UNIVERSIDAD DE COSTA RICA
SISTEMA DE ESTUDIOS DE POSGRADO

DESARROLLO DE UN MODELO DE DETECCIÓN DE URLS MALICIOSOS USANDO APRENDIZAJE AUTOMÁTICO SUPERVISADO

Trabajo final de investigación aplicada sometida a la consideración de la Comisión del Programa de Estudios de Posgrado en Ciencias de la Computación e Informática para optar al grado y título de Maestría Profesional en Computación e Informática

MICHELLE MARIE CERSOSIMO MORALES

Ciudad Universitaria Rodrigo Facio, Costa Rica
2023

Agradecimientos

A mi mamá en el cielo, que me apoyó hasta el último día.

A mi papá, hermanos y Dani que me motivaron siempre.

A mis profesores, por su paciencia y retroalimentación.

Este trabajo final de investigación aplicada fue aceptado por la Comisión del Programa de Estudios de Posgrado en Ciencias de la Computación e Informática de la Universidad de Costa Rica, como requisito parcial para optar al grado y título de Maestría Profesional en Ciencias de la Computación e Informática

ELENA GABRIELA BARRANTES SLIESARIEVA (FIRMA)
PERSONA FISICA, CPF-01-0866-0165.
Fecha declarada: 22/03/2023 04:21:13 PM
Esta representación visual no es fuente
de confianza. Valide siempre la firma.

Dra. Gabriela Barrantes Sliesarieva
Representante de la Decana Sistema de Estudios de Posgrado

ADRIAN LARA PETITDEMANGE (FIRMA)
PERSONA FISICA, CPF-01-1225-0679.
Fecha declarada: 22/03/2023 03:45:09 PM
Esta es una representación gráfica únicamente,
verifique la validez de la firma.

Dr. Adrián Lara Petitdemange
Profesor Guía

EDGAR ENRIQUE CASASOLA MURILLO (FIRMA)
PERSONA FISICA, CPF-01-0729-0430.
Fecha declarada: 27/03/2023 01:19:07 PM
Esta es una representación gráfica únicamente,
verifique la validez de la firma.

Dr. Edgar Casasola Murillo
Lector

RICARDO VILLALON FONSECA (FIRMA)
PERSONA FISICA, CPF-01-0699-0375.
Fecha declarada: 24/03/2023 03:51:01 AM
Esta es una representación gráfica únicamente,
verifique la validez de la firma.

Dr. Ricardo Villalón Fonseca
Lector

GABRIELA MARIN RAVENTOS (FIRMA)
Firmado digitalmente por GABRIELA MARIN RAVENTOS (FIRMA)
Fecha: 2023.03.22 16:37:05 -06'00'

Dra. Gabriela Marín Raventós
Directora del Programa de Posgrado en Computación e Informática

MICHELLE MARIE CERSOSIMO MORALES (FIRMA)
Digitally signed by MICHELLE MARIE CERSOSIMO MORALES (FIRMA)
Date: 2023.03.28 10:50:29 -07'00'

Michelle Marie Cersosimo Morales
Sustentante

Índice general

Resumen

La distribución de ataques cibernéticos mediante protocolos web ha sido una manera, para sus actores, de evadir detección y filtrado de red al camuflarse entre la cantidad de tráfico legítimo existente [1]. A pesar de la gran cantidad de sitios que son bloqueados diariamente, los atacantes siguen mejorando sus técnicas de evasión, por lo que se han realizado esfuerzos para mejorar el filtrado y bloqueo de estos ataques usando técnicas de aprendizaje automático. Los modelos de predicción basados en aprendizaje automático han ayudado a mejorar la capacidad de detección de los sistemas de monitoreo y presentan una alternativa de mayor escalabilidad, comparado a las tecnologías de bloqueo basadas en listas negras [10]. Estos modelos requieren una correcta caracterización del conjunto de datos para poder diferenciar un comportamiento de otro y su selección puede influir en los resultados de la predicción.

En este trabajo se usa caracterización léxica para la construcción de un clasificador que pueda detectar los principales tipos de URL de alto riesgo incluyendo malware, phishing, spam y botnet usando aprendizaje automático supervisado. Nuestros resultados mostraron que estas características otorgan flexibilidad a la etapa de implementación y logran obtener buenos resultados en su exactitud sin depender de características basadas en la observación de cambios en el tiempo.

Lista de tablas

Lista de figuras

Lista de abreviaturas

ML	Machine Learning
TTP	Tactics, Techniques and Procedures
APT	Advanced Persistent Threat
URL	Uniform Resource Locator
DNS	Domain Name System
DGA	Domain Generation Algorithm
PCA	Principal Component Analysis
TF-IDF	Term frequency – Inverse document frequency
C&C	Command and Control
ML	Machine Learning
TLD	Top Level Domain
SIEM	Security Information and Event Management
SPL	Search Processing Language
MLTK	Machine Learning Toolkit
RF	Random Forest
DT	Decision Tree
UT	URL Toolbox

UNIVERSIDAD DE COSTA RICA

SEP Sistema de Estudios de Posgrado

Autorización para digitalización y comunicación pública de Trabajos Finales de Graduación del Sistema de Estudios de Posgrado en el Repositorio Institucional de la Universidad de Costa Rica.

Yo, Michelle Marie Cersosimo Morales, con cédula de identidad 115750475, en mi condición de autor del TFG titulado DESARROLLO DE UN MODELO DE DETECCIÓN DE URLS MALICIOSOS USANDO APRENDIZAJE AUTOMÁTICO SUPERVISADO.

.Autorizo a la Universidad de Costa Rica para digitalizar y hacer divulgación pública de forma gratuita de dicho TFG a través del Repositorio Institucional u otro medio electrónico, para ser puesto a disposición del público según lo que establezca el Sistema de Estudios de Posgrado. **SI** [X] **NO** *[ ]

***En caso de la negativa favor indicar el tiempo de restricción:_____año (s).**

Este Trabajo Final de Graduación será publicado en formato PDF, o en el formato que en el momento se establezca, de tal forma que el acceso al mismo sea libre, con el fin de permitir la consulta e impresión, pero no su modificación.

Manifiesto que mi Trabajo Final de Graduación fue debidamente subido al sistema digital Kerwá y su contenido corresponde al documento original que sirvió para la obtención de mi título, y que su información no infringe ni violenta ningún derecho a terceros. El TFG además cuenta con el visto bueno de mi Director (a) de Tesis o Tutor (a) y cumplió con lo establecido en la revisión del Formato por parte del Sistema de Estudios de Posgrado.

MICHELLE MARIE CERSOSIMO MORALES (FIRMA)

Digitally signed by MICHELLE MARIE CERSOSIMO MORALES (FIRMA)
Date: 2023.03.28 10:42:56 -07'00'

FIRMA ESTUDIANTE

Nota: El presente documento constituye una declaración jurada, cuyos alcances aseguran a la Universidad, que su contenido sea tomado como cierto. Su importancia radica en que permite abreviar procedimientos administrativos, y al mismo tiempo genera una responsabilidad legal para que quien declare contrario a la verdad de lo que manifiesta, puede como consecuencia, enfrentar un proceso penal por delito de perjurio, tipificado en el artículo 318 de nuestro Código Penal. Lo anterior implica que el estudiante se vea forzado a realizar su mayor esfuerzo para que no sólo incluya información veraz en la Licencia de Publicación, sino que también realice diligentemente la gestión de subir el documento correcto en la plataforma digital Kerwá.

Capítulo 1: Introducción

La distribución de ataques cibernéticos mediante protocolos web ha sido una manera, para sus actores, de evadir detección y filtrado de red al camuflarse entre la cantidad de tráfico legítimo existente [1]. Los protocolos web se hallan entre aquellos más utilizados para llevar a cabo la comunicación entre víctima-adversario [2], donde el 80% de las familias de malware usan este medio para iniciar sus procedimientos de comando y control sobre los sistemas comprometidos [3]. Tan solo en el 2020 el servicio de seguridad Quad9 bloqueó un promedio de 10 millones de dominios diariamente [4]. Estos dominios forman parte principal de una URL, identificando el nombre por el cual un humano se refiere a la dirección IP del recurso que busca acceder, este y muchos más componentes pueden caracterizar una URL como maliciosa o legítima. Para el 2021 las cifras de URL de alto riesgo aumentaron aludido por la transición de la presencialidad al teletrabajo ocasionado por la pandemia [5]. A pesar de la gran cantidad de sitios que son bloqueados diariamente, los atacantes siguen mejorando sus técnicas de evasión, por lo que se han realizado esfuerzos para mejorar el filtrado y bloqueo de estos ataques usando técnicas de aprendizaje automático ([6], [7], [8], [9], [10], [11]). Los modelos de predicción basados en aprendizaje automático han ayudado a mejorar la capacidad de detección de los sistemas de monitoreo y presentan una alternativa de mayor escalabilidad, comparado a las tecnologías de bloqueo basadas en listas negras [10]. Estos modelos pueden dividirse en aprendizaje supervisado, el cual usa datos ya etiquetados para su entrenamiento, y el no supervisado el cual trabaja con los datos sin etiquetar. El aprendizaje supervisado logra mejores resultados en un menor tiempo [8], pero ambos requieren una correcta caracterización del conjunto de datos para poder diferenciar un comportamiento de otro. Estos patrones pueden ser extraídos gracias a que cuando una URL maliciosa revela su comportamiento, es inevitable que introduzca algunas características diferentes a las de un dominio legítimo [12]. La extracción de estas características juega un rol fundamental en los resultados de la predicción final, por esta razón proponemos una caracterización léxica para la construcción de un clasificador que pueda detectar los principales tipos de URL de alto riesgo incluyendo malware, phishing, spam y botnet usando aprendizaje automático supervisado.

1.1. Problema

Los modelos de clasificación de URL maliciosas existentes pueden tener un alcance de predicción único o múltiple basado en las categorías maliciosas que conforman su conjunto de datos [13]. Los de alcance único se enfocan en predecir un comportamiento malicioso específico como sitios de phishing o de botnet. Aquellos con alcance múltiple en cambio trabajan con un conjunto de datos más variado y pueden clasificar que tipo de comportamiento malicioso es [12]. Este último enfoque es más realista, pues las organizaciones no sólo se ven afectadas por un único comportamiento malicioso y un modelo de detección debe poder clasificar múltiples ataques. Los estudios que detectan múltiples comportamientos maliciosos, sin embargo, son rígidos con respecto a los métodos de caracterización utilizados. La caracterización de URL usada por estos modelos dependen del uso de características basadas en la observación de los cambios que posee una URL en una ventana de tiempo, llamadas también características dinámicas ([14], [6], [15]). Este tipo de caracterización es compleja y necesita almacenar cada estado en el tiempo para describir si un comportamiento es malicioso o no. Este método falla cuando hay URL compuestas por dominios con tiempos de vida cortos, más aún cuando se ha visto que las URL que forman parte de campañas maliciosas recientes tienen un tiempo de vida promedio de 21 horas [16], degradando así la relevancia de esa característica para la predicción final. Existen a su vez características léxicas que se basan en información presente de la hilera de caracteres de una URL, pero no existen modelos de clasificación que usen sólo características léxicas para poder predecir múltiples tipos de comportamientos maliciosos. Por esta razón, se busca responder con esta investigación la pregunta de si es posible obtener una alta exactitud en la detección de múltiples clases de URL maliciosas mediante un modelo de aprendizaje automático supervisado usando sólo caracterización léxica, con el fin de mejorar el procesamiento en la creación del modelo y evitar recolectar información a partir de la observación de cambios de URL en el tiempo.

1.2. Objetivos

A continuación, se describe el objetivo general, así como los objetivos específicos que son abordados en esta propuesta.

2.1.1. Objetivo general

Desarrollar un modelo de detección de URLs maliciosos usando aprendizaje automático supervisado.

2.1.2. Objetivos específicos

1. Identificar técnicas existentes para la caracterización y detección de tráfico de DNS mediante una revisión de literatura.

2. Implementar un clasificador con aprendizaje automático para identificar dominios legítimos y maliciosos.

3. Evaluar el clasificador implementado de acuerdo con su exactitud, precisión, recall y F1 para identificar clases de dominios legítimos y maliciosos.

1.3. Justificación

Se estima que en los próximos años el porcentaje de URL maliciosas no sólo aumentará, sino que además los atacantes usarán estas URL de manera estratégica basada en los acontecimientos que suceden durante el año [5]. A esta fluctuación se le suma la innovación en técnicas para evadir detección, como los algoritmos de generación de dominios (DGA), la suplantación de dominios, el uso de tiempos de vida cortos, entre otros ([17], [18], [19]). Por esta razón los modelos de clasificación de URL maliciosas deben poder detectar una variedad de comportamientos y tener la flexibilidad para adaptarse a diversos tipos de comportamientos maliciosos. Un clasificador que pueda adaptarse a estos cambios requiere de una caracterización ligera, para disminuir la cantidad de información necesaria para actualizar el modelo en tiempo real. En un contexto organizacional puede ayudar a mejorar la predicción de categorías maliciosas e influenciar el despliegue de acciones de bloqueo antes de que se materialice un ataque. En contraste a otros estudios, nuestro modelo usa características léxicas basadas en los caracteres de la URL, estas son rápidas de extraer y, dado que su recolección no depende del seguimiento de cambios en el tiempo, puede reducir la cantidad de datos a procesar y otorgar una mayor flexibilidad, para agregar o remover características según la innovación de los patrones por parte de los atacantes. Finalmente, los estudios actuales usan una combinación de herramientas para la creación de sus modelos, como lo son bases de datos y extractores de características ([20], [14], [21] y [22]), así como librerías externas para su análisis y visualización ([8], [9], [23], [24]). Para evitar esta división de procesos en múltiples herramientas se propone el uso de Splunk [25] para llevar a cabo todas las etapas del aprendizaje del modelo.

1.4. Estructura del documento

El resto de esta propuesta se organiza de la siguiente forma: en la sección 2 expondremos trabajo relacionado con nuestra investigación. En la sección 3 se presenta el marco conceptual donde se describen algunos conceptos sobre el uso malicioso de protocolos de capa de aplicación, aprendizaje automático y Splunk. Posteriormente en la sección 4 explicaremos la metodología usada en este trabajo. Se presentan los resultados obtenidos en la sección 5 y finalmente las conclusiones en la sección 6.

Capítulo 2: Estado del arte

En los estudios relacionados al tema de detección de URL maliciosas mediante aprendizaje automático, se identificaron cuatro etapas esenciales para explicar los avances en el área y diferenciar sus enfoques con nuestra investigación. En la etapa 1 se realiza la selección del conjunto de datos a trabajar, la etapa 2 incluye la extracción de las características a partir de los datos crudos, la etapa 3 es determinada por la técnica de aprendizaje a aplicar y finalmente la etapa 4 procede a evaluar en las distintas métricas el modelo generado. En la figura 1 se muestran tipos y ejemplos de cada una de estas etapas y son descritas en detalle a continuación.

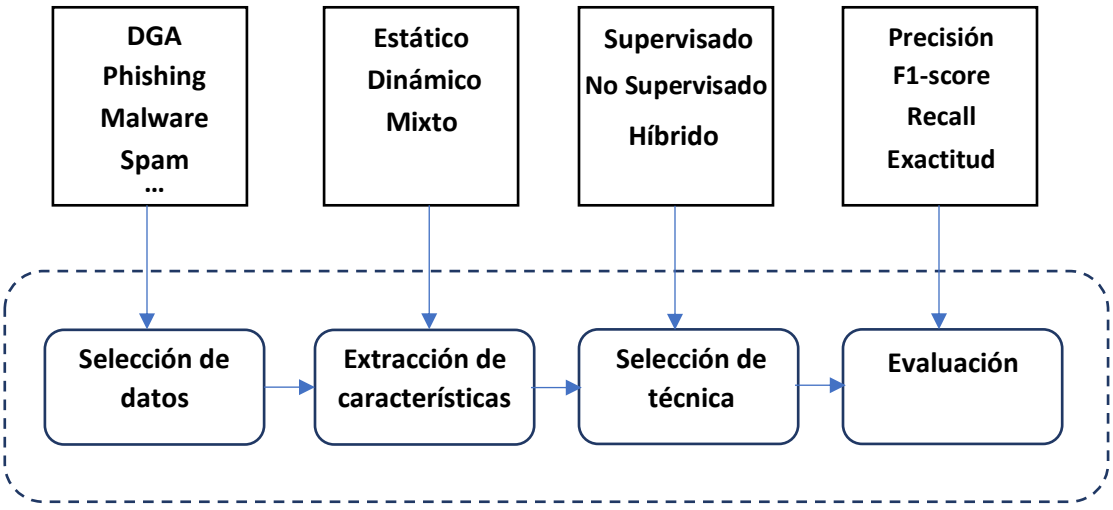

Figura 1: Etapas de aprendizaje automático

- **Selección de los datos:** se refiere al conjunto de datos usado para entrenar el clasificador y posteriormente diferenciar cada URL entre maliciosa o legítima. En esta etapa, podemos clasificar los estudios existentes en dos:
    - **Alcance de predicción único:** usan conjuntos de datos compuestos de una sola categoría de URL maliciosa. A pesar de que nuestra investigación no usa un alcance único, el análisis de las características usadas en estos modelos puede enriquecer nuestro conjunto de características final. Entre los modelos de detección encontrados en esta área podemos listar los siguientes:

        - **Botnets:** Se encuentran los modelos de Wang et al. [9], Kelley et al. [10], Bao et al. [25] y Yadav et al. [26].
        - **Phishing:** Analizados en Bhoj et al. [27], Kumar et al. [17], Le Page et al. [28], Pradeepthi et al. [29] y Patil et al. [30].
        - **Spam**: En menor escala, pero disponibles en los modelos de Patil et al [30] y Henke et al. [31].
        - **Malware:** Abarcados ampliamente en Hajaj [18], Antonakakis et al [20], **Bharadwaj** et al. [32], Yan et al. [33], **Wang** et al. [34] y Wu et al. [22].
    - **Alcance de predicción múltiple:** en este tipo de estudios el concepto de "URL malicioso" puede ser interpretado de múltiples maneras según su comportamiento y objetivo del adversario [13]. Aquí se usan una combinación de categorías de URL maliciosas con el fin de simular un escenario más realista asemejándose al ambiente que viven las organizaciones diariamente. Normalmente se usan las categorías de más alto riesgo según el estado actual del internet como lo son phishing, malware, contenido adulto, botnet, spam, entre otras. Podemos mencionar los modelos de Mahdavifar et al [6], Bilge et al [14] y Palaniappan et al. [15].

- **Extracción de características:** involucra la extracción de atributos a partir del conjunto de datos que puedan representar las diferentes categorías a predecir. La variedad de estas características ha sido documentada en múltiples revisiones de literatura ([12], [13]), agrupándolas según su forma de extracción o contexto. Cheng et al. [12] las separa en estáticas y dinámicas según la dependencia de la información con respecto al tiempo. Por su parte Zhauniarovich et al. [13] usa tres dimensiones para agrupar características: según el dónde provenga la información extraída, la dependencia de recolección de información histórica y la relación entre dominios del mismo conjunto de datos. Nuestra investigación usa el agrupamiento de Cheng para diferenciar los enfoques existentes, tanto aquellos que usan solo características estáticas: ([9], [10], [31], [30], [32], [22]) o una combinación de ambas ([17], [20], [14], [21], [25], [27], [28], [35], [36]). A continuación, describimos algunas de las características estáticas usadas en los estudios relacionados al tema de investigación.

- Las URL de botnets usan generadores pseudoaleatorios o basados en diccionario, por lo que estudios basados en la detección de estas como el de Kelley et al. [10] y Yadav et al. [26] suelen aprovechar esa aleatoriedad para extraer características basados en frecuencia de caracteres y distribución de letras u entropía de la hilera. La longitud es una de las características más utilizadas en estudios de detección de botnets así como el número de dígitos. En menor medida, se han usado características basadas en el índice de Jaccard y la distancia de Levenshtein para comparar grupos de caracteres.
- Para caracterizar las URL de malware se han usado también características léxicas como en el reciente estudio de Bharadwaj et al. [32] donde incluyen la longitud de diferentes secciones de la URL, extensiones del archivo referenciado en caso de existir, cantidad de dígitos, vocales y consonantes y aparición de la dirección IP. Hajaj [18] usa también la longitud, la entropía y el número de caracteres específicos consecutivos. Finalmente, Yan et al. [33] agregan el uso de palabras clave basadas en listas para discernir tendencias maliciosas.
- Con respecto a la caracterización de URL de phishing, se encuentran los estudios de Kumar et al [17], Le Page et al. [28] y Pradeepthi et al. [29] en los que se destacan la aparición de la dirección IP, sustantivos desconocidos, palabras sensibles como “banca” o “login”, nombre de compañías, el número de puntos, delimitadores y guiones, número de argumentos, entre otros.
- Para describir URL de spam, Henke et al. [31] resaltan la poca formalidad existente en la literatura para seleccionar características relevantes y realizan un análisis de efectividad para un modelo de clasificación de spam tomando en cuenta la evolución de 700 características iniciales a partir de signos de puntuación, cantidad de caracteres específicos y palabras encontradas, hasta su reducción a 300 características.
- Finalmente, hay métodos de detección de dominios maliciosos basado completamente en el uso de características léxicas como el de Wu et al. [21] y su modelo FTPB. En este estudio se extraen características interesantes, como la proporción de la mayor cantidad de caracteres significativos, puntuación de n-gramas, longitud de los nombres del subdominio y número de intercambios de

caracteres alfanuméricos. Además, como su punto de partida es únicamente el dominio, expanden la cantidad de información al aplicar TF-IDF para extraer términos relevantes y proceden a aplicar un método de reducción de dimensiones al vector generado usando el método de análisis de componente principal (PCA).

- **Selección de la técnica:** se refiere a los enfoques de clasificación utilizados y sus diferentes algoritmos. Estos enfoques pueden ser aprendizaje supervisado, aprendizaje no supervisado, un híbrido, entre otros. Los estudios existentes poseen una tendencia a usar algoritmos basados en árboles como bosques aleatorios (RF) y árboles de decisión (DT) ([9], [25], [17], [28], [29], [20], [22]), pero también se han hecho aplicaciones con algoritmos de regresión logística (LR) [15], vecino más cercano o *k-Nearest Neighbors* (KNN) ([27], [6]), redes neuronales (NN) [32], entre otros. En varios estudios hacen una comparación de algoritmos ([17], [9], [28]) donde los algoritmos basados en árboles sobresalen obteniendo los mejores resultados en términos de exactitud.

- **Evaluación:** comprende el tipo de métrica usada para evaluar los resultados de la clasificación. La mayoría de los estudios recopilados se enfocan en clasificar sitios legítimos de maliciosos en términos de exactitud, pero también se utilizan métricas de recall y la tasa de falsos positivos. Dado que nuestra investigación usa un conjunto de datos con varias categorías, nos enfocamos en los resultados obtenidos por estudios bajo esa misma hilera.

    - Mahdavifar et al. [37] usan aprendizaje automático para detectar dominios maliciosos en 3 categorías: spam, phishing y malware usando 32 características divididas en 3 grupos: 1) léxicas, 2) estadísticas históricas basadas en las respuestas de DNS y por último 3) información de fuentes externas como información del registrante, fecha de creación del dominio, etc. Alcanzan una exactitud del 94.8% aplicando KNN. Este estudio usa una alta cantidad de características significado en mayor cantidad de información por dominio y el uso de características externas normalmente posee costos asociados al uso de APIs para su recolección.

- Bilge et al. [14] introducen el sistema EXPOSURE para detectar dominios maliciosos en un conjunto de datos mixto de spam, malware, phishing, botnet, contenido adulto, entre otras. Extraen 15 características entre estáticas y dinámicas incluyendo características basadas en la respuesta de la consulta y el tiempo de vida de la petición, así como la longitud y la cantidad de dígitos. Obtienen un 98% de exactitud usando árboles de decisión.
- Similar a EXPOSURE, Antonakakis et al. [20] proponen el sistema de reputación Notos, que usa también información de DNS histórica pasiva para recolectar tres categorías a lo largo de 41 características: de red, basadas en zona y basadas en evidencia. Usan listas de dominios previamente bloqueados por lo que su conjunto de datos es mixto y obtienen un 96% de exactitud utilizando bosques aleatorios.
- Así mismo, Palaniappana et al. [15] usan un conjunto de datos con muestras de phishing, spam y malware, del cual extraen 17 características en tres grupos: basadas en información del dominio, basadas en características del sitio web y léxicas. Obtienen un 60% de exactitud usando el algoritmo de regresión logística.
- Finalmente, en un trabajo anterior en Cersosimo et al. [38] se realizó una primera aproximación del tema dirigido a la detección de dominios únicamente y no URL. Por lo que fue necesario realizar un procesamiento inicial en el que se extrajeron los dominios de los sitios web en el conjunto de datos inicial. Se usaron 12 características estáticas para describir los dominios y se obtuvo un 88% de exactitud con bosques aleatorios.

Resumiendo, la cantidad de estudios de predicción múltiple son menos comparados a los de predicción única, esto dado que la tarea de seleccionar un grupo de características para describir múltiples comportamientos es más compleja. Otro punto importante es que los estudios existentes han requerido tanto el uso de características estáticas como dinámicas cuando tratan de predecir conjuntos de datos con más de dos categorías. Finalmente vemos una tendencia en el uso de algoritmos basados en árboles al generar buenos resultados de predicción independiente del conjunto de datos seleccionado.

Capítulo 3: Marco conceptual

A continuación, explicamos algunos conceptos de importancia para la investigación con respecto a los tipos de URL maliciosas seleccionadas, los enfoques de aprendizaje automático con sus etapas más importantes y concluimos esta sección describiendo cómo funciona Splunk y sus comandos.

3.1. URL maliciosas

Las URL se utilizan para referirse a la ubicación de recursos de internet únicos. Cuando una URL se utiliza con fines maliciosos se entiende que fue creada intencionalmente para causar algún daño y posee un alto riesgo de amenaza para el usuario final. Las tecnologías de seguridad se basan en este riesgo como determinante para bloquear ciertas categorías o grupos de URL. Por ejemplo, aquellos sitios con categorías de malware y phishing son comúnmente asociadas a tener un alto riesgo y por ende se bloquean en tecnologías de filtrado [39]. Estas categorías no siempre están disponibles si el sitio no ha sido analizado o reportado anteriormente, por lo que mantener las listas actualizadas es uno de los mayores retos.

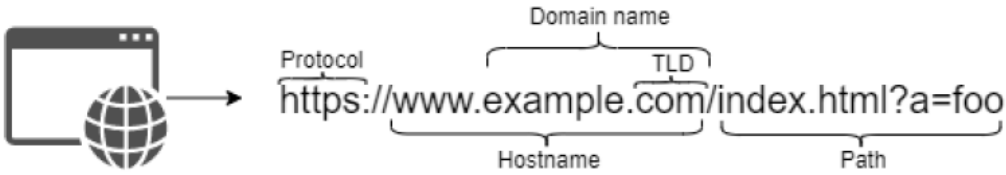

Figura 2: Estructura de la URL

En la figura 2 se muestra la estructura de una URL. Si se lee de izquierda a derecha, una URL incluye primero el esquema o protocolo, seguido del nombre de dominio y posibles subdominios con la extensión del dominio o *Top Level Domain* (TLD), terminando con la dirección a un determinado recurso o *path*. El dominio es una parte fundamental de la URL pues es el nombre otorgado al lugar en donde pueden accederse a los recursos de ese sitio web. Los nombres de dominio se apegan a una convención con respecto a su manera de escribirse: pueden constar solo de letras (A-Z), números (0-9) y signos de guión (-) y el primer carácter debe ser una letra. El *fully*

qualified domain name (FQDN) es el nombre del dominio completo del servidor albergando el recurso web, este nombre debe tener como mínimo 2 caracteres y máximo 255 caracteres, con un máximo de 63 caracteres por sección o etiqueta [40]. A su vez, es posible acceder al recurso web mediante su dirección IP directamente, por lo que algunas veces las URL pueden incluir la dirección IP y el puerto en caso de no utilizarse los puertos por defecto.

Los atacantes pueden abusar de los protocolos web y albergar diferentes comportamientos maliciosos en URL. Los comportamientos más comunes son phishing, botnet, spam y malware ([18], [12]). Para cada categoría hay técnicas comunes que son usadas por los atacantes en la construcción de su URL que pueden ayudar a diferenciarlas de una URL legítima. Podemos listar las siguientes:

- **Phishing:** cambios de TLD, similitud de suplantación de marca, ofuscación mediante URL largas.
- **Spam:** uso de palabras atrayentes, uso de identificadores únicos para evadir filtros de spam.
- **Botnets:** algoritmos de generación automática (DGA), uso de Fast-Flux y Domain-Flux.
- **Malware:** uso de redireccionamiento, incluye el nombre del archivo en sus URL de descarga, uso de IP para evitar resolución del dominio.

Para comprender ampliamente el conjunto de datos y su selección de características, se explican a continuación sus principales comportamientos por categoría, agregamos además un resumen de las técnicas y ejemplos de URL para cada categoría en el cuadro 1.

Cuadro 1: Ejemplo de URL maliciosas

Categoría	**Ejemplo de URL**
Malware	http://195.133.18.171/frggg.exe
Botnet	g1gqh760t2v2lc3uhom2v3o110ie2h.appsync-api.us-west-2.avsvmcloud.com
Phishing	http://amazon.com-verificationaccounts.darotob.com/Sign-in/5b60fcc60b36d1c3d
Spam	bsp42iv8wlefhh2.ru
Legítimo	www.newstimes.com

3.1.1. Phishing

Las URL categorizadas como phishing contienen formularios o páginas de inicio de sesión falsas con el fin de engañar a los usuarios para suministrar información, comúnmente haciéndose pasar por sitios legítimos [27]. El objetivo principal de los atacantes es realizar robo de credenciales o información sensible para luego extorsionar o monetizar con la información a las víctimas. Una técnica común en la creación de sus dominios es usar dominios léxicamente iguales a dominios legítimos pero con TLD distintos para engañar a los usuarios, normalmente se les llama *look-alike domains*. Para mitigar esta técnica las compañías compran estos dominios para proteger su marca y redireccionan a los usuarios a la página principal, pero el costo de adquirir múltiples dominios es alto y no todas las compañías pueden hacerlo. También utilizan métodos como *typosquatting o domainsquatting*, en el que escriben sus sitios web de manera errónea deliberadamente (por ejemplo, amazn o facebok) [17]. Mandiant en su reporte sobre el impacto de las comunicaciones de comando y control, detalla el uso de esta técnica para suplantar la identidad de los cinco dominios más visitados en Alexa Top 100 y reporta más de 200 dominios intentando simular google.com con pequeños errores tipográficos como “gooqle” [19]. Otro patrón común en las URL de phishing es el uso de ofuscación [17], usan una gran cantidad de caracteres para generar sus URL, escondiendo así el dominio real del recurso web como se ve en el ejemplo de phishing en el cuadro 1. Finalmente, este tipo de URL suelen traer consigo el correo de la persona a la que fue dirigido el ataque luego de que esta hiciese click en la URL, para luego ser utilizado en un formulario autocompletado y simular que es una página previamente visitada. Encontrar caracteres “@” dentro de la hilera de consulta es un indicador sospechoso en esta categoría ([30], [41]).

3.1.2. Spam

En esta categoría los adversarios utilizan métodos para incitar el flujo de un usuario hacia un sitio no deseado usando palabras como “*Dear Friend*”, “*Free sample*”, “*Brand new*” o “*Billion dollars*” [42]. Estas URL pueden ser colocadas dentro de anuncios web, en el cuerpo del mensaje o en imágenes dentro de correos. Su meta es dirigir el flujo a sitios de estafas y fraude. A pesar de que sus métodos son oportunísimos, no suelen ser completamente aleatorios. Estos correos o

anuncios buscan generar interés del usuario para incitar a seguir el enlace por lo que es común que se utilicen nombres de productos o subscripciones reales. Este no es el único método de acción, también puede ser generado por un virus previamente instalado como lo hizo DNSChanger en la operación Ghost Click [43] al cambiar la configuración de DNS de la víctima por servidores falsos redireccionando el tráfico a sitios de publicidad, obteniendo casi 14 millones de dólares ilegalmente e infectando 4 millones de computadoras. Su principal objetivo es la monetización mediante publicidad usando sistemas de Pay-per-Click (PPC) generando así fraude o *click-fraud*, cuyo fin es obtener una ganancia malversada generada por un click que no fue dado intencionalmente por un usuario [44].

3.1.3. Botnet

Las URL clasificadas como botnet abarcan el comportamiento malicioso más común basado en el abuso de protocolos de aplicación [12]. En una botnet se construye un canal de control remoto o comunicación de comando y control (C&C) entre el servidor padre y los sistemas infectados. La mayor debilidad de una red de nodos bajo control es que necesita estrictamente comunicarse con el servidor que lleva el mando [12]. Por esto, los atacantes utilizan varios métodos para construir la comunicación sin ser detectados. Entre ellos pueden mencionarse Domain-Flux y Fast-Flux [12].

- **Domain-Flux:** es una tecnología basada en el uso de algoritmos de generación de nombres, que utilizan una gran cantidad de caracteres aleatorios para la creación de sus hileras. Por lo general utilizan el mismo algoritmo múltiples veces para tener un gran conjunto de sitios, de modo que puedan fácilmente rotar el control, y la máquina infectada intentará contactar a cada uno de estos hasta obtener resolución del dominio activo [26]. Esta técnica se utiliza para evitar ser agregados a listas negras o mantener una puntuación baja en cuanto a perfiles de riesgo, aunque ahora hay tecnologías que aumentan dicha puntuación basada en proximidad, pero dicha técnica se sale del alcance de esta investigación [12]. Inicialmente utilizaban caracteres aleatorios con probabilidades de aparición iguales para cada carácter por lo que era fácil denotar a simple vista, pero el avance de algoritmos de generación de

nombres basados en diccionario resuelve este defecto. Por ello, investigadores han optado por usar características extraídas de las palabras encontradas en el dominio.

- **Fast-Flux:** es un método que cambia las relaciones entre nombres de dominios e IP, por lo que las peticiones a un dominio pueden devolver resultados diferentes para así ocultar la IP original del C&C. Entre sus principales características posee un tiempo de vida corto y la cantidad de resoluciones aparece más de una vez por lo que se pueden usar los registros A y la cantidad de registros NS asociados al dominio como un atributo en su identificación [12]. Las técnicas de Fast-Flux quedan fuera del alcance de esta investigación pues queremos evitar la recolección histórica de características dependientes de una ventana de tiempo determinada.

3.1.4. Malware

Los atacantes usan sitios cuya infraestructura está pensada para la distribución de software malicioso que pueden ser virus, trojanos o hasta ransomware y spyware normalmente son sitios clasificados como malware por el alto riesgo al usuario final. Usualmente utilizan técnicas para evitar ser agregados a listas negras similares a las de botnets [18] pero sus patrones de dominio son más legibles. A su vez, sus ataques son más sofisticados y comúnmente usan más de una capa de comando y control en su infraestructura, por lo que pueden utilizar tanto dominios con tiempo de vida corto (*short-haul*) como dominios a largo plazo (*long-haul*) para mantener su actividad constante en caso de que uno de sus dominios sea detectado y bloqueado. Para los dominios con un tiempo de vida corto, suelen no tener una categoría asignada al ser aún desconocidos para la mayor parte del Internet por lo que suelen pasar el filtrado por categoría de herramientas de proxy o firewall. Su función es ser el primer salto para redireccionar el flujo a otros sitios de distribución y contenido de software malicioso intencionadas (*drive-by downloads*). Otra característica es que suelen ser prescindibles de modo que, si terminan siendo bloqueados los atacantes solo deben de generar nuevos dominios. Por otra parte, los atacantes más habilidosos suelen comprometer sitios de blogs y páginas descontinuadas para aprovecharse del tiempo de vida del dominio. Un dominio con más de 1 año de creación y una actividad histórica regular suele disminuir las sospechas o al

menos dificultar el análisis en un incidente. Por esta razón el uso de características basadas en categorías dadas por proveedores o basadas en la edad del dominio pueden ser fácilmente falsificadas por los adversarios.

3.2. Caracterización de URL

La extracción de características se refiere al proceso de transformar datos crudos en características numéricas que puedan ser procesados mientras se conserva la información relevante del conjunto de datos original. Existen métodos para seleccionar o combinar estos datos en características y su objetivo es reducir la cantidad de datos a analizar, así como acelerar el aprendizaje de máquina [45]. En el caso de URL este concepto puede entenderse como al proceso de capturar el comportamiento de una URL basado en los atributos que la componen y poder representarlos numéricamente. Se acostumbra a usar operaciones matemáticas y estadísticas para su recolección, por ejemplo, conteo, promedios, entropía, entre otros. En la sección 2, se mencionaron varias revisiones de literatura características posibles para sitios maliciosos dependiendo de su tipo, ya sean estáticas o dinámicas.

La caracterización de URL influye directamente en el resultado de la predicción del modelo, por lo que vemos importante resumir los tipos de características, mencionados en la sección 2, agrupados en la revisión de literatura de Cheng et al. [12]. En su investigación agrupan la extracción de características de acuerdo con su tiempo de recolección en dos tipos: estáticas y dinámicas. A pesar de que las características dinámicas alcanzan una buena exactitud en detectar URL maliciosas, generalmente requieren un alto grado de esfuerzo y tiempo para recolectarlas, y no son recomendables para análisis de tiempo real [32]. Por ello vemos relevante explicar la variedad de características existentes para entender sus diferencias. En la sección 3.2.1 describimos los tipos de características usados en la literatura y en la sección 3.2.2 el rol que poseen estas características en la predicción final según su relevancia.

3.2.1. Tipos de características

La definición de características estática dada por Cheng la presenta como aquella que no se enfoca en los cambios dinámicos de un dominio en el tiempo, es decir que no posee una asociación entre la construcción del dominio y la puntualidad [12]. Según su clasificación se separan en 5 tipos:

I. **Basadas en la reputación de la IP**
   - Se obtiene al analizar información disponible de la dirección IP asociada al dominio y se determina por transitividad que si una dirección IP ha sido clasificada como maliciosa entonces el actual dominio tiene altas posibilidades de ser malicioso. Ha sido utilizado por [18], [43] y [44] en sus análisis de tráfico de DNS.

II. **Basadas en los componentes léxicos**
   - Se obtienen a partir de la hilera de caracteres que componen el nombre del dominio, como la longitud, la proporción de vocales y dígitos, el uso de palabras de diccionario, entre otras. Este tipo han sido utilizadas en su mayoría en estudios de detección de DGA ([26], [36], [46]).

III. **Basadas en información de la resolución del dominio**
   - Se obtienen al recolectar la distribución geográfica del dominio a partir del país y región y sistemas autónomos (ASN), dado que los ataques suelen dirigirse a varias regiones de forma oportunista, lo cual es anómalo pues las resoluciones suelen estar más concentradas por geografía. Este tipo de características han sido usadas por [36], [35] y [47].

IV. **Basadas en información de registro o *Whois***
   - Se obtienen al consultar la base de datos de Whois, la cual contiene una amplia lista de registros con información que pueden identificar al dueño del dominio e información de contacto, así como información de registro del dominio. Normalmente los dominios maliciosos usan servicios de hosting gratuitos como otra técnica de evasión. Estas han sido utilizadas por [35] y [48].

V. **Basadas en información de *Passive DNS***

- La información de *passive DNS* es información histórica de los registros de consultas de DNS, las cuales pueden determinar técnicas de atacantes como Fast-Flux. Estas han sido utilizadas por [49], [50], [20], [14] y [21].

Por su parte, las características dinámicas necesitan un periodo de observación para evaluar y aproximar un patrón de comportamiento [12]. En otras palabras, se necesita monitorear y analizar los cambios en la información del dominio en una ventana de tiempo.

I. Tasa de crecimiento de las direcciones IP

- La cantidad de apariciones distintas de direcciones IP asociadas a un dominio, que muestran la variación del sitio en el tiempo. Estas han sido utilizadas por [21].

II. Basadas en los cambios del tiempo de vida (TTL)

- El TTL hace referencia a la cantidad de tiempo que debe permanecer en caché un determinado dominio antes de actualizarse automáticamente. Los atacantes obtienen una mejor disponibilidad de sus dominios al usar valores de TTL más pequeños y éstas han sido usadas por [21], [14].

III. Basadas en la consulta

- Se obtienen a partir de la tasa de crecimiento en los cambios en la consulta, ya sea que muy rápidamente incremente o decrezca por el uso de métodos de Domain-Flux. Éstas han sido utilizadas por [14], [35], [20], [36] .

3.2.2. Relevancia

Como se mencionó en la sección 3.2, los problemas de clasificación usan técnicas de selección de características para reducir el espacio de características y acelerar el procesamiento y aprendizaje del algoritmo ([45], [31]), esta selección puede ser manual o automática, y es determinado por la relevancia de cada una de las características. La selección impacta la calidad final del clasificador en términos de exactitud y robustez [13]. A pesar de que la relevancia de una característica no está formalmente definida en la literatura, Henke et al. [31] la definen como:

> "Una característica f_i es suficientemente relevante si el alto rendimiento de la clasificación (según todas las características) decrece significativamente cuando f_i se elimina del conjunto de características actual. Por otra parte, una característica es débil de relevancia cuando la contribución de esta característica en la exactitud de la clasificación es variable o inexistente".

Aquellas características que describen adecuadamente el conjunto de datos pueden contribuir al éxito del enfoque de aprendizaje, por el contrario, características que no aportan conocimiento de su categoría pueden arruinar los algoritmos de detección por más válido que sea su aplicación [13]. A pesar de la importancia de éstas en la predicción, no es fácil evaluar que tan robusto es una selección de características de forma sistemática por la falta de un marco de evaluación cuantitativo y cualitativo de un conjunto de características [13], por lo que algunos estudios han realizado análisis de relevancia para el conjunto de características seleccionado ([31], [6]).

La selección manual de características relevantes es una tarea retadora e involucra el conocimiento de los comportamientos a describir y un conjunto de datos representativo. Pocos estudios realizan un análisis de efectividad luego de la selección de sus características ([31], [9]), siendo un proceso que se ha señalado como efectivo durante la selección de características.

3.3. Aprendizaje automático

La clasificación es un proceso de aprendizaje automático que categoriza un conjunto de datos determinado en clases [11]. Dentro de los problemas de clasificación, existen dos enfoques principales, los cuales son aprendizaje supervisado y no supervisado. El aprendizaje supervisado requiere un conjunto de datos completamente etiquetado, donde estas etiquetas pueden entenderse como el resultado esperado o clase de dicho conjunto de datos [7]. El aprendizaje no supervisado busca determinar patrones, estructuras o conocimiento en datos no etiquetados y no requiere entrenamiento.

Ambos poseen ventajas y desventajas en comparación al otro. El uso de aprendizaje supervisado depende del conjunto de datos con el que se entrenó y su correcto etiquetado, de

manera que reduce su alcance para detectar nuevos patrones. Segundo, el costo de etiquetar los datos es alto y depende de conocimiento experto o colección previa de fuentes confiables, que implica un gran trabajo manual. A pesar de estas desventajas, este enfoque es el más usado en la comunidad científica, dado que suele dar mejores resultados en comparación al no supervisado en un menor tiempo [8] [13]. Además, las desventajas señaladas en cuanto a dificultad de recolección han ido disminuyendo por la gran cantidad de bases de datos disponibles públicamente y la mejora de sus métodos de extracción mediante APIs. Ambos métodos de clasificación fueron comparados en [51] y los resultados favorecen en términos de exactitud al aprendizaje supervisado.

Por las cualidades mencionadas, en nuestra investigación se usa aprendizaje automático supervisado para crear nuestro clasificador. El proceso de clasificación de URLs usando este tipo de aprendizaje incluye ciertas etapas necesarias para la predicción resumidas en la figura 3, entre ellas la preparación del conjunto de datos, la recolección de características explicadas en la sección 3.2 y la selección de algoritmos para construir el modelo de aprendizaje explicada a continuación.

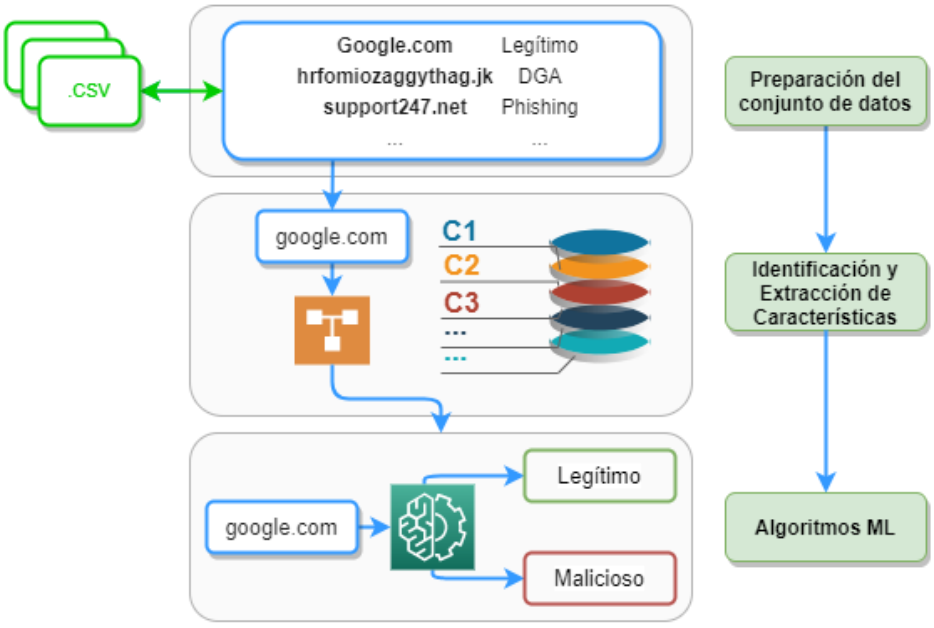

Figura 3: Etapas de aprendizaje automático

3.3.1. Clasificadores basados en árboles

Como parte de los modelos de aprendizaje supervisado existen una variedad de algoritmos que aprenden a partir de las características extraídas y modelan las relaciones entre estas entradas y la predicción final. En la sección 2 se destacaron los algoritmos basados en árboles como aquellos con mejores resultados tanto en clasificadores binarios como clasificadores de múltiples clases. Con base en esta información se seleccionaron los algoritmos de árboles de decisión (DT) y bosques aleatorios (RF) para crear nuestro modelo y entrenar el clasificador, y ambos sonn explicados a continuación.

- **Árboles de decisión:** Este tipo de algoritmo construye un árbol para categorizar datos en clases mediante un conjunto de reglas. La separación de sus nodos se basa en la información ganada anteriormente y en la entropía hasta alcanzar un determinado resultado. Usa recursividad para Estos algoritmos tienden a tener un sobreajuste (*overfitting*) cuando son entrenados en exceso. Además, pequeñas variaciones de los datos pueden generar un árbol completamente diferente [11].

- **Bosques aleatorios:** Este tipo de algoritmo es formado por muchos árboles de decisión compuestos de diferentes conjuntos de datos. El modelo final se basa en el promedio de múltiples árboles de decisión ya entrenados. En general posee un mejor rendimiento que los árboles de decisión y resuelve problemas de sobreajuste del modelo [11].

3.4. Splunk

Splunk es una solución de manejo de eventos y de seguridad de la información (SIEM) que permite monitorizar y analizar datos de distintas aplicaciones, sistemas e infraestructuras [25]. Provee una plataforma de unificación en la que se puede indexar datos y realizar búsquedas mediante su lenguaje de procesamiento de búsqueda. Más allá de las búsquedas sobre los datos indexados, permite correlacionar información para facilitar el monitoreo y mejorar la visibilidad operacional. En la figura 4 se muestran beneficios de la herramienta alrededor de una variedad de negocios, aplicaciones y tipos de monitoreo de infraestructura.

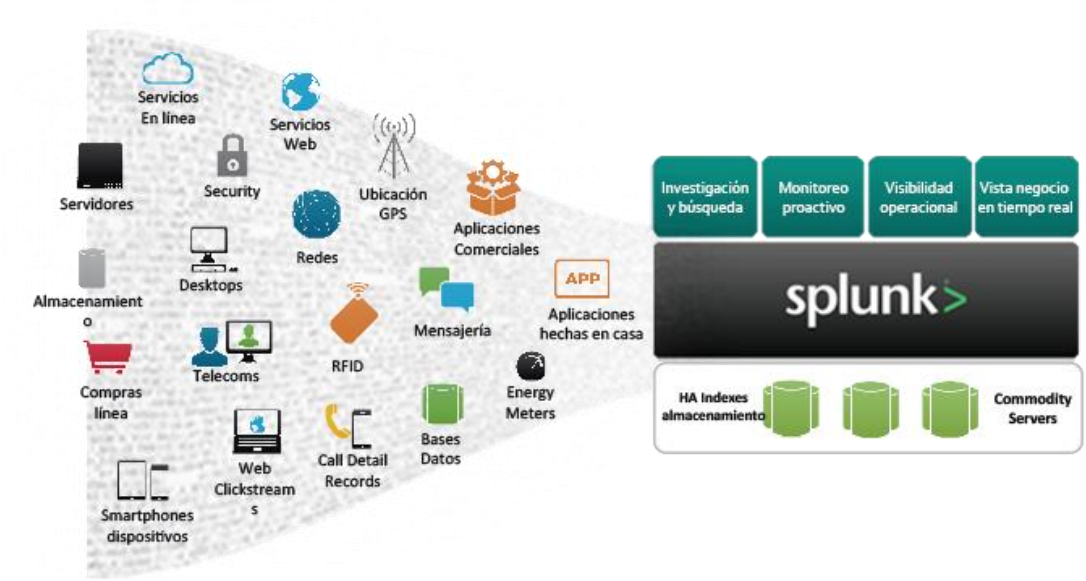

Figura 4: Beneficios de Splunk. Fuente: Cyberline [52].

El uso de Splunk en problemas de clasificación en la literatura se ha usado mayoritariamente para la extracción de características y el uso de sus visualizaciones. Jaworski [53] usó Splunk y su módulo de URL Toolbox (UT) en la detección de *DNS tunneling* para extraer diferentes características del dominio, como su entropía mediante los comandos de *ut_parse*, *ut_domain* y *ut_shannon*. También usa la función *geostats* para agregar características geográficas basadas en la ubicación de la IP. Por otra parte, Su et al. [23] lo usan para identificar ataques de denegación de servicio al determinar si el tráfico pertenece o no a un ataque según su modo de conexión y el número total de paquetes. En este estudio el uso de Splunk es acotado a sus visualizaciones y a sus comandos de conteo y desarrollo de estadísticas. Ninguno de los estudios existentes usa las capacidades de aprendizaje automático otorgadas por Splunk posterior a la extracción de características. La aproximación más cercana al tema se encuentra en un caso de estudio para la detección de familias de DGA publicado en un reporte técnico por la misma compañía Splunk [54], en el que describe la extensión de las capacidades de la plataforma usando diferentes algoritmos de clasificación y comparándolos. A continuación, explicamos más a fondo el lenguaje de procesamiento de Splunk y su módulo para realizar aprendizaje automático.

3.4.1. Search Processing Language (SPL)

El lenguaje de procesamiento de búsqueda de Splunk (SPL) abarca todos los comandos de búsqueda y sus funciones, argumentos y clausulas. Su sintaxis es basada en el lenguaje de Unix y SQL y pueden realizarse filtros, búsquedas, insertar, modificar y eliminar datos [55]. En esta investigación se hacen uso de algunos términos para referirse a la información indexada que vemos importante describir y se muestran señalados en la figura 5 usando un ejemplo de una búsqueda.

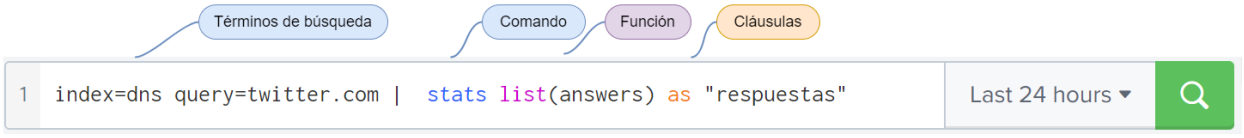

Figura 5: Lenguaje de Procesamiento de Splunk

Una *consulta* consiste en un programa escrito en SPL, consistiendo en etapas separadas por el símbolo "|" llamado "*pipe*" el cual indica que los resultados de la búsqueda serán procesados por el componente anterior. En el ejemplo, los comandos luego de este símbolo serán aplicados a los resultados filtrados luego de buscar la información en el índice con datos de DNS y específicamente que tengan como dominio a twitter.com. Un *comando* le especifica a Splunk que tipo de acción debe ser aplicada sobre los datos. Una *función* explica el cómo se agruparán, visualizarán o procesarán los datos a los que se aplica el comando. Finalmente, una *cláusula* especifica cómo se definen o agrupan los resultados de la búsqueda.

3.4.2. Machine Learning Toolkit (MLTK)

Para nuestra investigación, usamos un módulo dentro Splunk llamado Splunk Machine Learning Toolkit (MLTK). MLTK permite realizar aprendizaje automático a partir de la información indexada en la plataforma. Por lo que al alimentarlo con las diferentes clases de URLs maliciosas y legitimas se pueden aplicar operaciones estadísticas para convertir la información de forma textual a numérica, obteniendo así las características necesarias para aplicar aprendizaje automático dentro de la plataforma. Este módulo incluye modelos de clasificación basados en las librerías de sci-kit, NumPy y Statsmodels, lo cual facilita la aplicación de algoritmos de

aprendizaje removiendo las barreras de programación. También facilita la replicación de los resultados para otros investigadores pues el modelo puede exportarse y compartirse listo para ser usado en otra instancia de Splunk con otros datos.

En esta investigación se usan una variedad de comandos y funciones de SPL para la construcción del clasificador, pero el módulo de MLTK trae consigo comandos más especializados para el uso de aprendizaje automático. En la figura 6 observamos la interacción de los comandos *fit* y *apply*, los cuales son los comandos más importantes para el desarrollo de aprendizaje automático. Estos comandos entrenan y ajustan el modelo de aprendizaje basado en el algoritmo seleccionado. La funcionalidad de estos comandos es clave para la creación de nuestro clasificador y extraer características por lo que se describen a continuación en el cuadro 2.

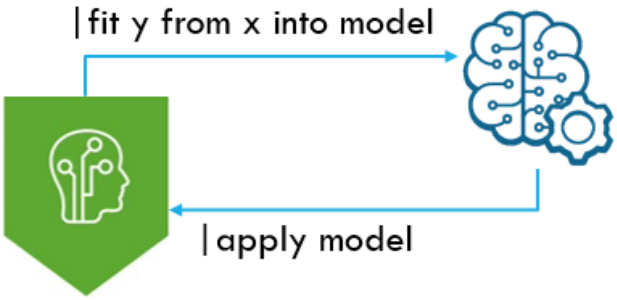

Figura 6: Flujo de Splunk MLTK

Cuadro 2*:* Comandos de Splunk

Comando	**Definición**
fit	Ajusta y aplica un modelo de aprendizaje automático a los resultados de búsqueda. La sintaxis es la misma para el aprendizaje supervisado (datos etiquetados) y no supervisado (datos no etiquetados).
apply	Calcula predicciones para los resultados de búsqueda actuales en función de un modelo que fue aprendido por el comando fit.
sample	Muestra algunos eventos aleatoriamente del conjunto de datos total.
analyzefields	Como comando de informes, analyzefields consume todos los resultados de entrada y genera una fila para cada campo numérico en los resultados de salida. Los valores en esa fila indican el desempeño del comando analyzefields al predecir el valor de un classfield. Para cada evento, si la distribución condicional del campo numérico con la mayor probabilidad z coincide con la clase real, el evento se cuenta como exacto. La probabilidad z más alta se basa en el campo de clase.
stats	Calcula estadísticas agregadas, como por ejemplo obtener un promedio, contar y sumar, sobre el conjunto de resultados.

Comando	Definición
lookup	Son tablas de datos que enriquecen los resultados de una búsqueda al ser utilizados en consultas. Tienen combinaciones campo-valor. Para fines de esta investigación nuestros lookups están basados en archivos CSV. Pueden existir input y output lookups en el que los primeros son usados para buscar valores de la tabla y el segundo es para escribir valores en la tabla.
fields	Permite escoger cuales columnas desplegar por cada uno de los eventos
eval	Calcula una expresión y coloca el valor resultante en un campo de resultados de búsqueda.
rex	Permite realizar expresiones regulares
´macro´	Un *macro* es un objeto de Splunk que representa de manera equivalente una expresión de SPL y es una forma de reutilizar código SPL al hacer una referencia a donde ese código vive. Estos objetos son referenciados al ser llamados entre comillas como por ejemplo *`ut_countset(domain, set)`*.
spath	El comando *spath* permite transformar el valor de campos con información estructurada (como lo es el formato JSON) para convertirlo en columna y filas por cada campo-valor.

Capítulo 4: Metodología

A continuación, se detalla la metodología que se siguió en este trabajo. Primero se realizó una revisión de literatura, descrita en la sección 4.1. En la sección 4.2 se detallan los pasos para implementar el clasificador. Y concluimos con la metodología creada para la evaluación del modelo en la sección 4.3.

4.1. Revisión de literatura (Objetivo Específico 1)

A partir de los trabajos de clasificación descritos en la sección 2 se realizó una revisión de literatura para determinar el estado actual de conocimiento durante el proceso de aprendizaje de un clasificador de URLs maliciosas. Se recopilaron los diversos enfoques usados durante las etapas de aprendizaje automático incluyendo la selección de los datos, la extracción de características, la construcción del clasificador y la evaluación del modelo. Para cada uno de los artículos se recolectaron los siguientes datos:

- El alcance de la predicción, en cuanto a tipos de categorías maliciosas involucradas en su conjunto de datos.
- La cantidad de características y su tipo ya sea estático o dinámico según el perfilado realizado por Cheng et al. [12] y descrito en la sección 2.
- El algoritmo con mayor exactitud en caso de compararse con otros y su resultado de predicción.

Posterior a la recolección mencionada, se realizó un análisis para cada una de las siguientes variables:

- Alcance de clases maliciosas: Describiendo la variedad de tipos de comportamientos maliciosos definidos como meta para su clasificador.
- Análisis de selección de características: Describiendo la frecuencia de uso para una característica particular en la literatura y si se hizo algún análisis de relevancia posterior a la selección.

- Análisis de resultados: Resaltando el algoritmo que obtuvo mejores resultados para el alcance de la investigación.

Para la selección inicial de las características, se realizó un análisis de características estáticas usadas en los trabajos recopilados para determinar la frecuencia de uso en la literatura. Además, por lo mencionado en la sección 3.2.2 pocos estudios realizan un análisis de las características luego de su selección a pesar de las ventajas en tiempo y rendimiento que posee este proceso. Por lo que incluimos un análisis de estabilidad para descartar aquellas características que no sean relevantes en la predicción del modelo.

Finalmente, a partir de la recolección de resultados, se seleccionaron dos de los algoritmos con mejores resultados de predicción para aplicar y medir nuestro modelo e incluimos una comparación con los algoritmos disponibles en Splunk para problemas de predicción categóricos con el fin de validar la base teórica de selección de estos algoritmos.

4.2. Implementación del clasificador (Objetivo Específico 2)

El proceso para aplicar aprendizaje automático a la predicción categórica está ampliamente definido. No obstante, tomamos en consideración las etapas detalladas en revisiones de literatura existentes relacionadas a la detección de URL maliciosas [13], [11]. A continuación, describimos las etapas seguidas en la implementación de nuestro clasificador, explicamos brevemente su función y mostramos en la figura 7 un ejemplo de las entradas esperadas y las salidas del clasificador.

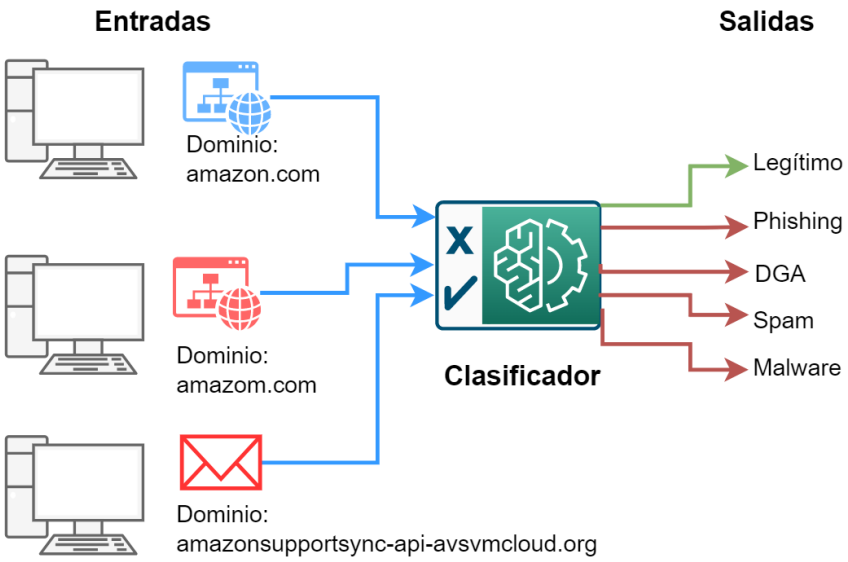

Figura 7: Entradas y salidas del clasificador

I. **Preparación:** Este paso involucró la selección de los datos, el etiquetado en sus diversas clases (malicioso y legítimo) y subclases (phishing, malware, spam y DGA).

II. **Identificación y extracción:** En este paso se recopilaron las características que describen el conjunto de datos durante el entrenamiento. También se incluyó el análisis de relevancia para estas características.

III. **Implementación:** En este paso se creó el modelo usando dos algoritmos de clasificación basados en árboles seleccionados según el análisis de literatura para problemas de clasificación multiclase.

4.2.1. Preparación del conjunto de datos

Se recolectaron URL maliciosas y legitimas a partir de tres fuentes disponibles públicamente tomadas como la base de la verdad (*ground truth*) en esta investigación y resumidas en el cuadro 3. De estas fuentes se recolectaron cuatro clases de URL maliciosas: malware, spam, phishing, DGA además de URL legitimas. Las URL de phishing, spam y benignas se tomaron del conjunto de datos desarrollado por el proyecto de inteligencia de ciber amenazas de Bell Canada en 2021 (CIC-Bell-DNS) [37]. Para las URL de malware se tomaron de la base de datos de abuse.ch URLHaus creado por el Instituto de Ciberseguridad e Ingeniería (ICE) de la Universidad de Ciencias Aplicadas de Bern en Suiza (BFH) [56] que permite descargar URL activas de malware

y reflejar así el estado actual de Internet. Finalmente, para las URL de DGA, se tomaron los indicadores de compromiso usados en el ataque de Sunburst [57] en el 2020 por ser considerado uno de los ciberataques más sofisticados en los últimos años, compilados por Splunk y FireEye para uso público [58].

Cuadro 3: Fuentes para la recolección de URLs

Clase	Subclase	Fuente	Bases de datos
Malicioso	Malware	ICE-BFH-2022	Abuse.ch URLHaus
Malicioso	Spam	CIC-Bell-DNS2021	jwspamspy
Malicioso	Phishing	CIC-Bell-DNS2021	OpenPhish, PhishTank
Malicioso	DGA	FireEye/Splunk	Sunburst IOC List
Legítimos	Legítimos	CIC-Bell-DNS2021	Majestic Million

Desbalanceo de clases

Con base en la revisión de literatura, el tipo de balanceo usado en las clases del conjunto de datos puede afectar la predicción del modelo. El tráfico legítimo supera al tráfico malicioso global de internet, por lo que el desbalanceo está intrínseco en los conjuntos de datos usados en enfoques de clasificación de sitios maliciosos [13]. Por esta razón se plantearon dos escenarios con dos distribuciones diferentes de URL maliciosas y legitimas para simular el comportamiento fluctuante de los atacantes y como se comportaría el modelo ante menor o mayor cantidad de sitios maliciosos.

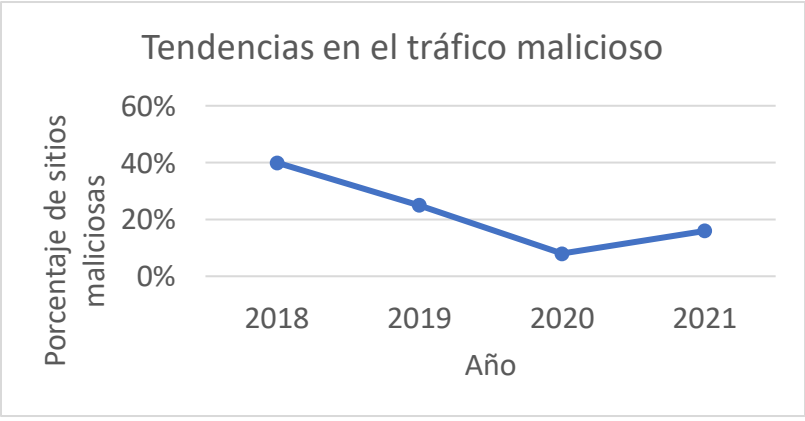

Figura 8: Análisis de riesgo anual. Fuente: Webroot y Carbonite en su reporte de Amenazas [5].

Los escenarios se construyen a partir de las tendencias del tráfico malicioso según los reportes de amenazas de los últimos 4 años publicados por Webroot [5]. Este reporte anual comprende un análisis de URL de alto riesgo basado en el procesamiento de 4.5 billones de peticiones diarias. Entre sus resultados obtuvieron que el 40% del total del tráfico analizado para el año 2018 fue malicioso, 25% para el año 2019 y 8% para el año 2020. Para este último año, la disminución en la cantidad de URL maliciosas se le atribuye a la pandemia de COVID-19. Estos números incrementaron a 16% para el año 2021. En la figura 8 se muestra la tendencia de URL de alto riesgo en los últimos cuatro años.

Con base en estos datos, para el primer escenario "A" se usó el porcentaje de URL maliciosas del año 2021, que fue de un 16%. Para el escenario "B" se promediaron las cantidades de URL maliciosas de los años 2018 y 2019, que fue de un 33%. Un resumen de estos escenarios se describe en el cuadro 4. Para ambos escenarios se incluyó un número mayor de URL de phishing dado que el 81% de todas las URLs de alto riesgo descubiertas en el 2020 fueron categorizadas como phishing, y 66% para el 2021 [5].

Cuadro 4: Distribución de datos para ambos escenarios.

	Clase/Subclase	**Legítimo**	**DGA**	**Malware**	**Phishing**	**Spam**	**Total**
Escenario A (71428 total de URL)	Legítimo	60000	0	0	0	0	**60000**
	Malicioso	0	1000	3000	5000	2428	**10428**
Escenario B (88439 total de URL)	Legítimo	60000	0	0	0	0	**60000**
	Malicioso	0	1743	8442	10000	8254	**28439**

Para el procesamiento y agrupamiento se extrajeron los conjuntos de datos en formato CSV por cada una de las categorías. Estos archivos fueron importados en Splunk y se utilizaron los comandos tipo *lookup* para leer y procesar el etiquetado y limpieza de los datos. La salida de este paso resultó en una tabla inicial con todas las URL y su respectiva clase y subclase. En las figuras 9 y 10 se observa la distribución de URL para cada clase y subclase en ambos escenarios.

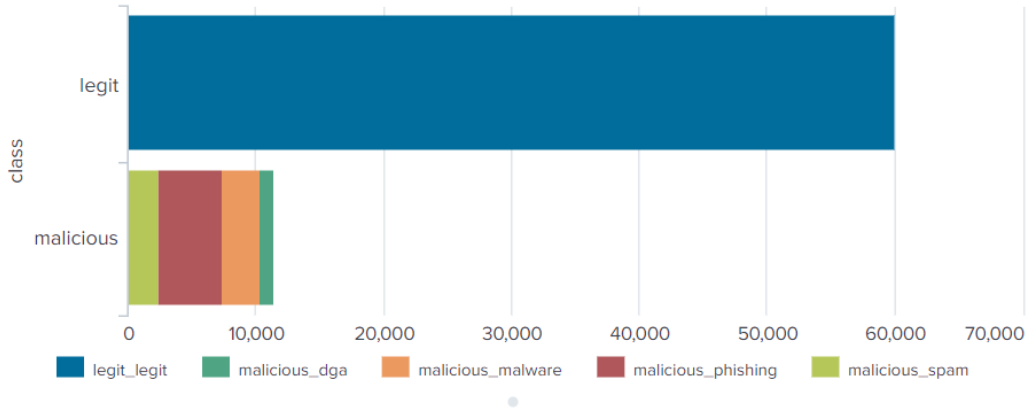

Figura 9: Distribución de datos para escenario A.

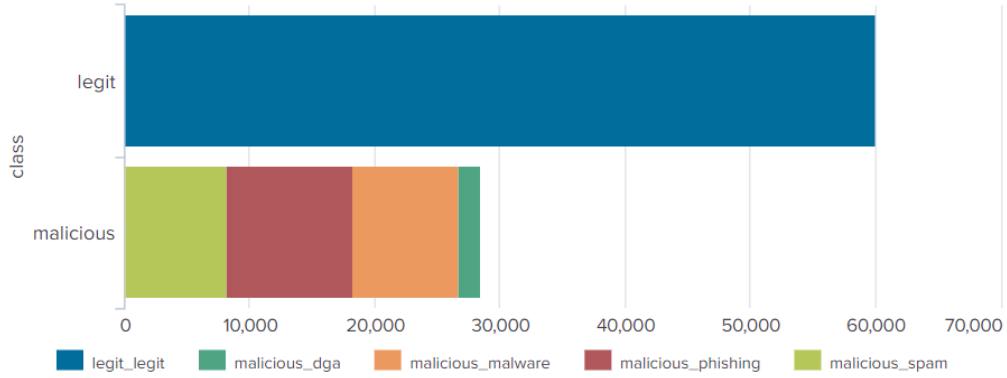

Figura 10: Distribución de datos para escenario B.

4.2.2. Identificación y extracción de características

En el estado del arte se mencionó que el proceso de selección de características es una tarea retadora por su impacto en la exactitud y robustez de la clasificación. Por ello, en la revisión de literatura se recolectaron las características estáticas usadas y se seleccionaron aquellas con mayor frecuencia de uso. De este análisis se recolectaron 12 características de tipo léxicas usando funciones de conteo y de análisis estadístico resumidas en el cuadro 5.

Cuadro 5: Selección de características

ID	Nombre	Algoritmo o función de SPL
C1	Longitud del dominio	*Len*
C2	Cantidad de puntos	*ut_countset*
C3	Cantidad de guiones	*ut_countset*
C4	Cantidad de consonantes	*Mvcount, max*
C5	Cantidad de vocales	*Mvcount, rex*
C6	Cantidad de dígitos	*Mvcount, rex*
C7	Entropía de Shannon	*ut_shannon*
C8	Distribución Bayesiana	*ut_bayesian*
C9	Significado basado en diccionario	*ut_meaning*
C10	Análisis de componente PC1	*TFIDF, PCA*
C11	Análisis de componente PC2	*TFIDF, PCA*
C12	Análisis de componente PC3	*TFIDF, PCA*

Para expandir la cantidad de información a partir de la URL se aplicó *Term Frequency Inverse Document Frequency* (TFIDF) para generar un vector de bigramas y trigramas a partir de cada URL y posteriormente se hizo un análisis de componente principal aplicando *Principal Component Analysis* (PCA) para reducir el número de dimensiones y mantener solamente los grupos más representativos del conjunto de datos. La salida de este paso resultó en una nueva matriz con valores numéricos representando cada característica, llamada la matriz de las características (*feature matrix*). Finalmente, para asegurar que las características fueran relevantes para el modelo se aplica el comando *analyzefields* para analizar la estabilidad de la relación entre los valores de la clase a predecir y los valores del conjunto de características. Este análisis permitió descartar aquellas características cuya relevancia en la predicción fue baja. En la figura 11 puede verse un diagrama de flujo de esta etapa.

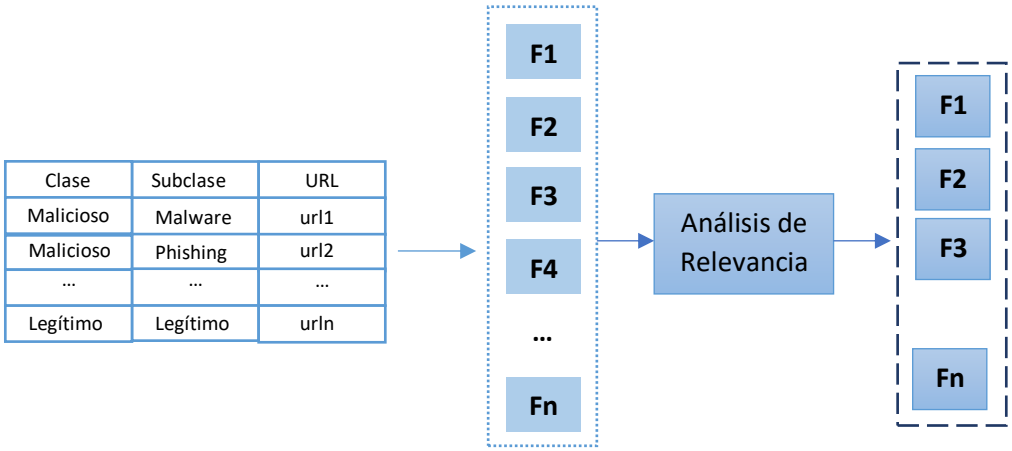

Figura 11: Flujo de extracción y análisis de características

4.2.3. Implementación del clasificador

Para la implementación del clasificador se usó la integración de MLTK disponible en Splunk, que permite predecir campos categóricos. Esta herramienta usa algoritmos de clasificación para aprender tendencias en datos que pertenecen a una categoría u otra. Se usaron comandos de SPL para ajustar (*fit*) y aplicar (*apply*) estos algoritmos de clasificación. De acuerdo con nuestra revisión de literatura, se seleccionaron dos algoritmos de clasificación que han demostrado tener una buena exactitud al aplicarse en problemas de clasificación de URL: árboles de decisión (DT) y árboles aleatorios (RF) a los dos escenarios descritos en la sección 4.2.1. La figura 12 muestra el flujo de esta etapa.

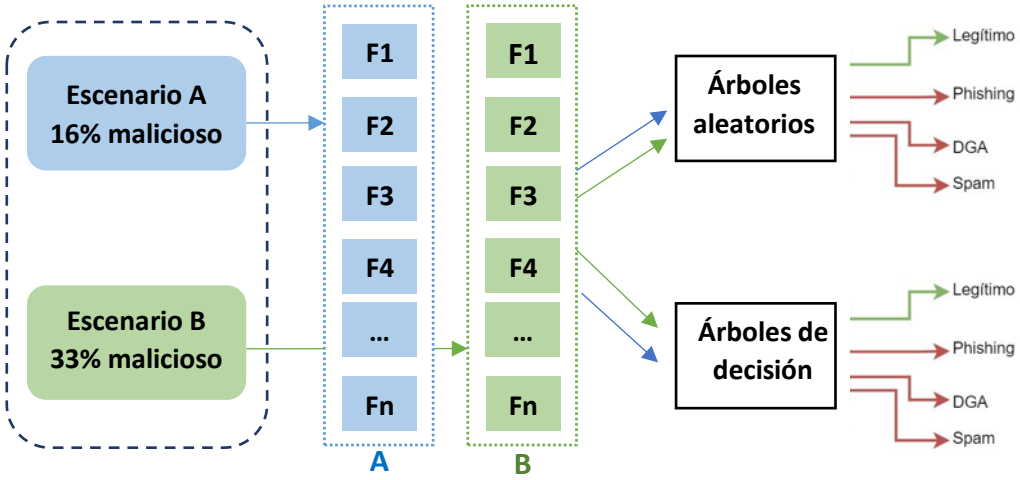

Figura 12: Flujo de aprendizaje automático

Para DT se usó Gini como criterio para medir la calidad de las divisiones (*splits*) de las características. Para el caso de RF se usaron 10 árboles en el bosque o estimadores (*n_estimators*). Ambos algoritmos se configuraron similar en cuanto a la profundidad del árbol, de manera que sus nodos se expanden hasta que la totalidad de las hojas sean puras (*max_depth*) y ambos usan todas las características (*max_features*). En el cuadro 6 se muestra un resumen de esta configuración.

Cuadro 6: Atributos de los algoritmos

DT	Criterion	Gini
	Max_depth	auto
	Max_features	auto
RF	N_estimators	10
	Max_depth	auto
	Max_features	auto

4.3. Evaluación y análisis (Objetivo Específico 3)

Para la evaluación, se midieron los resultados de las predicciones usando cuatro métricas: exactitud (*accuracy*), precisión, sensibilidad (*recall*) y F1 [22]. A pesar de que muchos estudios usan solamente la métrica de exactitud en su evaluación, en nuestro trabajo se tomó en consideración las limitaciones ya conocidas sobre la paradoja de la exactitud. Esta paradoja dice que la exactitud falla en mantener la integridad de la confianza en el resultado cuando se manejan clases desbalanceadas [59]. Por esto, es recomendable incluir métricas como F1 [13] que combina precisión y recall para mantener una mejor confiabilidad de los resultados.

La exactitud mide el porcentaje de predicciones correctas del modelo, su fórmula se representa como la relación entre las detecciones positivas y el total de observaciones realizadas. La precisión por su parte mide la calidad del modelo y su fórmula se representa como la relación entre las predicciones realizadas correctamente y el total de observaciones. El recall mide cuantas del total de las muestras fueron identificadas satisfactoriamente y su fórmula es representada por la relación entre las predicciones realizadas correctamente y las observaciones correctas en la predicción. Finalmente, F1 es el promedio de la precisión y el recall, que ayuda a comparar el rendimiento combinado de estas dos métricas [22]. Splunk permite aplicar cada una de estas métricas posterior

al entrenamiento del modelo usando las funciones incluidas en el módulo de métricas de *scikit-learn* [60]. A continuación, las ecuaciones 1, 2, 3 y 4 muestran las fórmulas de estas métricas.

$$Exactitud = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \quad (1)$$

$$Precisión = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (2)$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (3)$$

$$F1 = \frac{2 * \mathrm{Recall} * \mathrm{Precisión}}{\mathrm{Recall} + \mathrm{Precisión}} \quad (4)$$

Estas métricas están relacionadas a la clasificación errónea de las clases, que son basadas en falsos positivos, falsos negativos, verdaderos positivos y verdaderos negativos de la cual está compuesta una matriz de confusión [7]. La matriz de confusión permite la visualización del desempeño del modelo en términos de aciertos y errores luego de pasar por el proceso de aprendizaje. En esta matriz cada columna muestra el número de predicciones de cada clase, mientras que cada fila muestra el número instancias reales de cada clase. A continuación, describimos los posibles resultados de una predicción y su agrupamiento dentro de la matriz de confusión como se muestra en el cuadro 7.

Cuadro 7: Matriz de confusión

		Resultado de la clasificación	
		Negativo	**Positivo**
Clase por predecir	**Negativo**	Verdaderos Negativos	Falsos Positivos
	Positivo	Falsos Negativos	Verdaderos Positivos

- Verdadero positivo (TP): La predicción de modelo fue positiva y originalmente era positiva.
- Falso positivo (FP): La predicción de modelo fue positiva, pero originalmente era negativa.
- Verdadero negativo (TN): La predicción de modelo fue negativa y originalmente era negativa.
- Falso negativo (FN): La predicción de modelo fue negativa pero originalmente era positiva.

En esta sección se describieron los pasos para formular la base de conocimiento, implementar el clasificador y finalmente evaluarlo. En la siguiente sección mostramos los resultados obtenidos para cada uno de estos pasos.

Capítulo 5: Resultados

A continuación, se presentan los resultados obtenidos para cada uno de los objetivos de esta investigación, iniciando por la revisión de literatura, seguido de la implementación del clasificador y su evaluación.

5.1. Revisión de literatura

Para nuestra revisión se recolectaron estudios publicados en la última década desde el año 2012 al 2022 relacionados a la clasificación de dominios o URL. En el cuadro 8 presentamos un resumen de esta revisión.

Cuadro 8: Resumen de revisión de literatura

	Alcance Clases Maliciosas				***Caracteristicas***		***Resultados***	
	DGA	PH	SP	MW	Tipo	Cantidad	Algoritmo	Exactitud
Referencia	Alcance reducido							
Wang [9]	■				S	5	RF	97%
Kelley [10]	■				S	4	CLIP*	82%
Bao [25]	■				S-D	265	DT	98%
Bhoj [27]		■			S-D	3	KNN	95%
Kumar [17]		■			S-D	26	RF	98%
Le Page [28]		■			S-D	19	DT	95%
Pradeepthi [29]		■			S-D	9	DT	97%
Patil [30]		■	■		S	12	SVM	NS
Henke [31]			■		S	300	SVM	NS
Hajaj [18]				■	S-D	9	ELM	98%
Antonakakis [20]				■	S-D	41	RF	96%
Bharadwaj [32]				■	S	14	NN	89%
Yan [33]				■	S-D	8	NN	97%
Wang [34]				■	S-D	16	CB	98%
Wu [22]				■	S	9	RF	99%
	Alcance múltiple							
Mahdavifar [6]		■	■	■	S-D	35	KNN	98%
Palaniappana [15]		■	■	■	S-D	17	LR	60%
Bilge [44]	■	■	■	■	S-D	15	DT	98%
Cersosimo [38]	■	■	■	■	S	12	RF	88%
Este trabajo	■	■	■	■	**S**	**9**	**RF**	**98%**

NS: No especifica o utiliza otra métrica. **Tipo:** S (Estáticas), D (Dinámicas)
Clases: DGA (botnet), PH (Phishing), SP (Spam), MW (Malware)
Algoritmos: Random Forest (RF), Decision Tree (DT), K-Nearest Neighbors (KNN), Logistic Regression (LR), CatBoost (CB), Extreme Learning Machine (ELM), Neural Networks (NN), Support-vector machines (SVM), algoritmo nombrado por autor (*)

Para cada artículo se recolectó el alcance de su predicción con respecto a los comportamientos maliciosos principales: DGA, Phishing, Spam y Malware. A su vez, se recolectaron los enfoques de selección de características para dividirlas en estáticas (S) y dinámicas (D) o el uso de una combinación de ambas (S-D) con base a lo descrito en la sección 2. Finalmente, según los resultados obtenidos se recolectó el algoritmo con mayor exactitud (en caso de utilizar varios algoritmos) y su resultado. A continuación, realizamos un análisis para cada una de estas variables y concluimos esta sección con un resumen de lo encontrado en esta revisión.

I. <u>Análisis de alcance de predicción (único vs múltiple)</u>

Según el resumen del cuadro 8, la mayoría de los estudios (83% del total de los estudios revisados), se enfocan en la detección de un solo tipo de URL maliciosa. Hay una mayor inclinación hacia la detección de familias de DGA, phishing y malware. Sin embargo, en la clase de malware, varios estudios generalizan e incluyen fuentes que no necesariamente son categorizados como malware según nuestra definición de clase pero que se consideran maliciosos. Por ejemplo, combinan información de bases de datos de listas negras con sitios de phishing y spam bajo la misma clase de malware. Por otra parte solo 3 de los estudios (17% del total de estudios revisados) hicieron una separación formal en tres o más categorías, de los cuales solo Bilge et al [14] posee un conjunto de datos más variado al usar 10 categorías incluyendo spam, contenido adulto, malware, phishing y los dominios de la botnet *conficker* para representar la categoría de botnet y otras. A pesar de esto, en este estudio no mencionan el razonamiento detrás de la selección de sus categorías y se usan una combinación de características estáticas y dinámicas.

II. <u>Análisis de selección de características</u>

La mayoría de los estudios utilizan una selección de características mixto (estáticas y dinámicas). Solo seis de los estudios usan características estáticas, y de este grupo, el alcance de sus datos ha sido en su mayoría una única categoría de URL maliciosa y máximo 2 en Patil et al [30] enfocándose en phishing y spam. Para entender la variedad en el uso de características estáticas, fueron recolectaron y agrupadas en el cuadro 8. Entre las características más frecuentemente usadas resaltan la longitud de la hilera, la entropía, la cantidad de dígitos, cantidad de vocales y consonantes, el análisis de frecuencia mediante n-gramas y la cantidad de caracteres especiales como puntos, guiones y arrobas. Otro

descubrimiento interesante se observa en la frecuencia de las características basadas en listas (sean estas listas basadas en diccionario o en palabras sensibles o en listas de bloqueo).

Cuadro 9: Uso de características estáticas en la literatura

Característica	**Trabajos que usan esta característica**
Longitud	[18] [25] [32] [10] [17] [28] [6] [61] [34] [9] [22] [33] [30]
Cantidad de dígitos	[14] [32] [10] [28] [15] [11] [34] [22] [33]
Cantidad de caracteres especiales (. – @)	[32] [17] [28] [15] [29] [11] [34] [30]
Cantidad de vocales y consonantes	[32] [10] [6] [11] [34] [22]
Entropía	[18] [25] [10] [6] [9]
Presencia de palabras en listas	[17] [28] [15] [29] [33] [61]
N-gramas	[25] [6] [9] [22] [20]
Longitud de la palabra más larga	[14] [28] [6] [22]
Usa la IP en lugar del dominio	[32] [17] [29] [30]
Número y longitud de subdominios	[28] [22] [30]
Longitud del path o de los parámetros	[32] [17] [11]
TLD o SLD	[6] [29] [20]
Extensiones particulares	[32] [11]
Distancia de Edit	[28] [6]
Caracteristicas de argumentos y tokens	[17] [30]
Detección de ofuscación	[6]
Teorema Bayes	[61]
Esquema o protocolo	[11]
Servicio de acortado	[30]
Información externa de terceros (edad,rank)	[30]

Pocos estudios seleccionan sus características tomando en consideración la relevancia individual de cada una hacia la predicción final. En nuestra investigación, se usa primero los resultados de la revisión de características en el cuadro 8 para hacer una selección inicial para luego hacer un análisis de relevancia explicado en detalle en la sección 5.2.2 con el fin de reducir el espacio de características usado y acelerar el procesamiento del algoritmo. Por otra parte, se encontraron ciertos patrones de características usados comúnmente para diferenciar categorías maliciosas específicas. Por ejemplo, las características de longitud y entropía han sido utilizadas para distinguir entre dominios DGA y dominios legítimos [25], [10]. Además, las técnicas de distribución de caracteres como los n-gramas han demostrado ser buenas en la detección de URLs con mayor aleatoriedad [9]. Por su parte aquellas

características basadas en listas y tratan de capturar el nivel de significancia para un humano han sido diferenciadoras de sitios de phishing y malware [28].

III. **Análisis de técnica y resultados**

En cuanto a la implementación, se recolectó el algoritmo con mejor desempeño para cada uno de los artículos y los resultados muestran que los algoritmos basados en árboles desempeñan mejor en la clasificación de URLs maliciosas y legitimas para una variedad de categorías maliciosas. Según el cuadro 8, los bosques aleatorios y árboles de decisión presentan los mejores resultados comparados al resto de los algoritmos. Por esto, se seleccionaron ambos algoritmos para implementar los modelos utilizados en nuestra clasificación, descritos en la sección 5.2.3.

De este desglose, la mayoría de los trabajos publicados hasta el momento se limitan a un alcance de predicción único. Los escenarios enfocados en detección de una sola categoría pese a que poseen buenos resultados de predicción, no representan un escenario realista al detectar solo uno de los posibles comportamientos de un atacante. Por su parte, aquellos estudios con alcance de predicción múltiple y que detectan varias categorías pueden mejorar sus controles con base al tipo de categoría detectado ([14], [6]).

Con respecto a la recolección de características, se encontró que los estudios en el área usan una combinación de características estáticas y dinámicas para el aprendizaje de sus modelos y en su mayoría no usan ningún marco de referencia para guiar su selección ni plantean un análisis de relevancia posterior a la selección de características. En nuestro trabajo se seleccionaron únicamente características léxicas al poder recopilárseles en un menor tiempo pensando en la operacionalización del sistema. Además, al seleccionar características estáticas se reduce la cantidad de información recopilada al no tener que almacenar atributos dentro de una ventana de tiempo ni depender de información externa o histórica.

En resumen, a pesar del gran repertorio de estudios en el área, ninguno de los estudios revisados ha usado únicamente características estáticas para clasificar escenarios de categorías múltiples. Esta es una de las principales contribuciones en nuestro trabajo. Además, dada la importancia de una apropiada selección de las características en el rendimiento del clasificador se incluye un análisis de relevancia para identificar características estables que representen el conjunto de datos apropiadamente y remover aquellas que aporten ruido. Finalmente notamos que los estudios más

recientes guían la distribución o el balance de sus clases según el panorama actual de Internet y con base a este seleccionan la cantidad de datos maliciosos y legítimos para usar en su modelo [6]. Por esto, nuestro trabajo considera la variabilidad en la distribución del conjunto de datos al comparar dos escenarios con tasas distintas de sitios maliciosos y legítimos.

5.2. Implementación del clasificador

A continuación, presentamos los pasos realizados para implementar el clasificador. Se describen la preparación del conjunto de datos en la sección 5.2.1, la extracción de las características en la sección 5.2.2 y la implementación de los modelos en la sección 5.2.3. Se resumen estas tres etapas en la figura 13.

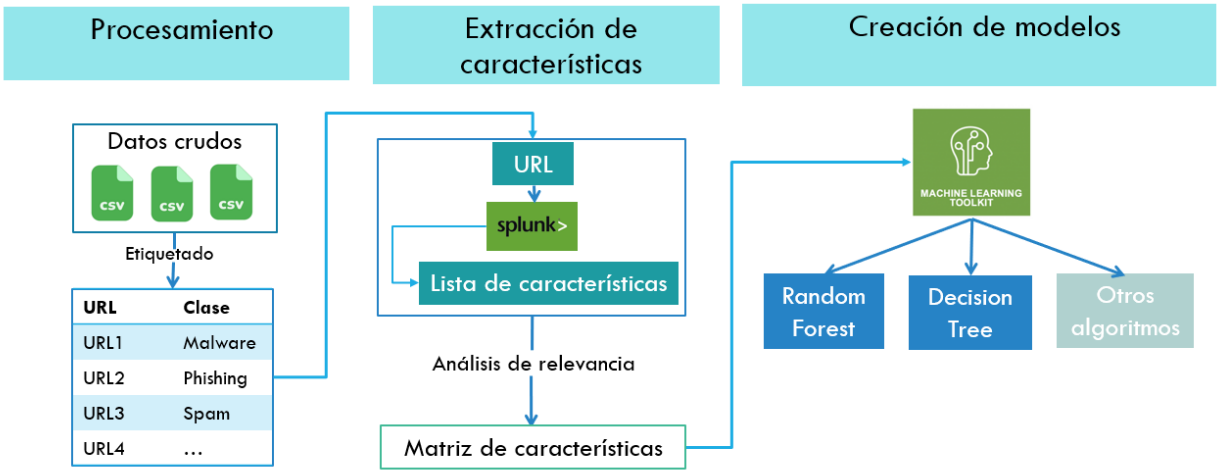

Figura 13: Resumen de la implementación de los modelos

La etapa de procesamiento involucra el agrupamiento y el etiquetado de los datos crudos. Una vez que a cada URL se le asignó una etiqueta entre las 5 clases posibles: malware, phishing, spam, dga y legítimo, se procede a extraer mediante Splunk y su lenguaje de procesamiento atributos de cada URL para generar una lista de características para cada URL. Estas características son luego evaluadas en un análisis de relevancia para descartar aquellas variables que producen ruido a la predicción final. Una vez obtenidas las características relevantes, usamos el módulo de Splunk MLTK para crear y entrenar los modelos a partir de la matriz de características conformadas por cada URL y su lista de características. Para el entrenamiento se seleccionaron dos algoritmos que han demostrado tener buenos resultados en problemas de clasificación similares según la revisión

de literatura: bosques aleatorios y árboles de decisión. Además, Splunk cuenta con más algoritmos incluidos en el módulo que son fácilmente aplicables una vez obtenida la matriz de características y se muestran algunas de sus otras aplicaciones con el fin de confirmar lo encontrado en la literatura pero la descripción de los demás algoritmos queda fuera del alcance de este trabajo.

5.2.1. Preparación del conjunto de datos

A partir de las fuentes de datos descritos en la sección 4.2.1, se obtuvieron 5 archivos CSV para cada clase de URLs: benignos, malware, phishing, DGA y spam. Estos fueron importados en Splunk como un objeto tipo *lookup* que consiste en una tabla de formato campo-valor con la que es posible leer la tabla con el comando *inputlookup* y escribir usando el comando *outputlookup*. Por ejemplo, en la figura 13 se muestra cómo usar el comando *inputlookup* para buscar los contenidos de esta tabla y mostrar solo los primeros 5 resultados con el comando *head*, en este caso para el lookup de DGA.

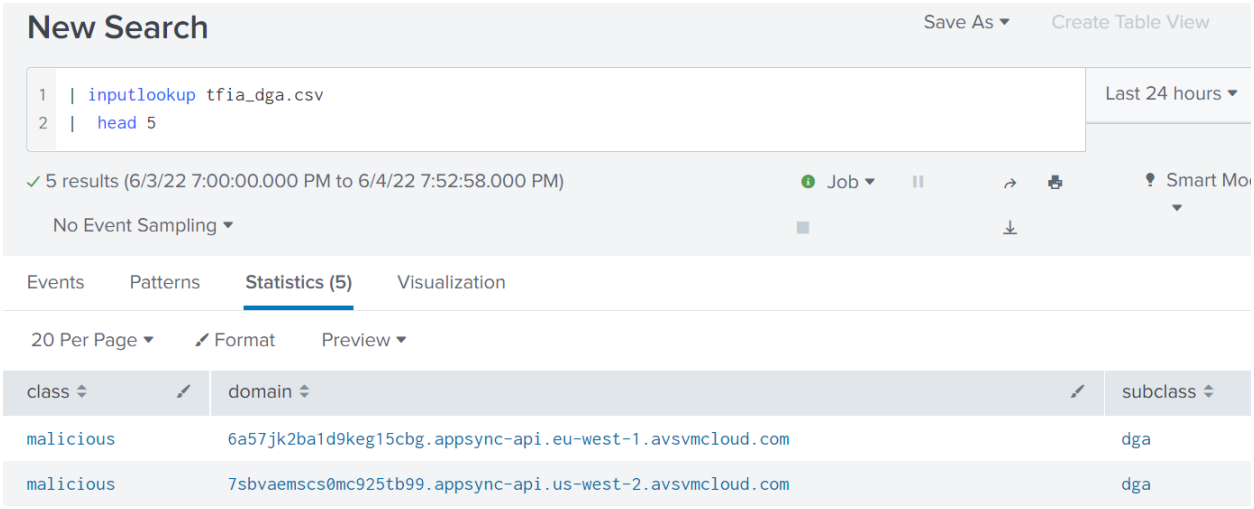

Figura 14: Distribución de datos para escenario A.

Luego de su indexación se procedió a asignar las etiquetas usando el comando *fillnull* para agregar la clase y subclase para cada URL en el conjunto de datos. En el cuadro 10 se observa parte del código SPL usado en el etiquetado para los dominios legítimos y de phishing y se puede encontrar el código completo en el cuadro A1 en la sección de anexos. Además, se usó el comando *outputlookup* para escribir sobre la misma tabla las nuevas etiquetas. Para el agrupamiento se usa la función *append*, con la que es posible incluir los datos de un lookup a otro y generar nuestra

primera matriz con todas las URLs del conjunto de datos. En el cuadro 11 se muestra parte del código usado para el agrupamiento. Finalmente, en la figura 14 se puede ver una muestra de los datos luego de su indexado y agrupamiento.

Cuadro 10: Comandos de Splunk utilizados para completar el etiquetado

```
| inputlookup tfia_legit.csv
| fillnull value="legit"
| outputlookup tfia_legit.csv
...
| inputlookup tfia_phishing.csv
| fillnull class value="malicious"
| fillnull subclass value="phishing"
| outputlookup tfia_phishing.csv
```

Cuadro 11: Comandos de Splunk utilizados para hacer el agrupamiento

```
| inputlookup tfia_phishing.csv
| inputlookup append=true tfia_legit.csv
| inputlookup append=true tfia_malware.csv
...
| outputlookup scenario_A_dataset.csv
```

class	subclass	domain
malicious	dga	2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com
malicious	spam	3-8h7eo0.jp
malicious	malware	http://219.157.215.122:56889/Mozi.m
malicious	phishing	http://www.tbook.top/
legit	legit	www.whirlpool.com

Figura 15: Muestra de datos para cada categoría

5.2.2. Identificación y extracción de las características

En esta sección se describe el resultado de la extracción de características seleccionadas a partir de la revisión de literatura realizada en la sección 4.2.2. En la figura 16 se muestra un resumen del proceso de extracción de características y un ejemplo de extracción para la hilera “twitter.com”. Para la extracción de las primeras seis características se usan solo funcionalidades básicas de Splunk como funciones de conteo y expresiones regulares. Para las características estadísticas se usó URL Toolbox, un módulo externo disponible dentro de Splunk que automatiza varias

funciones matemáticas. Finalmente, para expandir la cantidad de información a partir de la URL original, usamos el método estadístico TFIDF para extraer los principales n-gramas y posteriormente aplicar la técnica de reducción de dimensiones PCA para generar 3 características más con los principales componentes o conjuntos de caracteres basados en cada URL. A continuación, explicamos cada una de estas características, el código de Splunk para construirlas y ejemplos para una URL de cada clase.

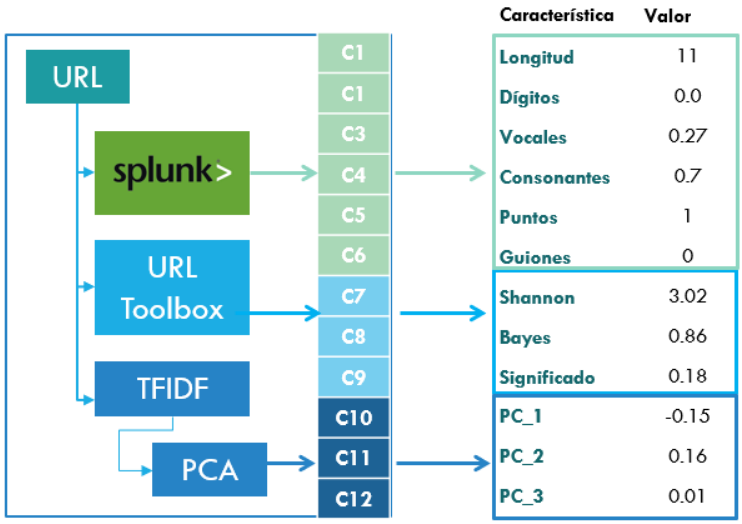

Figura 16: Resumen de etapa de extracción para twitter.com

Extracción de longitud (C1)

La longitud es una de las características más simples, pero posee mucho valor para detectar dominios autogenerados o de spam. Para extraer la longitud de la URL, se usó la función *len* para iterar por cada uno de los dominios en el lookup al especificar la columna deseada *len(domain)* y contar los caracteres de cada hilera. En la figura 17 puede verse el resultado de aplicar esta función.

ut_domain_length	domain
60	2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com
11	3-8h7eo0.jp
35	http://219.157.215.122:56889/Mozi.m
21	http://www.tbook.top/
17	www.whirlpool.com

Figura 17: Extracción de característica de longitud

Extracción de puntos y guiones (C2, C3)

Para el conteo de puntos y guiones se usó el macro *ut_countset* parte de URL Toolbox (UT), que nos permite sumar un determinado valor. Con las entradas apropiadas el macro *ut_countset* genera un nuevo conjunto campo-valor en formato JSON para la suma de las apariciones del carácter "." (punto). Para mayor claridad, se muestra la extracción de los valores en el JSON generado en la figura 18. En el cuadro 12 podemos ver el código usado para extraer esta característica. Mediante el comando spath colocamos el valor de la suma de los puntos en su propia columna para poder trabajar y renombrar el dato. El mismo proceso fue aplicado a la extracción de los guiones.

Cuadro 12: Código SPL para la extracción de puntos

```
// Extracción de Puntos
| eval set="."
| `ut_countset(domain, set)`
| spath input=ut_countset
| rename ut_countset.sum AS "ut_n_dots"
```

domain	ut_n_dots	ut_n_hyphens
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	4	3
3-8h7eo0.jp	1	1
http://219.157.215.122:56889/Mozi.m	4	0
http://www.tbook.top/	2	0
www.whirlpool.com	2	0

Figura 18: Extracción de características de puntos y guiones

Extracción de la razón para vocales, consonantes y dígitos (C4, C5, C6)

Para la extracción de vocales, consonantes y dígitos se utilizaron expresiones regulares para obtener cada grupo de caracteres usando el comando *rex*. Para los dígitos se utilizó la expresión "\d" para recolectar todos los números del 0 al 9. Y para las vocales la expresión se especificó manualmente [aeiou]. Una vez extraídos, se suman sus apariciones y se obtiene la razón para cada una al dividir la cantidad de apariciones entre la longitud total de la hilera extraída en C1. En el cuadro 13 podemos ver el código SPL para estas características y en la figura 19 la extracción de cada una de estas para las muestras anteriores.

Cuadro 13: SPL para la extracción de dígitos y vocales

```
// Extracción de Dígitos
| eval ut_digit_ratio = 0.0
| rex field=domain max_match=0 "(?<digits>\d)"
| eval ut_digit_ratio=if(isnull(digits),0.0,mvcount(digits) /
ut_domain_length)

// Extracción de Vocales
| eval ut_vowel_ratio = 0.0
| rex field=domain max_match=0 "(?<vowels>[aeiou])"
| eval ut_vowel_ratio=if(isnull(vowels),0.0,mvcount(vowels) /
ut_domain_length)

// Extracción de Consonantes
| eval ut_consonant_ratio = max(0.0, 1.000000 - ut_digit_ratio -
ut_vowel_ratio)
```

domain	ut_vowel_ratio	ut_digit_ratio	ut_consonant_ratio
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	0.21666666666666667	0.15	0.633333
3-8h7eo0.jp	0.18181818181818182	0.36363636363636365	0.454545
http://219.157.215.122:56889/Mozi.m	0.05714285714285714	0.4857142857142857	0.457143
http://www.tbook.top/	0.14285714285714285	0.0	0.9
www.whirlpool.com	0.23529411764705882	0.0	0.8

Figura 19: Extracción de características de vocales, consonantes y dígitos

Extracción de Entropía (C7)

La entropía representa la cantidad de información promedio contenida en un mensaje, representa la incertidumbre o aleatoriedad detrás de los caracteres de una hilera. URL Toolbox (UT) posee el macro *ut_shannon*, que aplica la función de la entropía a la entrada especificada. El código de la función en Python se encuentra en el Anexo A en el cuadro A2. En la figura 20 se muestra el resultado de extraer la entropía para cada URL en nuestra muestra. Esta característica refleja muy fácilmente la diferencia en aleatoriedad entre las categorías de DGA y legítima, por ejemplo, el valor con más aleatoriedad es el de DGA con un 4.6 y el menor valor de 3.10 para la URL legítima.

domain	subclass	ut_shannon
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	dga	4.612744920746109
3-8h7eo0.jp	spam	3.459431618637298
http://219.157.215.122:56889/Mozi.m	malware	3.943289445392767
http://www.tbook.top/	phishing	3.141619208183979
www.whirlpool.com	legit	3.1018812234760182

Figura 20: Extracción de característica de entropía

Extracción de Significado basado en Diccionario (C8)

Esta característica nos dice qué tanto significado poseen las palabras en una basado en términos de diccionarios y fue seleccionada para diferenciar URLs usando *typosquatting* o generadores aleatorios. Splunk posee el macro *ut_meaning* que usa un diccionario de términos "meaning.dic" compuesto por 5000 términos comunes en idioma inglés para medir si una palabra en particular tiene sentido semántico o no.

El código de la función calcula la razón entre la longitud de la hilera y la longitud de las palabras que fueron encontradas en el diccionario que lo componen. El rango definido para esta función va de 0 a 1, donde 0 representa un bajo significado semántico y un 1 representa un alto significado semántico. En la figura 21 se muestra el resultado de esta función para cada una de las muestras. Es interesante resaltar que para la URL en la categoría de spam se obtuvo un cero como valor semántico, y aquellos dominios con palabras como *book* o *pool* obtuvieron un puntaje más alto.

domain	subclass	ut_meaning_ratio
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	dga	0.25
3-8h7eo0.jp	spam	0.0
http://219.157.215.122:56889/Mozi.m	malware	0.02857142857142857
http://www.tbook.top/	phishing	0.3333333333333333
www.whirlpool.com	legit	0.35294117647058826

Figura 21: Extracción de característica de significado

Extracción de la distribución de Bayes (C9)

Para extraer esta característica basada en la distribución de bayes utilizamos el macro *ut_bayesian*. Como su nombre indica esta función está basada en el Teorema de Bayes, con la que se obtiene una probabilidad a partir de las múltiples posibilidades de una condición [61].

UT ayuda a determinar la probabilidad de que un dominio sea bueno o malo basado en dos diccionarios: "bayesian_bad.dic" y "bayesian_good.dic". El bayesian_bad.dic posee 29775 términos e incluye palabras usualmente utilizadas para estafas o falsificaciones comunes a dominios populares por ejemplo "Microsoft-update". Por su parte "bayesian_good.dic" incluye 27638 términos con términos que no son encontrados en el diccionario pero que son comunes para diversas áreas o culturas, por ejemplo "jira" o "newrelic". El rango de esta función va de 0 a 1, donde 0 representa un dominio legítimo y 1 representa un dominio malicioso. En la figura 22 podemos observar la función aplicada a nuestra muestra. Se puede apreciar que las categorías de *phishing* y *legit* poseen los puntajes más cercanos a 0 comparadas a las otras categorías.

domain	subclass	ut_bayesian
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	dga	0.9999999999998036
3-8h7eo0.jp	spam	0.9968526664271536
http://219.157.215.122:56889/Mozi.m	malware	0.9993372212497217
http://www.tbook.top/	phishing	0.8687932747681184
www.whirlpool.com	legit	0.934991943509487

Figura 22: Extracción de característica de distribución bayesiana

Extracción de componentes principales (C10, C11, C12)

Finalmente se extraen n-gramas para comparar la frecuencia entre diferentes grupos de términos que componen una URL. Esta técnica de procesamiento de lenguaje natural ha sido ampliamente utilizada en la detección de botnets [22] y fue una de las características más usadas en nuestra revisión de literatura pues se ha demostrado que ayuda a mejorar la precisión de la predicción del clasificador en varios tipos de clases maliciosas. Para proveer un ejemplo de su uso, al extraer grupos de 3 caracteres a partir de la URL podemos obtener más información sobre el TLD y detectar una posible suplantación de dominio, una técnica común en el comportamiento de phishing.

Para extraer estas características se usaron dos modelos de extracción de características disponibles en Splunk. Primero se aplicó TFIDF para encontrar la frecuencia de grupos de caracteres en una URL a través de los n-gramas o grupos de caracteres. Este algoritmo encuentra palabras y n-gramas que son comunes en un evento particular pero que son anómalos en un conjunto de eventos. Por lo que un puntaje de TFIDF alto indica que esa palabra o n-grama es potencialmente importante. Con fines ilustrativos, mostramos en el cuadro 14 la aplicación de este método para una sola URL para mostrar el paso intermedio donde se recolectan cada n-grama, que en nuestro caso se usa un rango de 2 a 3, que significa que se extraerán grupos de 2 y 3 caracteres. La URL fue redactada en texto por seguridad, pero disponible para su lectura en la figura 22. Splunk usa Scikit-Learn TfidfVectorizer [62] de fondo, para convertir una colección de valores de entrada, a una matriz con características numéricas de TFIDF. En la especificación de sus parámetros se necesitó detallar el tipo de analizador (en nuestro caso caracteres) y el rango esperado para generar los n-gramas como explicamos anteriormente, un rango con límite inferior de 2 y límite superior de 3.

Cuadro 14: SPL para generar el vector de características TFIDF.

```
| inputlookup scenario_a_dataset.csv
| search domain="https://ymtden--redactado---/pun/y/NapJRQtcu.zip"
| fit TFIDF domain analyzer=char ngram_range=2-3 into "exampleTFIDF"
| fields domain*
```

En la figura 23 se muestra la salida del código anterior, se seleccionaron 3 columnas por razones de espacio. Podemos observar que para esta URL el bigrama “/y” se repite 2 veces por lo que el valor generado para esta combinación es mayor que las de otras combinaciones. También podemos observar el trigrama “zip” contemplado en la última característica del vector. Por defecto este algoritmo agrega un prefijo al campo de salida *domain_tfidf_* a cada valor generado.

domain	domain_tfidf_10_/y	domain_tfidf_76_zi	domain_tfidf_77_zip
https://ymtdental.org/pun/y/NapJRQtcu.zip	0.2222222222222222	0.1111111111111111	0.1111111111111111

Figura 23: Extracción de característica de bigramas y trigramas

Las dimensiones del vector resultante de TFIDF puede llegar a ser muy grande si la frecuencia de los términos encontrados es muy pequeña o si existen demasiadas combinaciones. Por ello se

aplicó un algoritmo de reducción de dimensiones como PCA para mantener solo aquellas características relevantes y evitar que se degrade el rendimiento del algoritmo. Este algoritmo se encarga de mantener la esencia de los datos al seleccionar aquellos que posean más varianza o más información con respecto al resto de los datos. La meta es transformar las dimensiones en componentes principales que se ordenan según el nivel de variación que posean. Por ende, el primer componente principal (PC) va a tener la máxima varianza y así sucesivamente con el resto de los componentes extraídos. En la figura 24 se observan los tres componentes principales obtenidos luego de aplicar el modelo TFIDF y PCA.

class	subclass	domain	PC_1	PC_2	PC_3
malicious	phishing	http://www.tbook.top/	0.011532981507640197	-0.46849393234107756	0.6629514567975668
malicious	malware	http://219.157.215.122:56889/Mozi.m	-0.4687499429085151	-0.39496093854396547	0.6246723821058209
legit	legit	www.whirlpool.com	0.732110427642104	0.09267838410385304	0.20984553688693566
malicious	dga	2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	-0.16098585984263775	0.009633560168696823	-0.0161659407161289
malicious	spam	3-8h7eo0.jp	-0.11846403408790682	-0.10425951861258173	-0.07016152412472917

Figura 24: Características de análisis de frecuencia

El modelo de PCA también se basa en Scikit-Learn y Splunk agrega por defecto el prefijo “PC_” a cada nuevo componente creado. Para las propiedades del algoritmo, se especificó una cantidad de 3 componentes a extraer a partir de la aplicación del modelo. En el cuadro 15 puede verse el código SPL para la creación de los modelos utilizando nuestro conjunto de datos del escenario A mediante el comando *fit*. Una vez creadas las columnas de los 3 componentes principales seleccionados se aplican los modelos creados con el comando *apply* para guardar esa información en nuestra matriz final de características, el código para aplicar ambos modelos se observa en el cuadro 16.

Cuadro 15: Creación de los modelos TFIDF y PCA

```
| inputlookup scenario_a_dataset.csv
| fit TFIDF domain analyzer=char ngram_range=2-3 into "tfia_ngram"
| fit PCA domain_tfidf* k=3 into "tfia_pca"
| fields - domain_tfidf*
```

Cuadro 16: Aplicación de los modelos TFIDF y PCA

```
| inputlookup scenario_a_dataset.csv
| apply " tfia_ngram"
| apply " tfia_pca" | fields - domain_tfidf*
```

Matriz de características

En esta sección se describió la extracción para cada una de las características seleccionadas y se obtuvo un valor numérico según la función estadística o de conteo aplicada. En el cuadro A3 de los anexos se presenta el código completo utilizado para realizar la extracción. En la figura 25 presentamos la tabla completa de características para las cinco URL de muestra que hemos usado a lo largo de esta sección, redondeando los valores a 3 decimales por razones de espacio.

URL	Clase	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
http://219.157.215.122:56889/Mozi.m	malware	35.000	4.000	0.000	0.457	0.057	0.486	3.943	0.999	0.029	-0.469	-0.395	0.625
http://www.tbook.top/	phishing	21.000	2.000	0.000	0.900	0.143	0.000	3.142	0.869	0.333	0.012	-0.468	0.663
3-8h7eo0.jp	spam	11.000	1.000	1.000	0.455	0.182	0.364	3.459	0.997	0.000	-0.118	-0.104	-0.070
2ur7sef75htrb36g4mtirh7.appsync-api.us-east-2.avsvmcloud.com	dga	60.000	4.000	3.000	0.633	0.217	0.150	4.613	1.000	0.250	-0.161	0.010	-0.016
www.whirlpool.com	legit	17.000	2.000	0.000	0.800	0.235	0.000	3.102	0.935	0.353	0.732	0.093	0.210

Figura 25: Matriz de características

Se utiliza un esquema de color para separar las características basadas en conteo y funciones matemáticas (azul para las columnas de C1 a C9) de las recolectadas del análisis de componente principal (verde para las columnas de C10 a C12). De los valores obtenidos para cada característica se aplica también una escala de color representando aquellos con valores más altos con el color más oscuro y aquellos valores más bajos con el color más claro. Por ejemplo, para C3 (cantidad de guiones) se tiene que la URL asociada a la subclase de DGA posee la mayor cantidad de guiones con un total de 3, seguido por spam con un total de 1 y para el resto no se encontraron guiones en sus hileras.

Una vez obtenidas las características, se realizó un análisis de relevancia para una muestra aleatoria de los datos utilizando el comando *analyzefields*. Este comando determina la habilidad de predecir correctamente la subclase a partir de nuestra selección de características, obteniendo tanto la exactitud como la exactitud balanceada. La primera es dada por la relación entre las muestras clasificadas correctamente y el número total de muestras. Y la segunda representa el promedio de todos los valores de exactitud obtenidos. En la figura 26 podemos observar estas métricas para cada una de las características y en el cuadro 17 el código SPL utilizado para generar el resumen de exactitud y exactitud balanceada.

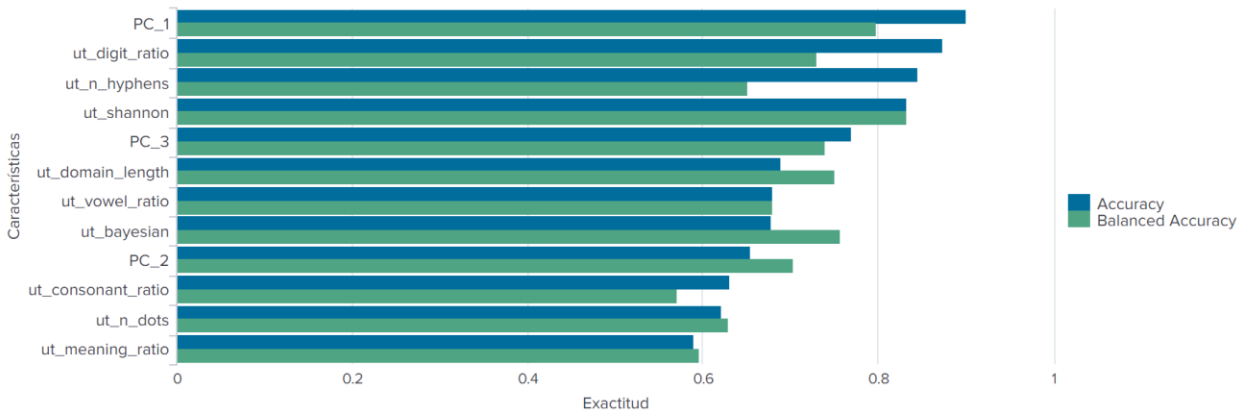

Figura 26: Análisis de características según su exactitud y exactitud balanceada

Cuadro 17: Código para analizar la exactitud de características

```
| inputlookup scenario_a_dataset_features.csv
| sample 1000 by subclass
| analyzefields classfield=subclass
| rename acc as “Accuracy”
| rename field as “Feature”
| rename balacc as "Balanced Accuracy"
| sort - Accuracy
```

Según los resultados del análisis, las características con mayor exactitud son el primer componente principal con un 0.90 de exactitud, la cantidad de dígitos con un 0.87 de exactitud y la cantidad de guiones con un 0.84 de exactitud. Las características con mayor exactitud balanceada son la entropía de Shannon con un 0.83, seguido del primer componente principal con un 0.79 y finalmente la característica de longitud y bayes empatados con un 0.75. Entre las características menos relevantes para el modelo se encuentran la cantidad de consonantes con un 0.63, el número de puntos con un 0.62 y la característica basada en el significado con 0.58, las cuales fueron descartadas para generar una nueva matriz de características solo con características relevantes.

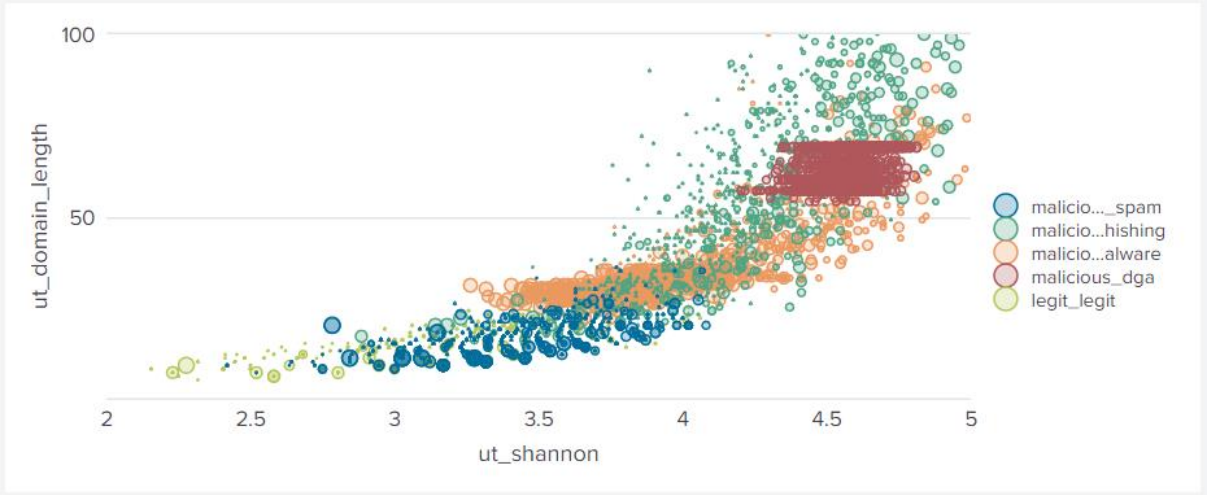

Figura 27: Ejemplo de distribución de entropía y longitud

Con este tipo de análisis de relevancia podemos también crear visualizaciones entre aquellas características con mejores valores de predicción, por ejemplo, en la figura 27 realizamos un diagrama de dispersión utilizando la longitud y la entropía. En esta figura se observa que aquellas categorías con una entropía mayor son phishing, DGA y malware. En cambio, las categorías de URL legitimas y de spam presentan menor entropía y son normalmente conformadas por hileras más cortas. Las categorías de spam y legitimas tienen cierta superposición entre ellas, similar a lo que pasa entre las categorías de phishing y malware.

Para poder evaluar el impacto del análisis de relevancia implementado, se generaron modelos tanto para las 9 características más relevantes según el análisis realizado en la sección anterior como para la totalidad de las características. Se seleccionaron las características con exactitud mayor al 65%, descartando así la cantidad de consonantes, el número de puntos y la tasa del significado cuyos resultados obtuvieron la menor probabilidad de acertar la predicción.

5.2.3. Implementación del clasificador

Para la implementación del clasificador, se usó Splunk MLTK y su asistente de predicción para valores categóricos según lo estipulado en la metodología. Este módulo nos permite construir, entrenar y probar modelos a partir de la tabla de características construida en la sección anterior. Cuenta con una biblioteca de algoritmos de aprendizaje aptos para problemas de predicción categórica incluidos los usados en esta investigación: árboles aleatorios y árboles de decisión,

cuyos resultados sobresalieron en el análisis de literatura realizado en la sección 5.1. Además, con el fin de conocer el impacto del análisis de relevancia en las características seleccionadas, se generaron dos modelos por cada algoritmo usando primero las 9 características tomadas del análisis de relevancia y segundo usando todas las características formuladas inicialmente. Un resumen de la elaboración de los modelos para cada grupo de características puede verse en la figura 28.

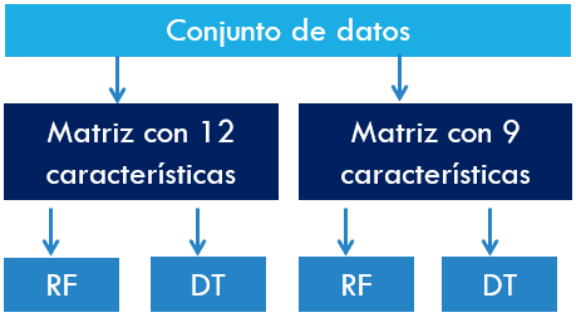

Figura 28: Resumen de modelos según sus características

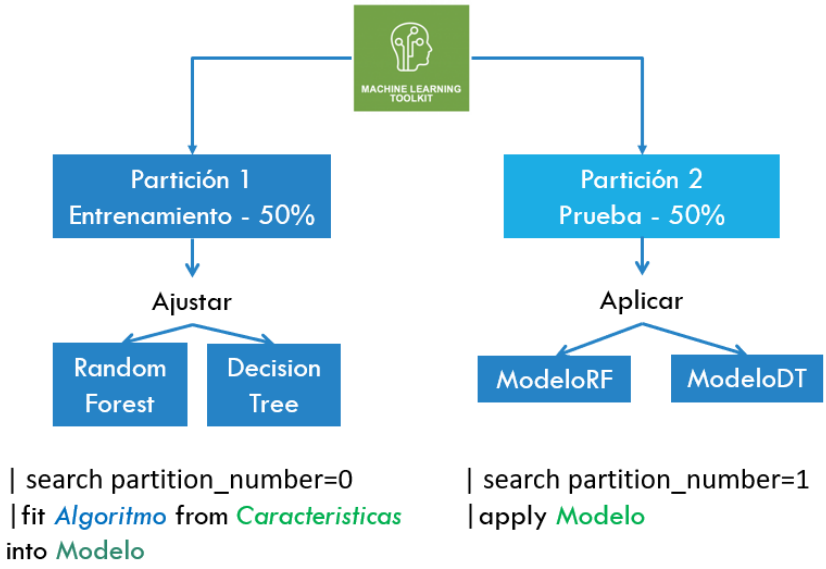

Figura 29: Resumen de implementación y aplicación de MLTK

Se usaron el 50% de los eventos como datos de entrenamiento y el otro 50% como datos de prueba. Los eventos de cada grupo son seleccionados aleatoriamente por Splunk MLTK al agregar

una etiqueta a cada fila de la matriz representando la partición a la que pertenece y utiliza esta etiqueta para entrenar los datos con la primera partición y validar el modelo con la segunda. Con los datos de entrenamiento se aplica el comando de ajuste (*fit*) con el algoritmo de aprendizaje sobre el campo por predecir (en nuestro caso la subclase) a partir de la lista de características. Posteriormente, se usó el comando *apply* para aplicar el modelo aprendido y retornar la tabla de características con el respectivo resultado de la predicción. El modelo resultante luego de aplicar el comando *fit* se guarda en un objeto similar a una tabla, que incluye la especificación inicial del modelo y sus atributos. En la figura 29 se puede ver un resumen del uso de los comandos fit y apply.

Con el fin de organizar las pruebas, se usó una convención para nombrar el modelo de modo que incluya los atributos con los que fueron creados. Por ejemplo: "`scenario_A_model_RF_9F_5050_Psubclass`" primero incluye a cuál escenario pertenece, luego que algoritmo se usó (Random Forest (RF)), cuantas características se incluyeron (9 características (9F)), la división del conjunto de datos de entrenamiento y prueba 50/50 (5050) y el campo a predecir, que para nuestro caso es la subclase (Psubclass). En los cuadros 18 y 19 se observa el código usado para ajustar y aplicar uno de los modelos.

Cuadro 18: Código SPL para ajustar el modelo

```
| inputlookup scenario_A_dataset_features.csv
| search partition_number=0
| fit RandomForestClassifier "subclass"
from "PC_1" "PC_2" "PC_3" "ut_bayesian" "ut_digit_ratio"
"ut_domain_length" "ut_n_hyphens" "ut_shannon" "ut_vowel_ratio" into
"scenario_A_model_RF_9F_5050_Psubclass"
```

Cuadro 19: Código SPL para aplicar el modelo

```
| inputlookup scenario_A_dataset_features.csv
| search partition_number=1
| apply "scenario_A_model_RF_9F_5050_Psubclass"
| table "subclass", "predicted(subclass)", "PC_1" "PC_2" "PC_3"
"ut_bayesian" "ut_digit_ratio" "ut_domain_length" "ut_n_hyphens"
"ut_shannon" "ut_vowel_ratio"
```

Se realizó este mismo proceso con el algoritmo de árboles de decisión. Y se repitieron los pasos para el escenario B. En la siguiente sección mostramos el proceso de evaluación para cada modelo y los resultados obtenidos durante la evaluación de ambos escenarios.

5.3. Evaluación y análisis

A continuación, se presentan las métricas usadas para la evaluación del clasificador, su implementación en Splunk y los resultados obtenidos, seguidamente presentamos un análisis de los resultados obtenidos.

5.2.4. Evaluación

Para la evaluación de los modelos se utilizaron las métricas de exactitud, recall, precisión y F1 como se mencionó en la sección 4.3. Para su implementación se usó la integración que posee MLTK con el módulo de Python for Scientific Computing (PSC) para poder compartir librerías de scikit-learn y usar algunas de sus funciones. Para medir la exactitud se usó la función *accuracy_score* y para medir la precisión, el recall y el F1 utilizamos la función de *precision_recall_fscore_support* que implementan las funciones disponibles en scikit-learn [63]. Para sus parámetros solo se modificó el atributo *average* el cual permite promediar los puntajes de cada clase y obtener el número de valores acertados para cada etiqueta.

Por otra parte, para generar la matriz de confusión se usó el macro *confusionmatrix*, que nos permite obtener la cantidad de predicciones para cada categoría con base a las etiquetas originales. específicamente cuenta cuantas de las etiquetas fueron categorizadas correctamente y rellena con ceros los valores vacíos. En los cuadros A4 y A5 de los anexos mostramos el código SPL para obtener estas mediciones. Las cuatro métricas fueron aplicadas a los dos escenarios utilizando 9 y 12 características para cada algoritmo y el resumen de los resultados del escenario A y B se muestran en los cuadros 20 y 21 respectivamente.

Cuadro 20: Resumen de resultados para el escenario A

Escenario A **(balance 16% malicioso – 84% legítimo)**					
Algoritmo	**# Caracteristicas**	**Exactitud**	**Precisión**	**Recall**	**F1**
Bosque Aleatorio	9	0.98	0.98	0.98	0.98
	12	0.98	0.98	0.98	0.98
Árboles de Decisión	9	0.97	0.97	0.97	0.97
	12	0.96	0.97	0.96	0.97

Cuadro 21: Resumen de resultados para el escenario B

Escenario B **(balance 33% malicioso – 66% legítimo)**					
Algoritmo	**# Caracteristicas**	**Exactitud**	**Precisión**	**Recall**	**F1**
Bosque Aleatorio	9	0.97	0.97	0.97	0.97
	12	0.98	0.98	0.98	0.98
Árboles de Decisión	9	0.96	0.96	0.96	0.96
	12	0.96	0.96	0.96	0.96

Según los resultados anteriores el algoritmo de bosque aleatorio presentó los mejores resultados con un 0.98 de exactitud usando tanto 9 como 12 características. El algoritmo de árboles de decisión mantiene una cercanía a este valor con 0.97 de exactitud para 9 características y 0.96 para 12 características. Finalmente, se compararon los resultados del escenario A usando todas las características con el resto de los algoritmos disponibles en el módulo de MLTK en el cuadro 22. Y se confirmó que ambos algoritmos desempeñan mejor al resto de los posibles algoritmos categóricos, afirmando el análisis de revisión literaria hecho en la sección 5.1.

Cuadro 22: Comparación de algoritmos en MLTK con 12 características

Algoritmo	**Exactitud**	**Precisión**	**Recall**	**F1**
RandomForest	**0.98**	**0.98**	**0.98**	**0.98**
DecisionTree	**0.96**	**0.97**	**0.96**	**0.97**
GaussianNB	0.93	0.83	0.83	0.87
BernoulliNB	0.91	0.92	0.92	0.90
SVM	0.94	0.75	0.75	0.82
LogisticRegression	0.93	0.94	0.94	0.93

Análisis de predicciones incorrectas

Para poder analizar las predicciones realizadas para cada categoría de URL se generaron las matrices de confusión de ambos algoritmos usando las 9 características más relevantes y se resumen estos resultados en el cuadro 23. Las matrices de confusión pueden encontrarse en los anexos en los cuadros A6, A7, A8 y A9.

Cuadro 23: Resultado de predicciones correctas e incorrectas

Algoritmo	Predicción	Legitimo	DGA	Malware	Phishing	Spam
Bosque Aleatorio (RF)	Correcta	99.8%	100%	86.6%	95.2%	70%
	Incorrecta	0.2%	0%	14.4%	4.8%	30%
Árboles de Decisión (DT)	Correcta	99.3%	100%	85%	90.8%	79.6%
	Incorrecta	0.7%	0%	15%	9.2%	21.4%

5.2.5. Análisis

En la sección anterior se presentaron los resultados obtenidos de nuestro modelo comparando dos escenarios. El escenario A teniendo un 16% de URL maliciosas y el escenario B teniendo un 33% de URL maliciosas.

Análisis de resultados para las métricas de exactitud, precisión, recall y F1

Según los resultados presentados en la sección anterior, el algoritmo de bosque aleatorio desempeñó mejor que el de árboles de decisión en ambos escenarios. En cuanto a diferencias en la cantidad de características el algoritmo de bosque aleatorio obtuvo la misma exactitud usando tanto 9 como las 12 características con un 98% para el escenario A. En el escenario B por su parte la exactitud se mantuvo en 98% al usarse todas las características, pero disminuyó a 97% al usar 9 características. Este fue el único caso de los cuatro escenarios en el que la exactitud fue menor al usarse solo las 9 características más relevantes.

Según la literatura, el algoritmo de bosque aleatorio erradica las limitaciones de un algoritmo de árboles de decisión pues reduce el sobreajuste de conjuntos de datos y aumenta la exactitud.

Sin embargo, el algoritmo de árboles de decisión obtuvo resultados positivos cercanos al de bosque aleatorio con un 97% y un 96% de exactitud para ambos escenarios usando las características relevantes únicamente. Estos dos algoritmos fueron seleccionados con base a la revisión de literatura por ser aquellos que obtuvieron mejores resultados a lo largo de los estudios existentes, pero se decidió confirmar esta selección al correr el mismo escenario con otros cuatro algoritmos disponibles en Splunk MLTK. El resultado de esta prueba mostró que los demás algoritmos no lograron acercarse a los resultados obtenidos con los algoritmos basados en árboles.

Además de la exactitud, los resultados muestran una alta precisión y recall, interpretándose de manera que no solo las predicciones realizadas son cercanas al valor real, sino además las predicciones de cada muestra son cercanas entre ellas. Esto otorga mayor confianza en el modelo, pues al tener una alta exactitud y precisión puede decirse que el modelo detecta con una alta probabilidad diferentes clases de URLs. Al incluirse las cuatro métricas podemos mantener la integridad de los resultados tomando en cuenta la paradoja de la exactitud al tratar clases desbalanceadas como se explicó en la sección 4.3.

Con respecto a la similitud de los resultados obtenidos para las cuatro métricas (mismos valores para F1 comparados a precisión y recall), sucede cuando los valores para recall (R) y precisión (P) son iguales y puede demostrarse en la ecuación 5 y 6. Los beneficios de incluir las cuatro métricas pueden apreciarse mejor al compararse con los resultados del resto de los algoritmos en MLTK en el cuadro 22. Por ejemplo, SVM presentó un 94% de exactitud, pero sus valores de precisión y recall fueron pobres con un 75%.

$$F1 = \frac{2 * \text{Recall} * \text{Precisión}}{\text{Recall} + \text{Precisión}} = \frac{2 * \text{R} * \text{P}}{\text{R} + \text{P}} \qquad (5)$$

$$F1 = \frac{2 * \text{R} * \text{P}}{\text{R} + \text{P}} = \frac{2 * \text{P} * \text{P}}{\text{P} + \text{P}} = \frac{2 * P^2}{2\text{P}} = \frac{P^2}{\text{P}} = P \qquad (6)$$

Análisis de falsos positivos

En cuanto a predicciones incorrectas, en el cuadro 23 presentado en la sección anterior se mostraron la cantidad de falsas predicciones obtenidas para cada categoría. Los resultados muestran que la categoría con mayor cantidad de falsos positivos fue la de spam con un 30%

usando bosque aleatorio y un 21.4% usando árboles de decisión. Por debajo de ella, le siguen las categorías de malware con un 14.4% usando bosques aleatorios y 15% usando árboles de decisión. Se encontró que aquellas URLs con pocos atributos diferenciadores como por ejemplo “m9u6eo3[.]jp”, “9880mal728[.]info” o “moonertis[.]fun” fueron categorizadas incorrectamente, especialmente asociados a la categoría de spam. Como un punto de mejora para estos sitios vemos importante tomar en consideración características específicas para extensiones inusuales y sitios que comiencen con dígitos en lugar de letras o con misma cantidad de dígitos como de letras.

Por otra parte, hubo categorías que tuvieron menos de un 10% de predicciones incorrectas como lo son las categorías de DGA con un 100% en ambos algoritmos, seguida por las URLs legitimas y phishing. Un punto importante es que solo se incluyó un tipo de familia de DGA en la categoría de botnet por lo que la semilla utilizada para crear la botnet posee una baja aleatoriedad. Por esta razón, como un punto de mejora se podrían agregar múltiples familias de botnet al conjunto de datos y medir cómo se comporta el modelo ante estos nuevos patrones.

Análisis de características

Según la literatura, la relevancia de las características es dependiente del conjunto de datos. Es decir, una gran cantidad de características puede causar resultados inconsistentes, y por otra parte muy pocas pueden no ser suficientes para identificar numéricamente el conjunto de datos. En nuestro trabajo se realizó un análisis de relevancia para cada característica con el fin de seleccionar aquellas variables de predicción más estables para el modelo. Con base a este análisis se redujo la cantidad de características de 12 a 9 y en los resultados obtenidos, 3 de los modelos generados presentaron una mayor o igual exactitud usando solo 9 características.

El rango de características usado en otros estudios según la revisión de literatura ronda en el rango de 3 a 300. En nuestro caso, se usaron solo 9 características y se obtuvieron buenos resultados lo cual es muy favorable y puede influir positivamente en el tiempo de creación del modelo en caso de operacionalizarse y actualizarse en tiempo real, así como en la cantidad de almacenamiento necesario para guardar las características de cada URL.

Capítulo 6: Conclusiones

En esta investigación se presentaron los resultados de construir un clasificador de URL usando aprendizaje automático supervisado usando Splunk y su módulo de aprendizaje MLTK a partir del uso de características estáticas para detectar tráfico malicioso en sus cuatro categorías principales: malware, spam, phishing y DGA. Se obtuvo un 98% de exactitud con el algoritmo de bosque aleatorio, demostrando así que es posible detectar URL maliciosas sin requerir la recolección de características dinámicas. A parte de este resultado, del conjunto de características inicial al cuál se le aplicó un análisis de relevancia con el cuál se logró reducir la cantidad de características de 12 a 9, reduciendo así el espacio de información requerida para lograr una buena predicción. Además, se obtuvo una mejora en la exactitud generada por el clasificador realizado en el estudio anterior de Cersosimo et al [63] al usar URLs en lugar de solo dominios de un 88% a un 98%.

Con respecto a los objetivos planteados al inicio de este trabajo se concluye que:

1. La revisión de literatura mostró una amplia selección de métodos para la detección de sitios maliciosos, pero ninguno de estos buscaba la reducción del espacio de características de recolección compleja como lo son las características dinámicas, ni maximizar la flexibilidad y operacionalización final del modelo al usar múltiples categorías maliciosas. Además, pocos estudios incluyen un análisis posterior a la extracción de sus características a pesar de la importancia que posee el seleccionar características relevantes. Con base a esto, se realizó un clasificador usando sólo características estáticas basadas en componentes léxicos. Nuestro trabajo mostró que es posible alcanzar resultados iguales o mayores a los encontrados en la literatura usando solo características léxicas, y descartando aquellas con baja relevancia para la predicción. Este hallazgo recalca la importancia de incluir análisis de relevancia luego de la selección inicial de las características para ayudar a reducir la cantidad de tiempo y espacio de la matriz de características y del procesamiento del modelo.
2. En cuanto a la implementación del clasificador, se obtuvieron buenos resultados a partir del marco de características definido con ambos algoritmos seleccionados y pudo confirmarse su desempeño al compararlos con otros algoritmos de predicción categórica disponibles en MLTK. Con respecto a la herramienta utilizada, Splunk nos

permitió realizar cada una de las etapas de aprendizaje automático, incluyendo el procesamiento inicial (etiquetado y agrupamiento), la extracción de características con su análisis de relevancia y finalmente la creación de los modelos con su respectiva evaluación. Además, es posible compartir estos modelos e importarlos en otra instancia de Splunk para la reproducción de los resultados. Algunas desventajas encontradas durante la investigación son que el comando *fit* y *apply* solo se pueden aplicar a búsquedas relativas en el tiempo y no en tiempo real, es decir, necesitan una tabla de características para la cual poder aplicar el modelo. Pero no es impedimento, pues Splunk provee la manera de programar búsquedas que puedan mantener la tabla de características actualizada para su operacionalización.

3. Los resultados del modelo presentaron un 98% de exactitud usando el algoritmo de bosque aleatorio tanto para 9 o 12 características usando el escenario con mayor desbalanceo y un 97% para el segundo escenario. Además, se tuvo una congruencia con sus otras métricas hermanas de precisión y recall con la que puede entenderse que el modelo de aprendizaje posee una alta exactitud en la predicción de las URL. En cuanto a los escenarios planteados, se descartó un impacto significativo en los resultados para rangos de entre 16% a 33% de las URL maliciosas.

Como conclusiones generales, la meta de este trabajo estaba dirigido a la operacionalización del modelo, por ello se seleccionó un escenario realista incluyendo diferentes comportamientos maliciosos encontrados en URL, una selección de características ligera sin depender de datos históricos o monitoreo de ventanas de tiempo y finalmente usar distribuciones de URL malicioso-legítimo similar a la actividad actual de internet para evaluar el modelo. Esto es una ventaja en términos de tiempo y almacenamiento en la etapa de caracterización y mejora el tiempo de procesamiento a la hora de crear el modelo. Esto puede beneficiar la actualización del modelo en tiempo real de una manera constante para alimentar el modelo con nuevos ataques.

Finalmente, el detectar dominios maliciosos depende de las técnicas empleadas por los adversarios para alcanzar sus objetivos. Estas técnicas deben poder ser caracterizadas para poder ser usadas en aprendizaje automático como los tipos de URL maliciosas usadas en este trabajo. Actualmente no se encuentra definido en la literatura un marco de referencia para la selección de características relevantes en la detección de sitios maliciosos y sus diferentes categorías. A pesar

de esto, el análisis de relevancia posterior a la selección de las características presenta un paso clave al trabajar con problemas de detección de URL. Pues, con el paso del tiempo algunas características podrían ir perdiendo relevancia mientras que otras puedan in influyendo más en la predicción. En este trabajo concluimos que un marco de referencia estático no es el indicado para la clasificación de URL, sino que se necesita un constante análisis de relevancia en las características del modelo con el que se pueda identificar la necesidad de agregar o remover características. El descarte de características puede ir de la mano con la mejora en la seguridad de los estándares, políticas y tecnologías de seguridad en las organizaciones por lo que puede ser maleable a la postura organizacional de una empresa en la que se desee aplicar el modelo.

Capítulo 7: Bibliografía

[1] The MITRE Corporation, «Application Layer Protocol: Web Protocols,» 15 March 2020. [En línea]. Available: https://attack.mitre.org/techniques/T1071/001/. [Último acceso: 30 May 2022].

[2] Crowdstrike, «Nowhere to Hide 2021 Threat Hunting Report,» Insights From the Falcon Overwatch Team, 2021.

[3] Palo Alto Networks Inc, «DNS Security Predict and Disrupt Today's Most Sophisticated DNS-Based Attacks,» 2021.

[4] IBM Security, «X-Force Threat Intelligence Index Report,» 2020.

[5] Webroot BrightCloud, «Threat Report 2022,» Carbonite, Webroot, 2022.

[6] S. Mahdavifar, N. Maleki, H. A. Lashkari, M. Broda y A. H. Razavi, «Classifying Malicious Domains using DNS Traffic Analysis,» de *The 19th IEEE International Conference on Dependable, Autonomic, and Secure Computing (DASC)*, 2021.

[7] Y. D. Goli y R. Ambika, «Network Traffic Classification Techniques-A Review,» de *International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, 2018.

[8] Q. Nguyen, R. Laborde, A. Benzekri y B. Qu'hen, «Detecting abnormal DNS traffic using unsupervised machine learning,» de *4th Cyber Security in Networking Conference (CSNet)*, 2020.

[9] T. Wang y L. Chen, «Detecting Algorithmically Generated Domains Using Data Visualization and N-Grams Methods,» de *Proceedings of Student-Faculty Research Day, CSIS*, 2017.

[10] T. Kelley y E. Furey, «Getting Prepared for the Next Botnet Attack : Detecting Algorithmically Generated Domains in Botnet Command and Control,» de *29th Irish Signals and Systems Conference (ISSC)*, 2018.

[11] Shantanu, J. B y J. Kumar, «Malicious URL Detection: A Comparative Study,» de *Proceedings of the International Conference on Artificial Intelligence and Smart Systems (ICAIS-2021)*, 2021.

[12] D. Cheng, Z. Liu, P. Zhang, Y. Zeng, J. Cui y L. Kong, «Profiling Malicious Domain by Multidimensional Features,» de *International Conference on Robots & Intelligent System (ICRIS)*, Changsha, China, 2018.

[13] Y. Zhauniarovich, I. Khalil, T. Yu y M. Dacier, «A Survey on Malicious Domains Detection through,» de *ACM Computing Surveys, Vol. 1, No. 1, Article 1. Publication date: May 2018*, 2018.

[14] L. Bilge, E. Kirda y C. Kruegel, «EXPOSURE: Finding Malicious Domans Using Passive DNS Analysis,» de *Network and Distributed Systems Security Symposioum*, 2011.

[15] G. Palaniappan, S. S, S. Rajendran, S. Goyal y B. S. Bindhumadhava, «Malicious Domain Detection Using Machine Learning On Domain Name Features, Host-Based Features and Web-Based Features,» de *Procedia Computer Science*, 2020.

[16] A. S. U. Adam Oest and Penghui Zhang, E. N. a. J. B. P. Brad Wardman, G. Ali Zand and Kurt Thomas, A. S. U. Adam Doupé y A. S. U. S. R. Gail-Joon Ahn, «Sunrise to Sunset: Analyzing the End-to-end Life Cycle and Effectiveness of Phishing Attacks at Scale,» de *USENIX* , 2020.

[17] J. Kumar, A. Santhanavijayan y B. Janet, «Phishing Website Classification and Detection,» de *2020 International Conference on Computer Communication and Informatics*, 2020.

[18] C. Hajaj, «Less is More: Robust and Novel Features for Malicious Domain Detection,» de *ArXiv*, 2020.

[19] FireEye, «LEVIATHAN: Command and Control Communications on Planet Earth,» Black Hat Las Vegas, 2014 .

[20] M. Antonakakis, R. Perdisci y D. Dagon, «Building a Dynamic Reputation System for DNS,» de *Usenix Security Symposium*, 2010.

[21] R. Perdisci, I. Corona y G. Giacinto, «Early Detection of Malicious Flux Networks via Large-Scale Passive DNS Traffic Analysis,» de *IEEE Transactions on Dependable and Secure Computing*, 2012.

[22] K. Wu, Y. Zhang y T. Yin, «FTPB: A Three-Stage DNS Tunnel Detection Method Based on Character Feature Extraction,» de *IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020.

[23] T. Su, S. Wang, Y. Chen y C. Liu, «Attack detection of distributed denial of service based on Splunk,» de *International Conference on Advanced Materials for Science and Engineering (ICAMSE)*, 2016.

[24] S. Alspaugh, M. A. Hearst, A. Ganapathi y R. Katz, «Building blocks for exploratory data analysis tools,» de *In Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics (IDEA '13)*, 2013.

[25] Z. Bao, W. Wang y Y. Lan, «Using Passive DNS to Detect Malicious Domain Name,» de *International Conference on Vision, Image and Signal Processing*, Vancouver, BC, Canada, 2019.

[26] S. Yadav, R. A. K. K, R. A. L. N y S. Ranjan, «Detecting Algorithmically Generated Domain-Flux Attacks With DNS Traffic Analysis,» de *IEEE/ACM Transactions on Networking*, 2012.

[27] N. Bhoj, R. Bawari, A. Tripathi y N. Sahai, «Naive and Neighbour Approach for Phishing Detection,» de *10th IEEE International Conference on Communication Systems and Network Technologies*, 2021.

[28] S. Le Page y G.-V. Jourdan, «Victim or Attacker? A Multi-dataset Domain,» de *2019 17th International Conference on Privacy, Security and Trust (PST)*, 2019.

[29] K. Pradeepthi y K. A, «Performance Study of Classification Techniques,» de *Sixth International Conference on Advanced Computing(ICoAC)*, 2014.

[30] P. Patil, R. Rashmi y M. Bhalekar, «Detecting Spam and Phishing Mails Using SVM,» de *International Conference on Inventive Systems and Control*, 2017.

[31] M. Henke, E. Souto y E. M. dos Santos, «Analysis of the Evolution of Features in Classification Problems with Concept Drift:,» de *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015.

[32] R. Bharadwaj, A. Bhatia, L. Chhibbar, K. Tiwari y A. Agrawal, «Is this URL Safe: Detection of Malicious URLs Using Global Vector for Word Representation,» de *2022 International Conference on Information Networking (ICOIN)*, 2022.

[33] G. Yan, Q. Li, D. Guo y X. Meng, «Discovering Suspicious APT Behaviors by Analyzing DNS Activities,» *Sensors (Basel)*, vol. 20(3), nº 731, 2020.

[34] Q. Wang, B. Jiang, Z. Lu, J. Liu y J. Shijie, «Malicious Domain Detection Based on K-means and SMOTE,» de *Computational Science – ICCS 2020*, 2020.

[35] G. Zhao, K. Xu y L. Xu, «Detecting APT Malware Infections Based on Malicious DNS and Traffic Analysis,» de *IEEE Access*, 2015.

[36] H. Choi y H. Lee, «Identifying botnets by capturing group activities in DNS traffic,» de *Computer Networks*, 2012.

[37] S. Mahdavifar, N. Maleki, A. H. Lashkari, M. Broda y A. H. Razavi, «CIC-Bell-DNS2021 Data Set,» [En línea]. Available: https://www.unb.ca/cic/datasets/dns-2021.html. [Último acceso: 12 Dec 2021].

[38] M. Cersosimo y A. Lara, «Detecting Malicious Domains using the Splunk Machine Learning Toolkit,» de *2022 IEEE/IFIP Network Operations and Management Symposium (NOMS)*, CEST Budapest, Hungary, 2022.

[39] P. A. Networks, «PAN-OS® Administrator's Guide Security-Focused URL Categories,» Palo Alto Networks, [En línea]. Available: https://docs.paloaltonetworks.com/pan-os/9-1/pan-os-admin/url-filtering/url-categories/url-risk-categories. [Último acceso: 4 June 2022].

[40] Microsoft, «Naming conventions in Active Directory for computers, domains, sites, and OUs,» 2022. [En línea]. Available: https://docs.microsoft.com/en-us/troubleshoot/windows-server/identity/naming-conventions-for-computer-domain-site-ou. [Último acceso: 30 May 2022].

[41] J. Umawing, «Long lost @ symbol gets new life obscuring malicious URLs,» Malwarebytes, 17 May 2022. [En línea]. Available: https://www.malwarebytes.com/blog/news/2022/05/long-lost-symbol-gets-new-life-obscuring-malicious-urls. [Último acceso: 9 November 2022].

[42] T. Dangkesee y S. Puntheeranurak, «Adaptive Classification for Spam Detection on Twitter with Specific Data,» de *21st International Computer Science and Engineering Conference*, 2017.

[43] FBI.gov, «DNSChanger Malware,» [En línea]. Available: https://www.fbi.gov/DNS-changer-malware.pdf.

[44] T. Blizard y N. Livic, «Click-fraud Monetizing Malware: A Survey and Case Study,» de *2012 7th International Conference on Malicious and Unwanted Software*, Fajardo, 2012.

[45] MathWorks, «Feature Extraction,» Matlab, [En línea]. Available: https://www.mathworks.com/discovery/feature-extraction.html. [Último acceso: 11 November 2022].

[46] Y. Zhang, T. Liu y H. Sha, «Malicious domain detection based on multiple-dimensional features,» de *Journal of Computer Applications*, 2016.

[47] Y. Chu, L. Ying y D. Feng, «Behavioral Analysis of Fast Flux Service Network,» de *Computer Systems and Applications*, 2013.

[48] L. Zhao, «Design and Development of the Maliciious Domain Identification System based on DNS,» de *Shandong University*, 2013.

[49] F. Weimer, «Passive DNS Replication,» 2005.

[50] B. Zdrnja y D. Wessels, «Passive Monitoring of DNS Anomalies,» de *International Conference on Detection of Intrusions and Malware, and Vulnerability Assesment*, 2007.

[51] Z. Fan y R. Liu, «Investigation of Machine Learning Based Network Traffic Classification,» de *International Symposium on Wireless Communication Systems (ISWCS)*, 2017.

[52] Cyberline, «Beneficios de Splunk, Plataforma, Infraestructura de TI y Servicios Cloud,» Cyberline , [En línea]. Available: https://www.cyberline.com.pe/web/servicio/splunk . [Último acceso: 12 12 2021].

[53] S. Jaworski, «Using splunk to detect dns tunneling,» de *SANS Institute InfoSec Reading Room*, 2016.

[54] Splunk Inc, «Operationalize Machine Learning for DGA Detection, White Paper,» Splunk, 2017.

[55] Splunk, «Splexicon: SPL,» Splunk, [En línea]. Available: https://docs.splunk.com/Splexicon:SPL. [Último acceso: 10 November 2022].

[56] I. f. C. a. E. a. t. B. U. o. A. S. o. Switzerland, «URLhaus API,» Abuse.ch, 2022. [En línea]. Available: https://urlhaus.abuse.ch/api/#csv. [Último acceso: 6 June 2022].

[57] D. Naeem, BT Security, M. Brenton y Zurich Insurance Group, «SUNBURST Malware Tracking S0559,» MITRE ATTACK, 2021. [En línea]. Available: https://attack.mitre.org/software/S0559/. [Último acceso: 6 June 2022].

[58] R. Kovar, «Sunburst Backdoor Github Lookup Compilation,» [En línea]. Available: https://github.com/rkovar/sunburstlookups. [Último acceso: 12 Dec 2021].

[59] M. F. Uddin, «Learning, Addressing Accuracy Paradox Using Enhanched Weighted Performance Metric in Machine,» de *Sixth HCT Information Technology Trends (ITT)*, Ras Al Khaimah, United Arab Emirates, 2019.

[60] s.-l. developers, «API Reference - Metrics,» Scikit-Learn, 2007 - 2022. [En línea]. Available: https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics. [Último acceso: 6 June 2022].

[61] L. Teo, «Automated Detection and Analysis using Mathematical Calculations,» The SANS Institute, 2018.

[62] S.-l. B. License, «TfidfVectorizer,» [En línea]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html. [Último acceso: 5 June 2022].

[63] S. Learn, «sklearn.metrics.precision_recall_fscore_support¶,» [En línea]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html. [Último acceso: 6 June 2022].

[64] M. Stevanovic, J. M. Pederson y A. D'Alconzo, «On the ground truth problem of malicious DNS traffic analysis,» de *Computers and Security*, 2015.

[65] J. Petrov, «Mitre Technique ID: T1071.004,» MITRE Corporation, 2020. [En línea]. Available: https://attack.mitre.org/techniques/T1071/004/.

[66] I. Khalil, T. Yu y B. Guan, «Discovering Malicious Domains through Passive DNS Data Graph Analysis,» de *ACM on Asia Conference on Computer and Communications Security*, 2016.

[67] Harvard Business Review Analytic, «The Power of Predictive IT,Improve Reliability and Prevent Outages Across Your Organization, White Paper,» MC211621218, 2019.

[68] Splunk Inc, «AI and Machine Learning in Your Org, White Paper,» Splunk, 2017.

[69] K. Barbosa y K. El-Khatib, «Identifying and Classifying Suspicious Network,» de *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications;*, 2015.

[70] Splunk, «Understanding Fit and Apply,» Splunk, [En línea]. Available: https://docs.splunk.com/Documentation/MLApp/5.3.1/User/Understandfitandapply . [Último acceso: 5 June 2022].

Capítulo 8: Anexos

Anexo A: Apartado de cuadros

Cuadro A1: Código para el etiquetado de los datos

```
| inputlookup tfia_legit.csv
| fillnull value="legit"
| outputlookup tfia_legit_alexa.csv

| inputlookup tfia_malware.csv
| fillnull class value="malicious"
| fillnull subclass value="malware"
| outputlookup tfia_malware.csv

| inputlookup tfia_spam.csv
| fillnull class value="malicious"
| fillnull subclass value="spam"
| outputlookup tfia_spam.csv

| inputlookup tfia_phishing.csv
| fillnull class value="malicious"
| fillnull subclass value="phishing"

| inputlookup tfia_dga.csv
| fillnull class value="malicious"
| fillnull subclass value="dga"
| outputlookup tfia_dga.csv
```

Cuadro A2: Función de entropía

```
def shannon(word):
    entropy = 0.0
    length = len(word)
    occ = {}
    for c in word :
        if not c in occ:
            occ[ c ] = 0
        occ += 1

    for (k,v) in occ.iteritems():
        p = float( v ) / float(length)
        entropy -= p * math.log(p, 2) # Log base 2
    return entropy
```

Cuadro A3: Código para la creación de las características

```
| inputlookup scenario_a_dataset.csv
| apply "tfia_ngram"
| apply "tfia_pca"
| fields - domain_tfidf*
| `ut_shannon(domain)`
| `ut_meaning(domain)`
| `ut_bayesian(domain)`
| eval set="."
| `ut_countset(domain, set)` | spath input=ut_countset | rename
ut_countset.sum AS "ut_n_dots"
| eval set="-"
| `ut_countset(domain, set)` | spath input=ut_countset | rename
ut_countset.sum AS "ut_n_hyphens"
| eval ut_digit_ratio = 0.0
| eval ut_vowel_ratio = 0.0
| eval ut_domain_length = max(1,len(domain))
| rex field=domain max_match=0 "(?<digits>\d)"
| rex field=domain max_match=0 "(?<vowels>[aeiou])"
| eval ut_digit_ratio=if(isnull(digits),0.0,mvcount(digits) /
ut_domain_length)
| eval ut_vowel_ratio=if(isnull(vowels),0.0,mvcount(vowels) /
ut_domain_length)
| eval ut_consonant_ratio = max(0.0, 1.000000 - ut_digit_ratio -
ut_vowel_ratio)
| fields - digits - vowels - ut_countset*
| sample partitions=2
| outputlookup scenario_a_dataset_features.csv
```

Cuadro A4: Código para evaluar el modelo

```
| inputlookup scenario_A_dataset_features.csv
| apply "scenario_A_model_RF_9F_5050_Psubclass"
| multireport
           // Obteniendo la exactitud
            [ score accuracy_score "class" against "predicted(class)"
            | eval accuracy = round(accuracy_score, 2)]

           // Obteniendo la precision, el recall y f1
            [ score precision_recall_fscore_support "class" against
"predicted(class)" average=weighted
            | rename fbeta_score as f1
            | fields f1 precision recall
            | foreach *
                [ eval <<FIELD>> = round(<<FIELD>>, 2)]]
           // Tabulando resultados
            | table accuracy f1 precision recall
            | stats first(*) as *
```

Cuadro A5: Código para generar la matriz de confusión

```
| inputlookup scenario_A_dataset_features.csv
| apply "scenario_A_model_RF_9F_5050_Psubclass"
| rename “subclass” as actual, "predicted(subclass)" as predicted
| stats count by actual predicted
| appendpipe [ eval predicted=actual, count=0 ]
| stats sum(count) as count by actual predicted
| xyseries actual predicted count
| rename * as "Predicted *"
| rename "Predicted subclass" as "Actual subclass"
| fillnull value=0
```

Anexo B: Apartado de figuras

Predicted actual	Predicted DGA	Predicted legit	Predicted malware	Predicted phishing	Predicted spam
DGA	486 (100%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Legitimo	0 (0%)	30185 (99.8%)	0 (0%)	0 (0%)	65 (0.2%)
malware	0 (0%)	0 (0%)	1270 (86.6%)	196 (13.4%)	0 (0%)
phishing	0 (0%)	0 (0%)	117 (4.8%)	2334 (95.2%)	0 (0%)
spam	0 (0%)	362 (30%)	0 (0%)	0 (0%)	843 (70%)

Figura 30: Matriz de confusión para el escenario A con bosques aleatorios

Predicted actual	Predicted DGA	Predicted legit	Predicted malware	Predicted phishing	Predicted spam
DGA	477 (100%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Legitimo	0 (0%)	29417 (99.3%)	1 (0%)	1 (0%)	514 (1.7%)
malware	0 (0%)	0 (0%)	1266 (85%)	223 (15%)	0 (0%)
phishing	0 (0%)	2 (0.1%)	235 (9.1%)	2351 (90.8%)	0 (0%)
spam	0 (0%)	248 (20.4%)	0 (0%)	0 (0%)	968 (79.6%)

Figura 31: Matriz de confusión para el escenario A con árboles de decisión

Predicted actual	Predicted DGA	Predicted legit	Predicted malware	Predicted phishing	Predicted spam
DGA	857 (99.3%)	6 (0.7%)	0 (0%)	0 (0%)	0 (0%)
Legitimo	0 (0%)	29815 (99.5%)	0 (0%)	4 (0%)	137 (0.5%)
malware	0 (0%)	0 (0%)	3627 (86.2%)	583 (13.8%)	0 (0%)
phishing	0 (0%)	3 (0.1%)	373 (7.4%)	4673 (92.6%)	0 (0%)
spam	0 (0%)	12 (0.3%)	0 (0%)	0 (0%)	4108 (99.7%)

Figura 32: Matriz de confusión para el escenario B con bosques aleatorios

Predicted actual	Predicted DGA	Predicted legit	Predicted malware	Predicted phishing	Predicted spam
DGA	857 (99.3%)	6 (0.7%)	0 (0%)	0 (0%)	0 (0%)
Legitimo	0 (0%)	29815 (99.5%)	0 (0%)	4 (0%)	137 (0.5%)
malware	0 (0%)	0 (0%)	3627 (86.2%)	583 (13.8%)	0 (0%)
phishing	0 (0%)	3 (0.1%)	373 (7.4%)	4673 (92.6%)	0 (0%)
spam	0 (0%)	12 (0.3%)	0 (0%)	0 (0%)	4108 (99.7%)

Figura 33: Matriz de confusión para el escenario B con árboles de decisión