UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

Instituto de Física y Matemáticas
Licenciatura en Matemáticas Aplicadas

Detección de intrusiones en redes LAN mediante un modelo de aprendizaje máquina predictivo

TESIS
que para obtener el título de
Licenciado en Matemáticas Aplicadas
presenta

Citlalli Joselyn Gómez Rivera

Director de tesis:
Dr. Tomás Pérez Becerra
Codirector de tesis:
Dr. Salvador Sánchez Perales

Heroíca Ciudad de Huajuapan de León, Oaxaca *Julio de 2023*

Dedicatoria

A Dios, a mi madre y a todas aquellas personas que, de manera directa e indirecta, han brindado su apoyo incondicional a lo largo de mi trayectoria académica y crecimiento personal.

Agradecimientos

A mis padres y hermanos: ustedes han sido mi mayor fuente de apoyo y motivación a lo largo de este arduo camino académico. Su amor incondicional, paciencia y sacrificio han sido pilares fundamentales en mi formación y en la culminación de esta tesis. Gracias por creer en mí, por alentarme en cada paso y por ser mi mayor inspiración.

A la familia López Rivera: quienes han sido mi segunda familia y han formado parte de este sueño desde el primer momento. Agradezco sinceramente el amor y apoyo que me han brindado a lo largo de toda mi vida.

A mis amigos: Adrián, Bety y May durante este tiempo se han convertido en mi familia, les agradezco de corazón por haber compartido conmigo estos años. Guardo con cariño los momentos que hemos vivido juntos: las noches de desvelo, las risas y también los desafíos. Su apoyo y compañía han sido un regalo invaluable en este camino. Gracias por estar siempre allí y por ser parte de mi vida durante esta etapa tan significativa.

Zori, gracias por haber sido el mejor amigo que la universidad pudo regalarme. Tu apoyo incondicional, las risas compartidas y todos los momentos que hemos vivido juntos son tesoros que guardo en mi corazón. A pesar del tiempo y la distancia, espero que nuestra amistad perdure siempre.

Ricardo, Bogdan y George, agradezco mucho su amistad sincera. Sus palabras de ánimo, su apoyo en los momentos difíciles y las experiencias compartidas han dejado una huella significativa en mi vida.

A mi director de tesis: Dr. Tomás, quiero expresar mi más sincero agradecimiento por el apoyo que me brindó durante todo este proceso. Su paciencia en los momentos en los que mi ánimo decaía fue fundamental para superar los desafíos. Agradezco especialmente por haberme dado la oportunidad de trabajar a su lado, lo cual ha sido una experiencia inolvidable en mi formación académica.

A mis sinodales y revisores: quiero expresar mi más sincero agradecimiento por el tiempo dedicado a revisar este trabajo, por sus valiosas observaciones de mejora y por todo el apoyo que han brindado a lo largo del proceso.

A mis profesores: gracias por contribuir tanto a mi formación académica como personal, a lo largo de este tiempo he podido aprender mucho de ustedes y sin duda atesoro cada una de sus enseñanzas.

Índice general

Introducción

Con el paso del tiempo, el internet ha tomado un papel relevante en la sociedad, llegando así a ser actualmente una necesidad [32]. Con este auge, la actividad en línea resulta peligrosa si se descuida la información que ahí circula. La poca o nula protección de los datos a los cuales se puede tener acceso a través de una red de internet (de ahora en adelante, solo la llamaremos red) hace de estos un blanco fácil de intrusiones o de software malicioso. De esta manera surgen los *sistemas de detección de intrusos* (IDS) con el fin de dar solución a los problemas recurrentes de seguridad cibernética. Estos ataques de red (conocidos como "intrusiones") tienen las siguientes etapas:

- Análisis de puertos: como primer paso, el atacante analiza los puertos UDP y TCP (véase el glosario de términos) que utilizan los servicios de red del equipo objetivo y determina la vulnerabilidad de los mismos. También permite al intruso determinar el sistema operativo y seleccionar los ataques de red más apropiados.

- Ataques de denegación del servicio: hacen que el sistema operativo de destino se vuelva inestable o totalmente inoperable.

- Ejecución de la intrusión: "secuestra" al sistema operativo del equipo y el intruso consigue un control total del sistema operativo.

Estas intrusiones se generan cuando el atacante quiere obtener datos confidenciales de un

equipo remoto, como números de tarjetas bancarias o contraseñas, o utilizarlo secretamente para sus propios fines, por ejemplo, para atacar a otros equipos.

Si bien el trabajo de los IDS es nuevo, resulta ser un área prometedora y con un gran potencial de crecimiento, pues la globalización ha orillado a la sociedad de forma inevitable a trasladar su vida a la red. La base del funcionamiento de los IDS consiste en la clasificación del estado de una red por medio de patrones, ya sea con el uso de redes neuronales, ciencia de datos o aprendizaje máquina. Es conveniente resaltar que existen diferentes maneras en las que se puede atacar, corregir o inclusive prever ataques maliciosos, a los que se enfocan en prevenir irrupciones se les conoce como *sistemas de predicción de intrusiones* (IPS).

En los últimos años la protección cibernética se ha convertido en un aspecto sumamente relevante en casi cualquier ámbito social. Hasta el primer semestre del 2022, México es el primer lugar en intentos de ciberataques con un total de 85 mil millones de intentos lo que representa un aumento del 40 % con respecto a reportes de años anteriores ([26]). El 78 % de líderes de seguridad de corporaciones consideran que sus organizaciones no cuentan con una defensa lo suficientemente buena. Más aún, 62.7 % de las empresas creen que los ataques han ido en aumento desde el año 2020 con motivo de la pandemia por COVID-19 (véase [33]).

Ante tales inconvenientes, que afectan directamente a diversos sectores, se ha decidido invertir en buscar soluciones que mejoren la seguridad cibernética. Es así como en 2021 un aproximado del 91 % de empresas han decidido aumentar su presupuesto destinado a protección informática, aunque a la fecha sólo el 51 % de ellos han realizado dicha inversión, según el reporte de Insight (veáse [18]). No obstante, con respecto a las inversiones en ciberseguridad del 2020, se observa que el porcentaje ha disminuido un 5 %. Se desconocen los motivos de este declive, si bien podría ser un reflejo de la pandemia por COVID-19 o un cambio en las prioridades de la empresa. De todo lo anterior, se estima que el impacto que tendrán los ciberataques para el 2025 llegará a los 10.5 trillones de dólares anuales (véase [33]).

Las disciplinas con aplicación directa a la generación de conocimientos y a tecnología

han cobrado mucho valor recientemente, este es el caso de la ciencia de datos que ha tomado apogeo en los últimos años debido a la multiplicidad de áreas que recopila para trabajar en conjunto, tales como estadística, inteligencia artificial y métodos numéricos. La ciencia de datos está compuesta por una serie de técnicas para el análisis y procesamiento de datos y el desarrollo de alternativas que resuelvan problemas de forma más simple, por ejemplo, utilizando aprendizaje profundo que es un tipo de aprendizaje automático que emplea algoritmos con el fin de funcionar de forma análoga al cerebro humano. Por tanto, resulta ser una herramienta útil en el combate de los ataques, específicamente, en la detección de amenazas que pongan en riesgo la seguridad de la información circulante.

Esta tesis se enmarca dentro de esta área y el objetivo general es diseñar el modelo y la simulación de un programa de aprendizaje máquina predictivo de tipo IPS, capaz de distinguir entre conexiones de redes seguras y no seguras, es decir, las que se encuentran propensas a recibir intrusiones. Para ello, se utiliza la base de datos (también denominada directorio) *KDD cyberattack* y se modela basándonos en la teoría de k vecindades cercanas (*KNN*, por sus siglas en inglés).

El directorio es un conjunto de datos de uso libre. Se encuentra alojada en el servidor del *Donald Bren School of Information and Computer Science* de la *University of California* en Irvine, California, en los Estados Unidos, disponible en el link:

`http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`

y liberada para el concurso *The Third International Knowledge Discovery and Data Mining Tools Competition*, el cual se celebró junto con el *KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining*. La tarea de la competencia era construir un detector de intrusos en la red, esto es, un modelo predictivo capaz de distinguir entre conexiones "malas", llamadas intrusiones o ataques, y normales o "buenas". Contiene una serie de registros de una amplia variedad de intromisiones llevadas a cabo en la simulación de un entorno de red militar y aún es utilizada en diversas investigaciones sobre el tema dando como resultado diferentes artículos (veáse [3, 17, 19, 20, 28, 29]), en las cuales se muestran diversos modelos para realizar la detección de posibles vulnerabili-

dades. En [8], los modelos propuestos se complementan con un proceso de extracción de características relevantes, también se puede encontrar un resumen de los trabajos realizados hasta el momento y los pre-procesamientos realizados. Sin embargo, en la revisión bibliográfica realizada hasta el 2022 no se encuentran modelos predictivos que cuenten con un análisis exploratorio previo y que utilicen el modelo KNN.

Esta investigación se enfoca en la realización del sistema con el debido sustento matemático basado en técnicas de ciencia de datos, siendo este el objetivo general. Este sistema se realiza solo hasta la fase de simulación y se espera una tasa de efectividad de al menos el 75 % con un tiempo de ejecución relativamente corto, posteriormente a este trabajo, es posible implementar su automatización en alguna interfaz de programación de aplicaciones (API).

Los siguientes objetivos específicos permiten que se cumpla el objetivo general.

1. Se realiza una investigación sobre las actuales tendencias en ataques relacionados con tráfico de información en redes.

2. Se lleva a cabo un análisis exploratorio a la base de datos *KDD cyberattack* para identificar entradas faltantes y se realiza una extracción de características.

3. Se diseña un programa de aprendizaje máquina y se entrena con la base para que sea capaz de identificar patrones y realizar clasificaciones de una red y, así reconocer comportamientos atípicos (comportamientos se apartan de los modelos representativos o de los tipos conocidos) con la finalidad de prevenir al usuario sobre un posible ataque cibernético.

Los objetivos específicos se cumplen al considerar la tesis con los siguientes capítulos principales:

Capítulo 1. En este capítulo se muestran los preliminares relacionados con redes de internet, tales como definiciones importantes, componentes esenciales de una red, servicios, y tipos de protocolos. El lector se puede auxiliar del glosario de términos

adicional que se encuentra en la parte final del trabajo, donde se recopilan los conceptos principales que se emplean en esta investigación y sirven de apoyo en la comprensión de la problemática que se aborda. Todo lo anterior es con el fin de brindar al lector un panorama general del área donde se desenvuelve la presente.

Capítulo 2. En primera instancia se realiza el análisis exploratorio a la base donde se observan los tipos de datos presentes en la misma y la no existencia de datos faltantes. Posteriormente, se inspecciona el directorio mediante técnicas de visualización particularmente a través de histogramas de frecuencias de variables de interés. Cabe resaltar que con base en los dos procesos descritos previamente, se obtienen hipótesis que resultan relevantes sobre las intrusiones en las redes, un ejemplo es el protocolo de red preferido por los atacantes.

Capítulo 3. Primeramente a través del diagrama de calor se realiza el análisis de correlación entre las características, de esto surge la idea que no todas las variables son necesarias para el desarrollo de la investigación. De lo anterior, se realiza la selección de características basándose en el algoritmo SelectKBest. Con esto, se reduce la dimensión de las columnas de la matriz generada a través de la base de datos haciendo más sencilla la manipulación del directorio y a su vez reduciendo el tiempo de ejecución del sistema que se desarrolla en el Capítulo 4.

Capítulo 4. Con base en la reducción de dimensionalidad del directorio, se plantea el modelo matemático clasificador, el algoritmo a ejecutar y se muestran las etapas de entrenamiento y prueba de este. Una vez culminadas las dos fases, se valida el modelo bajo algunas métricas con el objetivo de conocer la eficiencia del mismo con respecto a la eficiencia de otros modelos encontrados en la literatura. Finalmente, se discuten los resultados obtenidos y se plantean algunas líneas de investigación para desarrollar posteriormente.

Para finalizar y a manera de síntesis, el desarrollo de este estudio permite conectar el análisis exploratorio y el aprendizaje máquina para desarrollar un IPS que podrá alertar

al usuario sobre potenciales intrusiones o ataques en redes. Esto mejorará la eficiencia en la predicción y aumentará la seguridad en la navegación. Además de que nos posibilita comprender de una forma práctica el proceso de la ciencia de datos aplicada en problemas reales.

Capítulo 1

Preliminares en redes

Durante los últimos tres siglos, han surgido tecnologías predominantes en cada uno de ellos. En el siglo XIX, la invención de la máquina de vapor revolucionó el mundo del transporte, mientras que en el siglo XX, la tecnología desempeñó un papel fundamental en la recopilación, procesamiento y distribución de información.

Los avances tecnológicos han facilitado la comunicación a lo largo del tiempo. Se establecieron redes telefónicas a nivel mundial, surgieron la radio y la televisión, y la industria de la computación nació y creció. Finalmente, el lanzamiento de satélites y, por supuesto, la llegada del internet, revolucionaron la vida cotidiana.

En este capítulo, se presenta una visión general del internet, incluyendo sus componentes principales, así como algunas características, problemas y alternativas. El objetivo es brindar una comprensión más sólida de la teoría involucrada en el problema que aborda la tesis, buscando así proporcionar una mejor comprensión.

1.1. El internet

Desde sus orígenes, en 1969, el término *internet* ha cobrado renombre con el pasar de los años, siendo ahora uno de los medios más efectivos de comunicación a nivel mundial y desempeña un papel fundamental para la presente investigación, de acuerdo con la RAE

internet se define como:

> *"Red informática mundial, descentralizada, formada por la conexión directa entre computadoras mediante un protocolo especial de comunicación."*

Además, según Snell ([31]):

> *"El internet es una red o más exactamente una red de redes, una conexión vasta de diferentes tipos de computadoras esparcidas por todo el mundo que pueden compartir mensajes e información."*

Otros autores como James F. Kurose y Keith W. Ross plantean en [22] dos formas de definir al internet. La primera es a través de la descripción de sus componentes, es decir, del hardware y software básicos que lo conforman. La segunda es describiéndolo en términos de su infraestructura de red, lo que proporciona servicios a aplicaciones distribuidas.

1.1.1. Componentes esenciales

Como respuesta a la interrogante *¿qué es el internet?* desde la perspectiva del hardware, encontramos los siguientes componentes esenciales, ilustrados en la Figura 1.1:

Todo dispositivo con la capacidad de conectarse a internet recibe el nombre de *host* o *sistema terminal*. En los últimos años, objetos menos convencionales son capaces de acceder a internet como lo son termostatos, refrigeradores, bocinas, entre otros; por tanto el término *redes de computadoras* comienza a resultar obsoleto. Los host se interconectan mediante una red de *enlaces de comunicaciones* y *conmutadores de paquetes* que pueden transmitir datos a distintas velocidades, estas se miden en *bits/segundos*. Cuando un sistema terminal se tiene que comunicar con otro, el emisor segmenta la información y añade bytes a la cabecera de cada segmento, al llegar al destino, el host receptor los ensambla nuevamente para obtener el mensaje original.

Otro componente es el *conmutador de paquetes*, que toma un paquete entrante por uno de sus enlaces de comunicaciones y lo reenvía a través de otro. Si bien es cierto que hay gran variedad y tipos de conmutadores, los regularmente empleados en la actualidad son

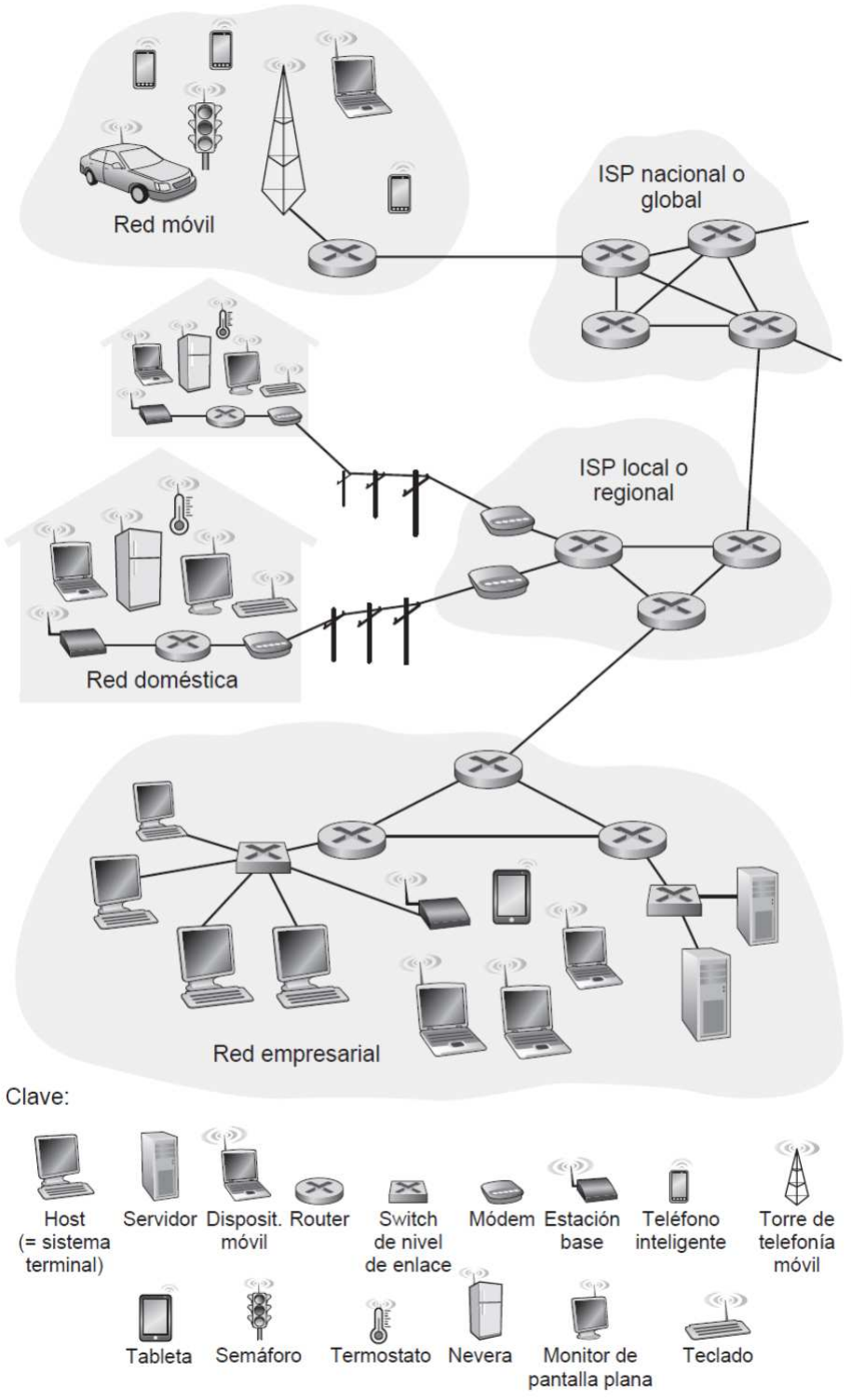


Figura 1.1: Algunos componentes de internet. Obtenida de [22].

los *routers* y *switches de la capa de enlace* (más adelante se abordan las capas del internet y su principal función). Los primeros suelen emplearse en el núcleo de la red mientras que los segundos lo hacen en las redes de acceso.

La *ruta a través de la red* es la secuencia de enlaces de comunicaciones y conmutadores que atraviesa un paquete desde el sistema terminal emisor hasta el receptor. Una buena analogía sobre el funcionamiento de las redes de conmutación de paquetes son las de transporte, siendo los vehículos los medios y las carreteras el camino a seguir.

Los host acceden a internet a través de los ISP (Proveedor de servicios de internet, por sus siglas en inglés). Existen diferentes tipos, algunos de los más conocidos se encuentran los corporativos, como lo son las compañías telefónicas; los universitarios; aquellos que proporcionan accesos inalámbricos (WIFI) en aeropuertos y plazas; y de datos móviles, utilizados en celulares y laptops.

Los ISP proporcionan a los host gran variedad de formas para acceder a una red, entre los cuales destacan el acceso de banda ancha residencial, el acceso *LAN* (Red de área local, por sus siglas en inglés) de alta velocidad y el acceso inalámbrico para dispositivos móviles. Es importante recordar que cada ISP es en sí mismo una red de conmutadores de paquetes y enlaces de comunicaciones.

Los sistemas terminales, los conmutadores de paquetes y otros dispositivos ejecutan *protocolos* que controlan el envío y la recepción de información en internet. En la sección 1.2 se abordan con mayor profundidad las características de cada uno.

1.1.2. Servicios

En la subsección previa se abordan los componentes esenciales con respecto al hardware del internet. En esta se habla de los servicios, es decir, del software que componen internet.

Dentro de internet existen diversas aplicaciones que proporcionan diferentes servicios como son recibir correos electrónicos, ver películas, escuchar música desde la nube, entre otros muchos más. A tales aplicaciones se les conoce como *aplicaciones distribuidas* y se ejecutan en los sistemas terminales. Si bien los conmutadores facilitan el intercambio de

información entre sistemas terminales, estos no prestan atención al papel de la aplicación, es decir, si actúa como origen o destino, por ejemplo, en el tráfico de datos dentro de una red local el conmutador se centra únicamente en dirigir los paquetes al destino correcto en función de las direcciones MAC (Media Access Control) y no tiene en cuenta la función específica de la aplicación.

Los host conectados a internet proporcionan una *interfaz de sockets de internet*, la cual es un conjunto de reglas que el programa que transmite los datos debe cumplir para que la información llegue al programa de destino. Tal interfaz especifica la manera en como un programa ejecutado en un sistema terminal pide a la infraestructura de internet que provea datos a un programa de destino que se ejecuta en otro sistema terminal.

En conclusión, se podría describir al internet como una *infraestructura que proporciona servicios a las aplicaciones.*

1.1.3. Modelos OSI y TCP/IP

Para comprender de mejor manera el funcionamiento del internet se recurre a la *arquitectura en capas*. Esta consta en dividir la aplicación en capas, con el fin de que cada una de ellas tenga una tarea específica. Las *capas* son una forma de designar responsabilidades y administrar dependencias, la dependencia es de la siguiente manera: una capa superior puede recurrir a los servicios de una capa inferior, pero no al revés.

En cuanto a los modelos de red, es posible encontrar dos modelos principales, el modelo OSI (Open Systems Interconnection) y el modelo TCP/IP (Transmission Control Protocol/Internet Protocol).

El *modelo OSI* es un sistema normativo que consta de siete niveles o capas. Define la interconexión de los sistemas en las distintas fases por las cuales atraviesan los datos para viajar de un dispositivo a otro dentro de una determinada red de telecomunicaciones. A continuación se brinda el panorama general de las capas así como su funcionalidad dentro de este arquetipo.

1. **Capa física:** es la capa más baja del modelo, como su nombre lo dice, en esta

recaen los aspectos físicos de la red, es decir, los aspectos tangibles como lo son cables, conectores, routers, entre otros.

2. **Capa del enlace de datos:** su función es desplazar los datos a través del enlace físico hasta el nodo receptor, el cual mediante la dirección de hardware de cada sistema terminal conectado a la red, es capaz de identificarlo.

3. **Capa de red:** en esta se determina la ruta que seguirán los datos y el intercambio correcto de los mismos dentro de la ruta marcada. Es decir, esta capa es la encargada de indicar la ruta de los paquetes así como de la entrega de los mismos.

4. **Capa de transporte:** esta capa tiene dos funciones. La primera es controlar el flujo de datos en los nodos que tienen comunicación. La segunda es valorar las dimensiones de los paquetes, buscando así que estos tengan medidas acordes a los requerimientos de capas inferiores.

5. **Capa de sesión:** es la encargada de instaurar el vínculo de comunicación o sesión entre el equipo emisor y el receptor; así mismo, gestiona la sesión establecida.

6. **Capa de presentación:** es considerado el "traductor" del modelo OSI pues aquí se convierten los paquetes provenientes de la capa de aplicación a un formato genérico que pueda ser leído por cualquier sistema terminal. También en caso de requerirse, en esta capa se cifran y comprimen los datos para reducir su dimensión. Usualmente los paquetes que aquí se forman son los que pasarán por todas las demás capas del modelo.

7. **Capa de aplicación:** suministra las herramientas con las que interactúa el usuario, además proporciona los servicios de red relacionados a las aplicaciones. Es decir, brinda la interfaz y servicios que soportan las aplicaciones de usuario.

El *modelo TCP/IP* es un sistema que se forma de cuatro capas, basándose en la funcionalidad de los protocolos TCP e IP, pero más que un modelo es una arquitectura que consta de una pila de protocolos:

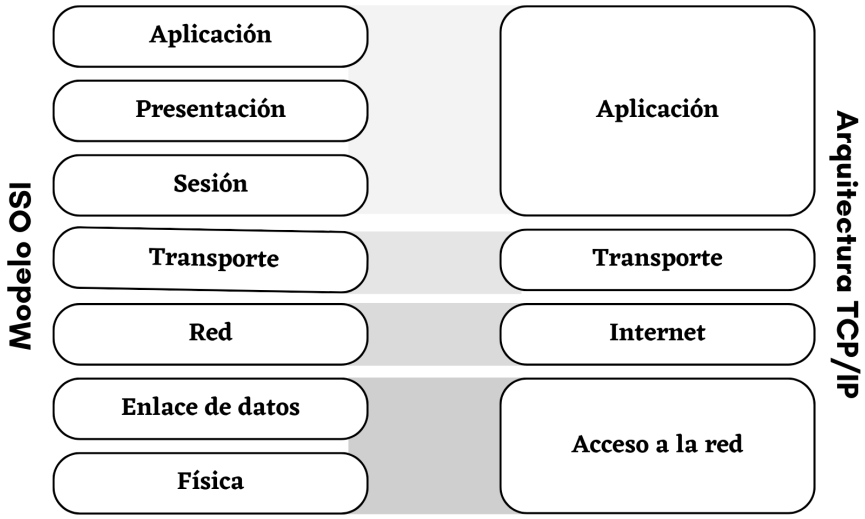


Figura 1.2: Comparación del modelo OSI con respecto a la arquitectura TCP/IP. Imagen extraída y modificada de [2].

1. **Capa de acceso a la red.**
2. **Capa de internet.**
3. **Capa de transporte.**
4. **Capa de aplicación.**

La funcionalidad de cada capa de esta arquitectura engloba funciones específicas del modelo OSI, mostradas en la Figura 1.2.

1.2. Protocolos

Como se mencionó en la sección 1.1.1, muchos dispositivos ejecutan protocolos para realizar sus tareas, por tanto, interesa ahondar en las principales características así como su importancia para la correcta realización de tareas de algunos dispositivos en internet.

Un *protocolo de red* es un conjunto de estándares y políticas formales, conformadas por restricciones, procedimientos y formatos que definen el intercambio de información para la comunicación entre dos o más servidores.

Toda actividad de internet que involucre la comunicación entre dos o más entidades no cercanas se rige por un protocolo, he ahí donde radica su importancia. Los protocolos se manejan en capas, donde los que se encuentran en una capa superior funcionan gracias a la implementación de los de capas inferiores.

Existen diferentes tipos de protocolos que se clasifican en dos tipos: de acuerdo a su funcionalidad y con base en la capa en la cual se ejecutan. Entre los del primer tipo se destacan:

- **Administradores de redes**: brindan las especificaciones para manejar una red con eficacia y se involucran los diferentes dispositivos que componen tal red. Ejemplos de estos son SNMP (Simple Network Management Protocol) e ICMP (Internet Control Message Protocol).

- **Comunicación**: permiten la comunicación entre dispositivos de una red. Es empleado tanto en comunicación digital como analógica. Tal es el caso de los protocolos TCP, IP y HTTP (Hypertext Transfer Protocol).

- **Seguridad**: trabajan en garantizar la seguridad de los datos enviados por la red y así usuarios no autorizados no puedan acceder a ellos. Algunos ejemplos son los protocolos HTTPS (Hypertext Transfer Protocol Secure), SSL (Secure Sockets Layer) y SFTP (Secure File Transfer Protocol).

Con respecto a la segunda clasificación, se tienen algunos de los protocolos más usados:

- **Capa de internet**: aquí destacan los protocolos IP, ARP (Address Resolution Protocol) e ICMP.

- **Capa de transporte**: en esta capa funcionan los protocolos TCP y UDP (User Datagram Protocol).

- **Capa de aplicación**: se encuentran los protocolos HTTP, FTP (File Transfer Protocol), telnet, SSH (Secure Shell), SMTP (Simple Mail Transfer Protocol), POP (Post Office Protocol) y DNS (Domain Name System).

1.3. Redes LAN y WAN

Con base en su dominio geográfico y en la tecnología empleada es posible clasificar en dos grupos a los dispositivos que conforman las redes: Redes de Área Local (LAN, por sus siglas en inglés Local Area Networks) y Redes de Área Amplia (WAN, por las siglas de Wide Area Networks). Las redes LAN usualmente son de propiedad privada y abarcan una espacio físico relativamente pequeño como una residencia o un edificio. Las redes WAN son de largo alcance, pues abarcan desde un país hasta un continente. Necesitan cruzar rutas de acceso público y utilizan circuitos proporcionados por un proveedor de servicios de telecomunicaciones. Cada red consiste en un conjunto de sistemas terminales interconectados. Su fin es transportar datos, por tanto, su principal función tiene relación con el enrutamiento de información. Son redes operadas por Proveedores de Servicios de internet.Una red WAN puede ser compuesta por redes LAN, veáse la Figura 1.3. Por lo general, los host de estas redes pertenecen a diferentes personas o bien son operadas por distintos usuarios. Por ejemplo, una empresa con sucursales alrededor del mundo.

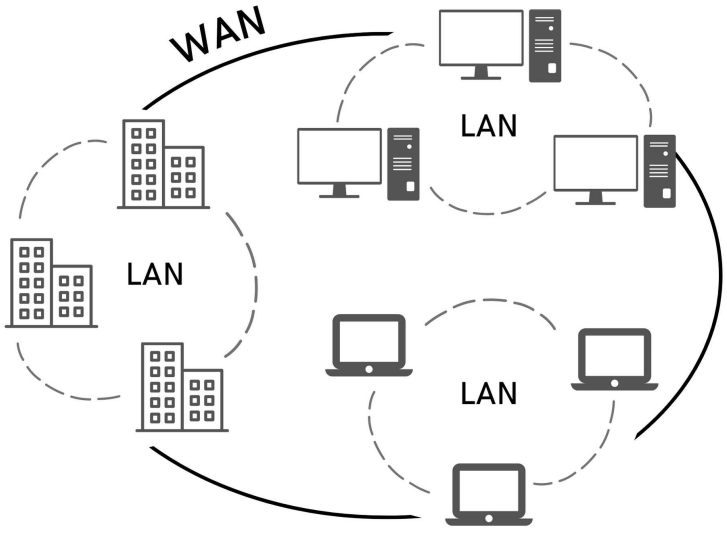


Figura 1.3: Tamaño LAN y WAN. Imagen extraída y modificada de [34].

1.4. Seguridad en redes

1.4.1. Principales riesgos

Existen muchos riesgos a los cuales se enfrentan los usuarios al tener acceso a internet, algunos por falta de conocimientos en el tema y otros más por ataques directos a la red. De acuerdo con [35], los siete problemas comunes en ciberseguridad son:

1. **Ignorancia**: el desconocimiento de algún tema puede ocasionar problemas, este es el caso del internet y los riesgos dentro de ella por falta de información de un uso seguro.

2. **Malware y bots**: el malware o "software malicioso" es un término muy amplio, hace referencia a programas o códigos maliciosos que resulten dañinos para los sistemas que acceden a internet, a menudo asumiendo el control parcial de las operaciones del dispositivo. Los bots se consideran maliciosos pues pueden programarse (o alterarse) para hackear cuentas de usuario, enviar correos no deseados, recorrer la Web en busca de datos de contacto o realizar otras acciones malignas.

3. **Cuentas hackeadas por phishing**: el phishing es una forma común para obtener el control de la información personal de los clientes a través de crímenes digitales. La manera de operar es mediante correos electrónicos aparentemente enviados por empresas reales solicitando datos, los mismos que en su mayoría se utilizan para realizar fraudes financieros.

4. **Spam (correo no deseado)**: es el envío masivo de mensajes no solicitados y no deseados, generalmente de naturaleza publicitaria. El envío de "spam", mayormente es dañino. Algunas de estos correos pueden contener un enlace o un archivo que descarga algún tipo de virus. Usualmente este programa maligno no es el problema más fuerte sino el troyano que viene dentro de él.

5. **Hogares inseguros con redes inalámbricas**: contar con internet en casa puede resultar peligroso si no se tienen las precauciones adecuadas. Al día de hoy existen

herramientas gratuitas que cualquier cibernauta tiene la posibilidad de emplear para descifrar claves sencillas. Aproximadamente, el 50 % de las contraseñas son: 1-2-3-4-5-6, 9-8-7-6-5-4-3-2-1, ABC, 99999 o 55555, seguidas por el 10 % que son fechas de nacimiento o el nombre del usuario.

6. **Datos perdidos**: la pérdida de información con motivo del extravío de dispositivos electrónicos conlleva más que gastos económicos. En muchas ocasiones los datos de los usuarios no se respalda por diversos motivos, misma que puede usarse para fines ilícitos, por ende es fundamental proteger los dispositivos electrónicos y la información ahí almacenada de posibles robos o extravíos.

7. **Ataques por Wi-Fi**: muchos lugares públicos cuentan con Wi-Fi gratuito, que resulta muy atractivo para los usuarios, pero así mismo son un blanco fácil de delincuentes que, a través de dichas redes, son capaces de acceder a la información de algún usuario o bien realizar acciones no correctas para no ser detectados con facilidad.

En cuanto a los ataques comunes con respecto a las capas de funcionamiento de las redes, se tiene la siguiente distribución:

1. **Nivel físico**: las vulnerabilidades se encuentran relacionadas con el acceso no autorizado a los dispositivos.

2. **Nivel de enlace de datos**: algunos ataques son falsificación de direcciones MAC y envenenamiento ARP (Address Resolution Protocol) que consiste en manipular la tabla de resolución ARP con el objetivo de redirigir el tráfico de red hacia un equipo malintencionado.

3. **Nivel de red**: un riesgo es la suplantación de mensajes, es decir, se envían paquetes de una dirección IP diferente y se suplanta por alguna falsa o de algún dispositivo legítimo en la red. Otro más es la denegación de servicio, cuyo objetivo es saturar la red de la víctima para que esta no pueda acceder de forma óptima.

4. **Nivel de transporte**: algunos problemas por intrusos son denegación de servicio, ataques de reconocimiento y contra el establecimiento de sesiones TCP.

5. **Niveles de sesión, presentación y aplicación**: hay muchos ataques que aprovechan las vulnerabilidades de estos niveles, como son aquellos sobre la confidencialidad, agotamiento de direcciones IP, escala de directorio, suplantación de servicio de nombres de dominio, entre otros.

6. **Ataques de denegación de servicios en redes**: algunos de los riesgos son inundación IP, la cual consiste en degradar los servicios de red a través del envío de tráfico masivo y falsificación de la IP de origen. Estos se dividen en *broadcast* y *smurf*.

1.4.2. Técnicas de protección

Proteger las redes de internet es algo de suma relevancia en la actualidad dada la importancia que han adquirido en diversos sectores de la sociedad. A continuación se presentan algunas soluciones a nivel técnico para este fin (véase [5]).

- **Cortafuegos**: es una barrera entre una red interna segura y una que no sea de confianza. El cortafuegos proporciona un único punto de estrangulamiento, el cual es un puerto de contacto controlado en medio de las redes confiable y la que puede no ser fiable. Sus políticas son:

  - Política restrictiva: se rechaza todo el tráfico por defecto y solo se permite el paso de ciertos paquetes muy específicos.
  - Política permisiva: se aceptan todos los paquetes en el tráfico a excepción de algunos previamente especificados.

  Entre las funciones destacan:

  - Permitir a los usuarios de la red interna echar mano de recursos situados fuera de la red local.

- Impedir que usuarios sin autorización de la red externa puedan utilizar recursos definidos en la red interna como lo son servidores, dispositivos de red, aplicaciones y servicios.

- **Zonas desmilitarizadas**: también conocidas como *DMZ*, son redes locales ubicadas en la frontera de una red interna y una red externa; generalmente la primera es del usuario y segunda es internet. La principal función es permitir el préstamo de servicios de la interior a la exterior.

- **Detector de intrusos**: son un tipo de alarmas ante intrusiones, su objetivo es detectar y alertar la presencia de intrusos en su radio de acción. Dentro de esta clase de técnicas se distinguen los IDS (Intrusion Detection System) y los IPS (Intrusion Prevention System).

  - Un IDS es una aplicación que tiene por objetivo detectar accesos sospechosos o bien sin autorización a una red u ordenador. Cuando estos detectan alguna amenaza suelen emitir alertas a los administradores del sistema, todo esto en tiempo real, lo cual resulta ventajoso para el usuario afectado, la desventaja es que no buscan erradicar la intrusión sino simplemente avisar de ella.

  - Un IPS es una clase particular de software que actúa de manera preventiva con el fin de salvaguardar los sistemas de probables ataques o invasiones. Su funcionamiento se basa en la inspección de las conexiones de forma continua, examina patrones y posibles anomalías para determinar si se está ejecutando o se va a ejecutar algún comportamiento sospechoso que ponga en riesgo la integridad de la red.

- **Proxies**: un proxy es una tecnología empleada como puente entre el origen (el dispositivo) y el destino (internet). Usualmente es un mecanismo intermedio que permite la conexión a internet indirectamente (véase [24]). Algunas ventajas de su implementación son:

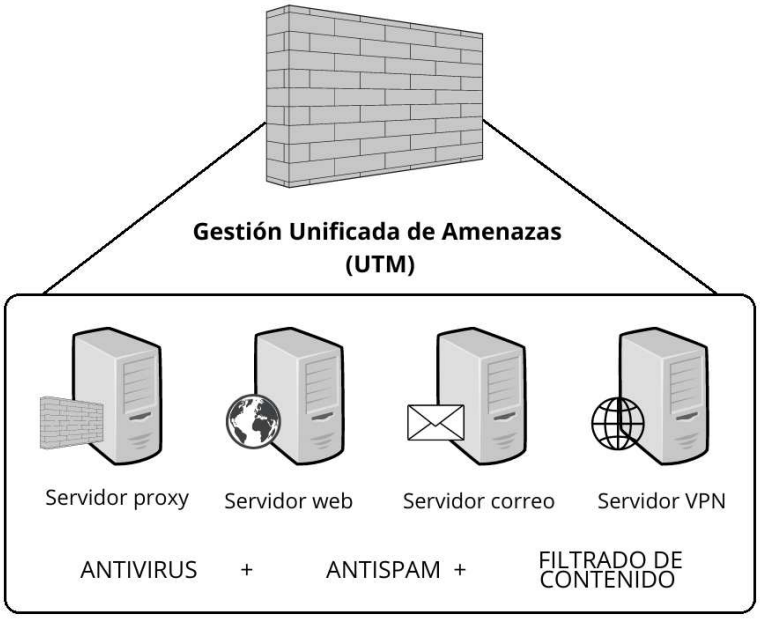


Figura 1.4: Funciones de una UTM. Imagen extraída y modificada de [12].

- El uso de un caché lo suficientemente grande permite una navegación más rápida;
- Proporciona seguridad protegiendo los equipos que tienen acceso a la red externa;
- Favorece la definición de filtros de contenido y listas de control de acceso para que las organizaciones tengan la posibilidad de realizar monitoreo del servicio que se está empleando.

- **Gestión unificado de amenazas**: los dispositivos conocidos como UTM (Unified Threat Management) combinan varias técnicas de protección de redes, por ejemplo, antivirus, antispyware, antispam, cortafuegos, advertencia y detección de intrusiones, filtrado de contenido y prevención de fugas. En la Figura 1.4 se engloban las funciones principales de una UTM. Algunas de las ventajas son bajo costo y complejidad menor, sin embargo las desventajas son problemas de rendimiento y un único punto de falla.

Capítulo 2

Análisis exploratorio

El modelo del sistema de prevención de intrusiones con el que se trabaja está creado sobre la base de datos *KDD cyberattack*, esta fue utilizada en The Third International Knowledge Discovery and Data Mining Tools Competition, la cual se desarrolló a la par con The Fifth International Conference on Knowledge Discovery and Data Mining (KDD-99). Este concurso se llevó a cabo en el año 1999 y su objetivo fue desarrollar un algoritmo en redes de internet capaz de discernir entre ataques, invasiones, conexiones buenas y normales.

En este capítulo se realiza un análisis detallado de la base, tal como la determinación de los tipos de datos de la misma, existencia o no de datos faltantes y el comportamiento de las variables con base en las medidas de tendencia central e histogramas de frecuencias.

2.1. Base KDD

En 1988 se inició el Intrusion Detection Evaluation Program de la *Defense Advanced Research Projects Agency* (DARPA) de los Estados Unidos de América y el laboratorio *Lincoln Labs* del *Massachusetts Institute of Technology* (MIT) se encargó de su administración. El objetivo era recabar información e iniciar una investigación acerca de la detección de intrusos.

Lincoln Labs configuró un entorno para adquirir, en principio durante nueve semanas, datos de volcado de TCP sin procesar de una red de área local, que simulaba una LAN típica de la Fuerza Aérea de los Estados Unidos. Operaron la LAN como si fuera un verdadero ambiente y fue objeto de múltiples ataques.

El tamaño de la base obtenida era de alrededor de cuatro gigabytes ya comprimidos, resultado solo de siete semanas de tráfico de red. Esto se procesó en aproximadamente cinco millones de registros de conexión, del mismo modo, las dos semanas de prueba restante arrojaron dos millones más. Con la característica de ser en su mayoría binarios dada la naturaleza propia del experimento; por ejemplo, debido a que una conexión es una secuencia de paquetes TCP que comienzan y terminan en momentos bien definidos, entre los cuales los datos fluyen hacia y desde una dirección IP de origen a una IP de destino bajo algún protocolo definido, cada conexión se etiqueta como normal o ataque (0 o 1), con exactamente un tipo de ataque específico; en este caso, cada registro de conexión consta de cerca de 100 bytes.

Los ataques se dividieron en cuatro categorías principales:

DOS: denegación de servicio, por ejemplo, *syn flood* que es un ataque de denegación de servicio (DDos) en el cual se consumen los recursos disponibles con el fin que algún servidor no esté disponible para el tráfico legítimo;

R2L: acceso no autorizado desde una máquina remota, por ejemplo, adivinar contraseña;

U2R: acceso no autorizado a privilegios de superusuario local (raíz), por ejemplo, varios ataques de "desbordamiento de búfer";

Sondeo: vigilancia y otros sondeos, por ejemplo, exploración de puertos.

Algunos expertos en intrusión creen que la mayoría de los ataques novedosos son variantes de otros ya conocidos y que la "firma" de estos puede ser suficiente para detectar versiones novedosas. Los conjuntos de datos contienen un total de 24 tipos de intrusiones con 14 adicionales de variedades.

Posteriormente, Lincoln Labs determinó que una parte del conjunto de datos fuera utilizado para fomentar nuevas investigaciones en este campo y actualmente se encuentra almacenada en el servidor del *Donald Bren School of Information and Computer Science* de la *University of California* en Irvine, California, en los Estados Unidos. El concurso de detección de intrusos KDD de 1999 utilizó el 10 % de la base y es la que se utiliza en esta investigación.

2.2. Tipos de datos

La base cuenta con 42 variables, también llamadas características o columnas y 494,020 registros o vectores fila. En el anexo se muestra la descripción de las variables y en la Tabla 2.2.1 se muestran los primeros 5 registros con los respectivos nombres de algunas características, en particular, la característica "labels" muestra si el registro representa una conexión segura o una intrusión mediante las etiquetas "normal", o bien, el nombre de la intromisión registrada. Por lo tanto, esta última será nuestra variable objetivo, con ello, el problema se convierte en uno de aprendizaje máquina supervisado.

duration	**protocoltype**	**service**	**flag**	**srcbytes**	...	**labels**
0	tcp	http	SF	181	...	normal
0	tcp	http	SF	239	...	normal
0	tcp	http	SF	235	...	normal
0	tcp	http	SF	219	...	normal
0	tcp	http	SF	217	...	normal

Tabla 2.2.1: Primeros cinco registros de la base, la mayoría de las variables no son mostradas.

Ahora, si consideramos a cada uno de los registros como vectores d dimensionales, entonces la base de datos se clasifica como en la definición siguiente obtenida de [1].

Definición 2.2.1. Un *conjunto de datos multidimensionales* $\mathcal{D}$ es un conjunto de n vectores de características, $\overline{X_1}...\overline{X_n}$, tal que cada vector de características $\overline{X_i}$ contiene un

conjunto de d características denotadas por $(x_i^1 ... x_i^d)$.

Con base en la Definición 2.2.1, se referirá a la i-ésima entrada del directorio como $\overline{X_i}$. Con lo anterior, algunos de los vectores de la base son

$$\begin{aligned}
\overline{X_1} &= (x_1^1, x_1^2, x_1^3, ..., x_1^d) = (0, tcp, http, DF, 181, 5450, ..., 0.0, 0.0, normal), \\
\overline{X_2} &= (x_2^1, x_2^2, x_2^3, ..., x_2^d) = (0, tcp, http, DF, 239, 486, ..., 0.0, 0.0, normal), \\
\overline{X_3} &= (x_3^1, x_3^2, x_3^3, ..., x_3^d) = (0, tcp, http, DF, 235, 1337, ..., 0.0, 0.0, normal), \\
&\vdots \\
\overline{X_{494020}} &= (x_{494020}^1, x_{494020}^2, x_{494020}^3, ..., x_{494020}^d) = (0, tcp, http, SF, 219, 1234, ..., 0, normal).
\end{aligned}$$

De ahora en adelante $\mathcal{D}$ representará a la base *KDD Cyberattack* y los valores de n y d serán igual a $494,020$ y 42, respectivamente.

Ahora bien, para comprender de mejor forma la estructura de los datos, se tiene la siguiente clasificación:

- *Datos cuantitativos multidimensionales*: se denominan de esta manera cuando cada una de las componentes x_i^j del conjunto $\mathcal{D}$ contiene únicamente datos en los que todos los campos son cuantitativos.

- *Datos de atributos categóricos y mixtos*: se conocen así cuando cada componente x_i^j es de tipo categórica o de valor discreto no ordenado. En el caso de datos con atributos mixtos, existe una combinación de atributos categóricos y numéricos.

- *Datos binarios y establecidos*: los *datos binarios* pueden ser considerados un caso particular de los *categóricos multidimensionales*, puesto que cada atributo categórico toma uno de dos valores discretos como máximo. Además, es posible considerarse como un caso especial de los *cuantitativos multidimensionales* pues existe un orden entre los dos valores. Usualmente, el valor que toman las variables es 1 o bien 0, donde el 1 indica que el elemento cumple cierta propiedad, en caso contrario la variable contiene al 0.

- *Datos de texto*: se pueden considerar como una cadena o como datos multidimensionales, según se representen. Un documento de texto en su forma original corresponde a una cadena y esta es una secuencia de caracteres (palabras) correspondientes al documento, aunque los documentos no suelen ser representados a través de cadenas.

De esta forma, se concluye que $\mathcal{D}$ contiene datos *mixtos*, debido a que posee cuantitativos como lo son *duration* y *urgent*, binarios como *land* y de texto, por ejemplo, *protocol_type*. Esto nos obliga a hacer un procesamiento previo de las variables, el cual se efectúa en el Capítulo 3; tal proceso nos permite uniformizar las características para, posteriormente, entrenar y validar el modelo de aprendizaje máquina.

2.3. Exploración y visualización de los datos

En esta sección se realiza el proceso de extraer información a partir de los datos mediante herramientas estadísticas, para ello, se requiere del cálculo de las medidas de tendencia central (MTC) y de dispersión para cada una de las variables cuantitativas de la base $\mathcal{D}$, que adaptadas a nuestro contexto son:

Definición 2.3.1. Consideremos la i-ésima variable cuantitativa de $\mathcal{D}$, esto es, la i-ésima columna con valores numéricos. La *media* de las n entradas $x_1^i, x_2^i, ..., x_n^i$ está dada por

$$\overline{x}^i = \frac{1}{n}\sum_{j=1}^{n} x_j^i.$$

La *media poblacional* correspondiente se denota como μ^i.

En la Tabla 2.3.1 localizada al final del capítulo se muestran las medias de algunas variables. Como ejemplo, la de la variable *duration* es 47.97 segundos. Por otro lado, en la Tabla 2.2.1 se observa que existen valores iguales a cero en algunos de los registros de la característica *dsthostsrverrorrate*, pero la media es 0.17, esto indica la existencia de valores distintos del nulo. Debido a la relativa enorme cantidad de datos, no es posible observar

por completo la base $\mathcal{D}$, sin embargo, la media nos proporciona una idea inicial del estado de la variable, esta es la razón principal de la realización de la etapa de exploración.

Definición 2.3.2. La *varianza* de la i-ésima variable cuantitativa de $\mathcal{D}$, con entradas $x_1^i, x_2^i, ..., x_n^i$, es la suma del cuadrado de las diferencias entre las mediciones y su media, dividida entre $n-1$. Simbólicamente, la varianza muestral es

$$(s^i)^2 = \frac{1}{n-1}\sum_{j=1}^{n}(x_j^i - \overline{x}^i)^2.$$

La correspondiente *varianza poblacional* está denotada por el símbolo $(\sigma^i)^2$.

A manera de ejemplo, la variable *duration* tiene una varianza de $500,895$ segundos, equivalente a 139.1375 horas, lo cual indica un rango relativamente grande de valores que recorre.

Definición 2.3.3. La *desviación estándar* de la i-ésima variable cuantitativa de $\mathcal{D}$, es la raíz cuadrada positiva de la varianza; esto es,

$$s^i = \sqrt{(s^i)^2}.$$

La correspondiente *desviación estándar poblacional* está denotada por $\sigma^i = \sqrt{(\sigma^i)^2}$.

En este caso, la variable *duration* tiene una desviación estándar de 707.74 segundos, o bien 0.1965 horas lo que representa una alta variabilidad en la característica, ya que sus valores se encuentran en el rango de 0 a $58,329$.

Con base en las medidas de tendencia central y de dispersión se pueden extraer la siguiente información:

1. La duración media de las conexiones es de 47 segundos (variable *duration*), con una desviación estándar de 707. Además, el tiempo que dura una conexión atacante varía en un rango muy amplio, desde valores muy cercanos a 0 segundos hasta 16.2 horas.

2. Las variables *srcbytes* y *dstbytes* son el número de bites transmitidos desde el origen hacia el destino y viceversa, respectivamente. En este caso, la primera variable tiene

un mínimo de 0 y un máximo de 693.37564 megabytes, la segunda tiene un rango desde 0 hasta 5.155468 megabytes. En promedio, ingresan a la red 3,025 bites y salen 868.

3. La variable *count* es el número de conexiones al mismo host realizadas en los últimos dos segundos, en este caso, tiene una media de 332 con una desviación estándar de 213, con un máximo de 511 enlaces.

4. La variable *srvcount*, al igual que la anterior, es el número de conexiones al mismo servicio en los últimos dos segundos. Esta presenta una media de 292.9 redondeada a 293 por ser una variable entera y una desviación estándar de 246.

5. El resto de variables presentan medias con valores cercanas a cero y desviaciones estándar relativamente pequeñas, lo que, en principio, indica una baja variabilidad y proporciona una primera idea de que tienen casi nula correlación con la existencia o no de intrusiones en las redes, sin embargo, esto no es concluyente hasta que se realice un análisis con mayor profundidad en las siguientes secciones, ya que descartarlas en este punto puede conducir a un error grave, esto es, eliminar una variable que afecte significativamente a la variable objetivo.

6. Existen, en su mayoría, diversas variables binarias. La media proporciona nociones sobre la prevalencia del 0 con respecto al 1 y viceversa, cuando el valor es muy cercano a 0, se entiende que la frecuencia del 0 es dominante en comparación con la del 1 (véase la Figura 2.1).

En la Figura 2.1 se observa la distribución de algunas de las variables, en particular, y como se mencionó anteriormente, existen variables binarias (con valores 0 y 1) que presentan una frecuencia relativamente más grande hacia uno de los valores, por ejemplo, la variable *land* muestra una enorme cantidad de valores 0 y casi una nula cantidad de valores 1, esto significa que las conexiones en los ataques provienen de distintas fuentes, es decir, la mayoría de los ataques realiza una comunicación desde distintos servidores, con el objetivo de que se dificulte su rastreo; sin embargo, al ser una variable de este tipo,

se concluye que es algo común en las intrusiones. Esto podría conducir al investigador a pensar que tales variables podrían ser no significativas; no obstante, al analizar a la variable *logged_in* se observa que la predominancia del 0 es considerablemente mayor que la del 1, gráficamente la proporción es aproximadamente de 7:1, por tanto se infiere que en este caso no sería conveniente descartarla, ya que es posible que sea significativa en la investigación. Nuevamente, se requiere de un análisis de mayor profundidad que se realiza en el Capítulo 3.

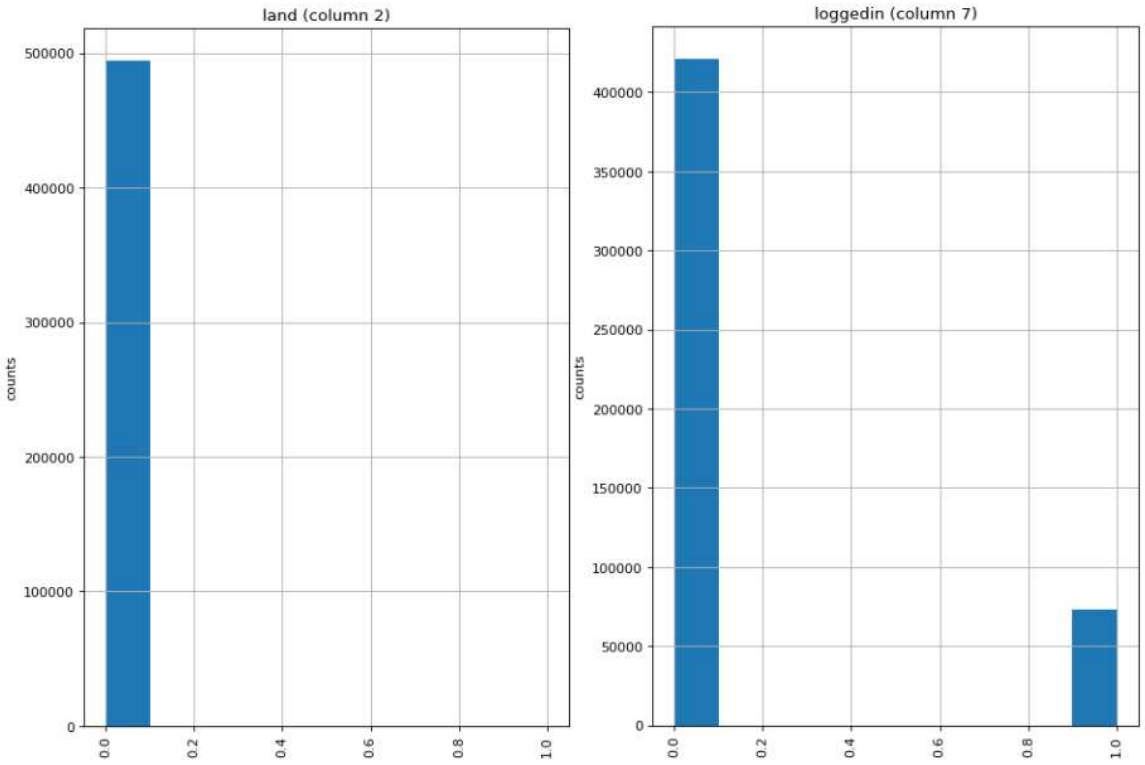


Figura 2.1: Frecuencias de las variables *land* y *loggedin*.

Ahora, obsérvese la gráfica de la variable *labels* en la Figura 2.2, en ella se muestran los principales tipos de intrusiones registradas así como la frecuencia de redes seguras, es decir, normales. La mayor predominancia son las intrusiones de tipo *smurf* y *neptune*, seguidas en frecuencia del estado *normal* (o seguro) de la red.

Las intrusiones tipo smurf son una forma de ataque de denegación de servicio distribuido (DDoS), las cuales principalmente producen errores en el funcionamiento de redes

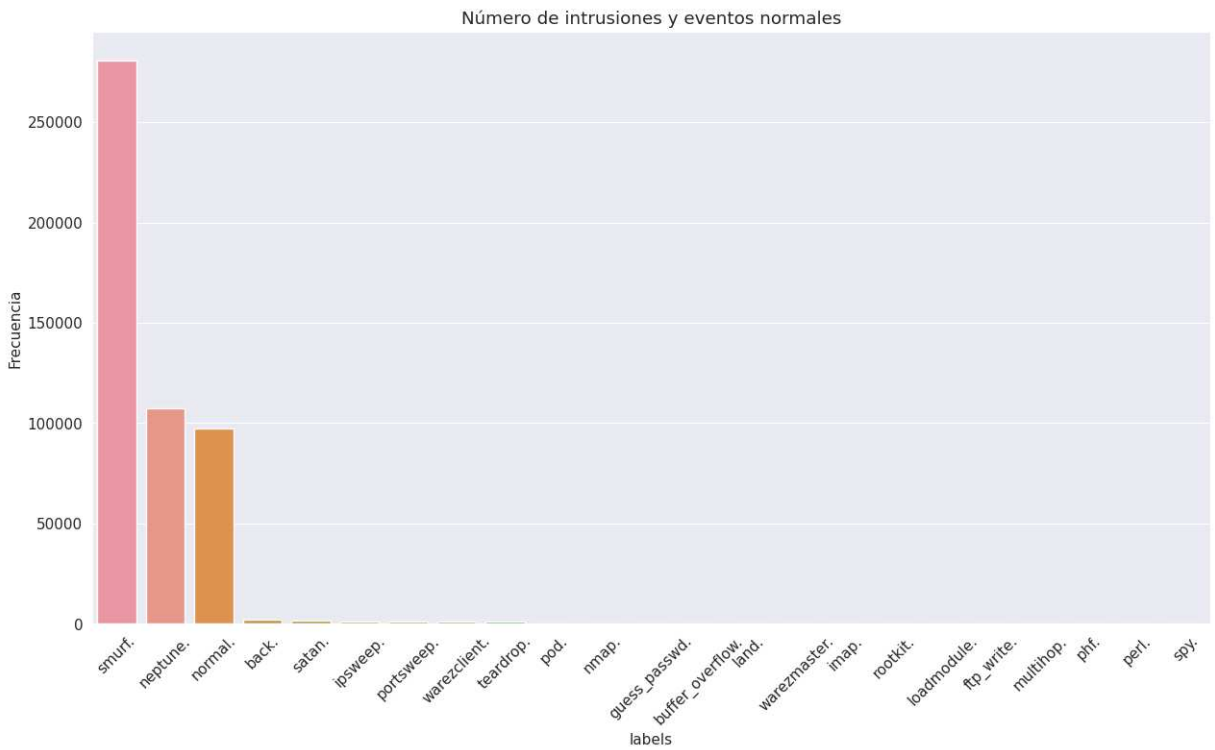


Figura 2.2: Tipos de intrusiones registradas

informáticas tomando las vulnerabilidades de los protocolos IP e ICMP. Un ataque smurf puede ser clasificado como:

- Básico: el ataque realizado es a una sola red empleando paquetes ICMP Echo Request.
- Avanzado: el funcionamiento es tal como el básico con la diferencia de poder atacar a más de un usuario a la vez.

Entre las consecuencias más relevantes de este tipo de intrusiones destacan:

- Pérdida de ingresos;
- Robo de datos;
- Daño a la reputación.

En cuanto a las intrusiones tipo neptune también conocidas como ataques TCP SYN medio abiertos, en [16] se menciona que para que estos ataques sean efectivos se aprovechan las fallas y vulnerabilidades del protocolo TCP, se envían continuamente grandes cantidades de paquetes SYN suplantados directamente a un servidor TCP. El objeto principal de este tipo de intrusiones es rechazar cualquier nueva conexión de un usuario TCP autorizado.

Con lo anterior y dado el enfoque de la presente investigación, interesa únicamente conocer el estado de la red, es decir, si es atacada por algún tipo de intrusión o bien se encuentra en estado normal. Por lo que se fijan a la variable objetivo como el estado de la red y a las etiquetas de clase como *attack* o *normal*, independientemente del tipo de ataque. En la Figura 2.3 se muestra la frecuencia con la que una red presenta algún ataque y cuando se encuentra en estado normal.

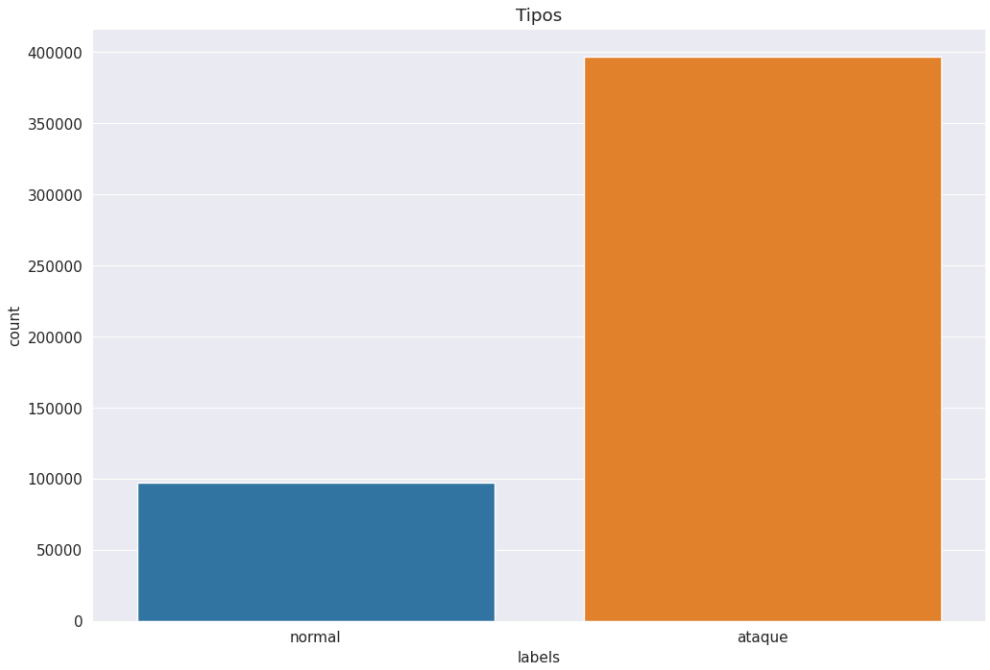


Figura 2.3: Frecuencia de ataques y redes seguras.

Ahora bien, el siguiente paso es realizar una transformación de las etiquetas de clase de categóricas a numéricas, esta modificación será realizada al aplicar la función construida en la Definición 2.3.4 extraída de .

Definición 2.3.4. Sea $\{ataque, normal\}$ el conjunto de clases de la variable objetivo *labels*. Se define la función $\phi : \{ataque, normal\} \to \{0, 1\}$ como

$$\phi(ataque) = 0;$$
$$\phi(normal) = 1.$$

A este proceso se le conoce como *portabilidad* de una variable, específicamente, *binarización* (véase [1, Capítulo 2, Sección 2.2.2.2]).

De esta manera, se obtiene la representación

$$0 \ : \ \text{Red con riesgo de intrusión;}$$
$$1 \ : \ \text{Red segura.}$$

En la Figura 2.4 se muestra la frecuencia de redes con intrusiones y redes seguras. Esta transformación permite establecer que el modelo matemático que será utilizado para determinar la etiqueta de clase 0 o 1 de un nuevo registro debe ser de tipo supervisado y de clasificación, el cuál se plantea en el Capítulo 4.

Otra característica de interés es el tipo de protocolo que se emplea en el funcionamiento de las redes como se mencionó en la sección 1.2, ya que se podría intuir que la elección de este guarda relación con el riesgo de intrusión. En la gráfica 2.5 se observa la frecuencia de los protocolos más comunes en la base: *ICMP*, *TCP* y *UDP*.

El *Protocolo de Mensajes de Control y Error de Internet* (ICMP por sus siglas en inglés) se encarga de informar si se han suscitado incidencias durante la entrega de paquetes o si existen errores en la red en general, sin embargo, no toma medidas al respecto.

El siguiente protocolo en cuanto a frecuencia es el *Protocolo de Control de Transmisión* (TCP) el cual permite que dos o más host se conecten e intercambien datos solicitando confirmación de las partes involucradas para llevar acabo tal acción; además, garantiza la entrega de datos y paquetes.

Por último, el tercer protocolo más utilizado es el *Protocolo de Datagramas de Usuarios*

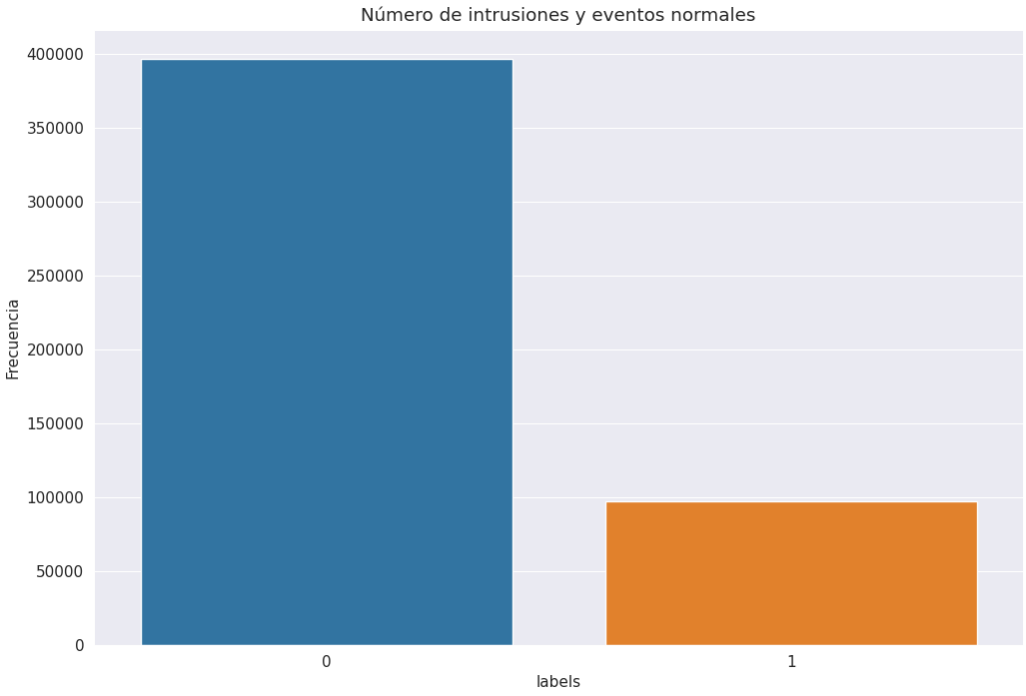


Figura 2.4: Frecuencia de ataques y redes seguras.

(UDP) el cual es simple pues no necesita conexión a una red y posee, como ventaja, la entrega de paquetes sin requerir muchos recursos.

En el Capítulo 3 se hace un análisis detallado de la influencia de los tres protocolos en la variable objetivo, antes, debemos realizar la búsqueda de valores perdidos en cada característica.

Un dato perdido generalmente hace referencia a algún dato faltante o, como su nombre lo dice, es información que se perdió durante la manipulación de la misma. De manera formal se describen como en la siguiente definición.

Definición 2.3.5. Sea $\mathcal{D} = \{x_i^j\}$, $i = 1, ..., n$, $j = 1, ..., d$ un $n \times d$ conjunto multidimensional de datos. Su i-ésimo renglón es $\overline{X_i} = (x_i^1, x_i^2, x_i^3, ..., x_i^d)$, donde x_i^j es el valor de la variable j-ésima para la entrada i-ésima. Un *valor perdido* significa que, para algunos valores i y j, el valor x_i^j no es observado.

En la exploración de la base $\mathcal{D}$ se observa que no tiene valores perdidos o faltantes (Figura 2.6). En el caso de haber encontrado valores de este tipo, es necesario aplicar

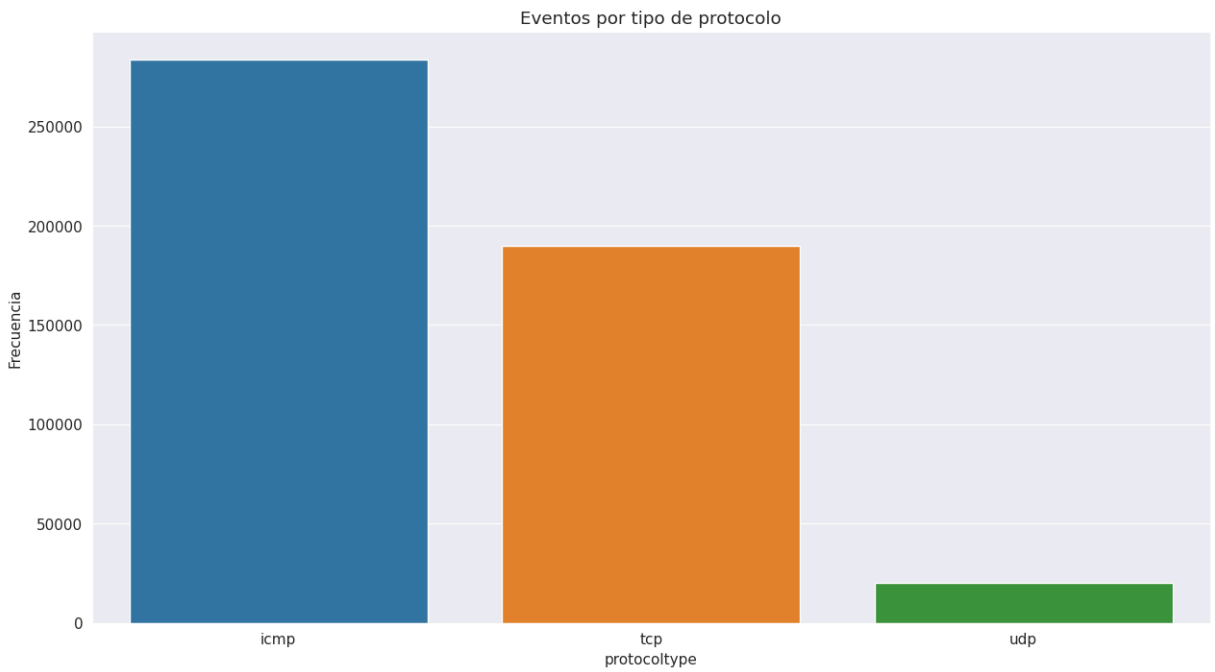


Figura 2.5: Eventos por protocolo.

algún método llamado de imputación. Estos métodos permiten cubrir los valores faltantes con algún valor acorde a la naturaleza de la variable y con relación en otros registros que se tengan. Además, tales métodos son clasificados como simples o múltiples, o bien como determinísticos o estocásticos.

Los métodos de imputación simple consisten en asignar a cada valor faltante un valor obtenido con base en la misma variable o en otros casos con base en otras variables. Un ejemplo es sustituir un dato faltante por alguna medida de tendencia central de los datos conocidos en la misma columna a la que pertenece el valor faltante x_i^j. Otra alternativa es realizar una regresión lineal con los datos en las filas del directorio para determinar los valores faltantes. Algunuos ejemplos de estos algoritmos son la imputación por media, deductiva, Cold Deck, Hot-Deck, regresión, regresión secuencial de imputación múltiple o por máxima verosimilitud. El proceso de imputación múltiple consiste en asignar a cada valor faltante una serie de valores; supongamos m valores distintos para cada elemento perdido x_i^j, al generar m conjuntos de datos completos, cada uno de esos conjuntos es sometido a una estimación de parámetros de interés y posteriormente se combinan los

resultados obtenidos. Un ejemplo de estos métodos es el método de imputación múltiple Markov Chain Monte Carlo.

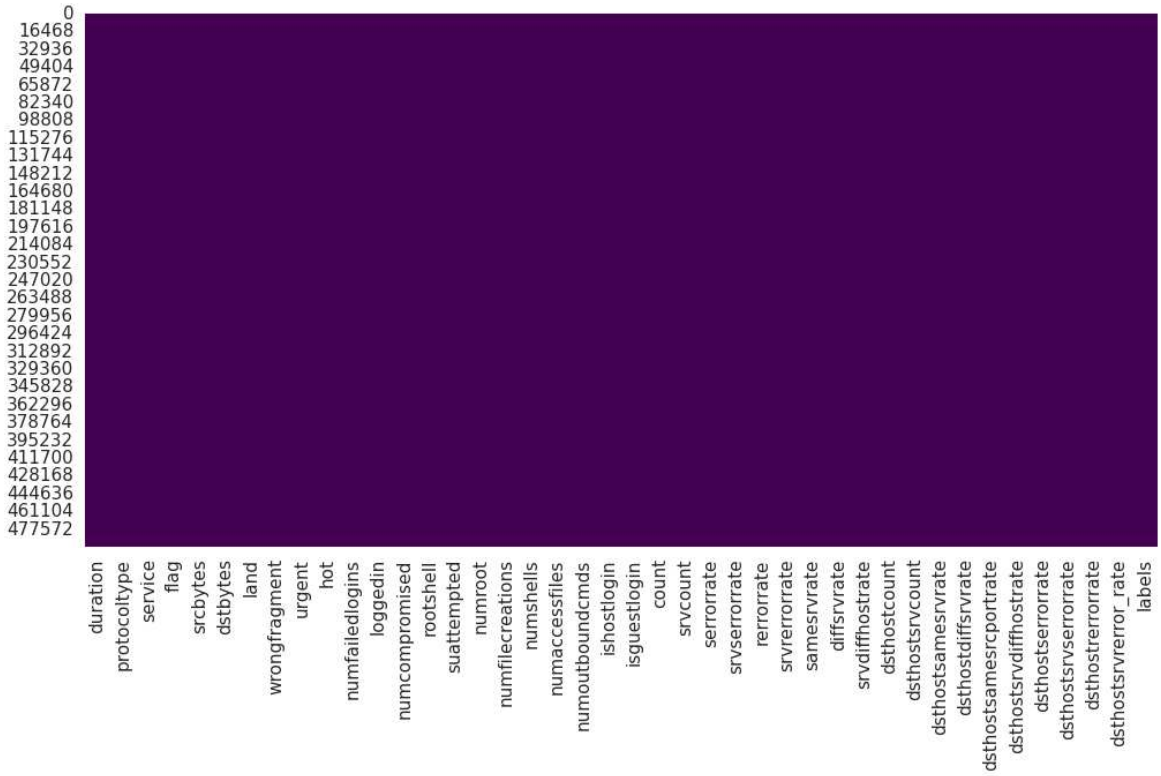


Figura 2.6: Diagrama de calor de datos perdidos. Los datos existentes son mostrados en color morado, en caso de existir datos faltantes, estos aparecen en amarillo.

Es importante aclarar que para saber que métodos pueden aplicarse en caso de la existencia de valores perdidos, es preciso conocer la naturaleza de la variable pues no todos las técnicas son aptas para algunas características.

Con lo anterior, se concluye que la base se encuentra lista para el siguiente proceso: la selección de características relevantes, que se realiza en el capítulo siguiente.

Variable	**Media**	**Desv. est.**	**Mín.**	**Máx.**
duration	47.979302	707.74	0.0	58329.0
srcbytes	3025.610296	988218.1	0.0	693375640.0
dstbytes	868.532425	33040	0.0	5155468.0
land	0.000045	0.006673	0.0	1.0
wrongfragment	0.006433	0.134805	0.0	3.0
urgent	0.000014	0.005510	0.0	3.0
hot	0.034519	0.782103	0.0	30.0
numfailedlogins	0.000152	0.015520	0.0	5.0
loggedin	0.148247	0.355345	0.0	1.0
numcompromised	0.010212	1.798326	0.0	884.0
rootshell	0.000111	0.010551	0.0	1.0
suattempted	0.000036	0.007793	0.0	2.0
...	...	...	...	...
ishostlogin	0.000000	0.000000	0.0	0.0
isguestlogin	0.001387	0.037211	0.0	1.0
count	332.285690	213.147412	0.0	511.0
srvcount	292.906557	246.322817	0.0	511.0
serrorrate	0.176687	0.380717	0.0	1.0
srvserrorrate	0.176609	0.381017	0.0	1.0
rerrorrate	0.057433	0.231623	0.0	1.0
srvrerrorrate	0.057719	0.232147	0.0	1.0
samesrvrate	0.791547	0.388189	0.0	1.0
...	...	...	...	...
dsthostsrvserrorrate	0.176443	0.380919	0.0	1.0
dsthostrerrorrate	0.058118	0.230590	0.0	1.0
dsthostsrvrerror_rate	0.057412	0.230140	0.0	1.0

Tabla 2.3.1: Medidas de tendencia central y de dispersión para las variables cuantitativas.

Capítulo 3

Selección de características

La selección de características es una operación que tiene como objetivo reducir el número de variables de $\mathcal{D}$ mediante filtrados para obtener subconjuntos $\mathcal{D}'$ de $\mathcal{D}$, donde las columnas del nuevo conjunto tienen una alta relación con la variable objetivo *labels*. Este procedimiento es necesario pues proporciona un mejor entendimiento del problema y se eliminan variables irrelevantes con el objeto de investigación, así el científico de datos podrá dar conclusiones más acertadas y enfocarse en observar sólo aquellas que afectan directamente al problema en cuestión; por ello, esta es la finalidad del presente capítulo.

3.1. Tipos de aprendizaje

El aprendizaje automático es una rama de la inteligencia artificial que facilita a los sistemas la suficiencia de aprender a través de la experiencia y por ende mejorar automáticamente. Los sistemas transfiguran los datos con los que disponen en información que pueda ayudar en la toma de decisiones. Los datos permiten que un modelo desarrolle predicciones de forma robusta, por tanto entre mayor sea la cantidad de estos, el desempeño del modelo mejora. Al disponer de los datos, inicia el proceso de aprendizaje del modelo, en el cual se implementa un algoritmo que analiza y explora los datos en busca de patrones no visibles. La manera en que los modelos aprenden puede catalogarse en cuatro formas.

- El *aprendizaje supervisado* se caracteriza por emplear conjuntos de datos con etiquetas para el entrenamiento de algoritmos capaces de clasificar datos o predecir resultados de la manera más precisa posible. Conforme se van introduciendo los datos al modelo, este adapta sus ponderaciones hasta ajustarse adecuadamente a lo requerido, lo cual forma parte del procedimiento de *validación cruzada*.

- El *aprendizaje no supervisado* trabaja con datos que no poseen etiqueta alguna, por tanto no hay etiqueta que predecir. Algoritmos de este tipo son principalmente empleados en los casos donde se necesita analizar un grupo de datos con el objeto de abstraer conocimiento nuevo o bien agrupar entidades bajo cierta afinidad. En estas cuestiones, el algoritmo delimita una métrica de similitud o distancia que sirva para comparar los datos entre sí. Ejemplo de esto son los algoritmos de agrupamiento o clustering.

  Otro escenario donde se emplea este tipo de aprendizaje es en algoritmos de reducción de dimensionalidad o simplificación de conjuntos de datos. Tal como se menciona al inicio de este capítulo, a través de este proceso se busca que los programas disminuyan su tiempo de entrenamiento, mejoren su rendimiento o su representación visual sea más sencilla. Algunos ejemplos son PCA (Principal Component Analysis), t-SNE (T-distributed Stochastic Neighbor Embedding) o ICA (Independent Component Analysis).

- El *aprendizaje semisupervisado* es una alternativa a problemas donde no se dispone de algún conjunto de datos etiquetados, pues posee características de los dos aprendizajes previamente mencionados. Esto es, algunos elementos se etiquetan manualmente, posteriormente se entrenan uno o varios algoritmos de aprendizaje supervisado tomando los datos etiquetados con antelo y luego se emplean los modelos obtenidos para colocar etiquetas a los datos restantes. Por último, se entrena un algoritmo más de aprendizaje supervisado tomando las etiquetas ingresadas manualmente así como las obtenidas de los algoritmos desarrollados anteriormente.

- El último tipo de aprendizaje es el *aprendizaje por refuerzo*. Este es un método

basado en retribuir los comportamientos deseados y sancionar los que no lo son. Bajo este modo, un elemento tiene la capacidad de advertir y descifrar el entorno, efectuar medidas y aprender mediante la experimentación. Más aún, los modelos que se rigen por este tipo de aprendizaje establecen objetivos a lo largo del tiempo para alcanzar una solución óptima. Algunas áreas de aplicación son la robótica, los sistemas de control y la optimización de recursos.

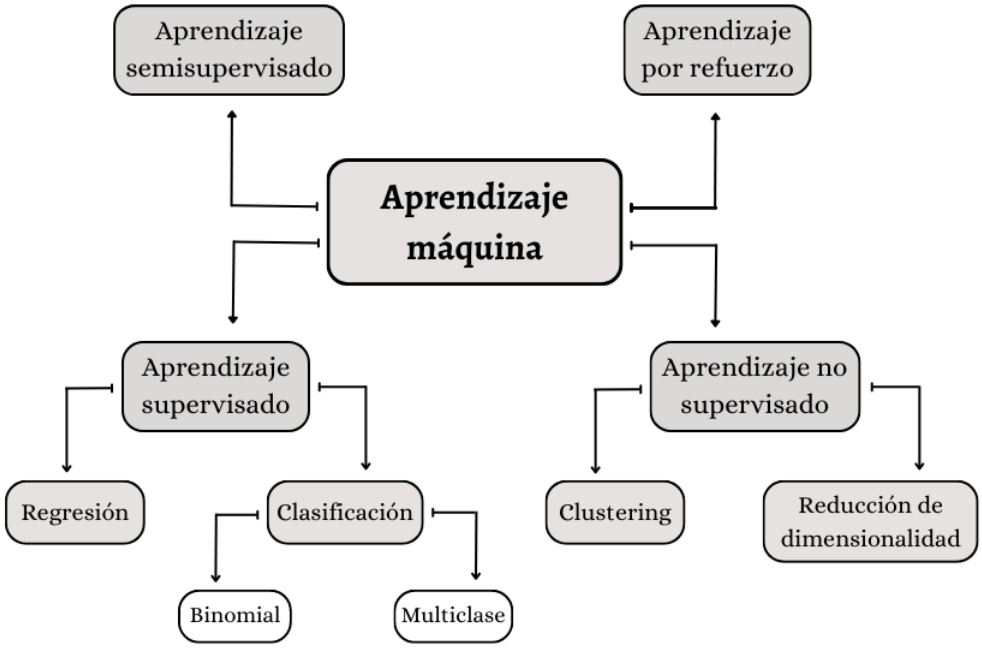


Figura 3.1: Tipos de aprendizaje automático. Imagen extraída y modificada de [9].

3.2. Análisis de correlación

En esta sección se identificarán las relaciones entre las variables de la base y se establecerán hipótesis con respecto a las variables que podrían ser relevantes en la investigación. El proceso se centrará en analizar el grado de asociación entre cada par de variables mediante un valor numérico entre -1 y 1 ([36]). Un estadístico que proporciona esta medida es el coeficiente de correlación de Pearson, pero antes de definirlo considérense las siguientes definiciones obtenidas de [10].

Definición 3.2.1. Dadas dos variables X y Y se dice que están *correlacionadas* si hay una relación sistemática entre ellas, es decir, si el valor de X tiene cierto poder predicativo sobre el valor de Y.

Definición 3.2.2. El *coeficiente de correlación* $r(X, Y)$ es un estadístico que mide la fuerza y la dirección de la relación lineal entre dos variables.

La correlación es una medida de la relación (covariación) lineal entre dos variables cuantitativas continuas (x, y). Es importante resaltar que la covariación entre variables no necesariamente implica causalidad, es decir, la correlación es eventual y el hecho de existir no garantiza que una variable pueda ser consecuencia de la otra o viceversa.

Existen dos tipos de coeficientes de correlación, el muestral y el poblacional. El muestral es un estadístico pues trabaja con información de una muestra, mientras el poblacional es un parámetro dado que se centra en los datos de la población; en esta investigación se calcularán coeficientes muestrales para cada pareja de características.

Como se menciona previamente, el valor del coeficiente de correlación es un valor numérico que toma valores entre -1 y 1. Si el valor es 1, las variables se encuentran altamente correlacionadas de forma positiva, lo que indica que si una de ellas modifica su valor la otra también lo hace en el mismo sentido. Por otra parte, si el valor es -1, la correlación es negativa lo que representa que cuando el valor de una variable se modifica, la otra también cambia pero en sentido contrario. Por último si el valor es 0, las variables no tienen correlación, es decir, son independientes. De manera más específica en la Tabla 3.2.1 se muestra una interpretación de la correlación. Para el análisis de correlación se emplea el coeficiente de Pearson, el cual se define de la manera siguiente:

Definición 3.2.3. Sean X y Y dos variables n-dimensionales con entradas x_j y y_j, $1 \leq j \leq n$, respectivamente, y con sus correspondientes medias $\overline{x}$ y $\overline{y}$. El *coeficiente de correlación de Pearson* se define como

$$r(X, Y) = \frac{\sum_{j=1}^{n}(x_j - \overline{x})(y_j - \overline{y})}{\sqrt{\sum_{j=1}^{n}(x_j - \overline{x})^2}\sqrt{\sum_{j=1}^{n}(y_j - \overline{y})^2}}.$$

El numerador recibe el nombre de *covarianza*, la cual tiene por ecuación

$$Cov(X,Y) = \sum_{j=1}^{n}(x_j - \overline{x})(y_j - \overline{y}).$$

Mientras que el denominador exhibe la desviación estándar de cada variable. Por tanto, al reescribir la ecuación que refleja al coeficiente de Pearson, esta queda como

$$r(X,Y) = \frac{Cov(X,Y)}{\sigma(X)\sigma(Y)}.$$

Significado	**Coeficiente**
Correlación negativa perfecta	-1
Correlación negativa fuerte moderada débil	-0.5
Ninguna correlación	0
Correlación positiva moderada fuerte	0.5
Correlación positiva fuerte	1

Tabla 3.2.1: Grados de correlación.

En la Figura 3.2 se observan los coeficientes de Pearson de cada par de variables de la base de datos. La matriz formada por estos es simétrica, por tanto es suficiente analizar los cuadrantes 1, 2 y 4 comenzando con el cuadrante superior izquierdo y terminando con el inferior derecho en sentido de izquierda a derecha.

En el primer cuadrante (arriba a la izquierda) encontramos en mayor proporción valores cercanos a cero, no obstante, existen algunas relaciones fuertes positivas y negativas. Un ejemplo de esto es la correlación entre la variable *service* y *flag* con un coeficiente de -0.73. Recordemos que *service* hace referencia al tipo de servicio de red en el destino ya sea http, telnet o bien algún otro. En el caso de *flag* nos indica el estado de red, es decir, si se encuentra en estado normal o con error de la conexión. La correlación indica que el incremento de algún servicio de red en particular podría disminuir la frecuencia de errores de la conexión, o bien que la disminución de cierto servicio incremente la posibilidad de

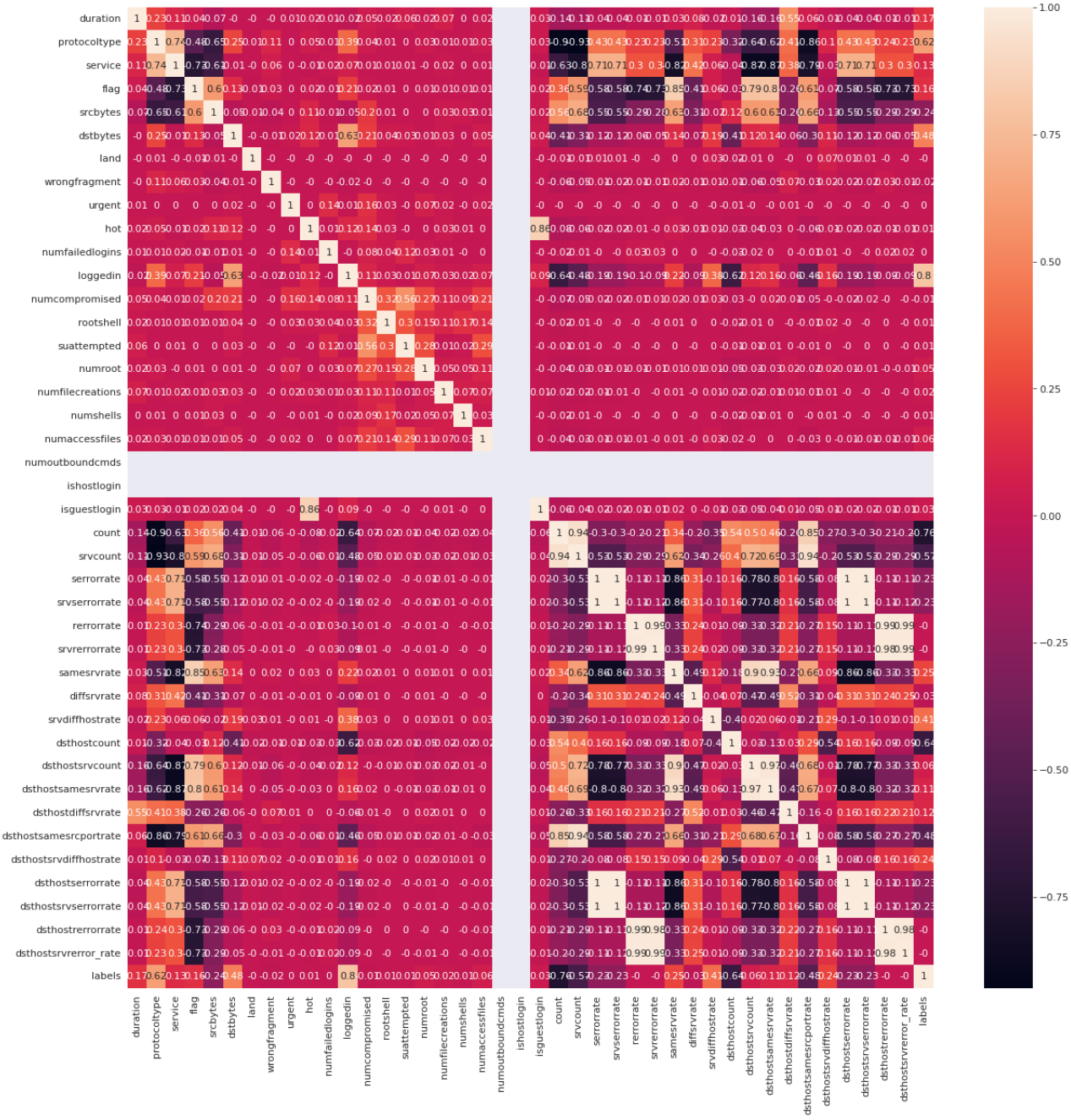


Figura 3.2: Coeficientes de Pearson.

contar con el estado normal de la red, es decir, una red segura. Por otro lado, *protocoltype* y *service* tienen un coeficiente de Pearson de 0.74, lo que indica que el tipo de protocolo y el servicio de red tienen una dependencia ente sí.

El segundo cuadrante (arriba a la derecha) difiere del primero ya que el número de

de correlaciones tanto positivas como negativas es mayor. Se observa que las variables *protocoltype* y *count* están correlacionadas negativamente, así como *flag* con *samesrvrate*, lo que indica que el estado correcto o con error de la conexión se encuentra relacionada con el porcentaje de conexiones al mismo servicio. En el caso de las correlaciones fuertes de tipo positivo se encuentran *hot* con *is_guest_login* y *logged_in* con *labels*, estas correlaciones indican que el inicio correcto de la sesión se encuentra altamente relacionado con la variable objetivo, que indica cuando la red es segura o cuando presenta alguna intrusión.

El cuarto cuadrante (abajo a la derecha) es bastante interesante pues la proporción de correlaciones positivas con respecto a las negativas se equilibran. En correlaciones positivas se encuentran las referentes a las variables *rerro_rate* con *dst_host_rerror_rate* y *srv_count* con *dst_host_same_src_port_rate*. Para el caso de las negativas fuertes *same_srv_rate* con *serror_rate* y *serror_rate* con *dsthostsamesrvrate*. Además, podemos inferir que las variables *same_srv_rate* con la variable *dst_host_same_srv_rate* también se encontrarán correlacionadas pero de forma positiva. Al recurrir al diagrama de calor, se puede observar que efectivamente se encuentran con una alta correlación positiva, la cual es inclusive mayor a la correlación negativa que presentaban con respecto a *serror_rate*.

En ciertas investigaciones, cuando existe una alta correlación entre dos variables es posible eliminar alguna de ellas (véase [30, Sección 9.2.4, pag. 277]), sin embargo, en esta investigación utilizaremos una técnica alternativa para seleccionar las variables significativas, denominada *Select K best features*, que se aborda en la sección 3.4, pero antes, en la sección siguiente se preparan las variables categóricas.

3.3. Codificación de variables

Antes de aplicar algún algoritmo de minería, es preciso codificar a las variables categóricas para posteriormente procesarlas (véase [1, Sección 2.2.2.2, pág. 31]). La importancia de este proceso deriva del hecho que en la base existen variables categóricas, es decir variables no numéricas las cuales tienen un número limitado de clases o bien categorías. Es posible agrupar a las variables categóricas en dos grupos:

- Variables ordinales: en este tipo de características puede establecerse un orden jerárquico con base en el objeto que representa tal característica.

- Variables nominales: no es posible establecer jerarquía alguna entre las clases de la variable.

Entre las técnicas más empleadas en la codificación de variables categóricas, se encuentran la *codificación ordinal* y la *codificación One Hot Encoding*.

El primer método consiste simplemente en sustituir cada valor de la variable con un número entero diferente. Este tipo de codificación es de mucha utilidad cuando se tienen datos ordinales pues establecer una jerarquía entre las clases permite manipular de mejor manera a la variable sin incrementar la dimensión del directorio.

El segundo método funciona de la siguiente manera; si un atributo categórico contiene m diferentes valores, entonces se crean m distintas variables binarias. Cada nueva variable binaria corresponde a solo un valor del atributo categórico. Luego, cada registro contendrá un 1 en la columna binaria correspondiente y en el resto contendrá ceros.

En este caso, se portabilizan como variables numéricas, pero de tal forma que se preserve la respectiva categoría. Por lo cual, se realiza un proceso de binarización *One Hot Encoding*, implementada en la librería *pandas* de *Python* mediante la instrucción *get_ dummies*.

Por ejemplo, consideremos la variable *protocoltype*, la correspondiente columna contiene tres valores: icmp, tcp y udp (Tabla 3.3.1). Al codificar se generan tres nuevas columnas: *protocol_ icmp*, *protocol_ tcp* y *protocol_ udp* y se elimina la original. Por lo tanto, las filas que contenían el valor icmp en la columna original ahora contienen un valor 1 en la columna *protocol_ icmp* y ceros en las otras dos, ocurre lo mismo con aquellos registros que contenían el valor tcp o udp. En la Tabla 3.3.2 se muestra el resultado de la codificación.

Esta misma codificación se le realiza a las variables *service* y *flag*. Cabe resaltar que este tipo de codificación aumenta la eficiencia de los métodos de minería, sin embargo, genera un aumento de la dimensión, obteniendo así 116 columnas de las 42 originales, este nuevo valor lo denotaremos por d'. De hecho, existen codificaciones que evitan el aumento

Registro	protocoltype
1	udp
2	tcp
3	icmp
4	tcp
5	udp

Tabla 3.3.1: Ejemplo de registros de la variable *protocoltype*.

Registro	protocol_icmp	protocol_tcp	protocol_udp
1	0	0	1
2	0	1	0
3	1	0	0
4	0	1	0
5	0	0	1

Tabla 3.3.2: One Hot Encoding de la variable *protocoltype*.

de la dimensionalidad, tal como la codificación entera, donde a cada valor se le asigna un número entero. En esta investigación, la codificación One Hot Encoding resultó ser la ideal dada la naturaleza del problema pues permite reducir significativamente el número de características en la sección siguiente.

3.4. Extracción de características

La selección de características es la primera etapa en el proceso de clasificación, esto debido a que las características irrelevantes generalmente dañan la precisión del modelo tal como se muestra en [1, 21, 30]; además el uso de variables no relevantes resulta ser una fuente de ineficiencia computacional.

Hay varios métodos de selección de características que se utilizan para identificar las variables más relevantes para el modelo. Algunos de los métodos más comunes se clasifican

con base en su funcionamiento como se sigue:

- **Selección de características basada en filtrado:** los métodos de este tipo basan la selección de características en la evaluación de alguna medida de relevancia, tal es el caso de la correlación, la varianza, entre otras.
- **Selección de características basada en envoltura:** para este caso, los algoritmos evalúan el rendimiento del modelo empleando diferentes subconjuntos de características. Su objetivo es encontrar algún subconjunto que maximice el rendimiento del modelo. Para problemas de alta dimensionalidad, estos algoritmos deben buscar entre todos los elementos del conjunto potencia del grupo de características.
- **Selección de características basada en incrustación:** estos métodos incorporan la selección de características directamente en el proceso de aprendizaje del modelo. Es decir, la selección de características se realiza simultáneamente con el ajuste del modelo.
- **Métodos de regularización:** los métodos de este tipo penalizan los coeficientes de las variables que no son relevantes para el modelo, lo que a su vez reduce su importancia en el modelo final.

Por lo tanto, el objetivo de los algoritmos de selección de características es elegir aquellas más informativas con respecto a la etiqueta de clase (en nuestro caso, susceptible a ataques y segura o 0 y 1), con lo que se logra, además, una reducción de dimensionalidad. Esto es, se desea calcular un mapeo puntual $\mathbf{F} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^q$ de un espacio altamente dimensional $\mathbb{R}^{d'}$ a un espacio de dimensión menor $\mathbb{R}^q$, con el cual se logre mapear a cada vector fila $\overline{X_i} \in \mathcal{D}$ d'-dimensional, $i = 1, ..., n$, en un vector $\overline{X_i}'$ de dimensión q y que preserve la información intrínseca del vector original; específicamente, para cada $\overline{X}_i \in \mathbb{R}^{d'}$, $1 \leq i \leq n$, se calcula un embebimiento de dimensión menor $\overline{X_i}' \in \mathbb{R}^q$. La nueva base de dimensión $n \times q$ es llamada *variedad* y el objetivo es encontrar una variedad que pierda la menor información topológica posible y, por lo tanto, que preserve la estructura en baja dimensión de los patrones de alta dimensión; tal estructura se puede definir de varios tipos

de formas, por ejemplo, geométrica y semánticamente, en esta investigación se utilizará un método que mantiene vecindades y preserva la información topológica, denominado *Select K Best Features* (selección de las K mejores características) y abreviado como *SelectKBest*. Este es un modelo de filtrado que elige las K características con varianza más alta y elimina el resto. De forma predeterminada, este algoritmo elimina todas las características de varianza cero, en otras palabras, las características que tienen el mismo valor en todas las muestras, lo que nos demuestra de forma intuitiva que no tienen efecto en la variable de etiquetas *labels*.

El modelo Select K Best utiliza una prueba de hipótesis basada en la correlación de Pearson y el valor p para evaluar la relación entre cada característica y la variable objetivo. En particular, se aplica la prueba *t de Student* para evaluar si la correlación entre la característica y la variable objetivo es estadísticamente significativa. Esta prueba también es utilizada para comparar la media de dos grupos y evaluar si son estadísticamente diferentes. Bajo el contexto del modelo, se emplea en la evaluación de correlación de las características con la variable objetivo con respecto al cero. El valor p obtenido a partir de la prueba t se utiliza para determinar si se rechaza o no la hipótesis nula de que no hay relación lineal entre la característica y la variable objetivo.

En resumen, el modelo Select K Best utiliza la prueba t de Student para evaluar la significancia estadística de la correlación entre cada característica y la variable objetivo. Esto se utiliza para seleccionar las K mejores características que tienen la relación más fuerte con la variable objetivo y descartar las demás características.

Describiremos el funcionamiento del modelo *SelectKBest* de manera formal. En [25, Def 4.1] presentan la selección de características como sigue.

Definición 3.4.1. Para un conjunto dado $\mathcal{D}^{(1:d')}$ de d' variables independientes e idénticamente distribuidas, un *método de selección* es un mapeo

$$\Phi(\mathcal{D}^{(1:d')}) : \mathcal{Z}^{d'} \mapsto 2^{V_{d'}},$$

donde $\mathcal{Z}^{d'} = (X1, X_2, \ldots, X_{d'}, Y)$ es el conjunto de los d' vectores de características y la

variable objetivo y $2^{V_{d'}} = \{S : S \subseteq V_{d'}\}$ es el conjunto potencia de $V_{d'} = \{1, 2, \ldots, d'\}$.

Una de las tareas de la selección de variables es determinar el *conjunto objetivo*, esto es, que el método sea consistente en el sentido de la definición dada a continuación.

Definición 3.4.2. Un algoritmo de selección $\Phi(\mathcal{D}^{(1:d')})$ es *consistente* con respecto a un conjunto de características si

$$\Phi(\mathcal{D}^{(1:d')}) \to S.$$

Esto es, converge al conjunto S. El conjunto S es llamado el *conjunto objetivo* de Φ.

Continuando con el proceso, es importante considerar la existencia de variables discretas condicionalmente dependientes con la variable objetivo, con base en las definiciones propuestas en [25, Pág.17] y adaptadas a nuestro contexto, se tienen las siguientes definiciones.

Definición 3.4.3. (Independencia) Dos variables discretas $\overline{X_i}$ y Y se dicen *independientes* si satisfacen,

$$\forall\ x, y,\ p(y, x_i) = p(y)p(x_i)$$

Esto se denota como $Y \perp \overline{X_i}$

Definición 3.4.4. (Independencia condicional) Dos variables $\overline{X_i}$ y Y se dicen *independientemente condicionales* si al considerar $Z = z$ satisfacen

$$\forall\ x, y,\ p(y, x_i\ |Z = z) = p(y\ |Z = z)p(x_i\ |Z = z)$$

Si lo anterior se satisface para todo valor de z,

$$\forall\ x, y, z,\ p(y, x_i\ |z) = p(y\ |z)p(x_i\ |z)$$

se dice que Y es independientemente condicional a $\overline{X_i}$ dado Z y se denota como $Y \perp \overline{X_i}|Z$

Para variables continuas, las definiciones anteriores son análogas.

Es deseable que el conjunto S esté compuesto de las características relevantes, las cuales son definidas como sigue (véase [25, Pág. 63]).

Definición 3.4.5. (Relevancia) Sean X el vector de características y, para

$$S \subseteq \{1, \ldots, d'\},$$

el sub-vector $X_{i \in S}$ de X. Una característica X_i es *relevante* a Y si y sólo si

$$\exists\, S \in V_{d'} : Y \not\perp X_i | X_S,$$

esto es, Y es dependientemente condicional de X_i dado X_S. Una característica X_i es *irrelevante* a Y si y sólo si no es relevante a Y. El conjunto de todas las características relevantes es denotado por S^A.

Para este trabajo, consideramos el caso univariado, en otras palabras, se desea determinar el conjunto S^A el cual es marginalmente dependiente de Y, cuyas componentes son las características X_i que satisfacen $Y \not\perp X_i|\emptyset$, o equivalentemente $P(Y|X_i) \neq P(Y)$.

Como se mencionó anteriormente, el método está basado en las pruebas estadísticas de hipótesis, estas pruebas miden la dependencia de características individuales X_i con la variable objetivo *labels* denotada como Y, y por lo tanto se enfocan sólo en la distribución marginal $f(x_i, y)$ (véase [25, Sec 4.1.1]). Esta prueba identifica a las características con dependencia significativa.

Ahora, la prueba de hipótesis que se desarrolla entre las características y la variable objetivo supone las siguientes hipótesis:

- La hipótesis nula H_0^i afirma que la característica X_i es irrelevante a (o independiente de) Y.
- La hipótesis alternativa H_1^i afirma que X_i es relevante a (o dependiente de) Y.

Claramente, cada característica cumple sólo una de las dos hipótesis. Para concluir alguna de las dos opciones se realiza una prueba estadística, la cual puede definirse como la

función $\phi_i : \mathcal{Z}^{d'} \mapsto \{0, 1\}$, para cada observación X_i, la prueba decide que H_0^i es cierta siempre y cuando $\phi_i(X_i, Y) = 0$; y H_1^i se elige cuando $\phi_i(X_i, Y) = 1$. La decisión obtenida puede ser o no ser correcta debido a los dos tipos de errores que se pueden cometer (véase [25, pág. 72]), los cuales logran medirse a través de dos tasas de error:

Tasa de error tipo I: $P(\phi_i = 1|H_0^i)$ mide la frecuencia con la que la característica X_i se clasifica como relevante cuando en realidad no lo es.

Tasa de error tipo II: $P(\phi_i = 0|H_1^i)$ mide la frecuencia con la que la característicaX_i se clasifica como irrelevante cuando en realidad es relevante.

La tasa del error tipo I es también llamada tasa de falso positivo y la de tipo II es la tasa de falso negativo.

Ahora, consideremos al parámetro α denominado como el nivel de la prueba. Este valor delimita la probabilidad de que falsamente la hipótesis nula sea rechazada, es decir, mide la tasa de error tipo I, esto es:

$$P(\phi_i = 1|H_0^i) \leq \alpha.$$

Cuando lo anterior se satisface, se dice que ϕ_i tiene una prueba de nivel (o de medida) α.

De un conjunto de pruebas de hipótesis ϕ_i, $i = 1, \ldots, d'$, en [25, pág. 72] se redefine el método de selección de características como

$$\Phi(D^{(1:d')}) = \{i : \phi_i = 1\}.$$

De esta manera, Φ depende sólo de las distribuciones marginales $f(x_i, y)$.

Ahora, con base en el coeficiente de correlación de Pearson, se tiene la prueba *z de Fisher*. Esta prueba calcula el estadístico

$$t = \frac{1}{2} \ln \frac{1 + r(X_i, Y)}{1 - r(X_i, Y)},$$

el cual es asintóticamente ajustado a una curva gaussiana, es decir, su distribución es

$$f(t) = N(\tanh^{-1} r(X_i, Y), (d' - 3)^{-1}),$$

para cualquier $f(x_i, y)$, esto debido al Teorema del Límite Central (véase [13]). Por lo tanto, la prueba es asintóticamente correcta.

Para nuestro caso, consideremos el conjunto de etiquetas como $\mathcal{Y} = \{+1, -1\}$, esto como resultado de redefinir los valores de las etiquetas, de tal manera que la etiqueta 0 ahora la consideraremos como -1. De esta forma, comprobar la hipótesis nula

$$H_0^i : Y \perp X_i | \emptyset$$

es equivalente a comprobar si las distribuciones condicionales $f(x_i|Y = +1)$ y $f(x_i|Y = -1)$ son distintas. La prueba más usual en este caso es la prueba *t de Student*, la cual es exacta en muestras pequeñas y determina si $f(x_i|Y = +1)$ y $f(x_i|Y = -1)$ son Gaussianas con igual varianza. Más aún, la prueba es de dos colas por la equivalencia de H_0^i dada previamente y del hecho que la distribución es simétrica alrededor del 0. Para alta dimensionalidad, como es nuestro caso, surge el siguiente *problema de testeo múltiple*. Supongamos que el método de selección de características realiza n pruebas $\phi_1, \ldots, \phi_n$, cada uno con un nivel α. Por lo tanto, se tiene que el número esperado de errores es

$$\mathbb{E}[\Phi(D^{(1:d')}) \cap \{i : H_0^i\}] = \mathbb{E}\left[\sum_{i:H_0^i} \phi_i\right].$$

Si suponemos en el peor de los casos que todos los H_0^i son ciertos, el valor esperado es igual a $n\alpha$, el cual puede ser relativamente grande, por ejemplo, si $n = 10^4$ y $\alpha = 0.05$, se tiene un error de 500, lo mismo se tiene aún si se considera α muy pequeño.

Para compensar el problema de multiplicidad, se controla la tasa de falsos positivos de cada una de las pruebas mediante el concepto de valor p.

Definición 3.4.6 (Valor p.)**.** Un valor p para una hipótesis nula dada H_0 es una variable

aleatoria $p \in [0, 1]$ que satisface

$$P(p \leq \alpha | H_0) \leq \alpha,$$

para toda $\alpha \in [0, 1]$.

El valor p puede interpretarse como el más bajo nivel para el cual la hipótesis nula podría ser rechazada (véase [25, pág. 77]). Por lo tanto, el valor p mide la confianza en el rechazo. De hecho, en la literatura ([23]) se demuestra que para n pruebas independientes de nivel α, se tiene que

$$P\left(\exists i : p_i \leq \alpha | H_0^{(1:n)}\right) \leq \alpha,$$

con p_i los valores p de cada variable X_i. Lo que nos lleva a que las k variables más relevantes con la variable Y son aquellas con los valores p_i más pequeños. Por ejemplo, para la variables $X_i = dsthostcount$ y $Y = labels$, , con $\alpha = 0.05$, el valor de $t = 6,357,877.47$ y el valor de p es relativamente cercano a cero.

De lo anterior el procedimiento a desarrollar queda como:

1. **Establecer las hipótesis:**

   H_0^i: la característica X_i es irrelevante a Y.

   H_1^i: la característica X_i es relevante a Y.

2. **Determinar el nivel de significancia (α):**

   Usualmente se utiliza un valor de $\alpha = 0.05$ [7], aunque puede cambiar con base en el problema a considerar.

3. **Calcular el estadístico de prueba t**.

4. **Calcular el valor** p.

5. **Interpretar los resultados:** Si el valor p es menor que el nivel de significancia, se rechaza la hipótesis nula en caso contrario se rechaza la hipótesis alternativa.

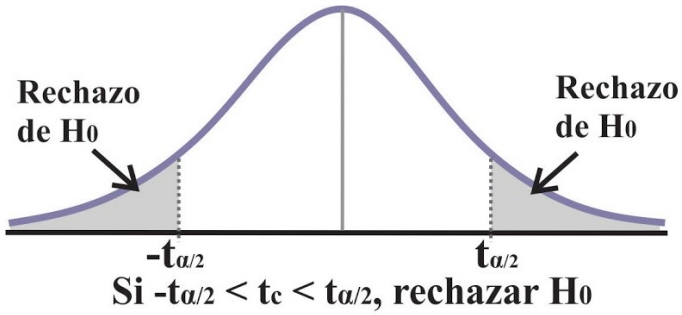


Figura 3.3: Prueba *t de Student* de dos colas. Obtenida de [11]

El proceso de la prueba de forma gráfica se ilustra en 3.3.

Con el objetivo de visualizar de mejor forma los valores p más pequeños, en la Figura 3.4 se muestran los valores de $-\log_{10}(p_i)$, donde estos son aquellos correspondientes a las líneas más altas, es decir las que se encuentran notablemente más grandes.

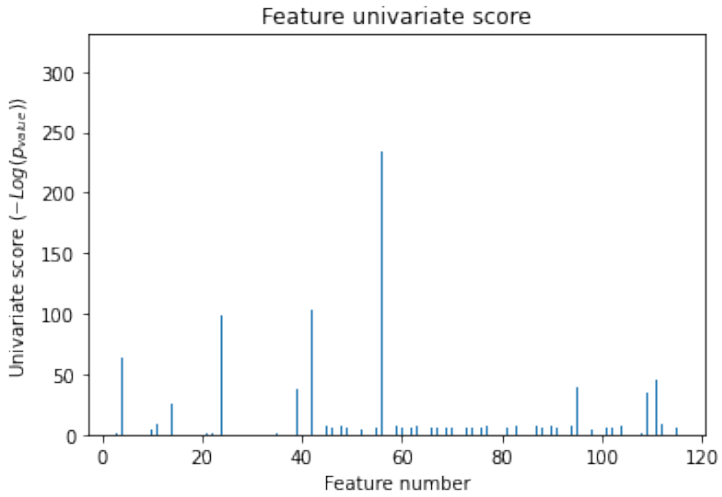


Figura 3.4: Diagrama de $-\log_{10}(p_i)$, $1 \leq i \leq d' - 1$.

De esta manera, al efectuarse el proceso de SelecKBets, las características de mayor relevancia con respecto a la variable objetivo son:

- *dstbytes*: representa el número de bytes de datos desde el destino hasta el origen.
- *loggedin*: 1 si inició sesión correctamente; 0 de lo contrario.
- *count*: número de conexiones al host en los últimos dos segundos.

- *srvcount*: número de conexiones al servicio en los últimos dos segundos.
- *dsthostcount*: número de conexiones que tienen el mismo host de destino.
- *dsthostsamesrcportrate*: porcentaje de conexiones al host actual que tienen el mismo puerto src.
- *protocol_icmp*: variable binaria alusiva al protocolo ICMP; 1 indica que la red cuenta con tal protocolo, 0 en caso contrario.
- *protocol_tcp*: variable binaria referente al protocolo TCP; 1 en caso de contar con tal, 0 en caso contrario.
- *service_ecr_i*: variable binaria; 1 si se accedió a la red con servicio ecr_i, 0 en caso contrario.
- *service_http*: variable binaria; 1 si se accedió a la red con el servicio http, 0 en otro caso.

Este método nos arroja una reducción de 116 columnas a sólo 10, las cuales tienen un mayor efecto en la variable *labels*; ahora, la nueva base será la formada por las columnas $S^A \cup \{labels\}$ y la denotaremos por $\mathcal{D}'$, por lo que la dimensión de $\mathcal{D}'$ será $d'' = 11$.

Finalmente, se concluye que una de las características que permanecieron tiene relación directa con la seguridad en redes, con lo que comprobamos las intuiciones mencionadas en la sección 2.3: el tipo de protocolo sí guarda relación con la variable objetivo.

Capítulo 4

Modelo clasificador

En Ciencia de Datos, los problemas comunes de aprendizaje automático supervisado se agrupan en dos tipos:

Regresión: Los problemas de este tipo tienen en particular el hecho que la variable de respuesta Y es cuantitativa y además se encuentra determinada por las entradas X del modelo, las cuales cuando presentan dependencia temporal se les llama *de predicción de series temporales* o bien *forecasting*. El objetivo de estos modelos es predecir el valor de Y a partir de los de X conocidos. Los valores predichos encontrados dentro del conjunto de registros de la base empleados en el ajuste del modelo se conocen como *interpolaciones*, en caso contrario son *extrapolaciones*. Entre mayor sea la distancia de las extrapolaciones con respecto al grupo de datos, existe más posibilidades de fallo, pues estas se basan en gran manera de supuestos.

Clasificación: Estos problemas poseen la variable objetivo Y de tipo cualitativa o categórica. El proceso de predecir la respuesta de una observación es llamado *clasificar*, esto es, determinar la categoría o etiqueta de clase. En múltiples ocasiones estos métodos calculan la probabilidad de que un registro introducido pertenezca o no a alguna de las categorías posibles en el problema.

Debido a la naturaleza del problema, el modelo matemático para resolverlo cae dentro

del segundo tipo. En este capítulo se plantea el modelo, se entrena, valida y se obtienen algunas simulaciones.

4.1. Tipos de clasificación

La clasificación de datos permite identificarlos y agruparlos asignando algún valor con base en los objetivos del problema que se desea resolver. De esta manera, algunas técnicas son las que siguen.

Clasificación binaria: en este tipo de clasificación, como su nombre lo dice, solo se dispone de dos clases diferentes, usualmente etiquetadas con 0 y 1. Algunos algoritmos que emplean esta clasificación son:

- Regresión logística.
- Árboles de decisión.
- K-NN (K-nearest neighbors).
- Bates ingenuos.
- SVMs (Support Vector Machines).

Clasificación multi-clase: estas clasificaciones se caracterizan por el hecho de contar con más de dos posibles etiquetas. Entre los algoritmos comúnmente empleados en clasificación multiclase destacan:

- K-NN.
- Árboles de decisión.
- Bosque aleatorio.
- Naïve-Bayes.
- Impulso por gradiente.

Clasificación ordinal: también es conocida como *regresión ordinal*, es un problema de reconocimiento de patrones situada entre regresión y clasificación nominal. En este caso, es posible encontrar alguna relación de orden entre el número finito de categorías que tiene la variable objetivo.

Clasificación multi-etiqueta: en este tipo de clasificación se permite que cada elemento del conjunto de datos se asocie a una o más etiquetas, el objetivo puede entenderse como predecir las propiedades de un elemento del conjunto que no son mutuamente excluyentes. Entre los métodos empleados, destacan:

- Árboles de decisión multi-etiqueta.
- Bosques aleatorios multi-etiqueta.
- Multi-etiqueta Gradient Boosting.

Debido a que el objetivo principal de esta investigación es plantear un modelo matemático de aprendizaje máquina capaz de clasificar a un nuevo registro con datos de una red en alguno de los dos tipos: en riesgo de intrusión o segura, resulta ser un problema de clasificación binaria. En la sección 4.2 se plantea formalmente el problema de clasificación en ciencia de datos para, posteriormente, en la sección 4.3, exhibir el modelo matemático, entrenarlo y, finalmente, realizar algunas simulaciones del funcionamiento. Con ello, se alcanza el objetivo general.

4.2. El problema de clasificación

Dado que la variable objetivo representa el valor de seguridad de una red, el cual es un dato binario referido como la etiqueta de clase, entonces se requiere diseñar y simular un modelo de aprendizaje máquina supervisado de tal forma que aprenda a detectar las relaciones entre los nuevos registros con su respectiva etiqueta directamente de las características intrínsecas de aquellos conocidos, esto es, de un subconjunto de la base $\mathcal{D}'$. A este proceso se le conoce como *entrenamiento* y los datos que son utilizados para aprender

estas correspondencias se denominan *de entrenamiento* y se denotan por $\mathcal{D}'_{Train}$. Una vez entrenado, el modelo se valida con los datos contenidos en el complemento del conjunto de entrenamiento, pero sin etiquetas, llamado *de prueba* y denotado por $\mathcal{D}'_{Test}$. Tal validación se realiza mediante el cálculo de algunas métricas de desempeño, como el *accuracy* que es el cociente entre las etiquetas acertadas de $\mathcal{D}'_{Test}$ entre el número total. Finalmente, el modelo se utiliza para determinar las etiquetas de clase estimadas para registros nuevos introducidos, es decir, en estos hará falta la etiqueta. Todo el procedimiento anterior se enmarca en la minería de datos y se le denomina *el problema de clasificación*. Cada uno de los procesos previamente mencionados son abordados con mayor detalle en secciones posteriores.

En [1, Capítulo 20, pág. 285] el autor define este problema. La siguiente definición es una adaptación de nuestra autoría.

Definición 4.2.1 (**Problema de clasificación de datos**)**.** Dada una $m \times (d'' - 1)$ matriz de datos de entrenamiento $\mathcal{D}'_{Train}$, con $m < n$ vectores fila $(d'' - 1)$-dimensionales de la matriz de datos $\mathcal{D}'$, y el conjunto de etiquetas de clase, formado por elementos de $\{0, 1\}$, asociados con cada uno de los m renglones en $\mathcal{D}'_{Train}$, crear un modelo de entrenamiento $\mathcal{M}$, el cual pueda ser utilizado para predecir la etiqueta de clase de una entrada $(d'' - 1)$-dimensional $\bar{Y} \notin \mathcal{D}'$.

En nuestro contexto, el modelo debe aprender la estructura del conjunto de datos de entrenamiento, es decir, de $\mathcal{D}'_{Train}$, mediante el reconocimiento de los dos grupos de registros "similares" con base en las etiquetas correspondientes, esto es, realiza dos clasificaciones. Una vez realizado el proceso de aprendizaje del modelo, este se denomina *modelo entrenado*. Posterior a esto, se introduce el conjunto de datos de prueba $\mathcal{D}'_{Test}$ al modelo entrenado, los cuales recordemos permanecen sin etiquetas y, más aún, el programa no los ha visto con anterioridad. Cada uno de los registros en $\mathcal{D}'_{Test}$ se denominan como *registro de prueba* pues su etiqueta de clase se desconoce. La tarea del modelo es clasificarlos en dos grupos, lo que es, calcular su etiqueta.

Por tanto, para nuestro caso, el problema de clasificación es de *aprendizaje supervisado*

ya que utiliza un conjunto de datos etiquetados en la etapa de aprendizaje. Los datos de entrenamiento son fundamentales para proporcionar orientación sobre cómo se definen los grupos.

En resumen y de manera intuitiva, el problema de clasificación puede formularse como:

Dado un conjunto de entradas de entrenamiento, cada una de las cuales están asociadas con una etiqueta de clase, determinar la etiqueta de clase de una o más instancias de prueba nunca antes vistas.

De lo anterior, el algoritmo de clasificación que se diseña para la resolución del problema cuenta con dos fases:

Fase de entrenamiento: se toma un modelo de aprendizaje automático, el cual es modificado tomando como referencia las instancias de entrenamiento en D'_{Train} previamente definidas y, con base en estas, se ajustan los parámetros a través de los datos etiquetados.

Fase de prueba: en esta fase, el modelo resultante del entrenamiento se usa para determinar la etiqueta de clase de cada una de las instancias de prueba en $\mathcal{D}'_{Test}$, a las cuales, como se menciona en páginas previas, este no ha tenido acceso, siendo de este modo desconocidas para el programa. Posteriormente, se recolectan tanto el número de aciertos como el número de fallos, con tales datos se puede calcular alguna o algunas métricas de desempeño del modelo que brindan un panorama de la eficiencia del mismo.

La salida del algoritmo de clasificación es la predicción de etiquetas; es decir, mostrará una etiqueta 0 o 1 para cada nuevo vector fila ingresado, la cual recordemos representa el valor de seguridad de la red.

Es importante recalcar que cuanto mayor sea la cantidad de datos con la que se dispone para entrenar los modelos de clasificación, estos tendrán mayor precisión; en caso contrario el rendimiento puede ser deficiente, esto es, el modelo puede describir las características aleatorias específicas del conjunto de entrenamiento, pero no será capaz de generalizar la estructura de grupo de instancias de prueba; en otras palabras, podría predecir con

precisión las etiquetas de las instancias utilizadas para construirlos, pero funcionar mal en bases desconocidas. Este fenómeno se conoce como *sobreajuste* y, para evitarlo, en este caso se utilizarán $\frac{2}{3}$ partes de las filas, es decir, de los registros de la base $\mathcal{D}'$ como conjunto de entrenamiento y el resto para la fase de prueba.

4.3. Planteamiento, entrenamiento y validación del modelo

Como se hizo mención en la sección 4.1, existen varios algoritmos de clasificación binaria. Para la presente investigación se optó por trabajar con el algoritmo de K- vecindades más cercanas (llamado KNN por sus siglas en inglés *k-nearest neighbors algorithm*). La razón de tal elección son las características que presenta, así como su eficiencia y simplicidad.

En estadística, este algoritmo es un método de aprendizaje supervisado no paramétrico, esto es, que su distribución de probabilidad no depende de parámetros como la media y varianza. Fue desarrollado por Evelyn Fix y Joseph Hodges en 1951 ([14]) y más tarde expandido por Thomas Cover ([6]). Es utilizado en resolución de problemas de clasificación y regresión, como se menciona al principio del capitulo, en nuestro caso se utiliza para clasificación binaria.

En clasificación, la salida de KNN es una etiqueta de clase. Un objeto se clasifica por una mayoría de "votos" de sus vecinos y el objeto se asigna a la clase más común entre sus k vecinos más cercanos, donde k es un número entero positivo, regularmente pequeño. Si $k = 1$, entonces el objeto simplemente se asigna a la clase de ese vecino más cercano.

La idea base del funcionamiento de este tipo de modelos aplicados a problemas de clasificación consiste en tomar el punto objetivo q y asignarle la misma etiqueta de clase que su(s) vecino(s) más cercano(s) bajo una métrica definida de forma correcta y un correcto espacio de características (véase [21]).

Formalmente, sean $\{(\overline{X_1}', \overline{Y_1}'), \ldots, (\overline{X_n}', \overline{Y_n}')\}$ el conjunto formado por observaciones

de patrones $(d'' - 1)$-dimensionales, $\mathcal{X} = \{\overline{X_i}'\}_{i=1}^{n} = (\mathcal{D}'_{Train} \setminus \{labels\})$ y su correspondiente conjunto de etiquetas $\mathcal{Y} = \{\overline{Y_i}'\}_{i=1}^{n} = \{labels\} \subset \mathcal{D}'_{Train}$. Tomemos como $\overline{\mathbf{X}}$ al registro cuya etiqueta es desconocida y lo denominamos como el registro objetivo.

Definamos una medida de similaridad entre el registro objetivo y los elementos de $\mathcal{X}$.

Definición 4.3.1. La *métrica de Minkowski*, o norma p, se expresa como

$$\|\overline{\mathbf{X}} - \overline{X_i}'\|^p = \left(\sum_{j=1}^{d''-1} |\mathbf{x_j} - x_j^i|^p \right)^{1/p},$$

con $1 \leq p < \infty$.

En esta investigación, se utiliza el valor $p = 2$, es decir, la métrica euclidiana.

Otras métricas que podrían ocuparse en este método son:

- **Manhattan:** es una métrica muy usada en las implementaciones del algoritmo KNN y es un caso particular de la métrica de Minkowski tomando $p = 1$;

  $$\|\overline{\mathbf{X}} - \overline{X_i}'\| = \sum_{j=1}^{d''-1} |\mathbf{x_j} - x_j^i|,$$

  mide el valor absoluto entre dos puntos considerando la distancia como la suma de los catetos y no como una línea recta entre los puntos.

  Tal métrica funciona con datos de alta dimensionalidad, es menos intuitiva que la euclidiana, pues probablemente el valor que se obtiene es mayor que el proporcionada por la euclidea dado a que no ofrece el camino más corto. Se emplea cuando el conjunto de datos posee atributos discretos o binarios.

- **Chebyshev:** se encuentra definida como la mayor de las diferencias de los vectores a lo largo de cualquiera de sus coordenadas, es decir, representa la distancia más grande a lo largo de alguno de sus ejes. Tiene como ecuación

  $$\|\overline{\mathbf{X}} - \overline{X_i}'\| = \max_j |\mathbf{x_j} - x_j^i|.$$

Se emplea para obtener el número mínimo de movimientos para desplazarse dentro de una cuadricula, siendo estos casos muy específicos, lo que dificulta su utilidad como métrica de uso general.

- **Hamming:** es comúnmente empleada en casos donde se dispone de vectores booleanos o de cadenas, identificando aquellos elementos donde los vectores no tienen coincidencia. De los resultados que se obtienen, esta métrica también es conocida como métrica de superposición. La ecuación que la representa es la que sigue:

$$\|\overline{\mathbf{X}} - \overline{X_i}'\| = \sum_{j=1}^{d''-1} \mathbf{x_j} \oplus x_j^i.$$

Su implementación, usualmente, se relaciona con la corrección o detección de errores cuando los datos son transmitidos por redes informáticas.

- **Jaccard:** calcula la similaridad y diversidad de conjuntos de muestras. La métrica se obtiene restando a 1 el índice de Jaccard, obtenido por el tamaño de la intersección dividida por el tamaño de la unión de los conjuntos, es decir, tiene la ecuación

$$\|\overline{\mathbf{X}} - \overline{X_i}'\| = 1 - \frac{\overline{\mathbf{X}} \cap \overline{X_i}'}{\overline{\mathbf{X}} \cup \overline{X_i}'}.$$

Algunas consideraciones a tomar en cuenta es el hecho de que la métrica se encuentra muy influenciada por el tamaño de los datos, pues en grandes conjuntos el tamaño de la unión se modifica. Entre sus aplicaciones se encuentran el análisis de similitud de texto y otras aplicaciones con datos binarios.

- **Mahalanobis:** su utilidad radica en el hecho de tratarse de una medida de distancia óptima para hallar la similitud existente entre dos elementos. A diferencia de la métrica euclideana, aquí se consideran las características de las variables que definen

los atributos de cada elemento a través de la matriz de covarianza:

$$\|\overline{\mathbf{X}} - \overline{X_i}'\|^p = \left((\overline{\mathbf{X}} - \overline{X_i}')^T \sum_{j=1}^{d''-2} (\mathbf{x_j} - x_j^i) \right)^{1/2}.$$

En la imagen 4.1 se ilustra el comportamiento de algunas métricas descritas previamente.

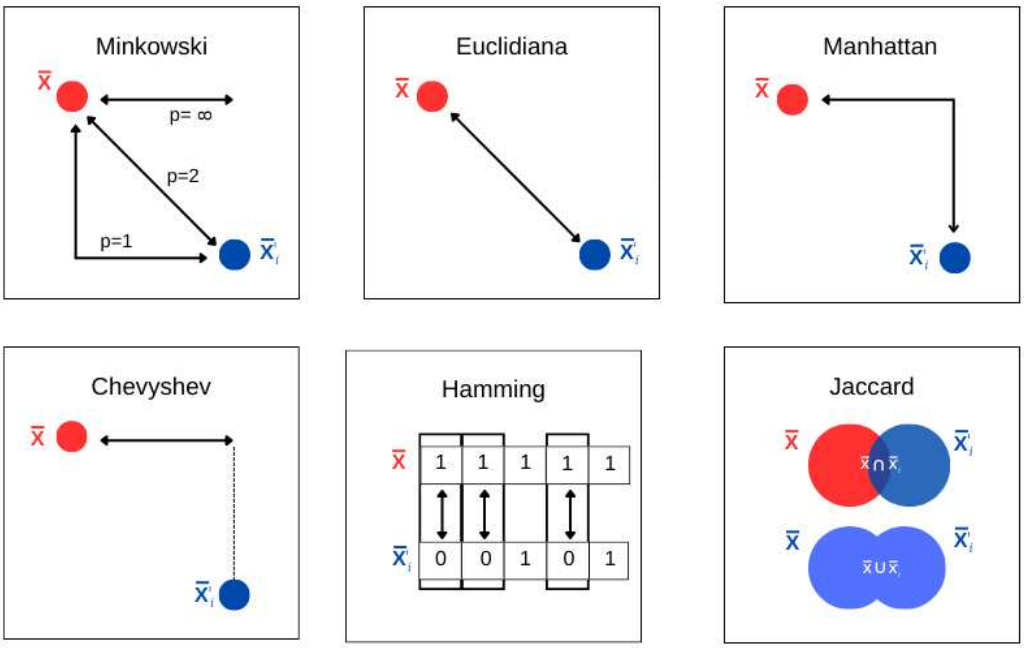


Figura 4.1: Métricas aplicables al método KNN. Modificada de [15]

A su vez en la Figura 4.2 se muestran las vecindades formadas por algunas de estas métricas.

Es importante resaltar que, de acuerdo a la naturaleza del problema, la métrica apta para trabajar es la euclidiana, retomemos el proceso con ella. Fijemos el valor de $k \in \mathbb{N}$, ahora, calculemos las distancias entre $\overline{\mathbf{X}}$ y $\overline{X_i}'$ y formemos el conjunto de distancias

$$Dist(\overline{\mathbf{X}}) = \{\|\overline{\mathbf{X}} - \overline{X_i}'\|^p : \ \overline{X_i}' \in \mathcal{X}\}.$$

El paso siguiente es tomar los k valores más pequeños del conjunto de distancias, con

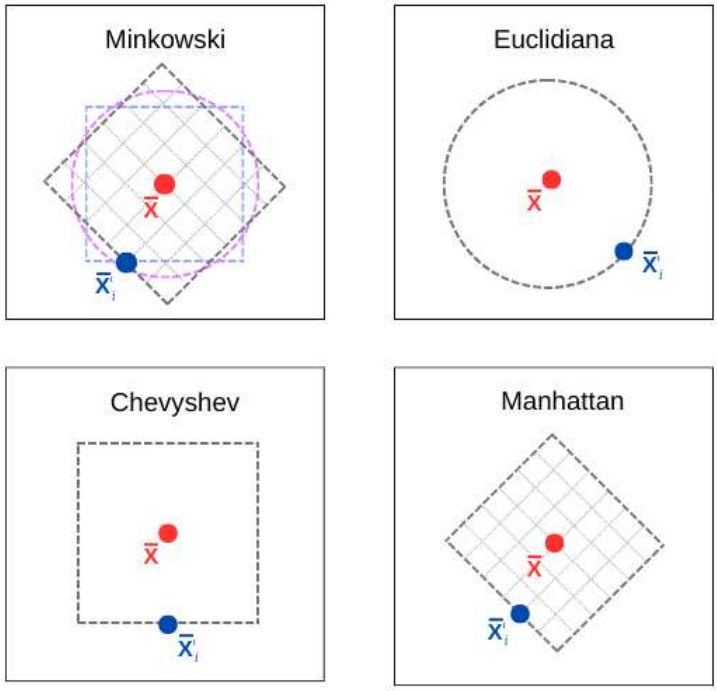


Figura 4.2: Vecindades formadas por las métricas Minkowski, Euclidiana, Manhattan y Chevyshev.

ellos, definimos como $\mathcal{N}_K(\overline{\mathbf{X}})$ al conjunto de índices de los registros correspondientes a los k valores más pequeños de $Dist(\overline{\mathbf{X}})$.

Ahora, dado que requerimos una clasificación binaria, la etiqueta 0 se redefine como -1, por lo que el conjunto de etiquetas toma los nuevos valores en $\{-1, 1\}$. Con lo anterior, definimos al modelo KNN como la función

$$f_{KNN}(\overline{\mathbf{X}}) = \begin{cases} 1, & si \quad \sum_{j\in\mathcal{N}_K(\overline{\mathbf{X}})} y_j \geq 0; \\ -1, & si \quad \sum_{j\in\mathcal{N}_K(\overline{\mathbf{X}})} y_j < 0. \end{cases} \tag{4.3.1}$$

La elección de k define la estructura de las vecindades, conocida como *localidad*. Esta localidad representa el patrón de la "región" que encierra a cada uno de los dos grupos. Para espacios bi-dimensionales puede ser representado como en la Figura 4.4, observamos que para valores pequeños de k el método KNN tiende a tener vecindades locales, mientras que para valores grandes el método ignora estas pequeñas aglomeraciones. En este caso,

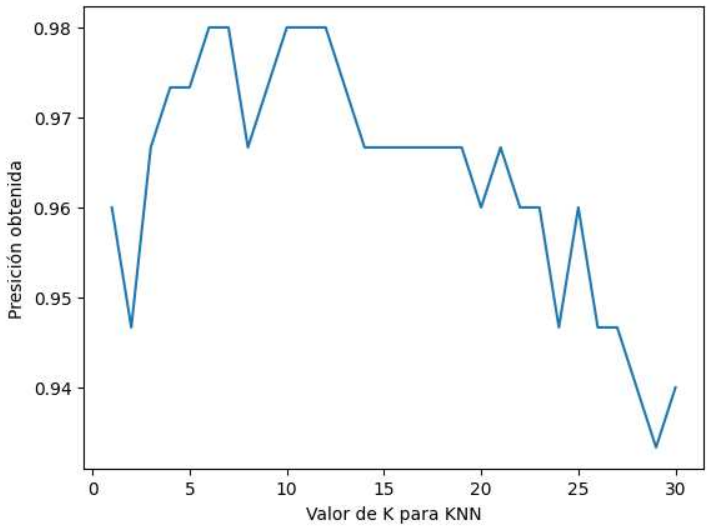


Figura 4.3: Precisión para modelos KNN para valores de k entre 1 y 31 para la base "Iris" de ScikitLearn.

surge la pregunta sobre la correcta elección del valor k, este problema es conocido como *selección de modelo* (*model selection*) y diversas técnicas, tal como *validación cruzada* (*cross-validation*), pueden ser empleadas para elegir el mejor modelo y parámetros.

Validación cruzada: el proceso consiste en dividir el número de registros del directorio en m partes iguales de tamaño $\frac{n}{m}$. De esta manera, se emplean $m-1$ subconjuntos para entrenar el modelo y el bloque restante es utilizado en las pruebas. El conjunto de prueba se elige al azar entre las m sub-bases y en cada uno de ellos se elige un valor fijo de k distinto al resto. Tal proceso se repite hasta que cada uno de los bloques haya sido considerado como conjunto de prueba, esto es, el proceso se repite m veces. Una vez hecho esto, la precisión del modelo en cada uno de las iteraciones se reporta y finalmente el valor de k con el que se obtiene la mejor métrica es el que se elige como óptimo (véase la Figura 4.3).

En esta investigación, se utilizó $k = 5$ debido a que se buscó de forma empírica el equilibrio entre ambas situaciones, de hecho, la instrucción *KNeighborsClassifier* de *ScikitLearn* lo contiene por defecto. En [30] se menciona que en clasificación binaria es recomendable considerar k impar con el objeto de evitar empates en la elección de la etiqueta. Se plantea

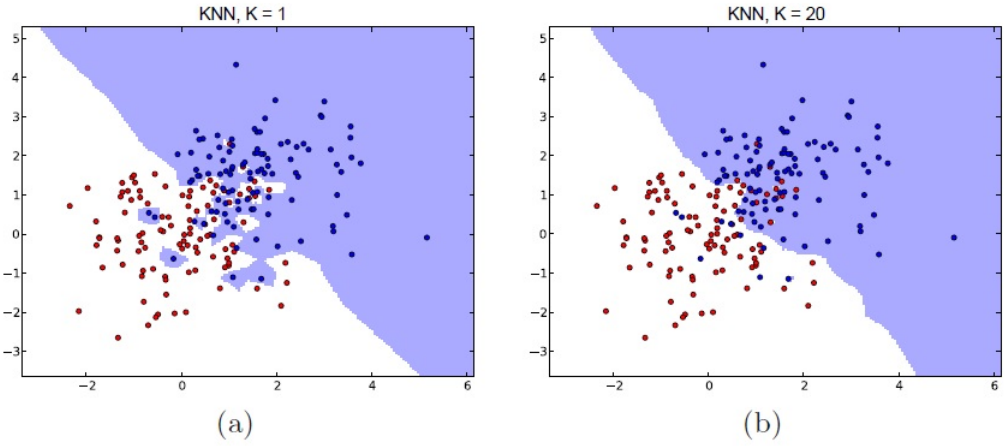


Figura 4.4: Comparación de las regiones para dos medidas de vecindades. (a) $k = 1$, se forman pequeñas vecindades fuera de la nube principal. (b) Tales regiones fuera de la general son desaparecidas. Imagen obtenida de [21].

para futuras investigaciones combinar la presente metodología con alguna selección de modelo.

Los métodos KNN no cuentan con una fase de entrenamiento como tal, sino con una fase de ejecución, cuya metodología a seguir es:

Fase de ejecución

1. Se toma $\overline{X}$ un registro arbitrario.
2. Se calcula el conjunto $Dist(\overline{X})$, el cual es las distancias de $\overline{X}$ con lo elementos de D_{Train}.
3. Se determinan los 5 valores más pequeños del conjunto obtenido en el punto anterior y se construye el conjunto $\mathcal{N}_K(\overline{X})$.
4. Se calcula $f_{KNN}(X)$ y se asigna el vector a alguna de las clases según el resultado obtenido.

Al proceso anterior le llamaremos *algoritmo KNN*. De esta manera, al realizar este proceso con todos los elementos del conjunto de entrenamiento, se obtienen los dos grandes grupos etiquetados por -1 o 1.

En este punto, surge la pregunta sobre si se realizó un buen entrenamiento del modelo, una forma de hacerlo es determinar el porcentaje de registros que se mantuvieron en la clasificación correcta y aquellos en los que la función determinó colocarlos en la contraria. El cociente entre el número de aciertos entre el total de registros es conocido como la precisión sobre el conjunto de entrenamiento, pero no es suficiente esta métrica, ya que fue sometido a registros con etiquetas conocidas, resta evaluar su comportamiento ante valores con etiquetas no vistas antes por el modelo.

En este paso y ya entrenado el modelo, se introducen a la metodología los vectores fila de la base $\mathcal{D}'_{Test}$, que fue destinada para este propósito. Para estos registros, que llamaremos $\overline{X}_{l,Test}$, el algoritmo determina una nueva etiqueta para este vector, y si este nuevo valor es igual a la etiqueta original se considera como un acierto, en caso contrario, se clasifica como un error. Esto se realiza con todos los elementos de la base de prueba, con lo que es posible calcular las siguientes métricas de desempeño.

Accuracy. Es el cociente del número de predicciones correctas entre el número total de predicciones realizadas por el modelo, su ecuación es:

$$Accuracy = \frac{Numero\ de\ predicciones\ correctas}{Numero\ total\ de\ predicciones}.$$

Se recomienda en casos donde los datos están balanceados, es decir, existe la misma cantidad de registros en cada etiqueta. El modelo obtuvo un accuracy de 0.9979623, lo que puede considerarse un buen desempeño.

Métricas para clasificadores binarios. En ocasiones el accuracy no es un buen estimador del desempeño (véase [4, Sección "Binary classification metrics", pág. 93]). Pero en clasificación binaria existen algunas otras métricas que están basadas en los siguientes indicadores:

Verdaderos positivos (TP): se refieren a todos los casos donde las clases originales y predicha son positivas.

Verdaderos negativos (TN): son todos los casos donde las clases originales y predicha son negativas.

Falsos positivos (FP): son aquellos casos en los que la clase original es negativa pero la predicha es positiva.

Falsos negativos (FN): en estos, la clase original es positiva pero la predicha es negativa.

Concluir que una red es segura cuando está en riesgo de ataque es un falso negativo. En caso contrario, cuando el modelo falla al determinar que una red está en riesgo cuando es segura se conoce como falso positivo.

Con los anteriores indicadores, se definen las siguientes métricas de desempeño del modelo, comúnmente denominadas por su nombre en inglés.

Precision: es el cociente del número de casos positivos que fueron correctamente predichos entre todos los casos que fueron identificados como positivos. Esto es:

$$Precision = \frac{TP}{TP + FP}.$$

Recall: es la división entre el número de casos positivos correctamente identificados entre el número de casos que también fueron correctamente colocados. Su representación es:

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

F1-score: es la media armónica entre Precision y Recall. Un valor igual a 1 implica un Precision y Recall perfectos, mientras que si es igual a 0 implica que el modelo no puede tener Precision y Recall. Se calcula como:

$$F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall}.$$

Se recomienda emplear esta métrica en problemas en los que el conjunto de datos a analizar está desbalanceado.

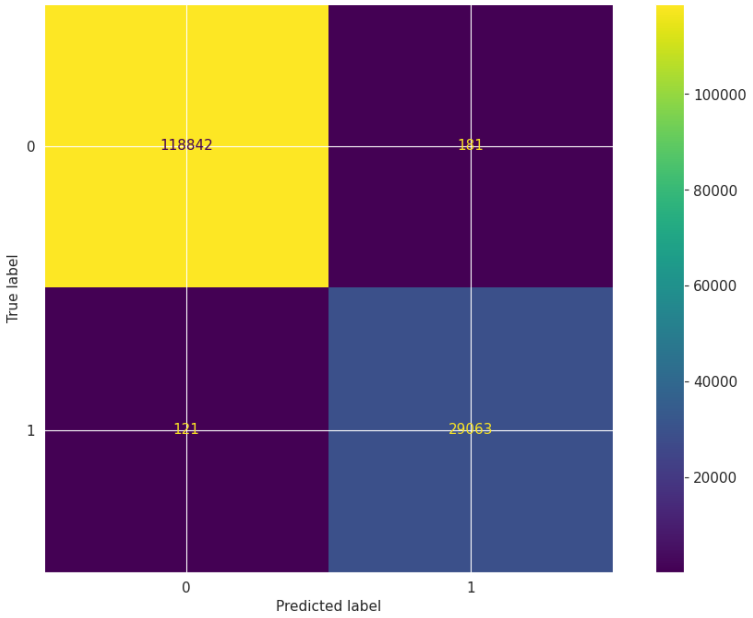


Figura 4.5: Matriz de confusión. Parte superior izquierda: Verdaderos positivos (TP); parte superior derecha: Falsos negativos (FN); parte inferior izquierda: Falsos positivos (FP); parte inferior derecha: Verdaderos negativos (TN).

Los valores respectivos de TP, FP, TN y FN del modelo propuesto se encuentran en la matriz de confusión de la Figura 4.5, los cuales son:

$$TP = 118,842, \tag{4.3.2}$$

$$FN = 181, \tag{4.3.3}$$

$$TN = 29,064, \tag{4.3.4}$$

$$FP = 121. \tag{4.3.5}$$

Con lo que, finalmente, se obtienen las siguientes métricas de desempeño:

$$Accuracy = 0.9979623; \tag{4.3.6}$$

$$Precision = 0.9989828; \tag{4.3.7}$$

$$Recall = 0.9984792; \tag{4.3.8}$$

$$F1 - score = 0.9987309. \tag{4.3.9}$$

De aquí que el modelo resulta tener un buen desempeño y se considera apto para realizar predicciones. Por consiguiente se somete al registro de la Tabla 4.3.1. En este caso, el modelo entrenado arroja una etiqueta de 1, por lo que se concluye que los datos del registro corresponden a una **red segura**.

dstbytes	**loggedin**	**count**	**srvcount**
2,534	1	5	12
dsthostcount	**dsthostsamesrcportrate**	**protocol_icmp**	**protocol_tcp**
6	8	0	1
service_ecr_i	**service_http**		
0	1		

Tabla 4.3.1: Valores del registro sometido al modelo entrenado para predicción de su etiqueta.

Con base en la investigación previa, existen diversos trabajos que indagan la problemática abordada en esta tesis con el mismo directorio, no obstante, los modelos de detección de intrusiones empleados son distintos al realizado en este estudio. A manera de comparación exhibiremos las métricas desempeño obtenidas en algunos de ellos.

- En [17] proponen un modelo de detección de intrusos basado en un modelo de seguridad denominado *Real-Time Sequential Deep Extreme Learning Machine Cybersecurity Intrusion Detection System (RTS-DELM-CSIDS)*. Obtienen un accuracy de 0.9273.

- En [28] se construyen diversos modelos de *CyberLearning* tomando en cuenta la clasificación binaria para detectar anomalías y determinados modelos de clasificación multiclase para varios tipos de ataques cibernéticos. Obtienen un accuracy que va de 0.98 a 0.99 dependiendo del tipo de clasificador considerado, con Precision, Recall y F1-Score entre estos mismos rangos. La diferencia es que en este trabajo consideran entre 17 y 42 características.

- En [20] proponen un sistema de detección de intrusiones centrado en una *máquina de aprendizaje extremo profundo (DELM)*, donde establece la evaluación de las características de seguridad y luego construyen un sistema adaptativo centrado en las características importantes. Este modelo tiene un accuracy de 0.9123.

- En [8] muestran un panorama general sobre los trabajos en este tema, también proporcionan una extracción de características y aplican diversos modelos posterior a ella. Se pueden encontrar modelos de Regresión Logística, Decision Tree Naive Bayes, Redes Neuronales, Maquinas de Soporte Vectorial, entre otros modelos ensamblados. Reportan que el modelo con el mejor desempeño fue uno de Regresión logística con la extracción de características basado en ANOVA, con un F1 de 0.884, un Accuracy de 0.877, un Precision de 0.96 y un Recall de 0.819. Todo lo anterior utiliza nueve características.

Gráficamente, la comparativa de los Accuracy previamente mencionados, se observa en la Figura 4.6. Similarmente son las gráficas comparativas de las métricas restantes de validación.

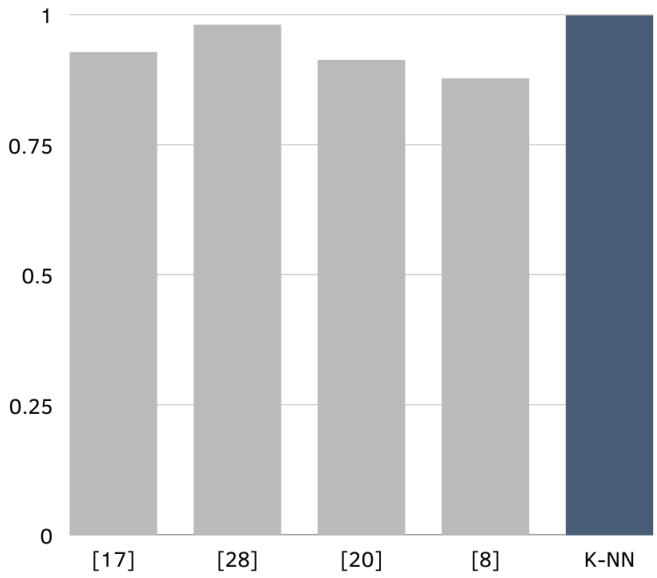


Figura 4.6: Accuracy en la literatura y el modelo K-NN propuesto.

Como se puede inferir, el modelo propuesto y desarrollado en esta investigación de tesis resulta contar con un buen desempeño, por lo que es apto para ser considerado en futuras líneas como una aportación más a la solución del problema de la ciberseguridad de las redes LAN, una problemática actual y de interés en la comunidad científica a nivel global.

Conclusiones

El objetivo general de esta investigación es proporcionar un modelo matemático de aprendizaje máquina para predicción de intrusiones. Se proporciona un modelo de tipo k-vecindades cercanas, que resulta ser un proceso con inicios en la comprensión y contextualización de la seguridad de las redes hasta su implementación.

En un principio se requiere contar con los conocimientos especializados sobre ciberseguridad, esta es la razón principal para iniciar con el Capítulo 1 que es preliminar y de proporcionar al lector un glosario de términos técnicos en la parte final de este manuscrito. Además, el necesario análisis exploratorio de la base a utilizar (*KDD-cyberattack*) nos proporciona la estructura general de la base, tal como su construcción, el tipo de datos y variables que contiene además de la distribución de cada una de ellas; más aún, las técnicas de visualización nos ayudan a extraer información relevante y a realizar hipótesis parciales sobre el problema, a manera de ejemplo, podemos decir que tanto el protocolo de mensajes de control y error de internet es el más empleado por los usuarios lo cual podría guardar relación con las intrusiones, este análisis se realiza en el Capítulo 2 y se verifica con los resultados obtenidos en el Capítulo 3.

Es conocido que, en general, el uso sólo de variables significativas incrementa la precisión de los modelos de aprendizaje máquina. El directorio originalmente contiene 42 variables o columnas, lo que es considerado un número alcanzable y trabajar con esa cantidad de variables recae en incremento de tiempo computacional o poca eficiencia en el

modelo; sin embargo, el análisis de correlación realizado en la sección 3.2 muestra que existen algunas características que se encuentran altamente correlacionadas tanto positiva como negativamente. Tales observaciones, fortalecen la intuición de someter la base de datos al proceso de selección de características más relevantes a la investigación. En la sección 3.4 se muestra un proceso funcional que se emplea en este problema para extraer características relevantes, no obstante, durante el desarrollo de la investigación en primera instancia se recurrió al análisis de componentes principales (PCA), sin embargo este procedimiento no resultó funcional a nuestro problema ya que después de la portabilidad, el directorio dispone de 116 variables y al aplicar PCA el directorio aún tiene dimensión mayor a 42. Con el proceso del *Select K best*, se obtienen 11 variables significativas en el problema, (considerando la propia variable objetivo); algunas de ellas son resultado de la codificación de algunas variables originales, proceso que se desarrolla en la sección 3.3.

El modelo objetivo de la investigación se plantea en el Capítulo 4. Se inicia exhibiendo formalmente el problema de clasificación en la sección 4.2, esto nos permite comprender el problema de una manera específica, por lo que en esta tesis se introduce el concepto de esta problemática en la Definición 4.2.1, que es una adaptación de aquellas existentes en la literatura. Posteriormente, en la sección 4.3 se plantea el modelo matemático de k-vecindades cercanas, desde la forma en la que se calculan las vecindades a través del cálculo del conjunto de distancias para cada registro ($Dist(\overline{X})$) y considerando la norma euclidiana, hasta la definición del modelo mediante la función $f_{KNN}(\overline{X})$ expresada como la Ecuación 4.3.1 que realiza las clasificaciones. En esta misma sección, se muestra el proceso de entrenamiento y validación, obteniendo buenos valores en las métricas de desempeño. Finalmente, se introduce al modelo un nuevo registro (4.3.1) y este predice su etiqueta al clasificarlo en alguno de los dos grupos, de hecho, lo considera como datos de una red segura. Es importante recordar que el modelo que se propone es predictivo, en caso de que se requiera hacer un modelo correctivo se debe realizar una automatización que monitore la red en tiempo real y genere una respuesta en el momento en que se registre una intrusión.

Este capítulo cierra con la comparación de las métricas de desempeño con algunas

obtenidas en diversos trabajos recientes, el modelo propuesto cumple con las perspectivas marcadas inicialmente.

Finalmente se pone a disposición del lector la programación requerida en el desarrollo de la tesis. Parte del código se encuentra disponible en la sección de anexos, mientras que el programa completo se encuentra de forma libre en

`https://colab.research.google.com/drive/1XWJFknpY3QrJtzlOxHFyE1NsoGW_MWKH?usp=sharing`.

Con todo lo anterior, se logra el objetivo general y se cumplen las expectativas e hipótesis originalmente planteadas. Más aún, los resultados obtenidos generan nuevas líneas de investigación donde se busca abordar la problemática desde otra perspectiva o bien mejorar el rendimiento del modelo así como subsanar las deficiencias que pueda tener al someterse a algunos cambios. Tal es el caso de un cambio en la métrica elegida, la selección del modelo, entre otros factores.

Glosario

- **Bit:** acrónimo de binary digit, es la unidad de información más pequeña. Solo puede considerar los valores 1 y 0 relacionados a encendido y apagado, respectivamente, o bien algún tipo de consideración dicotómica.
- **Bot**: el término proviene de acortar la palabra "robot". Es un programa que realiza tareas repetitivas, predefinidas y automatizadas. Está diseñado para imitar o sustituir el accionar humano.
- **Byte:** conjunto de 8 bits ordenados que establece el mínimo objeto de memoria orientable de un servidor.
- **Caché:** información generada por las aplicaciones y servicios que se almacena en la memoria en caso de ser necesaria posteriormente.
- **Datagrama:** es un paquete de datos que constituye el mínimo bloque de información en una red de conmutación de paquetes. Usualmente se encuentran estructurados en secciones de cabecera y datos transmitidos en payload.
- **Echo Request**: es un mensaje de control que se envía a un host con la expectativa de recibir de él un Echo Reply (Respuesta eco).
- **Estadístico:** es una cantidad numérica calculada sobre una muestra de una población con el fin de conocer el comportamiento de cierto atributo en particular. Es

variante y conocido.

- **Enrutar:** determinar el itinerario que debe seguir un paquete de datos dentro de una red de comunicación para llegar a su destino.

- **HTML:** de sus siglas en inglés, el lenguaje de marcado de hipertexto es un lenguaje descriptivo que especifica la estructura de las páginas web. Un documento HTML es un documento de texto plano estructurado por *elementos*, los cuales pueden contener datos, fragmentos de imágenes o textos, o bien estar vacíos.

- **HTTP:** de sus siglas en inglés, el protocolo de transferencia de hipertexto permite realizar una gestión de datos y recursos, como pueden ser documentos HTML. Es la base de cualquier intercambio de datos en la web, y además es un protocolo de estructura cliente-servidor,lo que indica que una petición de datos se inicia por quien recibirá la información, es decir, el cliente, el cual opera usualmente en un navegador Web.

- **Memoria caché:** es una capa de almacenamiento de datos, donde se resguardan datos usualmente transitorios o temporales, los cuales facilitan su acceso en caso de ser requeridos por el almacenamiento principal.

- **Metadatos:** describen la relación entre las personas que han creado y utilizado los atributos, la gestión y el uso de los documentos, además de las actividades en las cuales han sido creados y usados.

- **Paquetes IP:** fragmento de mensaje que contiene un encabezado de entre 20 o 24 bytes de longitud y datos de longitud, también en bytes, variable. El *encabezado* incluye las direcciones IP de la fuente y del destino, además de otros datos que ayudan a enrutar el paquete. Los *datos* son el contenido real, tales como una cadena de letras o parte de una página web.

- **Parámetro:** es una cantidad numérica que se calcula sobre el total de la población y tiene por fin resumir la información que esta toma sobre algún atributo determinado.

Es fijo y desconocido.

- **Playload:** es el conjunto de datos útiles transmitidos obtenidos de excluir cabeceras, metadatos, información de control y otros datos que son enviados para facilitar la entrega del mensaje.
- **Protocolo de internet (IP):** describe la estructura de los paquetes que viajan por internet.
- **REJ:** es un mensaje de protocolo enviado por un destinatario para indicar que se recibió una señal, pero que no se acepta por una variedad de razones, entre las cuales se encuentran fallo de suma de comprobación, datos incompletos, encabezados inválidos o mal formados, entre otros.
- **RST:** Es un bit que se encuentra en el campo del código en el protocolo TCP, y se emplea para reiniciar una conexión debido a paquetes corrompidos o a SYN duplicados, entre otros factores.
- **Servicio de red:** es un conjunto de equipo y software conectados entre sí mediante dispositivos, con el fin de enviar o recibir información a través de impulsos eléctricos u ondas electromagnéticas para poder realizar el transporte de los datos. Generalmente, se suelen compartir recursos, información y ofrecer algunos servicios especiales.
- **SYN:** es una abreviatura de sincronizar. Es un paquete TCP enviado a otra computadora solicitando que establezca una conexión entre ellos. Cuando se habla de *SYN flood* se hace referencia a un ataque de denegación de servicio; el ataque envía un flujo de paquetes de datos maliciosos a un sistema de destino con la intención de sobrecargar el objetivo y evitar así su uso ilegítimo.
- **Telnet:** proviene del acrónimo Telecomunication Network. Es un protocolo usado para establecer conexiones remotas con otros ordenadores, servidores, y otros dispositivos con sistema compatible en el acceso a través de este sistema de comunicación.

Bibliografía

[1] C. C. Aggarwal, *Data mining*, Springer International Publishing, 2015.

[2] E. Aguirre Hernández, J. Calva Bautista, et al., *Comparación de los modelos OSI Y TCP/IP*, `https://www.uaeh.edu.mx/scige/boletin/huejutla/n10/r1.html`, 2017.

[3] F. Alhaidari, S. H. Almotiri, M. A. Al Ghamdi, M. Adnan Khan, A. Rehman, S. Abbas, K. Masood Khan, and Atta ur Rahman, *Intelligent software-defined network for cognitive routing optimization using deep extreme learning machine approach*, Computers, Materials & Continua **67** (2021), no. 1, 1269–1285.

[4] R. Banik, *Hands-on recommendation systems with python*, Packt Publishing, Birmingham, UK, 2018.

[5] S. Canovas, *Técnicas de protección-seguridad informática*, `https://sites.google.com/site/seguridadinformaticabowenzhao/tema-9-seguridad-en-redes/9-3-tecnicas-de-proteccion`, 2018.

[6] T. Cover and P. Hart, *Nearest neighbor pattern classification*, IEEE Transactions on Information Theory **13** (1967), no. 1, 21–27.

[7] M. Cowles, C. Davis, et al., *On the origins of the* 0.05 *level of statistical significance*, The American Statistician **51** (1997), no. 1, 35–41.

[8] S. Das, S. Saha, A. T. Priyoti, E. K. Roy, F. T. Sheldon, A. Haque, and S. Shiva, *Network intrusion detection and comparative analysis using ensemble machine learning and feature selection*, IEEE Transactions on Network and Service Management (2021), 4821–4833.

[9] Secretaría de estado de digitalización e inteligencia artificial, *¿Cómo aprenden las máquinas: Machine Learning y sus diferentes tipos?*, `https://datos.gob.es/es/blog/como-aprenden-las-maquinas-machine-learning-y-sus-diferentes-tipos`, 2020.

[10] M. DeGroot and M. Schervish, *Probability and statistics*, Pearson, 2012.

[11] Estadísticando, *Prueba t de Student para dos muestras*, `http://estadisticando.blogspot.com/2016/04/prueba-t-student-para-dos-muestras.html`, 2016.

[12] Firewalls-Hardware, *UTM: Gestión Unificada de Amenazas (Unified Threat Management)*, `https://firewalls-hardware.com/utm-gestion-unificada-amenazas-unified-threat-management/`, s.f.

[13] R. A. Fisher, *Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population*, Biometrika **10** (1915), no. 4, 507–521.

[14] E. Fix and J. L. Hodges, *Discriminatory analysis. nonparametric discrimination: Consistency properties*, International Statistical Review / Revue Internationale de Statistique **57** (1989), no. 3, 238–247.

[15] M. Grootendorst, *9 Distance Measures in Data Science*, `https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa`, 2021.

[16] M. Haddadi and R. Beghdad, *DOS-DDOS: taxonomies of attacks, countermeasures, and well-known defense mechanisms in cloud environment*, EDPACS **57** (2018), no. 5, 1–26.

[17] A. Haider, M. Adnan Khan, A. Rehman, M. Ur Rahman, and Hyung Seok Kim, *A real-time sequential deep extreme learning machine cybersecurity intrusion detection system*, Computers, Materials & Continua **66** (2021), no. 2, 1785–1798.

[18] Insight, *Insight technology report 2022: It ambitions for bussiness transformation*, `https://www.insight.com/en_US/campaigns/hva/insight/navigating-post-pandemic-it.html`, 2022.

[19] M. A. Khan, T. M. Ghazal, Sang-Woong Leea, and A. Rehman, *Data fusion-based machine learning architecture for intrusion detection*, Computers, Materials & Continua **70** (2022), no. 2, 3399–3413.

[20] M. A. Khan, A. Rehman, K. M. Khan, M. A. Al Ghamdi, and S. H. Almotiri, *Enhance intrusion detection in computer networks based on deep extreme learning machine*, Computers, Materials & Continua **66** (2021), no. 1, 467–480.

[21] O. Kramer, *Dimensionality reduction with unsupervised nearest neighbors*, Intelligent Systems Reference Library, vol. 51, Springer Berlin Heidelberg, Berlin, 2013.

[22] J. Kurose and K. Ross, *Redes de computadoras: Un enfoque descendente*, 7 ed., Pearson Education, 2017.

[23] E. Lehmann and J. Romano, *Testing statistical hypotheses*, Springer Science & Business Media, 2005.

[24] McAfee, *¿qué es un proxy?*, `https://www.mcafee.com/blogs/es-es/privacy-identity-protection/que-es-un-proxy/`, 3 2022.

[25] R. Nilsson, *Statistical feature selection: With applications in life science*, Linköping studies in science and technology: Dissertations, Department of Physics, Chemistry and Biology, Linköping University, 2007.

[26] S. Ortiz and D. Enriquez, url=https://www.idc.com/getdoc.jsp?containerId=prLA49766122#:̃:tex

[27] R. Petersen, *Data mining for network intrusion detection. a comparison of data mining algorithms and an analysis of relevant features for detecting cyber-attacks.*, Master of science in engineering: Industrial engineering and management, MID Sweden University. Department of Information and Communication Systems (IKS), 2015.

[28] I. H. Sarker, *Cyberlearning: Effectiveness analysis of machine learning security modeling to detect cyber-anomalies and multi-attacks*, Internet of Things **14** (2021), 100393.

[29] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, *IntruDtree: A Machine Learning Based Cyber Security Intrusion Detection Model*, Symmetry **12** (2020), no. 5, 1–15.

[30] S. S. Skiena, *The data science design manual*, Springer International Publishing, 2017.

[31] N. Snell, *Internet que hay que saber*, Prentice Hall, España, 1995.

[32] M. Solórzano, *Internet: Una necesidad básica*, `https://www.linkedin.com/pulse/internet-una-necesidad-b%C3%A1sica-maria-fernanda-sol%C3%B3rzano-palacio/?originalSubdomain=es`, 2022.

[33] W. Stallings, *30 estadísticas de seguridad informática que importan (actualizadas al 2021)*, `https://preyproject.com/blog/es/30-estadisticas-seguridad-informatica/`, 2021.

[34] C. Standley, *¿Cuál es la diferencia entre una LAN y una WAN?*, `https://purple.ai/es/blogs/cual-es-la-diferencia-entre-una-lan-y-una-wan/`, 2021.

[35] F. Thompson, *Las 7 amenazas en ciberseguridad que pueden afectar su vida*, `https://cio.com.mx/las-7-amenazas-en-ciberseguridad-que-pueden-afectar-su-vida-2/`, 2014.

[36] D. Wackerly, W. Mendenhall, and Scheaffer R., *Mathematical statistics with applications*, Cengage Learning, 2014.

Anexo 1: Variables del directorio.

En las tablas a continuación se colocan las variables de las que se conforma el directorio con una breve explicación de su significado clasificadas bajo ciertos criterios. Toda la información proporcionada se obtiene de [27].

Características básicas de las conexiones TCP individuales.

Nombre de la variable	**Descripción**	**Tipo**
duration	Duración (número de segundos) de la conexión.	Entera
protocol_type	Tipo de protocolo de comunicación a través de internet por ejemplo tcp, udp, etc.	Texto
service	Servicio de red en el destino, por ejemplo, http, telnet, etc.	Texto
flag	Estado normal o de error de la conexión.	Texto
src_bytes	Número de bytes de datos transmitidos desde el origen hacia el destino.	Entera
dst_bytes	Número de bytes de datos transmitidos desde el destino hacia el origen.	Entera
wrong_fragment	Número de fragmentos "incorrectos".	Entera
urgent	Número de paquetes urgentes.	Entera

Continúa en la siguiente página

Tabla 4.3.2 – Continuación

Nombre de la variable	**Descripción**	**Tipo**
land	1 si la conexión es desde el mismo host o si la conexión es al mismo host; 0 en caso contrario.	Binaria

Funciones de contenido dentro de una conexión sugerida por el conocimiento del dominio.

Nombre de la variable	**Descripción**	**Tipo**
hot	Número de indicadores "calientes".	Entera
num_failed_logins	Número de intentos fallidos de inicio de sesión.	Entera
logged_in	1 si inició sesión correctamente; 0 de lo contrario.	Binaria
num_compromised	Número de condiciones "comprometidas".	Entera
root_shell	1 si se obtiene shell raíz; 0 de lo contrario.	Binaria
su_attempted	1 si se intentó el comando "su root"; 0 de lo contrario.	Binaria
num_root	Número de accesos "root".	Entera
num_file_creations	Número de operaciones de creación de archivos.	
num_shells	Número de avisos de shell.	Entera
num_access_files	Número de operaciones en archivos de control de acceso.	Entera
num_outbound_cmds	Número de comandos salientes en una sesión ftp.	Entera
is_hot_login	1 si el inicio de sesión pertenece a la lista "caliente"; 0 de lo contrario.	Binaria
is_guest_login	1 si el inicio de sesión es un inicio de sesión "invitado"; 0 de lo contrario.	Binaria

Características del tráfico calculadas utilizando una ventana de tiempo de dos segundos.

Nombre de la variable	**Descripción**	**Tipo**
count	Número de conexiones al mismo host que la conexión actual en los últimos dos segundos.	Entera
Las siguientes características se refieren a estas conexiones del mismo host.		
srv_count	Número de conexiones al mismo servicio que la conexión actual en los últimos dos segundos.	Entera
serror_rate	Porcentaje de conexiones que tienen errores "SYN".	Continua
rerror_rate	Porcentaje de conexiones que tienen errores "REJ".	Continua
same_srv_rate	Porcentaje de conexiones al mismo servicio.	Continua
diff_srv_rate	Porcentaje de conexiones a diferentes servicios.	Continua
Las siguientes características se refieren a estas conexiones del mismo servicio.		
srv_serror_rate	Porcentaje de conexiones que tienen errores "SYN".	Continua
srv_rerror_rate	Porcentaje de conexiones que tienen errores "REJ".	Continua

Funciones de tráfico basadas en el host.

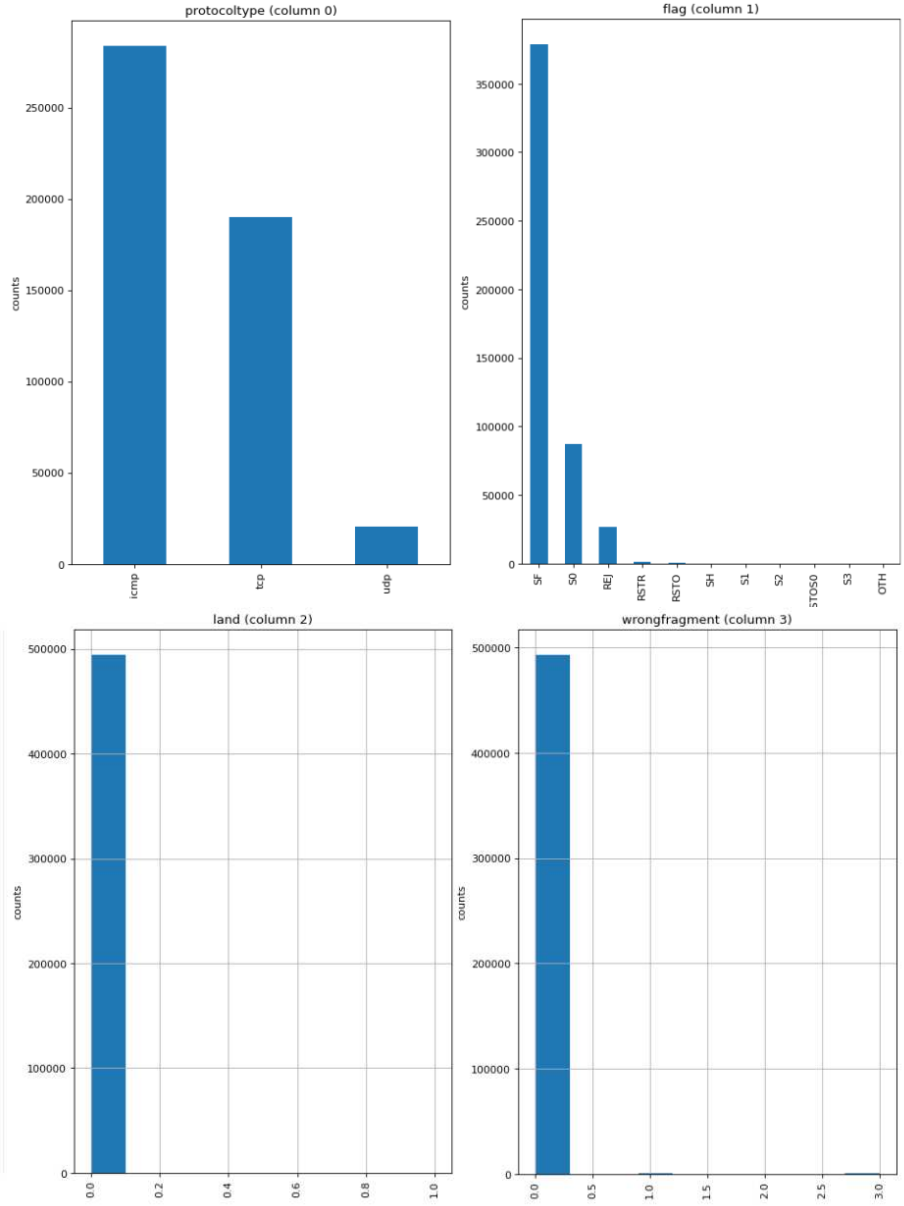
Nombre de la variable	**Descripción**	**Tipo**
srv_diff_host_rate	Porcentaje de conexiones diferentes hosts.	Continua
dst_host_count	Número de conexiones que tienen el mismo host de destino.	Entera

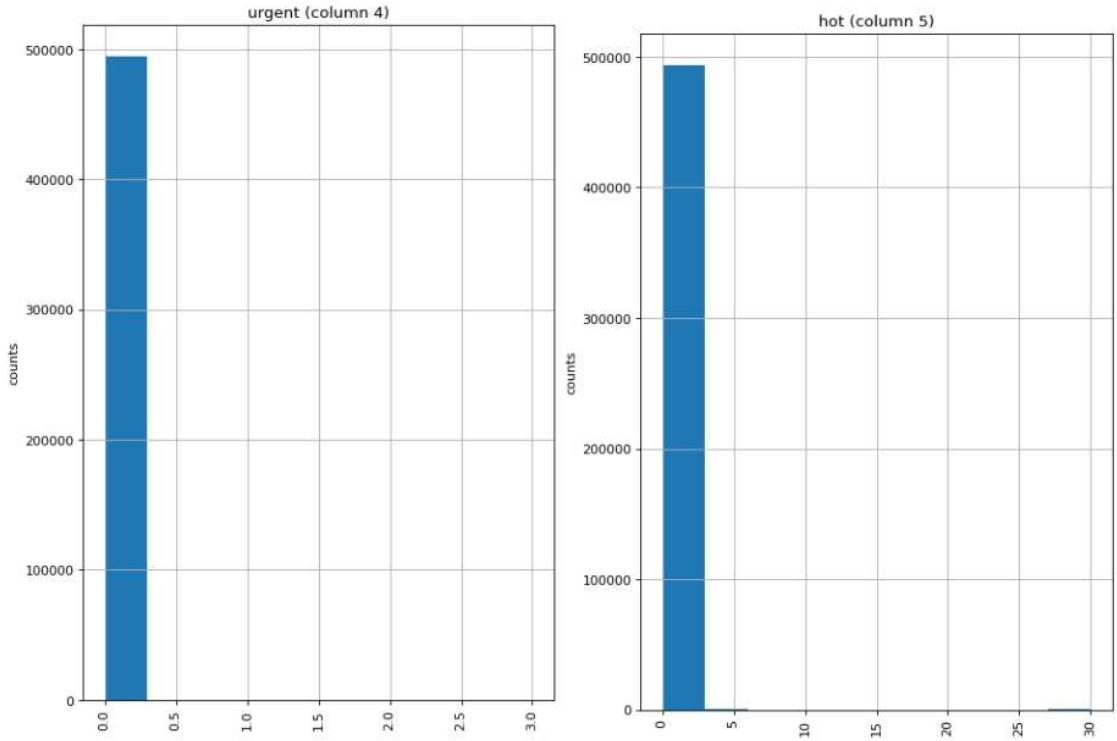
Continúa en la siguiente página

Tabla 4.3.5 – Continuación

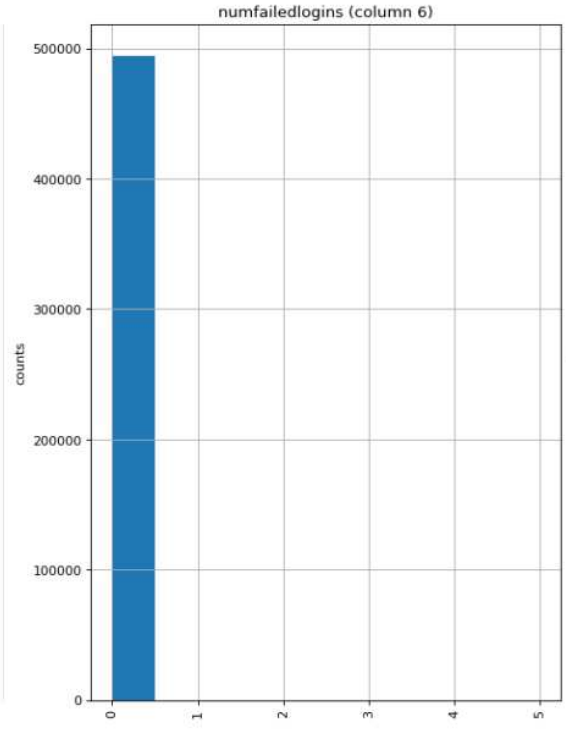
Nombre de la variable	**Descripción**	**Tipo**
dst_host_srv_count	Número de conexiones que tienen el mismo host de destino y usan el mismo servicio.	Entera
dst_host_same_srv_ rate	Porcentaje de conexiones que tienen el mismo host de destino y usan el mismo servicio	Continua
dst_host_diff_srv_rate	Porcentaje de diferentes servicios en el host actual.	Continua
dst_host_same_src_port_rate	Porcentaje de conexiones al host actual que tienen el mismo puerto src.	Continua
dst_host_srv_diff_host_rate	Porcentaje de conexiones al mismo servicio provenientes de diferentes hosts.	Continua
dst_host_serror_rate	Porcentaje de conexiones al host actual que tienen un error en el sistema operativo.	Continua
dst_host_srv_serror_rate	Porcentaje de conexiones al host actual y al servicio especificado que tienen un error en el sistema operativo.	Continua
dst_host_rerror_rate	Porcentaje de conexiones al host actual que tienen un error RST.	Continua
dst_host_srv_rerror_rate	Porcentaje de conexiones al host actual y al servicio especificado que tienen un error RST.	Continua

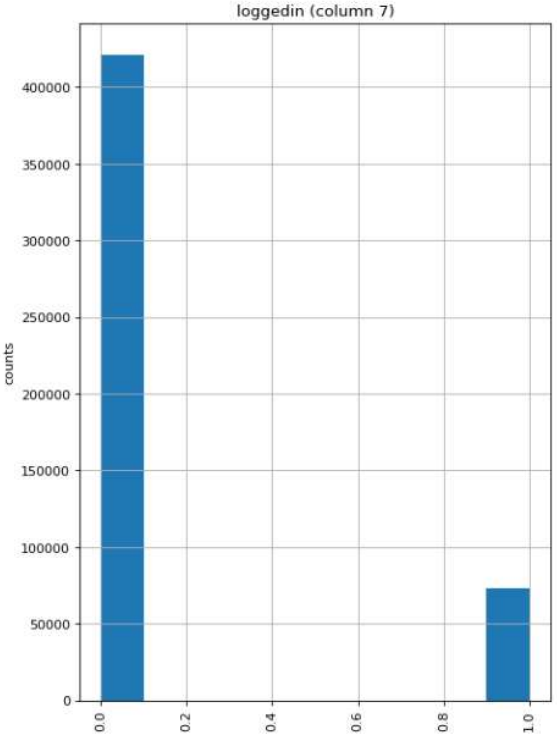
Anexo 2: Histogramas

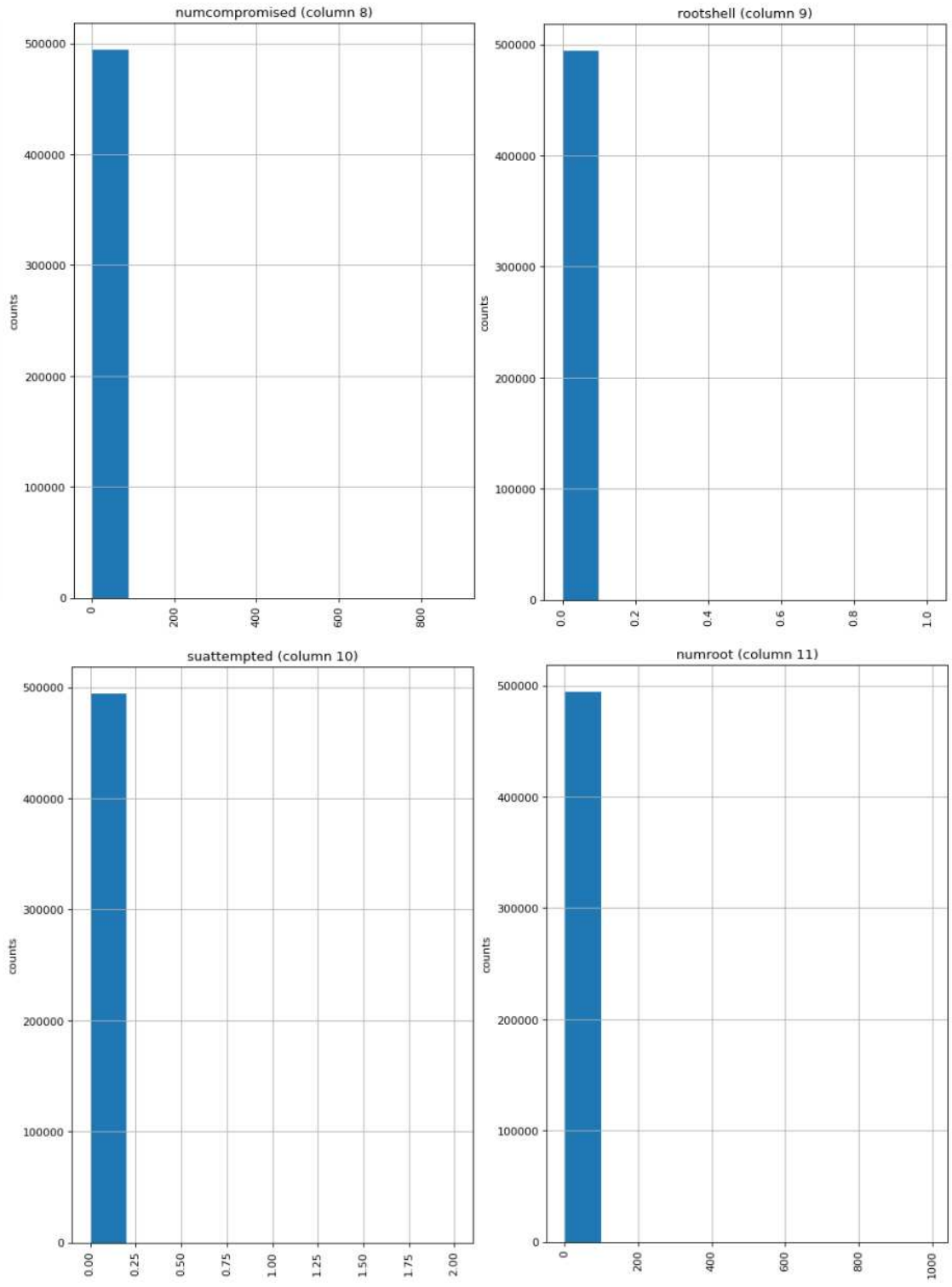


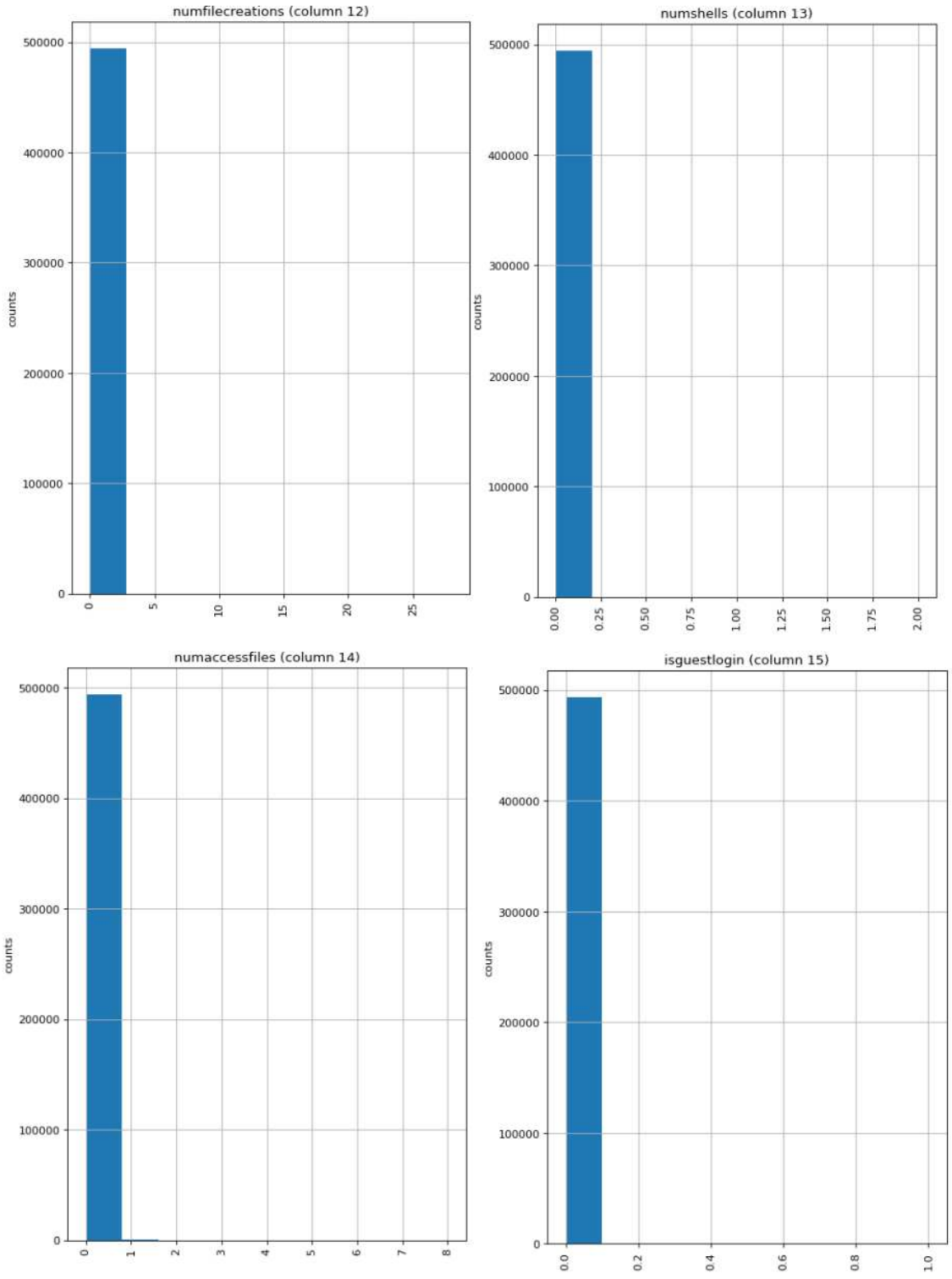

urgent (column 4)
counts
500000
400000
300000
200000
100000
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

hot (column 5)
counts
500000
400000
300000
200000
100000
0
0
5
10
15
20
25
30


numfailedlogins (column 6)
counts
500000
400000
300000
200000
100000
0
0
1
2
3
4
5


loggedin (column 7)
counts
400000
350000
300000
250000
200000
150000
100000
50000
0
0.0
0.2
0.4
0.6
0.8
1.0


numcompromised (column 8)
rootshell (column 9)
suattempted (column 10)
numroot (column 11)
counts
0
100000
200000
300000
400000
500000
0
200
400
600
800
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
1000


numfilecreations (column 12)
numshells (column 13)
numaccessfiles (column 14)
isguestlogin (column 15)
counts
0
100000
200000
300000
400000
500000
0
5
10
15
20
25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0
1
2
3
4
5
6
7
8
0.0
0.2
0.4
0.6
0.8
1.0

Anexo 3: Exploración del directorio

Para la parte práctica de este trabajo se utiliza *COLAB* como Notebook de desarrollo basado en Python. Con el objetivo de guiar en la investigación, se comparte parte del código utilizado.

El proceso de exploración comienza cuando se cargan algunas de las librerías necesarias:

```
import numpy as np # linear algebra
import pandas as pd # data processing
import matplotlib.pyplot as plt # plotting
import plotly
import seaborn as sns
```

Se carga la base de datos desde el servidor o desde la nube y con la instrucción *datos.head()* se visualizan los primeros cinco registros junto con algunas de las columnas. No obstante, para mejor entendimiento del significado de los datos, el siguiente fragmento de código permite añadir el nombre de las variables para una mejor visualización.

```
columns = ["duration", "protocoltype", "service", "flag",
       "srcbytes", "dstbytes", "land", "wrongfragment",
      "urgent", "hot", "numfailedlogins", "loggedin",
       "numcompromised", "rootshell", "suattempted",
```

```
        "numroot", "numfilecreations", "numshells",
        "numaccessfiles", "numoutboundcmds", "ishostlogin",
        "isguestlogin", "count", "srvcount", "serrorrate",
        "srvserrorrate", "rerrorrate", "srvrerrorrate",
        "samesrvrate", "diffsrvrate", "srvdiffhostrate",
        "dsthostcount", "dsthostsrvcount",
        "dsthostsamesrvrate", "dsthostdiffsrvrate",
        "dsthostsamesrcportrate", "dsthostsrvdiffhostrate",
        "dsthostserrorrate", "dsthostsrvserrorrate",
        "dsthostrerrorrate", "dsthostsrvrerror_rate",
        "labels"]


datos = pd.read_csv("kddcup.data_10_percent", names=columns)
matriz=datos.head()
matriz_t=matriz.T
matriz_t
```

La instrucción *datos.shape* muestra que la base de datos tiene un total 494,021 registros con 42 variables, donde se considera como registro adicional la columna añadida previamente. Además, con la instrucción *datos.dtypes* se visualizan los tipos de datos en el directorio.

Para la visualización gráfica de los valores faltantes se emplea la instrucción:

```
plt.figure(figsize=(16,6))
sns.heatmap(datos.isnull(),cmap='viridis',cbar=False)
```

Posteriormente, las medidas de tendencia central se obtienen con la instrucción *des = datos.describe()*.

Para obtener los histogramas de frecuencias se emplea el siguiente código:

- Histogramas generales

```
def plotPerColumnDistribution(df, nGraphShown, nGraphPerRow):
nunique = df.nunique()
df = df[[col for col in df if nunique[col] > 1 and nunique[col]
↪ < 50]] # For displaying purposes, pick columns that have
↪ between 1 and 50 unique values
nRow, nCol = df.shape
columnNames = list(df)
nGraphRow = 6
plt.figure(num = None, figsize = (6 * nGraphPerRow, 8 *
↪ nGraphRow), dpi = 80, facecolor = 'w', edgecolor = 'k')
for i in range(min(nCol, nGraphShown)):
    plt.subplot(nGraphRow, nGraphPerRow, i + 1)
    columnDf = df.iloc[:, i]
    if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
        valueCounts = columnDf.value_counts()
        valueCounts.plot.bar()
    else:
        columnDf.hist()
    plt.ylabel('counts')
    plt.xticks(rotation = 90)
    plt.title(f'{columnNames[i]} (column {i})')
plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad = 1.0)
plt.show()

plotPerColumnDistribution(datos, 38, 5)
```

- Frecuencias de intrusiones

```
sns.set_style("darkgrid")
plt.rcParams["figure.figsize"] = (12,8)
font = {"size"   : 11}
plt.rc('font', **font)
grouped_labels = datos.groupby("labels")["labels"].count().
↪ sort_values(ascending=False)
plt.xticks(rotation=45)
sns.barplot(x=grouped_labels.index, y=grouped_labels.values)
plt.title("Numero de intrusiones y eventos normales")
plt.ylabel("Frecuencia")
```

- Frecuencia de eventos por tipo de protocolo

```
grouped_labels = datos.groupby("protocoltype")["protocoltype"].
↪ count().sort_values(ascending=False)
plt.xticks(rotation=45)
sns.barplot(x=grouped_labels.index, y=grouped_labels.values)
plt.title("Eventos por tipo de protocolo")
plt.ylabel("Frecuencia")
```

Finalmente, se normaliza el directorio.

```
def remove_dot_normalize(label):
""" Quitar los puntos y separar las variables en normales y atacadas
↪ """
label = label.replace(".", "")
return label if label == "normal" else "attack"
```

```
datos["labels"] = datos["labels"].apply(lambda label:
↪ remove_dot_normalize(label))
print(pd.unique(datos["labels"]))
```

Anexo 4: Selección de características

En esta sección se cargan las librerías según requiera el programa.

Análisis de correlación

Para el mapa de calor del análisis de correlación, se emplea el código:

```
plt.figure(figsize=(20,20))
sns.heatmap(data=round(datos.corr(),2), annot=True)
```

Algoritmo *Select K best*

Se separan los vectores de características con respecto al vector objetivo.

```
X = datos.drop("labels", axis=1)
y = datos["labels"]


[col for col in X.columns if X[col].nunique() == 1]


X.drop(["numoutboundcmds", "ishostlogin"], axis=1, inplace=True)
```

Posterior a esto se importan las librerías necesarias.

Una vez hecho esto, se codifican las variables así como los predictores.

```
lr = LabelEncoder()
y = lr.fit_transform(y)

enc_protocol = pd.get_dummies(datos["protocoltype"], prefix="
↪ protocol_")
enc_service = pd.get_dummies(datos["service"], prefix="service_")
enc_flag = pd.get_dummies(datos["flag"], prefix="flag_")

X = pd.concat([X, enc_protocol, enc_service, enc_flag], axis=1)
X.drop("protocoltype", axis=1, inplace=True)
X.drop("service", axis=1, inplace=True)
X.drop("flag", axis=1, inplace=True)
```

Al realizar el proceso anterior la indicación *X.shape* arroja que el directorio ahora tiene 116 variables. Resta elegir las variables más relevantes, para ello el código que se usa una vez cargando las librerías necesarias es:

```
selector = SelectKBest(f_classif)
selector.fit(X, y)

cols = selector.get_support(indices=True)
features_df_new = X.iloc[:,cols]
```

Para graficar los valores p, se emplea:

```
scores = -np.log10(selector.pvalues_)
X_indices = np.arange(X.shape[-1])
plt.figure(1)
plt.clf()
plt.bar(X_indices - 0.05, scores, width=0.2)
```

```
plt.title("Feature univariate score")
plt.xlabel("Feature number")
plt.ylabel(r"Univariate score ($-Log(p_{value})$)")
plt.show()
```

Anexo 5: Modelo clasificador

Para esta sección la librería clave a usar es *sklearn* por tanto tiene que importarse. Por otro lado el acceso a las librerías se efectúa con

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
```

Con la instrucción $y = datos["labels"]$ se crea el vector de etiquetas.

Para crear los conjuntos de entrenamiento y de prueba de recurre a

```
X\_train, X\_test, y\_train, y\_test = train\_test\_split(features\
↪ _df\_new, y, test\_size=.3, stratify=y)
```

Posteriormente, se define el programa y el entrenamiento como

```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
```

Anexo 6: Validación

Para importar la librería empleada en esta sección se usa *from sklearn.metrics import f1_score, ConfusionMatrixDisplay.*

La instrucción *knn.score(X_test, y_test)* muestra el accurracy del modelo. Mientras que el código a continuación permite obtener la métrica f1 score:

```
y_knn_predict = knn.predict(X_test)
print("Evaluation f1 score {}".format(f1_score(y_test, y_knn_predict)
↪ ))
```

Por último, el código que permite visualizar la matriz de confusión es

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
cm = confusion_matrix(y_test, y_knn_predict, labels=knn.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=knn
↪ .classes_)
disp.plot()


plt.show()
```