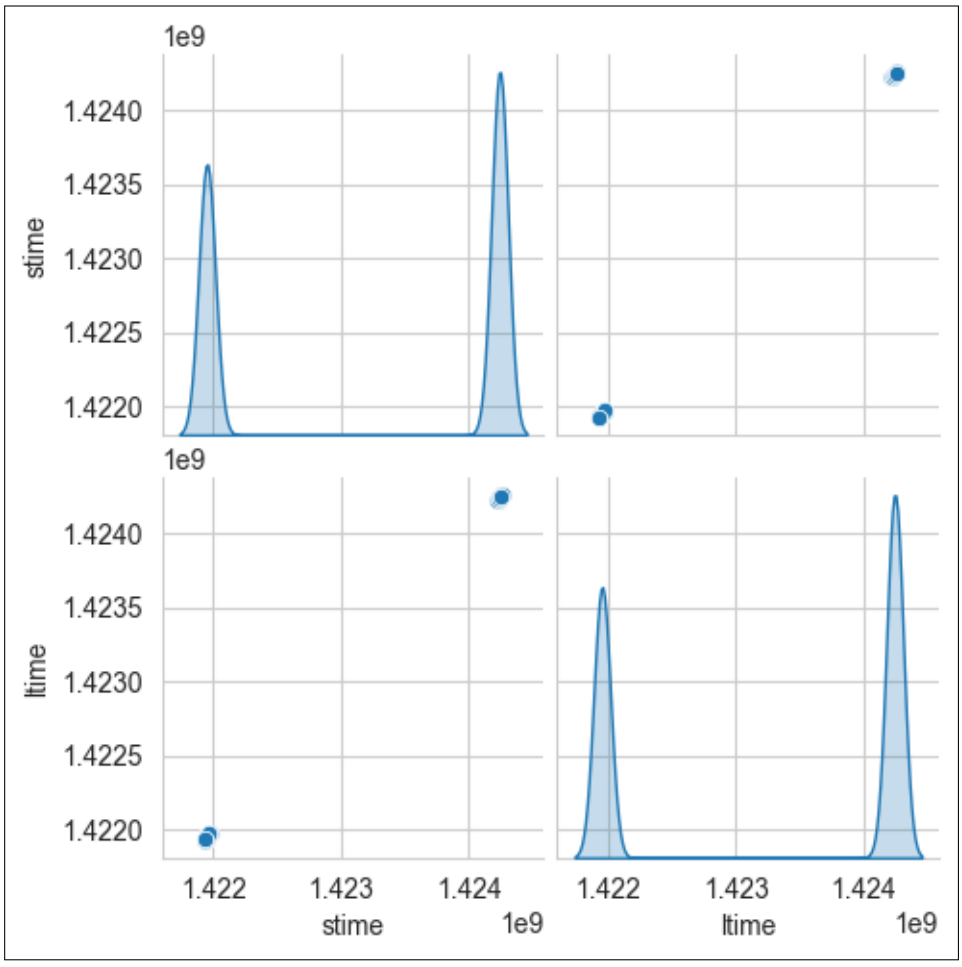

Figura B.8: Correlación entre columnas *stime* y *ltime*.

B.2.6. Variables del grupo TCP

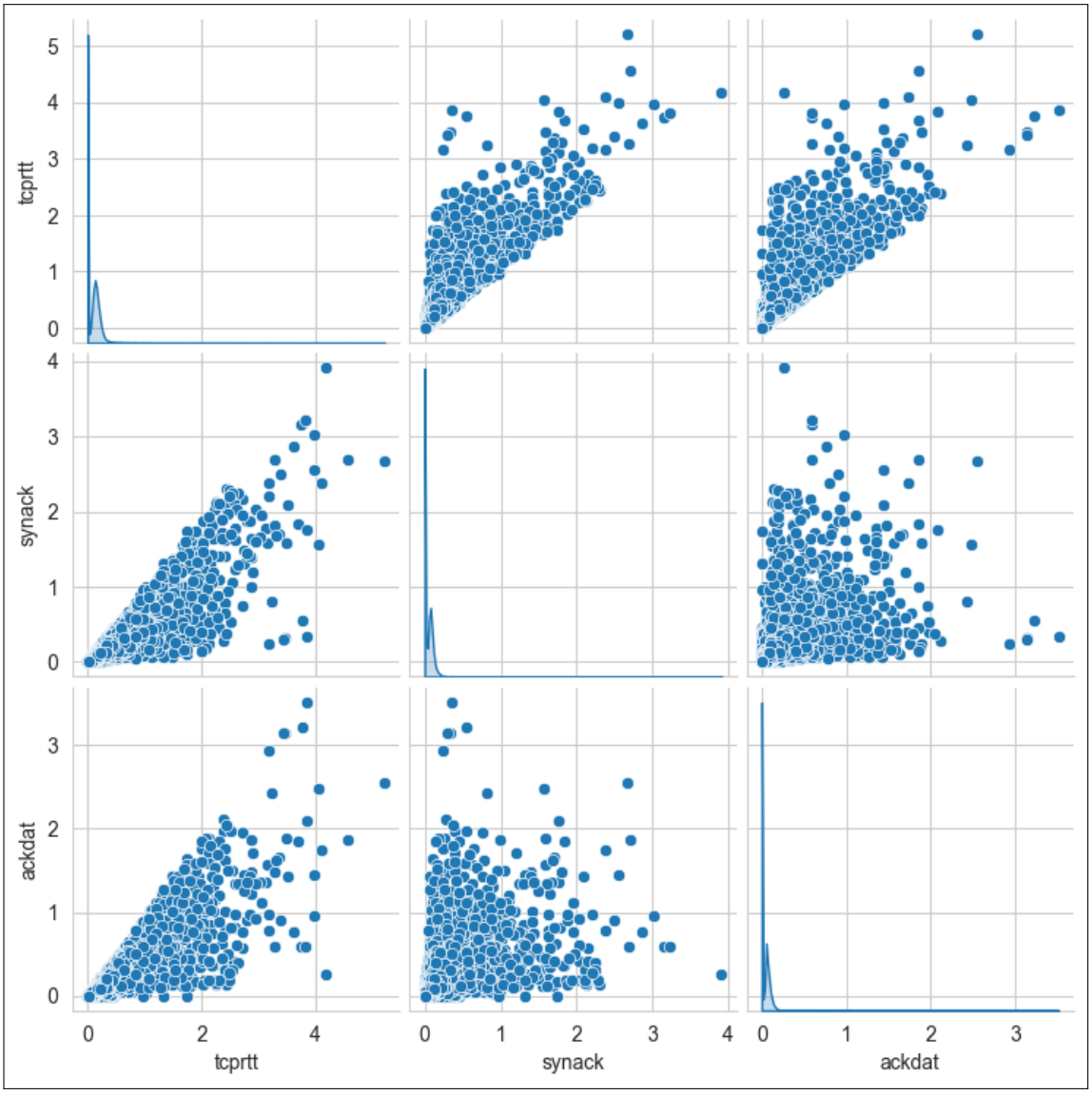

Figura B.9: Correlación entre columnas *tcprtt*, *synack* y *ackdat*

B.2.7. Variables del grupo srv

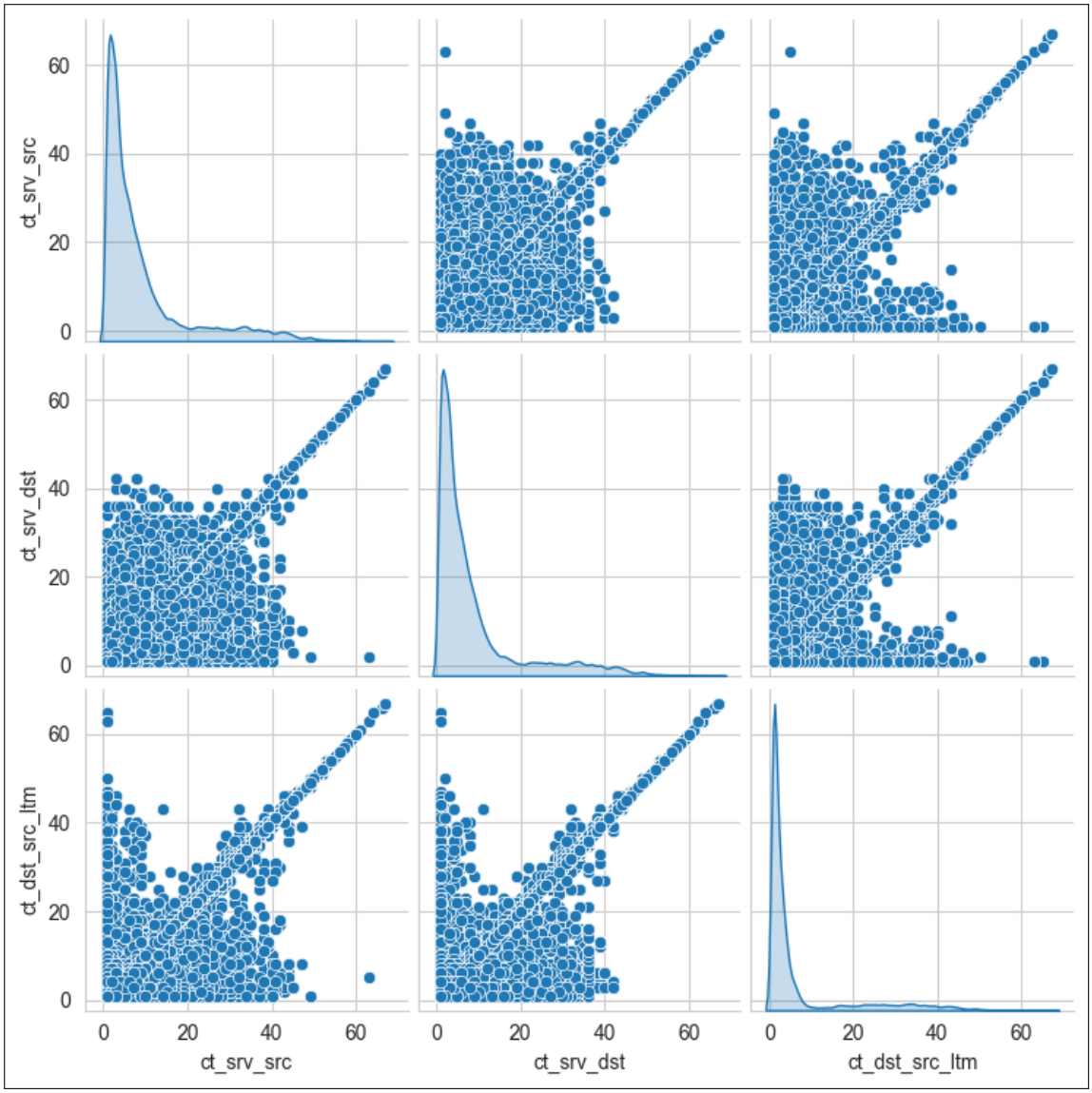

Figura B.10: Correlación entre columnas *ct_srv_src*, *ct_srv_dst* y *ct_dst_src_ltm*

B.2.8. Variables del grupo ltm

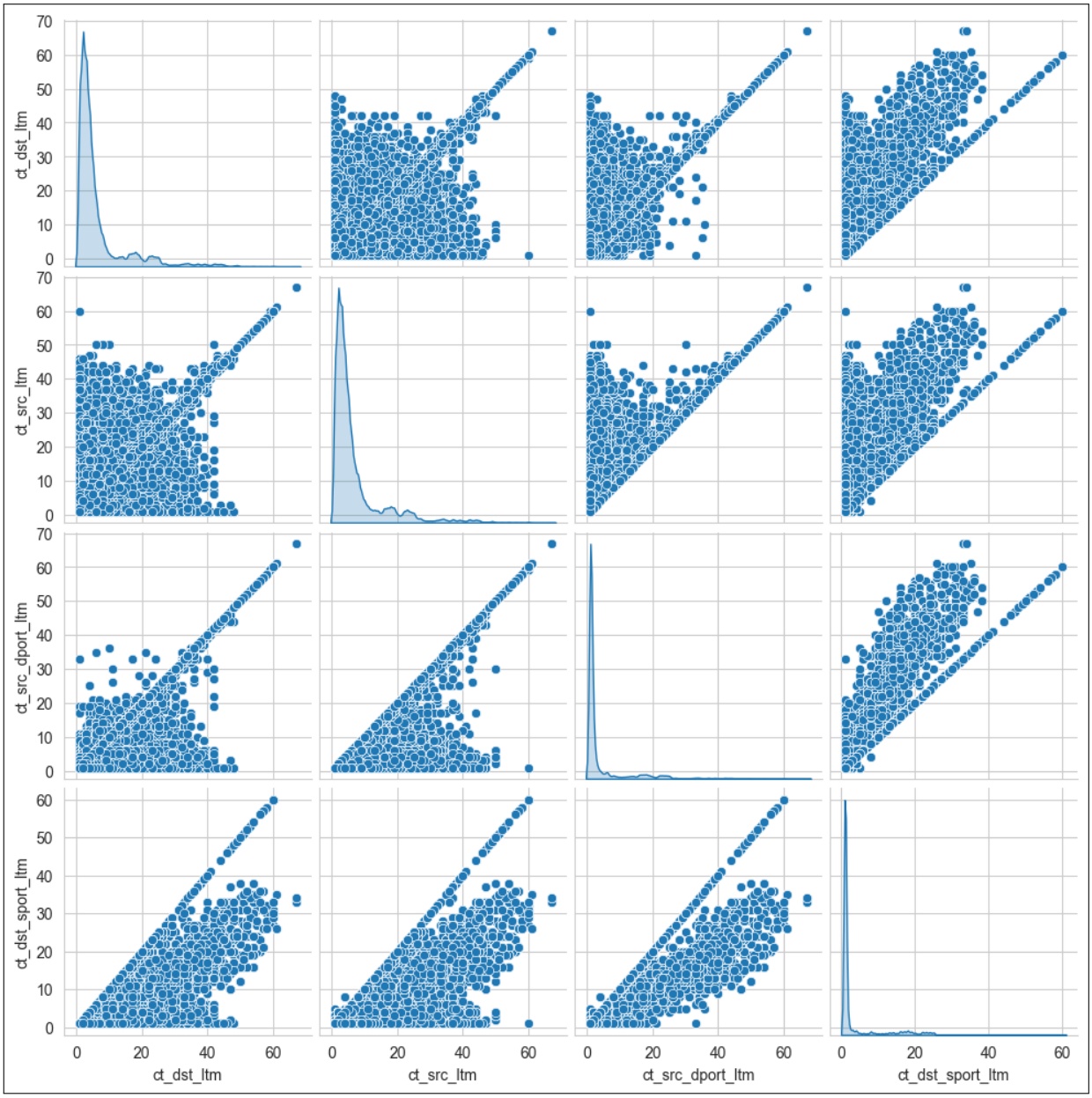

Figura B.11: Correlación entre columnas *ltm*.

B.3. Conjunto de datos CIC_IDS2017

B.3.1. Variables flow duration y fwd iat total

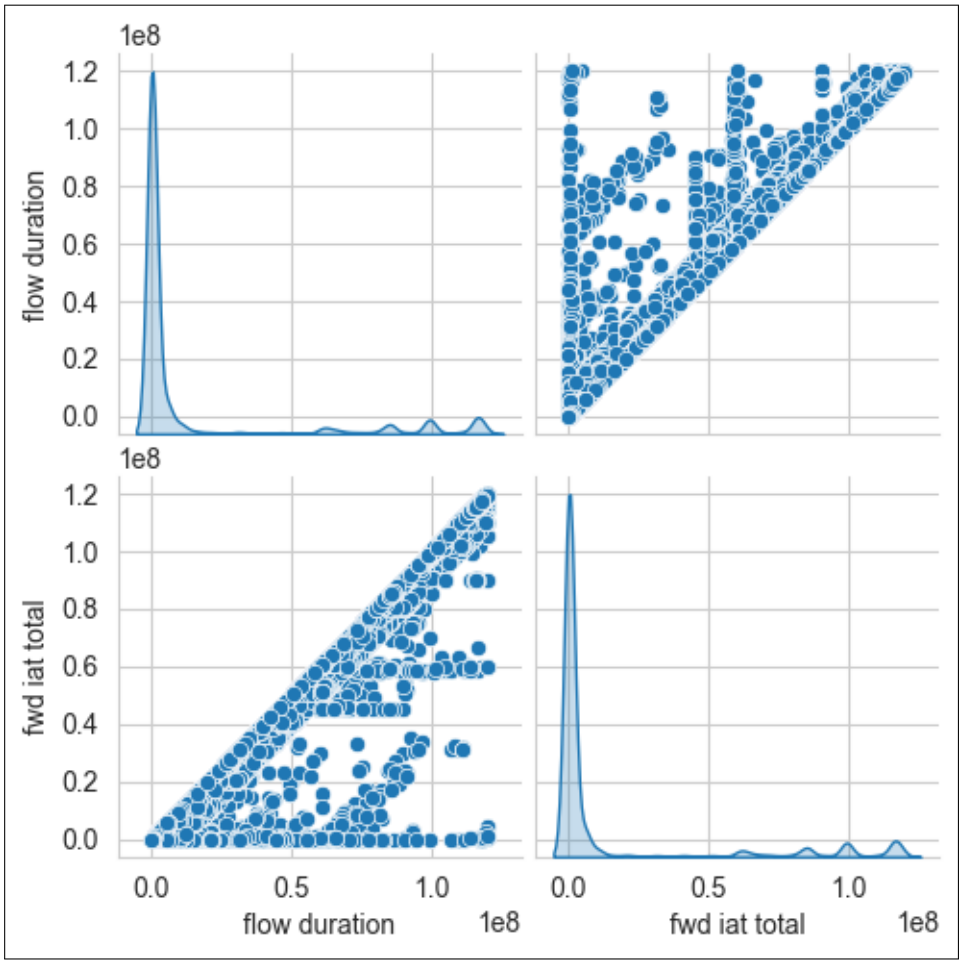

Figura B.12: Correlación entre columnas *flow duration* y *fwd iat total*.

B.3.2. Variables de grupo totalfwd y subflow

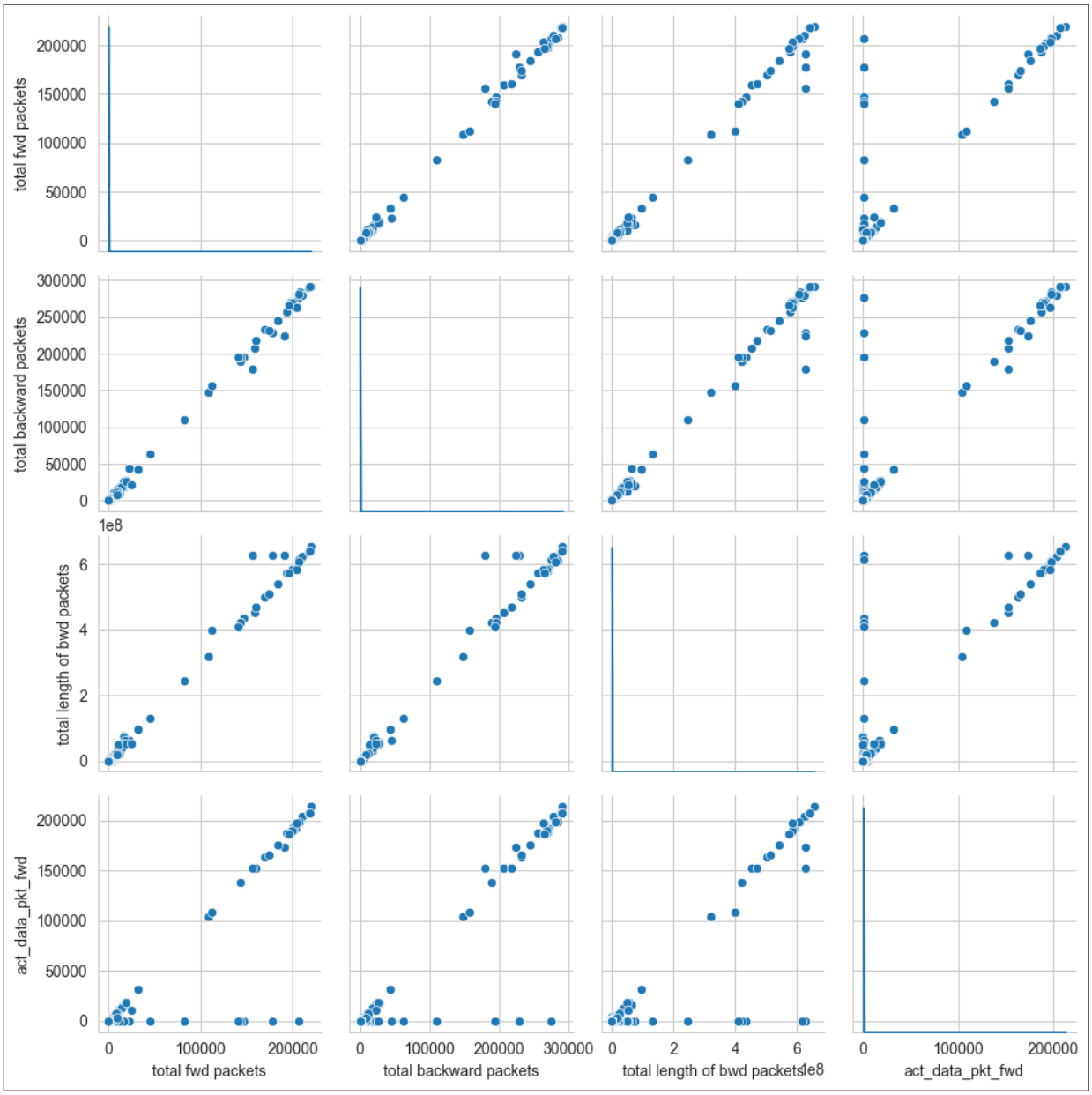

Figura B.13: Correlación entre columnas *total fwd packets*, *total backward packets*, *total length of bwd packets* y *act_data_pkt_fwd*.

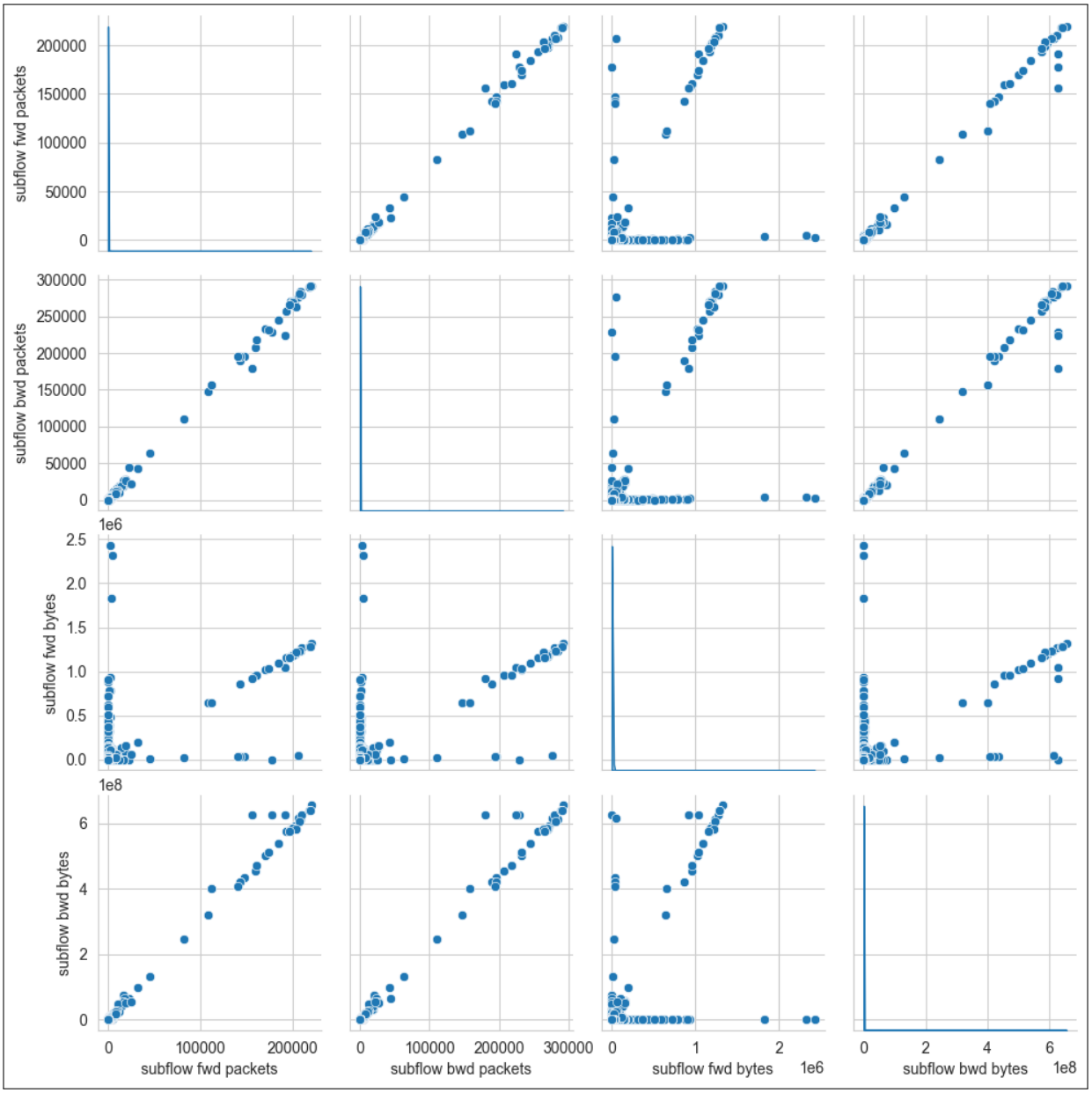

Figura B.14: Correlación entre columnas *subflow fwd packets*, *subflow bwd packets*, *subflow fwd bytes* y *subflow bwd bytes*.

B.3.3. Variables fwd packet length max y fwd packet length std

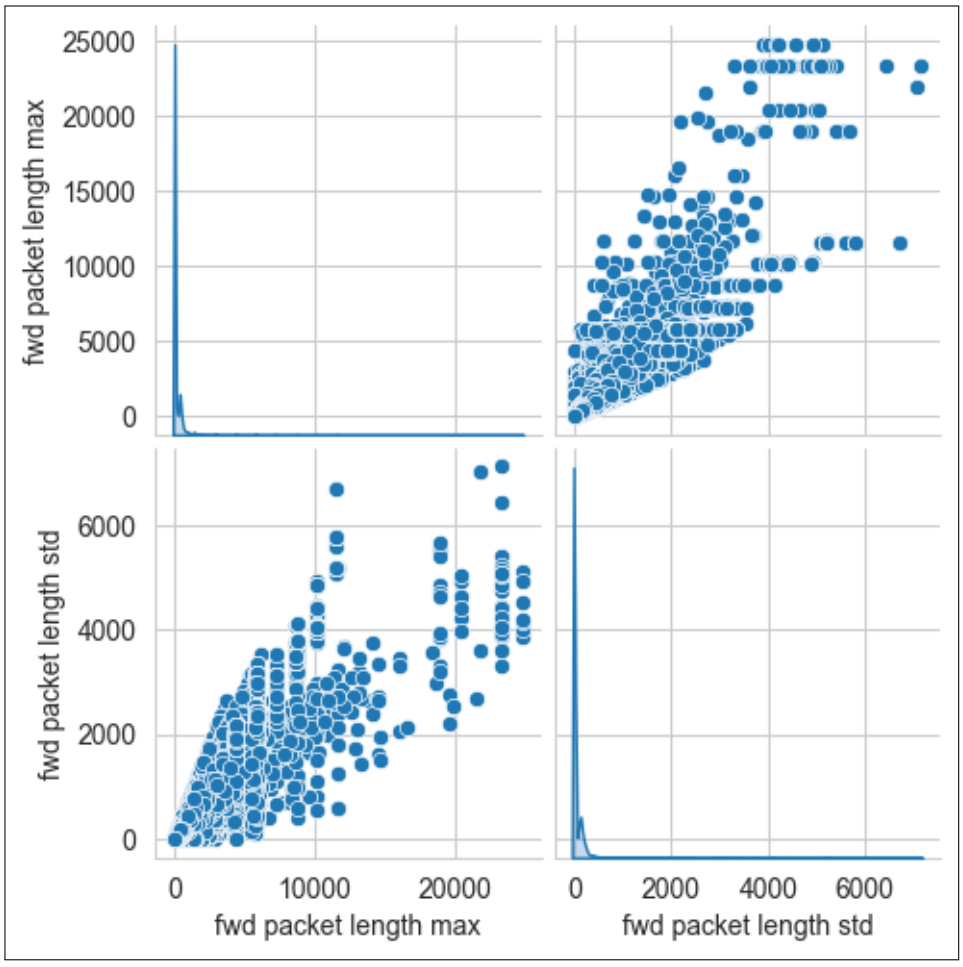

Figura B.15: Correlación entre columnas *fwd packet length max* y *fwd packet length std*.

B.3.4. Variables fwd size

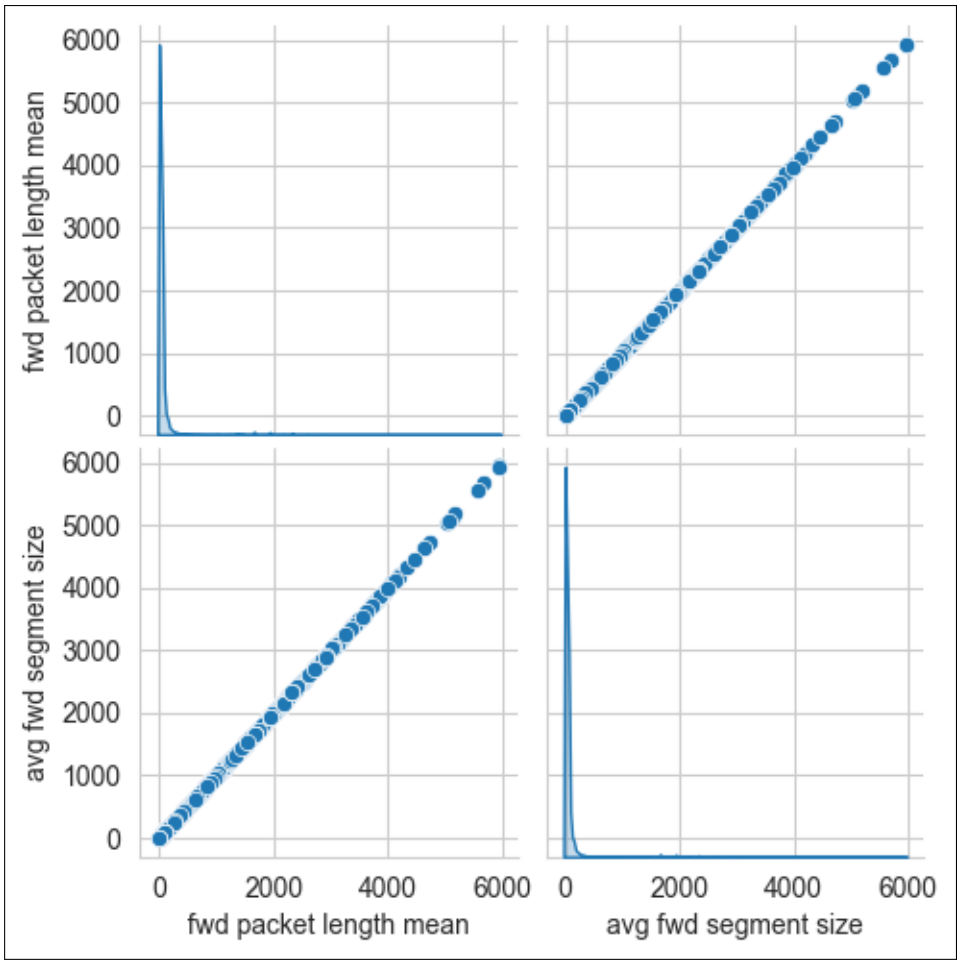

Figura B.16: Correlación entre columnas *fwd packet length mean* y *avg fwd segment size*.

B.3.5. Variables del grupo bwd packet length y packet length

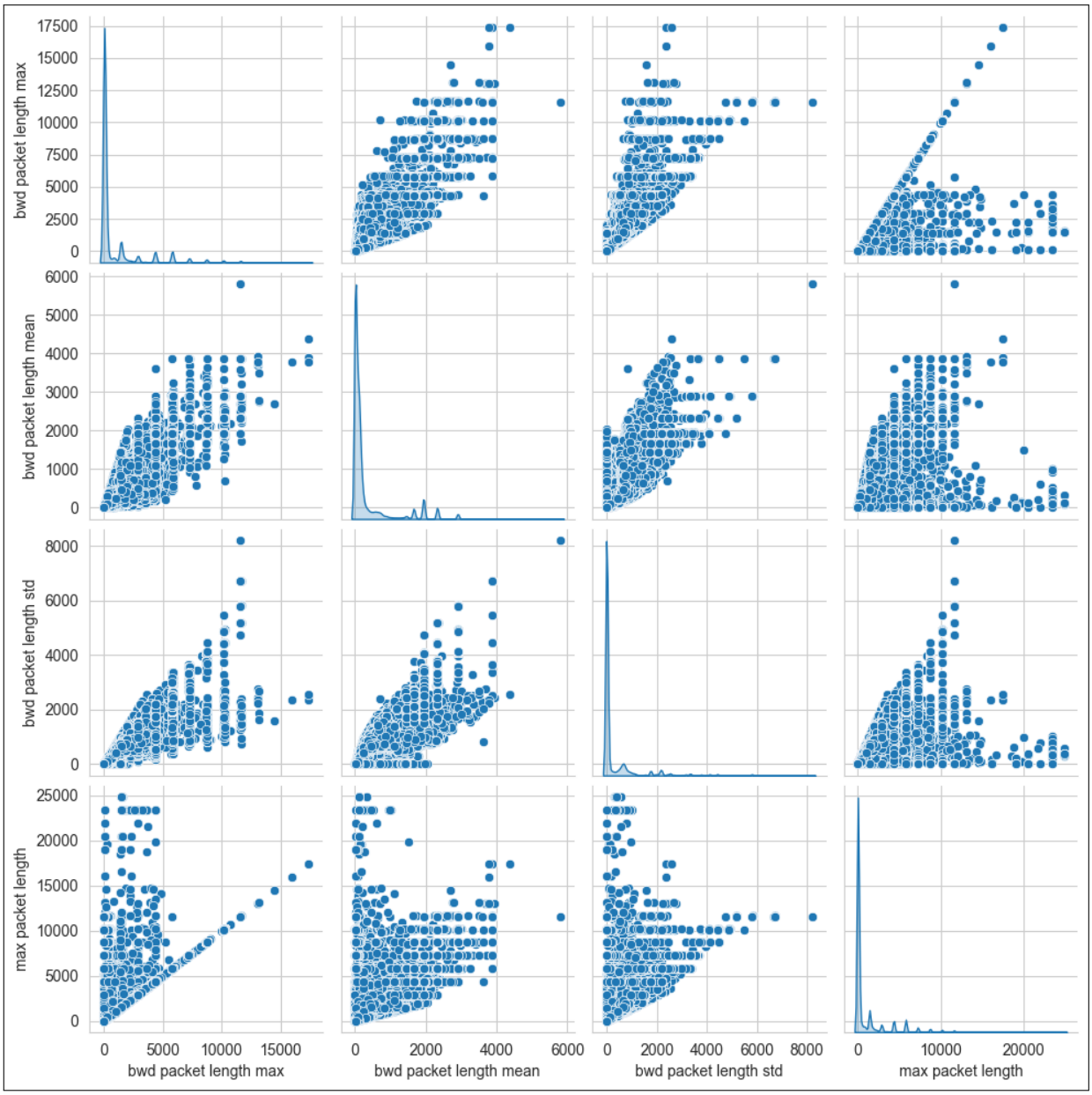

Figura B.17: Correlación entre columnas *bwd packet length*.

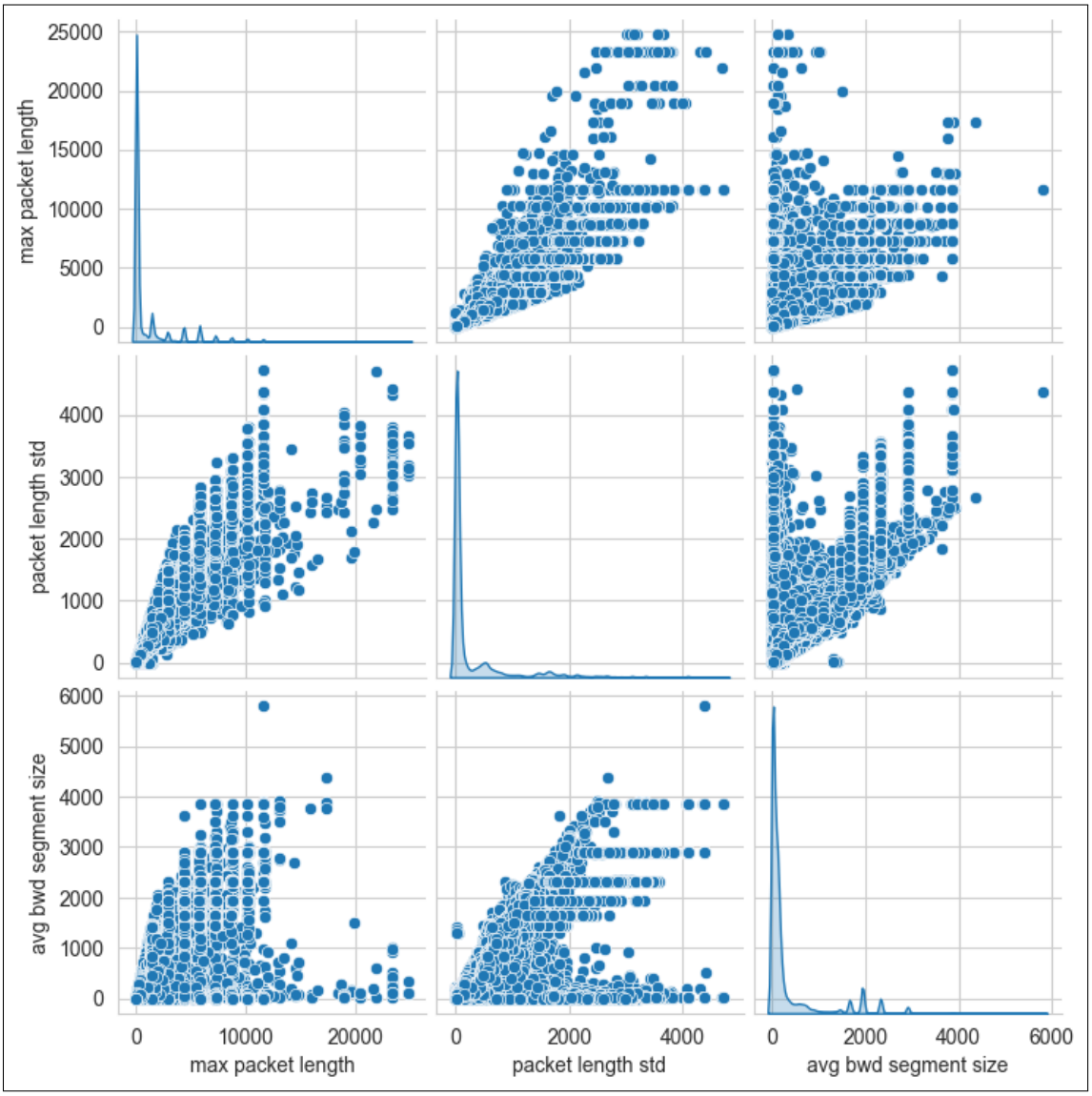

Figura B.18: Correlación entre columnas *max packet length*, *packet length std* y *avg bwd segment size*.

B.3.6. Variables de grupo packets/s

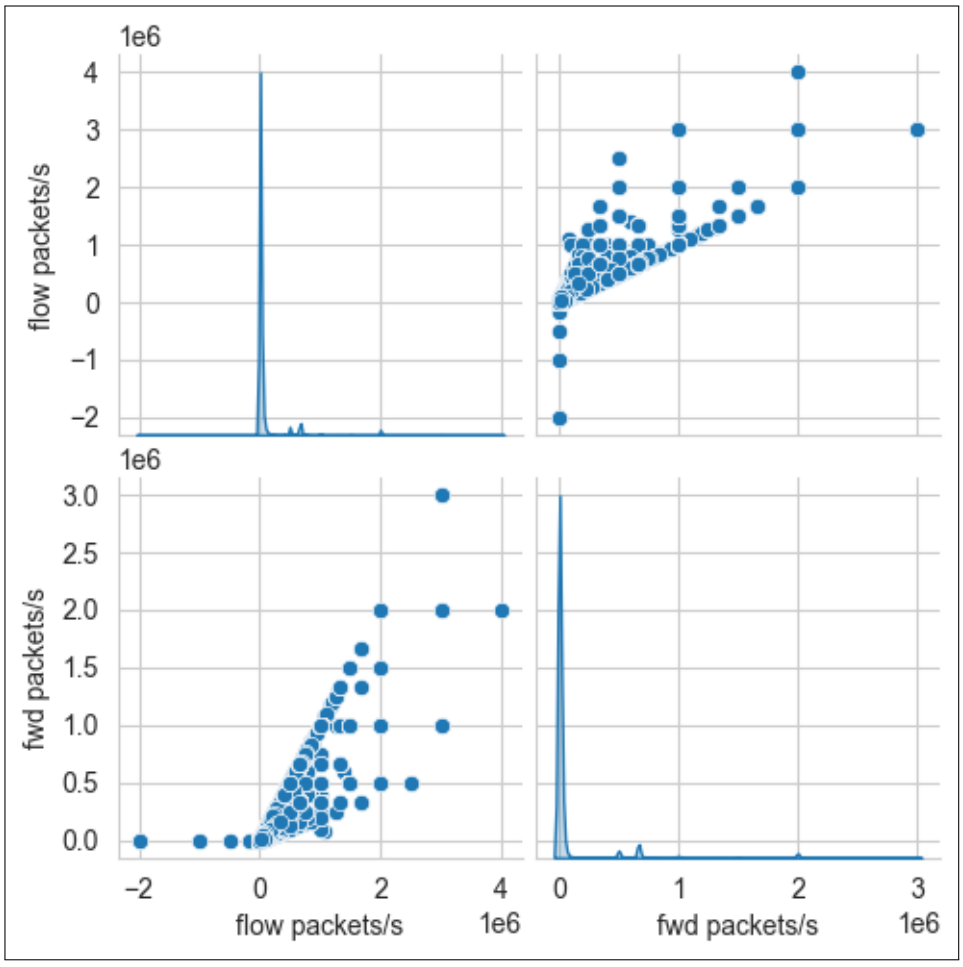

Figura B.19: Correlación entre columnas *flow packets/s* y *fwd packets/s*.

B.3.7. Variables de grupo iat mean

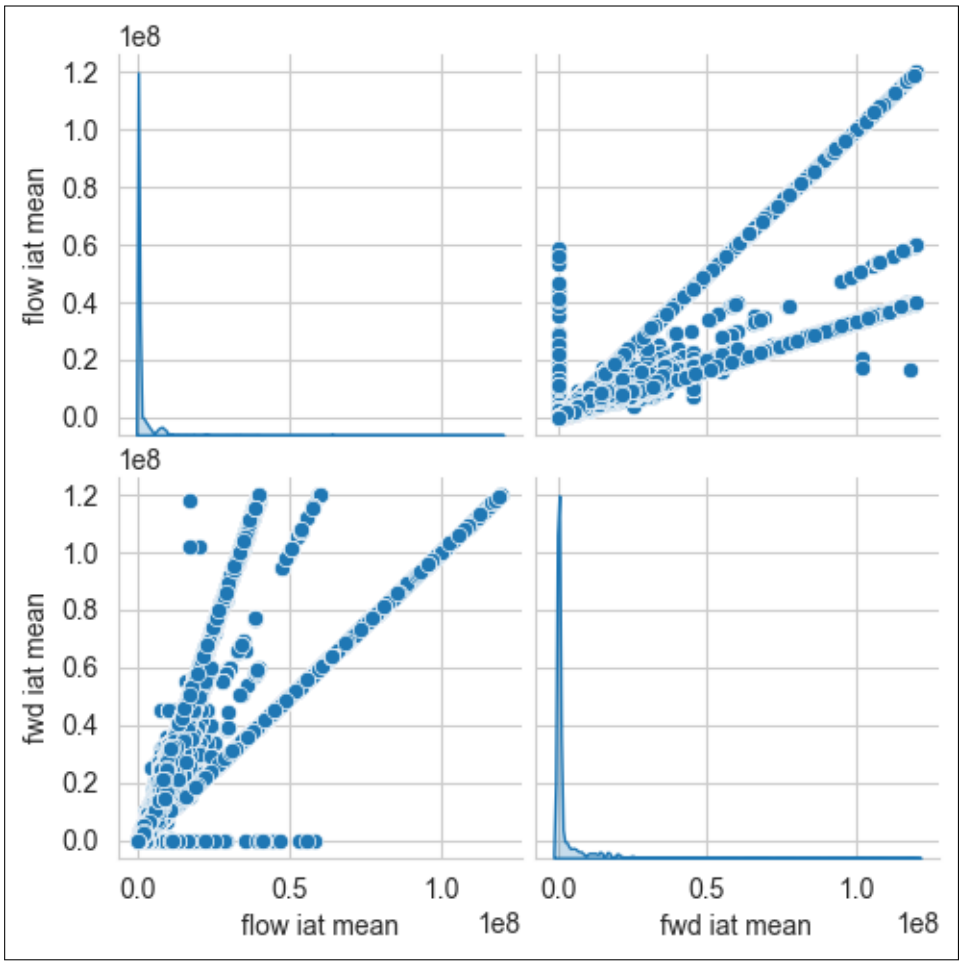

Figura B.20: Correlación entre columnas *flow iat mean* y *fwd iat mean*.

B.3.8. Variables de grupo iat e idle

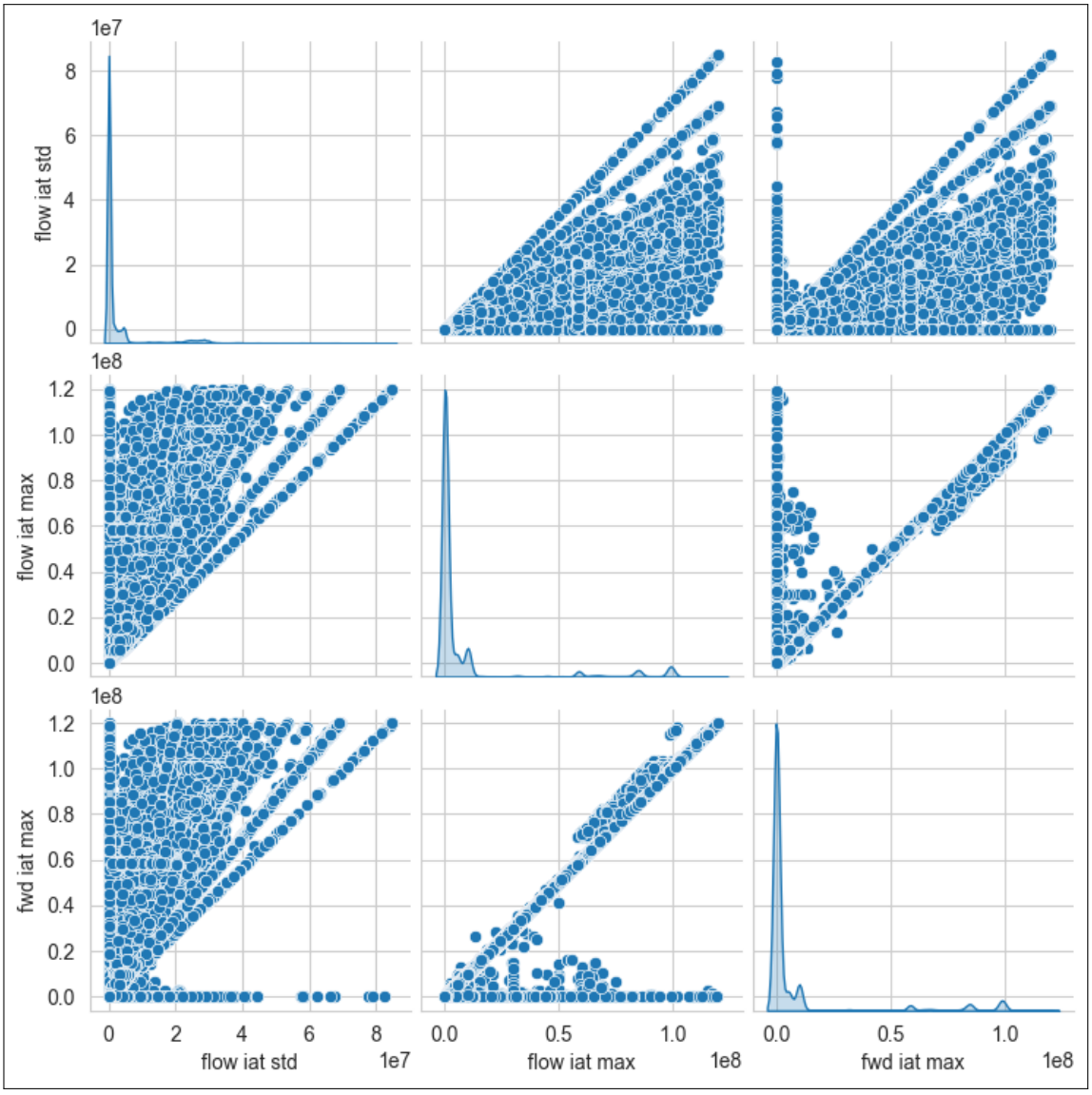

Figura B.21: Correlación entre columnas *iat*.

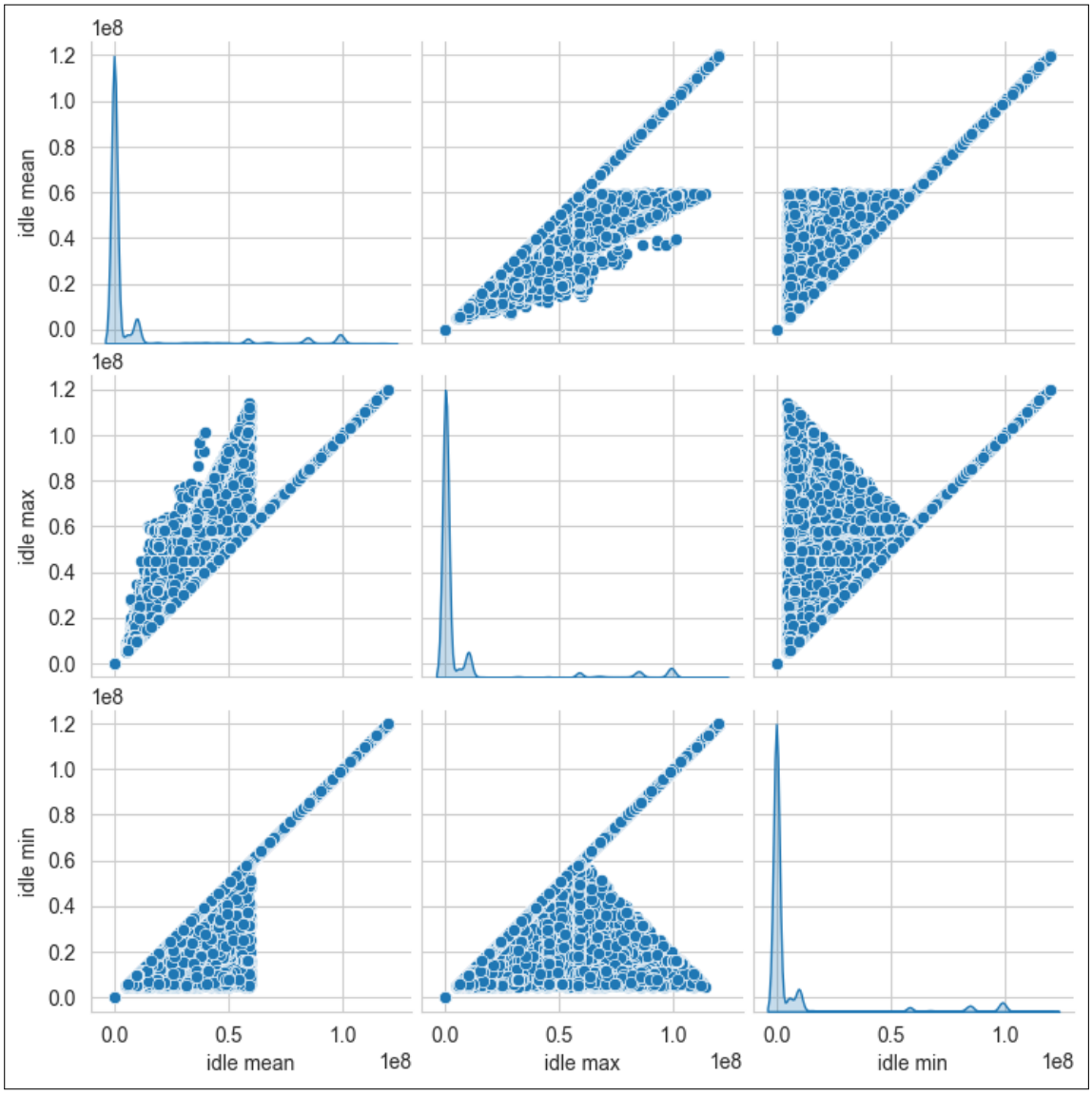

Figura B.22: Correlación entre columnas *idle*.

B.3.9. Variables de grupo bwd iat

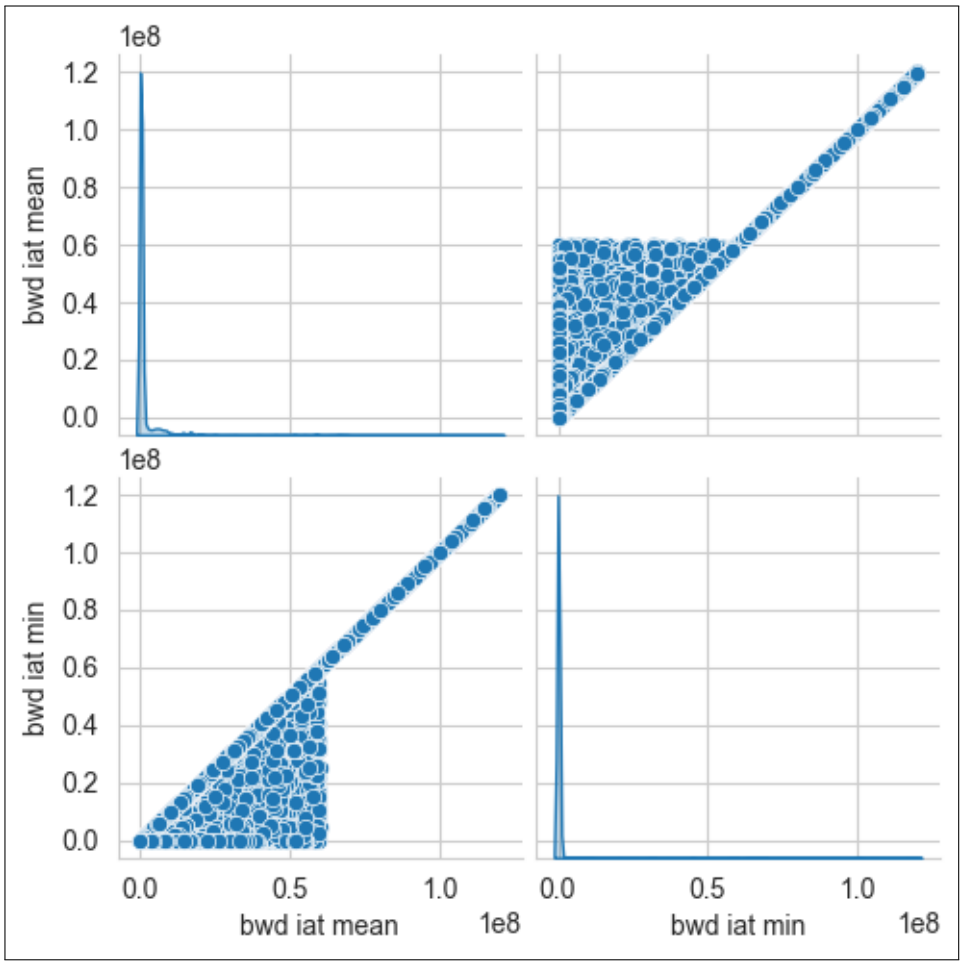

Figura B.23: Correlación entre columnas *bwd iat*.

B.3.10. Variables de flag PSH y SYN

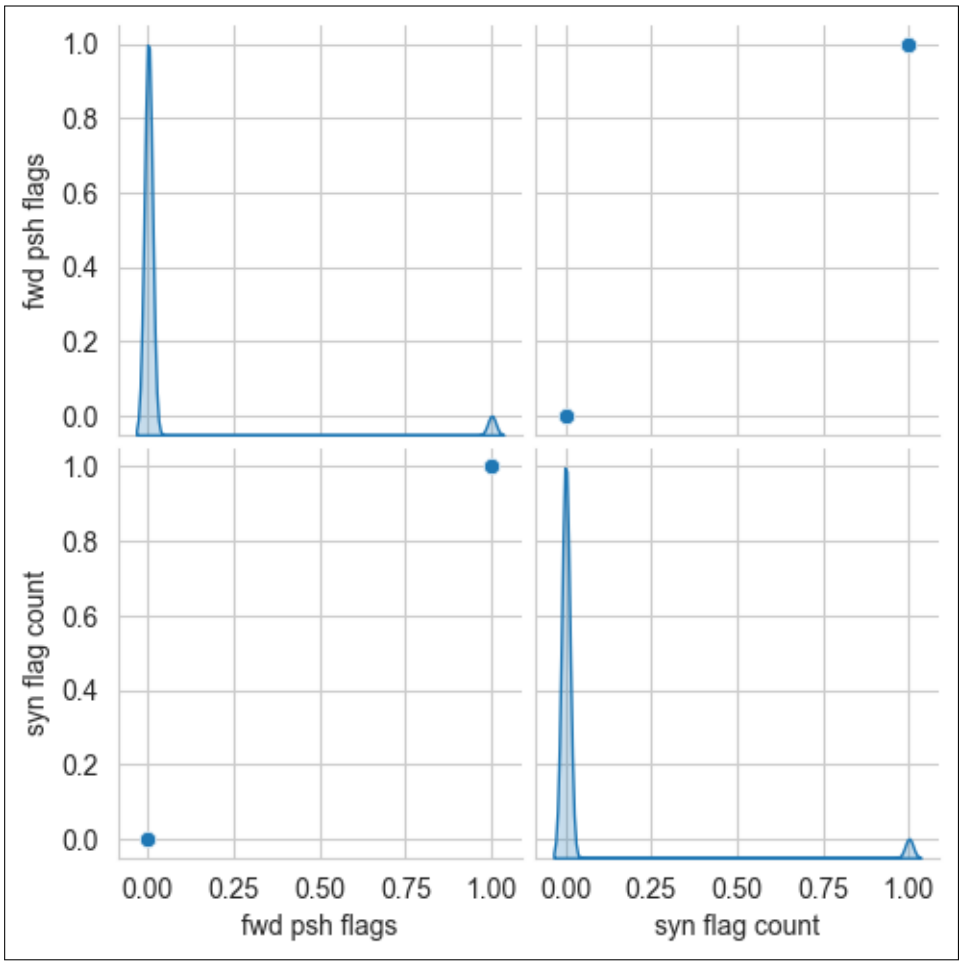

Figura B.24: Correlación entre columnas *fwd psh flags* y *syn flag count*.

B.3.11. Variables de flag URG y CWE

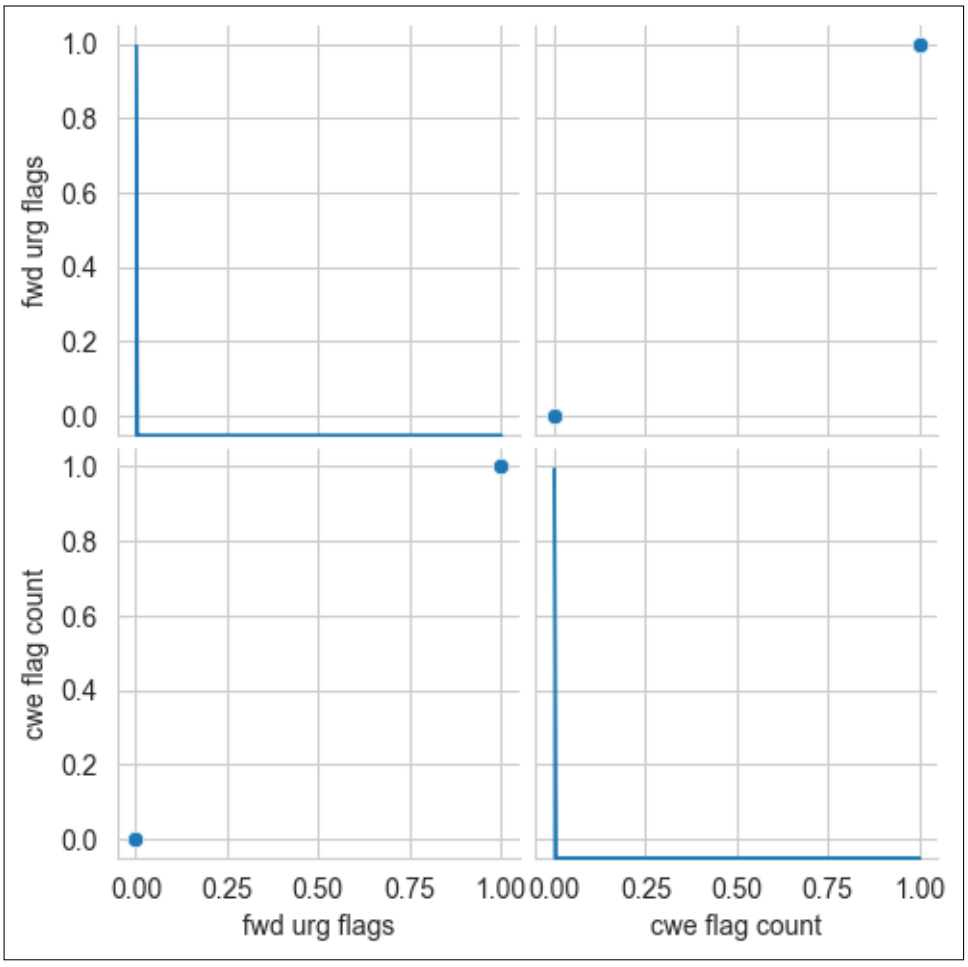

Figura B.25: Correlación entre columnas *fwd urg flags* y *cwe flag count*.

B.3.12. Variables header length

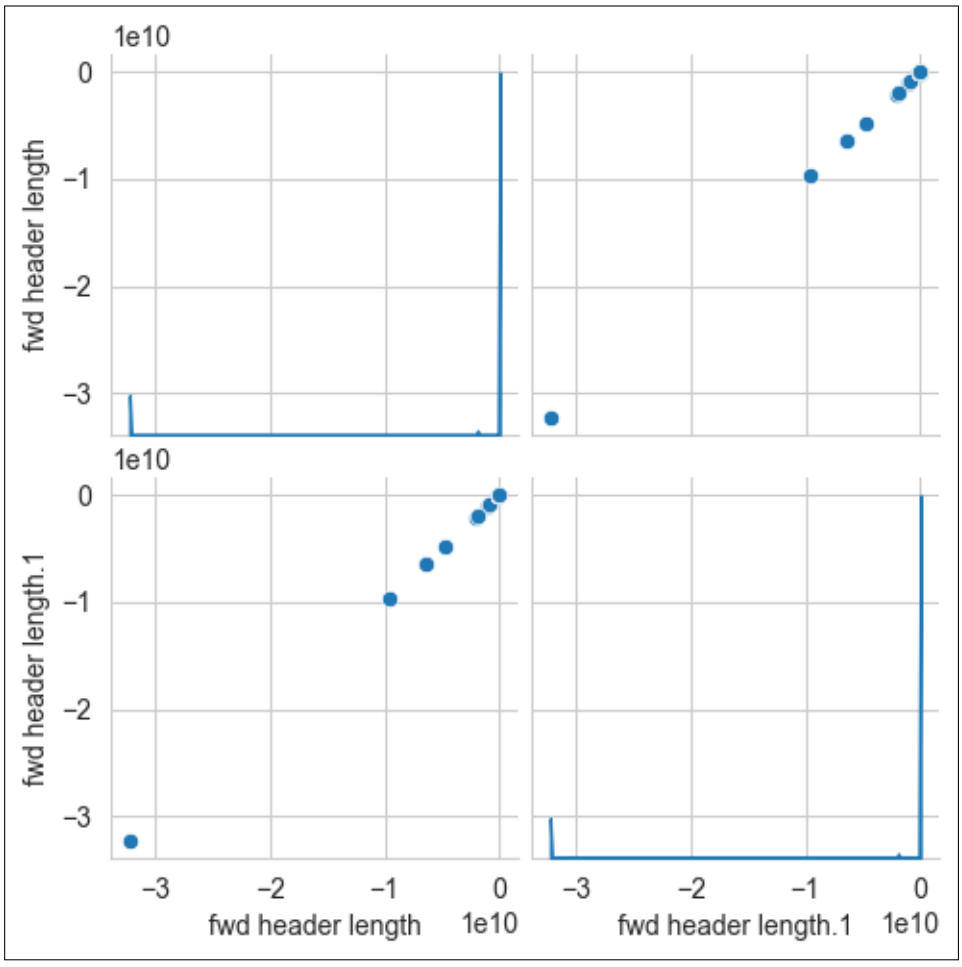

Figura B.26: Correlación entre columnas repetidas *header length*.

B.3.13. Variables packet length y packet

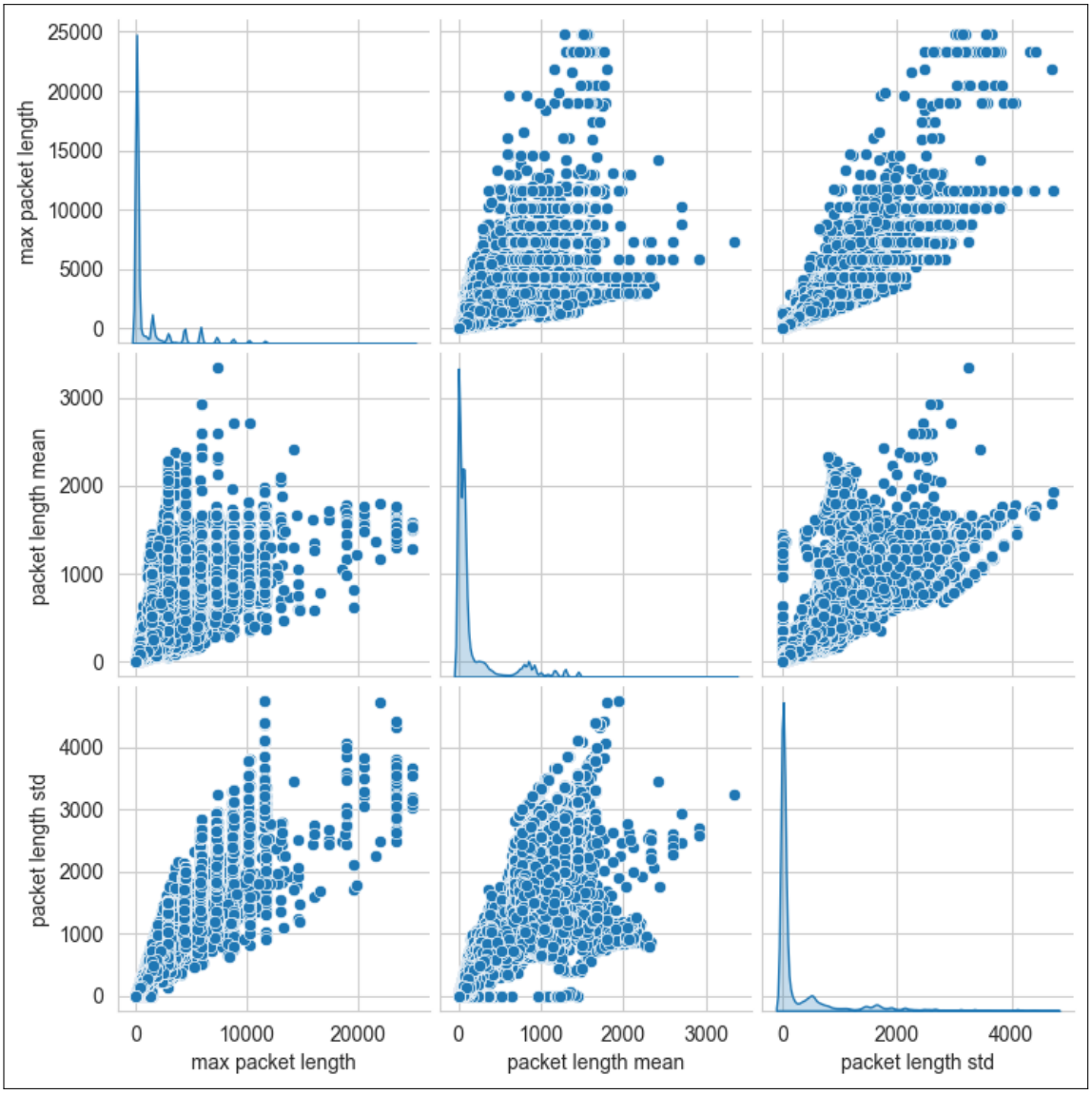

Figura B.27: Correlación entre columnas *packet length*.

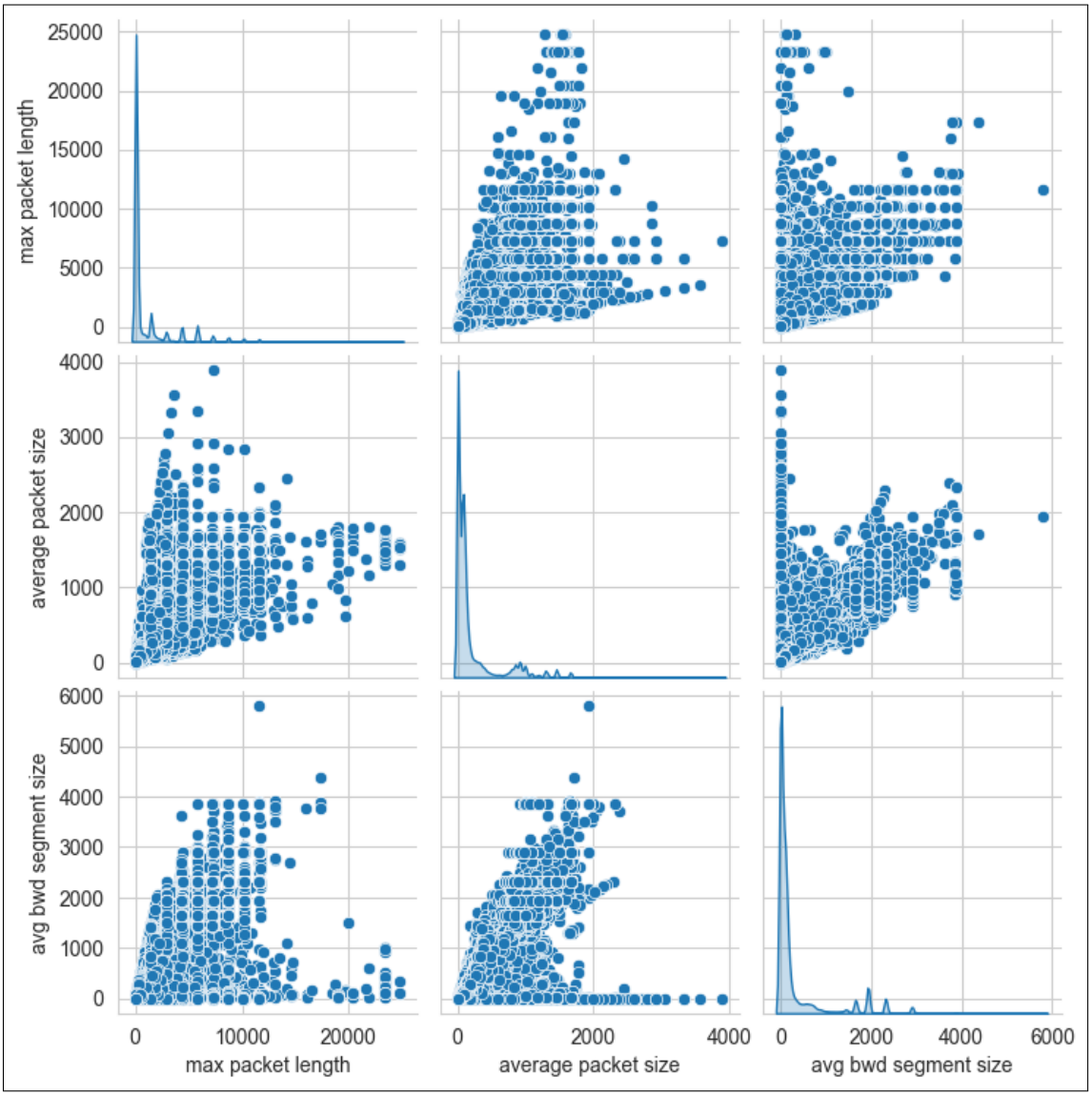

Figura B.28: Correlación entre columnas *max packet length*, *average packet size* y *avg bwd segment size*.

B.3.14. Variables flag RST y ECE

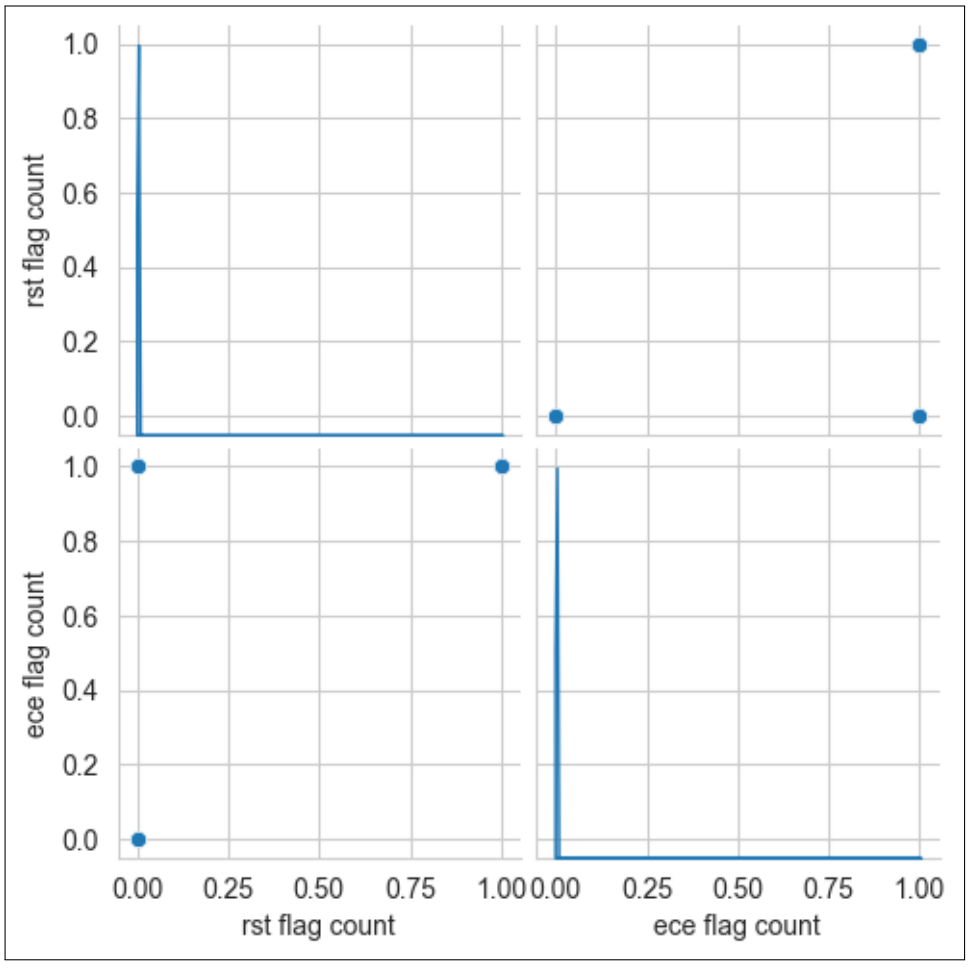

Figura B.29: Correlación entre columnas *RST flag count* y *ECE flag count*.

B.3.15. Variables de grupo active

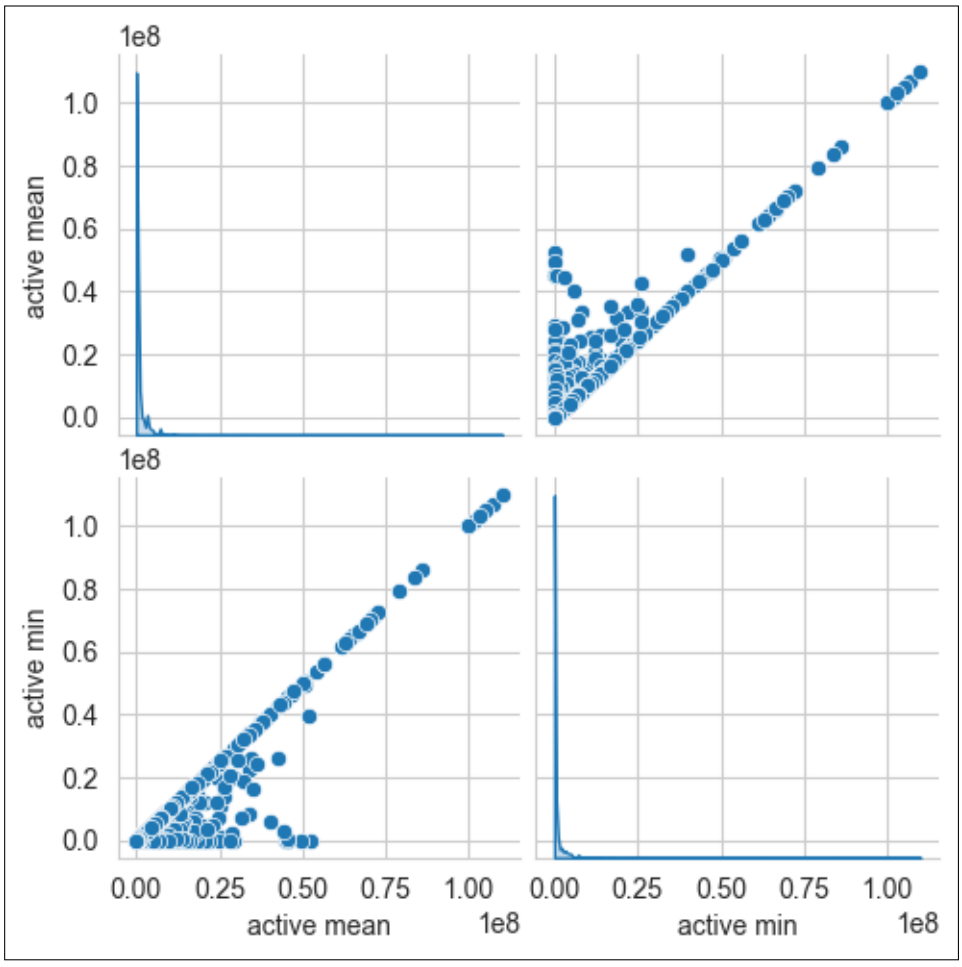

Figura B.30: Correlación entre columnas *active mean* y *active min*.

Apéndice C

Listado de ataques

C.1. Conjunto de datos NSL_KDD

C.1.1. Conjunto de entrenamiento (train)

- En la sección de **DoS**: Neptune (41214), Smurf (2646), Back (956), Teardrop (892), Pod (201), Land (18).
- En la sección de **Probe**: Satan (3633), IPSweep (3599), PortSweep (2931), Nmap (1493).
- En la sección de **R2L**: Warezclient (890), GuessPassword (53), Warezmaster (20), IMAP (11), FtpWrite (8), Multihop (7), Phf (4), Spy (2).
- En la sección de **U2R**: BufferOverflow (30), Rootkit (10), LoadModule (9), Perl (3).

C.1.2. Conjunto de prueba (test)

- En la sección de **DoS**: Neptune (4657), Apache2 (737), ProcessTable (685), Smurf (665), Back (359), Pod (41), Teardrop (12), Land (7), UDPStorm (2), Worm (2).
- En la sección de **Probe**: MScan (996), Satan (735), Saint (319), PortSweep (157), IPSweep (141), Nmap (73).
- En la sección de **R2L**: GuessPassword (1231), Warezmaster (944), SNMPGuess (331), Mailbomb (293), SNMPgetattack (178), HTTPTunnel (133), Multihop (18), Named (17), Sendmail (14), Xlock (9), Xsnoop (4), Ftp write (3) Phf (2), IMAP (1).
- En la sección de **U2R**: Buffer overflow (20), Ps (15), Rootkit (13), Xterm (13), LoadModule (2), Perl (2), SQLAttack (2).

C.2. Conjunto de datos UNSW_NB15

- En la sección de **DoS**: DoS (16353).
- En la sección de **Rastreo**: Generic (215481), Reconnaissance (13987), Analysis (2677)
- En la sección de **Vulnerabilidades**: Exploits (44525), Fuzzers (24246), Backdoors (2329), Shellcode (1511) y Worms (174).

C.3. Conjunto de datos CIC_IDS2017

- En la sección de **DoS**: HULK (230124), DDoS (128025), GoldenEye (10293), SlowLoris (5796), SlowHTTPTest (5499), Bot (1956).
- En la sección de **Rastreo**: PortScan (158804).
- En la sección de **Fuerza Bruta**: FTP-Patator (7935), SSH-Patator (5897).
- En la sección de **Ataques Web**: WebAttack-BruteForce (1507), WebAttack-XSS (652), WebAttack-SQLInjection (21).
- En la sección de **Vulnerabilidades**: Infiltration (36), Heartbleed (11).

Apéndice D

Gráficas de características importantes

D.1. Conjunto de datos NSL_KDD

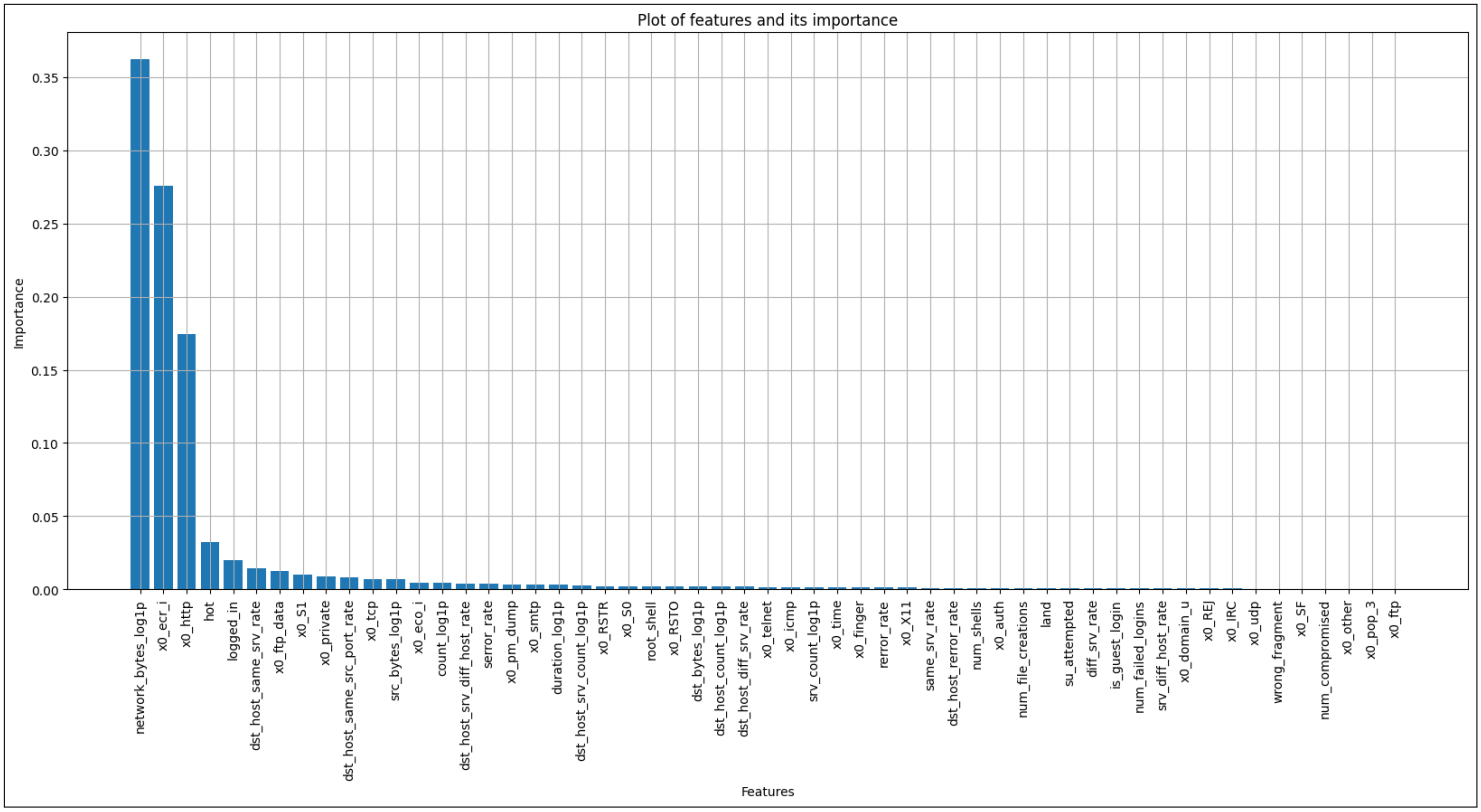

Figura D.1: Características importantes en el conjunto de datos NSL_KDD.

D.2. Conjunto de datos UNSW_NB15

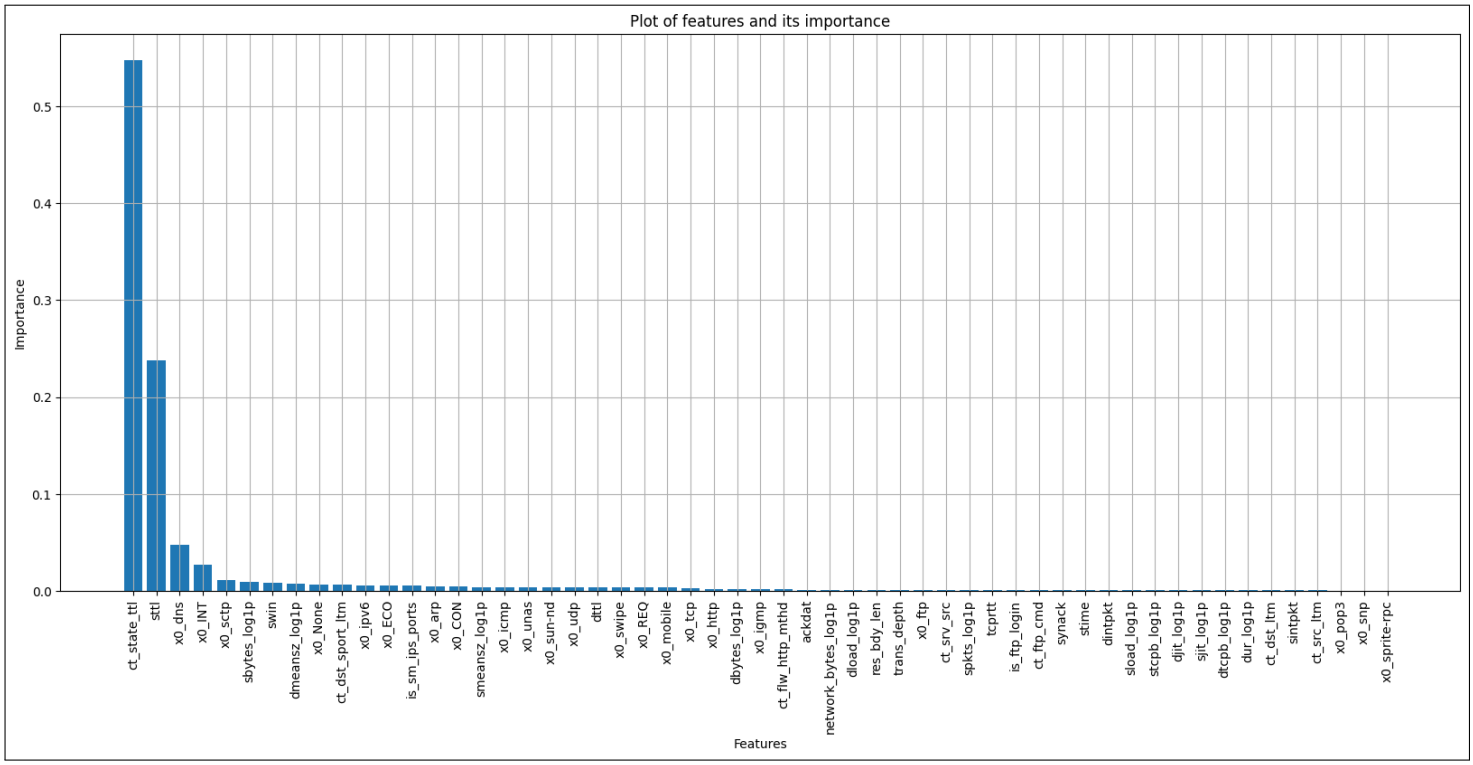

Figura D.2: Características importantes en el conjunto de datos UNSW_NB15.

D.3. Conjunto de datos CIC_IDS2017

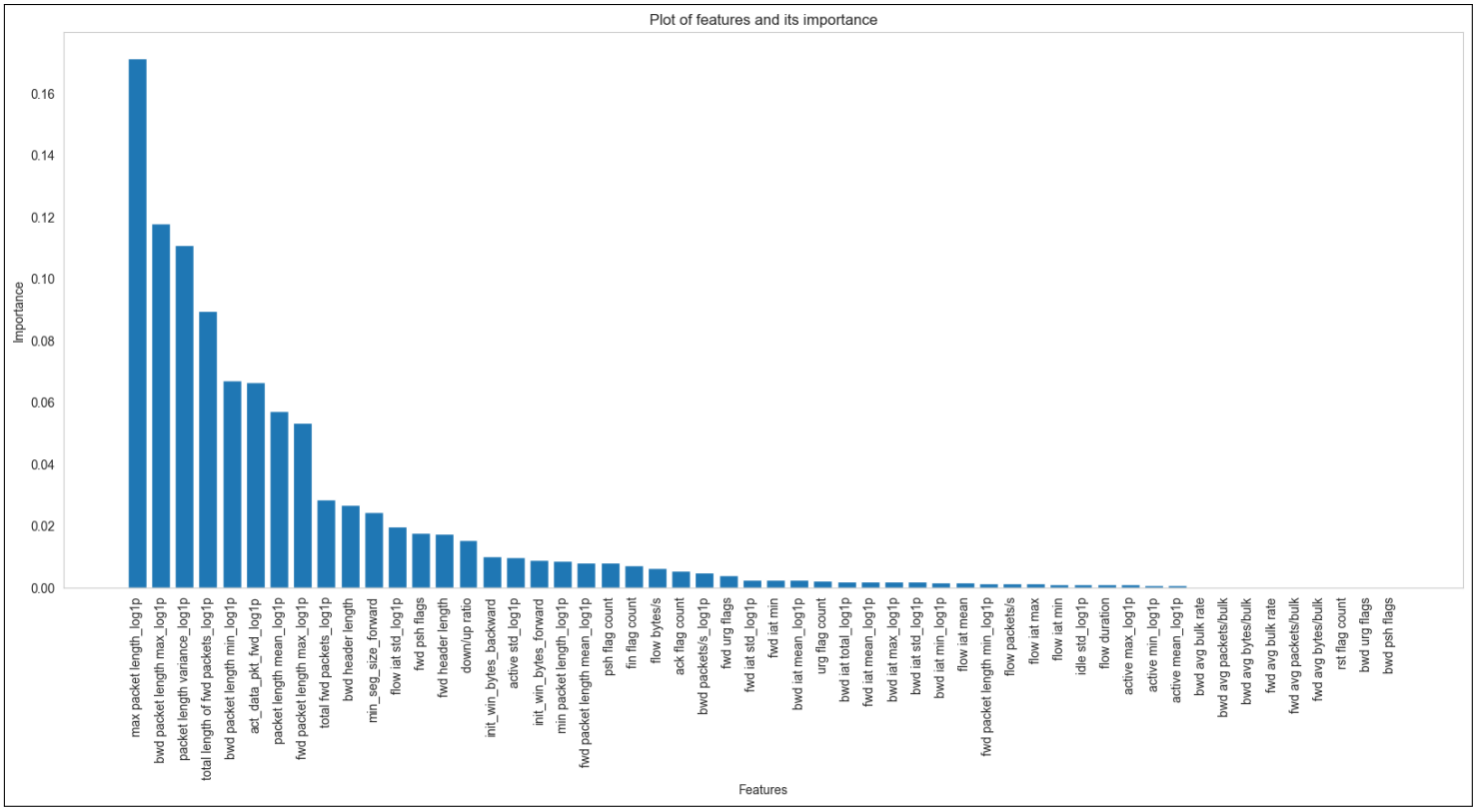

Figura D.3: Características importantes en el conjunto de datos CIC_IDS2017.

Bibliografía

[1] “Tecnologías de la Industria 4.0: ¿Qué son y cómo funcionan?” Sydle, 2022

[2] “¿Qué es la Industria 4.0 y cómo funciona?” IBM, 2023.

[3] Luis Ochoa. “Cómo prevenir ciberataques en redes de control industrial” Grupo CMC, 2017.

[4] Andrew Ginter. “The Top 20 Cyberattacks on Industrial Control Systems.” Waterfall, 2017.

[5] “¿Qué son los ataques DoS y DDoS?” OSI, 2018.

[6] Martin Roesch, Cisco Systems. (1998). Snort [Software].

[7] Vern Paxson. (1998). Zeek [Software].

[8] Open Information Security Foundation. (2009). Suricata [Software].

[9] “¿Qué es la inteligencia artificial (IA)?” Oracle, 2020.

[10] UNB: NSL-KDD dataset. University of New Brunswick (2009)

[11] L. Dhanabal and S. P. Shantharajah. "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms." (2015): 2-4

[12] Moustafa, Nour, and Jill Slay. "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, 2015. 1-6

[13] Yasir Hamid, Balasaraswathi Ranganathan, Ludovic Journaux and Muthukumarasamy Suguraman. “Benchmark Datasets for Network Intrusion Detection: A Review.” Research Gate, 2018.

[14] “Intrusion Detection Evaluation Dataset (CIC-IDS2017)” University of New Brunswick (2017)

[15] Maxime Lanvin, Pierre-François Gimenez, Yufei Han, Frédéric Majorczyk, Ludovic Mé, et al.. “Errors in the CICIDS2017 dataset and the significant differences in detection performances it makes”. pp.1-16, 2023.

[16] Juan Ignacio Bagnato. “Algoritmo k-Nearest Neighbor.” Aprende Machine Learning, 2018.

[17] Ligdi Gonzalez. “Regresión Logística – Teoría.” Aprende IA, 2019.

[18] Brayan Buitrago. “Regresión Logística I — Machine Learning.” Medium, 2020

[19] Jose Martinez Heras. “Máquinas de Vectores de Soporte” IArtificial, 2019.

[20] Joaquín Amat Rodrigo. "Máquinas de Vector Soporte (Support Vector Machines, SVMs)." Ciencia de Datos, 2017.

[21] Alexandre Kowalczyk. "Support Vector Machines Succinctly." Syncfusion, 2017

[22] “Árbol de decisión en Machine Learning (Parte 1)” Sitiobigdata, 2019

[23] “Árboles de decisión: qué son y cuál es su uso en Big Data” UNIR, 2021.

[24] “¿Qué es un bosque aleatorio?” TIBCO Software, 2022.

[25] Hyacinth Ampadu. "Random Forests Understanding." AI Pool, 2021.

[26] “Árboles de decisión con boosting de gradiente” Google Developers, 2022.

[27] Cheng Li. "A Gentle Introduction to Gradient Boosting." College of Computer and Information Science, 2016.

[28] “Aprendizaje no Supervisado.” AprendeIA, 2020.

[29] [Keepler.Io] “Isolation Forest: el algoritmo estrella para detección de anomalías” Medium, 2019.

[30] Joaquín Amat Rodrigo. “Detección de anomalías: Isolation Forest.” Ciencia de Datos, 2020.

[31] Francisco Sanz. “K-Means Clustering: algoritmo, aplicaciones y desventajas.” The Machine Learners, 2020

[32] S. García, J. Luengo, F. Herrera. Data Preprocessing in Data Mining. Berlin, Germany: Springer, 2015.

[33] Jose Mariano Alvarez. “Categorías y la codificación One-Hot.” 2018.

[34] Nagesh Singh Chauhan, “Métricas De Evaluación De Modelos En El Aprendizaje Automático” DataSourceAI, 2020.

[35] Jose Martinez Heras. “Precision, Recall, F1, Accuracy en clasificación.” IArtificial, 2020.

[36] Pol Martí Sanahuja, “Entendiendo la curva ROC y el AUC: Dos medidas del rendimiento de un clasificador binario que van de la mano.” 2021.

[37] “Performance Measures: Cohen’s Kappa statistic” The Data Scientist, 2021.

[38] "Metrics: Matthew's correlation coefficient" The Data Scientist, 2022.

[39] Quang Hung Nguyen, Hai-Bang Ly, Lanh Si Ho, Nadhir Al-Ansari, Hiep Van Le, Van Quan Tran, Indra Prakash, Binh Thai Pham. "Influence of Data Splitting on Performance of Machine Learning Models in Prediction of Shear Strength of Soil." Mathematical Problems in Engineering, vol. 2021, 2021.

[40] Juan Ignacio Bagnato. "Qué es overfitting y underfitting y cómo solucionarlo." Aprende Machine Learning, 2018.

[41] Daniel Nelson. "What is Ensemble Learning?" UniteAI, 2020.

[42] Stephanie Glen. "Decision Tree vs Random Forest vs Gradient Boosting Machines: Explained Simply" Data Science Central, 2019.

[43] "¿Qué son las Redes Neuronales?" IBM, 2020

[44] "Recursive Feature Elimination." Scikit, 2019

[45] Juan Ignacio Bagnato. "Comprende Principal Component Analysis." Aprende Machine Learning, 2018.

[46] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009

[47] Moustafa, Nour, and Jill Slay. "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset.Ïnformation Security Journal: A Global Perspective (2016): 1-14.

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

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

[50] Sarhan, Mohanad, Siamak Layeghy, Nour Moustafa, and Marius Portmann. 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). Springer Nature.

[51] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018