UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE CIENCIAS PURAS Y NATURALES

CARRERA DE INFORMÁTICA

TESIS DE GRADO

DETECCIÓN DE MALWARE EN UNA RED CON MACHINE LEARNING

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA

MENCIÓN INGENIERÍA DE SISTEMAS INFORMÁTICOS

POSTULANTE: FERNANDO ROGER CORNEJO TARQUI

TUTOR METODOLÓGICO: M. Sc. ALDO VALDEZ ALVARADO

ASESOR: Ph. D. YOHONI CUENCA SARZURI

LA PAZ – BOLIVIA

2020

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

- **a)** visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
- **b)** copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
- **c)** copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

DEDICATORIA

A Dios, por permitirme el haber llegado hasta este momento tan importante de mi formación profesional.

A mi mamá, por ser el pilar más importante y por demostrarme siempre su cariño y apoyo incondicional sin importar nuestras diferencias de opiniones. A mi papá por compartir momentos significativos conmigo y por siempre estar dispuesto a escucharme y ayudarme en cualquier momento.

A mis hermanas por su cariño y apoyo incondicional, durante todo este proceso, y por estar conmigo en todo momento gracias.

AGRADECIMIENTOS

A mi familia, por haber sido mi apoyo durante todo este tiempo.

A mi tutor M. Sc. Aldo Ramiro Valdez Alvarado por haberme guiado durante cada etapa de la tesis, por sus valiosas observaciones y consejos.

A mi Asesor Ph. D. Yohoni Cuenca Sarzuri, por su predisposición e interés durante la revisión de cada etapa de la tesis vertiendo observaciones, consejos que fueron de mucha ayuda para mí y sobre todo la paciencia y su compresión.

RESUMEN

En esta tesis se analizan diferentes métodos para la detección de malware en una red, es decir, aquel flujo de tráfico de red que se encuentran en nuestro conjunto de datos pero que no tiene un comportamiento normal. Encontrar patrones en el flujo de trafico de red, sigue siendo un reto interesante en la seguridad informática. Las anomalías en el flujo del tráfico de red aparecerán por varias razones como actividades maliciosas o caídas, en este sentido el flujo del tráfico de red y una combinación con técnicas de Machine Learning, puede convertirse en un buen aliado para ir más allá de las firmas y ser capaz así de encontrar patrones previamente desconocidos. En cada predicción realizada por un diferente modelo será precisa o no tan precisa, esto dependerá de cómo el algoritmo se comporte a nuestro DataSet y se logre ver si el algoritmo cumple con la predicción necesaria a donde se pretende llegar, que es la detección de malware en la red. Los algoritmos de detección de anomalías que se van a estudiar son el covarianza robusta que trata de encontrar una elipse que contenga los puntos considerados como normales. El segundo es de las máquinas de vectores de soporte de una clase, se basa en la construcción de árboles binarios con la idea de que las anomalías se encuentren en hojas más cercanas a la raíz mientras que los puntos normales se ubiquen a mayor profundidad. El tercero de los bosques de aislamiento se basa en la construcción de árboles binarios con la idea de que las anomalías se encuentren en hojas más cercanas a la raíz mientras que los puntos normales se ubiquen a mayor profundidad. Finalmente el del factor de anomalías local busca asignar a cada uno de los datos una puntuación en función de la distancia a la que se encuentran sus vecinos, de tal manera que los puntos más aislados sean considerados como anomalías. Después de analizar y evaluar cada método de métodos de predicción, los resultados obtenidos no tan precisos en la predicción de detectar malware son el del el Covarianza Robusta, Vector de una Clase y por último el método que nos ofreció mejores resultados Bosques de Aislamiento cumpliendo con nuestro objetivo que es detectar el malware.

PALABRAS CLAVE: Malware, Anomalías, Trafico de Red y Algoritmo.

ABSTRACT

In this thesis different methods for the detection of malware in a network are analyzed, that is, the flow of network traffic that is in our data set but does not have normal behavior. Finding patterns in the flow of network traffic remains an interesting challenge in computer security. The anomalies in the flow of network traffic will appear for several reasons such as malicious activities or crashes, in this sense the flow of network traffic and a combination with Machine Learning techniques, can become a good ally to go beyond the signatures and being able to find previously unknown patterns. In each prediction made by a different model it will be precise or not so precise, this will depend on how the algorithm behaves to our DataSet and it is possible to see if the algorithm meets the necessary prediction where it is intended to arrive, which is the detection of malware In the net. The anomaly detection algorithms to be studied are the robust covariance that tries to find an ellipse that contains the points considered normal. The second is from the support vector machines of a class, it is based on the construction of binary trees with the idea that the anomalies are in leaves closer to the root while the normal points are located deeper. The third of the isolation forests is based on the construction of binary trees with the idea that the anomalies are found in leaves closer to the root while the normal points are located deeper. Finally, the one of the local anomaly factor seeks to assign to each one of the data a score based on the distance to which its neighbors are, in such a way that the most isolated points are considered as anomalies. After analyzing and evaluating each method of prediction methods, the results obtained not so precise in the prediction of detecting malware are that of the Robust Covariance, Vector of a Class and finally the method that offered us the best results. Insulation Forests complying with Our goal is to detect malware.

KEY WORDS: Malware, Anomalies, Network Traffic and Algorithm.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capitulo 1

Marco Referencial

1.1 Introducción

En la medida que las organizaciones informatizan sus procesos y se hacen más dependientes de las tecnologías, aumenta el impacto que la degradación de los servicios telemáticos y el incumplimiento de los niveles de operación, provocados por las fallas en las infraestructuras Tecnologías de la Información (TI) pueden provocar en las mismas. De ahí que la gestión de fallas debe realizarse de manera temprana, tomando rápidas decisiones correctivas, basadas en la evaluación del impacto de los daños en el equipamiento provoca en las organizaciones. Múltiples esfuerzos se realizan en torno a la gestión de fallas, que incluye acciones de monitoreo para la detección, aislamiento de estas y acciones de control para la corrección de problemas (Hassett, 2016).

Desde el inicio de las computadoras los intrusos informáticos han intentado aprovechar las debilidades o fallas en el sistema para causar diferentes ataques que pueden producir distintos tipos de daños en la información, software o el hardware y su modo de operación ha estado evolucionando a la par con la tecnología.

Entre las soluciones que han sido desarrolladas para la detección, localización y aislamiento de fallas en la red, se destaca el empleo de: máquinas de estado finito (Hassett 2016), métodos estadísticos, aproximaciones basadas en reglas, ajustes de patrones, redes neuronales, lógica fuzzy y teoría de grafos (Matsumoto y Chenaru, 2016).

El incremento de ataques informáticos hacia las redes, se ha notado incluso más con el avance de las nuevas tecnologías de la información y comunicación, con el objetivo de seguir intentando causar daños en contra de la integridad, disponibilidad y confidencialidad de la información, en tal caso se implementara un modelo de análisis en el trafico de red con el fin de localizar las actividades sospechosas.

Los resultados alcanzados a través de las mismas solo han proporcionado una vista local de la falla, y por lo tanto no pueden describirla, hasta que sus consecuencias son visibles, de ahí que sea necesario complementarlos con mecanismos para la localización, los cuales son responsables del análisis de las alarmas generadas por los componentes de la red proponiendo posibles hipótesis sobre la causa de la falla (Souza, 2016).

En la presente tesis, se pretende investigar y analizar métodos para la detección de malware en una red, con la finalidad de buscar algún indicio de un ataque conocido usando herramientas de Machine Learning; donde el analista demora en realizar éste análisis, pueda utilizar éste modelo y logré hacerla de manera automática.

1.2 Antecedentes

Encontrar patrones en los datos de red que no se ajusten al comportamiento esperado (anomalías), sigue siendo un reto interesante en la seguridad informática. Al pasar los años surgen propuestas para mitigar limitaciones y se centrará en la entrega de diferentes aproximaciones académicas que han tenido en cuenta Netflow y Machine Learning como centro del estudio. A continuación una breve información de trabajos que se realizaron estos últimos años.

En el 2012 se utilizaron máquinas de Boltzmann un tipo de red neuronal recurrente estocástica para detectar los eventos anómalos en la red. En la etapa de pre procesamiento usa Bro junto con una herramienta para analizar las trazas (Torrano, 2016).

En el 2014 el trabajo de Francesco Palmieri, Ugo Fiore y Aniello Castiglione utiliza Blind source separation para modelar Independent component analysis como primer paso. Luego utiliza árboles de decisión para la etapa de clasificación. Sin embargo, algunos trabajos sí comparan diferentes algoritmos y a partir de ahí pueden extraerse algunas conclusiones. Utilizando el algoritmo J48 proporcionando mejores resultados que BayerNet y OneR (Torrano, 2016).

El 2015 los resultados del estudio de Félix Iglesiasy Tanja Zseby reflejan que ANN obtiene mejores resultados que DTC, kNN, Bayes, SVM en el caso en el que se seleccionan dieciséis características de las cuarenta y uno que componen los conjuntos NSL-KDD o KDD (Torrano, 2017).

El 2016 se realiza una etapa de pre procesamiento aplicando clustering y extracción de datos. A continuación se utiliza como algoritmo incremental majority, que es una variación de redes neuronales, realizando varias iteraciones al utilizar una ventana deslizante. Posteriormente se refinan los límites que definen el intervalo de normalidad aplicando un algoritmo que llaman de detección de anomalías, escogiendo aquellos límites que optimizan la tasa de aciertos y minimizan los errores (Torrano, 2017).

El 2017 se implemento la inteligencia artificial/machine learning para detectar trafico anómalo/malware y análisis de sentimientos. Se muestra una topología típica de una empresa, donde se necesita generar tráfico Netflow hacia un servidor en cloud el cual recibe el netflow producido por la empresa, más el tráfico de un ramsoware que ha propósito se ha agregado. Para el servidor Colector de Netflow se ha instalado Nfdump que nos permite enviarle el tráfico generado localmente. Aplicando algoritmo de Machine Learning, él resultado y el sistema de aprendizaje se obtiene que reporta conexiones típicas y no típicas que en algunos casos pertenecen a conexiones a C&C (Command Control Servers) que son necesarios para descargarse la otra parte del malware faltante y que deberían ser bloqueadas (Farro, 2017).

Llama la atención que varios de estos trabajos, los mejores resultados utilizan variaciones de redes neuronales para la clasificación. Es destacable la adaptación de estos algoritmos ante muestras no vistas como ser el flujo del tráfico de una red, además de ser complejo e impredecible, está sujeto a cambios puesto que las anomalías continúan evolucionando. Otra notable mención el uso del protocolo NetFlow que ayuda a recopilar, agregar y registra datos de flujo de tráfico en una red; este protocolo fue desarrollado por Cisco y está integrado en el software IOS de los equipos de Cisco de los enrutadores y conmutadores.

1.3 Planteamiento del Problema

En los últimos años muchas soluciones han sido desarrolladas para la detección, localización y aislamiento de fallas en la red (Mohiuddin, 2016). La detección de fallas puede entenderse como una indicación en línea de que algún componente de la red está funcionando incorrectamente. Estos componentes solo tienen una vista local de la falla, y por lo tanto no pueden describir la falla, hasta que sus consecuencias son visibles (Kleinsteuber y Mohiuddin, 2016).

Los algoritmos para la detección de fallas están divididos en dos categorías; los basados en el patrón de la anomalía y los basados en el comportamiento normal de la red. Los basados en el patrón de la anomalía requieren tener un conocimiento previo sobre las fallas para su posterior modelación, lo que no siempre es posible debido a la propia complejidad de los entornos de red, de ahí los nuevos tipos de fallas que posiblemente ocurran sin que puedan ser detectadas (Chu, 2018).

En una red de computadoras, una única falla puede generar varias alarmas, lo que frecuentemente dificulta el aislamiento de la causa primaria de esta. Algunas de las técnicas que permiten la localización o aislamiento de la causa raíz de la falla son: los sistemas basados en reglas, la teoría de grafos y algoritmos que se desarrollan considerando la estructura de la red. Los sistemas basados en reglas presentan insuficiencias para la adaptación a nuevos problemas que no forman parte de la base de datos, lo que requiere el incremento constante de reglas y dificulta su manutención. Los grafos, que representan las relaciones de dependencia entre los componentes de la red y permiten la modelación de la propagación, pueden ser adicionados para mejorar la capacidad de diagnóstico y adaptación a nuevos problemas. La definición de un modelo propagación de anomalías no es sencillo y se pueden ignorar anomalías no previstas durante el diseño (Matsumoto y Chenaru, 2016).

1.4 Problema Central

¿Cómo detectar de manera automática malware en una red?

1.5 Problemas Secundarios

- Desafortunadamente, el número de amenazas crece exponencialmente y cada una es más compleja que la anterior, lo que hace muy aburrido y agotador la detección de un alto porcentaje de ellas.
- La captura del flujo de tráfico de red de un solo equipo es sencillo, pero capturar el flujo de tráfico de red de toda una red donde existen más de 4 o 5 equipos ses más complicado.
- Cuando el flujo del tráfico de la red comienza a subir, se hace una tarea imposible de detectar toda la actividad no autorizada siguiendo un conjunto de reglas.
- Si la cantidad de datos del flujo de red no tiene un número suficiente de casos es decir es de tamaño reducido, será dificultoso distinguir los datos que fueron infectados por el malware de los datos que tienen un comportamiento normal.
- Si bien el malware es infectado en toda una red desde un destino el cual va cambiando, se hace una tarea difícil de identificar el equipo que ha sido infectado por el malware.

1.6 Definición de Objetivos

1.6.1 Objetivo General

Plantear un modelo que detecte de manera automática malware en una red.

1.6.2 Objetivos Específicos

- Automatizar la búsqueda de nuevos patrones de ataque, gracias a herramientas estadísticas de búsqueda, y al análisis de tráfico anómalo.
- Monitorizar y analizar las actividades de los usuarios de modo que se puedan conocer los servicios que usan y el contenido del tráfico.

- Automatizar las tareas como la actualización de reglas, la obtención y análisis de datos.
- Obtener tantos conjuntos de datos como sea posible, procedentes de trafico de red con un número sustancial de filas y columnas.
- Establecer cuáles son las técnicas de Machine Learning que se pueden usar para realizar la clasificación del tráfico anómalo y del tráfico normal.

1.7 Hipótesis

El uso de algoritmos de Machine Learning permite detectar Malware en una red, con una confiabilidad del 90%.

1.8 Operacionalización de Variables

- **Variable independiente X:** Detección de Malware en una red.
- **Variable dependiente Y:** Algoritmos de Machine Learning.

1.9 Justificación

Este modelo ayudará en la detección de malware, funcionará por medio del tráfico de una red donde los datos serán analizados, como si una persona o un equipo de trabajo analizara dichos datos pero en realidad será un algoritmo de Machine Learning.

1.9.1 Justificación Económica

Reducción de tiempo y costo. El modelo de aprendizaje automático es capaz de detectar varias anomalías en el mismo plazo de tiempo en el que un experto en datos tarda en detectar uno solo. Además puedes ahorrar dinero al invertir en una tecnología que te permita ofrecer un servicio todos los días de la semana que no dependa constantemente de la intervención humana ni de un gran equipo de trabajo.

1.9.2 Justificación Social

La detección de actividades sospechosas ofrece un apoyo para la empresa al disponer de los datos adecuados para una mejor toma de decisiones, además de colaborar en la integración de una mejor ciberseguridad de la empresa. Los administradores y usuarios en general podrán fomentar una educación acerca de los tipos de ataques informáticos y vulnerabilidades a las que puede ser propensa una red.

1.9.3 Justificación Científica

Concretamente será usada en seguridad informática para la detección de amenazas avanzadas y presenta la ventaja de poder identificar malware nunca antes visto, donde los sistemas tradicionales fallan al basar la detección únicamente en firmas o patrones conocidos.

1.10 Alcances y Límites

1.10.1 Alcances

- Detectar una conexión fuera de hora, reintentos de conexión fallidos y otros, porque existe la posibilidad de que se esté en presencia de una intrusión.
- Obtendrá información de otros sistemas como firewalls, routes y switches, lo que permitirá tener un análisis detallado del tráfico de red.
- Detectar la anomalía por la IP de origen, protocolo, puerto de destino y tiempo de inicio del flujo.
- Evaluar las direcciones IP que han intervenido en estos ataques como origen y destino, donde estos eventos se realizaron en una inundación de red.
- Automatizar la búsqueda de nuevos patrones de ataque, gracias a herramientas de búsqueda y al análisis de tráfico anómalo.

- Visualizar el conjunto de datos (dataset) permitiendo encontrar y analizar detalles, tendencias y problemas que a simple vista no encontraríamos.

1.10.2 Límites

- No será posible detectar ataques furtivos; este tipo de ataques se encuentran ocultos en una gran cantidad de comportamiento normal.

- Seguridad reactiva, se refiere que para detectar debe existir primero un ataque. De manera que si un ataque es pequeño y efectivo el modelo reaccionará demasiado tarde y el ataque logrará su objetivo.

- No es posible capturar el trafico de una red virtual privada (VPN) debido que utiliza algoritmos de cifrado simétrico para las transmisión de datos.

- La manipulación de datos y con su respectiva autorización solo lo hace una Proveedor de servicios de internet IPS, en cambio el modelo de detección de anomalías solo monitorea flujos de TCP, UDP y entre otros.

- El modelo de seguridad pasiva se limita únicamente a detectar las anomalías y el administrador de red es quien deberá tomar acciones oportunas.

1.11 Aportes

1.11.1 Práctico

Él modelo pretende brindar el prototipo de una herramienta de gestión que ayude a analizar el tráfico de una red y permita detectar anomalías a partir de la información obtenida de este análisis; además se tienen los paquetes de datos y la información del flujo de tráfico lo que resulta muy útil. Todo esto tendrá éxito si se puede detectar cualquier posible alteración y tomar buenas decisiones tan rápido como sea posible.

1.11.2 Teórico

El modelo consiste en utilizar machine Learning, donde las técnicas de aprendizaje no supervisado serán un punto de partida. Se detectara un ataque capturando y analizando paquetes de red. Escuchando en un segmento de red, donde podrá monitorizar el tráfico de red que afecta a múltiples hosts que están conectados a ese segmento de red, la monitorización del tráfico de red, se realizara mediante un análisis local de este tráfico.

1.12 Metodologías

1.12.1 Metodología de investigación

El modelo tiene por objeto el estudio de todo el flujo del tráfico de una red, este análisis será observable, medible y se podrá adquirir de manera precisa. Por tanto se utilizara la metodología cuantitativa; para la evaluación de cada método de predicción que se implementara, se procederá mediante la utilización de las estadísticas y la identificación de variables. Con la finalidad de validar la hipótesis basado en la práctica, experiencia y en la observación de los hechos.

1.12.2 Metodología de Data Science

El modelo se apoyara en la metodología CRISP-DM, primero se trabajara en acceder a tener datos de fuentes originales, luego se garantizara que éstos tengan la calidad suficiente como para generar conclusiones y resultados. Se estructurara los datos y se creara nuevas variables, luego de toda esta transformación de los datos se pasara a la modelización; que consiste en identificar y aplicar el mejor algoritmo analítico para cumplir con el objetivo. Llegando a la parte final donde se evaluara la calidad del modelo para concluir con la implantación del modelo.

Capitulo 2

Marco Teórico

2.1 Seguridad Informática

La seguridad informática hace referencia a todas las medidas y controles que se deben establecer para el aseguramiento de los sistemas informáticos, impidiendo que intrusos internos o externos realicen procedimientos no autorizados sobre la red. La seguridad informática comprende la protección de (Rascagneres, 2016):

- **Aplicaciones:** Para evitar algún problema de seguridad informática las aplicaciones se deben de descargar en fuentes confiables y actualizarlas frecuentemente.
- **Comunicaciones:** Es necesario evitar la interceptación de la comunicaciones haciendo uso de canales de comunicación cifrada, además de tener un constante control de las conexiones para impedir conexiones no autorizadas.
- **Datos:** Para mantener protegidos los datos es necesario realizar constantemente copias de seguridad, el cifrado de la información, realizar un almacenamiento redundante de datos y deshabilitar componentes que supongan la entrada o salida de información no autorizada.
- **Equipos:** Para evitar el robo de equipos se debe de cifrar los contenidos críticos, controlar o evitar los intentos de conexión de equipos externos no autorizados, y realizar mantenimientos preventivos.

2.1.1 Principios Básicos

La seguridad informática trata de mantener seguro la: Integridad, Privacidad, Disponibilidad, Control y Autenticidad de la información, que se encuentra almacenada en una computadora.

En la Enciclopedia de la Seguridad informática, el autor Gómez Vieites Álvaro explica la definición de los siguientes conceptos (Gómez, 2011):

- **Integridad:** Se encarga de garantizar que un mensaje o fichero no sea modificado desde su creación o durante su transmisión a través de una red informática hasta llegar al receptor.

- **Confidencialidad:** Es la necesidad de garantizar que cada mensaje transmitido o almacenado en un sistema informático solo podrá ser leído por su legítimo destinatario. Si dicho mensaje cae en manos de terceras personas, estas no podrán tener acceso al contenido del mensaje original.

- **Disponibilidad:** Se encarga de garantizar la permanente disposición de los servicios a los que los usuarios deseen acceder.

- **Control:** Permite asegurar que sólo los usuarios autorizados puedan decidir cuándo y cómo permitir el acceso a los servicios o información.

- **Autenticidad:** Se encarga de garantiza que la identidad del creador de un mensaje o documento sea legítima.

Para poder definir a un sistema informático como seguro debe de cumplir con los tres principios básicos de la seguridad de la información de acuerdo al estándar ISO 27002 (ISO, 2017), es decir, mantener segura la integridad, confidencialidad y disponibilidad de la información, como se muestra en la Figura 2.1. De todas formas, como en la mayoría de los ámbitos de la seguridad, lo esencial sigue siendo la capacitación de los usuarios. Una persona que conoce cómo protegerse de las amenazas sabrá utilizar sus recursos de la mejor manera posible para evitar ataques o accidentes. En otras palabras, puede decirse que la seguridad informática busca garantizar que los recursos de un sistema de información sean utilizados tal como una organización o un usuario lo ha decidido, sin intromisiones.

Figura 2.1. Principios básicos de la seguridad de la información.
Fuente: (ISO, 2017)

La finalidad de la seguridad informática es proteger el almacenamiento, procesamiento y transmisión de la información digital en cualquier tipo de red informática para que no pueda ser interceptada por ningún tipo de intruso informático (Roa, 2013).

2.2 Malware

Según definición, malware es una abreviación de las palabras malicious software, software malicioso. Esto significa que el software está diseñado y creado para causar daño a un dispositivo o a su usuario. Es un término general usado para clasificar archivos o software que causan daños una vez que entran en su sistema. Este daño se puede manifestar de muchas formas, a menudo implica robo de datos del ordenador del usuario, encriptar esos datos o simplemente eliminarlos. El malware también tiene la capacidad de cambiar funciones dentro del ordenador o tomar el control sobre él por completo, como ocurre con los botnets y los rootkits (Moes, 2019).

El malware tiene por objetivo el infiltrarse dentro de un sistema computacional y una vez llevar a cabo una serie de actividades encubiertas, que van desde el sabotaje del propio sistema, el robo de datos confidenciales, la apropiación de sus recursos informáticos y/o el contagio de otros sistemas que puedan estar en red.

2.2.1 Tipos de Malware

Dependiendo de la intención de su creador, el malware puede oscilar desde ser un software muy sofisticado, capaz de realizar numerosas funciones, hasta ser simplemente una molestia. Existen muchos tipos de malware, cuyas diferencias se basan en sus elementos o modo de operar. Algunos de estos son (Moes, 2019):

- **Virus Informático:** Un virus informático es la forma clásica de malware. Es un componente de código o programa que entra en su dispositivo sin que usted lo sepa. Una vez allí, puede causar una serie de daños, desde ralentizar su sistema, deshabilitar partes específicas o tomar el control por completo. Igual que con los virus biológicos, está diseñado para expandirse automáticamente a través de redes y dispositivos.
- **Spyware:** Estos son malware diseñados para recopilar datos del ordenador y sus usuarios. Lo hace con infiltración en el ordenador del usuario y monitorización de sus actividades. Se instala en el ordenador del usuario directamente o mediante explotación de huecos en la ciberseguridad.
- **Ransomware:** Tal y como su nombre indica, el ransomware es un software creado con el propósito de secuestrar datos del ordenador del usuario. El software está diseñado para encriptar los datos delicados del objetivo. Entonces los creadores exigen dinero al usuario para desencriptar los datos.
- **Troyano Informático:** Este tipo de malware se ha creado para parecer un programa normal. Tanto es así que convence a los usuarios inconscientes para que lo instalen en su ordenador. Una vez instalado y ejecutado, el caballo de Troya puede empezar a realizar la función maliciosa para la que fue creado. Al contrario que los virus y los gusanos, los caballos de Troya rara vez intentan reproducirse y expandirse.
- **Rootkit:** Este tipo de malware es creado para brindar a los cibercriminales permisos del nivel de administrador en el equipo objetivo. Este acceso les permite modificar el sistema del ordenador del usuario. Además, se usa para ocultar la presencia de otro malware en el sistema del ordenador.

- **Virus Backdoor:** Este tipo de malware crea una entrada secreta dentro del ordenador del objetivo. A través de esta puerta de atrás, los cibercriminales tienen la capacidad de acceder al ordenador sin el conocimiento del usuario. Los backdoors son creados por otros tipos de malware, como gusanos o caballos de Troya. Con el uso de backdoor, los cibercriminales también eluden los programas de seguridad del ordenador. Un tipo de virus backdoor es el Troyano de Acceso Remoto (RAT).

2.2.2 Función del Malware

La propagación de malware depende ampliamente de la intención del creador. Para muchos virus y gusanos, la expansión se lleva a cabo con la intención de llegar al mayor número de ordenadores posible. Por definición, la infección se produce cuando los datos se comparten. Esto puede pasar en internet con archivos descargados, archivos adjuntos de emails, enlaces maliciosas o descargas ocultas que se completan sin el conocimiento del usuario.

También ocurre cuando la gente comparte archivos fuera de la red con ordenadores infectados o cuando comparten algunos archivos multimedia. Las infecciones personales en ocasiones se llevan a cabo cuando se usa un puerto USB que contiene malware. A menudo pasa cuando se instalan backdoors o rootkits que permiten a los creadores acceso remoto o acceso de administrador al ordenador de la víctima (Moes, 2019).

Los avances en ciberseguridad a menudo van de la mano con los avances en malware. Las nuevas capas están programadas con técnicas más sofisticadas para evadir la detección de los programas antimalware así como los usuarios de los ordenadores. Estas técnicas van desde tácticas sencillas, como el uso de proxies de red para ocultar las IP de los creadores, hasta formas más sofisticadas de malware independiente. En el segundo caso, el malware evita la detección al ocultarse dentro del sistema RAM. El malware también aprovecha las vulnerabilidades en la seguridad del ordenador. Lo hacen al explotar las similitudes en sistemas operativos para infectar múltiples sistemas. Dicho de otro modo, explotan los defectos del software de seguridad (Moes, 2019).

2.3 Ransomware

El ransomware es un programa de software malicioso que infecta tu computadora y muestra mensajes que exigen el pago de dinero para restablecer el funcionamiento del sistema. Este tipo de malware es un sistema criminal para ganar dinero que se puede instalar a través de enlaces engañosos incluidos en un mensaje de correo electrónico, mensaje instantáneo o sitio web. El ransomware tiene la capacidad de bloquear la pantalla de una computadora o cifrar archivos importantes predeterminados con una contraseña (Kaspersky, 2019).

2.3.1 Scareware

El scareware es el tipo más simple de ransomware. En concreto, utiliza tácticas de amedrentamiento o intimidación para hacer que las víctimas paguen. Este tipo de malware puede adoptar la forma de un programa de software antivirus que muestra un mensaje en el que se informa que la computadora tiene varios problemas y el usuario debe efectuar un pago en línea para corregirlos (Kaspersky, 2019).

El nivel de este tipo de ataque es variable. En ocasiones, abruma a los usuarios con alertas y mensajes emergentes interminables. En otros, la computadora deja de funcionar por completo. Existe otro tipo de ransomware que se hace pasar por una fuerza de seguridad y abre una página aparentemente perteneciente a la oficina de un organismo de seguridad. Aparece un mensaje que afirma que el usuario de la computadora fue atrapado realizando actividades ilegales en línea. A continuación, los archivos se bloquean con cifrados complejos difíciles de recuperar por los usuarios a menos que paguen un rescate.

Los ataques típicos suelen pedir montos de $100 a $200. Otros ataques son mucho más ambiciosos, especialmente si el atacante es consciente de que los datos que capturó pueden causar pérdidas financieras directas y considerables a una empresa. Como resultado, los cibercriminales que orquestan estas estafas pueden ganar mucho dinero.

Independientemente de la situación, incluso si el usuario paga el rescate, no existe garantía de que volverá a acceder completamente a sus sistemas. Aunque algunos hackers indican a

las víctimas que deben pagar a través de Bitcoin, MoneyPak u otros métodos en línea, los atacantes también pueden exigir información de tarjetas de crédito, lo que supone otra vía de pérdida financiera (Kaspersky, 2019).

2.4 Data Science (Ciencia de Datos)

Data Science es la ciencia centrada en el estudio de los datos. Se encarga de extraer información de grandes cantidades de datos. Data Science combina la estadística, las matemáticas y la informática para interpretar datos. El objetivo es tomar decisiones. Estos datos se obtienen a través de diferentes canales. Los teléfonos móviles, las redes sociales y las encuestas son solo algunas de las fuentes utilizadas. Nuestros gustos, rutinas o movimientos generan datos de gran valor para las empresas que quieren conocer a sus clientes al detalle. Sin embargo, la interpretación de los datos no estructurados no aporta valor a las compañías. De ahí surge la necesidad de contar con científicos de datos en sus equipos. Gracias al Data Science las empresas pueden anticiparse a la hora de tomar decisiones (Neoland, 2019).

El término Data Science ha estado presente durante las últimas tres décadas. Pero no fue hasta la década de los 70 cuando el término se comenzó a usar para definir los métodos de procesamiento de datos. Finalmente, 2001 fue el año en el que la ciencia de datos se introdujo como una disciplina independiente (Neoland, 2019).

2.4.1 Data Mining (Minería de Datos)

Data Mining o Knowledge Discovery in Databases (KDD) consiste en extraer información de un conjunto de datos y transformarla en una estructura comprensible, esto es, en información útil y accesible para que pueda ser usada posteriormente. Data Mining es un término que está de moda y que se utiliza, en muchas ocasiones, de manera incorrecta. Algunas de las referencias erróneas que se le atribuyen son que se trata de cualquier forma de datos a gran escala, o de procesamiento de la información (Licona, 2014).

Según Arturo Licona, Data Mining es un conjunto de técnicas de extracción de datos para detectar patrones de comportamiento a través de algoritmos matemáticos especialista de Deloitte Consulting Group (Licona, 2014).

La tendencia a acumular grandes cantidades de datos en las organizaciones continúa hoy en día a un ritmo creciente. A ello han contribuido, entre otros, las nuevas técnicas de captura de datos y el avance de la tecnología de almacenamiento de datos y su reducción de costes. La disponibilidad posterior de esos datos es también sumamente importante y es donde entra en juego el concepto de Data Mining (Fractalia, 2015).

La enorme cantidad de datos que se generan y procesan a diario en las organizaciones lleva de la mano la necesidad de analizarlos en tiempo real, el problema es que los métodos tradicionales de análisis de datos no son suficientes. De ahí surgen los conceptos de Data Mining y Big Data. Ambos se refieren al hecho de saber aprovechar convenientemente las gigantescas masas de información de que disponen las organizaciones en su día a día, para extraer información útil que les ayude en la toma de decisiones (Fractalia, 2015).

Cada día generamos una gran cantidad de información, algunas veces conscientes de que lo hacemos y otras veces inconscientes de ello porque lo desconocemos. Nos damos cuenta de que generamos información cuando registramos nuestra entrada en el trabajo, cuando entramos en un servidor para ver nuestro correo, cuando pagamos con una tarjeta de crédito o cuando reservamos un billete de avión. Otras veces no nos damos cuenta de que generamos información, como cuando conducimos por una vía donde están contabilizando el número de automóviles que pasan por minuto, cuando se sigue nuestra navegación por Internet o cuando nos sacan una fotografía del rostro al haber pasado cerca de una oficina gubernamental. ¿Con qué finalidad se quiere generar información? Son muchos los motivos que nos llevan a generar información, ya que nos pueden ayudar a controlar, optimizar, administrar, examinar, investigar, planificar, predecir, someter, negociar o tomar decisiones de cualquier ámbito según el dominio en que nos desarrollemos (Molina, 2002).

La información por sí misma está considerada un bien patrimonial. De esta forma, si una empresa tiene una pérdida total o parcial de información provoca bastantes perjuicios. Es evidente que la información debe ser protegida, pero también explotada. ¿Qué ha permitido poder generar tanta información? En los últimos años, debido al desarrollo tecnológico a niveles exponenciales tanto en el área de cómputo como en la de transmisión de datos, ha sido posible que se gestionen de una mejor manera el manejo y almacenamiento de la información. Sin duda existen cuatro factores importantes que nos han llevado a este suceso (Molina, 2002):

- El abaratamiento de los sistemas de almacenamiento tanto temporal como permanente.
- El incremento de las velocidades de cómputo en los procesadores.
- Las mejoras en la confiabilidad y aumento de la velocidad en la transmisión de datos.
- El desarrollo de sistemas administradores de bases de datos más poderosos.

Actualmente todas estas ventajas se han llevado a abusar del almacenamiento de la información en las bases de datos. Se puede decir que algunas empresas almacenan un cierto tipo de datos al que hemos denominado dato-escritura, ya que sólo se guarda en el disco duro, pero nunca se hace uso de él. Generalmente, todas las empresas usan un dato llamado dato-escritura-lectura, que utilizan para hacer consultas dirigidas.

Un nuevo tipo de dato al cual se denomino dato-escritura-lectura-análisis es el que proporciona en conjunto un verdadero conocimiento y apoya en las tomas de decisiones. Es necesario contar con tecnologías que ayuden a explotar el potencial de este tipo de datos (Molina, 2002).

El Data Mining es una tecnología compuesta por etapas que integra varias áreas y que no se debe confundir con un gran software. Durante el desarrollo de un proyecto de este tipo se usan diferentes aplicaciones software en cada etapa que pueden ser estadísticas, de visualización de datos o de inteligencia artificial, principalmente.

2.5 Metodología CRISP-DM

CRISP-DM, que son las siglas de Cross Industry Standard Process for Data Mining, es un método probado para orientar sus trabajos de minería de datos. Las siglas en español significa Proceso Estándar de la Industria Cruzada para Minería de Datos. En la siguiente Figura 2.2 se puede apreciar el ciclo de vida del CRISP-DM.

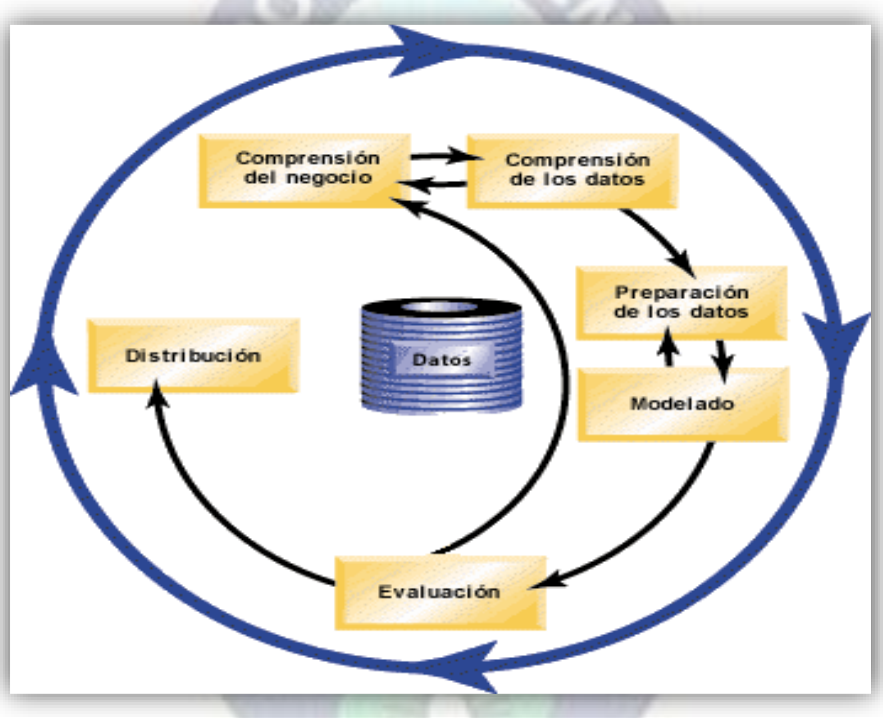


Figura 2.2. Ciclo de vida del CRISP-DM.
Fuente: (IBM, 2012)

El ciclo vital del modelo contiene seis fases con flechas que indican las dependencias más importantes y frecuentes entre fases. La secuencia de las fases no es estricta. De hecho, la mayoría de los proyectos avanzan y retroceden entre fases si es necesario.

- Como metodología, incluye descripciones de las fases normales de un proyecto, las tareas necesarias en cada fase y una explicación de las relaciones entre las tareas.

- Como modelo de proceso, CRISP-DM ofrece un resumen en función de todas sus fases de su ciclo vital de minería de datos.

El modelo de CRISP-DM es flexible y se puede personalizar fácilmente. Por ejemplo, si una organización intenta detectar actividades de blanqueo de dinero, es probable que necesite realizar una criba de grandes cantidades de datos sin un objetivo de modelado específico. En lugar de realizar el modelado, el trabajo se centrará en explorar y visualizar datos para descubrir patrones sospechosos en datos financieros. CRISP-DM permite crear un modelo de minería de datos que se adapte a sus necesidades concretas.

En tal situación, las fases de modelado, evaluación e implementación pueden ser menos relevantes que las fases de preparación y comprensión de datos. Sin embargo, es muy importante considerar algunas cuestiones que surgen durante fases posteriores para la planificación a largo plazo y objetivos futuros de minería de datos (IBM, 2012).

2.5.1 Comprensión del negocio

Primero se debe dedicar tiempo a explorar las expectativas de la organización con respecto a la minería de datos. Intente implicar a la mayor cantidad de personas que sea posible en estas discusiones y documente los resultados. El paso final de la fase de CRISP-DM trata de cómo producir un plan de proyecto utilizando la información que se contiene en esta documentación. Aunque este estudio pueda parecer prescindible, no lo es. Conozca las razones comerciales para que sus esfuerzos en minería de datos aseguren que todos los usuarios están de acuerdo antes de asignar recursos (IBM, 2012).

Para la determinación de los objetivos comerciales, su primera tarea es obtener la máxima información posible de los objetivos comerciales de la minería de datos. Es posible que esta tarea no sea tan fácil como parece, pero puede reducir los futuros riesgos clarificando los problemas, objetivos y recursos. La metodología CRISP-DM proporciona una forma estructurada de alcanzar estos objetivos (IBM, 2012).

- Comience a recopilar información acerca de la situación comercial actual.
- Registre los objetivos comerciales específicos que decidan los gerentes.

- Consensue los criterios que se utilizarán para determinar el rendimiento del proceso de minería de datos desde una perspectiva comercial.

2.5.2 Comprensión de los datos

La fase de comprensión de datos de CRISP-DM implica estudiar más de cerca los datos disponibles de minería. Este paso es esencial para evitar problemas inesperados durante la siguiente fase (preparación de datos) que suele ser la fase más larga de un proyecto. La comprensión de datos implica acceder a los datos y explorarlos con la ayuda de tablas y gráficos que se pueden organizar. De esta forma podrá determinar la calidad de los datos y describir los resultados de estos pasos en la documentación del proyecto.

Para la recopilación de datos iníciales, los datos provienen de diferentes fuentes como:

- Datos existentes. Incluye una amplia variedad de datos, como datos transaccionales, datos de encuesta, registros Web, etc. Tenga en cuenta si los datos existentes son suficientes para adaptarse a sus necesidades.
- Datos adquiridos. ¿Su organización utiliza datos adicionales, como datos demográficos? Si no los utiliza, considere si son necesarios.
- Datos adicionales. Si las fuentes anteriores no satisfacen sus necesidades, es posible que necesite realizar encuestas o realizar seguimientos adicionales para servir de complemento a los almacenes de datos actuales.

2.5.3 Preparación de datos

La preparación de datos es uno de los aspectos más importantes y con frecuencia que más tiempo exigen en la minería de datos. De hecho, se estima que la preparación de datos suele llevar el 50-70 % del tiempo y esfuerzo de un proyecto. Dedicar los esfuerzos adecuados a las primeras fases de comprensión comercial y comprensión de datos puede reducir al mínimo los gastos indirectos relacionados, pero aún deberá dedicar una buena cantidad de esfuerzo para preparar y empaquetar los datos para la minería (IBM, 2012).

Dependiendo de su organización y sus objetivos, la preparación de datos suele implicar las tareas siguientes:

- Fusión de conjuntos y/o registros de datos.
- Selección de una muestra de un subconjunto de datos.
- Agregación de registros.
- Derivación de nuevos atributos.
- Clasificación de los datos para el modelado.
- Eliminación o sustitución de valores en blanco o ausentes.
- División en conjuntos de datos de prueba y entrenamiento.

En la selección de datos, será en función de la recopilación de datos inicial realizada en la fase CRISP-DM anterior, ahora puede comenzar a seleccionar los datos relevantes a sus objetivos de minería de datos. De forma general, existen dos formas de seleccionar datos:

- Selección de elementos (filas) implica la toma de decisiones como las cuentas, productos o clientes que se van a incluir.
- Selección de atributos o características (columnas) implica la toma de decisiones sobre el uso de características como la cantidad de las transacciones o los ingresos por hogar.

2.5.4 Modelado

Este es el punto donde todo el duro trabajo anterior comienza a tener sentido. Los datos que ha preparado se incorporan a las herramientas analíticas del modelo y los resultados comenzarán a arrojar algo de luz al problema planteado en Comprensión del negocio.

El modelado se suele ejecutar en múltiples iteraciones. Normalmente, los analistas de datos ejecutan varios modelos utilizando los parámetros por defecto y ajustan los parámetros o

vuelven a la fase de preparación de datos para las manipulaciones necesarias por su modelo (IBM, 2012).

Es extraño que las cuestiones relativas a la minería datos de una empresa se solucionen satisfactoriamente con un modelo y ejecución únicos. Esto es lo que hace la minería de datos tan interesante; existen muchas formas para resolver un problema concreto y SPSS Modeler ofrece una amplia variedad de herramientas para ello.

Para la Selección de técnicas de modelado, aunque pueda tener algunos conocimientos acerca de los tipos de modelado que sean los más adecuados para las necesidades de su organización, es el momento de tomar la decisión de los tipos de modelado que se van a utilizar. La determinación del modelado más adecuado se basará en las siguientes consideraciones (IBM, 2012):

- Los tipos de datos disponibles para la minería. Por ejemplo, ¿los campos de interés son categóricos (simbólicos)?
- Sus objetivos de minería de datos. ¿Sólo quiere tener un mejor conocimiento de los almacenes de datos transaccionales y descubrir patrones de compras interesantes? ¿Necesita producir una puntuación indicando, por ejemplo, las posibilidades de impago de un préstamo a un estudiante?
- Requisitos específicos de modelado. ¿Necesita el modelo un tipo o un tamaño de datos concreto? ¿Necesita un modelo con unos resultados fácilmente presentables?

2.5.5 Evaluación

En este punto, se habrá completado la mayor parte del proyecto de minería de datos. También habrá determinado, en la fase de modelado, que los modelos son técnicamente correctos y efectivos en función de los criterios de rendimiento de minería de datos que ha definido previamente. Sin embargo, antes de continuar, debe evaluar los resultados de sus esfuerzos utilizando los criterios de rendimiento comercial establecidos en el inicio del

proyecto. Es la clave para asegurar que su organización pueda utilizar los resultados que ha obtenido. La minería de datos produce dos tipos de resultados (IBM, 2012):

- Los modelos finales seleccionados en la fase anterior de CRISP-DM.
- Las conclusiones obtenidas de los modelos y del proceso de minería de datos.

2.6 Protocolos

Se explica la forma en que los procesos de establecimiento y finalización de sesión TCP promueven una comunicación confiable. También la forma en que se transmiten y se reconocen las unidades de datos del protocolo TCP para garantizar la entrega, así como los procesos de cliente UDP para establecer la comunicación con un servidor (Cisco, 2018).

2.6.1 Protocolo de TCP

La función del protocolo de transporte TCP es similar al envío de paquetes de los que se hace un seguimiento de origen a destino. Si se divide un pedido de envío en varios paquetes, el cliente puede revisar en línea el orden de la entrega (Cisco, 2018).

- Existen tres operaciones básicas que habilitan la confiabilidad con TCP.
- La numeración y el seguimiento de los segmentos de datos transmitidos hacia un host específico desde una aplicación determinada.
- El acuse de recibo de datos
- La retransmisión de cualquier dato sin acuse de recibo después de un período determinado.

2.6.1.1 Características de TCP

- Orientado a la conexión: crea una sesión entre el origen y destino.
- Entrega confiable: retransmite datos perdidos o dañados.
- Reconstrucción de datos ordenada: numeración y secuenciación de segmentos para rearmar datos en el orden correcto.

- Control del flujo: regula la cantidad de datos que se transmiten ya que los hosts de red tienen recursos limitados, como la memoria o la capacidad de procesamiento.
- Protocolo con estado: realiza un seguimiento de la sesión.

2.6.2 Protocolo UDP

UDP proporciona las funciones básicas para distribuir segmentos de datos entre las aplicaciones adecuadas, con muy poca sobrecarga y comprobación de datos. Si bien las funciones de confiabilidad de TCP proporcionan una comunicación más sólida entre aplicaciones, también representan una sobrecarga adicional y pueden provocar demoras en la transmisión. Existe una compensación entre el valor de la confiabilidad y la carga que implica para los recursos de la red. Agregar sobrecarga para garantizar la confiabilidad para algunas aplicaciones podría reducir la utilidad a la aplicación e incluso ser perjudicial (Cisco, 2018).

2.6.2.1 Características de UDP

- Sin conexión.
- Menos demoras en la transmisión.
- No hay reconstrucción de datos ordenada (lo realiza la aplicación).
- Entrega poco confiable.
- Sin control del flujo.
- Protocolo sin estado, ni el cliente ni el servidor están obligados a hacer un seguimiento del estado de la sesión de comunicación.

2.6.3 WireShark

Wireshark se ha convertido en una bendición para cualquier administrador de sistemas o profesional de seguridad. Se trata de un software gratuito que permite analizar el tráfico red en tiempo real. Pero su particularidad es que a menudo es la mejor herramienta para solucionar los problemas de Red como la latencia o actividad maliciosa como intentos de

piratería. No obstante, se habla de una aplicación técnica, que requiere conocimientos de los conceptos básicos de redes. Para la mayoría de las empresas, eso significa comprender la pila de TCP/IP, cómo leer e interpretar los encabezados de los paquetes y cómo funcionan el enrutamiento, el reenvío de puertos y el DHCP (CSO, 2018).

2.6.3.1 Funcionamiento de WireShark

La herramienta intercepta el tráfico y lo convierte en un formato legible para las personas. Esto hace que sea más fácil identificar qué tráfico está cruzando la red, con qué frecuencia y la latencia que hay entre ciertos saltos. Si bien Wireshark admite más de 2.000 protocolos de red, muchos de ellos inusuales o antiguos, los profesionales encuentran una gran utilidad en el análisis de identidades IP. La mayoría de los paquetes son TCP, UPD e ICMP.

Dado el gran volumen de tráfico que atraviesa una red comercial típica, las utilidades de Wireshark ayudan a filtrarlo. Los filtros de captura solo recopilan los tipos de tráfico que le interesan al comercio y los de visualización le ayudan a acercarse al tráfico que quiere inspeccionar. El analizador de protocolo de red proporciona herramientas de búsqueda, que incluyen expresiones regulares y resaltado en color, para que sea más fácil encontrar lo que se está buscando.

Wireshark existe desde 1998. Fue ideado por Gerald Combs y, en un principio, llamado Ethereal. A lo largo de los años, ha recibido grandes cantidades de soporte y parches de la comunidad, y es ampliamente aceptado como el analizador de protocolo de red de facto disponible en la actualidad. Se ejecuta en todos los sistemas operativos más importantes y más pequeños, incluidas las distribuciones usuales de Linux, Windows, OS X, FreeBSD, NetBSD y OpenBSD (WireShark, 2019).

2.7 Inteligencia Artificial

La Inteligencia Artificial ha existido durante mucho tiempo, desde los mitos griegos que contienen historias de hombres mecánicos diseñados para imitar nuestro propio

comportamiento. Las primeras computadoras europeas fueron concebidas como máquinas lógicas y mediante la reproducción de capacidades como la aritmética básica y la memoria, los ingenieros vieron una oportunidad de intentar crear cerebros mecánicos.

A medida que la tecnología y, sobre todo, nuestra comprensión de cómo funcionan nuestras mentes ha progresado, nuestro concepto de lo que constituye la Inteligencia Artificial ha cambiado. En lugar de realizar cálculos cada vez más complejos, el trabajo en el campo de la Inteligencia Artificial se ha concentrado en imitar los procesos en la toma de decisiones de los seres humanos y llevar a cabo las tareas de maneras cada vez más humanas.

Dentro de Inteligencia Artificial podemos clasificar en dos grupos fundamentales: aplicada o general. La aplicada es mucho más común: los sistemas diseñados para negociar inteligentemente acciones o maniobrar un vehículo autónomo caerían en esta categoría. Los generalizados son sistemas o dispositivos que en teoría pueden manejar cualquier tarea, son menos comunes, pero aquí es donde están sucediendo algunos de los avances más emocionantes de hoy. También es el área que ha llevado al desarrollo del Aprendizaje Automático (Analytics10, 2018).

La inteligencia artificial es la tecnología más avanzada en lo que al tratamiento del dato se refiere. Los coches autónomos y robots inteligentes son, entre otros, los campos de investigación en los que la inteligencia artificial juega un papel fundamental. También en la analítica web y en la omnicanal, ya son varias las empresas que automatizan tareas de lanzamiento de campañas personalizadas a cada usuario apoyándose en el aprendizaje de las computadoras (Martinez, 2018).

El término Inteligencia Artificial está en boca de todos: empresas, centros de investigación y ramas del sector público. Y ha llegado a nuestros tiempos para quedarse. Además, cada vez son más las empresas que se apoyan en la Inteligencia Artificial para la toma de decisiones, o para impulsar ciertas acciones. Por ello es importante entender qué significa, y cuáles son las diferencias conceptuales y las relaciones con otros de los términos más

populares dentro del campo de la inteligencia algorítmica del dato: Machine y Deep Learning (Martinez, 2018).

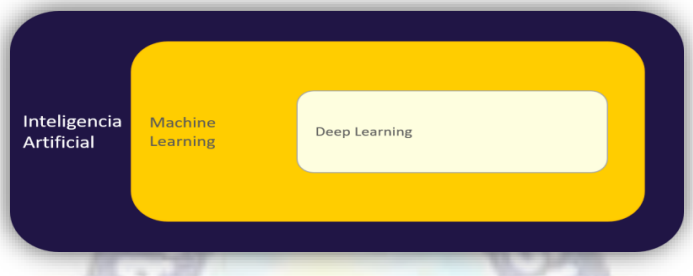


Figura 2.3. Campos que abarca la Inteligencia Artificial.
Fuente: (Martínez, 2018)

2.7.1 Machine Learning

El Aprendizaje Automático consiste en una disciplina de las ciencias informáticas, relacionada con el desarrollo de la Inteligencia Artificial, y que sirve, para crear sistemas que pueden aprender por sí solos. Es una tecnología que permite hacer automáticas una serie de operaciones con el fin de reducir la necesidad de que intervengan los seres humanos. Esto puede suponer una gran ventaja a la hora de controlar una gran cantidad de información de un modo mucho más efectivo.

Como estas acciones se realizan de manera autónoma por el sistema, se dice que el aprendizaje es automático, sin intervención humana (Apd, 2019).

Se denomina aprendizaje consiste en la capacidad del sistema para identificar una gran serie de patrones complejos determinados por una gran cantidad de parámetros. Es decir, la máquina no aprende por sí misma, sino un algoritmo de su programación, que se modifica con la constante entrada de datos en la interfaz, y que puede, de ese modo, predecir escenarios futuros o tomar acciones de manera automática según ciertas condiciones. Como estas acciones se realizan de manera autónoma por el sistema, se dice que el aprendizaje es automático, sin intervención humana (Apd, 2019).

En la informática clásica, el único modo de conseguir que un sistema informático hiciera algo era escribiendo un algoritmo que definiera el contexto y detalles de cada acción. En

cambio, los algoritmos que se usan en el desarrollo del Machine Learning realizan buena parte de estas acciones por su cuenta. Obtienen sus propios cálculos según los datos que se recopilan en el sistema, y cuantos más datos obtienen, mejores y más precisas serán las acciones resultantes (Apd, 2019).

Las computadoras se programan a sí mismas, hasta cierto punto, usando dichos algoritmos. Estos funcionan como ingenieros que pueden diseñar nuevas respuestas informáticas, como respuesta a la información que se les suministra a través de su interfaz u otros medios. Todo nuevo dato se convierte en un nuevo algoritmo, y a más datos, mayor complejidad y efectividad de cálculo puede proporcionar el sistema informático (Apd, 2019).

La clave de la capacidad de un sistema de Aprendizaje Automático se encuentra en la construcción y adaptación de los árboles de decisiones en base a los datos previamente conocidos por el sistema. Pero también influye la aplicación de fórmulas heurísticas en los nodos que forman el árbol, para el que se elabora un sistema de inferencias.

El sistema de Machine Learning necesita contar con un volumen de datos de relevancia para poder suministrar respuestas realmente válidas. El mínimo que se recomienda es de 6 entradas de datos reales para cada respuesta nueva diseñada, y esto debe repetirse para cada variable que conforman el sistema de trabajo del sistema.

2.7.2 Tipos de Machine Learning

Un sistema informático de Aprendizaje Automático se sirve de experiencias y evidencias en forma de datos, con los que comprender por sí mismo patrones o comportamientos. De este modo, puede elaborar predicciones de escenarios o iniciar operaciones que son la solución para una tarea específica (Apd, 2019).

A partir de un gran número de ejemplos de una situación, puede elaborarse un modelo que puede deducir y generalizar un comportamiento ya observado, y a partir de él realizar predicciones para casos totalmente nuevos. Como ejemplo, se puede considerar la

predicción del valor de unas acciones en el futuro según el comportamiento de las mismas en periodos previos (Apd, 2019).

Existen tres tipos principales de Aprendizaje Automático:

- **Aprendizaje supervisado:** Este tipo de aprendizaje se basa en lo que se conoce como información de entrenamiento. Se entrena al sistema proporcionándole cierta cantidad de datos definiéndolos al detalle con etiquetas. Por ejemplo, proporcionando a la computadora fotos de perros y gatos con etiquetas que los definen como tales.

  Una vez que se le ha proporcionado la suficiente cantidad de dichos datos, podrán introducirse nuevos datos sin necesidad de etiquetas, en base a patrones distintos que ha venido registrando durante el entrenamiento. Este sistema se conoce como clasificación.

  Otro método de desarrollo del Aprendizaje Automático consiste en predecir un valor continuo, utilizando parámetros distintos que, combinados en la introducción de nuevos datos, permite predecir un resultado determinado. Este método se conoce como regresión.

  Lo que distingue al Aprendizaje Supervisado es que se utilizan distintos ejemplos a partir de los que generalizar para nuevos casos (Apd, 2019).

- **Aprendizaje no supervisado:** En este tipo de aprendizaje no se usan valores verdaderos o etiquetas. Estos sistemas tienen como finalidad la comprensión y abstracción de patrones de información de manera directa. Este es un modelo de problema que se conoce como clustering. Es un método de entrenamiento más parecido al modo en que los humanos procesan la información (Apd, 2019).

- **Aprendizaje por refuerzo:** En la técnica de aprendizaje mediante refuerzo, los sistemas aprenden a partir de la experiencia. Como ejemplo se puede observar el

comportamiento de un coche autónomo. Cuando el vehículo toma una decisión errónea, es penalizado, dentro de un sistema de registro de valores. Mediante dicho sistema de premios y castigos, el vehículo desarrolla una forma más efectiva de realizar sus tareas.

Es una técnica basada en la prueba y error, y en el uso de funciones de premio que optimizan el comportamiento del sistema. Es una de las maneras más interesantes de aprendizaje para sistemas de Inteligencia Artificial, pues no requiere de la introducción de gran cantidad de información (Apd, 2019).

La tecnología del Aprendizaje Automático está sirviendo para recopilar y modelar el conocimiento, con el fin de proporcionar información más específica y elaborar mejores herramientas de trabajo para las personas. El uso de algoritmos marcará la competitividad y la profesionalidad durante los próximos años.

Por ello, no son pocas las empresas que utilizan el Machine Learning en sus servicios y productos, aprovechando los beneficios que puede reportar su aplicación, tanto para los procesos de sus organizaciones como para mejorar la experiencia de trabajo y entretenimiento de sus clientes.

2.8 Algoritmos de Machine Learning

En esta etapa de estudio sobre el Aprendizaje Automático se investigo diversos algoritmos que se utilizara para la resolución de problemas y que se repiten con mayor frecuencia. Realizaré un listado con una breve descripción de los principales algoritmos que se utilizara en el modelo.

2.8.1 Covarianza Robusta

Una de las dificultades a la hora de detectar anomalías en conjuntos de datos con dimensión **d > 3** es que no se puede apoyar en una inspección visual. Una forma de abordar este problema es la utilización de estimadores robustos para la localización y dispersión de

distribuciones multivariantes. Uno de los estimadores que hacen esto es el del elipsoide de volumen mínimo (MVE).

Este método lo que busca es el elipsoide de volumen mínimo que contiene **h** puntos, donde $n/2 \leq h < n$, con n siendo el número de observaciones. El algoritmo que reproduce esta metodología se llama MINVOL, que empieza con un subconjunto de **d+1** datos y calcula su media y su matriz de covarianza. El elipsoide correspondiente se va agrandando o empequeñeciendo hasta que contenga **h** puntos. Esto se repite varias veces y se escoge el elipsoide con un volumen menor (Rousseuw y Driessen, 1999).

Otro método similar es el del mínimo determinante de la covarianza (MCD). En este caso, lo que se busca son las **h** observaciones de la muestra cuya matriz de covarianzas tiene el menor determinante (Rousseuw y Driessen, 1999).

Según Rousseuw y Driessen, el método de mínimo determinante de la covarianza (MCD) presenta varias ventajas sobre el del elipsoide de volumen mínimo (MVE), por ejemplo:

- Su eficiencia estadística es mejor.
- Presenta una mayor precisión.
- Los estimadores obtenidos de MCD son más robustos y precisos que los obtenidos por MVE, lo que hace que sea mejor a la hora de detectar anomalías.

El principal problema de MCD es que no es sencillo de calcular. Como solución a esto, entonces se propone un nuevo algoritmo basado en MCD y que supera en velocidad a los basados en MVE (Rousseuw y Driessen, 1999).

2.8.2 Local Outlier Factor

En ocasiones, se puede encontrar con situaciones en las que definir un punto como anomalía no sea una decisión binaria. En estos casos tener una medida que me indique cómo de diferente es un punto, que de tal manera se pueda decidir a partir de qué momento un punto empieza a ser considerado anómalo (Breunig, 2000).

Para hacer esto, se debe mirar la definición de anomalía desde un punto de vista más cercano al concepto de agrupación de datos o cluster. En función de esto, una anomalía es un punto que no pertenece a ninguno de los clusters presentes en el conjunto de datos. El método de Local Outlier Factor propuesto por Breunig utiliza estas ideas para otorgar a cada instancia una puntuación en función de sus vecinos más cercanos (Breunig, 2000).

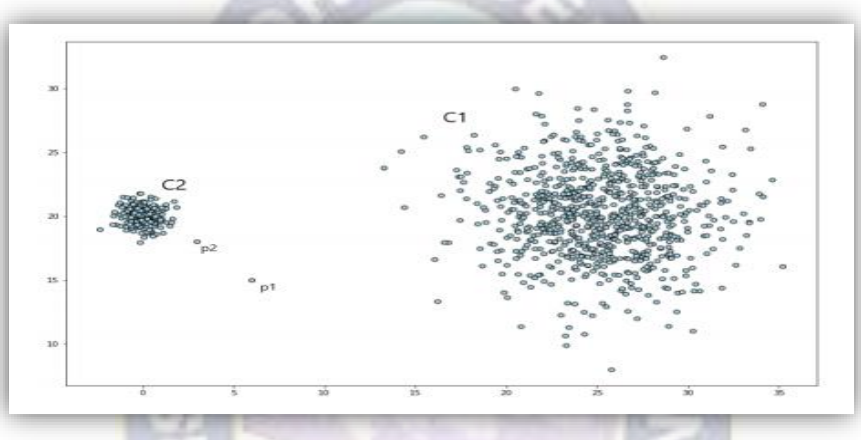


Figura 2.4. Ejemplo de outlier con respecto a varios clusters.
Fuente: (Bella, 2019)

Una definición de outlier desde este punto de vista puede ser la DB anomalía, que dice que un punto x_i en una muestra X es un anomalía si al menos un porcentaje pct de los objetos en X se encuentran a una distancia superior a dmin de x_i Breunig. El problema de esta definición es que solo hace referencia a anomalías globales. Por ejemplo, en la siguiente Figura 2.5, no hay forma de hacer que el punto P_2 sea considerado como una anomalía con respecto a C_2 sin hacer que el cluster C_1 también lo sea (Breunig, 2000).

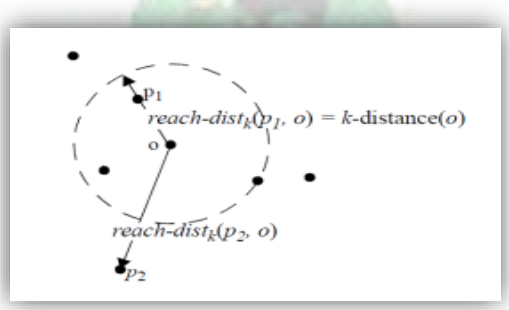


Figura 2.5. Ejemplo de la distancia de alcance entre dos puntos.
Fuente: (Bella, 2019).

2.8.3 Isolation Forest

Las anomalías u outliers son aquellos datos que presentan características diferentes a aquellos considerados como normales. En la mayoría de modelos que tratan de identificarlas intentan definir qué es una instancia normal para después catalogar como anomalías a todas aquellas que no se ajustan a esta definición Liu. Este tipo de modelos presentan dos inconvenientes (Liu, 2008):

- Puesto que están optimizados para detectar datos normales, a la hora de detectar anomalías se puede encontrar con modelos poco eficaces.
- La complejidad de algunos de estos modelos aumenta con la dimensión y el tamaño de los datos.

Para evitar esto, el algoritmo de Isolation Forests (iForests) se centra en aislar explícitamente las anomalías en lugar de definir los datos normales (Liu, 2008). Para hacer esto se utiliza una estructura basada en árboles, ya que se puede construir de tal manera que todos los puntos se encuentren aislados. Como las anomalías tienen características diferenciadas, tenderán a quedarse más cerca de la raíz del árbol, mientras que las instancias normales tenderán a quedarse en zonas más profundas. A estos árboles se les llama Isolation Trees o iTrees se aprecia en la siguiente Figura 2.6 (Liu, 2008).

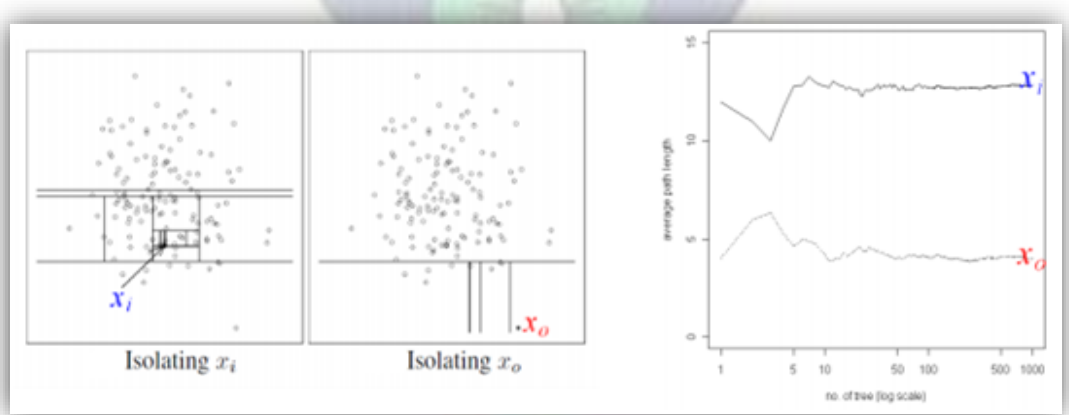


Figura 2.6. Aislamiento de dos puntos diferentes (izquierda) y profundidad media de dos puntos en función del número de árboles (derecha).
Fuente: (Bella, 2019)

El algoritmo de iForest utiliza un conjunto de iTrees para identificar como anomalías aquellos puntos que, de media, tienen poca profundidad. Algunas características de iForests son (Liu, 2008):

- Como las anomalías tienden a quedarse en la parte superior del árbol, no es necesario construirlo completamente.
- Muestras pequeñas de datos producen buenos resultados, ya que reducen los efectos de empantanamiento o swamping (identificar erróneamente datos normales como anomalías) y masking o enmascaramiento (un grupo grande de anomalías puede ser identificado como normal).
- Al no utilizar medidas basadas en la distancia o en la densidad, tiene un bajo coste computacional.
- Presenta una complejidad temporal lineal con poco uso de memoria.
- Permite el uso de grandes conjuntos de datos con altas dimensiones.

2.8.3.1 Construcción del algoritmo Isolation Forest

El algoritmo de bosque de aislamiento utiliza el hecho de que las observaciones anómalas son pocas y significativamente diferentes de las observaciones normales. El bosque se construye sobre la base de árboles de decisión, cada uno de los cuales tiene acceso a una submuestra de los datos de capacitación. Para crear una rama en el árbol, primero se selecciona una característica aleatoria. Luego, se elige un valor de división aleatorio (entre el valor mínimo y máximo) para esa característica. Si la observación dada tiene un valor más bajo de esta característica, entonces la seleccionada sigue a la rama izquierda, de lo contrario, la derecha. Este proceso continúa hasta que se aísla un solo punto o se alcanza la profundidad máxima especificada (Bella, 2019).

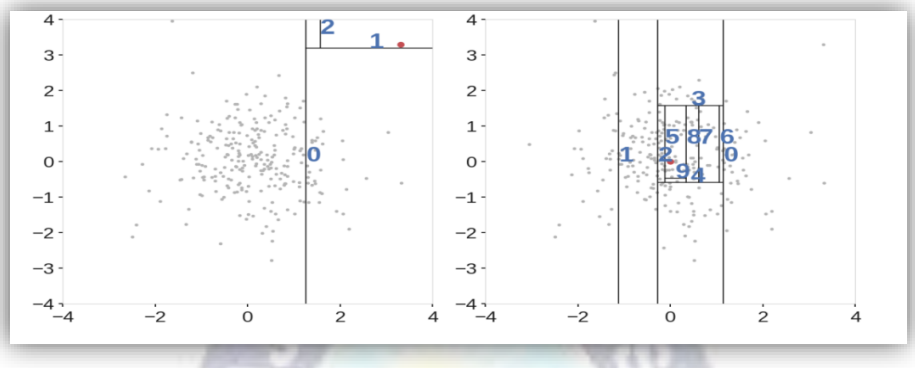


Figura 2.7. Se detecta un valor atípico (izquierda), un valor atípico dentro de los datos regulares (derecha).
Fuente: (Bella, 2019).

En principio, los valores atípicos son menos frecuentes que las observaciones regulares y son diferentes de ellos en términos de valores que se encuentran más lejos de las observaciones regulares en el espacio de características. Es por eso que al usar esta partición aleatoria se deben identificar más cerca de la raíz del árbol (longitud de ruta promedio más corta, es decir, el número de bordes que debe pasar una observación en el árbol que va desde la raíz al nodo terminal), con menos divisiones necesarias (Bella, 2019). El puntaje de anomalía se crea sobre la base de todos los árboles en el bosque y la profundidad que alcanza el punto en estos árboles.

2.8.4 One-class SVM

Durante los últimos años, han surgido numerosas técnicas de aprendizaje supervisado basadas en la utilización de kernels. Estos desarrollos se han dado sobre todo en los campos de reconocimiento de patrones y problemas de regresión Schölkopf. Esta popularidad ha hecho que se intente utilizar este tipo de algoritmos en el aprendizaje no supervisado, que se puede caracterizar por la búsqueda y estimación de funciones sobre los datos, con el fin de obtener cómo se distribuyen. Este tipo de aprendizaje puede ser visto como la búsqueda de la función de densidad de los datos, ya que conociendo esta información, podemos resolver, esencialmente, cualquier problema sobre los mismos. El método de One-class

SVM pretende hacer justamente eso, encontrar una función binaria que delimite la región en la que la mayoría de los datos de entrada se encuentran (Schölkopf, 2001).

Las máquinas de vectores de soporte (SVM) son modelos de aprendizaje supervisado que analizan datos y reconocen patrones, y que pueden usarse tanto para tareas de clasificación como de regresión.

Típicamente, el algoritmo SVM recibe un conjunto de ejemplos de entrenamiento etiquetados como pertenecientes a una de dos clases. Un modelo SVM se basa en dividir los puntos de muestra de entrenamiento en categorías separadas por un espacio lo más amplio posible, mientras penaliza las muestras de entrenamiento que caen en el lado equivocado del espacio. El modelo SVM luego hace predicciones asignando puntos a un lado de la brecha u otro (Schölkopf, 2001).

A veces, el sobre muestreo se usa para replicar las muestras existentes para que pueda crear un modelo de dos clases, pero es imposible predecir todos los nuevos patrones de fraude o fallas del sistema a partir de ejemplos limitados. Además, la recopilación de ejemplos incluso limitados puede ser costosa.

Por lo tanto, en SVM de una clase, el modelo de vector de soporte está entrenado en datos que tienen solo una clase, que es la clase normal. Infiere las propiedades de los casos normales y a partir de estas propiedades puede predecir qué ejemplos son diferentes a los ejemplos normales. Esto es útil para la detección de anomalías porque la escasez de ejemplos de capacitación es lo que define las anomalías: es decir, generalmente hay muy pocos ejemplos de intrusión en la red, fraude u otro comportamiento anómalo.

2.9 Matriz de Confusión

La matriz de confusión y sus métricas asociadas son parte fundamental del científico de datos, ya que, para saber qué modelo funciona mejor para un determinado problema, se necesita métricas o herramientas que ayuden a evaluarlo. Sin embargo, a pesar de su gran utilidad, es un concepto que de primeras resulta algo complejo (EcuRed, 2019).

Es una herramienta fundamental a la hora de evaluar el desempeño de un algoritmo de clasificación, ya que dará una mejor idea de cómo se está clasificando dicho algoritmo, a partir de un conteo de los aciertos y errores de cada una de las clases en la clasificación. Así se puede comprobar si el algoritmo está clasificando mal las clases y en qué medida.

El desempeño de un sistema es usualmente evaluado usando los datos en dicha matriz. La siguiente tabla muestra la matriz de confusión para un clasificador en dos clases:

Matriz de Confusión		**Predicción**	
		Positivos	**Negativos**
Observación	**Positivos**	**Verdaderos Positivos (VP)**	**Falsos Negativos (FN)**
	Negativos	**Falsos Positivos (FP)**	**Verdaderos Negativos (VN)**

Tabla 2.1. Matriz de confusión.

- **VP:** es la cantidad de positivos que fueron clasificados correctamente como positivos por el modelo.
- **VN:** es la cantidad de negativos que fueron clasificados correctamente como negativos por el modelo.
- **FN:** es la cantidad de positivos que fueron clasificados incorrectamente como negativos. Error tipo 2 (Falsos Negativos).
- **FP:** es la cantidad de negativos que fueron clasificados incorrectamente como positivos. Error tipo 1 (Falsos Positivos).

En el campo de la inteligencia artificial una matriz de confusión es una herramienta que permite la visualización del desempeño de un algoritmo que se emplea en Machine Learning. En base a los valores de ésta nueva matriz de confusión, más completa, se define una serie de métricas que serán muy útiles (EcuRed, 2019).

- **Exactitud (Accuary)** se refiere a lo cerca que esta el resultado de una medición del valor verdadero. En términos estadísticos, la exactitud está relacionada con el sesgo de una estimación. Se representa por la proporción entre los positivos reales predichos por el algoritmo y todos los casos positivos (EcuRed, 2019).

$$\boldsymbol{Exactitud} = \frac{(VP + VN)}{(VP + FP + FN + VN)} \qquad (2.8.1)$$

- **Precisión (Precision)** se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión. Se representa por la proporción entre el número de predicciones correctas tanto positivas como negativas y el total de predicciones (EcuRed, 2019).

$$\boldsymbol{Precisión} = \frac{VP}{(VP + FP)} \qquad (2.8.2)$$

- **Sensibilidad (Recall)** también se conoce como tasa de verdaderos positivos. Es la proporción de casos positivos que fueron correctamente identificadas por el algoritmo.

$$\boldsymbol{Sensibilidad} = \frac{VP}{(VP + FN)} \qquad (2.8.3)$$

- **Especificidad (Especificity)** también conocida como la tasa de verdaderos negativos. Se trata de los casos negativos que el algoritmo ha clasificado correctamente.

$$\boldsymbol{Especificidad} = \frac{VN}{(VN + FP)} \qquad (2.8.4)$$

2.10 Overfitting y Underfitting

Cuando se utiliza el aprendizaje automático, hay muchas formas de equivocarse. Algunas de las cuestiones más comunes en el aprendizaje automático son el Overfitting y Underfitting. Para comprender estos conceptos, imaginemos un modelo de aprendizaje automático que está tratando de aprender a clasificar números y tiene acceso a un conjunto de datos de capacitación y un conjunto de datos de prueba (Nautiyal, 2019)

- **Overfitting:** Se dice que un modelo estadístico está sobre ajustado cuando se entrena con una gran cantidad de datos. Cuando un modelo se entrena con tanta información, comienza a aprender del ruido y las entradas de datos inexactas en nuestro conjunto de datos. Entonces, el modelo no clasifica los datos correctamente, debido a demasiados detalles y ruido. Las causas del sobreajuste son los métodos no paramétricos y no lineales porque estos tipos de algoritmos de aprendizaje automático tienen más libertad para construir el modelo basado en el conjunto de datos y, por lo tanto, realmente se pueden construir modelos poco realistas. Una solución para evitar el sobreajuste es usar un algoritmo lineal si se tiene datos lineales o usar parámetros como la profundidad máxima si se usa árboles de decisión (Nautiyal, 2019)

- **Underfitting:** Un modelo estadístico o un algoritmo de aprendizaje automático tiene un ajuste insuficiente cuando no puede capturar la tendencia subyacente de los datos. La adaptación interior destruye la precisión del modelo de aprendizaje automático. Su aparición simplemente significa que el modelo o el algoritmo no se ajusta lo suficientemente bien a los datos. Suele ocurrir cuando se tiene menos datos para construir un modelo preciso y también cuando se trata de construir un modelo lineal con datos no lineales. En tales casos, las reglas del modelo de aprendizaje automático son demasiado fáciles y flexibles para aplicarse con datos tan mínimos y, por lo tanto, el modelo probablemente hará muchas predicciones erróneas. Se puede evitar la falta de ajuste utilizando más datos y también reduciendo las características mediante la selección de características (Nautiyal, 2019)

2.11 Curva ROC y Área bajo la Curva (AUC)

En Machine Learning, la medición del rendimiento es una tarea esencial. Entonces, cuando se trata de un problema de clasificación, se puede contar con una curva AUC-ROC. Esta es una de las métricas de evaluación más importante para verificar el rendimiento de cualquier modelo de clasificación.

Es una de las métricas de evaluación para verificar el rendimiento de cualquier modelo de clasificación. También está escrito como AUROC (Área bajo las características operativas del receptor) (González, 2019).

2.11.1 Curva ROC

La curva ROC muestra qué tan bueno puede distinguir el modelo entre dos cosas, por ejemplo, si un paciente tiene una enfermedad o no. Mejores modelos pueden distinguir con precisión entre los dos, mientras que un modelo pobre tendrá dificultades para distinguir entre los dos. En la siguiente Figura 2.8 se puede apreciar la curva de color verde que son los pacientes sin enfermedad, la curva de color rojo que son los pacientes que están enfermos y por último el umbral que divide estas dos curvas (González, 2019).

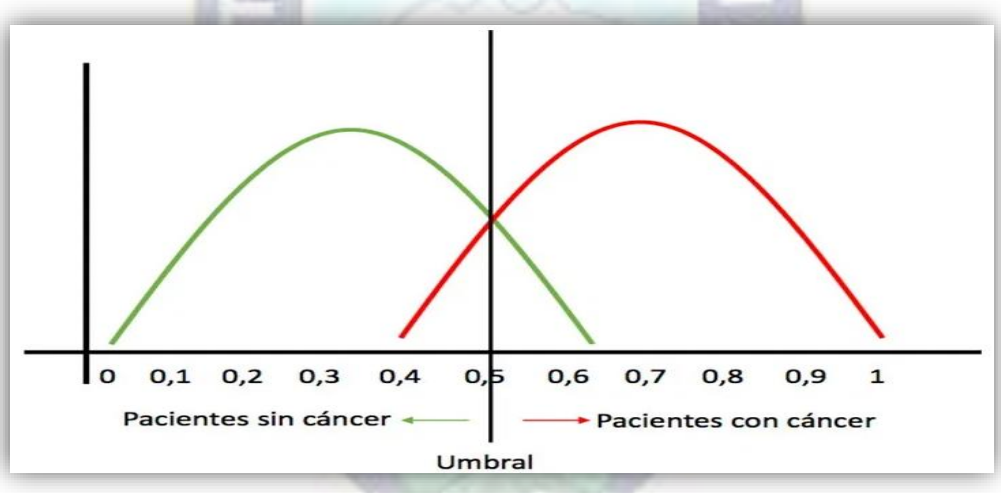


Figura 2.8. Curva de color Verde pacientes con enfermedad y curva roja pacientes que tienen enfermedad.
Fuente: (González, 2019)

Tomando los conceptos aprendidos en la matriz de confusión, todos los valores positivos por encima del umbral serán **verdaderos positivos** y los valores negativos por encima del umbral será **falsos positivos**, ya que se predicen incorrectamente como positivos. Todos los valores negativos por debajo del umbral serán **verdaderos negativos** y los valores positivos por debajo del umbral serán **falsos negativos**, ya que se pronostican incorrectamente como negativos. Como se muestra en la siguiente Figura 2.9.

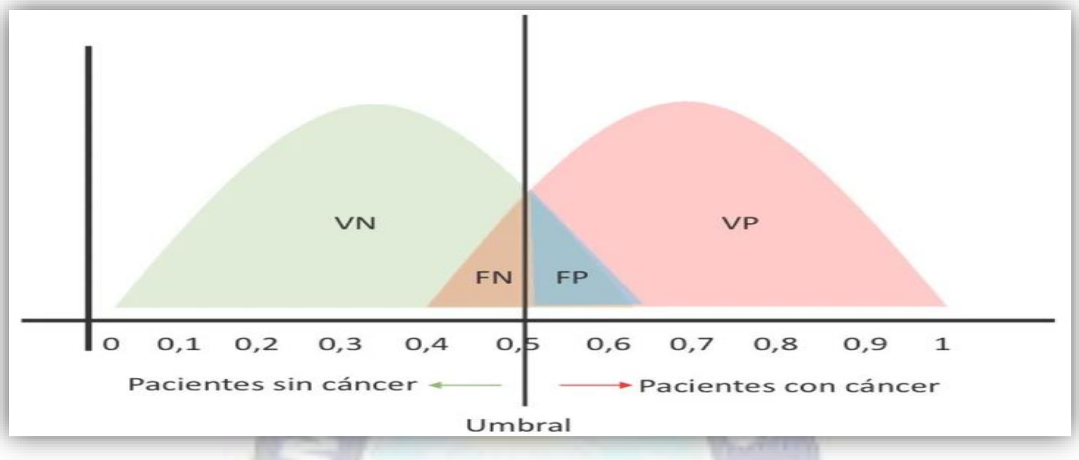


Figura 2.9. Curvas en Base a la matriz de Confusión.
Fuente: ()

2.11.2 Área bajo la curva (AUC)

El AUC es el área bajo la curva ROC. Este puntaje da una buena idea de qué tan bien funciona el modelo. Cuando dos distribuciones se superponen, introducimos errores. Dependiendo del umbral, se puede minimizar o maximizar. Cuando AUC es 0.7, significa que hay 70% de probabilidad de que el modelo pueda distinguir entre clase positiva y clase negativa, como se muestra en la siguiente Figura 2.10 (González, 2019).

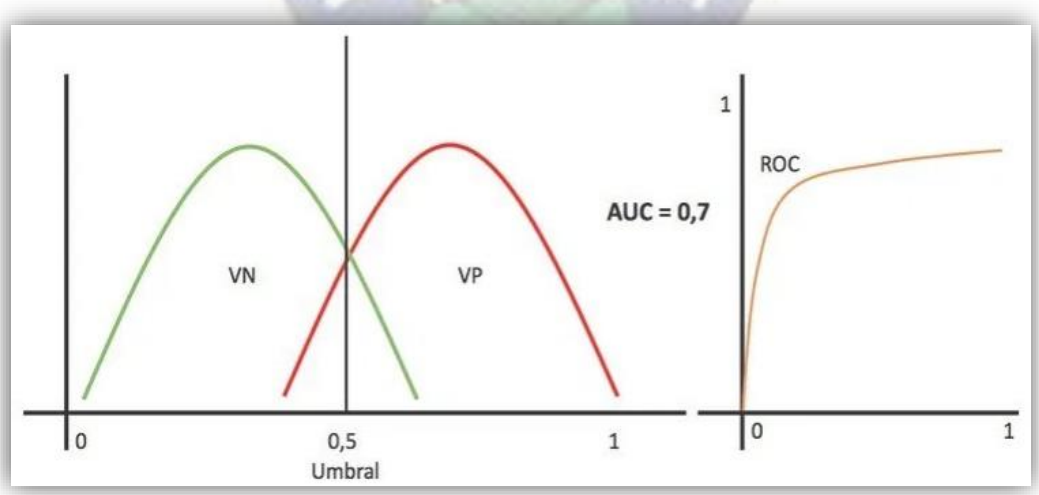


Figura 2.10. Curva de ROC al 70% de probabilidad.
Fuente: (González, 2019)

Cuando AUC es aproximadamente 0, el modelo en realidad está correspondiendo las clases. Significa que el modelo predice la clase negativa como una clase positiva y viceversa. Por tanto el modelo no puede predecir, se puede apreciar en la siguiente Figura 2.11.

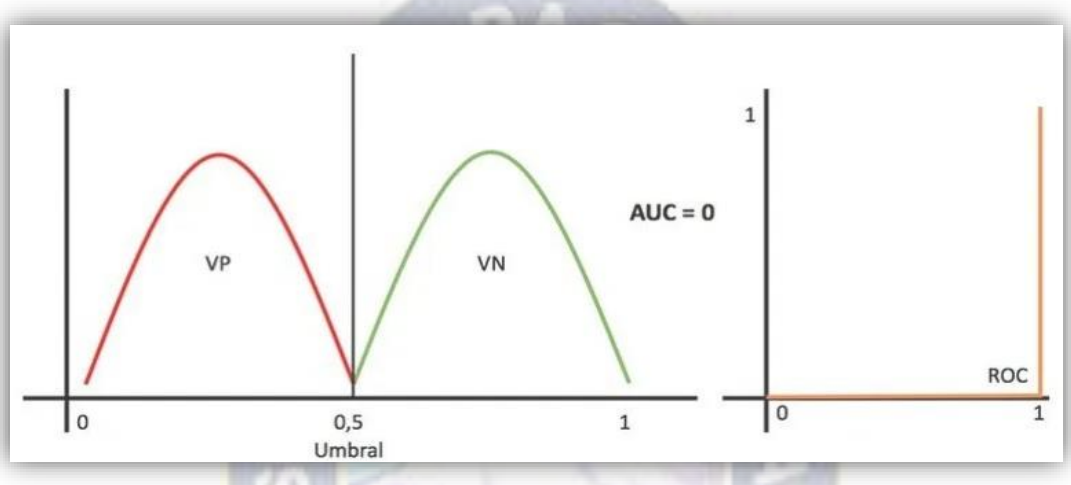


Figura 2.11. Curva de ROC con probabilidad AUC=0.
Fuente: (González, 2019)

Este método es conveniente por las siguientes razones:

- Es invariable con respecto a la escala, mide qué tan bien se clasifican las predicciones, en lugar de sus valores absolutos.
- Es invariable con respecto al umbral de clasificación, mide la calidad de las predicciones del modelo, sin tener en cuenta qué umbral de clasificación se elige.

Capitulo 3

Marco Aplicativo

3.1 Introducción

El ciclo vital del modelo contiene seis fases, la secuencia de las fases no es estricta. El modelo de CRISP-DM es flexible y se pueden personalizar fácilmente. Por otra parte el hecho de que más del 75% del esfuerzo se produzca en las primeras fases (en el caso de pre tratamiento de los datos) es importante.

En la fase inicial de la comprensión del negocio se analiza cómo detectar el malware en una red con solo observar el tráfico de red, también se comprende con qué tipo de malware se trabaja. La fase de entendimiento de los datos se comienza a analizar los datos iníciales e identificar los problemas de calidad de los datos; es decir si los datos son óptimos, pero para tener estos datos iníciales se trabajo en la obtención de los mismos, también se muestra como se capturo el flujo de trafico de red para luego almacenarla en una hoja de cálculo.

En la fase de análisis se analiza los datos iníciales para luego almacenar en un DataSet, una vez almacenado se realizara un tratamiento a nuestro conjunto de datos con la finalidad de convertir en un Dataset limpio con el cual se pueda trabajar. Hasta esta fase se abarca la mayor parte del tiempo y esfuerzo para así tener un mejor resultado al momento de aplicar el modelado.

En el modelado se selecciona varios algoritmos para luego aplicar técnicas de modelado que sean pertinentes al problema, si las técnicas aplicadas no reaccionan como uno espera retornara a la fase de preparación de los datos con el fin de realizar un mejor tratamiento de los datos adaptados a las técnicas de modelado. En la etapa de evaluación se analiza la calidad del entrenamiento del modelo, es importante esta etapa para evaluar a fondo y revisar todo lo ejecutado con el objetivo de obtener una decisión sobre los resultados del

proceso de análisis de datos. Finalmente la fase de despliegue es la creación del modelo; no es el final del proyecto, incluso si el objetivo del modelo es de aumentar el conocimiento de los datos, el conocimiento obtenido tendrá que organizarse y presentarse para que el cliente pueda usarlo.

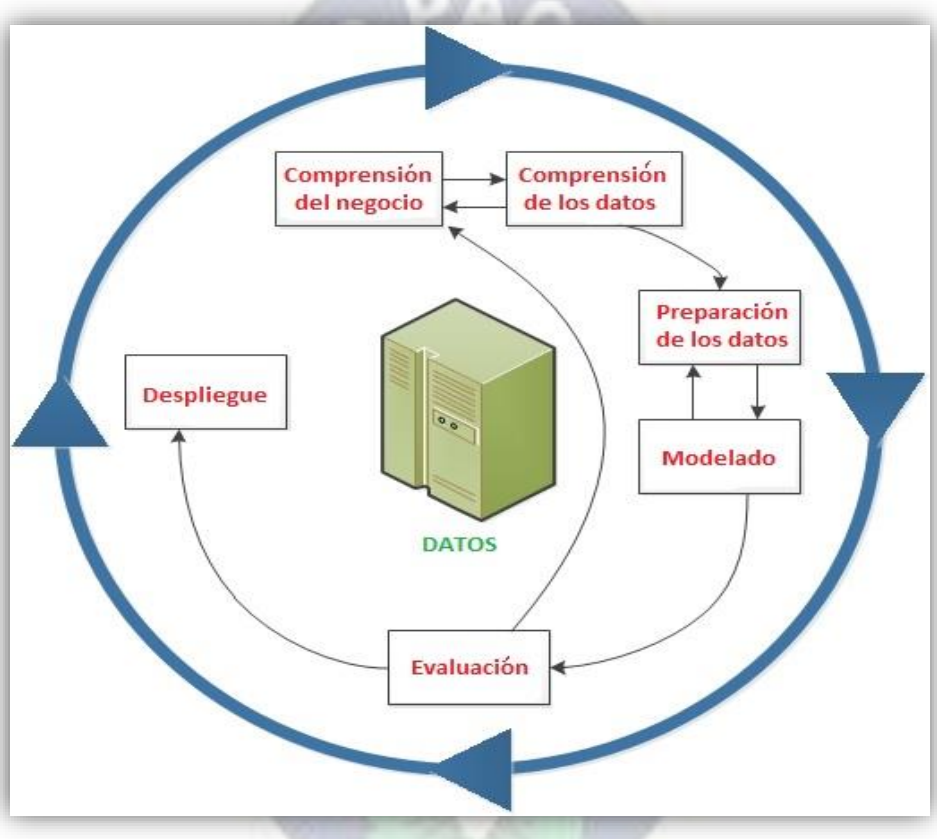


Figura 3.1. Procesos que seguiremos para el diseño del modelo.
Fuente: (IBM, 2012)

3.2 Comprensión del negocio

La seguridad Informática se enfoca en la protección de la infraestructura computacional y todo lo relacionado con esta; especialmente la información contenida en una computadora o circulante a través de las redes de computadoras. El objetivo es detectar el malware en una red, se comprende que en el flujo de tráfico de red se puede identificar vulnerabilidades. Para realizar este análisis se necesita el flujo del tráfico de una red, con el objetivo de ver los protocolos de red más usados y lograr detectar anomalías.

Encontrar patrones en los datos del flujo del tráfico de red que no se ajusten al comportamiento esperado es decir anomalías, sigue siendo un reto interesante en la seguridad informática. Las anomalías en el tráfico aparecerán por varias razones como actividades maliciosas o caídas, en este sentido la captura del flujo de red puede convertirse en un buen aliado para ir más allá de las firmas y ser capaz así de encontrar patrones previamente desconocidos.

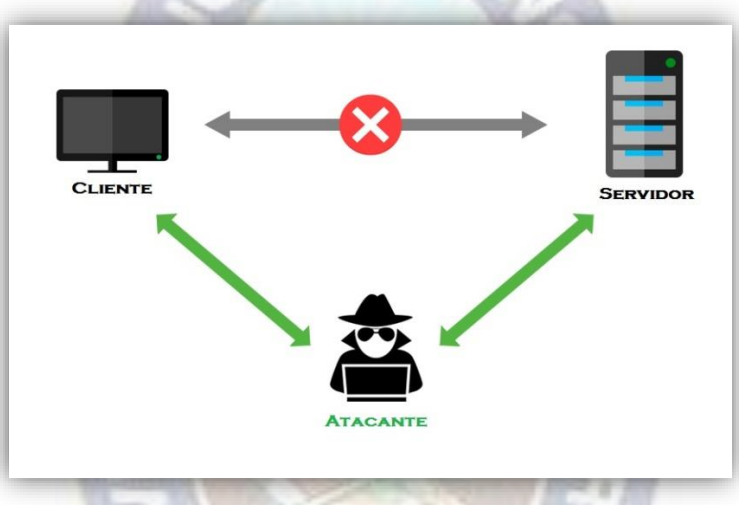


Figura 3.2. Los datos se direccionan primero al atacante para luego reportar al servidor, muestra de cómo se infecta al cliente y no al servidor.

La gran cantidad de tráfico que circula por las redes puede llegar a ser abrumadora y ahí es donde el análisis podrá mostrar estadísticas sobre el tráfico de red que permita conocer que está pasando.

Ahora los ciberdelincuentes consiguen amenazar tanto a un usuario cualquiera de Internet que navega tranquilamente desde casa como autónoma o empresas de todos los tamaños. Pequeñas, medianas o grandes, públicas o privadas, todas las organizaciones se encuentran el punto de mira de un tipo de malware que es capaz de acceder a equipos informático, cifrar información sensible, bloquear el sistema y reclamar pagos a cambio de su restablecimiento, casi sin darle a tiempo a las víctimas a pestañear. De ahí su nombre, ya que **ransom** significa tanto **secuestro** como **rescate**, que es justo el proceso que se acciona en cuanto el dispositivo es infectado y, en consecuencia, bloqueado. Se trata de un chantaje

puro y duro que pone en jaque el propio funcionamiento del negocio y nos mostrara una ventana como en la siguiente Figura 3.3.

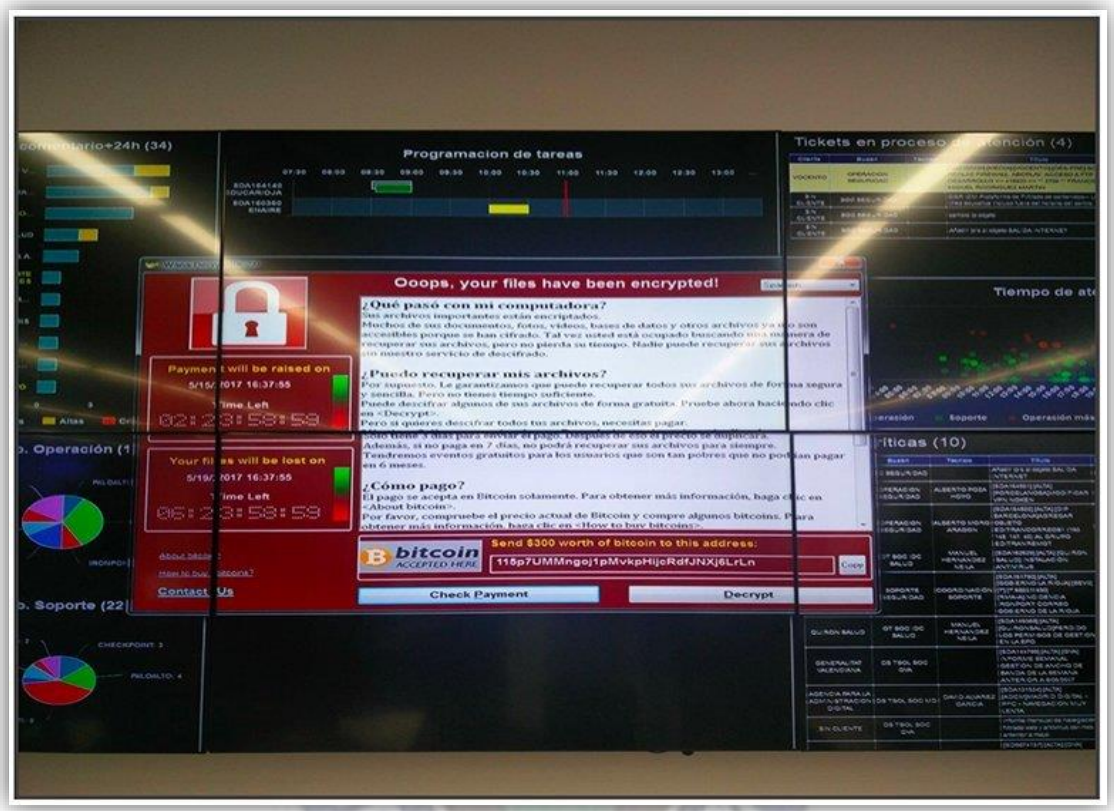


Figura 3.3. Captura de pantalla que contiene el mensaje que indica que se han cifrado los archivos importantes, además ejecuta una nota con un temporizador que indica el tiempo restante para realizar el pago y obtener la clave de descifrado.
Fuente: (Losada, 2017)

Las amenazas ransomware aprovechan los correos electrónicos acompañados de ingeniería social como vector de ataque, pero no suelen propagarse tan rápidamente. Por tal motivo se trabaja con el malware **ransomware**, a ver qué sucede en el flujo del tráfico de red.

Ahora se mostrara cómo se comporta el ransomware en la red. Primero se captura el flujo de tráfico de red, para identificar flujo de tráfico con comportamiento normal y flujo de tráfico no anómalo; es decir cómo se comporta el ransomware. En la siguiente Figura 3.4 se aprecia el flujo de tráfico de una red y los campos de información.

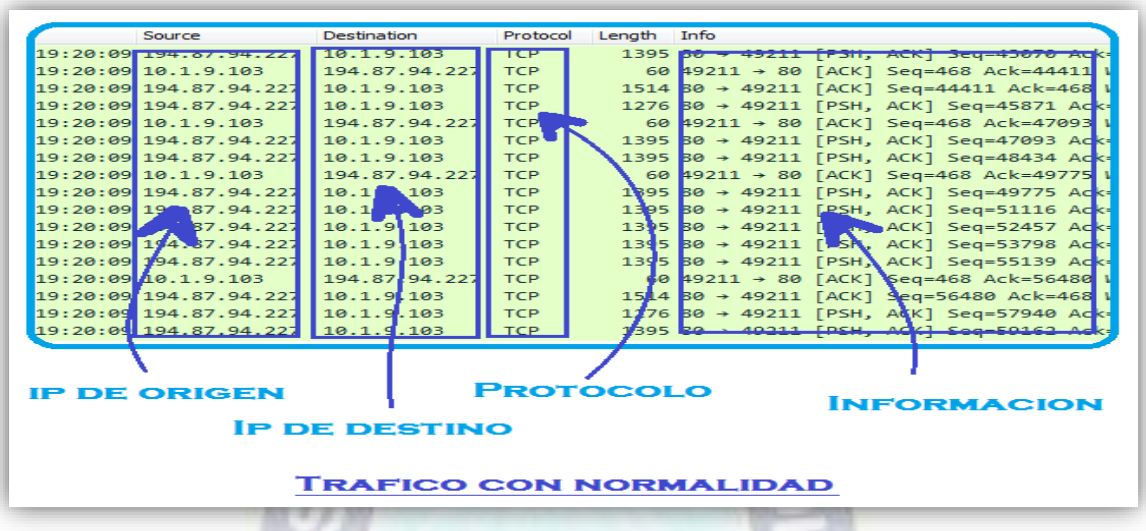


Figura 3.4. El flujo del tráfico de red, utilizando la herramienta de Wireshark.

Como se pudo apreciar en la anterior Figura 3.4 el flujo de tráfico de red se encuentra con normalidad, por el momento se continúa haciendo el respectivo análisis. Pero cuando en el análisis del flujo de red nos percatamos que existe una anomalía la cual puede derivar a más anomalías, como se muestra en la siguiente Figura 3.5.

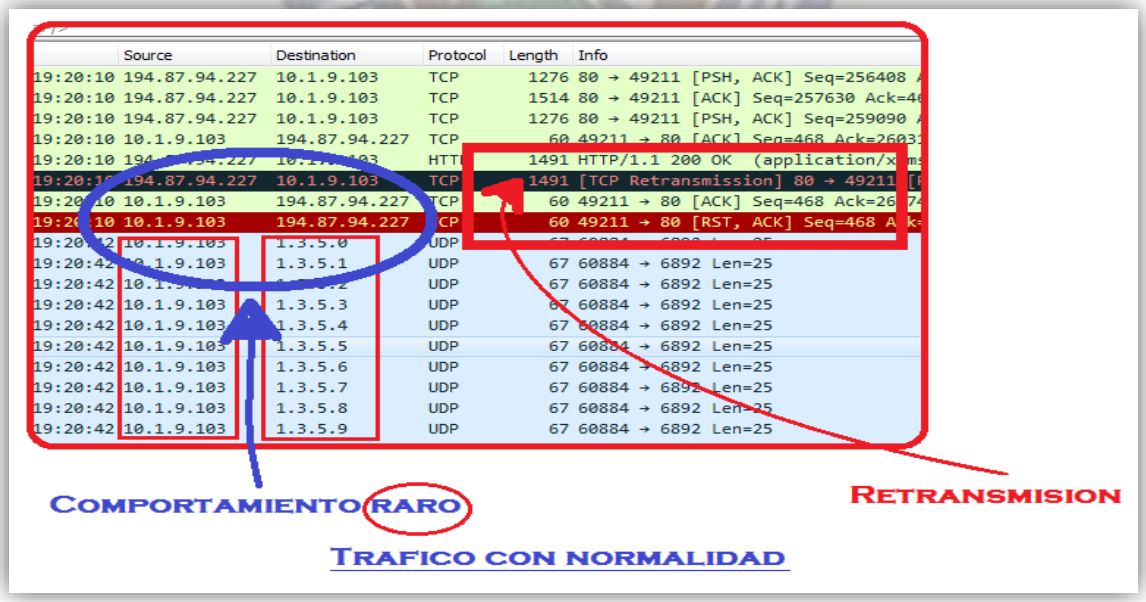


Figura 3.5. Retransmisión de paquetes por congestión de paquetes, y se muestra una anomalía en IP de destino (Trafico con Normalidad).

En la anterior Figura 3.5 hay muchas cosas por analizar, primero hubo congestión de red, este fenómeno producido cuando a la red, o parte de ella, se le ofrece más trafico del puede cursar. Existe varias causas de congestión una puede ser la memoria es insuficiente de los conmutadores, es cuando los paquetes se reciben demasiado deprisa para ser procesados. Además puede ser que en la memoria de salida existan demasiados paquetes esperando ser atendidos, entonces se llena la memoria de salida. Otra causa puede ser la Insuficiente CPU (Unidad Central de Proceso) en los nodos, puede que el nodo sea incapaz de procesar toda la información que le llega, con lo que hará que se saturen las colas.

Por lo tanto cuando existe una congestión de red como se muestra en la siguiente Figura 3.6, el sistema intenta recuperar los paquetes haciendo una retransmisión con el paquete que se perdió.

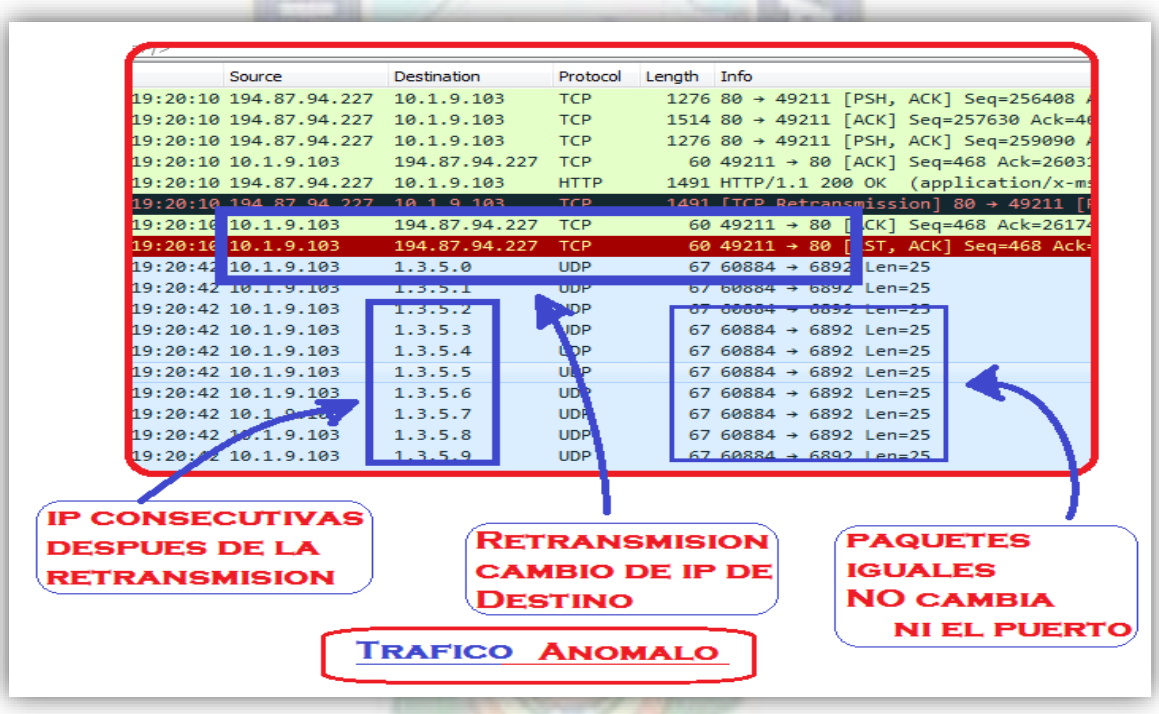


Figura 3.6. Cambio de IP de destino al momento de la retransmisión de paquetes direccionando a una IP diferente (tráfico de red Anómalo).

Pero en el flujo de red que se muestra en la anterior Figura 3.6, se ve que recupera el paquete de inicio de sección, en el momento de la recuperación del paquete se puede

apreciar que la IP de destino cambia, lo cual en algunos casos es normal pero en este caso las IPs que siguen después de esa, son consecutivas al primer IP de destino de la recuperación, otro detalle que se puede observar es la información de paquetes, no cambian el puerto y se repite el mismo paquete. Esta observación no siempre estará presente en algunas anomalías pero la anomalía de la IP de destino esa sí estará presente.

3.3 Comprensión de los Datos

El DataSet es un componente fundamental para el acceso a datos, son objetos en memoria que pueden contener tablas, vistas y relaciones. Totalmente es recomendable crear tu propio DataSet sin embargo otra opción sería obtener de fuentes de datos gratuitas, la ventaja está en que no tendrá que invertir de inicio en crear uno. Pero si se quiere de verdad, sacar provecho del modelo basado en Data Science no tendremos más remedio que crear nuestro propio repositorio de datos

Lo complejo será crear un DataSet de un flujo del tráfico de red, donde todo el tráfico de flujo de red que comparte ciertas características comunes y se mueve desde un host de red a otro. Por ejemplo, si se considera todo el tráfico que pueden compartir una estación y un servidor, como flujo de aquel tráfico que hace parte de una misma conversación o tiene el mismo objetivo.

No se almacena el flujo como tal, solo la metadata. La idea es utilizar los dispositivos involucrados en el pase del tráfico de red, de tal forma sin almacenar los paquetes que conforman el flujo de tráfico, generar información sobre el flujo de tráfico o su metada. Luego está metadata deberá ser almacenada y reprocesada para finalmente ser mostrada con la idea de permitir el análisis. El análisis de flujo de tráfico de red se ha consolidado sobre la base de un grupo de protocolos que permiten implementar los procesos de generación, transporte, almacenaje y pre procesamiento de la metadata.

En la siguiente Figura 3.7 se muestra dos formas de obtener una terminal, para que pueda capturar todo el flujo de tráfico de una red.

Figura 3.7. Dos formas de obtener una terminal, insertándolo directamente al router (Exportador) y otra forma insertándolo al servidor (Recaudador).

Es importante aclarar que estos procedimientos no especifican como debe hacerse el análisis; eso lo dejan a las herramientas que utilizan la metadata para alcanzar sus objetivos.

Para crear el DataSet se necesita de la captura del tráfico del flujo de red; el cual se obtiene de un Switchport, generalmente los puertos de un switchport tienen una combinación de troncalización de Vlan y enlaces de acceso. La principal utilidad de los puertos de Access es conectar equipos finales, los puertos de acceso solo transportan tráfico de una sola Vlan y aunque los puertos de acceso también se pueden utilizar para conectar switches no es recomendable ya que una implementación de este tipo no es escalable.

En la siguiente Figura 3.8 se puede apreciar cómo se coloco los enlaces desde un host al switchport, estos enlaces que están conectados directamente a los host estarán en modo Access, y los equipos como ser el router estarán en modo Trunk pero eso se verá más adelante.

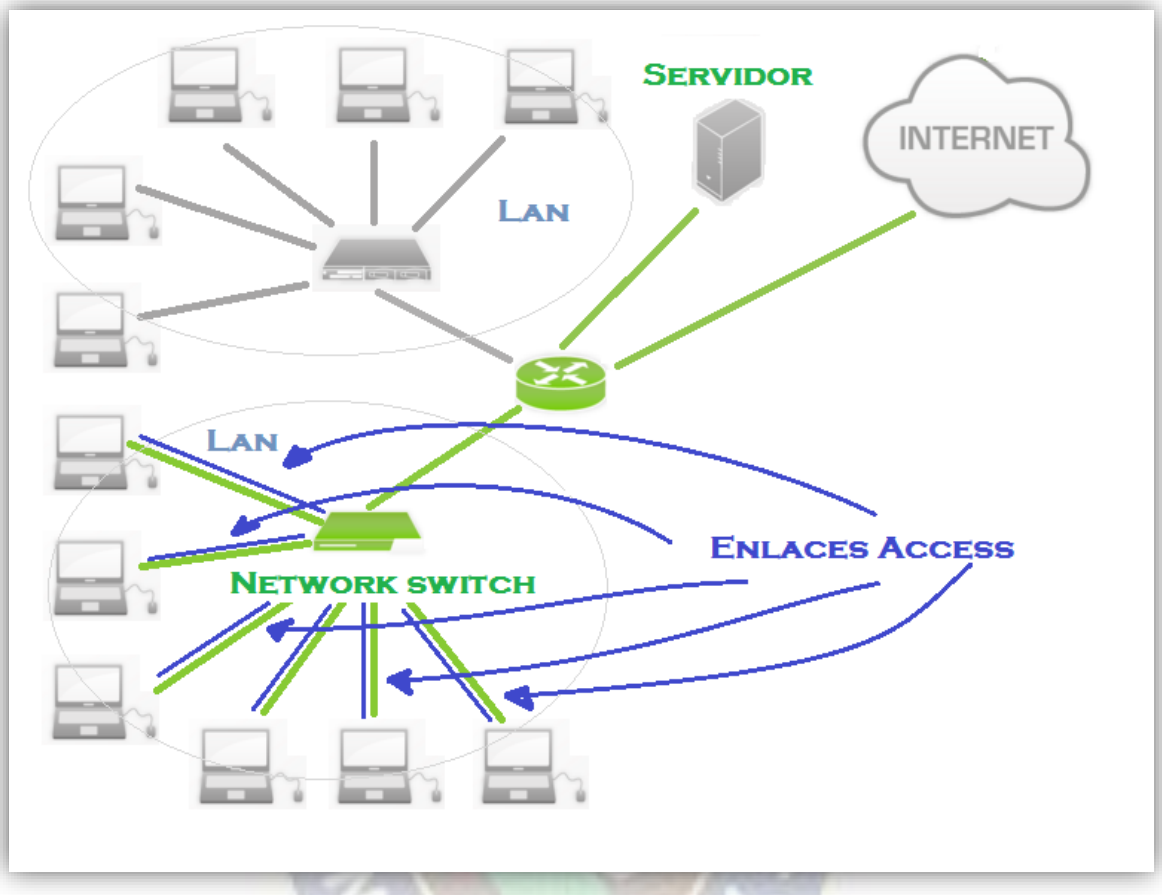


Figura 3.8. Clientes (host) conectados al switch en modo de enlace Access.

Tomando como referencia la topología anterior de la Figura 3.8, cuando una host manda un paquete, dicho paquete no lleva ningún identificador de la vlan a la que este asociado el puerto. Un detalle importante que en una interfaz solo puede estar asociado una Vlan.

Los puertos de modo enlace Trunk son para realizar la conexión entre switches, un puerto trunk puede transportar tráfico de múltiples vlans en los switches y solo un enlace para transportar todo el tráfico. Estos enlaces estarán conectados directamente con el router, en la siguiente Figura 3.9 se puede apreciar cuales serán los enlaces en modo Trunk. Los enlaces modo Trunk no se pueden colocar entre routers, ya que este tipo de modo solo se coloca en switches y no routers.

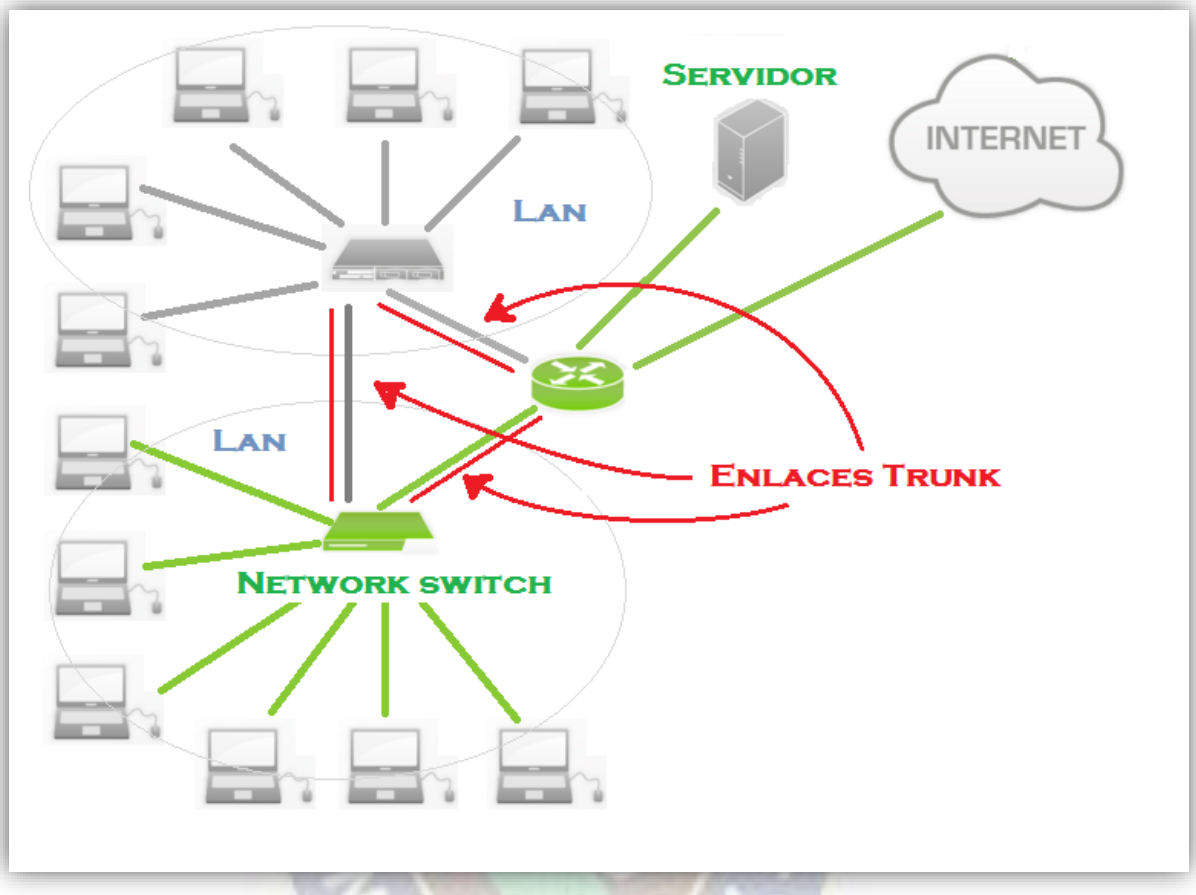


Figura 3.9: Routers y Switches con modo de enlace Trunk. Router (Recaudador) conectado con el servidor e internet.

Una vez lista la red, incluyendo los puertos destinados del switchport tanto como los puertos de Access y puertos Trunks de cada uno de los dispositivos. Se designa un puerto del Switchport para un Host, el propósito de este host es poder capturar todo el flujo de trafico de red, el puerto del switchport donde se puso el host será de modo Trunk como resultado tenemos opción de ver todas las múltiples Vlans que están conectadas a este switchport.

Si el puerto se pusiera en modo Access solo podrá capturar el flujo de red a la VLAN que pertenece dicho puerto, lo cual no es conveniente ya que se quiere capturar el flujo de todos los host de la red, por tal motivo el puerto en está en modo Trunk.

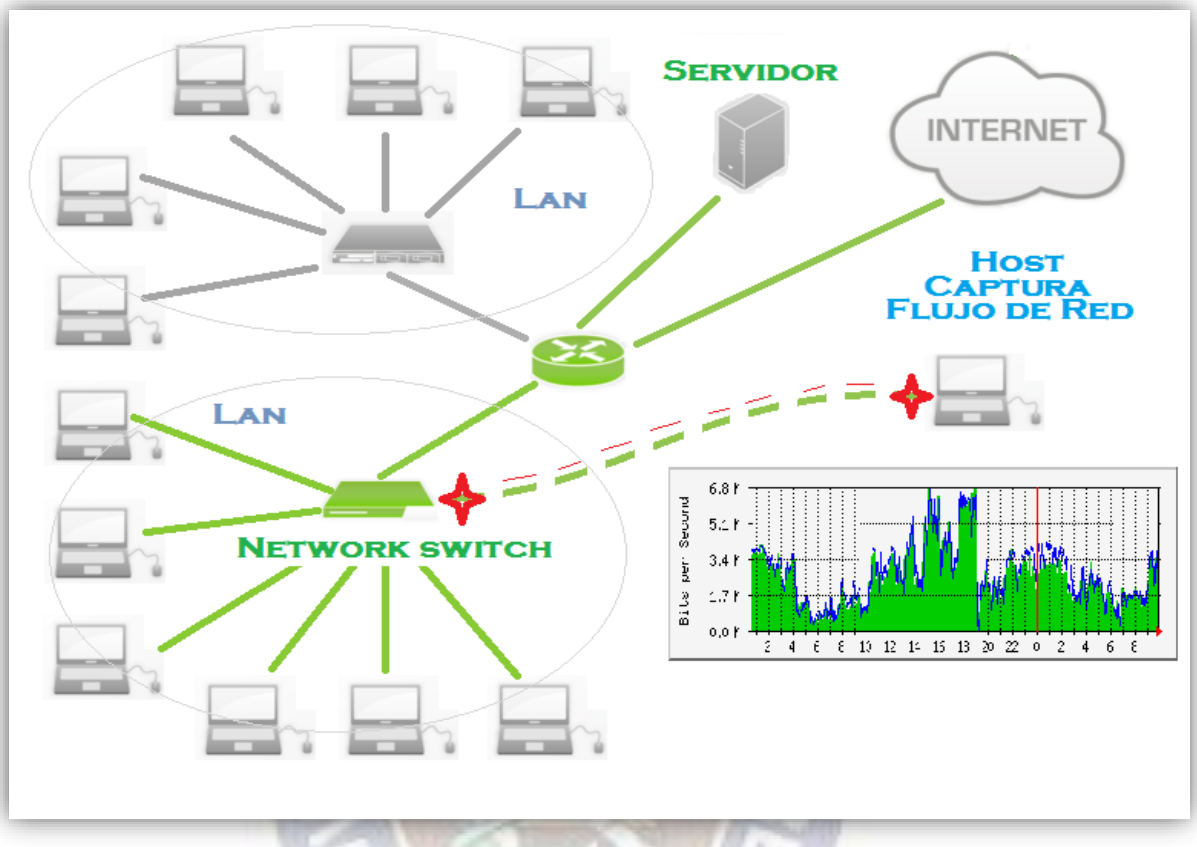


Figura 3.10. Terminal conectada a un switchport en modo de enlace Trunk. Host captura el flujo de tráfico de la red Lan.

Una vez ubicado el host, se encargará de recabar la metadata de los flujos de trafico IP entrante y saliente de algún dispositivo de red. Esto se logra gracias a los exportadores que nos ofrece NetFlow de hecho es un protocolo del software que están contenido en dispositivos como switches y enrutadores de los equipos Cisco. Este protocolo ayuda a almacenar la información sobre los flujos que van capturando a medida que el trafico entra y sale a través del dispositivo switch o enrutador, luego entran los colectores encargados de recibir la metadata de los exportadores, almacenarla y pre procesarla. En la siguiente Figura 3.11 se puede apreciar que al final el analizador será el elemento encargado de permitir el análisis sobre la información contenida en los colectores. Todo la información capturada se almacena en un fichero .pcacp es el formato para almacenar capturas de trafico de red y soportado por herramientas populares como Wireshark.

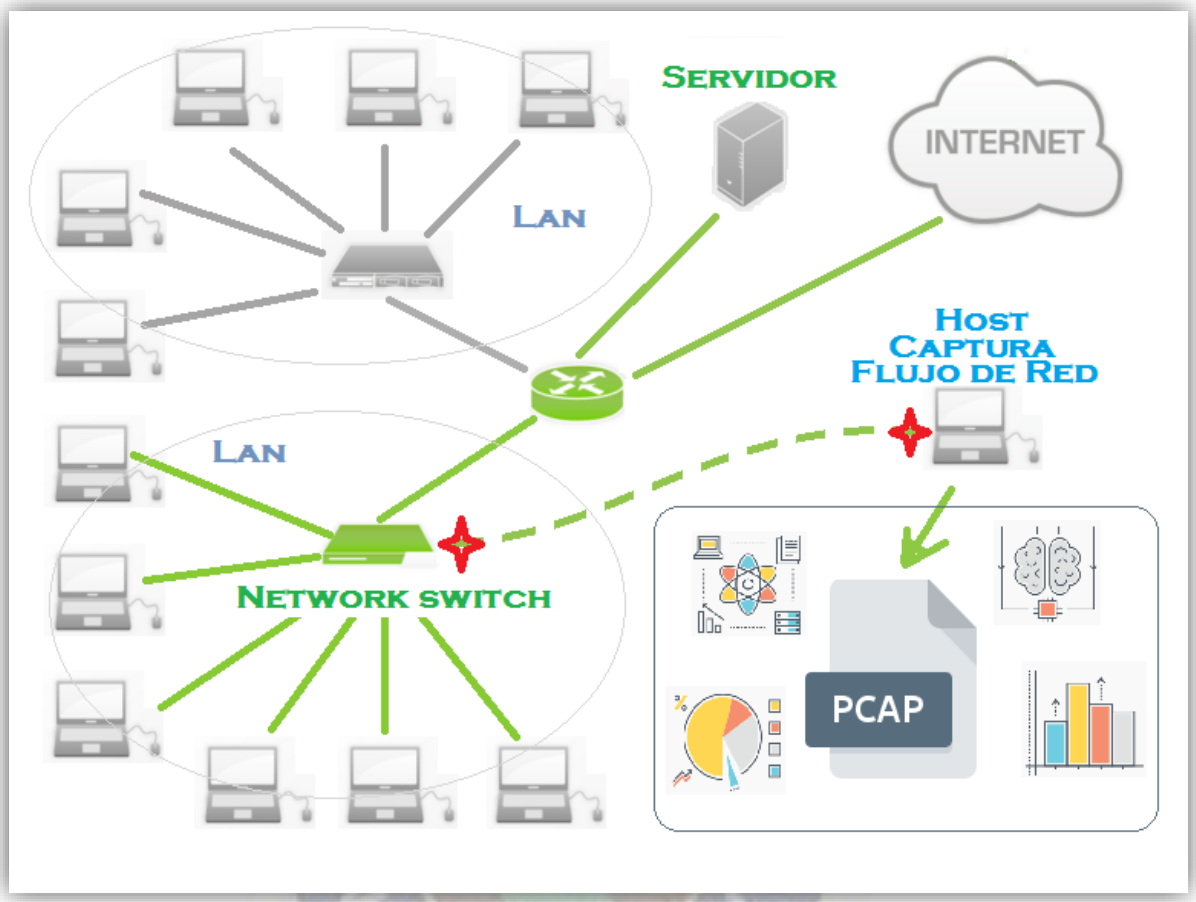


Figura 3.11. Host exporta el flujo de tráfico de red en un archivo de formato (.pacp). Este fichero contiene el tráfico de la red LAN.

El fichero (.pcap) será el Dataset ya que contiene todo los datos del flujo de trafico de red, pero en este formato que se encuentra el fichero no es posible manipular los datos, por tal motivo el fichero se tendrá que llevar a un (.csv) que es un archivo de texto que almacena los datos de columnas, separadas por coma y filas se distinguen por saltos de línea; es decir una hoja de cálculo.

Normalmente para imp**o**rtar o exportar en cualquier otro formato, se tiene que utilizar una herramienta, existe una gran variedad de herramientas. Se utilizara la Herramienta Wireshark que ayudara a visualizar todo el flujo del tráfico de la red. En la siguiente Figura 3.12 se puede apreciar como es el entorno del Wireshark.

Figura 3.12. Información de la Herramienta de Wireshark.

Esta herramienta permite analizar la información capturada, a través de los detalles de cada paquete, también nos ofrece la opción de grabar nuestro archivos en formato (.csv). Finalmente se guarda el fichero (.pcap) en formato de hojas de cálculo (.csv) el resultado se puede apreciar en la siguiente Figura 3.13.

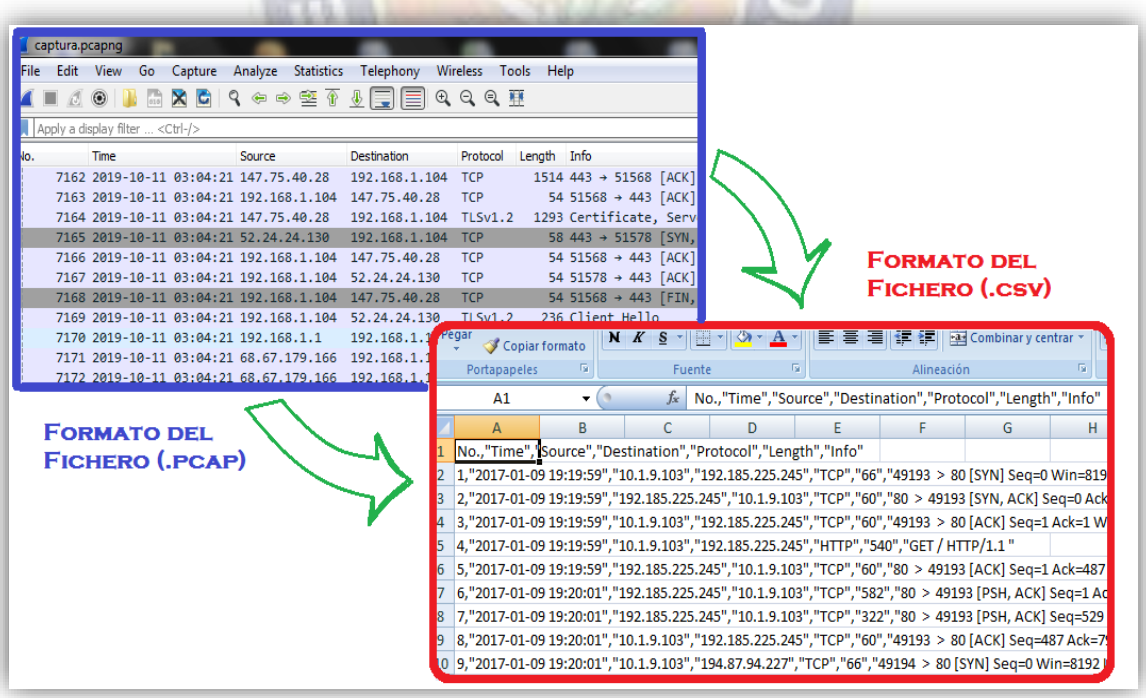


Figura 3.13. Conversión del fichero (.pcap) a una hoja de cálculo (.csv). Con la ayuda de la herramienta de Wireshark.

En la hoja de cálculo (.csv) existen algunos de los campos más destacados incluyen la fecha y hora de comienzo y finalización del flujo, dirección IP de origen y destino, puerto de origen y destino, el tipo de protocolo que se utilizo al enviar el paquete, interfaz lógica de entrada y salida, tamaño del paquete, la información de ruteo y entre otros. En la siguiente Figura 3.14 se puede apreciar todos los campos que están en el entorno de la herramienta Wireshark.

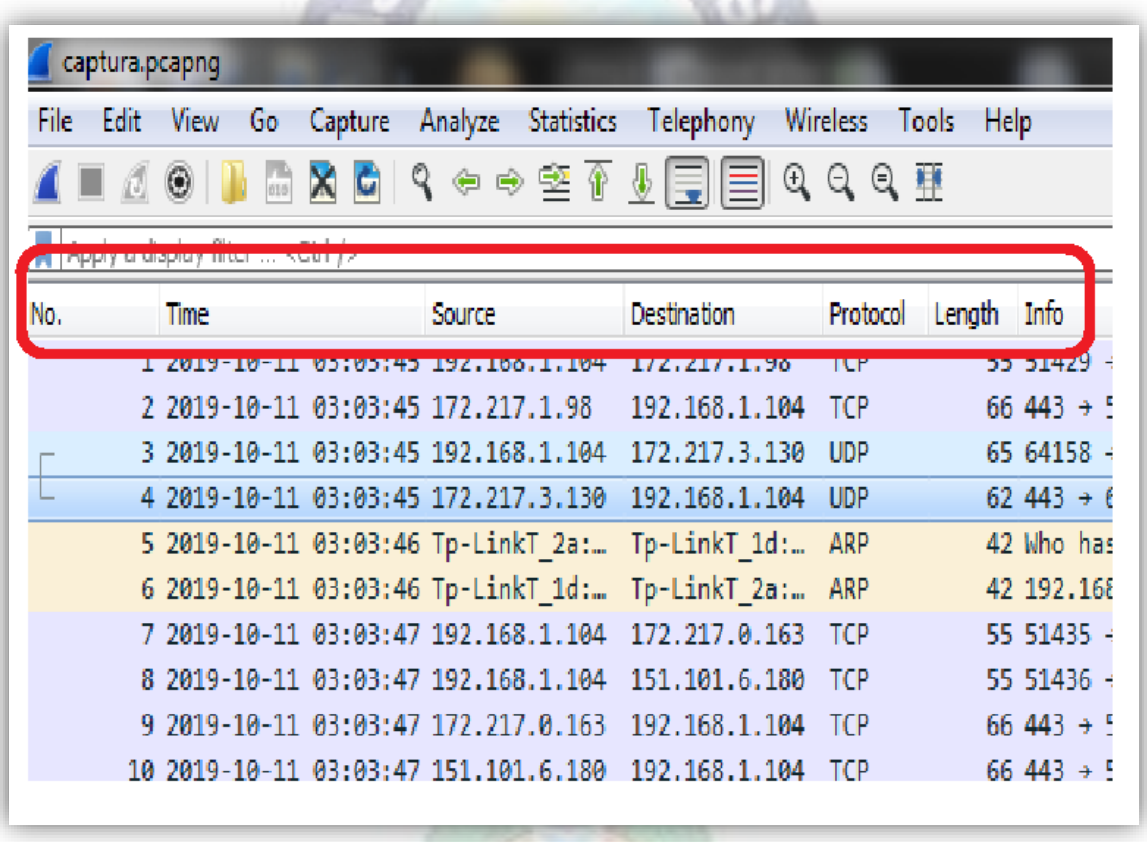


Figura 3.14. Captura de Wireshark, se aprecia los campos que nos proporciona.

El panel de captura muestra los paquetes de red capturados en un orden secuencial. Cada línea en esta lista refleja un sólo paquete capturado. Este panel inteligente divide la información en columnas y filas. Cada fila representa un sólo paquete de datos, mientras cada columna representa información adicional sobre el paquete. Las columnas con las que trabajaremos y su descripción se muestran en la siguiente tabla:

CAMPO		DESCRIPCIÓN
No.	(Número)	Representa el número de secuencia del paquete para identificar de manera única los paquetes.
Time	(Tiempo)	Representa la marca de tiempo cuando el paquete se capturó.
Source	(Origen)	Representa la dirección IP o dispositivo desde donde proviene el paquete.
Destination	(Destino)	Representa la dirección IP o dispositivo donde se dirige el paquete.
Protocol	(Protocolo)	Representa el tipo de protocolo del paquete capturado.
Length	(Longitud)	Representa el tamaño del paquete.
Info	(Información)	Representa información adicional sobre el paquete.

Tabla 3.1 Campos que utilizaran para formar el DataSet.

Ahora el DataSet sobre la captura del flujo de tráfico de red, se tiene listo para trabajar y empezar hacer la preparación de los Datos. Cabe aclarar que el estudio se mantendrá dentro del ámbito específico de la seguridad para lograr detectar malware.

3.4 Preparación de los Datos

En esta fase de la metodología se trata de preparar los datos para adecuarlos a las técnicas de minería de datos que se van a emplear sobre ellos. Esto implica seleccionar el subconjunto de datos que se va a utilizar, limpiarlos para mejorar su calidad, añadir nuevos datos a partir de los existentes y darles el formato requerido por la herramienta de modelado.

Para la preparación de los Datos y a partir de esta fase se utilizara Python que es un lenguaje de programación de alto nivel, interactivo e interpretado, es de código abierto, multi plaforma y se adecua a diversos paradigmas de programación. Debido a la continua evolución revisión del lenguaje de programación, actualmente existen dos versiones de

Python cuyo código no es compatible. Se utilizara Python 3 ya que es una versión con mejoras notables con respecto a Python 2. Python ya viene con una gran biblioteca estándar, sin embargo existen algunas **distribuciones** que pretenden extender al lenguaje con propósitos particulares.

Una de las principales distribuciones de **python** vendría siendo Anaconda, es una distribución de Python 2 y Python 3 especializada en cómputo científico, sin embargo es de muy fácil instalación y gestión tanto en Windows como en Mac OS X y GNU/Linux. Es una alternativa muy recomendable a las versiones oficiales de Python. Anaconda será la distribución que se utilizará, ahora dentro de esta distribución existen varios entornos de trabajo Spyder, JupyterLab, Jupyter Notebook y entre otros.

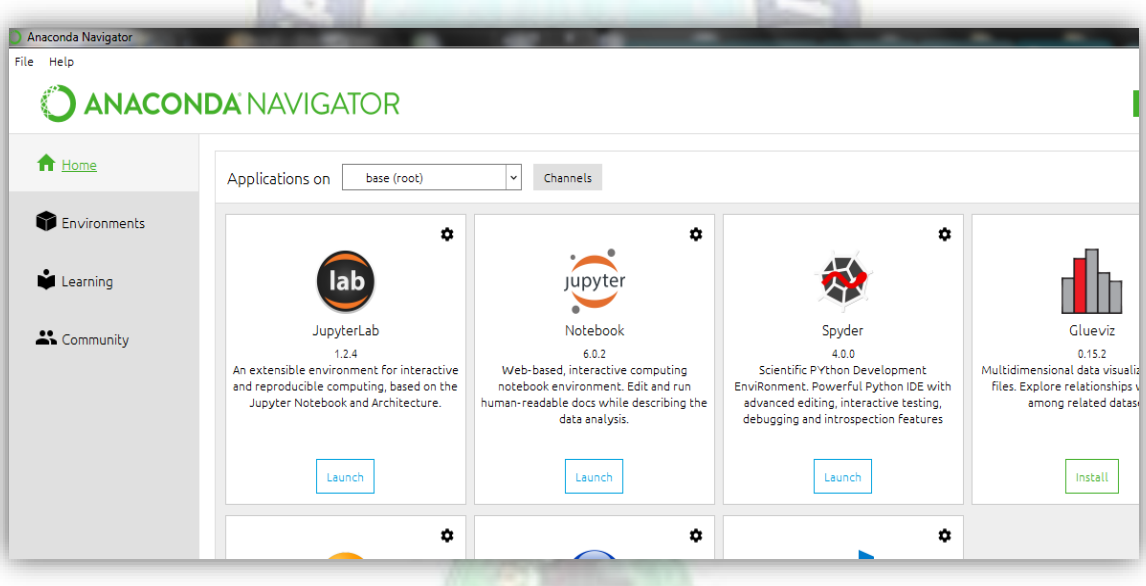


Figura 3.15. Anaconda principal distribución de Python en sistema operativo Windows. Y sus respectivos entornos de trabajo.

En adelante Jupyter Notebook será el entorno de trabajo interactivo, que permite desarrollar código en Python de manera dinámica, a la vez que integrar en un mismo documento tanto bloques de código como texto, graficas o imágenes. Al inicializar Jupyter, se abrirá en la carpeta en la que se encuentra, generalmente será el home de usuario. Se creara una carpeta donde está el cuaderno que se llamara Deteccion_de_Malware_ML.ipynb.

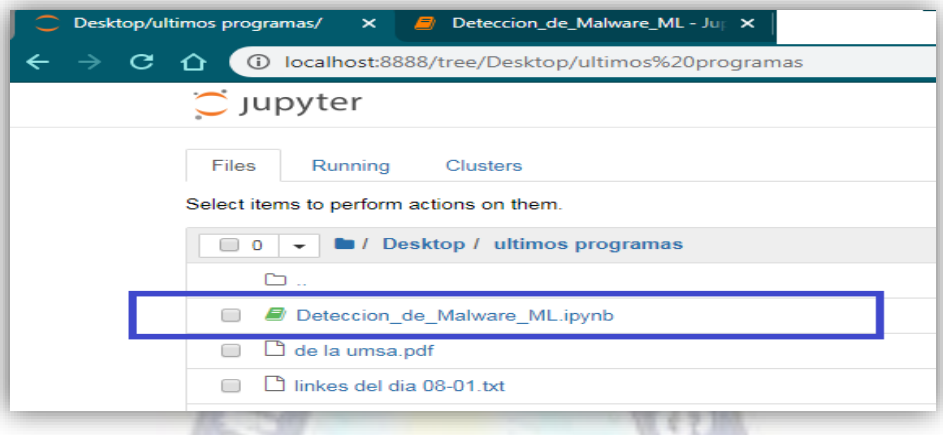


Figura 3.16. Proyecto creado con nombre Deteccion_de_Malware.ipynb.

Ahora se empieza con la preparación de los datos, es decir tener un DataSet limpio para así mandarlo al modelado y pueda tener un mejor manejo de los datos. Se tiene el DataSet en formato (.csv) lo cual no será complicado poder llamarlo al cuaderno. Se hará un simple llamamiento al fichero sin aplicar nada, con el fin de mirar cómo están los datos en bruto, esto se puede apreciar en la siguiente Figura 3.17.

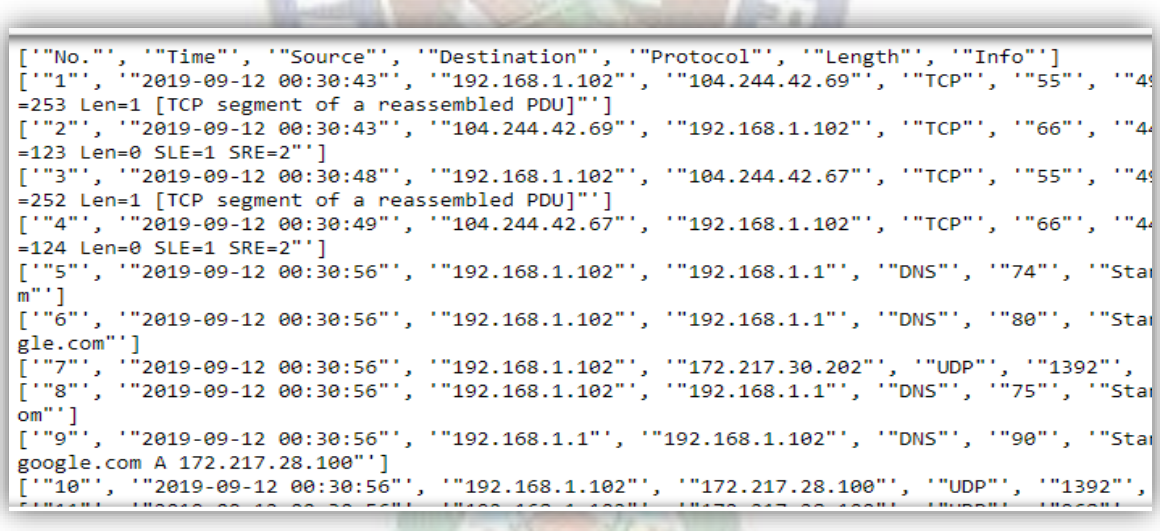


Figura 3.17. Todos los campos de datos en bruto.

Como se puede apreciar se tiene los datos en bruto, almacenada en una hoja de cálculo separados por (**,**) entonces se llevara a un Dataframe. Pero para realizar el modelo también hay que tener datos donde exista malware, entonces lo que se hizo fue conseguir un archivo

donde exista malware y se lo llevo a una hoja de cálculo, con el fin de trabajar con datos reales y sacar todo lo mejor del modelo. Antes de llevar el fichero de datos a un Dataframe, vamos a unir las dos hojas de cálculo así con el objetivo de tener solo un conjunto de datos. Una vez unido los dos ficheros, llevaremos la hoja de cálculo a un Dataframe, como se muestra en la siguiente Figura 3.18.

	Time	Source	Destination	Protocol	Length	Info
2733	2019-09-12 00:31:38	192.168.1.102	216.58.222.202	UDP	1392	55177 > 443 Len=1350
2734	2019-09-12 00:31:38	192.168.1.102	216.58.222.202	UDP	506	55177 > 443 Len=464
2735	2019-09-12 00:31:38	192.168.1.1	192.168.1.102	DNS	129	Standard query response 0x054a A ajax.googleap...
2736	2019-09-12 00:31:38	192.168.1.102	172.217.30.202	UDP	1392	55178 > 443 Len=1350
2737	2019-09-12 00:31:38	192.168.1.102	192.168.1.1	DNS	67	Standard query 0x570f A s.w.org
2738	2019-09-12 00:31:38	192.168.1.1	192.168.1.102	DNS	106	Standard query response 0x9037 A glossingly.co...
2739	2019-09-12 00:31:38	192.168.1.102	104.24.121.24	TCP	66	49690 > 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1...
2740	2019-09-12 00:31:39	192.168.1.1	192.168.1.102	DNS	83	Standard query response 0x570f A s.w.org A 192...
2741	2019-09-12 00:31:39	172.217.28.98	192.168.1.102	UDP	62	443 > 52960 Len=20
2742	2019-09-12 00:31:39	104.18.39.165	192.168.1.102	TCP	54	443 > 49685 [ACK] Seq=22839 Ack=900 Win=3174...

Figura 3.18. Datos llevados a un Dataframe. Respetando sus respectivos campos.

En el DataSet se puede ver los campos como ser el Time la hora de inicio de un paquete y su duración, Source la IP de origen, Destination la IP de destino, Protocol el protocolo que se uso para mandar el paquete, Length el tamaño del paquete y por ultimo Info que muestra la información del paquete. Lo que se hará es agregar otra columna llamada Tipo, como se muestra en la siguiente Figura 3.19.

	Time	Source	Destination	Protocol	Length	Info	Tipo
649	2017-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1276	80 > 49211 [PSH, ACK] Seq=256408 Ack=468 Win...	NORMAL
650	2017-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1514	80 > 49211 [ACK] Seq=257630 Ack=468 Win=6424...	NORMAL
651	2017-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1276	80 > 49211 [PSH, ACK] Seq=259090 Ack=468 Win...	NORMAL
652	2017-01-09 19:20:10	10.1.9.103	194.87.94.227	TCP	60	49211 > 80 [ACK] Seq=468 Ack=260312 Win=6424...	NORMAL
653	2017-01-09 19:20:10	194.87.94.227	10.1.9.103	HTTP	1491	HTTP/1.1 200 OK (application/x-msdownload)	NORMAL
654	2017-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1491	[TCP Retransmission] 80 > 49211 [PSH, ACK] S...	NORMAL
655	2017-01-09 19:20:10	10.1.9.103	194.87.94.227	TCP	60	49211 > 80 [ACK] Seq=468 Ack=261749 Win=6280...	NORMAL

Figura 3.19. Agregamos un nuevo campo (Tipo) para diferenciar los datos.

En esta columna Tipo se colocara si bien el paquete es de tipo NORMAL que se refiere a un paquete sin anomalía y MALWARE será cuando un paquete es anómalo. Esta parte del etiquetado de paquetes se puede hacer a parte pero en este caso se hara en el mismo proyecto. Lo que se hizo fue especificar que un paquete es NORMAL cuando sus IP tanto de origen y destino no exista cambia al momento de finalizar el paquete, especialmente la IP de destino es la cambia ya que es la se reporta con un usuario final, está puede cambiar pero no drásticamente, también se considero el protocolo tanto sea TCP o UDP, es importante aclarar que TCP si proporciona una transferencia de datos fiable, también proporciona control de flujo, números de secuencia, mensajes de reconocimiento y temporizadores, también TCP garantiza que los mensajes se envíen correctamente y en orden, por ultimo también proporciona control de congestión, para no colapsar los enlaces. Por otro lado UDP no proporciona control de congestión, por tanto podrá enviar los datos a cualquier velocidad sin tener en cuenta la posible saturación de los nodos.

Por lo tanto el protocolo fue pieza fundamental para aclarar el tipo, también fue el tiempo que el paquete dura y por ultimo también en función de la información del paquete a veces las IP tanto de origen y destino pueden ser normal, pero la información del paquete puede ser anómala y ya se estaría en un caso de tipo MALWARE, en resumen cada uno de los campos ayudaron al momento de agregar el tipo del paquete.

Time	Source	Destination	Protocol	Length	Info	Tipo
-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1276	80 > 49211 [PSH, ACK] Seq=256408 Ack=468 Win..	NORMAL
-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1514	80 > 49211 [ACK] Seq=257630 Ack=468 Win=6424..	NORMAL
-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1276	80 > 49211 [PSH, ACK] Seq=259090 Ack=468 Win..	NORMAL
-01-09 19:20:10	10.1.9.103	194.87.94.227	TCP	60	49211 > 80 [ACK] Seq=468 Ack=260312 Win=6424..	NORMAL
-01-09 19:20:10	194.87.94.227	10.1.9.103	HTTP	1491	HTTP/1.1 200 OK (application/x-msdownload)	NORMAL
-01-09 19:20:10	194.87.94.227	10.1.9.103	TCP	1491	[TCP Retransmission] 80 > 49211 [PSH, ACK] S..	NORMAL
-01-09 19:20:10	10.1.9.103	194.87.94.227	TCP	60	49211 > 80 [ACK] Seq=468 Ack=261749 Win=6280..	NORMAL
-01-09 19:20:10	10.1.9.103	194.87.94.227	TCP	60	49211 > 80 [RST, ACK] Seq=468 Ack=261749 Win..	NORMAL
-01-09 19:20:42	10.1.9.103	1.3.5.0	UDP	67	60884 > 6892 Len=25	MALWARE
-01-09 19:20:42	10.1.9.103	1.3.5.1	UDP	67	60884 > 6892 Len=25	MALWARE
-01-09 19:20:42	10.1.9.103	1.3.5.2	UDP	67	60884 > 6892 Len=25	MALWARE
-01-09 19:20:42	10.1.9.103	1.3.5.3	UDP	67	60884 > 6892 Len=25	MALWARE

Figura 3.20. Flujo sin anomalías NORMAL, flujo detectado anómalo MALWARE.

Ahora el DataSet se trabajara con el tiempo, pero como se pudo apreciar en la anterior Figura 3.20 la columna Time tienen tanto la fecha y la hora, lo cual no es conveniente ya que se trabajara más con las horas que con las fechas, entonces se separara en dos columnas tanto Fecha y Hora, así será más fácil de trabajar con las horas. En la siguiente Figura 3.21 se puede apreciar cómo se separa la columna Time.

	0	Fecha	Hora	Source	Destination	Protocc
230	2019-09-12 00:31:40	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TC
231	2019-09-12 00:31:40	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TC
232	2019-09-12 00:31:40	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TC
233	2019-09-12 00:31:40	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TC
234	2019-09-12 00:31:40	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TC
235	2019-09-12 00:31:40	2019-09-12	00:31:40	192.168.1.102	104.18.39.165	TC
236	2019-09-12 00:31:40	2019-09-12	00:31:40	192.168.1.102	172.217.28.98	UD

Figura 3.21. Se divide el campo (Time) en dos columnas Fecha y Hora.

Una vez separado la columna de Time en otras dos tanto Fecha y Hora como se aprecia en la Figura 3.21, se trabajara con el campo de Hora, cuando un paquete tiene una duración de más de 5 segundos a veces este puede ser normal o anómalo, hay casos que cuando se utiliza el protocolo de UDP tiene una duración de más de 40 segundos lo que no es normal, como se sabe que el protocolo UDP no proporciona control de congestión, por tanto podrá enviar los datos a cualquier velocidad sin tener en cuenta la posible saturación de los nodos.

Y esto puede ser anómalo pero identificar un solo paquete a simple vista es muy complejo y se puede pasar sin darse cuenta que ahí hay una anomalía. En la siguiente Figura 3.22 se podrá ver cuál es la diferencia de duración en el saludo que tiene un paquete expresado en segundos.

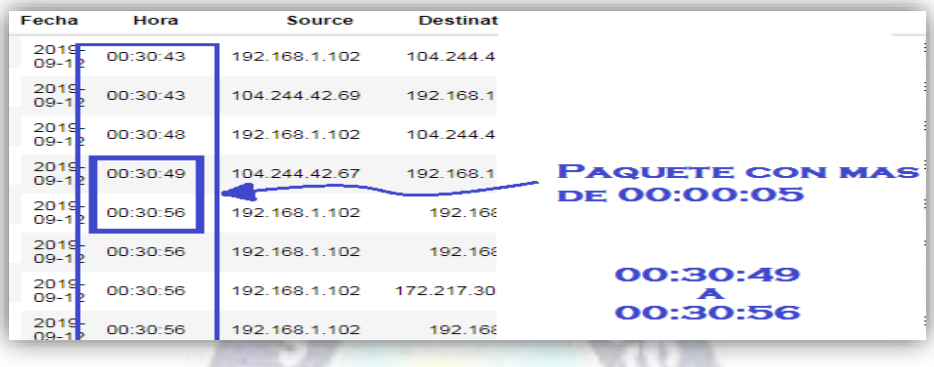


Figura 3.22. Paquete con duración de 5 segundos. Comportamiento extraño.

Entonces aquellos paquetes que duren más de 4 segundos se extenderán, será pieza fundamental para el preparado de datos, para expandir se trabajo tanto con el protocolo, ya como se menciono el protocolo UDP puede ser normal o anormal porque no tiene un tiempo de control de congestión y puede mandar varios paquetes con el fin de saturar la red, otro detalle también será las IP de destino ya que cuando estas son consecutivas vendría siendo una actividad anómala, también la información del paquete será pieza fundamental para ver qué tipo de información está mandando y también se podrá ver si el tamaño del paquete cambie cada vez que van enviando los paquetes. Y por ultimo también se verá que la IP de origen no se repita cuando la IP de destino sea consecutiva o simplemente cambie drásticamente de IP, todas estas características serán importantes para tener un DataSet más limpio.

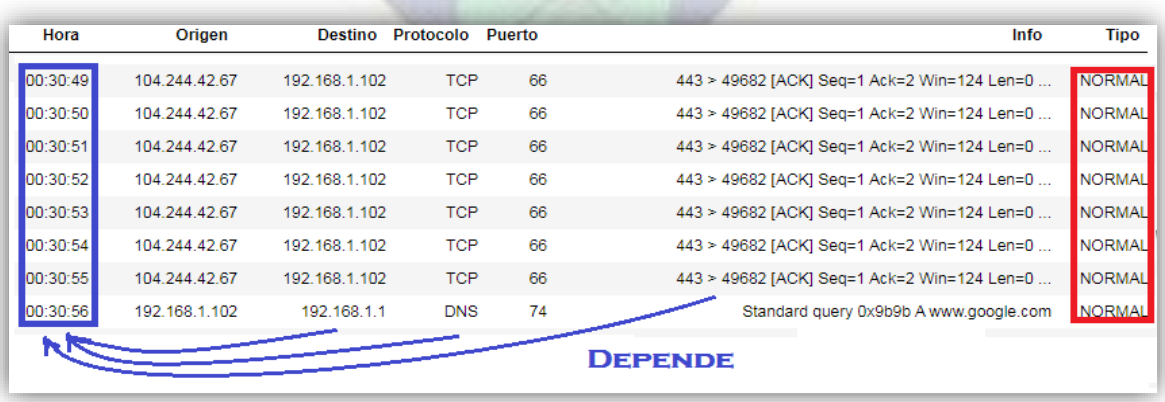


Figura 3.23. El etiquetado de la columna (Tipo) dependerá de todos los campos, pero en función de la columna (Hora). Sera un proceso muy importante.

En esta parte se pasa mucho tiempo para ver como se puede tener un DataSet más limpio y manejable, al final se trabajo con el campo de Hora y no dejar de lado las demás columnas que fueron de gran ayuda al momento del etiquetado.

Ahora se agrupara varios paquetes y se enfocaran en varios campos del DataSet, se agrupara por la Hora, Origen, Protocolo y Tipo, es decir aquellos paquetes que en un determinado tiempo tuvieron varios envíos de una determinada IP de Origen. También se agrupara por los protocolos que fueron enviados todos los paquetes y al final por el tipo de paquetes. Un detalle aclarar que se trabaja con la IP de Origen y no con la IP de destino, es porque la IP de destino siempre será cambiante pero la IP de origen será el host que ha sido infectado por el ransomware, si bien la IP de origen manda paquetes a una IP de destino que esta fuera de la red, el culpable no es la de destino será la del origen.

El DataSet se redujo en pocas columnas Hora, IP de origen, Protocolo y Tipo, ahora se agregara una columna más; que se llamara Contador, este campo lo que hará es contar cuantas IP de Origen enviaron varios paquetes, bajo qué tiempo y bajo que protocolo se enviaron dichos paquetes, como se aprecia en la siguiente Figura 3.24.

Hora	Origen	Protocolo	Tipo	Contador
19:20:07	10.1.9.103	TCP	NORMAL	11
19:20:07	194.87.94.227	HTTP	NORMAL	1
19:20:07	194.87.94.227	TCP	NORMAL	22
19:20:08	10.1.9.103	TCP	NORMAL	53
19:20:08	194.87.94.227	TCP	NORMAL	147
19:20:09	10.1.9.103	TCP	NORMAL	41
19:20:09	194.87.94.227	TCP	NORMAL	113
19:20:10	10.1.9.103	TCP	NORMAL	47
19:20:10	194.87.94.227	HTTP	NORMAL	1
19:20:10	194.87.94.227	TCP	NORMAL	130
19:20:11	10.1.9.103	TCP	NORMAL	1
19:20:12	10.1.9.103	TCP	NORMAL	1

Figura 3.24. Agrupando por los campos Hora luego la IP de Origen y el Protocolo por el que se envió el paquete. Contaremos la cantidad de envíos de paquetes (Contador).

Para que funcionen mejor muchos algoritmos de Machine Learning usados en Data Science, hay que normalizar las variables de entrada al algoritmo. Normalizar significa, en este caso, comprimir o extender los valores de la variable que estén en un rango definido. Sin embargo, una mala aplicación de la normalización, o una elección descuidada del método de normalización puede arruinar los datos y con ello el análisis.

Se utilizara el escalado de variables (Feature Scaling o MinMax Scaler)

$$X_{normalizacion} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Es un método simple y consiste en reescalar el rango de características para escalar el rango en [0,1] [-1,1]. Seleccionar el rango objetivo depende de la naturaleza de los datos. Lo que se hará es aplicar la normalización a la columna Contador.

	Contador	Contador_nor
196	2	0.003145
197	2	0.003145
198	73	0.226415
199	6	0.015723
200	30	0.091195
201	10	0.028302
202	26	0.078616
203	3	0.006289
204	4	0.009434
205	1	0.000000
206	2	0.003145
207	132	0.411950

Figura 3.25. Normalizamos la columna Contador creando un nuevo campo Contador_nor.

Se aplica la normalización, creando una columna llamada Contador_nor este campo estaría listo para mandarlo al modelo de predicción. Este proceso de refinamiento de los datos sería un último pasó antes de aplicar los algoritmos de Machine Learning.

3.5 Modelado

En esta fase, se selecciona y aplica las técnicas de modelado que sean pertinentes al problema cuantas más sea mejor, y se calibran sus parámetros a valores óptimos. Típicamente hay varias técnicas para el mismo tipo de problema de minería de datos. Por lo tanto, casi siempre en cualquier proyecto se acaba volviendo a la fase de preparación de datos.

Una vez que se tiene normalizado nuestro Data se evalúa con varios algoritmos, para ver cómo se comportan cada uno con nuestros datos que tenemos. Algunos tal vez se acomoden a nuestras expectativas. Antes recordar que solo se aplicara los algoritmos, en la fase de evaluación se analizara con cual modelo se quedara para así finalizar el modelo.

La detección de valores atípicos es similar a la detección de novedades en el sentido de que el objetivo es separar un núcleo de observaciones regulares de algunas contaminantes llamadas valores atípicos. Sin embargo, en el caso de la detección de valores atípicos, no tenemos un conjunto de datos limpio que represente la población de observaciones regulares que se pueden utilizar para entrenar cualquier herramienta.

- **Robust Covariance.-** Una forma común de realizar de realizar una detección de valores atípicos es asumir que los datos regulares provienen de una distribución conocida. A partir de esta suposición, generalmente se intenta definir la forma de los datos, y se puede definir las observaciones periféricas como observaciones que se encuentran lo suficientemente lejos de la forma ajustada.

  El procedimiento del algoritmo, se ajusta a una estimación de covarianza robusta a los datos y, por lo tanto, ajusta una elipse a los puntos de datos centrales, ignorando los puntos fuera del modo central.

  Se aplica los datos al modelo de predicción, el resultado de este procedimiento se puede apreciar en la siguiente Figura 3.26.

Hora	Origen	Protocolo	Tipo	Contador	Prediccion
00:31:39	104.24.121.24	TCP	NORMAL	5	1
00:31:39	104.24.121.24	TLSv1.2	NORMAL	6	1
00:31:39	172.217.28.100	UDP	NORMAL	3	1
00:31:39	172.217.28.98	UDP	MALWARE	15	1
00:31:39	172.217.28.98	UDP	NORMAL	4	1
00:31:39	172.217.30.202	UDP	MALWARE	30	1
00:31:39	172.217.30.202	UDP	NORMAL	6	1
00:31:39	192.16.58.8	OCSP	NORMAL	1	1
00:31:39	192.168.1.1	DNS	NORMAL	1	1
00:31:39	192.168.1.102	HTTP	NORMAL	1	1
00:31:39	192.168.1.102	TCP	NORMAL	69	1
00:31:39	192.168.1.102	TLSv1.2	NORMAL	9	1
00:31:39	192.168.1.102	UDP	MALWARE	25	1

Figura 3.26. Se aplica el Modelo de Robust Covariance, creando un nuevo campo llamado Predicción (Anomalía [-1] y No anomalía [1]).

Ahora se puede ver los datos donde la predicción es [-1], lo que significa que es una actividad anómala y también es de tipo MALWARE, es decir que tanto el modelo de predicción funciona para predecir. En la siguiente Figura 3.27 se puede ver que si bien 8 de los datos son MALWARE y están predichos que son anómalos, esto no implica que el modelo tiene un nivel de predicción optimo, para demostrar que el modelos tiene un buen nivel de aceptación se verá más adelante en la fase de Evaluación.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
207	00:31:41	172.217.28.98	UDP	MALWARE	132	-1
225	00:31:42	172.217.28.97	UDP	MALWARE	108	-1
235	00:31:42	192.168.1.102	UDP	MALWARE	81	-1
254	00:31:43	172.217.28.97	UDP	MALWARE	90	-1
341	19:20:42	10.1.9.103	UDP	MALWARE	318	-1
342	19:20:43	10.1.9.103	UDP	MALWARE	256	-1
345	19:20:46	10.1.9.103	UDP	MALWARE	319	-1
346	19:20:47	10.1.9.103	UDP	MALWARE	256	-1

Figura 3.27. Son 8 datos que predice que son anómalos y de tipo MALWARE.

Pero no todo modelo de predicción es muy bueno, en la siguiente Figura 3.28 se puede apreciar que existen 5 datos donde son de tipo NORMAL y el modelo predijo [-1], es decir que son anomalías. Y este margen de error se puede reducir para que el modelo sea más exacto, para mejorar el modelo hay que realizar un mejor tratamiento de los datos antes del modelado.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
171	00:31:39	104.18.39.165	TLSv1.2	NORMAL	81	-1
190	00:31:40	104.18.39.165	TCP	NORMAL	108	-1
304	19:20:08	194.87.94.227	TCP	NORMAL	147	-1
306	19:20:09	194.87.94.227	TCP	NORMAL	113	-1
309	19:20:10	194.87.94.227	TCP	NORMAL	130	-1

Figura 3.28. Son 5 datos que predice que son anómalos, pero en este caso son de tipo NORMAL.

Hay que aclarar que el modelo de predicción, predice [-1] que significa que se considera una anomalía y se compara con nuestro tipo MALWARE, no significa que nuestro modelo es optimo. La evaluación del modelo se verá en la siguiente fase, solo se puede decir por el momento que nuestro modelo de predicción funciona.

- **SVM One-Class.-** Considere un conjunto de datos de n observaciones de la misma distribución descrita por p características. Considere ahora que se agrega una observación más a ese conjunto de datos. En general, se trata de aprender una frontera aproximada y delimitada que delimita el contorno de la distribución de observaciones iníciales, trazada en la incrustación p- dimensional espacio. Luego, si hay más observaciones dentro del subespacio delimitado por fronteras, se considera que provienen de la misma población que las observaciones iníciales. De lo contrario, si se encuentran fuera de la frontera, se puede decir que son anormales con una confianza dada en nuestra evaluación.

  Se aplica los datos en el modelo de predicción, se puede apreciar en la Figura 3.29.

Hora	Origen	Protocolo	Tipo	Contador	Prediccion
00:31:40	192.168.1.102	UDP	NORMAL	10	-1
00:31:41	104.18.39.165	TCP	NORMAL	26	-1
00:31:41	104.18.39.165	TLSv1.2	NORMAL	3	1
00:31:41	104.248.100.36	TCP	NORMAL	4	1
00:31:41	172.217.28.97	UDP	MALWARE	1	-1
00:31:41	172.217.28.97	UDP	NORMAL	2	1
00:31:41	172.217.28.98	UDP	MALWARE	132	-1
00:31:41	172.217.28.98	UDP	NORMAL	13	1
00:31:41	192.168.1.1	DNS	NORMAL	4	1
00:31:41	192.168.1.102	DNS	NORMAL	4	1
00:31:41	192.168.1.102	TCP	NORMAL	24	1
00:31:41	192.168.1.102	TLSv1.2	NORMAL	4	1
00:31:41	192.168.1.102	UDP	MALWARE	21	-1

Figura 3.29.Aplicamos el Modelo de SVM One-Class, creando un nuevo campo llamado Predicción (Anomalía [-1] y No anomalía [1]).

En la siguiente Figura3.30 los datos donde la predicción es [-1], lo que significa que es una actividad anómala y también es de tipo MALWARE, es decir que tanto el modelo de predicción funciona para predecir. Pero estos datos son más de 100 como se muestra en la siguiente Figura 3.30, el modelo predijo que todas estas actividades son anómalas, si solo lo se compara con el anterior modelo los datos de predicción es más reducido y solo por el momento se puede suponer que su nivel de predicción no es tan bueno como el anterior modelo.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
14	00:30:56	172.217.28.100	UDP	MALWARE	1	-1
30	00:30:58	172.217.28.100	UDP	MALWARE	7	-1
32	00:30:58	172.217.28.110	UDP	MALWARE	1	-1
34	00:30:58	172.217.28.98	UDP	MALWARE	1	-1
51	00:31:00	192.168.1.102	UDP	MALWARE	6	-1
60	00:31:05	192.168.1.102	UDP	MALWARE	6	-1
63	00:31:07	192.168.1.102	UDP	MALWARE	1	-1
64	00:31:08	172.217.28.100	UDP	MALWARE	1	-1
67	00:31:09	172.217.28.100	UDP	MALWARE	1	-1
72	00:31:11	172.217.28.100	UDP	MALWARE	31	-1
81	00:31:12	192.168.1.102	UDP	MALWARE	6	-1
90	00:31:14	192.168.1.102	UDP	MALWARE	7	-1
94	00:31:15	172.217.28.110	UDP	MALWARE	1	-1
96	00:31:15	172.217.28.98	UDP	MALWARE	1	-1
100	00:31:16	172.217.28.100	UDP	MALWARE	6	-1
114	00:31:20	192.168.1.102	UDP	MALWARE	1	-1
122	00:31:22	172.217.28.100	UDP	MALWARE	70	-1
124	00:31:22	192.168.1.102	UDP	MALWARE	10	-1
177	00:31:39	172.217.30.202	UDP	MALWARE	30	-1

Figura 3.30. Son más de 100 datos que predice que son anómalos y de tipo MALWARE.

Su margen de error de este modelo no es nada bueno, en la siguiente Figura 3.31 se puede apreciar que existen más de 200 datos donde son de tipo NORMAL y el modelo predijo [-1], es decir que son anomalías. Lo cual que por el momento se puede decir que su margen de error es un grande.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
0	00:30:43	104.244.42.69	TCP	NORMAL	1	-1
1	00:30:43	192.168.1.102	TCP	NORMAL	1	-1
2	00:30:44	104.244.42.69	TCP	NORMAL	1	-1
3	00:30:45	104.244.42.69	TCP	NORMAL	1	-1
4	00:30:46	104.244.42.69	TCP	NORMAL	1	-1
5	00:30:47	104.244.42.69	TCP	NORMAL	1	-1
6	00:30:48	192.168.1.102	TCP	NORMAL	1	-1
7	00:30:49	104.244.42.67	TCP	NORMAL	1	-1
8	00:30:50	104.244.42.67	TCP	NORMAL	1	-1
9	00:30:51	104.244.42.67	TCP	NORMAL	1	-1
0	00:30:52	104.244.42.67	TCP	NORMAL	1	-1

Figura 3.31. Son más de 200 datos que predice que son anómalos, pero en este caso son de tipo NORMAL. Margen de error nada bueno.

Se tiene que aclarar que el modelo de predicción, predice [-1] que significa que se considera una anomalía y se compara con nuestro tipo MALWARE, no significa que nuestro modelo es optimo. La evaluación del modelo se verá en la siguiente fase, solo se puede suponer por el momento que el modelo de predicción funciona. Pero con solo ver los datos de predicción y el margen de error es grande, se puede decir que su evaluación no será muy buena.

- **Isolation Forest**.- Una forma eficiente de realizar la detección de valores atípicos en conjuntos de datos de alta dimensión es utilizar bosques aleatorios, es decir que se seleccionan aleatoriamente una característica y luego seleccionan aleatoriamente un valor dividido entre los valores máximo y mínimo de la característica seleccionada.

Dado que la partición recursiva se puede representar mediante una estructura de árbol, el número de divisiones necesarias para aislar una muestra es equivalente a la longitud de la ruta desde el nodo raíz hasta el nodo de terminación. La partición aleatoria produce rutas notablemente más cortas para las anomalías. Por lo tanto, cuando un bosque de arboles aleatorios produce colectivamente longitudes de camino más cortas para muestras particulares, es muy probable que sean anomalías.

Hora	Origen	Protocolo	Tipo	Contador	Prediccion
19:20:34	10.1.9.103	TCP	NORMAL	1	1
19:20:35	10.1.9.103	TCP	NORMAL	1	1
19:20:36	10.1.9.103	TCP	NORMAL	1	1
19:20:37	10.1.9.103	TCP	NORMAL	1	1
19:20:38	10.1.9.103	TCP	NORMAL	1	1
19:20:39	10.1.9.103	TCP	NORMAL	1	1
19:20:40	10.1.9.103	TCP	NORMAL	1	1
19:20:41	10.1.9.103	TCP	NORMAL	1	1
19:20:42	10.1.9.103	UDP	MALWARE	318	-1
19:20:43	10.1.9.103	UDP	MALWARE	256	-1
19:20:44	10.1.9.103	UDP	NORMAL	1	1
19:20:45	10.1.9.103	UDP	NORMAL	1	1

Figura 3.32. Se aplica el Modelo de Isolation Forest, creando un nuevo campo llamado Predicción (Anomalía [-1] y No anomalía [1]).

Ahora se puede ver los datos donde la predicción es [-1], lo que significa que es una actividad anómala y también es de tipo MALWARE, es decir que tanto el modelo de predicción funciona para predecir. Se puede ver en la siguiente Figura 3.33 que son 8 los datos que se lograron predecir con exactitud, este modelo se puede comparar al Robust Covariance. Luego se podrá apreciar qué sucede con sus respectivas evaluaciones, tal vez tengan los mismos resultados pero su capacidad de predicción no sea tan exacta.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
207	00:31:41	172.217.28.98	UDP	MALWARE	132	-1
225	00:31:42	172.217.28.97	UDP	MALWARE	108	-1
235	00:31:42	192.168.1.102	UDP	MALWARE	81	-1
254	00:31:43	172.217.28.97	UDP	MALWARE	90	-1
341	19:20:42	10.1.9.103	UDP	MALWARE	318	-1
342	19:20:43	10.1.9.103	UDP	MALWARE	256	-1
345	19:20:46	10.1.9.103	UDP	MALWARE	319	-1
346	19:20:47	10.1.9.103	UDP	MALWARE	256	-1

Figura 3.33. Son 8 datos que predice que son anómalos y de tipo MALWARE.

Pero el modelo de predicción no es muy bueno, en la siguiente Figura 3.34 se puede apreciar que existen 5 datos donde son de tipo NORMAL y el modelo predijo [-1], es decir que son anomalías. Y este margen de error se puede reducir para que el modelo sea más exacto, para mejorar el modelo hay que realizar un mejor tratamiento de los datos antes del modelado.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
171	00:31:39	104.18.39.165	TLSv1.2	NORMAL	81	-1
190	00:31:40	104.18.39.165	TCP	NORMAL	108	-1
304	19:20:08	194.87.94.227	TCP	NORMAL	147	-1
306	19:20:09	194.87.94.227	TCP	NORMAL	113	-1
309	19:20:10	194.87.94.227	TCP	NORMAL	130	-1

Figura 3.34. Son 5 datos que predice que son anómalos, pero en este caso son de tipo NORMAL.

Por último aclarar que el modelo de predicción, predice [-1] que significa que se considera una anomalía y se compara con nuestro tipo MALWARE, no significa que nuestro modelo es óptimo. La evaluación del modelo se verá en la siguiente fase, solo se puede decir por el momento que el modelo de predicción funciona. Un detalle que se pudo apreciar fue que los datos de predicción y el margen de error, se asemejan demasiado al Robust Covariance, en la fase de evaluación se verá cual tiene un nivel de predicción aceptable.

- **Factor de valor atípico local.-** Calcula una puntuación que refleja el grado de anormalidad de las observaciones. Mide la desviación de densidad local de un punto de datos dado con respecto a sus vecinos. La idea es detectar las muestras que tienen una densidad sustancialmente menor que sus vecinas. En la práctica, la densidad local se obtiene de los k vecinos más cercanos. La puntuación LOF de una observación es igual a la razón de la densidad local promedio de sus k vecinos más cercanos y su propia densidad local: se espera que una instancia normal tenga una densidad local similar a la de sus vecinos, mientras que los datos anormales se espera que tenga una densidad local mucho menor.

	Hora	Origen	Protocolo	Tipo	Contador	Prediccion
0	00:30:43	104.244.42.69	TCP	NORMAL	1	1
1	00:30:43	192.168.1.102	TCP	NORMAL	1	1
2	00:30:44	104.244.42.69	TCP	NORMAL	1	1
3	00:30:45	104.244.42.69	TCP	NORMAL	1	1
4	00:30:46	104.244.42.69	TCP	NORMAL	1	1
5	00:30:47	104.244.42.69	TCP	NORMAL	1	1
6	00:30:48	192.168.1.102	TCP	NORMAL	1	1
7	00:30:49	104.244.42.67	TCP	NORMAL	1	1
8	00:30:50	104.244.42.67	TCP	NORMAL	1	1
9	00:30:51	104.244.42.67	TCP	NORMAL	1	1
10	00:30:52	104.244.42.67	TCP	NORMAL	1	1

Figura 3.35.Se aplica el Modelo de Local Outlier Factor, creando un nuevo campo llamado Predicción (Anomalía [-1] y No anomalía [1]).

Este modelo queda descartado, por motivo que el algoritmo LOF tiene en cuenta las propiedades locales y globales de los conjuntos de datos. Es decir que los datos donde las muestras anormales tienen diferentes densidades subyacentes. Además otro detalle al aplicar este modelo no tiene la opción de predict y cuando se aplica el fit_predict se usa en los nuevos datos no vistos. Este modelo no será necesario que pase por la fase de evaluación lo que significa que este modelo no funciona para los datos que se maneja.

3.6 Evaluación

Se llegaron a construir varios modelos que parecen alcanzar la calidad suficiente desde la perspectiva de análisis de datos. Antes de proceder al despliegue final del modelo, es importante evaluarlo a fondo y revisar los pasos ejecutados para crearlo, comparar el modelo obtenido con los objetivos de negocio. Un objetivo clave es determinar si hay alguna cuestión importante de negocio que no haya sido considerada suficientemente. Al final de esta, se debería obtener una decisión sobre la aplicación de los resultados del proceso de análisis de datos.

En esta fase se evaluará a todos los modelos que tenemos, por su precisión de entrenamiento, su precisión de evaluación y al final se analizara su matriz de confusión. Se tiene tres modelos a evaluar, el Robust Covariance, SVM one-class y por ultimo Isolation Forest. Una vez evaluados los modelos se realiza una pequeña tabla de comparación para mostrar y luego elegir cual modelo tiene un buen nivel de precisión.

Para la evaluación de precisión tanto de entrenamiento como de evaluación hay que entender uno de los conceptos más importantes en Machine Learning, que es overfitting o también llamado sobreajuste del modelo. Comprender como un modelo se ajusta a los datos es muy importante para entender las causas de baja precisión en las predicciones. Un modelo va a estar sobre ajustado cuando vemos que se desempeña bien con los datos de entrenamiento, pero su precisión es notablemente más baja con los datos de evaluación, esto se debe a que el modelo ha memorizado los datos que ha visto y no pudo generalizar las reglas para predecir los datos que no ha visto. De aquí también la importancia de siempre contar con dos conjuntos de datos distintos, uno para entrenar el modelo y otro para evaluar su precisión; ya que si se utiliza el mismo dataset para los dos tareas, no tendríamos forma de determinar como el modelo se comporta con datos que nunca ha visto.

- **Robust Covariance.-** El modelo no cuenta con dos conjuntos de datos, de tal manera se agarra el conjunto de datos y se divide en dos conjuntos de datos [X,y]; una parte

será para evaluar la precisión de entrenamiento y otra parte será la precisión de evaluación.

```
# separ los datos en train y eval
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.35,train_size=0.65,random_state=42)

# creando el modelo sin control de profundidad, va a continuar hasta
# que todas las hojas sean puras
arbol = DecisionTreeClassifier(criterion='entropy')

# Ajustando el modelo
arbol.fit(x_train, y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Figura 3.36. Parámetros puestos para la evaluación del Robust Covariance.

El algoritmo de Robust Covarience que se implemento en nuestro modelo puede ser preciso, pero también con mucha tendencia al sobreajuste. Para construir estos modelos se aplica un procedimiento recursivo para encontrar los atributos que proporcionan más información sobre distintos subconjuntos de datos, cada vez más pequeños. Si se aplica este procedimiento en forma reiterada, eventualmente se llegara a un árbol en el que cada hoja tenga una sola instancia de nuestro objetivo a clasificar. En este caso extremo, el árbol de decisión va a tener una pobre generalización y estar bastante sobre ajustado; ya que cada instancia de los datos de entrenamiento encontrara el camino que lo lleve eventualmente a la hoja que lo contiene, alcanzando así una precisión muy alta con los datos de entrenamiento y muy baja en la precisión de evaluación.

```
print("precisión entranamiento: {0: .2f}".format(arbol.score(x_train, y_train)))
```

```
precisión entranamiento:  0.99
```

Figura 3.37. Evaluación del modelo, precisión de entrenamiento al 99%.

Se evaluó en nivel de precisión de entrenamiento del modelo de Robust Covariance, tiene una precisión de entrenamiento del 99%; entonces se puede decir que el entrenamiento se comporto muy bien. Ahora vamos con la precisión de evaluación del modelo.

```
# precisión del modelo en datos de evaluación.
print("precisión evaluación: {0: .2f}".format(arbol.score(x_test, y_test)))
```

```
precisión evaluación:  0.96
```

Figura 3.38. Evaluación del modelo, precisión de evaluación del modelo al 96%.

La precisión de evaluación del modelo es del 96%; lo que indica que el modelo Robust Covarience no es tan preciso pero quedo muy cerca de serlo, no todo modelo puede salir perfecto porque caería en el sobre ajuste, y si tuviera una precisión de evaluación muy baja también se diría que cae en el sobre ajuste.

Por tanto el modelo tiene un buen nivel de precisión, ahora se graficara una matriz de confusión del modelo con la ayuda de las librerias de sklearn la misma librería que ofrece el algoritmo Robust Covariance. Al final de todas las evaluaciones de cada modelo se hara una tabla donde se verá con cual se queda para la última fase de despliegue.

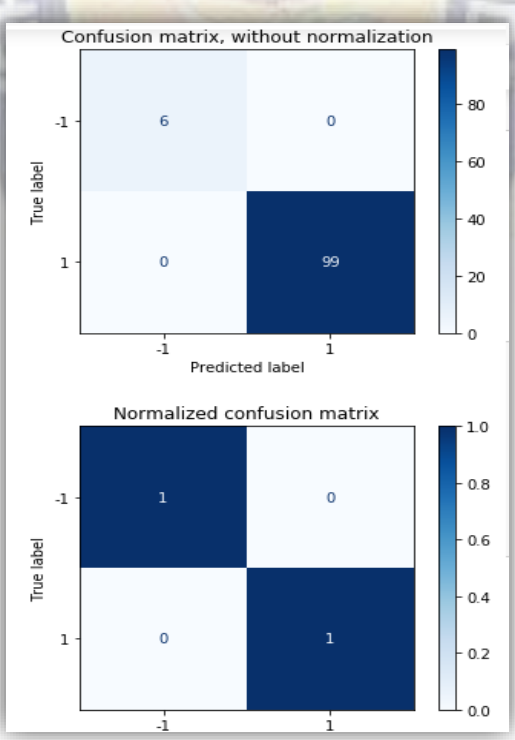


Figura 3.39. Gráfica de la matriz de confusión utilizando librerías, matriz de confusión sin normalización (arriba) y matriz de confusión con normalización (abajo).

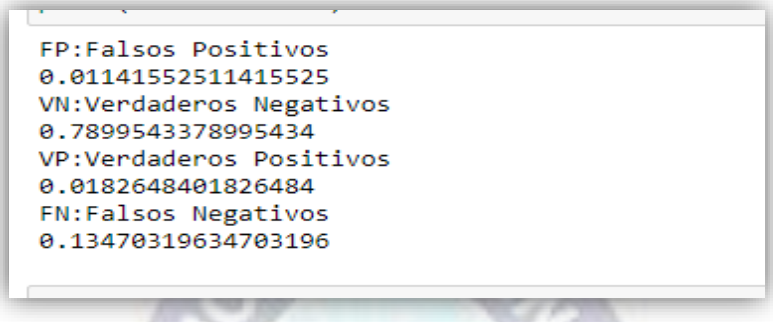
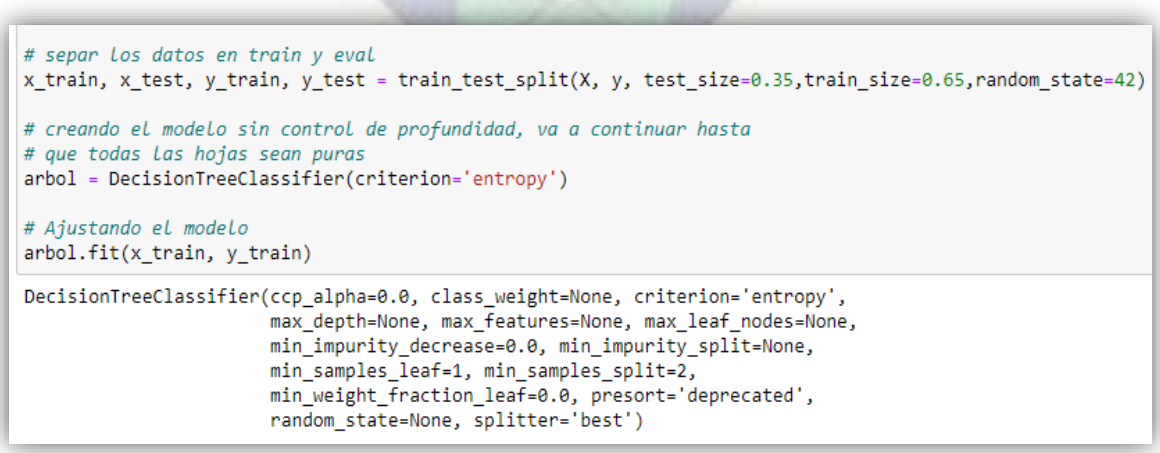


Figura 3.40. Average (Promedio) del modelo de Falsos Positivos, Verdaderos Negativos, Verdaderos Positivos y Falsos Negativos.

El promedio de datos de la matriz solo ayuda a ver si es aceptable el modelo, en el siguiente Capítulo 4 se evaluara la matriz de confusión mas afondo. Por otro lado, los Falsos Positivos 0.011455 (FP) sería el promedio que el algoritmo se confunde y dice que es una Anomalía cuando en realidad es un dato Normal. También se puede ver que los Verdaderos Positivos 0.0182648 (VP) sería el promedio que el algoritmo predice que son anomalías y resulta que también son de tipo Malware.

- **SVM One-Class.-** El modelo no cuenta con dos conjuntos de datos, de tal manera se agarra el conjunto de datos y lo se divide en dos conjuntos de datos [X,y]; una parte será para evaluar la precisión de entrenamiento y otra parte será la precisión de evaluación.



```
# separ los datos en train y eval
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.35,train_size=0.65,random_state=42)

# creando el modelo sin control de profundidad, va a continuar hasta
# que todas las hojas sean puras
arbol = DecisionTreeClassifier(criterion='entropy')

# Ajustando el modelo
arbol.fit(x_train, y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Figura 3.41. Parámetros puestos para la evaluación del SVM One - Class.

Hay que aclarar como el anterior modelo; el algoritmo SVM One-Class que se implemento en el modelo puede ser una herramienta precisa, pero también con mucha tendencia al sobreajuste. Para construir este modelo, igual que el Robust Covariance se aplica un procedimiento recursivo para encontrar los atributos que se proporcionara más información sobre distintos subconjuntos de datos, cada vez más pequeños. Si se aplica este procedimiento en forma reiterada, eventualmente se podrá llegar a un árbol en el que cada hoja tenga una sola instancia del objetivo a clasificar.

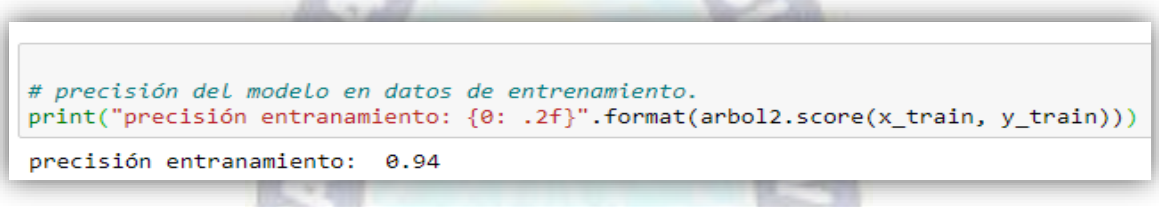


Figura 3.42. Evaluación del modelo, precisión de entrenamiento al 94%.

Se evaluó en nivel de precisión de entrenamiento del modelo de SVM One-Class, tiene una precisión de entrenamiento del 94%; entonces se puede decir que el entrenamiento se comporto muy bien. Si se compara con el Robust Covariance que tuvo una precisión de entrenamiento del 99% se puede decir que no es tan preciso porque le lleva por 5%; pero lo importante será como será su precisión de evaluación. Ahora se verá con la precisión de evaluación del modelo.

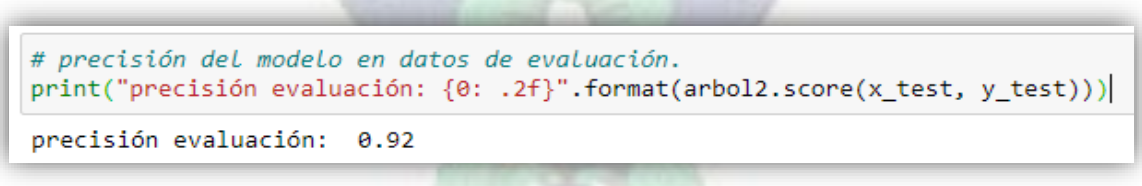


Figura 3.43. Evaluación del modelo, precisión de evaluación del modelo al 92%.

La precisión de evaluación del modelo es del 92%; lo que indica que el modelo SVM One-Class no es tan preciso pero quedo muy cerca de serlo, no todo modelo puede salir perfecto porque caería en el sobre ajuste, y si tuviera una precisión de evaluación muy baja también se diría que cae en el sobre ajuste. Si se hace una comparación con el anterior modelo de Robust Covariance se puede decir que SVM One-Class tiene una mejor precisión en la evaluación pero para precisión de entrenamiento no es muy buena.

Por tanto el modelo tiene un buen nivel de precisión, ahora se graficara una matriz de confusión del modelo con la ayuda de las librerias de sklearn la misma librería que nos ofreció el algoritmo SVM One-Class. Al final de todas las evaluaciones de cada modelo se hará una tabla donde se verá con cual se quedara para la última fase de despliegue.

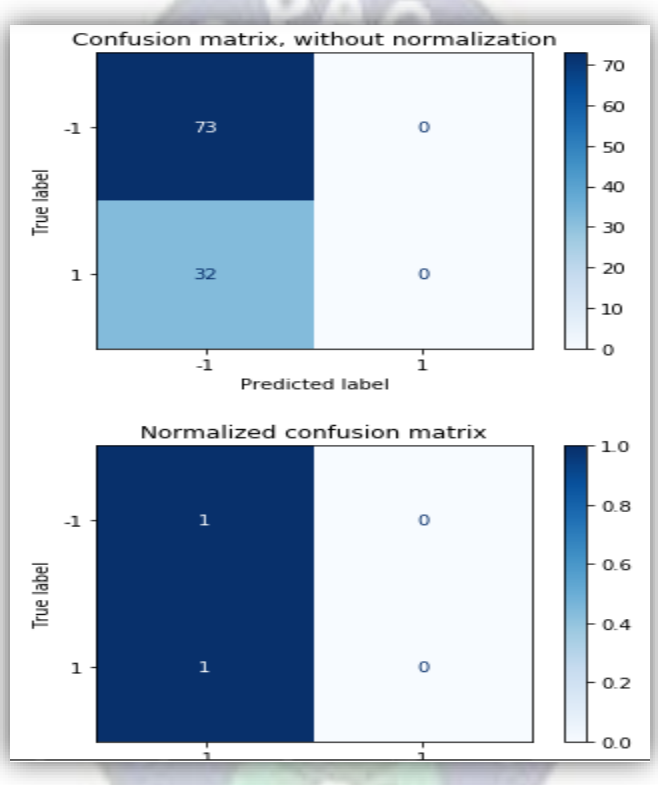


Figura 3.44. Gráfica de la matriz de confusión utilizando librerías, matriz de confusión sin normalización (arriba) y matriz de confusión con normalización (abajo).

```
FP:Falsos Positivos
0.502283105022831
VN:Verdaderos Negativos
0.2990867579908676
VP:Verdaderos Positivos
0.0821917808219178
FN:Falsos Negativos
0.07077625570776255
```

Figura 3.45. Average (Promedio) del modelo SVM One-Class de Falsos Positivos, Verdaderos Negativos, Verdaderos Positivos y Falsos Negativos.

El promedio de datos de la matriz solo ayuda a ver si es aceptable el modelo, en el siguiente Capítulo 4 se evaluara la matriz de confusión mas afondo. Por otro lado, los Falsos Positivos 0.5022831 (FP) sería el promedio que el algoritmo se confunde y dice que es una Anomalía cuando en realidad es un dato Normal. También se puede ver que los Verdaderos Positivos 0.08219178 (VP) sería el promedio que el algoritmo predice que son anomalías y resulta que también son de tipo Malware.

- **Isolation Forest.-** El modelo no cuenta con dos conjuntos de datos al igual que los anteriores modelos, de la misma manera se agarra el conjunto de datos y se divide en dos conjuntos de datos [X,y]; una parte será para evaluar la precisión de entrenamiento y otra parte será la precisión de evaluación.

```
# separ los datos en train y eval
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.35,train_size=0.65,random_state=42)

# creando el modelo sin control de profundidad, va a continuar hasta
# que todas las hojas sean puras
arbol = DecisionTreeClassifier(criterion='entropy')

# Ajustando el modelo
arbol.fit(x_train, y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Figura 3.46. Parámetros puestos para la evaluación del Isolation Forest.

El algoritmo Isolation Forest que se implemento en el modelo puede ser una herramienta precisa, pero también con mucha tendencia al sobreajuste al igual que en los anteriores modelos. Para construir este modelo, igual que los otros dos modelos se aplica un procedimiento recursivo para encontrar los atributos que proporcionara más información sobre distintos subconjuntos de datos, cada vez más pequeños. Si se aplica este procedimiento en forma reiterada, eventualmente se podrá llegar a un árbol en el que cada hoja tenga una sola instancia del objetivo a clasificar.

```
print("precisión entranamiento: {0: .2f}".format(arbol.score(x_train, y_train)))
precisión entranamiento:  0.99
```

Figura 3.47. Evaluación del modelo, precisión de entrenamiento al 99%.

Se evaluó en nivel de precisión de entrenamiento del modelo de Isolation Forest, tiene una precisión de entrenamiento del 99%; entonces se puede decir que el entrenamiento se comporto muy bien. Si se compara con el Robust Covariance que tuvo una precisión de entrenamiento del 99%, es decir que ambos tienen la misma precisión; pero si se compara con el SVM One-Class se puede decir que no es tan preciso porque le lleva por 5%; pero lo importante será como será su precisión de evaluación. Ahora se verá la precisión de evaluación del modelo.

```
# precisión del modelo en datos de evaluación.
print("precisión evaluación: {0: .2f}".format(arbol.score(x_test, y_test)))
precisión evaluación:  0.98
```

Figura 3.48. Evaluación del modelo, precisión de evaluación del modelo al 98%.

La precisión de evaluación del modelo es del 98%, lo que indica que el modelo Isolation Forest no es tan preciso pero quedo demasiado cerca de serlo, no todo modelo puede salir perfecto porque caería en el sobre ajuste, y si tuviera una precisión de evaluación muy baja también se diría que cae en el sobre ajuste. Si se hace una comparación con el anterior modelo de Robust Covariance se puede decir que Isolation Forest tiene un mejor precisión en la evaluación y también en precisión de entrenamiento es demasiado buena.

Por tanto el modelo tiene un muy buen nivel de precisión, ahora se graficara una matriz de confusión del modelo con la ayuda de las librerias de sklearn la misma librería que nos ofreció el algoritmo Isolation Forest. Al final de todas las evaluaciones de cada modelo se hara una tabla donde se podrá ver con cual se quedara para la última fase de despliegue.

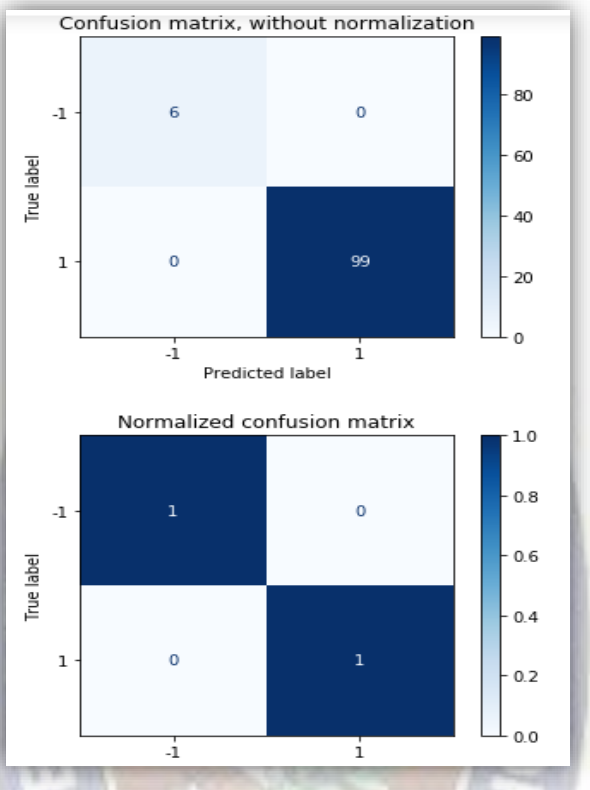


Figura 3.49. Gráfica de la matriz de confusión utilizando librerías, matriz de confusión sin normalización (arriba) y matriz de confusión con normalización (abajo).

```
FP:Falsos Positivos
0.01141552511415525
VN:Verdaderos Negativos
0.7899543378995434
VP:Verdaderos Positivos
0.0182648401826484
FN:Falsos Negativos
0.13470319634703196
```

Figura 3.50. Average (Promedio) del modelo Isolation Forest Falsos Positivos, Verdaderos Negativos, Verdaderos Positivos y Falsos Negativos.

El promedio de datos de la matriz solo ayudara a ver si es aceptable el modelo, en el siguiente Capítulo 4 se evaluara la matriz de confusión mas afondo. Por otro lado, los Falsos Positivos 0.0114155 (FP) sería el promedio que el algoritmo se confunde y dice que

es una Anomalía cuando en realidad es un dato Normal. También se puede ver que los Verdaderos Positivos 0.18264840 (VP) sería el promedio que el algoritmo predice que son anomalías y resulta que también son de tipo Malware.

Se termina con la evaluación de los modelos; el modelo **Robust Covariance** tiene un buen nivel de entrenamiento pero no tan preciso a la hora de la evaluación, el modelo **SVM One-Class** tiene un buen nivel de entrenamiento y de evaluación, pero no es tan bueno al comparación de los otros, y por ultimo tenemos el modelo de **Isolation Forest** que tiene muy bien su precisión de entrenamiento como también su precisión de evaluación.

3.7 Despliegue

En esta etapa en el proyecto, se ha construido uno o varios modelos que parecen alcanzar calidad suficiente desde la una perspectiva de análisis de datos. Antes de proceder al despliegue final del modelo, es importante evaluarlo a fondo y objetivos de negocio. Un objetivo clave es determinar si hay alguna cuestión importante de negocio que no haya sido considerada suficientemente. Al final de esta fase, se debería obtener una decisión sobre el modelo de los resultados del proceso de análisis de datos.

	Precisión de Entrenamiento	**Precisión de Evaluación**
Robust Covariance	99 %	96 %
SVM One-Class	94 %	92 %
Isolation Forest	99 %	98 %

Tabla 3.2. Evaluaciones de cada modelo. Desde su precisión de entrenamiento y de su precisión de evaluación.

Se tiene los resultados de todos los modelos, a simple vista se puede ver que Robust Covariance y Isolation Forest tienen la misma precisión de entrenamiento al 99 %, pero sus precisiones de evaluación son distintas. Por otro lado tenemos SVM One-Class que también tiene 94 % una buena precisión de entrenamiento, y su precisión de evaluación 92 %

también es buena respecto a su precisión de entrenamiento. En los modelos de predicción lo importante es su precisión en la evaluación no tiene que tener un porcentaje bajo a comparación de su entrenamiento, tiene que estar cerca de su precisión de entrenamiento.

Como hemos visto a lo largo de este trabajo, un outlier o anomalía es un punto que no se ajusta a los patrones de comportamiento del resto de puntos de la muestra. Con el fin de identificarlos se han estudiado cuatro métodos:

- Robust Covariance.
- Local Outlier Factor.
- Isolation Forests.
- One-class SVM.

Como hemos visto en el Fase 2, estos cuatro métodos están basados en modelos clásicos de aprendizaje automático. Específicamente hemos estudiado vecinos próximos que es la base de los dos primeros, Random Forest que está directamente relacionado con Isolation Forest y máquinas de vectores de soporte, ya que One-class SVM es una modificación de estas.

Tras el estudio de los diferentes métodos en el capítulo 3 se puede extraer ciertas conclusiones de cada uno de ellos. El método de Covarianza Robusta aunque nos permite encontrar una solución de forma rápida mediante la creación de diversos conjuntos iníciales, si el problema al que se enfrenta presenta varios clusters las soluciones que ofrece son menos satisfactorias que la del resto de métodos. Por otro lado, el método de Local Outlier Factor, aunque se basa en uno de los métodos más sencillos, presenta un gran potencial, ya que permite identificar outliers incluso cuando tenemos varios clusters de outliers de diferentes densidades; aunque para esto se debe ajustar muy bien el número de vecinos a tener en cuenta a la hora de detectar outliers. El método de Isolation Forests, al estar basado en Random Forest, nos permite crear varios árboles con poca correlación entre ellos que nos permite promediar una puntuación que nos indica cómo de anómalo es un punto. Además, como lo que se está utilizando son árboles, los resultados obtenidos son

fácilmente interpretables. Como contrapartida, al igual que en Random Forests, la gran cantidad de parámetros que presenta este modelo hace que la búsqueda de los valores óptimos se más costosa que en el resto de métodos. Por último, One-class SVM presenta las ventajas de las máquinas de vectores clásicas, como por ejemplo, la posibilidad de utilizar kernels para poder transformar los datos a dimensiones mayores evitando el coste asociado a ello. Esto, añadido a que tiene pocos parámetros, hace que sea un método rápido de hiperparametrizar y que ofrece buenos resultados.

Para comprobar la eficiencia de cada uno de estos métodos, en primer lugar se procedido a utilizarlos en dos problemas sintéticos. En estos experimentos hemos podido observar que la detección de anomalías no es algo sencillo, ya que, pequeñas modificaciones en los diferentes parámetros nos llevan a soluciones muy diferentes. Además, se puede observar que la calidad de la solución no viene dada únicamente por el modelo seleccionado sino que también afecta el problema en el que se aplica, como sucede en el caso del compuesto por 4 clusters. En éste se puede observar cómo el método de covarianza robusta no ofrece una solución aceptable para ninguna de las combinaciones de valores probadas mientras que para el problema sintético con 2 clusters sí que lo hacía. Estas razones llevan a pensar que el proceso de detección de anomalías precisa de una etapa de supervisión en la que se debe analizar si los resultados obtenidos son coherentes con los datos que estamos tratando.

Después de esto, se utilizo los diferentes métodos en conjuntos de datos reales, tanto de clasificación como de regresión. La primera situación a la que se ha tenido que enfrentar es la escasez de problemas cuyos datos están bien etiquetados como outliers e inliers, y la ausencia de una función score que permita evaluar los resultados de predicción. Para solucionar esto, se ha propuesto un nuevo estimador que permite concatenar un método de detección de anomalías con uno de predicción. De esta manera y bajo la idea de que con un conjunto libre de outliers las predicciones realizadas serán más fiables, se puede aplicar la detección de anomalías a cualquier problema. De esta manera, se utilizo el estimador propuesto con los diferentes métodos estudiados para detección de outilers junto con una

regresión logística en el caso de problemas de clasificación y un modelo Ridge en el caso de problemas de regresión. Haciendo esto se pudo comprobar cómo, en mayor o menor medida, se está consiguiendo una mejora en la predicción, especialmente cuando se aplica los modelos de Local Outlier Factor, One-class SVM y el sistema de votación. También se pudo ver mediante la representación en diagramas de cajas que los diferentes métodos son capaces de diferenciar las principales características de outliers e inliers. De todas maneras, se observo que, al igual que sucedía en los problemas sintéticos, los métodos funcionan mejor o peor según el problema en el que se apliquen, por lo que sigue existiendo la necesidad de supervisar los resultados obtenidos.

Capítulo IV

Análisis de Datos y Resultados

4.1 Introducción

En este capítulo se evalúa a los modelos, si bien se tenía en la fase 5 de la metodología de CRISP-DM se pudo apreciar la evaluación del modelo, pero lo que se hará en este capítulo será una evaluación más profunda. Teniendo en cuenta el cumplimento de los criterios de éxito del problema. Debe considerarse además, que la fiabilidad calculada para el modelo se aplica solamente para los datos sobre los que se realizo en el análisis. Es preciso revisar el proceso, con los resultados obtenidos, para luego repetir algún paso anterior, en la que posiblemente se cometido algún error. Considerar que se pueden emplear múltiples herramientas para la interpretación de los resultados.

4.2 Matrices de Confusión

Ahora se evaluara cada modelo de manera que se tenga su respectiva matriz de confusión, será diferente las matrices de cada modelo. La matriz de confusión es una de las métricas mas intuitivas y sencillas que se utiliza para encontrar la precisión y exactitud del modelo. Vamos a empezar con las evaluaciones.

4.2.1 Modelo de predicción Robust Covariance

Se evaluara al modelo de predicción Robust Covariance.

- **VP:** Cantidad de datos que se predijo como anomalías y son datos de tipo MALWARE.
- **VN:** Cantidad de datos que se predijo datos normales y son datos de tipo NORMAL.
- **FN:** Cantidad de datos que se predijo datos normales y son datos de tipo MALWARE.
- **FP:** Cantidad de datos que se predijo como anomalías y son datos de tipo NORMAL.

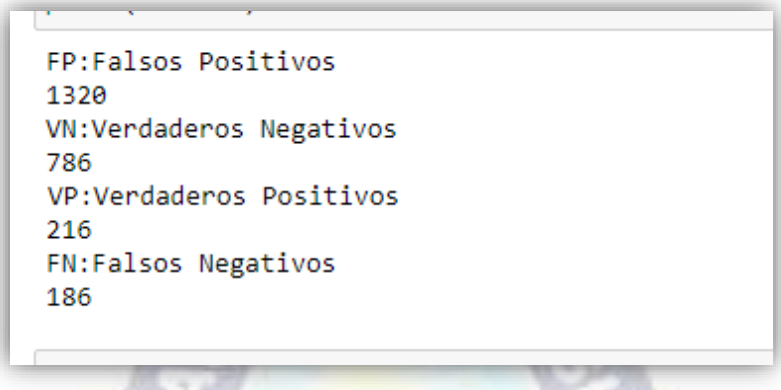


Figura 4.1. Datos obtenidos del Modelo Robust Covariance. Falso Positivos, Verdaderos Negativos, Verdaderos Positivos, Falsos Negativos.

Se puede apreciar los siguientes datos obtenidos por el modelo Robust Covarience, de acuerdo a estos datos se creara la matriz de confusión en base a la Tabla 2.1, y luego se analizara la matriz de confusión.

Matriz de Confusión		**Predicción**	
		Positivos	**Negativos**
Observación	**Positivos**	**Verdaderos Positivos 216**	**Falsos Negativos 186**
	Negativos	**Falsos Positivos 1320**	**Verdaderos Negativos 786**

Tabla 4.1. Matriz de confusión del modelo Robust Covariance.

Para entender la matriz vamos explicar cómo funciona. Primero los Verdaderos Negativos 786 (VN) sería el número de veces que el algoritmo ha acertado al decir que un dato se ve Normal y no es Malware. Por otro lado, los Falsos Positivos 1320 (FP) sería el número de veces que el algoritmo se confunde y dice que es una Anomalía cuando en realidad es un dato Normal. Falsos Negativos 186 (FN) serian los casos en que el algoritmo predice que es no es una anomalía, y resulta que el dato es de tipo Malware. Y por último, Verdaderos Positivos 216 (VP) serian los aciertos del algoritmo al predecir que son anomalías y resulta que también son de tipo Malware.

Basándonos en los valores de esta nueva matriz de confusión Tabla 4.1, más completa, se definirá una serie de métricas que serán muy útiles.

- **Exactitud (Accuary)**

$$\boldsymbol{Exactitud} = \frac{(VP + VN)}{(VP + FP + FN + VN)} \qquad (2.8.1)$$

$$Exactitud = \frac{(216 + 786)}{(216 + 1320 + 186 + 786)}$$

$$Exactitud = \frac{(1002)}{(2508)}$$

$$Exactitud = \ 0{,}399521$$

- **Precisión (Precision)**

$$\boldsymbol{Precisión} = \frac{VP}{(VP + FP)} \qquad (2.8.2)$$

$$Precisión = \frac{216}{(216 + 1320)}$$

$$Precisión = \frac{216}{(1536)}$$

$$Precisión = \ 0{,}140625$$

- **Sensibilidad (Recall)**

$$\boldsymbol{Sensibilidad} = \frac{VP}{(VP + FN)} \qquad (2.8.3)$$

$$Sensibilidad = \frac{216}{(216 + 186)}$$

$$Sensibilidad = \frac{216}{(402)}$$

$$Sensibilidad = \ 0{,}537313$$

- **Especificidad (Especificity)**

$$\boldsymbol{Especificidad} = \frac{VN}{(VN + FP)} \qquad (2.8.4)$$

$$Especificidad = \frac{786}{(786 + 1320)}$$

$$Especificidad = \frac{786}{(2106)}$$

$$Especificidad = 0{,}373219$$

4.2.2 Modelo de predicción SVM One-Class

Se evaluara al modelo de predicción SVM One-Class.

- **VP:** Cantidad de datos que se predijo como anomalías y son datos de tipo MALWARE.
- **VN:** Cantidad de datos que se predijo datos normales y son datos de tipo NORMAL.
- **FN:** Cantidad de datos que se predijo datos normales y son datos de tipo MALWARE.
- **FP:** Cantidad de datos que se predijo como anomalías y son datos de tipo NORMAL.

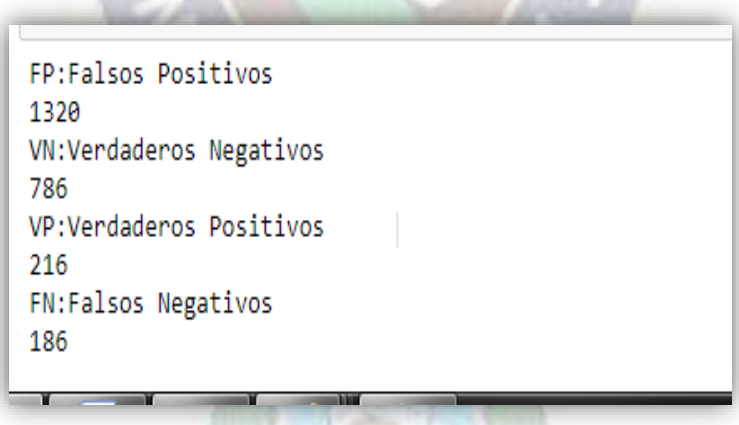


Figura 4.2. Datos obtenidos del Modelo SVM One-Class. Falso Positivos, Verdaderos Negativos, Verdaderos Positivos, Falsos Negativos.

Se puede apreciar los siguientes datos obtenidos por el modelo SVM One-Class, de acuerdo a estos datos se creara la matriz de confusión en base a la Tabla 2.1 y luego se analizara la matriz de confusión.

Matriz de Confusión		Predicción	
		Positivos	Negativos
Observación	Positivos	Verdaderos Positivos 216	Falsos Negativos 186
	Negativos	Falsos Positivos 1320	Verdaderos Negativos 786

Tabla 4.2. Matriz de confusión del modelo SVM One-Class.

Para entender la matriz vamos explicar cómo funciona. Primero los Verdaderos Negativos 786 (VN) sería el número de veces que el algoritmo ha acertado al decir que un dato se ve Normal y no es Malware. Por otro lado, los Falsos Positivos 1320 (FP) sería el número de veces que el algoritmo se confunde y dice que es una Anomalía cuando en realidad es un dato Normal. Falsos Negativos 186 (FN) serian los casos en que el algoritmo predice que es no es una anomalía, y resulta que el dato es de tipo Malware. Y por último, Verdaderos Positivos 216 (VP) serian los aciertos del algoritmo al predecir que son anomalías y resulta que también son de tipo Malware.

Basándonos en los valores de esta nueva matriz de confusión Tabla 4.2, más completa, se definirá una serie de métricas que serán muy útiles.

- **Exactitud (Accuary)**

$$\boldsymbol{Exactitud} = \frac{(VP + VN)}{(VP + FP + FN + VN)} \qquad (2.8.1)$$

$$Exactitud = \frac{(216 + 786)}{(216 + 1320 + 186 + 786)}$$

$$Exactitud = \frac{(1002)}{(2508)}$$

$$Exactitud = 0{,}399521$$

- **Precisión (Precision)**

$$\boldsymbol{Precisión} = \frac{VP}{(VP + FP)} \quad (2.8.2)$$

$$Precisión = \frac{216}{(216 + 1320)}$$

$$Precisión = \frac{216}{(1536)}$$

$$Precisión = 0{,}140625$$

- **Sensibilidad (Recall)**

$$\boldsymbol{Sensibilidad} = \frac{VP}{(VP + FN)} \quad (2.8.3)$$

$$Sensibilidad = \frac{216}{(216 + 186)}$$

$$Sensibilidad = \frac{216}{(402)}$$

$$Sensibilidad = 0{,}537313$$

- **Especificidad (Especificity)**

$$\boldsymbol{Especificidad} = \frac{VN}{(VN + FP)} \quad (2.8.4)$$

$$Especificidad = \frac{786}{(786 + 1320)}$$

$$Especificidad = \frac{786}{(2106)}$$

$$Especificidad = 0{,}373219$$

4.2.3 Modelo de predicción Isolation Forest

Se evaluara al modelo de predicción Isolation Forest.

- **VP:** Cantidad de datos que se predijo como anomalías y son datos de tipo MALWARE.
- **VN:** Cantidad de datos que se predijo datos normales y son datos de tipo NORMAL.

- **FN:** Cantidad de datos que se predijo datos normales y son datos de tipo MALWARE.
- **FP:** Cantidad de datos que se predijo como anomalías y son datos de tipo NORMAL.

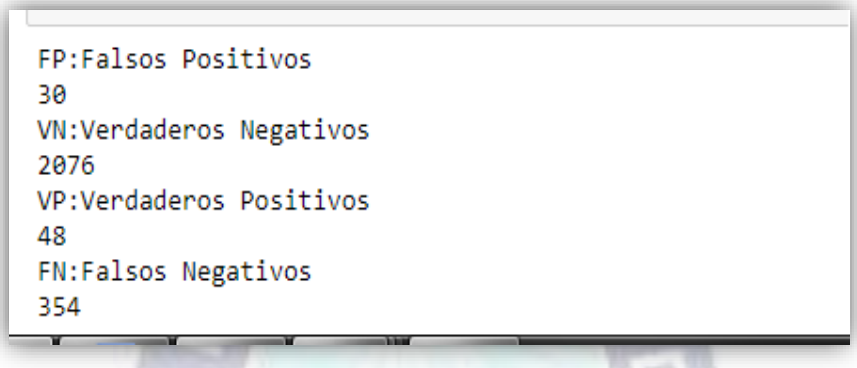


Figura 4.3. Datos obtenidos del Modelo Isolation Forest. Falso Positivos, Verdaderos Negativos, Verdaderos Positivos, Falsos Negativos.

Se puede apreciar los siguientes datos obtenidos por el modelo Isolation Forest, de acuerdo a estos datos se creara nuestra matriz de confusión en base a la Tabla 2.1, y luego se analizara la matriz de confusión.

Matriz de Confusión		**Predicción**	
		Positivos	**Negativos**
Observación	**Positivos**	**Verdaderos Positivos 48**	**Falsos Negativos 354**
	Negativos	**Falsos Positivos 30**	**Verdaderos Negativos 2076**

Tabla 4.3. Matriz de confusión del modelo Isolation Forest.

Para entender la matriz vamos explicar cómo funciona. Primero los Verdaderos Negativos 2076 (VN) sería el número de veces que el algoritmo ha acertado al decir que un dato se ve Normal y no es Malware. Por otro lado, los Falsos Positivos 30 (FP) sería el número de veces que el algoritmo se confunde y dice que es una Anomalía cuando en realidad es un dato Normal. Falsos Negativos 354 (FN) serian los casos en que el algoritmo predice que es no es una anomalía, y resulta que el dato es de tipo Malware. Y por último, Verdaderos

Positivos 48 (VP) serian los aciertos del algoritmo al predecir que son anomalías y resulta que también son de tipo Malware.

Basándonos en los valores de esta nueva matriz de confusión Tabla 4.3, más completa, se definirá una serie de métricas que serán muy útiles.

- **Exactitud (Accuary)**

$$\boldsymbol{Exactitud} = \frac{(VP + VN)}{(VP + FP + FN + VN)} \quad (2.8.1)$$

$$Exactitud = \frac{(48 + 2076)}{(48 + 30 + 354 + 2076)}$$

$$Exactitud = \frac{(2124)}{(2508)}$$

$$Exactitud = 0{,}846889$$

- **Precisión (Precision)**

$$\boldsymbol{Precisión} = \frac{VP}{(VP + FP)} \quad (2.8.2)$$

$$Precisión = \frac{48}{(48 + 30)}$$

$$Precisión = \frac{48}{(78)}$$

$$Precisión = 0{,}615384$$

- **Sensibilidad (Recall)**

$$\boldsymbol{Sensibilidad} = \frac{VP}{(VP + FN)} \quad (2.8.3)$$

$$Sensibilidad = \frac{48}{(48 + 354)}$$

$$Sensibilidad = \frac{48}{(402)}$$

$$Sensibilidad = 0{,}119402$$

- **Especificidad (Especificity)**

$$\boldsymbol{Especificidad} = \frac{VN}{(VN + FP)} \qquad (2.8.4)$$

$$Especificidad = \frac{2076}{(2076 + 30)}$$

$$Especificidad = \frac{2076}{(2106)}$$

$$Especificidad = 0{,}985754$$

4.3 Elección del Mejor Modelo de Predicción

Luego de haber obtenido de cada modelo de predicción su matriz de confusión y la serie de métricas, se crea la siguiente tabla

	Exactitud	Precisión	Sensibilidad	Especificidad
Robust Covariance	**40%**	**14%**	**54%**	**37%**
SVM One Class	**40%**	**14%**	**54%**	**37%**
Isolation Forest	**85%**	**62%**	**12%**	**98%**

Tabla 4.4. Métricas de cada modelo de predicción, basado en la matriz de confusión.

Ahora una vez obtenidos estas métricas se podrá elegir al mejor modelo de predicción empleando la curva de ROC y luego ver si en el área bajo la curva AUC satisface nuestra hipótesis que declaramos antes de iniciar todo el estudio.

4.3.1 Curvas de ROC y Areas bajo la curva (AUC)

Primero se tiene el modelo **Robust Covariance**, en la siguiente Figura 4.4 se puede apreciar en el lado **izquierdo** la curva de color verde que representa a todos los datos **no anómalos**, mientras que la curva de color rojo representa los datos que son **malware**.

Ahora se utiliza las métricas de sensibilidad y especificidad. Recordando que la sensibilidad es la porción de datos que se identificaron correctamente por ser malware, es decir verdadero positivo, sobre el número total de datos que realmente son malware. Por otra parte la especificidad es la proporción de datos que se identificaron correctamente por no ser malware, verdadero negativo, sobre el número total de no tener malware.

Ahora en la misma Figura 4.4 lado **derecho** se puede apreciar la curva ROC, pero para trazar esta curva en lugar de especificidad usamos **1-especificidad** y por lo que la curva ROC está definida por la sensibilidad que es la tasa de verdadero positivo y **1-especificiadad** es la tasa de falso positivo. También se puede apreciar el **AUC=0.76** lo que significa que hay **76%** de probabilidad de que el modelo pueda distinguir entre clase positiva y clase negativa.

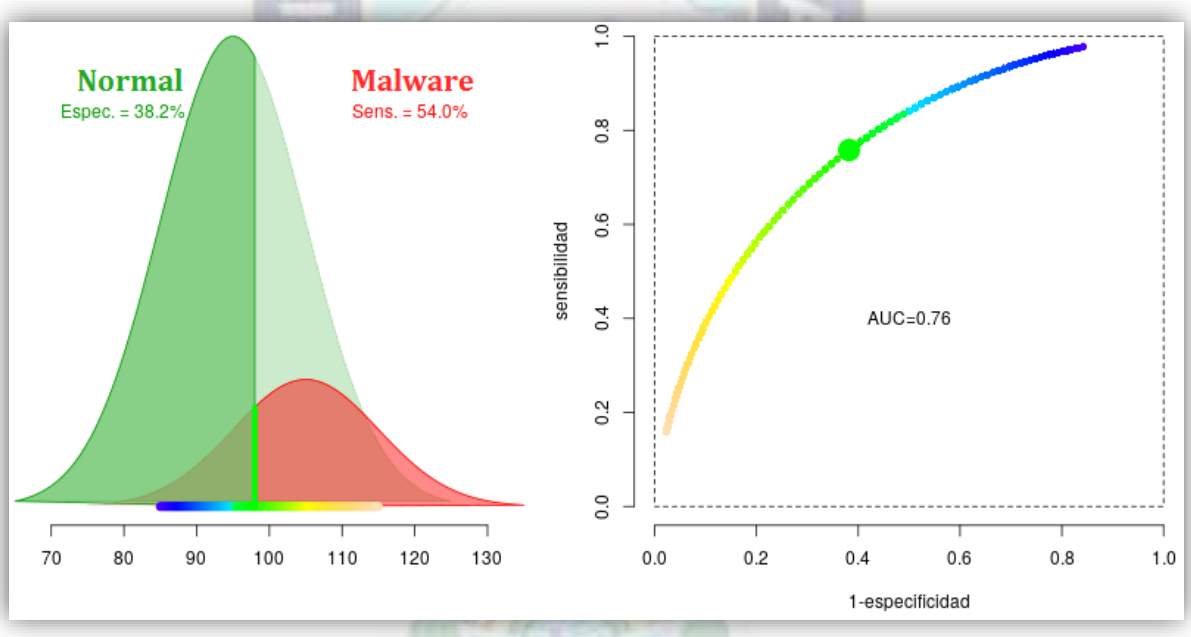


Figura 4.4. Curvas ROC del Robust Covariance (izquierdo) y área bajo la Curva del Robust Covariance (derecho)

Como segundo se tiene el modelo **SVM One-Class**, en la siguiente Figura 4.5 se puede apreciar en el lado **izquierdo** la curva de color verde que representa a todos los datos **no anómalos**, mientras que la curva de color rojo representa los datos que son **malware**.

Ahora se utiliza las métricas de sensibilidad y especificidad. Recordando que la sensibilidad es la porción de datos que se identificaron correctamente por ser malware, es decir verdadero positivo, sobre el número total de datos que realmente son malware. Por otra parte la especificidad es la proporción de datos que se identificaron correctamente por no ser malware, verdadero negativo, sobre el número total de no tener malware.

En la misma Figura 4.5 lado **derecho** se puede apreciar la curva ROC, pero para trazar esta curva en lugar de especificidad usamos **1-especificidad** y por lo que la curva ROC está definida por la sensibilidad que es la tasa de verdadero positivo y **1-especificiadad** es la tasa de falso positivo. También se puede apreciar el **AUC=0.76** lo que significa que hay **76%** de probabilidad de que el modelo pueda distinguir entre clase positiva y clase negativa.

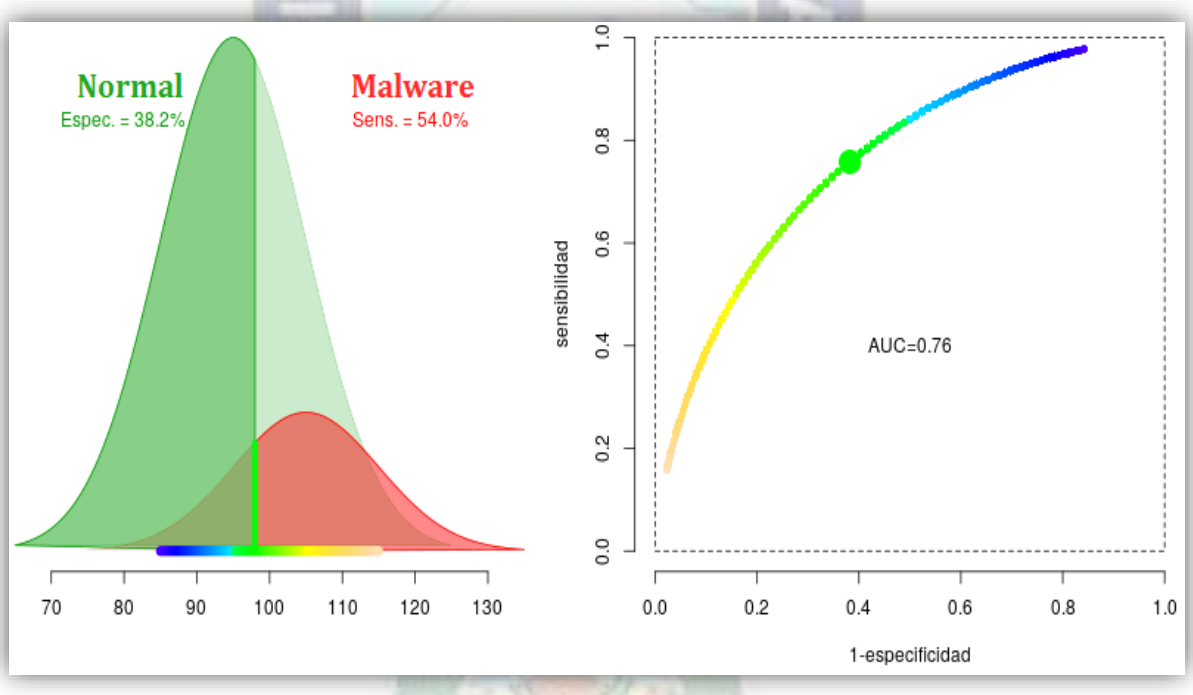


Figura 4.5. Curvas ROC del SVM One-Class (izquierdo) y área bajo la Curva del SVM One-Class (derecho).

Y finalmente se tiene el modelo **Isolation Forest,** en la siguiente Figura 4.6 se puede apreciar en el lado **izquierdo** la curva de color verde que representa a todos los datos **no**

anómalos, mientras que la curva de color rojo representa los datos que son **malware**. Ahora se utiliza las métricas de sensibilidad y especificidad. Recordando que la sensibilidad es la porción de datos que se identificaron correctamente por ser malware, es decir verdadero positivo, sobre el número total de datos que realmente son malware. Por otra parte la especificidad es la proporción de datos que se identificaron correctamente por no ser malware, verdadero negativo, sobre el número total de no tener malware.

En la misma Figura 4.6 lado **derecho** se puede apreciar la curva ROC, pero para trazar esta curva en lugar de especificidad usamos **1-especificidad** y por lo que la curva ROC está definida por la sensibilidad que es la tasa de verdadero positivo y **1-especificiadad** es la tasa de falso positivo. También se puede apreciar el **AUC=0.92** lo que significa que hay **92%** de probabilidad de que el modelo pueda distinguir entre clase positiva y clase negativa.

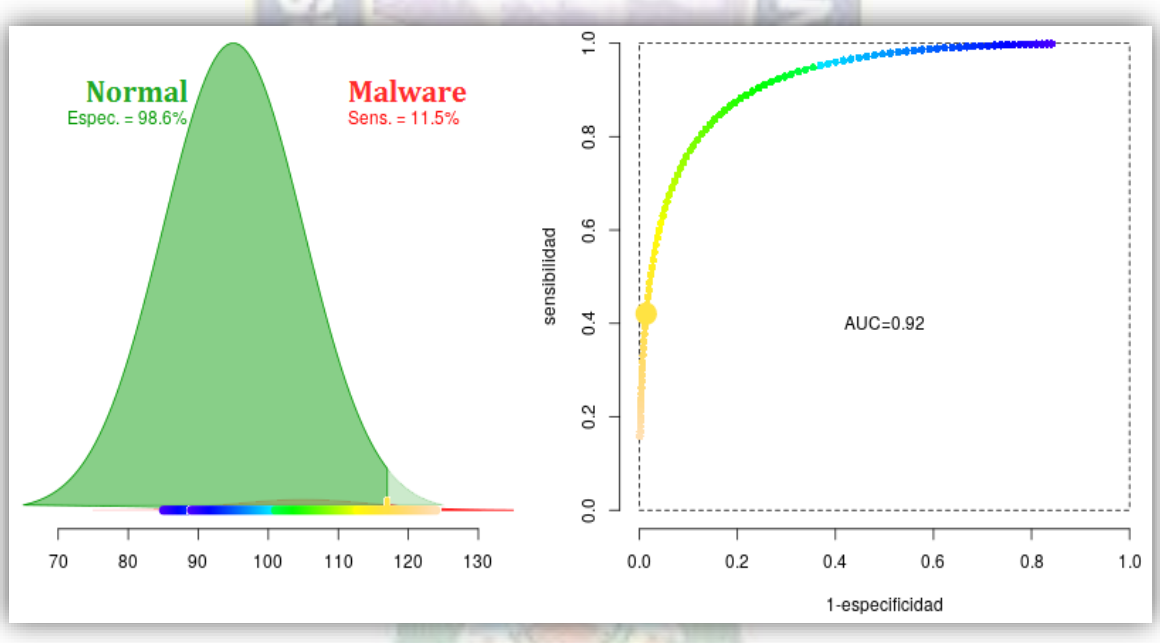


Figura 4.6. Curvas ROC del Isolation Forest (izquierdo) y área bajo la Curva del Isolation Forest (derecho).

De acuerdo con la siguiente Tabla 4.5 se podrá interpretar el área bajo la curva (AUC) de los modelos que fueron evaluados.

Área Bajo la Curva (AUC)	
0.90 – 1	Excelente
0.80 – 0.90	Buena
0.70 – 0.80	Justo
0.60 – 0.70	Pobre
0.50 – 0.60	Falla

Tabla 4.5. Interpretación de el Área Bajo la Curva.

Primero se evaluó el modelo **Robust Covariance** que tiene una probabilidad del **76%** de distinguir de datos no anómalos y datos de malware. Segundo el modelo **SVM One-Class** que tiene también una probabilidad del **76%** de distinguir de datos no anómalos y datos de malware. Y por último se evaluó el modelo **Isolation Forest** con una probabilidad del **92%** de distinguir datos no anómalos y datos de malware. Basado en la Tabla 4.5 la interpretación de cada modelo se muestra en la siguiente Tabla 4.6.

Probabilidades de (AUC)		
Probabilidad de distinguir de datos no anómalos y datos que son Malware		
Robust Covariance	**76%**	**Justo**
SVM One-Class	**76%**	**Justo**
Isolation Forest	**92%**	**Excelente**

Tabla 4.6. Probabilidades de los modelos evaluados.

Finalmente por la Tabla 4.6 se deduce que el **mejor modelo de predicción** es el **Isolation Forest** con una probabilidad del **92% excelente**; de que el modelo pueda distinguir entre datos no anómalos y datos que son malware. La hipótesis que en un principio se supuso al 90% de confiabilidad, con el fin de iniciar el modelo de predicción, se pudo satisfacer con **éxito**.

Capitulo V

Conclusiones y Recomendaciones

5.1 Conclusiones

- En el presente tesis se ha podido evidenciar que el crecimiento de los ataques informáticos sigue avanzando constantemente, así como la gran cantidad de herramientas de detección de vulnerabilidades y de protección que existen, por lo tanto, no realizar la protección de un sistema informático permite el fácil acceso de intrusos al sistema poniendo en riesgo la confidencialidad, integridad y disponibilidad de la información, además de aumentar los costos por realizar mantenimiento correctivo a los dispositivos dañados.
- Por otro lado, a través del estudio de flujo del tráfico de una red que se observo gracias a la herramienta de WireShark, se pudo observar como la red es atacada por malware, el cual dio pauta a la elaboración propuesta de diseñar un modelo de detección de malware en una red.
- Para obtener el Dataset se tuvo que reenviar todas las estadísticas de la red a un host, donde este host estaba conectado un Port Mirror. Una vez obtenido estas estadísticas se capturo gracias al protocolo Netflow, para así poder tener un mejor manejo de todo el flujo del tráfico capturado.
- Se profundizo el uso de algoritmos de Machine Learning para la detección de anomalías en un conjunto de datos reales. Donde algoritmos como Robust Covariance, SVM One-Class y Isolation Forest influyeron bastante en la detección de malware.
- Antes de iniciar el modelo se propuso un 90% de confiabilidad como hipótesis de detectar el malware en una red con algoritmos de Machine Learning. Una vez terminado las evaluaciones de todos los modelos, se concluye que el modelo Isolation Forest con una probabilidad del 92% de distinguir datos no anómalos y datos que son malware, satisface y sobrepasa la hipótesis planteada.

- Finalmente se han cumplido los objetivos y se ha comprendido la utilidad de la estadística y la inteligencia artificial, en un campo donde no se creía que podría funcionar. Además de ser dificultoso y el reto que se propuso en lograr detectar malware en un flujo de trafico de una red, fue la motivación necesaria para encarar el modelo.

5.2 Recomendaciones

- Mejorar la seguridad informática de una red, mediante el uso de los métodos de la propuesta de detección de ataques como la actualización del sistema operativo, software y el navegador, además del uso de antivirus así como herramientas anti malware.
- Los administradores de las redes informáticas deben estar en una constante actualización en el área de seguridad informática, de este modo mejorarían sus acciones de respuesta ante las inminentes amenazas que se están generando, es necesario enfatizar que la seguridad de la información es la parte central de cualquier tipo de red, por lo que siempre se deben realizar búsqueda de las posibles vulnerabilidades que pudiera presentar la red.
- Tener un mejor tratamiento de los datos con el fin de tener un Dataset más limpio, para tener una mejor evaluación y predicción de los datos a la hora de implementar los algoritmos de Machine Learning.
- Seguir los procesos de la metodología que se implementa, ya que será pieza fundamental para orientarse sobre minería de datos. La secuencia de las fases te permitirá avanzar y retroceder las veces que sean necesarias con el fin de obtener mejores resultados.

Bibliografía

Analytics. 2018. ¿CUÁL ES LA DIFERENCIA ENTRE INTELIGENCIA ARTIFICIAL (AI) Y MACHINE LEARNING (ML)? [en línea] <https://www.analytics10.com/blog/cual-es-la-diferencia-entre-inteligencia-artificial-ai-y-machine-learning-ml/> [Consulta: 7 de agosto 2019].

Aukera Mastering Data. 2018. Data Science ¿Que es y que no es? [en línea] <https://aukera.es/blog/data-science-que-es-y-que-no-es/> [Consulta: 7 de mayo 2019].

Apd. 2019. Data Science ¿Qué es Machine Learning y cómo funciona? [en línea] <https://www.apd.es/que-es-machine-learning/> [Consulta: 18 de mayo 2019].

Breunig Markus M. 2000. Hans-Peter Kriegel Raymond T. Ng, and Jörg Sander. LOF: identifying density- based local outliers. ACM SIGMOD Record, 29(2):93_104.

Chenaru, O.D.. 2016. Practical fault management using real-time decision tree analysis. 24th Mediterranean Conference on Control and Automation (MED).

Chu, Q. Z. A. T. 2018. "Structure regularized traffic monitoring model for traffic matrix estimation and anomaly detection. Control Conference (CCC). Chinese, Hangzhou**:** 4980-4985.

Cisco. 2019. CCNA 1 Introduccion a Networks [en línea] <www.cisco.net/> [Consulta: 22 de agosto 2019].

EcuRed. Matrices de confusión [en linea] <https://www.ecured.cu/Matrices_de_confusi%C3%B3n> [Consultado: 31 de mayo 2019].

Eset. 2012. ¿Qué es y cómo funciona una VPN para la privacidad de la información? [en linea] <https://www.welivesecurity.com/la-es/2012/09/10/vpn-funcionamiento-privacidad-informacion/> [Consultado: 16 de mayo 2019].

Fractalia. 2015. Data Mining: qué es y para qué sirve [en linea] <https://fractaliasystems.com/2015/06/25/data-mining-que-es-y-para-que-sirve-2/> [Consultado: 24 de junio 2019].

Gómez Vieites Álvaro. 2011. Enciclopedia de la Seguridad Informática 2da Edición Actualizada, Madrid, RA-MA.

Gonzalez Ligdi. 2019. Aprende todo sobre inteligencia artificial [en linea] <https://ligdigonzalez.com/curvas-roc-y-area-bajo-la-curva-auc-machine-learning/> [Consultado: 31 de mayo 2019].

Hassett B. 2016. Recovery from multiple faults in a communications network US 9331897 B2**.**

IBM, 2012. IBM SPSS MODELER 15, Edicion 15

ISO. 2017. La norma ISO 27001 del Sistema de Gestión de Garantía de confidencialidad, integridad y disponibilidad [en linea] <https://www.aec.es/c/document_library/get_file?uuid=a89e72de-d92b-47cfba5e-5ea421fcbeb4&groupId=10128> [Consultado: 28 de mayo 2019].

Kaspersky. 2019. ¿Qué es el ransomware? [en linea] <https://latam.kaspersky.com/resource-center/definitions/what-is-ransomware> [Consultado: 22 de Agosto 2019].

Kleinsteuber, H. K. W. K. M. 2016. "Network Volume Anomaly Detection and Identification in Large-scale Networks based on Online Time-structured Traffic Tensor Tracking." IEEE Transactions on Network and Service Management PP (99): 1.

Licona Arturo. 2014. Data Mining Deloitte Consulting Group

Liu Fei Tony. 2008. And Zhi-Hua Zhou. Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining. IEEE.

Martinez Mario. 2018. INTELIGENCIA ARTIFICIAL, MACHINE LEARNING Y DEEP LEARNING [en linea] <https://www.doctormetrics.com/inteligencia-artificial/> [Consultado: 22 de Agosto 2019].

Matsumoto, A., et al. 2016. Fault management system. Fault management server, and non-transitory computer-readable storage medium in which fault management program is stored.

Mira Alfaro Emilio José. 2002. Sistemas de Detección de Intrusos [en línea] <http://mural.uv.es/emial/informatica/html/IDS.html> [Consulta: 10 de mayo 2019].

Moes Tibor. 2019. ¿Qué es malware? La definición y los 5 tipos principales [en línea] <https://softwarelab.org/es/que-es-malware/> [Consulta: 22 de septiembre 2019].

Mohiuddin, A. N. M., JIANKUN HU. 2016. "A survey of network anomaly detection techniques." Journal of Network and Computer Applications 60: 19–31.

Molina LC. 2002. Data Mining: torturando los datos hasta que confiesen.

Nautiyal Dewang, 2019 Underfitting y Overfitting en Machine Learning [en línea] <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/> [Consulta: 15 de Julio 2019].

Neoland. 2019. ¿Qué es Data Science? [en línea] <https://www.neoland.es/blog/que-es-data-science> [Consulta: 12 marzo 2019].

Raffino María Estela. 2018. Concepto de Malware [en línea] <https://concepto.de/malware/> [Consulta: 10 de mayo 2019].

Rascagneres Paul. 2016. Seguridad informática y Malwares, Europa, ENI, Colección Epsilon.

Roa Buendía Jose Fabián. 2013. Seguridad Informática, Aravaca, Madrid: McGraw-Hill.

Rousseeuw Peter J. and Driessen Katrien Van. 1999. A fast algorithm for the minimum covariance determinant estimator. Technometrics, 41(3):212_223.

Simpl¡learn. 2019. Data Sciencie vs Big Data vs. Data Analytics [en línea] <https://www.simplilearn.com/data-science-vs-big-data-vs-data-analytics-article> [Consulta: 11 de mayo 2019].

Singular Data & Analytics. 2016. CRISP-DM: La metodología para poner orden en los proyectos de Data Science [en linea] < https://data.sngular.com/es/art/25/crisp-dm-la-metodologia-para-poner-orden-en-los-proyectos-de-data-science> [Consultado: 20 de mayo 2019].

Souza, V. S., SE. 2016. Method and Arrangement for fault Management in infrastructure as a Service clouds Telefonaktiebolaget L M Ericsson (publ) (Stockholm, SE).

Schölkopf Bernhard. 2001. John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. Neural Computation,13(7):1443_1471.

Tk Keanini. 2019. El estado del aprendizaje automático en 2019 [en línea] <https://blogs.cisco.com/security/the-state-of-machine-learning-in-2019> [Consulta: 2 de mayo 2019].

Torrano Carmen. 2017. Netflow, machine leraning y la detección de anomalías en una red [en línea] <https://empresas.blogthinkbig.com/netflow-machine-learning-y-la-deteccion/> [Consultado: 15 de mayo 2019].

Wireshark. 2019. Wireshark [en línea] <https://www.wireshark.org/> [Consultado: 21 de mayo 2019].

ANEXOS

ANEXO A
Preparación de los Datos

```
import os
import csv
import pandas as pd
import matplotlib.pyplot as plt

os.chdir('C:/Users/FER/Desktop/FINAL')
data_muestra = pd.read_csv('muestra2.csv')

os.chdir('C:/Users/FER/Desktop/FINAL')
data_malware= pd.read_csv('malware_ramsonware.csv')
```

```
with open('muestra2.csv') as File:#abriendo el fichero pero sin aplicar el close ocierre
    reader = csv.reader(File, delimiter=',', quotechar=',', quoting=csv.QUOTE_MINIMAL)
    for row in reader:#de la lista reader sacamos una varible para luego mostrarla con print
        print(row)
```

```
['"No."', '"Time"', '"Source"', '"Destination"', '"Protocol"', '"Length"', '"Info"']
['"1"', '"2019-09-12 00:30:43"', '"192.168.1.102"', '"104.244.42.69"', '"TCP"', '"55"', '"49666  >  443 [ACK] Seq=1 Ack=1 Win
=253 Len=1 [TCP segment of a reassembled PDU]"']
['"2"', '"2019-09-12 00:30:43"', '"104.244.42.69"', '"192.168.1.102"', '"TCP"', '"66"', '"443  >  49666 [ACK] Seq=1 Ack=2 Win
=123 Len=0 SLE=1 SRE=2"']
['"3"', '"2019-09-12 00:30:48"', '"192.168.1.102"', '"104.244.42.67"', '"TCP"', '"55"', '"49682  >  443 [ACK] Seq=1 Ack=1 Win
=252 Len=1 [TCP segment of a reassembled PDU]"']
['"4"', '"2019-09-12 00:30:49"', '"104.244.42.67"', '"192.168.1.102"', '"TCP"', '"66"', '"443  >  49682 [ACK] Seq=1 Ack=2 Win
=124 Len=0 SLE=1 SRE=2"']
['"5"', '"2019-09-12 00:30:56"', '"192.168.1.102"', '"192.168.1.1"', '"DNS"', '"74"', '"Standard query 0x9b9b A www.google.co
m"']
['"6"', '"2019-09-12 00:30:56"', '"192.168.1.102"', '"192.168.1.1"', '"DNS"', '"80"', '"Standard query 0x9cc7 A translate.goo
gle.com"']
```

```
data_mix= data_malware.merge(data_muestra, how='outer')
pd.options.display.max_rows = None
data_mix
```

	Time	Source	Destination	Protocol	Length	Info
0	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	TCP	66	49193 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=14...
1	2017-01-09 19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [SYN, ACK] Seq=0 Ack=1 Win=64240 ...
2	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	TCP	60	49193 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
3	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	HTTP	540	GET / HTTP/1.1
4	2017-01-09 19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [ACK] Seq=1 Ack=487 Win=64240 Len=0
5	2017-01-09 19:20:01	192.185.225.245	10.1.9.103	TCP	582	80 > 49193 [PSH, ACK] Seq=1 Ack=487 Win=6424...
6	2017-01-09 19:20:01	192.185.225.245	10.1.9.103	TCP	322	80 > 49193 [PSH, ACK] Seq=529 Ack=487 Win=64...
7	2017-01-09 19:20:01	10.1.9.103	192.185.225.245	TCP	60	49193 > 80 [ACK] Seq=487 Ack=797 Win=63444 L...
8	2017-01-09 19:20:01	10.1.9.103	194.87.94.227	TCP	66	49194 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=14...
9	2017-01-09 19:20:02	194.87.94.227	10.1.9.103	TCP	60	80 > 49194 [SYN, ACK] Seq=0 Ack=1 Win=64240 ...
10	2017-01-09 19:20:02	10.1.9.103	194.87.94.227	TCP	60	49194 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0

```
pd.options.display.max_rows = None
```

```
    #    k=k-1
etiqueta.iloc[i,6]='NORMAL'
    #etiqueta = etiqueta.iloc[i]['type'] = "etiq"
#print(pres)
#print(des)
etiqueta
#data_group1['prediction']= prediction
```

	Time	Source	Destination	Protocol	Length	Info	Tipo
0	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	TCP	66	49193 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=14...	NORMAL
1	2017-01-09 19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [SYN, ACK] Seq=0 Ack=1 Win=64240 ...	NORMAL
2	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	TCP	60	49193 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0	NORMAL
3	2017-01-09 19:19:59	10.1.9.103	192.185.225.245	HTTP	540	GET / HTTP/1.1	NORMAL
4	2017-01-09 19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [ACK] Seq=1 Ack=487 Win=64240 Len=0	NORMAL
5	2017-01-09 19:20:01	192.185.225.245	10.1.9.103	TCP	582	80 > 49193 [PSH, ACK] Seq=1 Ack=487 Win=6424...	NORMAL
6	2017-01-09 19:20:01	192.185.225.245	10.1.9.103	TCP	322	80 > 49193 [PSH, ACK] Seq=529 Ack=487 Win=64...	NORMAL
7	2017-01-09 19:20:01	10.1.9.103	192.185.225.245	TCP	60	49193 > 80 [ACK] Seq=487 Ack=797 Win=63444 L...	NORMAL
8	2017-01-09 19:20:01	10.1.9.103	194.87.94.227	TCP	66	49194 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=14...	NORMAL
9	2017-01-09 19:20:02	194.87.94.227	10.1.9.103	TCP	60	80 > 49194 [SYN, ACK] Seq=0 Ack=1 Win=64240 ...	NORMAL
10	2017-01-09 19:20:02	10.1.9.103	194.87.94.227	TCP	60	49194 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0	NORMAL

```
#das[['fecha', 'hora']] = tim.str.split(' ', n=0, expand=True)

#data_s = pd.DataFrame(tim.str(' ',1).tolist(), columns = [['flips','row']])
#data_tiempo
data_seg
data_union = pd.merge(data_seg, data_info, left_index=True, right_index=True)#el reset_index() no es necesario

data_union
```

	0	Fecha	Hora	Source	Destination	Protocol	Length	Info	Tipo
0	2017-01-09 19:19:59	2017-01-09	19:19:59	10.1.9.103	192.185.225.245	TCP	66	49193 > 80 [SYN] Seq=0 Win=8192 Len=0 MSS=14...	NORMAL
1	2017-01-09 19:19:59	2017-01-09	19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [SYN, ACK] Seq=0 Ack=1 Win=64240 ...	NORMAL
2	2017-01-09 19:19:59	2017-01-09	19:19:59	10.1.9.103	192.185.225.245	TCP	60	49193 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0	NORMAL
3	2017-01-09 19:19:59	2017-01-09	19:19:59	10.1.9.103	192.185.225.245	HTTP	540	GET / HTTP/1.1	NORMAL
4	2017-01-09 19:19:59	2017-01-09	19:19:59	192.185.225.245	10.1.9.103	TCP	60	80 > 49193 [ACK] Seq=1 Ack=487 Win=64240 Len=0	NORMAL
5	2017-01-09 19:20:01	2017-01-09	19:20:01	192.185.225.245	10.1.9.103	TCP	582	80 > 49193 [PSH, ACK] Seq=1 Ack=487 Win=6424...	NORMAL
6	2017-01-09 19:20:01	2017-01-09	19:20:01	192.185.225.245	10.1.9.103	TCP	322	80 > 49193 [PSH, ACK] Seq=529 Ack=487 Win=64...	NORMAL

```
i = 0
j = 1
k = 0
#dif = ' '
data_mejorado = pd.DataFrame(columns=['Fecha','Hora','Origen','Destino','Protocolo','Puerto','Info','Tipo'])
fin=numero_filas-1
while i < fin:
    uno = datetime.strptime(vectores[i][1], "%H:%M:%S")
    dos = datetime.strptime(vectores[j][1], "%H:%M:%S")
    if dos > uno:
        dif = diferencia_hora(vectores[j][1],vectores[i][1])
        seg = horas_segundos(dif)
        while seg > k:
            #restando_segundos(dif,'00:00:01')
            #inc = incrementando_segundos(vectores[i][1],'00:00:01')
            data_mejorado = data_mejorado.append({
            'Fecha' :  vectores[i][0],
            'Hora' : incrementando_segundos(vectores[i][1],seg_hora2(k)),
            'Origen' :  vectores[i][2],
            'Destino' : vectores[i][3],
            'Protocolo' :  vectores[i][4],
            'Puerto' : vectores[i][5],
            'Info' : vectores[i][6],
            'Tipo' : vectores[i][7]}, ignore_index=True)
            k+=1
            #print("e")
    elif dos == uno:
        data_mejorado = data_mejorado.append({
        'Fecha' :  vectores[i][0],
        'Hora' : vectores[i][1],
```

```
            'Tipo' : vectores[i][7]}, ignore_index=True)
            k+=1
            #print("e")
    elif dos == uno:
        data_mejorado = data_mejorado.append({
        'Fecha' :  vectores[i][0],
        'Hora' : vectores[i][1],
        'Origen' :  vectores[i][2],
        'Destino' : vectores[i][3],
        'Protocolo' :  vectores[i][4],
        'Puerto' : vectores[i][5],
        'Info' : vectores[i][6],
        'Tipo' : vectores[i][7]}, ignore_index=True)
    else:
        print('a')
    k = 0
    i += 1
    j += 1
data_mejorado = data_mejorado.append({
        'Fecha' :  vectores[fin][0],
        'Hora' : vectores[fin][1],
        'Origen' :  vectores[fin][2],
        'Destino' : vectores[fin][3],
        'Protocolo' :  vectores[fin][4],
        'Puerto' : vectores[fin][5],
        'Info' : vectores[fin][6],
        'Tipo' : vectores[fin][7]}, ignore_index=True)
#else:
#print("esto es fin del whille")
data_mejorado
```

data_mejorado

3338	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=219652 Ack=2030 Win=32... NORMAL
3339	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TLSv1.2	372	Application Data NORMAL
3340	2019-09-12	00:31:40	104.24.121.24	192.168.1.102	TCP	54	443 > 49690 [ACK] Seq=4435 Ack=751 Win=31744... NORMAL
3341	2019-09-12	00:31:40	192.168.1.102	104.18.39.165	TCP	54	49685 > 443 [ACK] Seq=2210 Ack=221468 Win=86... NORMAL
3342	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=221468 Ack=2210 Win=33... NORMAL
3343	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=222928 Ack=2210 Win=33... NORMAL
3344	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=224388 Ack=2210 Win=33... NORMAL
3345	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=225848 Ack=2210 Win=33... NORMAL
3346	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=227308 Ack=2210 Win=33... NORMAL
3347	2019-09-12	00:31:40	192.168.1.102	104.18.39.165	TCP	54	49685 > 443 [ACK] Seq=2210 Ack=228768 Win=86... NORMAL
3348	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=228768 Ack=2210 Win=33... NORMAL
3349	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=230228 Ack=2210 Win=33... NORMAL
3350	2019-09-12	00:31:40	104.18.39.165	192.168.1.102	TCP	1514	443 > 49685 [ACK] Seq=231688 Ack=2210 Win=33... NORMAL

```
#vamos agrupar por fechas horas detino y protocño Fecha','Hora','Source','Destination','Protocol','Length','Info'
#data_mejorado=data_union.copy()
#data_group=pd.DataFrame({'count': data_mejorado.groupby(['Hora','Destino','Protocolo',data_mejorado.index]).size()}).reset_index
data_group1=pd.DataFrame({'Contador': data_mejorado.groupby(['Hora','Origen','Protocolo','Tipo']).size()}).reset_index()
#data_group=data_mejorado.groupby(['Destino','Protocolo']).mean()
#data_group1

data_copia = data_group1.copy()
#agrupamos el datagrpuo
#data_group_sel=data_group[['Hora','Origen','Protocolo']]
#data_group_sel
#agrupamos con el count
#data_group1=pd.DataFrame({'count':data_group_sel.groupby(['Hora','Origen','Protocolo']).size()}).reset_index()
data_group1
```

	Hora	Origen	Protocolo	Tipo	Contador
0	00:30:43	104.244.42.69	TCP	NORMAL	1
1	00:30:43	192.168.1.102	TCP	NORMAL	1
2	00:30:44	104.244.42.69	TCP	NORMAL	1
3	00:30:45	104.244.42.69	TCP	NORMAL	1
4	00:30:46	104.244.42.69	TCP	NORMAL	1
5	00:30:47	104.244.42.69	TCP	NORMAL	1
6	00:30:48	192.168.1.102	TCP	NORMAL	1
7	00:30:49	104.244.42.67	TCP	NORMAL	1

In [11]:
```
data_sel_char1=data_group1[['Contador']]#agrupamos x el count
data_sel_char1

#normalizamos el dataset
data_normalized1=data_sel_char1.copy()

data_normalized1['Contador_nor'] = (data_sel_char1['Contador'] - data_sel_char1['Contador'].min()) / (data_sel_char1['Contador'].
pd.options.display.max_rows = None
data_normalized1
```

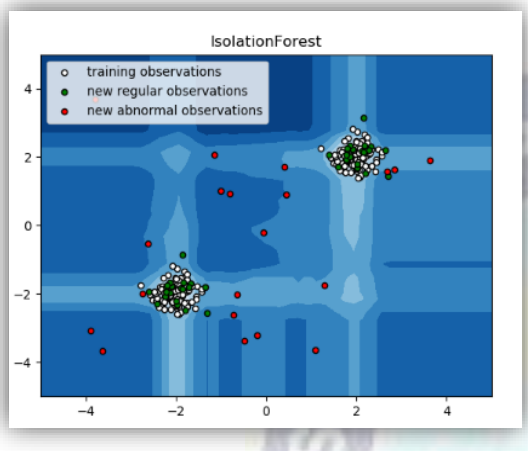
Out[11]:

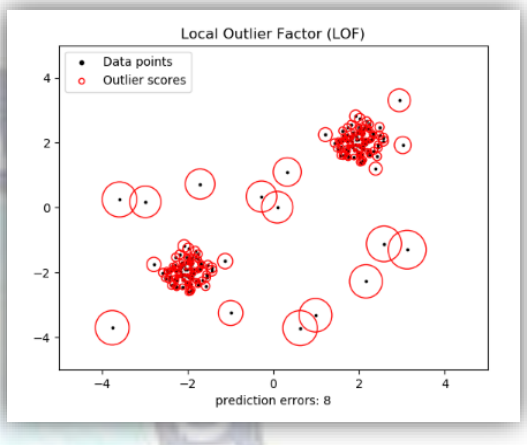
	Contador	Contador_nor
0	1	0.000000
1	1	0.000000
2	1	0.000000
3	1	0.000000
4	1	0.000000
5	1	0.000000
6	1	0.000000
7	1	0.000000
8	1	0.000000
9	1	0.000000
10	1	0.000000

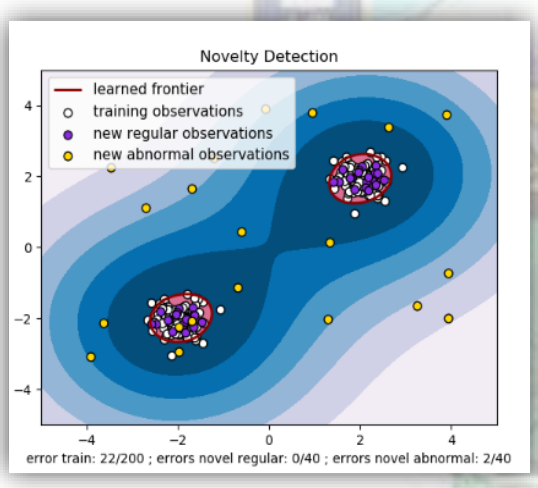
ANEXO B
Algoritmos de Machine Learning

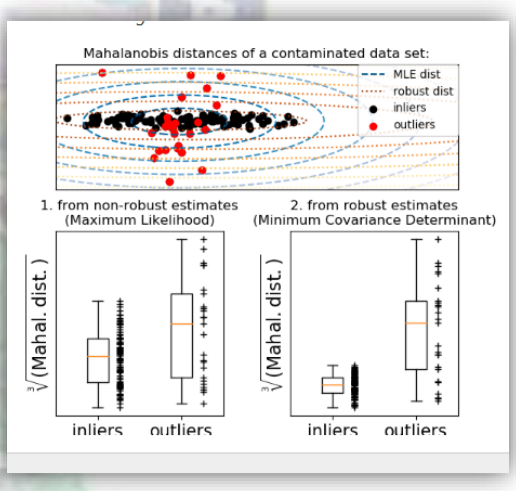
```
import time

import numpy as np
import matplotlib
import matplotlib.pyplot as plt

from sklearn import svm
from sklearn.datasets import make_moons, make_blobs
from sklearn.covariance import EllipticEnvelope
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor

print(__doc__)

matplotlib.rcParams['contour.negative_linestyle'] = 'solid'

# Example settings
n_samples = 300
outliers_fraction = 0.15
n_outliers = int(outliers_fraction * n_samples)
n_inliers = n_samples - n_outliers

# define outlier/anomaly detection methods to be compared
anomaly_algorithms = [
    ("Robust covariance", EllipticEnvelope(contamination=outliers_fraction)),
    ("One-Class SVM", svm.OneClassSVM(nu=outliers_fraction, kernel="rbf",
                                      gamma=0.1)),
    ("Isolation Forest", IsolationForest(contamination=outliers_fraction,
                                         random_state=42)),
    ("Local Outlier Factor", LocalOutlierFactor(
        n_neighbors=35, contamination=outliers_fraction))]
```

```
for i_dataset, X in enumerate(datasets):
    # Add outliers
    X = np.concatenate([X, rng.uniform(low=-6, high=6,
                       size=(n_outliers, 2))], axis=0)

    for name, algorithm in anomaly_algorithms:
        t0 = time.time()
        algorithm.fit(X)
        t1 = time.time()
        plt.subplot(len(datasets), len(anomaly_algorithms), plot_num)
        if i_dataset == 0:
            plt.title(name, size=18)

        # fit the data and tag outliers
        if name == "Local Outlier Factor":
            y_pred = algorithm.fit_predict(X)
        else:
            y_pred = algorithm.fit(X).predict(X)

        # plot the levels lines and the points
        if name != "Local Outlier Factor":  # LOF does not implement predict
            Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
            Z = Z.reshape(xx.shape)
            plt.contour(xx, yy, Z, levels=[0], linewidths=2, colors='black')

        colors = np.array(['#377eb8', '#ff7f00'])
        plt.scatter(X[:, 0], X[:, 1], s=10, color=colors[(y_pred + 1) // 2])

        plt.xlim(-7, 7)
        plt.ylim(-7, 7)
        plt.xticks(())
        plt.yticks(())
        plt.text(.99, .01, ('%.2fs' % (t1 - t0)).lstrip('0'),
                 transform=plt.gca().transAxes, size=15,
                 horizontalalignment='right')
        plot_num += 1

plt.show()
```


IsolationForest
training observations
new regular observations
new abnormal observations

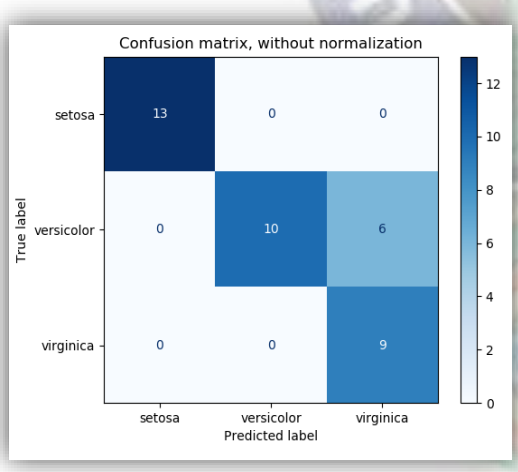

Local Outlier Factor (LOF)
Data points
Outlier scores
prediction errors: 8

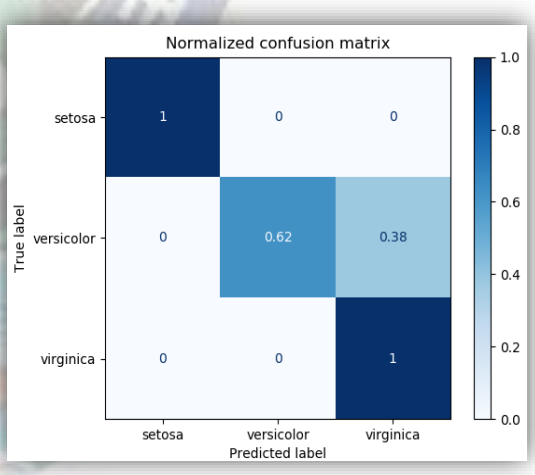

Novelty Detection
learned frontier
training observations
new regular observations
new abnormal observations
error train: 22/200 ; errors novel regular: 0/40 ; errors novel abnormal: 2/40


Mahalanobis distances of a contaminated data set:
MLE dist
robust dist
inliers
outliers
1. from non-robust estimates
(Maximum Likelihood)
2. from robust estimates
(Minimum Covariance Determinant)
inliers
outliers
inliers
outliers

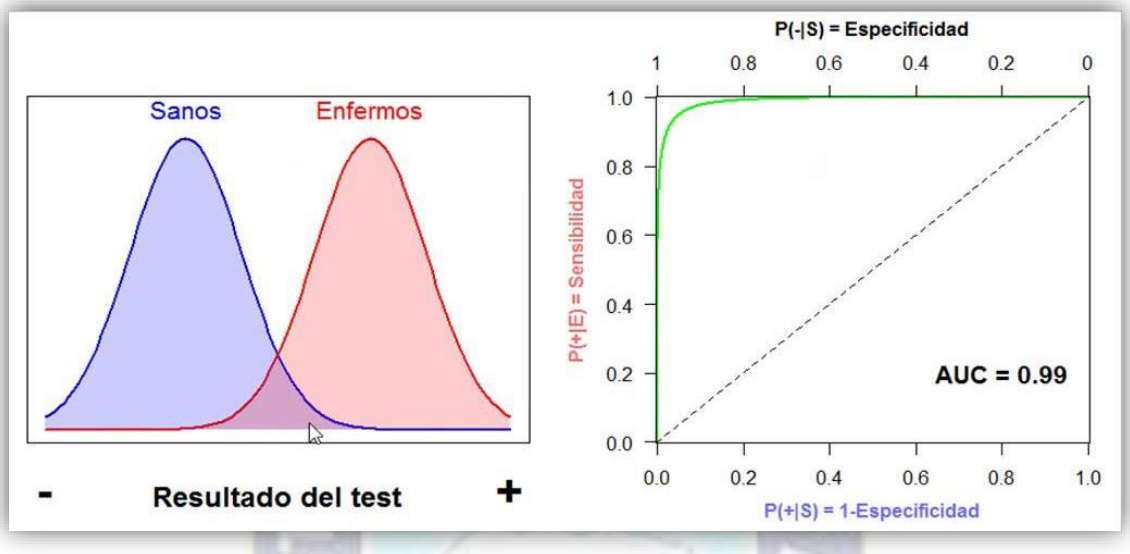
ANEXO C
Evaluación por Matriz de Confusión y ROC-AUC

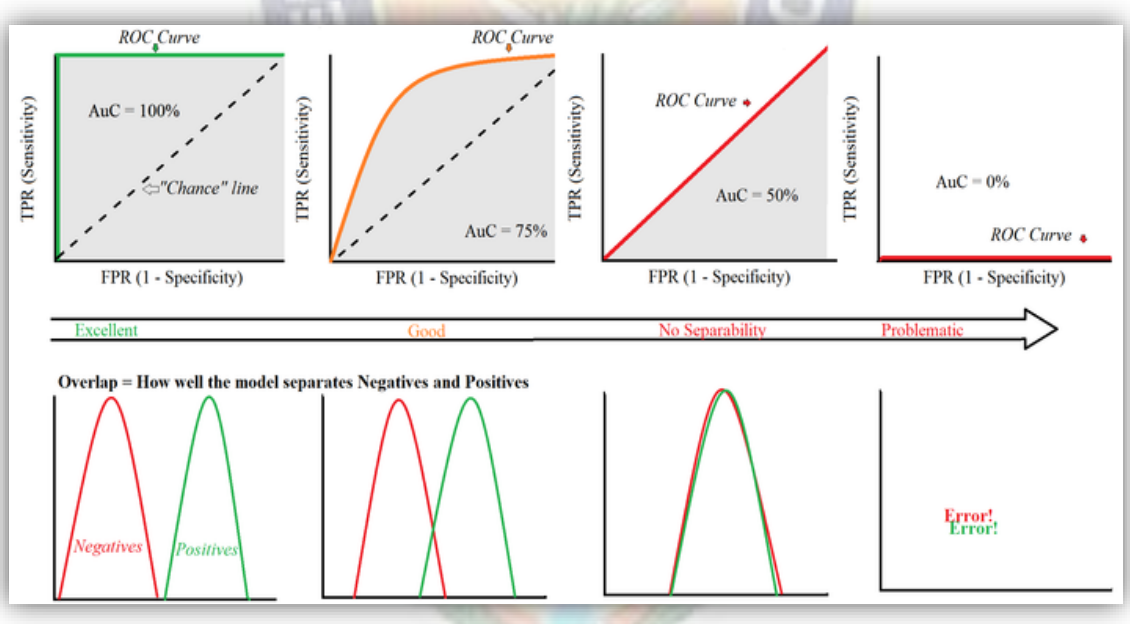
```python
print(__doc__)

import numpy as np
import matplotlib.pyplot as plt

from sklearn import svm, datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import plot_confusion_matrix

# import some data to play with
iris = datasets.load_iris()
X = iris.data
y = iris.target
class_names = iris.target_names

# Split the data into a training set and a test set
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

# Run classifier, using a model that is too regularized (C too low) to see
# the impact on the results
classifier = svm.SVC(kernel='linear', C=0.01).fit(X_train, y_train)

np.set_printoptions(precision=2)

# Plot non-normalized confusion matrix
titles_options = [("Confusion matrix, without normalization", None),
                  ("Normalized confusion matrix", 'true')]
for title, normalize in titles_options:
    disp = plot_confusion_matrix(classifier, X_test, y_test,
                                 display_labels=class_names,
                                 cmap=plt.cm.Blues,
                                 normalize=normalize)
    disp.ax_.set_title(title)

    print(title)
    print(disp.confusion_matrix)

plt.show()
```





ROC y AUC


Sanos
Enfermos
-
Resultado del test
+
P(-|S) = Especificidad
1
0.8
0.6
0.4
0.2
0
1.0
0.8
0.6
0.4
0.2
0.0
P(+|E) = Sensibilidad
AUC = 0.99
0.0
0.2
0.4
0.6
0.8
1.0
P(+|S) = 1-Especificidad


ROC Curve
AuC = 100%
"Chance" line
TPR (Sensitivity)
FPR (1 - Specificity)
ROC Curve
AuC = 75%
TPR (Sensitivity)
FPR (1 - Specificity)
ROC Curve
AuC = 50%
TPR (Sensitivity)
FPR (1 - Specificity)
AuC = 0%
ROC Curve
TPR (Sensitivity)
FPR (1 - Specificity)
Excellent
Good
No Separability
Problematic
Overlap = How well the model separates Negatives and Positives
Negatives
Positives
Error!
Error!

ANEXO D
Malware