Guardián virtual: propuesta de herramienta para la protección contra el *phishing* basado en mensajes, URL y contenido de páginas web

Andrés Alejandro de la Roca Pineda

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Guardián virtual: propuesta de herramienta para la protección contra el *phishing* basado en mensajes, URL y contenido de páginas web

Trabajo de graduación presentado por Andrés Alejandro de la Roca Pineda para optar al grado académico de Licenciado en Ingeniería en Ciencia de la Computación y Tecnologías de la Información

Guatemala,

2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Guardián virtual: propuesta de herramienta para la protección contra el *phishing* basado en mensajes, URL y contenido de páginas web

Trabajo de graduación presentado por Andrés Alejandro de la Roca Pineda para optar al grado académico de Licenciado en Ingeniería en Ciencia de la Computación y Tecnologías de la Información

Guatemala,

2024

Vo.Bo.:

(f) _____

MSc. Jorge Andrés Yass

Tribunal Examinador:

(f) _____

MSc. Jorge Andrés Yass

(f) _____

MSc. Oscar Canek

(f) _____

PhD. Gabriel Barrientos

Fecha de aprobación: Guatemala, 2 de diciembre de 2024.

Agradecimientos

Me gustaría expresar mi gratitud a todas aquellas personas que, de una u otra forma, contribuyeron a la realización de este proyecto de graduación. Agradezco a mi asesor Jorge Yass, por su gran apoyo y orientación durante todo este proceso. Su experiencia y recomendaciones fueron claves para llevar a cabo este proyecto.

A mis padres, Kennia Pineda y Juan Manuel de la Roca, por su incondicional amor, apoyo y sacrificio todos estos años. Muchas gracias por creer en mí, por su cariño y por siempre estar a mi lado, este trabajo es, en gran parte, un reflejo de su amor y confianza en mí.

A mis compañeros y amigos, que compartieron conmigo tiempo y ánimos, haciendo de esta etapa una experiencia enriquecedora.

Índice

Lista de figuras

Lista de cuadros

Resumen

El presente trabajo de graduación aborda la creciente amenaza de *phishing*, una técnica de ingeniería social utilizada por ciberdelincuentes para obtener información confidencial, como credenciales bancarias o datos personales. Investigaciones previas han desarrollado soluciones para detectar y mitigar estos ciberataques; sin embargo, la gran mayoría de estas soluciones proponen la detección de solo un posible vector de ataque. Este trabajo propone una extensión web de detección de *phishing* que combina detección por listas y por machine learning, enfocada a la detección de ataques por medio de mensajes, URL y contenido web. Se logró obtener una alta precisión en los modelos de *machine learning* de la herramienta, logrando una alta robustez ante diferentes ataques de *phishing*. Este trabajo pretende ayudar a la evaluación de diferentes enfoques para el desarrollo de herramientas dedicadas a la detección y mitigación de *phishing*.

Abstract

This graduation project addresses the growing threat of phishing, a social engineering technique mainly utilized by cybercriminals to obtain sensitive information such as banking credentials or personal data. Previous investigations have developed different tools to detect and mitigate these cyberattacks, however, most of these tools propose having coverage over only one possible attack vector. This project proposes a web extensión for phishing detection which combines List-based detection and machine-learning model detection, its main focus being detecting phishing attacks based on text messages, URL, and web content. The tool achieved high accuracy in its machine learning models, resulting in strong protection against various phishing attacks. This work aims to contribute to the evaluation of different approaches in the development of tools dedicated to phishing detection and mitigation.

CAPÍTULO 1

Introducción

El *phishing* ha sido una amenaza prevalente en el internet durante las últimas cuatro décadas, aún con investigaciones exhaustivas y un desarrollo constante de soluciones que combaten el phishing, estos ciberataques siguen siendo uno de los principales peligros cibernéticos hasta el día de hoy [1]. Registros de 2019 a 2022 reportan que la tasa de ataques de *phishing* ha aumentado más del 100 % dentro de ese plazo de tiempo [2], lo cual puede representar una evolución de las tácticas que los ciberdelincuentes están empleando para eludir los controles de prevención establecidos.

Para poder hablar en detalle sobre el *phishing* es importante comprenderlo en su totalidad debido a que este se puede manifestar de diferentes formas y tener efectos negativos sobre los usuarios de internet [3]. El término *phishing* se refiere a un conjunto de técnicas que utilizan ingeniería social, es decir, que un ciberdelincuente manipula a una víctima para poder conseguir su objetivo mediante interacciones sociales, por ejemplo, se pueden hacer pasar por familiares, personas de soporte técnico, compañeros de trabajo u otras personas de confianza. El objetivo principal de estos ataques es que el atacante pueda conseguir información importante y privada del usuario como credenciales de acceso, credenciales bancarias, dirección personal, entre otras [4].

El *machine learning* es un acercamiento muy utilizado para la detección del phishing, estudios han demostrado que con modelos basados en *machine learning* se puede alcanzar un buen desempeño, en especial si se les entrena con un conjunto de datos robusto. Debido a su alta precisión y eficiencia, se les considera la manera más efectiva de detectar y prevenir estos ataques [5].

Por medio de sistemas de detección basada en listas, es posible mitigar de manera muy eficiente cierto tipo de intentos de phishing, especialmente los más habituales que contienen un URL adjunto al mensaje del atacante. Comúnmente, este acercamiento utiliza dos listas, lista blanca y lista negra, que contienen URL legítimas y URL maliciosas, respectivamente. La mayor desventaja de utilizar este método es el hecho de que es susceptible a ataques con URL maliciosas que no hayan sido reportadas con anterioridad, por lo que requiere que la lista negra se esté actualizando constantemente [6].

Este trabajo se sumerge en el diseño y desarrollo de una extensión web con capacidades de detección basadas en listas y modelos de *machine learning*, proponiendo una combinación de estas dos estrategias para la detección de *phishing*, con la finalidad de tener cobertura sobre las mayores amenazas del *phishing*; *phishing* basado en texto, URL y contenido de páginas web.

Durante el desarrollo del proyecto se logró definir un sistema que permite integrar la detección basada en listas y detección basada en *machine learning*, con el objetivo de tener una primera y segunda línea de defensa ante ataques de *phishing*. Se logró identificar diferentes características dentro de URL, mensajes y contenido web que pueden ayudar a la detección del *phishing*. Se destacaron algunas áreas de mejora, como la necesidad de optimizar los modelos para reducir los tiempos de entrenamiento y mejorar la precisión. También se consideró la posibilidad de expandir la herramienta a otros navegadores y plataformas, además de incorporar soporte multilingüe para una mayor cobertura.

CAPÍTULO 2

Antecedentes

Dentro del campo de las herramientas de detección de *phishing* se han hecho diferentes avances a lo largo del tiempo con el objetivo de ofrecerle a los posibles usuarios una defensa ante el *phishing* en sus diferentes formatos.

En 2019, [1] presentó una investigación sobre una extensión web con capacidades de detección de *phishing* por medio de contenido web, utilizando un enfoque híbrido basado en diferentes funciones de detección. Se presentó un sistema para una hipotética extensión en la que los usuarios, al interactuar con un enlace, se activará automáticamente una detección que actuaría a través de tres pasos diferentes:

1. Detección por lista blanca para detectar de manera rápida sitios considerados como seguros
2. Detección por lista negra para detectar sitios que hayan sido identificados como inseguros y pertenecientes a ataques de *phishing*.
3. Extracción de los hipervínculos y formas del sitio web que se desea visitar para realizar un análisis a través de reglas y una pila de modelos clasificadores de *machine learning*.

Mientras que la investigación propone un sistema para una extensión web que intenta implementar varias técnicas combinadas, no profundiza en el desarrollo de la propia extensión ni discute la interacción que la herramienta tendría con el usuario. Vale la pena mencionar que, las fuentes que utilizó para el desarrollo de la detección basada en listas ya no se encuentran fácilmente disponibles al público. Adicionalmente, los resultados presentados se basan únicamente en las métricas de precisión y no se realiza ningún análisis sobre matrices de confusión para observar la tasa de error del modelo, ni ninguna validación cruzada sobre los modelos de *machine learning*.

En 2021, [4] presentó una extensión web enfocada en la detección de *phishing* a través de correos electrónicos. Esta herramienta emplea redes neuronales para detectar *phishing* basa-

da en correos electrónicos. Esta investigación cuenta con un enfoque mayor en la experiencia de usuario e interfaz gráfica, para hacer de la herramienta lo más accesible posible.

Una investigación realizada [7] en 2021 propuso una extensión web para la detección de sitios web de *phishing* utilizando algoritmos de similitud, que emplea diferentes técnicas de extracción de características en las páginas web con el objetivo de poder realizar diferentes análisis comparativos sobre la página web.

Las herramientas propuestas en estas investigaciones tienen cobertura sobre solo uno de los vectores de ataque de *phishing*, sin embargo, el presente trabajo de graduación propone una herramienta en forma de extensión web que tenga cobertura sobre tres posibles vectores de ataque: mensajes, URL y contenido web; e integrando el uso de listas de detección con fuentes públicas disponibles para el filtrado de enlaces web.

CAPÍTULO 3

Justificación

El fenómeno del *phishing* ha tenido un gran impacto en la seguridad de la información durante los últimos años, es una problemática que puede llegar a tener efectos desastrosos para individuos y empresas involucradas. La principal preocupación es la filtración de datos a terceros que puedan realizar actos maliciosos con estos. Los datos que se consideran un objetivo principal para los atacantes son datos personales, datos internos de la empresa, credenciales, información bancaria, etc. Este problema ha estado en constante aumento debido al crecimiento exponencial de las actividades en línea. Según [8], en 2023 se registró que un aproximado del 20 % de los ataques informáticos de mayor uso fueron los ataques de *phishing* por correo electrónico, además, los correos electrónicos conforman el 40 % de los vectores de ataque para ataques cibernéticos, llegando a conformar casi el 100 %, si se habla específicamente de ataques de ingeniería social.

Estos son datos alarmantes que son un llamado a la acción para los investigadores de herramientas de ciberseguridad. Por lo tanto, es crucial investigar y desarrollar nuevas técnicas y herramientas para combatir eficazmente el *phishing* para proteger la integridad y confidencialidad de la información en entornos digitales. Entre las soluciones a destacar están las basadas en *machine learning*, las cuales son utilizadas en la mayoría de contextos en los que se pueden presentar vectores de ataques, como, correo electrónico, mensajería instantánea y contenido de página web. Las soluciones basadas en heurística también se han explorado mucho recientemente, su principal ventaja es el poder detectar ataques de día cero debido a sus características. El *deep learning* también es utilizado para el desarrollo de estas soluciones, pero debido a su complejidad, las herramientas presentan tiempos de respuesta elevados a comparación de las demás metodologías [5]. El progreso de las herramientas enfocadas en seguridad cibernética ha ayudado a reforzar y disminuir los posibles riesgos al que se exponen los dispositivos electrónicos.

Las herramientas actuales suelen tener algunas desventajas, entre ellas, la falta de rendimiento, ya que por lo general suelen correr procesos que pueden tomar un tiempo considerable en ejecutarse, algo lo cual afecta la experiencia de usuario y lo cual puede llegar a ser perjudicial, ya que el usuario puede decidir dejar de usar la herramienta debido a los

efectos negativos que esta ralentización puede tener sobre su productividad. También, se suele tener una cobertura parcial hacia el *phishing*, ya que por lo general se intenta enfocar la herramienta en un solo aspecto del ataque, ya sea por medio del análisis de la URL, cuerpo del texto o de la página maliciosa en sí. Esto puede llevar a vulnerabilidades en las otras áreas donde no se está tomando en cuenta esta cobertura por parte de la herramienta. Adicionalmente, la poca interacción que existe entre las herramientas y los usuarios puede generar esa sensación de seguridad total sobre el usuario, pero es importante que se tenga una interacción de algún tipo para que el usuario se pueda informar sobre las posibles vulnerabilidades a las que está expuesto durante la navegación web, para que en caso el análisis de vulnerabilidades falle, este pueda actuar de manera adecuada ante estas amenazas. En pocas palabras, las implementaciones de estas soluciones vienen con sus desventajas como falta de rendimiento, cobertura parcial hacia varios de los ataques de *phishing* y poca interacción con el usuario, esto sumado a que la mayoría de soluciones tienen un solo enfoque lo cual puede llegar a afectar negativamente a la herramienta o modelo [5].

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Desarrollar un sistema de protección basado en capas de detección para la mitigación de *phishing* durante la navegación web.

4.2. Objetivos específicos

- Implementar una capa de detección basada en listas blancas y negras.
- Implementar capas de detección basadas en modelos de *machine learning* .
- Desplegar una extensión web para Chromium que le permita al usuario monitorear e interactuar con el sistema de detección.

CAPÍTULO 5

Marco teórico

5.1. ¿Qué es el *phishing*?

El *phishing* es un término utilizado para definir un conjunto de técnicas que utilizan ingeniería social con el objetivo de obtener información sensible o privada de una persona o un grupo de personas, la situación más común en la que se presenta es cuando un ciberdelincuente manipula a una víctima para poder conseguir información, ya sea mediante interacciones sociales o engaños de percepción. Estos ataques explotan una de las vulnerabilidades más grandes de la ciberseguridad, el usuario, el cual se le considera el eslabón más débil dentro de los procesos de seguridad cibernética, esto hace del *phishing* uno de los ataques cibernéticos más efectivos [9].

Estos tipos de ataques puede presentarse en diferentes formas, los principales vectores de ataque siendo correos electrónicos falsificados, enlaces web engañosos, páginas web con contenido malicioso, entre otras. Más adelante se explicarán estos métodos con mayor detalle. El objetivo principal de estos ataques es obtener información sensible y privada como credenciales de acceso, credenciales bancarias, secretos corporativos, etc. [4]

Según [10] los usuarios son especialmente vulnerables ante estos ataques debido a estas 5 razones principales:

- Desconocimiento sobre las características de URL.
- Desconocimiento de que páginas web pueden ser confiables.
- Falta de observación de la URL completa de las páginas web, en especial durante redirecciones.
- Los usuarios tienen poco tiempo para consultar la URL o accidentalmente acceden a páginas indeseadas.

- Falta de experiencia al realizar la distinción entre mensajes de *phishing* y mensajes legítimos.

El estudio también menciona que por lo general los usuarios que son blancos de ataques de *phishing* se encuentran en países altamente desarrollados como China y Estados Unidos, adicionalmente, los usuarios más vulnerables son personas menores de edad o personas de la tercera edad las cuales pueden ser engañadas mediante el uso de técnicas de ingeniería social.

5.1.1. Importancia de su detección

El *phishing* ha sido una problemática persistente en ciberseguridad durante décadas, adaptándose y evolucionando junto con el avance de la tecnología y medidas de seguridad cibernéticas. Desde sus primeras apariciones en 1995, cuando los atacantes utilizaban ingeniería social para obtener credenciales de servicios de correo electrónico, como *AOL*, el término y su alcance se han expandido de manera significativa. A medida que nuestra dependencia a la tecnología ha crecido, también lo han hecho las técnicas de *phishing*, llegando a abarcar una amplia gama de métodos y técnicas que ha consolidado a este tipo de ciberataque como uno de los más efectivos y peligrosos en nuestro entorno digital actual [11].

Una de las razones principales por la que el *phishing* es un ciberataque tan efectivo es debido a la utilización de técnicas de imitación, cuando un atacante le hace creer a su víctima que está en contacto con una organización o persona de confianza. En el caso de que el atacante esté haciéndose pasar por una empresa, se utiliza la falsificación de mensajes automáticos como un vector de ataque. El atacante hace pensar a la víctima que está siendo contactada desde un medio seguro y automatizado para darle una advertencia, la cual por lo general es de naturaleza urgente, intentando forzar a la víctima a tomar la decisión apresurada de responder con información sensible o siendo redirigida a alguna página falsificada que intentara robar su información. Por otro lado, en el caso de que el atacante se haga pasar por una persona de confianza, este puede hacerse pasar por una persona de servicio técnico, una persona en una posición de poder o incluso como una persona de confianza como un familiar o amigo, en estos casos se le exhorta a la víctima que de información personal, la cual el atacante utilizara para beneficio propio [12].

Se ha entrado en detalle del porqué esta amenaza es tan peligrosa discutiendo sus características principales, sin embargo, ahora se podría preguntar: ¿qué factores hacen que una persona u organización sea más vulnerable al *phishing*? De acuerdo a [13] existen tres principales factores a tomar en cuenta que determinan lo vulnerable que es una persona u organización a ataques de *phishing*.

- **Factores individuales**

  Factores como el riesgo y la vulnerabilidad percibida ante el *phishing* influyen en la precaución que un individuo tendrá ante estos ataques, reduciendo su susceptibilidad a estos ataques al reconocer las posibles consecuencias negativas. Sin embargo, la percepción de una recompensa, como ahorrar tiempo al realizar tareas cotidianas,

puede aumentar la probabilidad de caer en *phishing* al priorizar la eficiencia sobre la seguridad. El miedo a la vergüenza por la exposición de información privada puede ser un factor disuasorio, motivando al individuo a ser más cauteloso con sus acciones. Por otro lado, los hábitos automáticos o memoria muscular, como hacer clic en enlaces de correos electrónicos sin previamente analizar si es seguro, pueden incrementar el riesgo de ser víctima de *phishing* al actuar sin conciencia plena de las consecuencias.

- **Factores organizacionales**

  Abarca todos aquellos factores en las organizaciones que afecten a la exposición ante el *phishing*, se argumenta que normas subjetivas y creencias normativas tienen un impacto positivo en la actuación de los que conforman la organización ante intentos de *phishing*, adicionalmente, la creación de contra-medidas como estándares de seguridad, políticas y reglamentos tienen un impacto positivo en educar y concientizar a los miembros de la organización sobre el *phishing* y la ciberseguridad en general.

- **Factores tecnológicos**

  Factores que comprenden elementos tecnológicos que pueden aportar hacia reducir posibles brechas de seguridad provenientes de ataques de *phishing*. Se menciona en el estudio que los controles y contra-medidas técnicas tienen un rol esencial en evitar ciberataques. Las contra-medidas de detección proveen la capacidad de monitorear y realizar revisiones sobre la actividad de usuarios. Por otro lado, las contra-medidas preventivas tienen un rol en el que se le provee al usuario con protección ante posibles amenazas como *malware* o *phishing*.

5.1.2. Taxonomía del *phishing*

Ahora que se conoce a detalle que es el *phishing*, sus riesgos potenciales y cuál es su alcance, es necesario poder entender cuáles son los componentes esenciales de un ataque de *phishing* y como se diferencia de otros ciberataques.

Con el fin de ilustrar con mayor claridad estos ciberataques [9] propone el siguiente ejemplo: Un atacante planea atacar una organización con el objetivo de utilizar de manera ilícita información sensible, se puede clasificar como información sensible a toda aquella información que un individuo u organización no quiere que se haga pública porque les podría perjudicar de alguna forma, sus pasos a seguir son los siguientes:

1. Planeamiento y reconocimiento:

   el atacante identifica la organización o grupo de personas sobre la que lanzara el ataque. Se establece una estrategia para hacer un reconocimiento general sobre el objetivo, durante este tiempo se descubrirán vulnerabilidades que le podrían resultar útiles al atacante al momento de realizar su ataque.

2. Construcción de un sitio falsificado:

   durante esta fase se crea una página web falsificada que se vea idéntica a la página que se busca replicar, esta página web contiene ciertas modificaciones que le serán útil al atacante al momento de recolectar la información de sus víctimas.

3. Esparcimiento del *phishing*:

   se elige un medio o vector de ataque apropiado para la distribución del acceso a la página falsificada, comúnmente se podría distribuir por medio de mensaje de texto o correo electrónico.

4. Explotación:

   el enlace malicioso redirige al usuario al sitio falsificado donde muy probablemente otorguen datos sensibles como credenciales al creer que es un sitio en el que pueden confiar.

5. Recolección de datos:

   los atacantes al tener acceso a la base de datos de la página web pueden acceder a la información que las víctimas le otorgaron por medio del ataque. En esta fase el atacante ya cumplió con su objetivo principal.

6. Exfiltración:

   el atacante intentara cubrir toda evidencia que haya dejado del ataque, es decir que intentara borrar el sitio falsificado, cuentas de correo, dominios utilizados, etc., esta fase tiene como objetivo hacerle más difícil de rastrear.

Posterior a las fases descritas, el atacante puede proceder a utilizar ilícitamente las credenciales obtenidas durante el ataque. Estas credenciales podrían ser utilizadas con el objetivo de obtener acceso a sistemas dentro de la organización con el objetivo de lanzar un ataque que le lograra perjudicar aún más a la empresa [14].

El estudio [11] propone algunas categorías para la clasificación del *phishing*, dentro de las cuales se encuentran:

1. Ataques sobre un solo elemento:

   ataques que consisten en atacar a un solo individuo de interés para el atacante, que posiblemente tenga accesos o credenciales a otros recursos que le serán útiles al atacante.

2. Ataques sobre múltiples elementos:

   se realiza un ataque sobre varios individuos dentro de una organización, el enfoque de este ataque siendo que el atacante se hace pasar por una persona en la que varios individuos pueden confiar.

3. Ataques consecutivos:

   se realizan ataques consecutivos sobre una organización de manera masiva con el objetivo de infiltrarse, este tipo de ataque puede ser más fácilmente detectado que los anteriormente mencionados.

4. Ataques aleatorios:

   ataque realizado sobre varios individuos sin conexión alguna con el objetivo de aprovecharse de las credenciales de quien caiga por el ataque.

Asimismo, esta investigación describe algunas técnicas comunes que utilizan los atacantes con el objetivo de poder lanzar ataques de manera efectiva:

- Suplantación de sitios web:

  el atacante puede engañar a las víctimas mediante la suplantación de un sitio web de confianza utilizando una réplica falsificada. Comúnmente los atacantes únicamente generan la página de login de usuario, ya que de esta manera es más difícil de detectar por detección anti-*phishing*.

- Suplantación de correo electrónico:

  esta técnica se centra en engañar a la víctima en creer que el correo electrónico recibido es de confianza, esto la hará más propensa a ingresar a un URL de un sitio web malicioso o incluso llegar a descargar archivos maliciosos adjuntos.

- *Spear phishing*:

  la técnica de *spear phishing* se centra en dirigir el ataque a una única persona, por lo general esta suele tener un puesto de alta gerencia o de alto interés para el atacante. El atacante monitorizará constantemente las actividades de la víctima con el objetivo de poder engañarla de la manera más efectiva posible.

- Envenenamiento de DNS (*Domain Name System*):

  con esta técnica el atacante toma control del servidor DNS y envenena la caché de la DNS, esto tiene como objetivo redirigir el tráfico desde una página oficial hacia la página suplantada del atacante.

- *phishing* a través de redes sociales:

  este ataque engloba diferentes técnicas que son altamente efectivas en las redes sociales, la suplantación de identidad a través de perfiles falsos o robados es altamente efectivo en engañar a las víctimas, en conjunto con las falsificaciones y enmascaramiento de URL maliciosas suelen ser las técnicas de *phishing* más utilizadas dentro del contexto de las redes sociales.

Las motivaciones de los atacantes pueden ser variadas, puede que algunos atacantes vean un beneficio económico al vender información de organizaciones o individuos de interés al realizar los ataques, el robo de identidad es otra de las razones más comunes, por medio de esto pueden llevar a cabo acciones ilícitas. Adicionalmente, el obtener fama y reconocimiento es otra de las razones por las cuales los cibercriminales pueden llegar a realizar este tipo de ataques [14]. A partir de estos ataques, algunos de los efectos negativos que las organizaciones o individuos pueden sufrir son los siguientes:

- El *phishing* puede revelar información personal de la víctima y en ocasiones puede representar grandes pérdidas financieras.
- El ataque puede generar una impresión negativa hacia las interacciones en internet y puede incluso destruir la reputación que tienen algunas plataformas digitales.
- El uso de plataformas digitales bancarias, *e-commerce* y redes sociales puede disminuir debido la pérdida de confianza en la seguridad que estas ofrecen.

- Los ataques dirigidos a empresas pueden generar desconfianza en su clientela, afectar negativamente su imagen pública y reputación. En conjunto puede generar pérdidas económicas significativas para la empresa.

5.1.3. ¿Cómo se puede mitigar?

En todas sus manifestaciones y formas, el *phishing* se ha convertido en una amenaza difícil de mitigar, agravado por el hecho de que estos ataques suele ser un vector para lanzar ciberataques que puedan llegar a empeorar una situación ya de por sí delicada. Tomando en cuenta su constante evolución y sus consecuencias, hay mucha investigación hecha al respecto de como contrarrestar el *phishing*. De acuerdo a [15], existen tres principales categorías con las que se pueden clasificar los diferentes métodos de mitigación *phishing*:

- Sistemas anti-*phishing*:

  se refiere a toda herramienta digital que se utilice para mitigar el *phishing*. Puede incluir, pero no se limita a *software*, modelos de *machine learning* y diseño preventivo en herramientas

- Marcos de trabajo:

  esta categoría abarca todo lo que se refiere a guías y estándares que dan orientación sobre actividades a realizar para mitigar el *phishing*.

- Mitigación centrada en el usuario:

  engloba estrategias con las que influenciar al usuario a que obtenga una conciencia sobre la seguridad de la información en general y que modifique su comportamiento de tal manera que pueda lidiar adecuadamente ante ataques de *phishing*.

Los sistemas anti-*phishing* suelen ser una manera efectiva para lidiar con el *phishing*, su objetivo principal es poder clasificar de manera automática las amenazas y tener una mínima intervención por parte del usuario. La investigación [14] expone que existen varios tipos de sistemas basados en diferentes técnicas de detección, las cuales se discutirán a continuación:

- Detección basada en *machine learning*:

  utiliza técnicas de *machine learning* para entrenar modelos de aprendizaje a través de diferentes algoritmos para poder clasificar de manera automática diferentes características que pueden detectar el *phishing*, estas características pueden ser extraídas de texto, URL (*Uniform Resource Locator*), contenido HTML (*Hypertext Markup Language*), entre otros.

- Detección basada en buscador:

  estos sistemas de detección se basan en utilizar herramientas de búsqueda como Google para poder detectar sitios *phishing* por medio de la búsqueda, ya que por lo general los sitios de *phishing* no estarán indexados dentro de los resultados de búsqueda de Google, mientras que los sitios legítimos sí lo estarán.

- Detección basada en listas:

  se basa en buscar dentro de listas si un sitio es legítimo o no, por lo general se utilizan listas blancas que contienen URL legítimos, mientras que las listas negras contienen URL que han sido detectados previamente como falsos o maliciosos.

- Detección basada en elementos visuales:

  los sistemas basados en elementos visuales basan sus detecciones en características visuales como imágenes, contenido textual y archivos CSS en páginas web. La finalidad de estos sistemas es poder comparar elementos visuales entre de sitios web originales con sitios web que podrían ser maliciosos o estar suplantando a otra página.

- Detección en dispositivos móviles:

  los sistemas basados en dispositivos móviles aplican varias de las técnicas anteriormente mencionadas para detectar *phishing*. Adicionalmente, se utilizan técnicas para la detección de *smishing*, el cual es una categoría de *phishing* que tiene como vector de ataque los mensajes de texto. También se busca detectar el *vishing* que es una categoría de *phishing* que utiliza como vector los mensajes de voz o llamadas telefónicas para llevar a cabo el ataque.

Los estándares y marcos de trabajo en ciberseguridad son de las medidas más utilizadas en especial dentro de empresas, su uso intenta orientar a la organización a tener un control más riguroso sobre los procesos que se realizan de manera física y digital. Los estándares de ciberseguridad son aplicables a la mayoría de organización sin importar su tamaño o sector de la industria en el que opera. Estos estándares pueden ser utilizados como una guía para establecer adecuadamente sistemas que puedan mantener un estándar alto en el manejo seguro de la información dentro de las organizaciones. Los estándares ISO/IEC 27000, 27001 y 27002 son internacionalmente reconocidos y adoptados, se les conoce como el lenguaje común de las organizaciones en lo que refiere a la seguridad de la información, lo cual es testimonio de su efectividad ante vulnerabilidades y amenazas [9].

Las estrategias centradas en la mitigación por medio del usuario son unas de las más importantes, al ser el usuario uno de los puntos más vulnerables que un atacante puede aprovechar por medio del *phishing*. Esta estrategia puede ser una de las más difíciles de implementar adecuadamente, ya que como se discutió anteriormente, hay muchos factores humanos a tomar en cuenta para que esta estrategia sea exitosamente implementada. Una estrategia común es la generación de conciencia de la ciberseguridad en los individuos o miembros de una organización a través de programas de entrenamiento o charlas informativas. Educar a los usuarios sobre los riesgos y consecuencias de los ciberataques es clave para poder generar un ambiente más seguro en una plataforma u organización, ya que generando conciencia sobre este problema los usuarios podrán tomar decisiones mejor informadas y con mayor cuidado, disminuyendo el riesgo de que ataques de *phishing* lleguen a afectar las personas u organizaciones [12].

5.2. ¿Qué es detección basada en listas?

La estrategia por excelencia de los navegadores como Microsoft Edge, Firefox y Google Chrome, la utilización de detección basada en lista es una de las técnicas de detección más utilizadas dentro de la detección de *phishing* por medio del URL [5], esta técnica se compone de dos métodos principales:

- *Blacklisting*:

  también conocido como lista negra, este método utiliza una lista con URL maliciosos extraídos de diversas fuentes con la principal función de prevenir el acceso a estos sitios que han sido reportados como maliciosos.

- *Whitelisting*:

  el método de lista blanca es todo lo contrario a la lista negra. En este caso, la lista está compuesta de URL que han sido confirmados como seguros y/o tienen un número alto de visitas de usuarios.

Ambos métodos [1] pueden ser utilizados en conjunto para obtener una precisión alta sobre ataques de *phishing* ya conocidos mediante su URL, llegando a funcionar como una especie de filtro. Esta técnica de detección tiene tanto ventajas como desventajas. A continuación, se exploraran a detalle en las siguientes secciones.

5.2.1. Ventajas

El tiempo de acceso a las listas es rápido, lo que hace de todo el proceso de detección un proceso muy eficiente a comparación de otras técnicas, lo que le hace ideal para aplicaciones livianas que requieran de un tiempo de respuesta corto. Es el medio más eficiente para detectar *phishing* que utiliza como vector principal una URL [5].

5.2.2. Desventajas

Por otro lado, esta técnica cuenta con algunas deficiencias que le podrían hacer poco eficiente en algunos casos. Es, especialmente, vulnerable ante ataques de día cero, es decir, que le es casi imposible el detectar un URL de *phishing* que nunca ha sido registrado con anterioridad. Las listas requieren de una tasa constante de actualización para mantenerse al día con todas las posibles amenazas, lo que significa que la efectividad de la técnica depende exclusivamente de la calidad y lo reciente de los datos [5].

5.3. ¿Qué es *machine learning*?

El *machine learning* es la ciencia que estudia los algoritmos y modelos estadísticos de aprendizaje automático que son utilizados como una forma de realizar predicciones para varias aplicaciones, como procesamiento de imágenes, análisis predictivo, pronósticos, etc.

Considerada como parte de la inteligencia artificial, es una forma en la que una computadora puede obtener cierto raciocinio y capacidad de tomar decisiones de manera independientemente, siendo influenciada casi exclusivamente de los datos de entrenamiento que se le proporcionen. El uso de técnicas de *machine learning* es especialmente útil al momento de realizar trabajos que se beneficien de tener un grado de automatización, es decir, que, podría realizar tareas o chequeos de manera independiente [16]. Hay muchas categorías de problemas que pueden ser resueltos de manera eficiente con un enfoque basado en el *machine learning*, desde problemas de clasificación, donde se debe determinar si un cierto valor de entrada pertenece a una u otra clase de clasificación, hasta problemas de detección de anomalías, donde a través de un análisis de patrones y cambios se pueden detectar anomalías que pueden determinar si se está corriendo un riesgo durante la realización de un proceso [17].

5.3.1. ¿Qué es detección basada en *machine learning*?

Las técnicas de detección basadas en *machine learning* entrenan modelos de aprendizaje con el uso de ciertas características que pueden apuntar el modelo a hacer una predicción sobre si lo que está detectando es un intento de *phishing*. Estas características son comúnmente extraídas de diferentes fuentes como el contenido de una página web, el URL, el texto de un mensaje, imágenes, etc. [14] La característica principal de las técnicas de *machine learning* es la habilidad de poder crear modelos que se adapten a diferentes categorías de problemas, la detección de *phishing* en la mayoría de casos suele ser un problema de clasificación, por lo que por medio del aprendizaje de patrones en los datos y características de entrenamiento un modelo podría aprender a diferenciar entre un valor de entrada que contenga *phishing* y otro que sea legítimo. Adicionalmente, estos modelos pueden ser evaluados de manera crítica y objetiva por medio de diferentes mediciones sobre el desempeño del modelo, como la precisión, alcance y exactitud [18].

5.3.2. Ventajas

Este método se adapta fácilmente a diferentes tipos de aplicaciones, por lo cual puede utilizarse para detectar una variedad grande de ataques de *phishing*, haciéndolo uno de los métodos de detección más utilizados debido a la versatilidad con la que se puede adaptar a diferentes tipos de datos. La precisión con la que los modelos es considerablemente alta a comparación de otros métodos como la detección por listas, lo cual en conjunto con un tiempo corto de detección hace de este método una opción formidable para detectar *phishing* de manera eficiente [5].

5.3.3. Desventajas

Uno de los desafíos más grandes al utilizar este método es el sobreajuste, este es un problema común dentro de los modelos de *machine learning* que significa que el modelo ha aprendido como predecir el resultado correcto de los datos de entrenamiento, pero no ha aprendido los patrones como para hacer estas predicciones con datos reales, esto resulta en

un porcentaje de precisión alta durante el proceso de entrenamiento del modelo, pero un porcentaje extremadamente bajo al predecir datos reales o que sean diferentes a los datos de entrenamiento [19].

Otro posible reto al momento de desarrollar un modelo de detección es el tiempo de entrenamiento, ya que dependiendo del tamaño del conjunto de datos y tipo de características podría tomar un tiempo considerable. Además, el hecho de entrenar un modelo puede consumir bastantes recursos si el conjunto de datos es lo suficientemente grande o con características robustas, lo cual es un punto a tomar en cuenta si se quisiera utilizar este método de detección [20].

5.4. ¿Qué es una extensión web?

Una extensión web es un programa complementario diseñado para ampliar las funcionalidades básicas de un navegador, creando un ecosistema donde desarrolladores puedan publicar contenido dirigido a distintos tipos de usuarios y casos de uso. Comúnmente son accesibles a través de plataformas digitales que permiten a los usuarios descargarlas y utilizarlas para mejorar su experiencia de navegación. Su versatilidad permite que puedan ser desarrolladas para una amplia gama de aplicaciones, desde bloqueadores de anuncios hasta herramientas para visualizar archivos, lo que las convierte en herramientas esenciales y de uso diario para muchos usuarios de los navegadores web [21].

5.4.1. ¿Cómo puede utilizarse en la mitigación de *phishing*?

Las extensiones web pueden utilizarse como una herramienta efectiva contra el *phishing*, como se mencionó anteriormente, su versatilidad es una de sus grandes fortalezas por lo que se puede adaptar a diferentes tipos de usos en este rubro, desde herramientas centradas en el aprendizaje y concientización sobre el *phishing* y otros tipos de ataques cibernéticos hasta herramientas que realizan detección y acciones de mitigación de manera automática [4].

El objetivo principal al utilizar una extensión web como medida de mitigación es poder ofrecerle al usuario una herramienta del lado del navegador que en ningún momento comprometa su privacidad, que sea intuitiva, segura y eficiente. Idealmente, este tipo de solución debe de ser liviana y tener un tiempo de respuesta corto, por lo que no requerirá de mucho poder computacional por parte de la máquina local, aspecto que hace de este tipo de solución mucho más accesible a la mayoría de usuarios [1].

5.5. Sistemas de detección de *phishing*

Los sistemas de detección basados en *software* tienen dos métodos principales para guiar el enfoque de su desarrollo, el método tradicional, que utiliza técnicas de detección basadas en listas y el método automático, cuyo funcionamiento gira en torno a la detección por medio del uso híbrido de varias técnicas como la detección por medio de la heurística, listas, modelos *machine learning*, entre otros. El uso del método automático puede aumentar la

precisión en la detección de *phishing* debido a la utilización de varias técnicas de manera híbrida para realizar un análisis en tiempo real sobre el contenido sospechoso [22].

5.5.1. Implementaciones existentes

Hay una gran variedad de enfoques diferentes sobre los cuales los sistemas de detección pueden ser desarrollados. A continuación, se mencionan algunos que proponen diferentes combinaciones de técnicas para detectar ataques de *phishing*.

- MailTrout:

  utiliza técnicas de detección por *machine learning* para crear un sistema de protección de *phishing* por correo electrónico, este sistema es presentado por medio de una extensión web en la cual el usuario puede seleccionar un correo electrónico para posteriormente ser analizado por medio del modelo de *machine learning* [4].

- SpoofCatch:

  propone un sistema que utiliza técnicas de detección basadas en heurística que comparan la secularidad visual de páginas web que podrían ser parte de un ataque de *phishing* con páginas web genuinas. De igual forma, se propone analizar los URL y contenido HTML de las páginas de login para detectar algún tipo de contenido que podría resultar sospechoso. Se propone una extensión web como prueba de concepto de este sistema, principalmente porque además de proteger al usuario constantemente cuando utiliza el navegador, puede mantener una experiencia de usuario amigable [7].

5.5.2. Ventajas de las técnicas combinadas

La utilización de técnicas de detección de manera combinada en los sistemas de detección está siendo utilizada cada vez más debido a las ventajas que esta propone, a continuación se mencionaran algunas de estas:

- El uso de múltiples técnicas dentro de una misma herramienta puede ser útil para tratar diferentes tipos de ataques de *phishing* y tener una cobertura mayor sobre los diferentes enfoques que un ataque puede tener [23].
- La eficiencia y velocidad de las herramientas puede aumentar, ya que al tener múltiples medios por los cuales el *phishing* puede ser detectado el proceso de detección se agiliza debido a que se puede detectar prematuramente por medio de alguna técnica de detección que requiera de poco tiempo para completar su análisis [1].

CAPÍTULO 6

Metodología

6.1. Recolección de datos y fuentes de información

Las fuentes principales utilizadas para el desarrollo de los conjuntos de datos que fueron utilizados para realizar consultas en tiempo real en los sistemas de detección de la herramienta se obtuvieron a través de las siguientes plataformas:

- Kaggle: es una plataforma que ofrece herramientas y recursos orientados al aprendizaje automático e inteligencia artificial
- Cisco Umbrella: es un servicio de seguridad en la nube que protege contra amenazas como el *malware*, *ransomware* y *phishing*. Su listado "*top 1 million*.[es] una clasificación de los sitios web más confiables a nivel mundial
- Google Safe Browsing: servicio que se dedica a la recopilación de sitios peligrosos, su API permite a desarrolladores consultar la base de datos de Google para verificar si una URL está asociada con algún tipo de contenido malicioso
- Open PageRank: herramienta que ofrece un valor de PageRank estimado para un sitio web, utilizando datos disponibles al público para realizar esta estimación. Las métricas de PageRank evalúan la relevancia y autoridad de las páginas web basándose en la estructura de enlaces externos dentro de las páginas web

Para la utilización de fuentes con el fin de realizar consultas para la detección basada en listas, se tuvieron en cuenta los siguientes parámetros para asegurar que los datos fueran confiables y accesibles:

- Las fuentes utilizadas deben de ser de acceso abierto y que puedan utilizarse de manera gratuita para de esta manera asegurar su disponibilidad continua.
- Las fuentes deben de ser mantenidas y actualizadas regularmente para asegurar que la información refleje las amenazas actuales.
- Las fuentes deben provenir de entidades reconocidas y confiables, como organizaciones reconocidas, instituciones académicas o proyectos colaborativos respaldados por la comunidad.
- Se prioriza el uso de fuentes que ofrezcan datos estructurados y en formatos estandarizados que faciliten la integración con el proyecto en desarrollo.

Para garantizar que los conjuntos de datos utilizados para la detección basada en *machine learning* sean confiables y puedan generar modelos efectivos, se tuvieron en cuenta los siguientes criterios al seleccionar y recopilar las fuentes de datos:

- Los conjuntos de datos deben de ser de acceso abierto y estar disponibles de tal forma que permitan su uso gratuito, redistribución y modificación. Esto asegura que los datos puedan ser utilizados y adaptados según el proyecto lo requiera.
- Los conjuntos seleccionados deben de ser recientes de manera que reflejen correctamente las tendencias y amenazas recientes.
- Se seleccionaron fuentes que ofrecieran una buena diversidad de datos, como diferentes tipos de amenazas y características importantes dentro del rubro de cada uno de los conjuntos (URL, mensaje de texto, contenido web). La diversidad es esencial para tener un entrenamiento de modelos capaces de generalizar una amplia variedad de amenazas.
- Las fuentes deben ofrecer datos en formatos estandarizados y estructurados de manera que pueda facilitar la integración de los datos dentro del entrenamiento de los modelos.

6.2. Exploración y partición de los conjuntos de datos

Para cada de uno de los conjuntos de datos se realizó un análisis exploratorio de datos con el objetivo principal de identificar patrones e indicios de relaciones existentes entre las características del conjunto de datos.

6.2.1. URL

En el caso del conjunto de datos [24] utilizado para la detección de *phishing* por medio de URL , este contiene 549,346 entradas y dos columnas, URL y Label. La columna Label es la columna de predicción, con dos categorías:

- Good: indica que el URL no contiene contenido malicioso y no es un sitio de *phishing*.

- Bad: indica que el URL contiene contenido malicioso y es un sitio de *phishing*.

Como se puede observar en la figura 1 la distribución de datos se encuentra en una proporción de 392,924 URL seguros y 156,422 URL inseguros.

La columna URL como su nombre lo indica, contiene el URL en formato de cadena de texto.

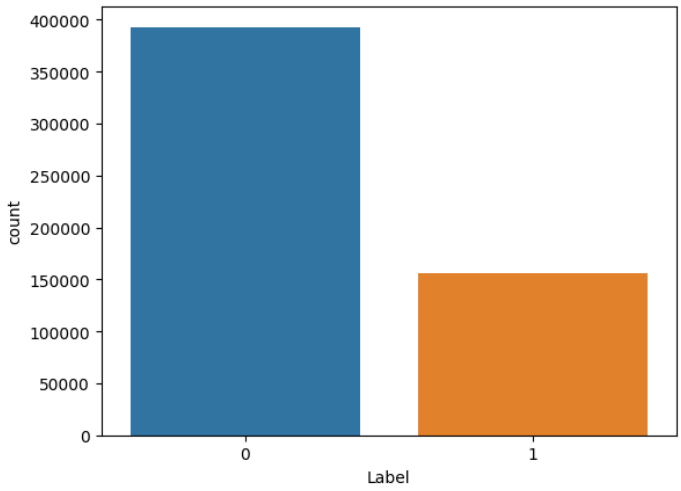

Figura 1: Distribución de los datos de URL

Como primer paso del preprocesamiento se realizó un cambio en los indicadores de Label, pasando de Good and Bad hacia 0 y 1 respectivamente, posteriormente a través de técnicas de ingeniería de características se crearon las siguientes características con el propósito de mejorar la capacidad predictiva del modelo y facilitar su aprendizaje al momento de realizar el entrenamiento:

1. ngrams: representa la puntuación media de los n-grams del URL en comparación con un diccionario de palabras comunes en inglés. Esto mide cuánto se parece el URL a una palabra o secuencia de caracteres típica en inglés
2. entropy: la entropía de Shannon del URL, que mide la incertidumbre o aleatoriedad en la cadena de caracteres del URL. URL con alta entropía suelen ser más sospechosas, ya que pueden indicar contenido generado aleatoriamente
3. entropyRelative: la entropía relativa del URL, calculada como la divergencia de Kullback-Leibler entre la distribución de caracteres en el URL y una distribución esperada basada en dominios conocidos. Esto ayuda a identificar URL que difieren significativamente de los patrones normales
4. vowel-cons: la relación entre vocales y consonantes en el URL. Una relación inusual podría indicar que el URL ha sido manipulada para parecer legítima, pero en realidad es maliciosa
5. firstDigitIndex: el índice del primer dígito que aparece en el URL. Esto podría ser útil para detectar URL que contienen números inusuales en posiciones específicas

6. length: la longitud total del URL, que puede ser un indicador de URL sospechosas, ya que URL de *phishing* a menudo son más largas para ocultar información maliciosa

7. digits: la cantidad de dígitos presentes en el URL. Un número elevado de dígitos puede ser un indicador de URL generadas o manipulación

8. ip: indica la presencia (1) o ausencia (0) de una dirección IP en el URL. URL con direcciones IP en lugar de nombres de dominio suelen ser sospechosas

9. special: indica la presencia (1) o ausencia (0) de caracteres especiales en el URL. La presencia de estos caracteres puede ser un signo de manipulación o intento de evitar filtros

10. port: indica la presencia (1) o ausencia (0) de un número de puerto en el URL. Los puertos en las URL pueden ser utilizados para propósitos maliciosos

11. subdomain: el número de subdominios presentes en el URL. Un número elevado de subdominios puede ser indicativo de una estructura de URL diseñada para confundir o esconder su verdadera intención.

12. common: el número de términos comunes en el URL, como "www", "com", "net", etc. Un conteo inusual puede ayudar a identificar URL sospechosas

13. hyphen: indica la presencia (1) o ausencia (0) de guiones en el URL. Los guiones son frecuentemente utilizados en URL de *phishing* para crear variantes de nombres de dominio legítimos

14. doubleHyphen: indica la presencia (1) o ausencia (0) de dobles guiones en el URL. Los dobles guiones son menos comunes y pueden ser un signo de manipulación

15. shortening: indica la presencia (1) o ausencia (0) de un servicio de acortamiento de URL en el URL. Los acortadores de URL son frecuentemente utilizados para ocultar el destino final del URL

16. abnormal: indica la presencia (1) o ausencia (0) de subdominios anormales en el URL. Subdominios como "login", "secure", "admin", etc., pueden ser utilizados en URL de *phishing* para engañar a los usuarios

17. Label: la etiqueta que indica si el URL es legítima (good = 0) o maliciosa (bad = 1)

Posteriormente, debido al desbalance de clases encontrado en el conjunto se realizó un balanceo a traves del *under-sampling*, disminuyendo la cantidad total de URL pertenecientes a la clase de mayor frecuencia (0), el conjunto resultante obtuvo una proporción de clases aproximada a 50/50 como se puede apreciar en la figura 2.

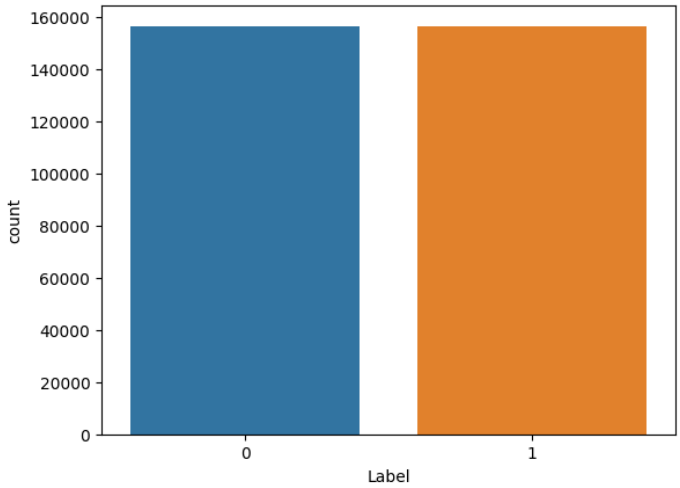

Figura 2: Distribución de los datos de URL - Post-balanceo

6.2.2. Mensajes de texto

El conjunto de datos [25] empleado para el entrenamiento del modelo de detección de *phishing* por mensajes de texto, contiene 82,486 entradas, conteniendo dos columnas; *text-combined* y label. Este conjunto fue originalmente compilado por investigadores para estudiar patrones y técnicas utilizadas en correos electrónicos de *phishing*. Combina correos de varias fuentes, incluyendo los datasets Enron, Ling, CEAS, Nazario, Nigerian Fraud y SpamAssasin. Los datos proporcionados a través de la columna text-combined incluyen tanto el asunto como el cuerpo del mensaje así como el contexto (remitente, destinatario y fechas). El conjunto incluye 42,891 correos clasificados como maliciosos y 39,595 correos legítimos [26], la distribución de estos datos puede ser observada en la figura 3, donde se observa una distribución similar entre las dos clases.

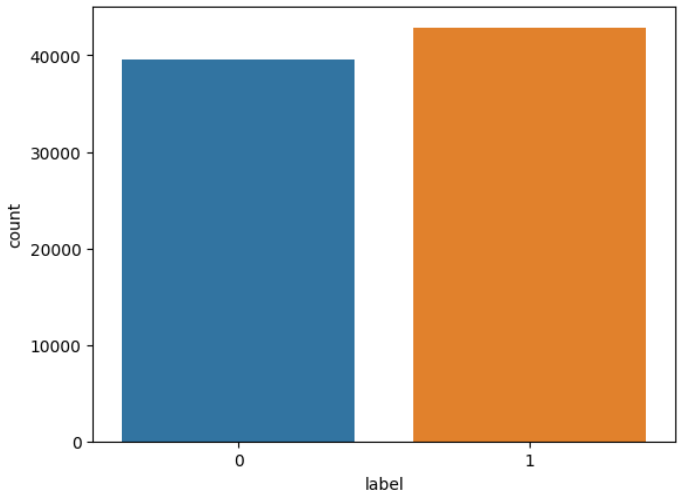

Figura 3: Distribución de los datos de mensajes

A partir de los datos del conjunto se aplicaron diferentes técnicas para el preprocesamiento, de tal manera que el conjunto de datos ofrezca datos de entrenamiento de manera óptima, los pasos a seguir fueron los siguientes:

1. Conversión a minúsculas: se convierte todo el texto a minúsculas para evitar inconsistencias en la lectura de datos.
2. Eliminación de acentos y caracteres especiales: Se eliminan acentos y caracteres especiales para poder obtener únicamente el contenido a manera de caracteres de texto comunes
3. Corrección de contracciones: se expanden las contracciones, es decir que palabras como "*don't*" se convierten en do not, con el fin de mejorar la consistencia de los datos
4. Tokenización: se divide el texto en palabras/*tokens* individuales
5. Eliminación de *stopwords*: se eliminan palabras comunes como "*the*", "*and*", etc., ya que estas no aportan ningún valor semántico a los mensajes
6. Stemming: se reducen las palabras a su raíz o forma base para mejorar la consistencia (por ejemplo, "*running*" a "*run*")
7. Lematización: similar a *stemming*, pero lleva las palabras a su forma canónica (Por ejemplo, "*better*" a "*good*").
8. Eliminación de palabras cortas: Se eliminan las palabras con menos de 4 caracteres, que suelen tener un bajo valor semántico
9. Normalización del texto: se unifican las palabras procesadas en una cadena de texto
10. Transformación TF-IDF: se aplica el vectorizador TF-IDF al texto preprocesado, convirtiendo el texto en una matriz de características donde cada fila representa una entrada del conjunto de datos original y cada columna representa una palabra o *token*. Los valores en la matriz son el punteo TF-IDF (*Term Frequency - Inverse Document Frequency*), que indican la importancia de cada palabra en el contexto de los mensajes basándose en su frecuencia de uso.

La importancia de conocer la frecuencia de ciertas palabras radica en que permite identificar patrones y tendencias en el texto. Al analizar de esta manera el texto se pueden identificar patrones lingüísticos que pueden diferenciar mensajes legítimos de aquellos que pueden ser maliciosos, y como se observa en las figuras 4 y 5, algunas palabras identificadas con mayor influencia en el texto son "*plea*", "*watch*", "*workz* "*would*". El uso del TF-IDF en particular puede ayudar al modelo a destacar palabras que pueden ser importantes para detectar el *phishing* a través de la identificación de diferentes señales semánticas y contextuales [27].

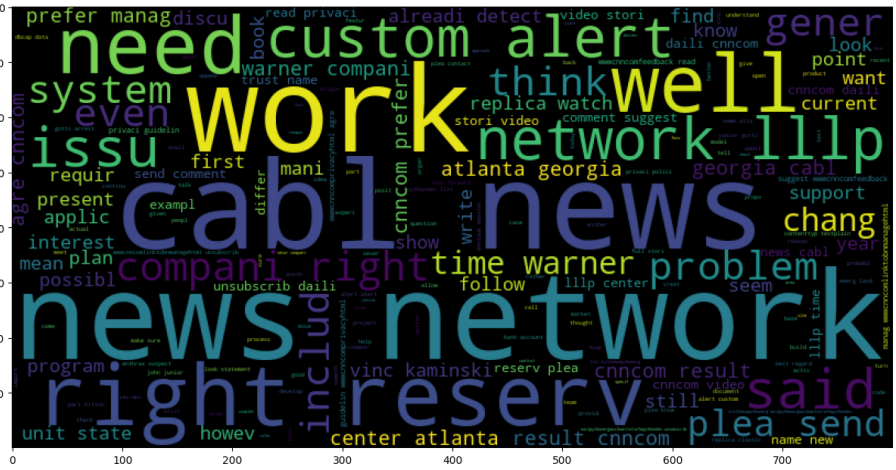

Figura 4: Nube de palabras

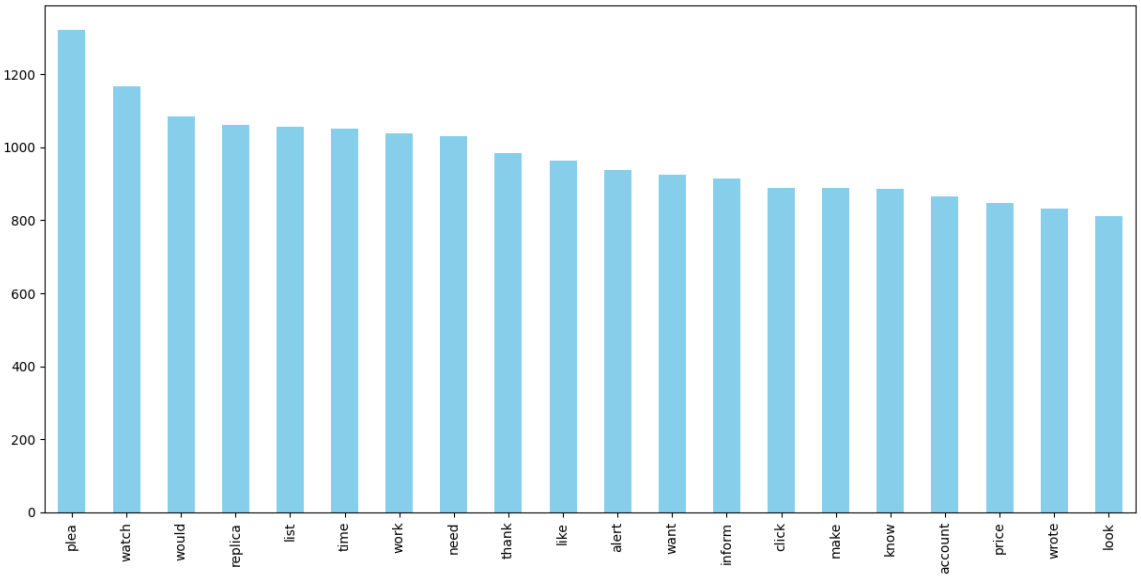

Figura 5: Frecuencia de palabras

6.2.3. Contenido web

Para la detección de contenido web a través de *machine learning*, se utilizó el conjunto de datos [28], originalmente incluía 48 características extraídas de 10,000 páginas web, divididas equitativamente entre páginas legítimas y páginas maliciosas como se puede observar en la figura 6.

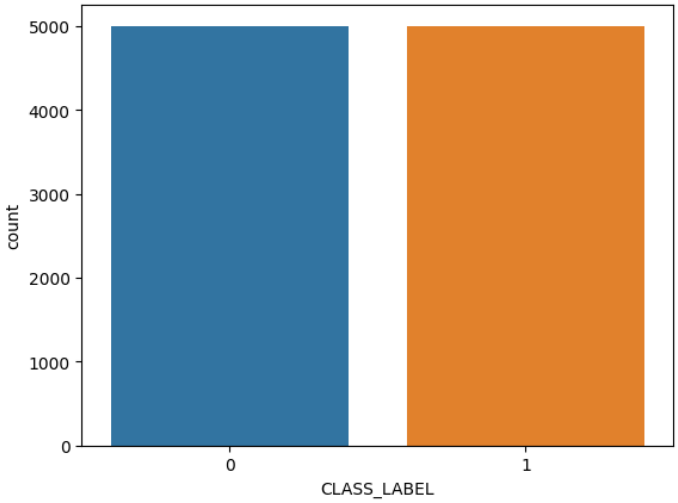

Figura 6: Distribución de los datos de contenido web

Debido al enfoque que se le dio al contenido web fue necesario preprocesar y limpiar el conjunto de datos de tal forma contuviera características relevantes dentro de este contexto. Por lo que se tomaron en cuenta las siguientes características para el conjunto de datos final:

1. PctExtHyperlinks: mide el porcentaje de enlaces externos en el código fuente HTML
2. PctExtResourceUrls: cuenta el porcentaje de URL de recursos externos en el código fuente HTML
3. ExtFavicon: verifica si el favicon se carga desde un dominio diferente al dominio del URL
4. InsecureForms: verifica si el atributo de acción del formulario contiene un URL sin el protocolo HTTPS, lo que indica un formulario inseguro
5. RelativeFormAction: verifica si el atributo de acción del formulario contiene un URL relativa
6. ExtFormAction: verifica si el atributo de acción del formulario contiene un URL que apunta a un dominio externo
7. AbnormalFormAction: verifica si el atributo de acción del formulario contiene valores anormales como "*about*", "#" etc.
8. PctNullSelfRedirectHyperlinks: mide el porcentaje de campos de hipervínculo que contienen un valor vacio, un valor de auto-redirección, el URL actual o algún valor anormal
9. FrequentDomainNameMismatch: verifica si el nombre de dominio más frecuente en el código HTML no coincide con el nombre de dominio del URL
10. FakeLinkInStatusBar: verifica si el código fuente HTML contiene un comando "*onMouseOver*" que muestra un enlace falso en la barra de estado

11. RightClickDisabled: verifica si el código fuente HTML contiene un *script* que deshabilite el clic derecho

12. PopUpWindow: verifica si el código fuente HTML contiene un comando para lanzar ventanas emergentes

13. SumbitInfoToEmail: verifica si el código HTML contiene la función "*mailto*"

14. IframeOrFrame: verifica el uso de etiquetas "*iframe*" o "*frame*"

15. MissingTitle: verifica si la etiqueta de título está vacía en el código fuente HTML

16. ImagesOnlyInForm: verifica si el formulario en el código HTML contiene solo imágenes y ningún texto

17. PctExtResourceUrlsRT: cuenta el porcentaje de URL de recursos externos en el código fuente HTML de la página web

18. AbnormalExtFormActionR: verifica si el atributo de acción del formulario contiene un dominio externo

19. ExtMetaScriptLinkRT: cuenta el porcentaje de etiquetas "*meta*", "*script*" y "*link*" que contiene un URL externa en sus atributos

20. PctExtNullSelfRedirectHyperlinksRT: mide el porcentaje de hipervínculo en el código fuente HTML que utilizan nombres de dominio diferentes, comienzan con "#" o usan "*JavaScript::void(0)*"

6.3. Desarrollo de capa basada en listas

Para la capa de detección basada en listas se plantearon tres listas por las cuales un URL puede ser analizado para detectar si dirige a un sitio legítimo o malicioso. Las listas planteadas fueron:

- Lista blanca: consiste de un listado de sitios comúnmente visitados, verificados como seguros y generalmente de uso cotidiano, por lo cual se utiliza como primer chequeo dentro de esta capa de detección. Si un URL se encuentra dentro de la lista blanca, se considera confiable y no se realizan ningún análisis adicional para detectar si es malicioso. La fuente utilizada para esta lista fue [29], que contiene una lista con el primer millón de dominios más buscados globalmente a través de los servicios de DNS de Cisco Umbrella.

- Lista negra: esta lista contiene un conjunto de URL que han sido identificados previamente como maliciosos, fraudulentos o asociados con actividades sospechosas referentes al *phishing*. Al comparar un URL con esta lista, si se encuentra una coincidencia, el URL será considerado como potencialmente peligroso y se le notificará al usuario. Esta es una medida preventiva crítica para evitar el acceso a sitios que previamente han sido identificados como una amenaza. La fuente utilizada para evaluar los URL es Google Safe Browsing [30], la cual a través de su API permite detectar URL maliciosos.

- Lista de popularidad: incluye URL que son ampliamente visitados por los usuarios en general, pero no están necesariamente incluidos en ninguna de las listas anteriormente mencionadas. Se utiliza para identificar sitios que, aunque no son completamente verificados, tienen un nivel de confianza e importancia basado en la familia de algoritmos PageRank, la cual utiliza como indicador los enlaces externos entre páginas y la fiabilidad que estos aportan para darle un punteo a cada URL registrada, por lo que si un URL se encuentra en esta lista y tiene un punteo alto es muy poco probable que dirija hacia una página maliciosa [31], [32].

Esta capa se hizo disponible a través de una API para poder recibir peticiones y enviar respuestas acerca del análisis de detección perteneciente a esta capa. Fue desarrollada a través de la librería de Python, Flask.

6.4. Desarrollo de capas basadas en modelos de *machine learning*

Las capas de detección basadas en modelos de *machine learning* fueron planteadas para principalmente detectar tres de los vectores de ataque más utilizados para realizar *phishing*: URL, mensajes de texto y contenido web, para cada uno de estos medios se diseñó una capa a la que pertenece un modelo que detecta *phishing* en su vector de ataque respectivo. El objetivo principal de esta decisión de diseño fue poder dividir el análisis por cada una de las capas, permitiéndole a la herramienta decidir si el contenido a analizar podría ser malicioso basándose en el resultado independiente de cada uno de los análisis, la herramienta podrá alertar al usuario de un posible ataque de *phishing* aun cuando no se hayan completado todos los análisis, de esta forma se podrá obtener una respuesta más rápida por parte de la herramienta.

De manera similar a la capa basada en listas, esta se hizo disponible a través de una API desarrollada por medio de Flask.

Para las tres capas se dividió el conjunto de datos en un conjunto de entrenamiento conteniendo aproximadamente el 70 % de los datos del conjunto original y un conjunto de pruebas que consiste del 30 % de los datos originales. Para el proceso de entrenamiento de modelos se utilizó la librería de Python, LazyPredict. La principal razón de su uso es poder realizar varios entrenamientos a través de diferentes algoritmos de aprendizaje para posteriormente poder observar el desempeño de estos y facilitar la selección de los algoritmos para el entrenamiento de los modelos en cada una de las capas de detección. Al momento de que dos o más modelos tuvieran un rendimiento similar en cuanto a métricas de precisión se realizó un análisis del tiempo de entrenamiento, el cual según [33] aunque puede ser una métrica que dependa de varios factores como el tamaño del conjunto de datos, especificaciones de la máquina utilizada para entrenamiento, es un factor el cual es importante debido la oportunidad de escalabilidad que le ofrece al modelo y a la herramienta, esto va de la mano con la mitigación de las diferentes técnicas de *phishing* que tienen una constante evolución con el paso del tiempo. Otra métrica que se tomó en cuenta para la selección de modelos fue la validación cruzada, la cual de acuerdo con [34] es un procedimiento de remuestreo que determina la precisión y desempeño del modelo a través de la evaluación de diferentes

subconjuntos del conjunto de datos inicial. La técnica de validación cruzada utilizada fue k-fold, como se logra observar en la figura 7, esta técnica de validación cruzada divide el conjunto de datos inicial en k partes, cada subconjunto creado a partir de esta división se utilizara como data de prueba para validar los resultados del modelo, mientras que la parte restante del conjunto se utilizara como parte del subconjunto de entrenamiento.

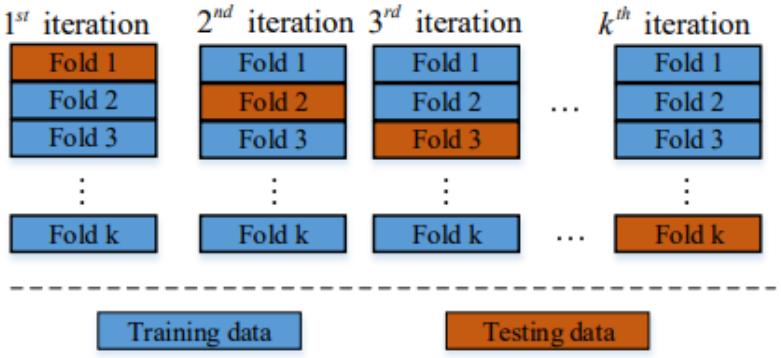

Figura 7: Validación cruzada k-fold de [35]

6.5. Desarrollo de extensión web

El desarrollo de la extensión web se llevó a cabo siguiendo estándares propuestos por la base de código Chromium. La elección de desarrollar la herramienta en esta plataforma se fundamenta en su amplia adopción y compatibilidad con una amplia gama de navegadores populares, haciendo que en un futuro la herramienta pueda ser fácilmente accesible por más usuarios. Durante el desarrollo se utilizaron tecnologías web como HTML, CSS y JavaScript, empleando el framework React como una herramienta para agilizar y facilitar el desarrollo.

Se desarrolló un archivo manifiesto que define las configuraciones, permisos, *script* y acciones que la extensión realiza. Este archivo fue esencial para el desarrollo de la herramienta, ya que permite manejar todo lo relacionado con las extensiones, permitiendo ejecutar *script* en segundo plano para realizar el análisis de *phishing* sobre el contenido seleccionado por el usuario. Adicionalmente, se implementó comunicación a través de las API de las demás capas para poder analizar de manera eficiente el contenido seleccionado.

La figura 8 ilustra el flujo de operaciones de la herramienta, el proceso comienza con la entrada y parseo del contenido, seguido de una verificación si contiene URL . Si hay URL en el contenido, primero se revisan contra la lista blanca, si la URL está presente, el contenido se marca como seguro. Si no está en la lista blanca, se revisa la lista negra, si aparece en esta lista, se clasifica como contenido inseguro. De acuerdo con [1], esta capa inicial basada en listas puede actuar como una especie de filtro y primera linea de defensa ante posibles ataques de *phishing* ya conocidos por medio de su URL, proporcionando un resultado rápido y conciso al usuario. Si la URL no se encuentra en ninguna de las listas, se chequea su popularidad y se evalúa la URL, el mensaje y el contenido web a través de los modelos de *machine learning*. Posteriormente, los resultados de estas evaluaciones se registran y se utilizan para calcular el índice de seguridad, que finalmente se le muestra al usuario.

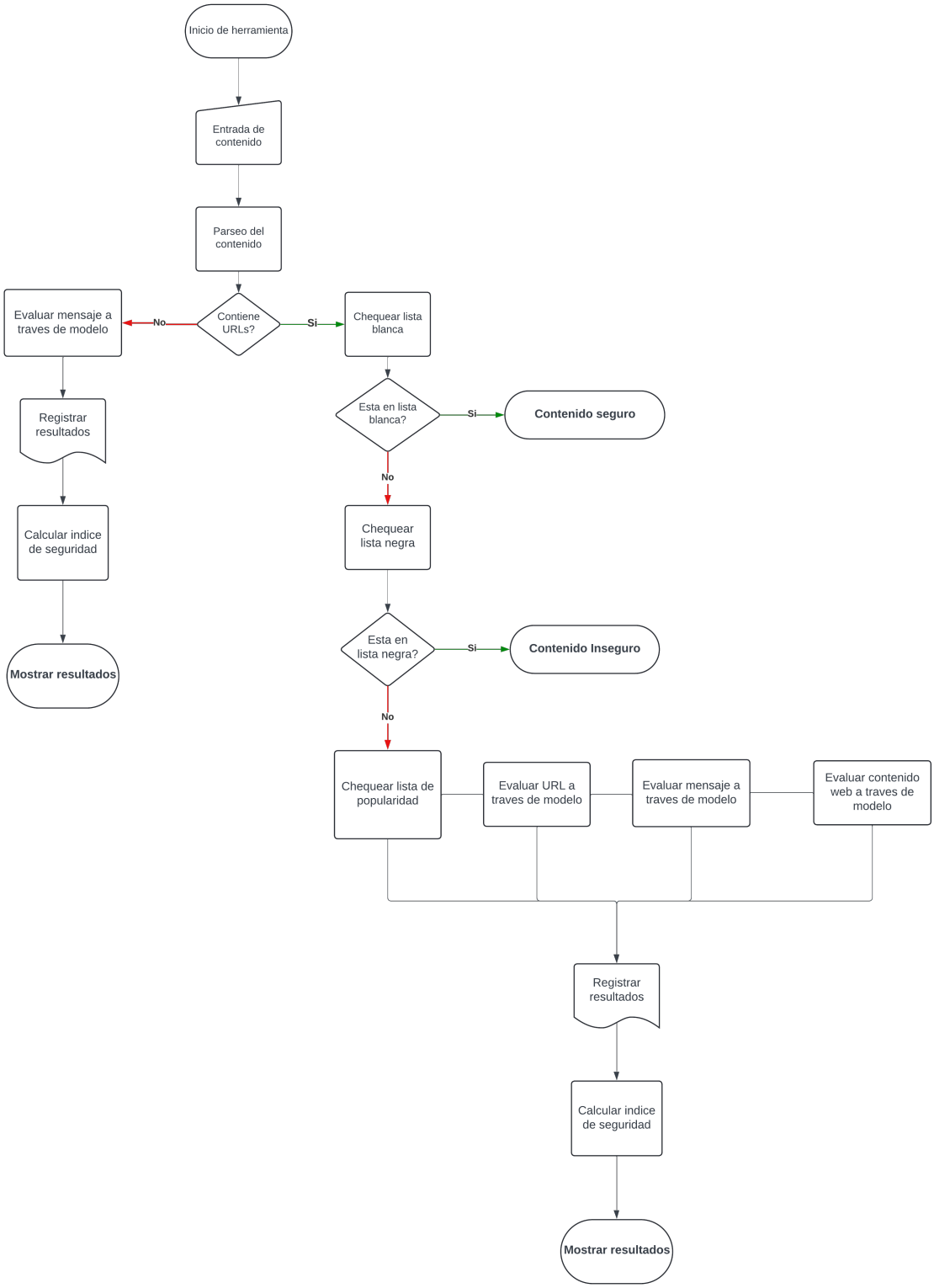

Figura 8: Diagrama de flujo herramienta

En la figura 9 se observa el popup principal de la herramienta, en donde brevemente se

le instruye al usuario sobre el uso de la herramienta, adicionalmente, este permite ingresar a un popup que tiene información adicional acerca del proyecto.

Figura 9: *Popup* principal de extensión

La figura 10 nos muestra el *popup* mencionado anteriormente. En este se le explica al usuario información a grandes rasgos acerca del proyecto y su funcionamiento.

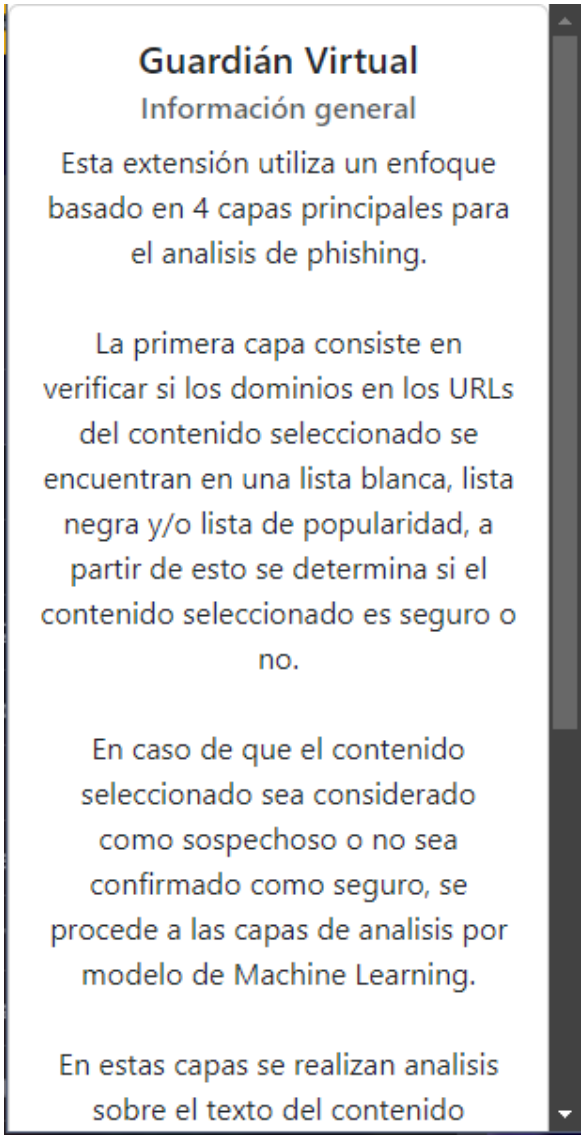

Figura 10: *Popup* información general

Como se logra apreciar en la figura 11, luego de realizar el análisis del contenido con la herramienta se le muestra al usuario la pantalla de resultados, en donde con un círculo de progreso se representa que tan seguro es el contenido de acorde al índice de seguridad calculado con la fórmula 1

$$\text{Índice de seguridad} = \frac{w_{\text{popularity}} \cdot f_{\text{popularity}} + w_{\text{url}} \cdot f_{\text{url}} + w_{\text{msg}} \cdot f_{\text{msg}} + w_{\text{content}} \cdot f_{\text{content}}}{w_{\text{popularity}} + w_{\text{url}} + w_{\text{msg}} + w_{\text{content}}} \times 100 \tag{1}$$

Donde:

$$f_{\text{popularity}} = \begin{cases} 1 & \text{Todos los elementos analizados con } \texttt{lista de popularidad} \\ & \text{tienen un } \texttt{rank > 500000}, \\ 0 & \text{De lo contrario.} \end{cases} \tag{2}$$

$$f_{\text{url}} = \begin{cases} 1 & \text{Todo elemento analizado por } \texttt{detección por URL} \\ & \text{tiene una } \texttt{prediction = 'Safe'}, \\ 0 & \text{De lo contrario.} \end{cases} \tag{3}$$

$$f_{\text{msg}} = \begin{cases} 1 & \text{Todo elemento analizado por } \texttt{detección por mensaje} \\ & \text{tiene una } \texttt{prediction = 'Safe'}, \\ 0 & \text{De lo contrario.} \end{cases} \tag{4}$$

$$f_{\text{content}} = \begin{cases} 1 & \text{Todo elemento analizado por } \texttt{detección por contenido} \\ & \text{tiene una } \texttt{prediction = 'Safe'}, \\ 0 & \text{De lo contrario.} \end{cases} \tag{5}$$

Los pesos $w_{\text{popularity}}, w_{\text{url}}, w_{\text{msg}}, w_{\text{content}}$ están definidos como 20 %, 20 %, 20 %, y 40 % respectivamente:

$$w_{\text{popularity}} = 20, \quad w_{\text{url}} = 20, \quad w_{\text{msg}} = 20, \quad w_{\text{content}} = 40.$$

El valor para los pesos de cada uno de los rubros fue basado en la importancia que pueden tener según lo importante que sea la detección individual de estos, de acuerdo con [11] entre los vectores de *phishing* más comunes se encuentran los URL , correos electrónicos, mensajes en redes sociales y sitios web, sin embargo, de estos vectores los sitios web suelen ser los que más relevancia tienen durante estos ciberataques, principalmente debido a que esta es la herramienta con la que los propios ciberdelincuentes obtienen credenciales o información importante de las víctimas, además, las víctimas suelen ser más vulnerables dentro de un sitio web malicioso que está haciéndose pasar por un sitio legítimo, ya que tienden a confiar en este, pudiendo llegar a otorgar información sensible. Por esta razón se tomó la decisión de otorgarle un mayor peso al contenido web, ya que suele ser el punto clave en donde el atacante puede obtener información privada de la víctima.

Condiciones especiales:

Si todos los elementos analizados por `lista blanca` tienen `status = 'In Whitelist'`, entonces:

$$\text{Índice de seguridad} = 100.$$

Y si al menos un elemento analizado por `lista negra` tiene `status = 'In Blacklist'`, entonces:

$$\text{Índice de seguridad} = 0.$$

Figura 11: *Popup* resultados

Si se desean observar los resultados de manera más detallada se pueden apreciar en el *popup* de resultados avanzados como se observa en la figura 12. En este *popup* se detalla el resultado para cada uno de los rubros que toma en cuenta la herramienta para calcular el índice de seguridad, es posible observar el resultado del análisis para cada dominio o URL que fue extraído del contenido analizado. Esta visualización de resultados le puede permitir al usuario tomar una decisión más informada sobre si debe evitar el contenido analizado o si lo puede considerar seguro.

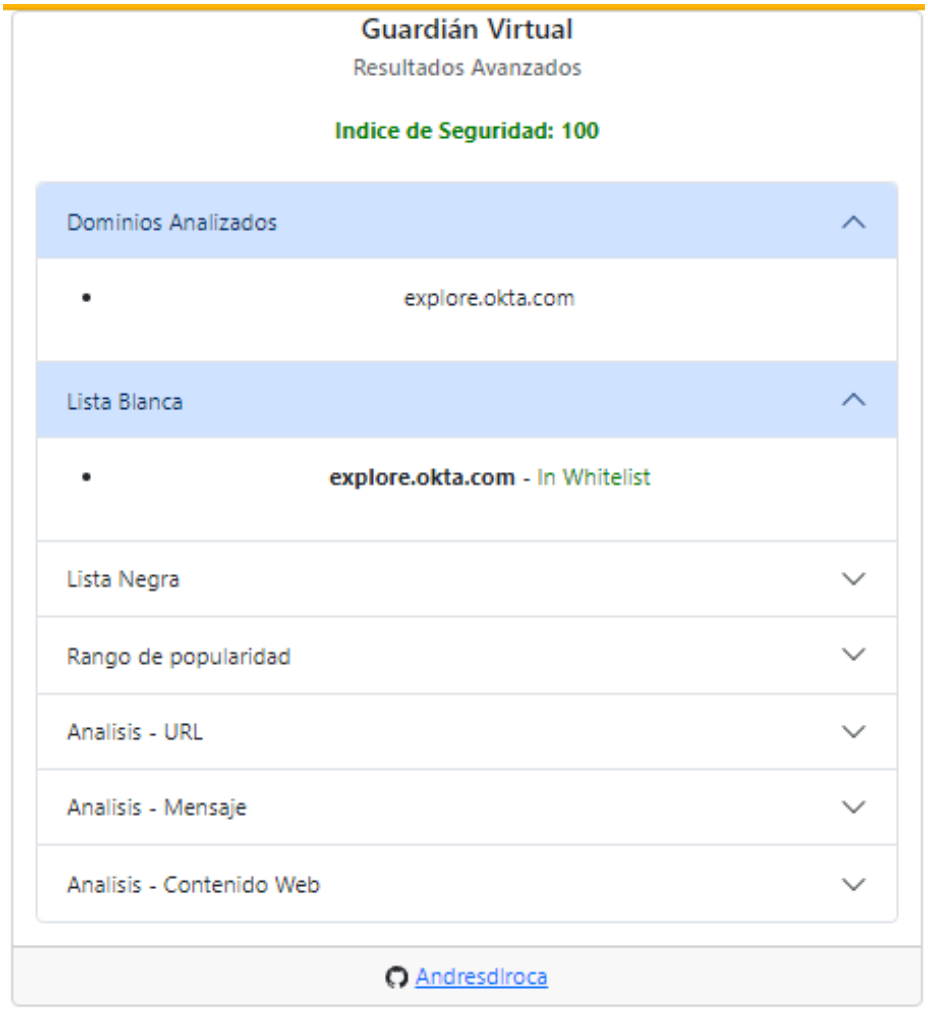

Figura 12: *Popup* resultados avanzados

CAPÍTULO 7

Resultados

Los resultados obtenidos durante el entrenamiento y pruebas de los modelos de *machine learning* nos permiten observar el rendimiento que estos alcanzaron en sus respectivos rubros de clasificación y detección, todos los modelos fueron entrenados con los métodos descritos previamente para de esta manera optimizar al máximo las capacidades de predicción de cada uno de los modelos, poder evaluarlos de manera efectiva y seleccionar los que mejor se desempeñen para utilizar en la herramienta.

7.0.1. Detección por URL

Se realizó un entrenamiento masivo de varios de los modelos de *machine learning* más utilizados, como se puede observar en el cuadro 1, los algoritmos Extra Trees y Random Forest muestran ambos una precisión general del 88 %. En el caso de Random Forest, se puede observar un tiempo de entrenamiento de 72.42 segundos, por otro lado, Extra Trees cuenta con un tiempo de entrenamiento de 47.09 segundos.

Cuadro 1: Métricas de modelos de clasificación - Detección por URL

Modelo	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Tiempo de entrenamiento (s)
ExtraTreesClassifier	0.88	0.88	0.88	0.88	47.09
RandomForestClassifier	0.88	0.88	0.88	0.88	72.42
BaggingClassifier	0.87	0.87	0.87	0.87	35.52
DecisionTreeClassifier	0.83	0.83	0.83	0.83	4.45
XGBClassifier	0.82	0.82	0.82	0.82	12.96
ExtraTreeClassifier	0.82	0.82	0.82	0.82	0.76
KNeighborsClassifier	0.81	0.81	0.81	0.81	30.33
LGBMClassifier	0.80	0.80	0.80	0.80	2.15
NuSVC	0.79	0.79	0.79	0.79	7700.24
SVC	0.78	0.78	0.78	0.78	3774.78
AdaBoostClassifier	0.74	0.74	0.74	0.74	22.00
CalibratedClassifierCV	0.67	0.67	0.67	0.67	387.97
LinearSVC	0.67	0.67	0.67	0.67	71.94
LogisticRegression	0.67	0.67	0.67	0.67	0.76
SGDClassifier	0.67	0.67	0.67	0.66	1.04
LinearDiscriminantAnalysis	0.66	0.66	0.66	0.66	0.67
RidgeClassifier	0.66	0.66	0.66	0.66	0.51
RidgeClassifierCV	0.66	0.66	0.66	0.66	0.86
QuadraticDiscriminantAnalysis	0.64	0.64	0.64	0.59	0.65
GaussianNB	0.63	0.63	0.63	0.59	0.48
Perceptron	0.63	0.63	0.63	0.63	1.34
BernoulliNB	0.62	0.62	0.62	0.62	0.61
NearestCentroid	0.62	0.62	0.62	0.61	0.36
PassiveAggressiveClassifier	0.55	0.55	0.55	0.53	0.85
DummyClassifier	0.50	0.50	0.50	0.33	0.36

Los resultados de la matriz de confusión perteneciente al modelo Random Forest se pueden observar en la figura 13, se puede apreciar que el modelo cuenta con una tasa de aciertos del 44.55 % para casos en los que no hay *phishing* en el URL (verdaderos negativos) y un 43.60 % para casos en los que sí hay *phishing* en el URL, existen algunos errores de clasificación, un 5.51 % de URL legítimas son clasificadas incorrectamente como *phishing* (falsos positivos).

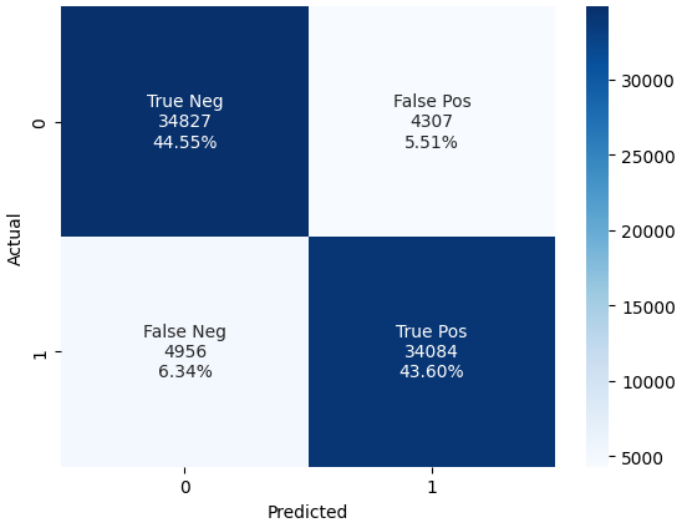

Figura 13: Matriz de confusión - Random Forest

Al observar la figura 14, perteneciente al modelo Extra Trees, se puede observar que tiene

una tasa de URL legítimos identificados correctamente (verdaderos negativos) del 44.48 %, y una tasa de URL de *phishing* identificados correctamente (verdaderos positivos) de 43.68 %. Los falsos positivos presentan una tasa de error de 5.66 %, el cual es el error menos crítico, por otro lado, el error más crítico, los falsos negativos, tienen una tasa de 6.18 %.

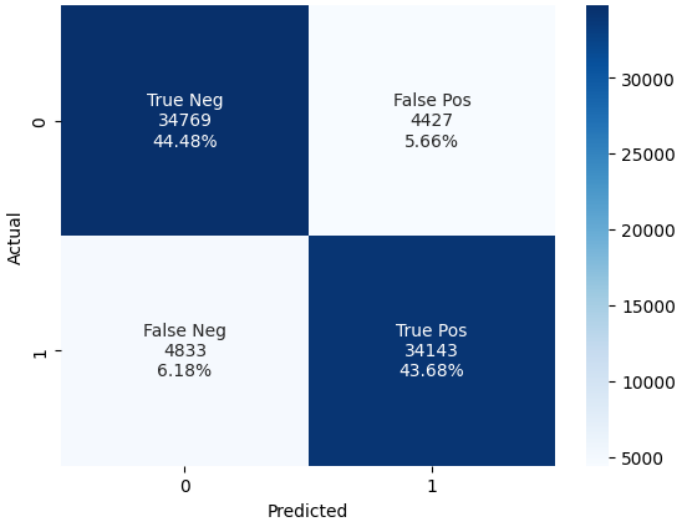

Figura 14: Matriz de confusión - Extra Trees

En las figuras 15 y 16 se puede observar que ambos modelos cuentan con un área bajo la curva de 0.96. Un AUC de 0.96 está muy cercano a 1, métricas que muestra la capacidad de los modelos de discernir entre los casos pertenecientes a verdadero positivo y verdadero negativo.

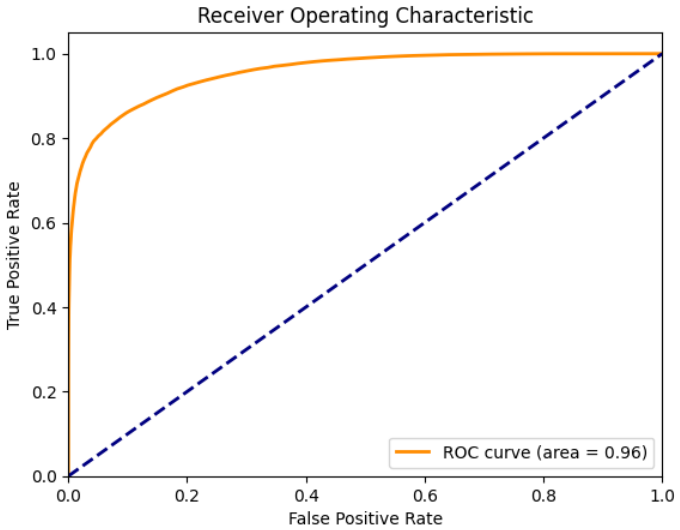

Figura 15: Curva ROC - Random Forest

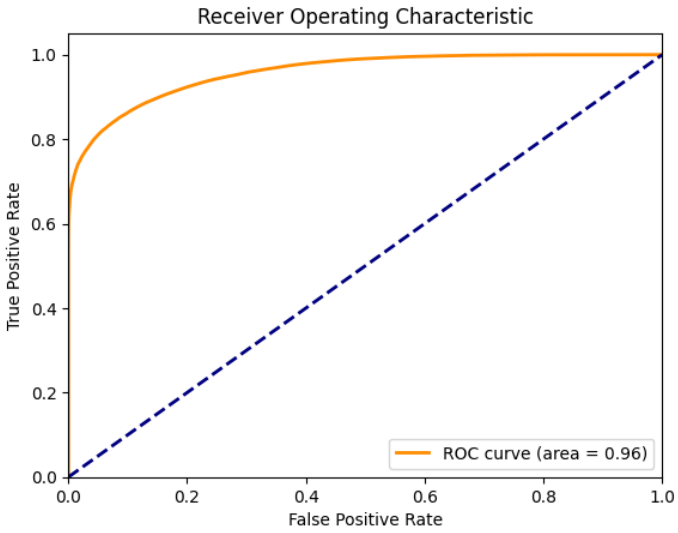

Figura 16: Curva ROC - Extra Trees

Los resultados de validación cruzada por K-Fold como se aprecia en el cuadro 2 muestran que el modelo Extra Trees obtuvo una precisión del 89.31 %, por su parte el modelo Random Forest obtuvo una precisión del 89.07 %.

Cuadro 2: K-Fold accuracy de modelos - URL

Modelo	**K-Fold accuracy**
Extra Trees	0.8931
Random Forest	0.8907

7.0.2. Detección por mensaje de texto

Los resultados de las métricas de los modelos de clasificación como se observan en el cuadro 3 muestran que el clasificador Extra Trees obtuvo una precisión general del 98 %, aunque requiere un tiempo de entrenamiento de 429.01 segundos. Otros modelos como CalibratedClassifierCV y Regresión Logística, tienen un rendimiento general del 97 %. El CalibratedClassifierCV toma un tiempo de 666.13 segundos en entrenar, por otro lado, el modelo de Regresión Logística toma 115.51 segundos. Los modelos LinearSVC y NuSVC cuentan con un 96 % de precisión en general, NuSVC cuenta con un tiempo de entrenamiento de 14959.14 segundos, mientras que, LinearSVC, 203.44 segundos.

Cuadro 3: Métricas de modelos de clasificación - Detección por mensajes de texto

Modelo	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Tiempo de entrenamiento (s)
ExtraTreesClassifier	0.98	0.98	0.98	0.98	429.01
CalibratedClassifierCV	0.97	0.97	0.97	0.97	666.13
LogisticRegression	0.97	0.97	0.97	0.97	115.51
LinearSVC	0.96	0.96	0.96	0.96	203.44
NuSVC	0.96	0.96	0.96	0.96	14959.14
BaggingClassifier	0.96	0.96	0.96	0.96	2184.03
DecisionTreeClassifier	0.94	0.94	0.94	0.94	389.95
ExtraTreeClassifier	0.91	0.91	0.91	0.91	90.25
AdaBoostClassifier	0.90	0.90	0.90	0.90	338.17
GaussianNB	0.84	0.85	0.85	0.84	57.30
KNeighborsClassifier	0.78	0.77	0.77	0.77	171.01
BernoulliNB	0.75	0.76	0.76	0.74	64.14
NearestCentroid	0.69	0.70	0.70	0.66	53.67
DummyClassifier	0.52	0.50	0.50	0.36	38.23

Como se aprecia en la figura 17, la matriz de confusión del modelo Extra, el modelo tiene una tasa del 46.56 % de verdaderos negativos. La clasificación correcta de mensajes legítimos. De la misma forma, tiene una tasa de verdaderos positivos del 51.03 %. Por otro lado, los errores de clasificación para ambos falsos positivos y falsos negativos obtuvieron tasas del 0.69 % y del 1.72 % respectivamente.

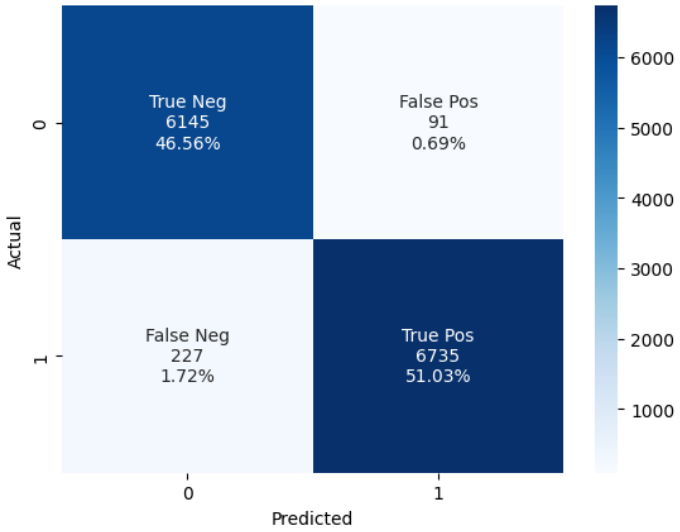

Figura 17: Matrix de confusión - Extra Trees

La curva ROC muestra un área bajo la curva de 1.00 para el modelo Extra Trees como se observa en la figura 18, este es un indicador de su capacidad para realizar distinciones entre mensajes de *phishing* y mensajes legítimos.

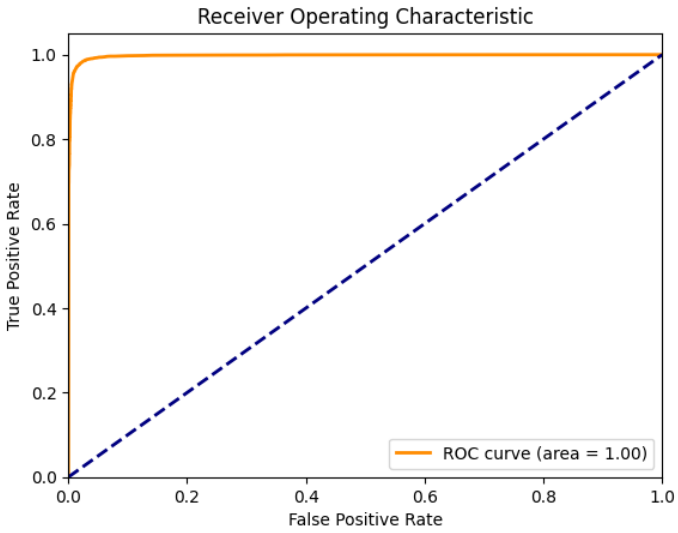

Figura 18: Curva ROC - Extra Trees

Al explorar las métricas obtenidas por medio de validación cruzada sobre el modelo Extra Trees, como se observa en el cuadro 4 se obtuvo una precisión del 97.76 %.

Cuadro 4: K-Fold accuracy de modelos - Mensajes

Modelo	**K-Fold accuracy**
Extra Trees	0.9776

7.0.3. Detección por contenido web

Los resultados obtenidos durante el entrenamiento de varios modelos en el cuadro 5 muestran que hubo cuatro modelos con un rendimiento general similar, con una tasa de precisión general del 98 %. La principal diferencia entre los modelos radica en el tiempo de entrenamiento, sin embargo, en los cuatro modelos este no supera ni siquiera 1 segundo.

Cuadro 5: Métricas de modelos de clasificación - Detección por contenido web

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Tiempo de entrenamiento (s)
XGBClassifier	0.98	0.98	0.98	0.98	0.93
RandomForestClassifier	0.98	0.98	0.98	0.98	0.87
ExtraTreesClassifier	0.98	0.98	0.98	0.98	0.77
LGBMClassifier	0.98	0.98	0.98	0.98	0.24
BaggingClassifier	0.97	0.97	0.97	0.97	0.49
DecisionTreeClassifier	0.97	0.97	0.97	0.97	0.05
KNeighborsClassifier	0.96	0.96	0.96	0.96	0.37
ExtraTreeClassifier	0.96	0.96	0.96	0.96	0.03
AdaBoostClassifier	0.95	0.95	0.95	0.95	0.49
LabelPropagation	0.94	0.94	0.94	0.94	5.60
LabelSpreading	0.94	0.94	0.94	0.94	6.24
SVC	0.93	0.93	0.93	0.93	2.05
RidgeClassifier	0.89	0.89	0.89	0.89	0.07
LinearDiscriminantAnalysis	0.89	0.89	0.89	0.89	0.05
RidgeClassifierCV	0.89	0.89	0.89	0.89	0.14
SGDClassifier	0.89	0.89	0.89	0.89	0.09
LogisticRegression	0.89	0.89	0.89	0.89	0.08
CalibratedClassifierCV	0.89	0.89	0.89	0.89	2.46
QuadraticDiscriminantAnalysis	0.89	0.89	0.89	0.89	0.04
LinearSVC	0.89	0.89	0.89	0.89	0.83
NuSVC	0.87	0.88	0.88	0.87	4.65
GaussianNB	0.87	0.88	0.88	0.87	0.02
BernoulliNB	0.86	0.86	0.86	0.86	0.19
PassiveAggressiveClassifier	0.82	0.82	0.82	0.82	0.04
Perceptron	0.81	0.81	0.81	0.81	0.03
NearestCentroid	0.81	0.80	0.80	0.80	0.02
DummyClassifier	0.49	0.50	0.50	0.32	0.02

El modelo Random Forest, como se aprecia en la figura 19, tiene una tasa de verdaderos positivos del 50.28 %, para los verdaderos negativos cuenta con un 47.80 % de tasa de detección. En cuanto a los falsos positivos, estos cuentan con una tasa de 1.00 %. Su tasa de falsos positivos es de 1.12 %,

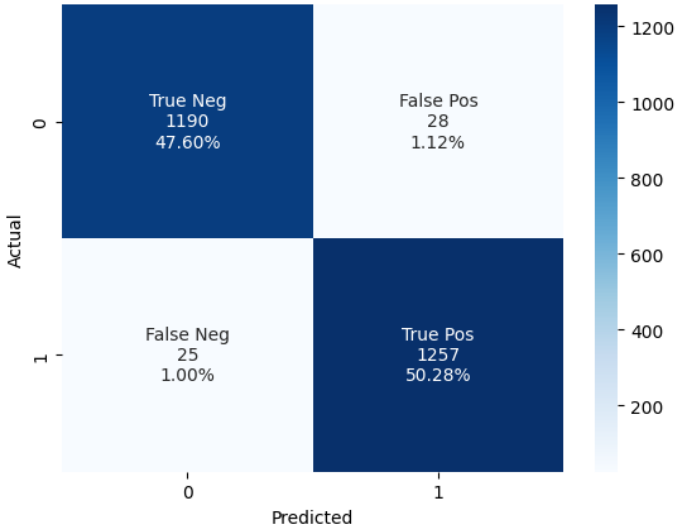

Figura 19: Matriz de confusión - Random Forest

El modelo LGBM, como se observa en la figura 20 cuenta con una tasa de verdaderos negativos de 47.72 % y verdaderos positivos de 50.16 %. En cuanto a los errores de clasificación,

la tasa de falsos negativos es de 1.12 % y la de falsos positivos es de 1.00 %.

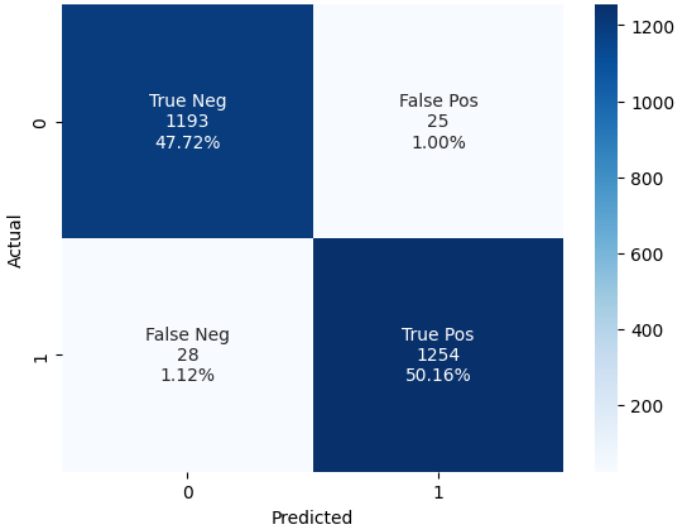

Figura 20: Matriz de confusión - LGBM

XGBoost, como está ilustrado en la figura 21, tiene una tasa de verdaderos negativos del 47.76 %, mientras que los verdaderos positivos tienen una tasa del 50.04 %, por su parte los errores de clasificación, 0.96 % es la tasa correspondiente a los falsos positivos y 1.24 % es correspondiente a los falsos negativos.

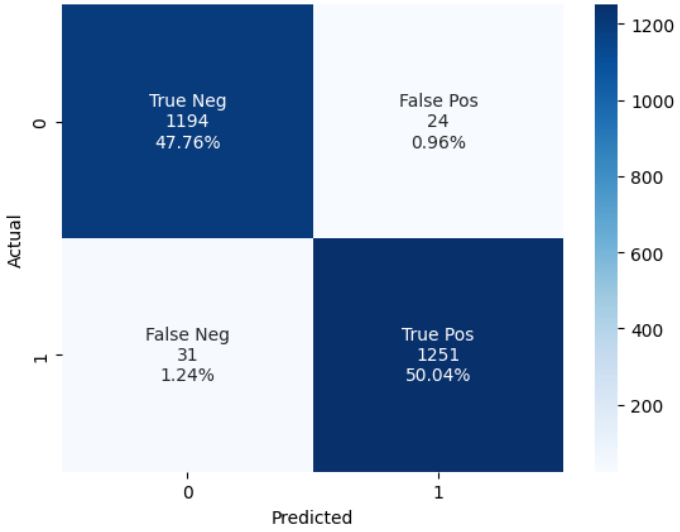

Figura 21: Matriz de confusión - XGB

El modelo Extra Trees, como se plasma en la figura 22, tiene una tasa de verdaderos positivos del 50.08 %, cuenta con una tasa de falsos negativos del 1.20 % y una tasa de falsos positivos de 0.96 %.

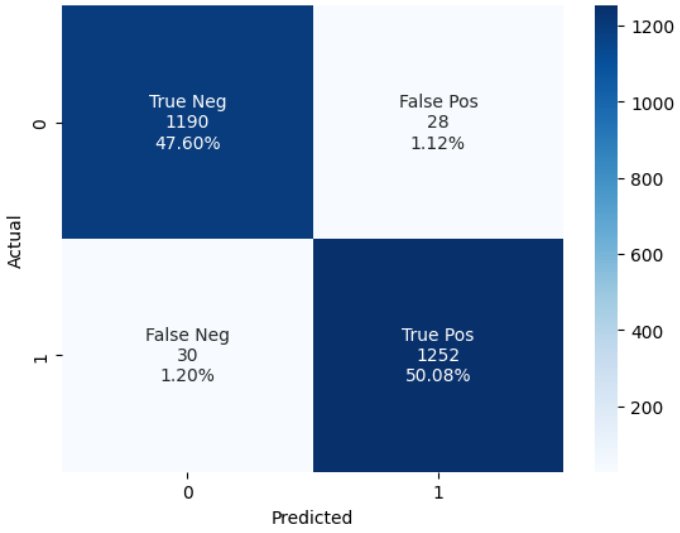

Figura 22: Matriz de confusión - Extra Trees

Tras analizar las matrices de confusión de manera detenida, se puede observar que los dos modelos que se podrían adaptar mejor a la herramienta son LGBM y Random Forest, esto se debe a su tasa baja en falsos negativos, ya que el principal motivo de la herramienta es evitar poner en riesgo al usuario, por lo que tener una tasa baja de este error crítico puede ayudar a cumplir este propósito.

Tras observar detalladamente las curvas ROC de los modelos Random Forest y LGBM, en las figuras 23 y 24, se puede observar un desempeño idéntico de parte de ambos para la discriminación entre contenido web *phishing* y contenido web legítimo, contando con un valor de 1.00.

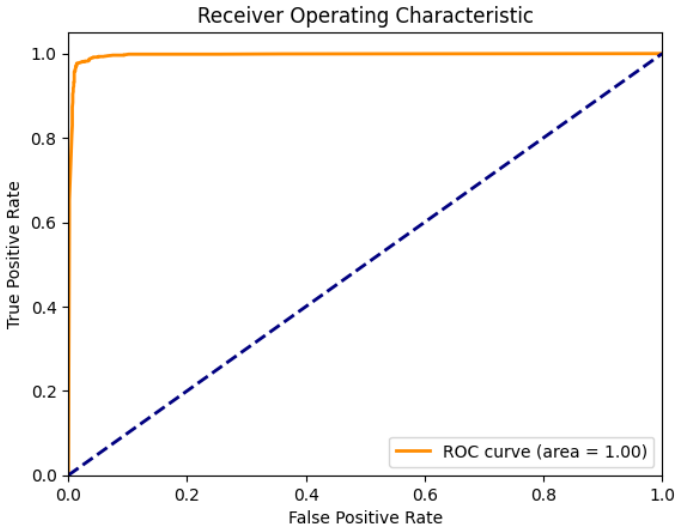

Figura 23: Curva ROC - Random Forest

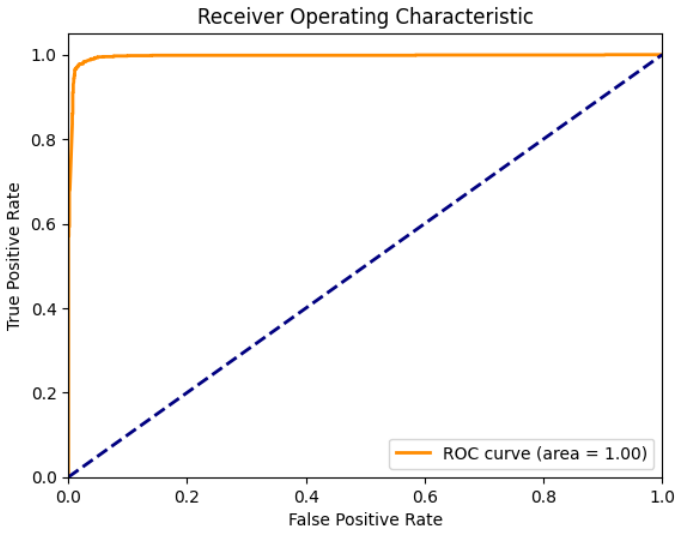

Figura 24: Curva ROC - LGBM

Al observar los resultados de validación cruzada observados en el cuadro 6, se observa una precisión del 97.39 % por parte del modelo de Random Forest, LightGBM por su parte cuenta con una precisión del 97.31 %.

Cuadro 6: K-Fold accuracy de modelos

Modelo	**K-Fold accuracy**
Random Forest	0.9739
LightGBM	0.9731

CAPÍTULO 8

Discusión de resultados

A partir de los resultados obtenidos a partir de las diferentes pruebas realizadas, se eligieron diferentes modelos para cada uno de los tipos de detección (URL, mensaje de texto y contenido web). A continuación, se discute sobre la elección de cada uno de los modelos que se tomaron en cuenta para utilizarse en la herramienta.

8.1. Detección por URL

Los resultados obtenidos durante el entrenamiento y la evaluación del los modelos de detección de *phishing* por URL muestran un alto rendimiento en general, con diferencias sutiles entre los algoritmos evaluados, sin embargo, los dos modelos con mejor desempeño fueron Extra Trees y Random Forest, ambos clasificadores de ensamblaje basados en árboles de decisión.

Al comparar los tiempos de entrenamiento, se observa que Extra Trees ofrece un entrenamiento más rápido (47.09 segundos) en comparación con Random Forest (72.42 segundos), factor el cual es relevante dentro de situaciones que demanden constante actualización de los modelos y rapidez. Sin embargo, este factor no fue el único determinante en la selección del modelo, ya que se consideraron métricas adicionales para garantizar un balance entre rapidez y efectividad.

Los resultados indican un desempeño casi idéntico en términos de precisión general entre ambos modelos, con áreas bajo la curva de 0.96, lo que evidencia un alto poder discriminativo para identificar correctamente URL de *phishing* y legítimas. Este valor cercano a 1 indica que ambos modelos son robustos en términos de detección, con la capacidad de diferenciar de manera efectiva entre las clases. Sin embargo, al examinar más detalladamente las matrices de confusión, se pueden identificar diferencias importantes en la clasificación de

falsos positivos y falsos negativos.

El modelo de Random Forest presento una tasa ligeramente inferior de falsos positivos (5.51 %), lo cual es ventajoso en la minimización de falsas alarmas, sin embargo, también tuvo una tasa más alta de falsos negativos (6.34 %), error el cual es más crítico para la seguridad del usuario. Los falsos negativos representan un riesgo considerable, ya que URL maliciosas pueden ser clasificadas incorrectamente como legítimas, exponiendo al usuario a potenciales ataques de *phishing*. Por otro lado, Extra Trees, aunque tiene una tasa marginalmente superior de falsos positivos (5.66 %), mejora ligeramente en la detección de falsos negativos (6.18 %), lo que lo hace más eficaz en la identificación de URL de *phishing*.

Adicionalmente, la validación cruzada con K-Fold reafirma la robustez de ambos modelos, mostrando que mantienen un rendimiento consistente a lo largo de las particiones del conjunto de datos. Extra Trees obtuvo una precisión del 89.31 %, ligeramente superior a la de Random Forest con un 89.07 %, lo que refuerza su capacidad de generalización y su efectividad en la detección de *phishing*.

Aunque ambos modelos son altamente eficaces, los resultados indican que Extra Trees logra ser marginalmente mejor en términos de detección de *phishing* y reducción de falsos negativos, lo que lo convierte en una opción más adecuada para su implementación en la herramienta de detección de *phishing*. La combinación de un tiempo de entrenamiento más rápido, una mayor precisión en la detección de URL maliciosas y una tasa más baja de errores críticos justifica la elección de este modelo como el más adecuado para la herramienta.

8.2. Detección por mensaje de texto

Evaluando los resultados obtenidos de los modelos de clasificación evaluados para la detección por mensaje de texto, hubo varios con un rendimiento alto, se puede destacar el clasificador Extra Trees como el mejor modelo en términos de precisión y efectividad general. Con una precisión de 98 %, este modelo demostró ser altamente robusto, lo que lo posiciona como una opción altamente confiable para la detección de mensajes de *phishing*. Si bien su tiempo de entrenamiento fue de 429.01 segundos, este tiempo es aceptable considerando su alto rendimiento.

Otros modelos como CalibratedClassifierCV y la Regresión Logística presentaron una precisión de 97 %. Sin embargo, el modelo CalibratedClassifierCV requirió un tiempo significativamente mayor (666.13 segundos), mientras que regresión logística se destacó por su rapidez con un tiempo de 115.51 segundos, aunque con un rendimiento ligeramente inferior. Por otro lado, los modelos LinearSVC y NuSVC tuvieron un rendimiento de 96 % de precisión, pero NuSVC demostró ser ineficiente debido a su extenso tiempo de entrenamiento de 14,959.14 segundos, lo que lo descarta como una opción factible.

El rendimiento superior del modelo Extra Trees fue validado a través de la matriz de confusión, donde mostró tasas de verdaderos negativos del 46.56 %, lo que indica que clasifica correctamente la mayoría de los mensajes legítimos. Asimismo, su tasa de verdaderos positivos del 51.03 % demuestra su capacidad para identificar eficazmente los mensajes de *phishing*. Los errores de clasificación fueron mínimos, con un 0.69 % de falsos positivos y un

1.72 % de falsos negativos. Este último tipo de error es crítico para la herramienta, ya que los mensajes de *phishing* mal clasificados representan un riesgo para los usuarios, aunque el modelo ha mostrado un excelente control sobre estos casos.

La curva ROC del modelo Extra Trees mostró un área bajo la curva (AUC) de 1.00, lo que confirma un rendimiento excepcional al diferenciar entre mensajes legítimos y mensajes de *phishing*. Este valor tan alto indica un equilibrio ideal entre sensibilidad y especificidad, lo que refuerza su capacidad para clasificar correctamente ambos mensajes.

Por último, la validación cruzada con K-Fold confirmo la robustez del modelo, mostrando una precisión del 97.76 %. Esto refleja que el modelo es consistente y generaliza bien sobre diferentes subconjuntos de datos, lo que indica que puede mantener un rendimiento estable. La combinación de estos resultados sugieren que Extra Trees es un modelo confiable y altamente eficaz, lo que justifica su integración en la herramienta de detección.

8.3. Detección por contenido web

En la detección de *phishing* mediante contenido web, los resultados de los modelos de clasificación evaluados mostraron un rendimiento general sumamente alto, con cuatro modelos alcanzando una tasa de precisión del 98 %. Las diferencias entre estos modelos radican principalmente en los tiempos de entrenamiento, aunque estas diferencias son marginales. Sin embargo, es crucial evaluar otros aspectos como las tasas de error, especialmente los errores críticos, como falsos negativos, ya que como se mencionó anteriormente, estos representan el mayor riesgo para el usuario.

El modelo Random Forest destacó por tener la mayor tasa de detección de contenido *phishing* con un 50.28 % de verdaderos positivos y la tasa más baja de falsos negativos, un 1.0 %. Esto indica que comete pocos errores al clasificar sitios de *phishing*, lo cual es crucial para proteger a los usuarios de la herramienta. No obstante, su tasa de falsos positivos fue ligeramente más alta en comparación con otros modelos, con un 1.12 %, lo que significa que podría clasificar algunos sitios legítimos como *phishing*, aunque este error es menos crítico en comparación con los falsos negativos.

Por su parte, el modelo LGBM mostró un rendimiento similar con una tasa de verdaderos positivos del 50.16 % y verdaderos negativos del 47.72 %. Sin embargo, su tasa de falsos negativos fue marginalmente más alta que la de Random Forest, alcanzando un 1.12 %. Aunque esta diferencia es pequeña, sigue siendo significativa en el contexto de seguridad, ya que implica un leve incremento en la clasificación incorrecta de sitios de *phishing* como legítimos. A su favor, el modelo LGBM presentó una tasa de falsos positivos más baja (1.00 %).

El modelo XGBoost tuvo la mejor tasa de falsos positivos con un 0.96 %, lo que lo hace menos propenso a clasificar sitios legítimos como *phishing*. Sin embargo, su tasa de falsos negativos fue la más alta entre los modelos evaluados, con un 1.24 %. Esto representa un riesgo mayor de no detectar sitios de *phishing*, lo que podría poner en peligro a los usuarios.

Finalmente, el modelo Extra Trees presentó una tasa de verdaderos positivos del 50.08 % y falsos negativos de 1.20 %, situándose entre LGBM y XGBoost en términos de rendimiento,

con una ligera ventaja en la tasa de verdaderos positivos frente a XGBoost.

Los resultados indican que Random Forest y LGBM podrían ser los modelos más adecuados hasta este punto, al analizar sus respectivas curvas ROC se puede identificar que ambos tienen un rendimiento similar para identificar sitios de *phishing* y sitios legítimos.

La validación cruzada muestra resultados muy cercanos para ambos modelos, con una precisión del 97.39 % para Random Forest y 97.31 % para LGBM. Aunque la diferencia es marginal, Random Forest mantiene una ligera ventaja. Al considerar también las tasas de falsos negativos, los resultados sugieren que Random Forest es el modelo más robusto y confiable para la implementación en la herramienta de detección de *phishing*.

CAPÍTULO 9

Conclusiones

- El sistema propuesto logra consolidar la detección basada en listas como una primera linea de defensa para la identificación de dominios, mitigando ataques ya conocidos y la detección por medio de modelos de *machine learning* como una segunda linea de defensa, con cobertura sobre *phishing* basado en mensajes, URL y contenido web para aumentar la robustez del sistema.
- La implementación de la herramienta como una extensión web para navegadores permitirá al usuario poder analizar contenido basándose en el sistema propuesto.
- La combinación de enfoques sobre la detección de ataques basados en mensajes, URL y contenido web en las capas de *machine learning* permite tener una cobertura más amplia sobre los diferentes vectores de ataque.
- Se logró definir un sistema que permite integrar la detección basada en listas y detección basada en *machine learning* como estrategias principales para la detección de phishing.
- Los modelos Extra Trees demostraron ser ideales para la detección de *phishing* por medio de URL y mensajes, mientras que, el modelo Random Forest demostró ser adecuado para la detección de *phishing* por medio de contenido web.
- La utilización de varios modelos durante el análisis por medio de *machine learning* le permitirá al usuario tener diferentes perspectivas dependiendo del resultado del análisis sobre el mensaje, URL y/o contenido web evaluado.

CAPÍTULO 10

Recomendaciones

1. El presente sistema se limita a la cobertura sobre mensajes, URL y contenido web. Se sugiere la posibilidad de aumentar la cobertura del sistema sobre otros vectores de ataque para hacerlo más robusto ante diferentes ataques.

2. Se deben realizar pruebas de usabilidad con usuarios para validar la interfaz gráfica y la experiencia de uso de la herramienta. Esto puede ser clave en próximas iteraciones del proyecto con el objetivo de identificar posibles puntos de mejora en la experiencia de usuario, velocidad de respuesta y efectividad de la herramienta en diferentes entornos.

3. Para futuras iteraciones, los modelos de *machine learning* propuestos en este trabajo deben ser reentrenados con datos los más actualizados posibles para que estos puedan adaptarse de manera adecuada a las futuras estrategias de phishing.

4. Durante el desarrollo de este trabajo se limitó la compatibilidad a la plataforma de Chromium, sin embargo, con el objetivo de ofrecer mayor accesibilidad en futuras iteraciones se podría ampliar su compatibilidad con otros navegadores e incluso adaptarse para su uso en sistemas operativos.

5. La falta de conjuntos de datos relacionados con el *phishing* en español limitó la posibilidad de realizar una localización completa de la herramienta, por lo cual se redujo su enfoque únicamente a ataques de *phishing* por mensaje en inglés. En una próxima iteración, se podría enfocar en dar un soporte en español e incluso multilingüe para incrementar el alcance de la herramienta a una mayor base de usuarios.

6. La creación de conjuntos de datos en español u otros lenguajes podría ser de gran utilidad en futuras iteraciones con el objetivo de aumentar su accesibilidad y cobertura sobre ataques en otros lenguajes.

CAPÍTULO 11

Bibliografía

[1] S. Maurya, H. S. Saini y A. Jain, "Browser extension based hybrid anti-phishing framework using feature selection," *International Journal of Advanced Computer Science and Applications*, vol. 10, n.º 11, 2019.

[2] A.-P. W. Group, *Phishing Activity Trends Report | 4th Quarter 2022*, https://docs.apwg.org/reports/apwg_trends_report_q4_2022.pdf, 2022.

[3] A. Y. Fu, "Web identity security: advanced phishing attacks and counter measures," *unpublished doctoral dissertation, City University of Hong Kong, Hong Kong*, 2006.

[4] P. Boyle y L. A. Shepherd, "MailTrout: A Machine Learning Browser Extension for Detecting Phishing Emails," en *Electronic Workshops in Computing*, BCS Learning & Development, jul. de 2021. DOI: 10.14236/ewic/hci2021.10. dirección: http://dx.doi.org/10.14236/ewic/HCI2021.10.

[5] A. Safi y S. Singh, "A systematic literature review on phishing website detection techniques," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, n.º 2, págs. 590-611, feb. de 2023, ISSN: 1319-1578. DOI: 10.1016/j.jksuci.2023.01.004. dirección: http://dx.doi.org/10.1016/j.jksuci.2023.01.004.

[6] N. A. Azeez, S. Misra, I. A. Margaret, L. Fernandez-Sanz et al., "Adopting automated whitelist approach for detecting phishing attacks," *Computers & Security*, vol. 108, pág. 102 328, 2021.

[7] W. Khan, A. Ahmad, A. Qamar, M. Kamran y M. Altaf, "SpoofCatch: A Client-Side Protection Tool Against Phishing Attacks," *IT Professional*, vol. 23, n.º 2, págs. 65-74, 2021. DOI: 10.1109/MITP.2020.3006477.

[8] Verizon, *2024 Data Breach Investigations Report | Verizon*, https://www.verizon.com/business/resources/reports/dbir/, Accessed: 2024-07-10, 2024.

[9] J. J. Praba, *Cyber Security and Threats*, 2016.

[10] O. K. Sahingoz, E. Buber, O. Demir y B. Diri, "Machine learning based phishing detection from URLs," *Expert Systems with Applications*, vol. 117, págs. 345-357, mar. de 2019, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2018.09.029. dirección: http://dx.doi.org/10.1016/j.eswa.2018.09.029.

[11] R. Alabdan, "Phishing Attacks Survey: Types, Vectors, and Technical Approaches," *Future Internet*, vol. 12, n.º 10, pág. 168, sep. de 2020, ISSN: 1999-5903. DOI: 10.3390/fi12100168. dirección: http://dx.doi.org/10.3390/fi12100168.

[12] Z. Alkhalil, C. Hewage, L. Nawaf e I. Khan, "Phishing Attacks: A Recent Comprehensive Study and a New Anatomy," *Frontiers in Computer Science*, vol. 3, mar. de 2021, ISSN: 2624-9898. DOI: 10.3389/fcomp.2021.563060. dirección: http://dx.doi.org/10.3389/fcomp.2021.563060.

[13] H. Shahbaznezhad, F. Kolini y M. Rashidirad, "Employees' Behavior in Phishing Attacks: What Individual, Organizational, and Technological Factors Matter?" *Journal of Computer Information Systems*, vol. 61, n.º 6, págs. 539-550, oct. de 2020, ISSN: 2380-2057. DOI: 10.1080/08874417.2020.1812134. dirección: http://dx.doi.org/10.1080/08874417.2020.1812134.

[14] A. K. Jain y B. Gupta, "A survey of phishing attack techniques, defence mechanisms and open research challenges," *Enterprise Information Systems*, vol. 16, n.º 4, págs. 527-565, mar. de 2021, ISSN: 1751-7583. DOI: 10.1080/17517575.2021.1896786. dirección: http://dx.doi.org/10.1080/17517575.2021.1896786.

[15] B. Naqvi, K. Perova, A. Farooq, I. Makhdoom, S. Oyedeji y J. Porras, "Mitigation strategies against the phishing attacks: A systematic literature review," *Computers &; Security*, vol. 132, pág. 103 387, sep. de 2023, ISSN: 0167-4048. DOI: 10.1016/j.cose.2023.103387. dirección: http://dx.doi.org/10.1016/j.cose.2023.103387.

[16] B. Mahesh, "Machine learning algorithms-a review," *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, n.º 1, págs. 381-386, 2020.

[17] M. B. Caffaro, "Implementación de inteligencia artificial explicable en un modelo de detección de Leishmaniasis cutánea," Bachelor's Thesis, Universidad del Valle de Guatemala, 2022.

[18] V. Shahrivari, M. M. Darabi y M. Izadi, "Phishing Detection Using Machine Learning Techniques," 2020. arXiv: 2009.11116 [astro-ph.IM]. dirección: http://arxiv.org/pdf/2009.11116.

[19] A. Odeh, I. Keshta y E. Abdelfattah, "Machine LearningTechniquesfor Detection of Website Phishing: A Review for Promises and Challenges," en *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, 2021, págs. 0813-0818. DOI: 10.1109/CCWC51732.2021.9375997.

[20] J. Rashid, T. Mahmood, M. W. Nisar y T. Nazir, "Phishing Detection Using Machine Learning Technique," en *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, 2020, págs. 43-46. DOI: 10.1109/SMARTTECH49988.2020.00026.

[21] A. Sjösten, S. Van Acker y A. Sabelfeld, "Discovering Browser Extensions via Web Accessible Resources," en *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, ép. CODASPY '17, ACM, mar. de 2017. DOI: 10.1145/3029806.3029820. dirección: http://dx.doi.org/10.1145/3029806.3029820.

[22] S. Hawa Apandi, J. Sallim y R. Mohd Sidek, "Types of anti-phishing solutions for phishing attack," *IOP Conference Series: Materials Science and Engineering*, vol. 769, n.º 1, pág. 012 072, feb. de 2020, ISSN: 1757-899X. DOI: 10.1088/1757-899x/769/1/012072. dirección: http://dx.doi.org/10.1088/1757-899X/769/1/012072.

[23] Y. Lin, R. Liu, D. M. Divakaran et al., "Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages," en *30th USENIX Security Symposium (USENIX Security 21)*, 2021, págs. 3793-3810.

[24] T. Tiwari, *Phishing Site Predict Dataset*, https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls/data.

[25] A. K. Naser Abdullah Alam, *Phish No More: The Enron, Ling, CEAS, Nazario, Nigerian SpamAssassin Datasets*, https://www.kaggle.com/datasets/naserabdullahalam/phishing-email-dataset/data.

[26] A. Al-Subaiey, M. Al-Thani, N. A. Alam, K. F. Antora, A. Khandakar y S. A. U. Zaman, *Novel Interpretable and Robust Web-based AI Platform for Phishing Email Detection*, 2024. arXiv: 2405.11619 [cs.LG]. dirección: https://arxiv.org/abs/2405.11619.

[27] M. Somesha y A. R. Pais, "Classification of phishing email using word embedding and machine learning techniques," *Journal of Cyber Security and Mobility*, págs. 279-320, 2022.

[28] C. L. Tan, "Phishing dataset for machine learning: Feature evaluation," *Mendeley Data*, vol. 1, pág. 2018, 2018.

[29] C. Umbrella, *Cisco Popularity List*, https://umbrella-static.s3-us-west-1.amazonaws.com/index.html, 2024.

[30] G. for Developers, *Google Safe Browsing*, https://developers.google.com/safe-browsing, 2024.

[31] B. Guo, Y. Zhang, C. Xu, F. Shi, Y. Li y M. Zhang, "HinPhish: An Effective Phishing Detection Approach Based on Heterogeneous Information Networks," *Applied Sciences*, vol. 11, n.º 20, pág. 9733, oct. de 2021, ISSN: 2076-3417. DOI: 10.3390/app11209733. dirección: http://dx.doi.org/10.3390/app11209733.

[32] DomCop, *Open PageRank*, https://www.domcop.com/openpagerank/what-is-openpagerank, 2024.

[33] Y. Duan, N. Wang y J. Wu, "Minimizing Training Time of Distributed Machine Learning by Reducing Data Communication," *IEEE Transactions on Network Science and Engineering*, vol. 8, n.º 2, págs. 1802-1814, 2021. DOI: 10.1109/TNSE.2021.3073897.

[34] Y. Jung, "Multiple predicting K-fold cross-validation for model selection," *Journal of Nonparametric Statistics*, vol. 30, n.º 1, págs. 197-215, 2018. DOI: 10.1080/10485252.2017.1404598. eprint: https://doi.org/10.1080/10485252.2017.1404598. dirección: https://doi.org/10.1080/10485252.2017.1404598.

[35] I. K. Nti, O. Nyarko-Boateng, J. Aning et al., "Performance of machine learning algorithms with different K values in K-fold CrossValidation," *International Journal of Information Technology and Computer Science*, vol. 13, n.º 6, págs. 61-71, 2021.

CAPÍTULO 12

Anexos

12.1. Repositorio extensión

Aquí se muestra el repositorio donde se guardó la extensión utilizada en el desarrollo de este proyecto: `https://github.com/andresdlRoca/Guardian_Virtual_Extension`

12.2. Repositorio API

A continuación se muestra el repositorio donde se guardó la API a las que la extensión realizo consultas: `https://github.com/andresdlRoca/Guardian_Virtual`