Universidad de Lima

Facultad de Ingeniería

Carrera de Ingeniería de Sistemas

DETECCIÓN DE PHISHING EN CORREOS ELECTRÓNICOS MEDIANTE PROCESAMIENTO DE LENGUAJE NATURAL DEL CONTENIDO Y URLS OFUSCADAS

Tesis para optar el Título Profesional de Ingeniero de Sistemas

Adriana Cabezas Diaz

Código 20170227

Jerome Zack Ricardo Gutierrez Sisniegas

Código 20172204

Asesor

Oscar Efrain Ramos Ponce

Lima – Perú

Diciembre de 2023

Detección de Phishing en Correos Electrónicos mediante Procesamiento de Lenguaje Natural del Contenido y URLs Ofuscadas

Cabezas Diaz, Adriana
20170227@aloe.ulima.edu.pe
Universidad de Lima

Gutierrez Sisniegas, Jerome Zack
20172204@aloe.ulima.edu.pe
Universidad de Lima

Resumen: El *phishing* es un tipo de fraude informático común que, por medio de mensajes como correos electrónicos o similar, solicita el ingreso de información personal o el acceso a enlaces maliciosos. Frente a esta problemática, distintos autores han desarrollado modelos de detección de phishing en correos electrónicos basados en análisis de contenido que han demostrado tener altas tasas de detección. Sin embargo, los ciberdelincuentes aplican nuevas técnicas de phishing como el uso de URLs ofuscadas, que consiste en modificar la URL maliciosa para que parezca legítima mediante distintos métodos, como acortar la URL. La presente investigación tiene por objetivo general desarrollar dos métodos de detección de phishing en correos electrónicos mediante procesamiento de lenguaje natural, el primero enfocado en el análisis del contenido y el segundo enfocado en el análisis de URLs ofuscadas. Asimismo, se busca determinar cuál método incrementa la tasa de detección de phishing. Ambos modelos tienen una fase de preprocesamiento que extrae un conjunto de atributos relativos al contenido del correo electrónico y aplica técnicas de procesamiento de lenguaje natural sobre el contenido del correo electrónico. Para el entrenamiento de los modelos, se utilizaron tres datasets: “Enron”, “Spam Archive” y “Ebbu 2017”. Para la validación de los modelos, se tomaron en cuenta las métricas de precisión, sensibilidad, exactitud y puntaje del área debajo de la curva (AUC). Los modelos implementados lograron una precisión máxima del 97.02% y 98.70%. Además, se observó que los modelos propuestos presentaron resultados con una diferencia mínima para detectar phishing en correos electrónicos.

Palabras Clave: phishing, aprendizaje supervisado, PLN, URL ofuscada, emailing

Abstract: Phishing is a common type of computer fraud that, through methods like emails or similar messages, requests the input of personal information or access to malicious links. Faced with this issue, various authors have developed email phishing detection models based on content analysis that have demonstrated high detection rates. However, cybercriminals employ new phishing techniques such as the use of obfuscated URLs, which involve modifying malicious URLs to appear legitimate through various methods, such as URL shortening. The present research aims to develop two general methods for detecting phishing in emails using natural language processing. The first method focuses on content analysis, while the second concentrates on analyzing obfuscated URLs. Likewise, the objective is to determine which method enhances the phishing detection rate. Both models involve a preprocessing phase that extracts a set of attributes related to email content and applies natural language processing techniques to the email content. Three datasets, namely "Enron," "Spam Archive," and "Ebbu 2017," were used for model training. To validate the models, metrics such as precision, sensitivity, accuracy, and the area under the curve (AUC) score were considered. The implemented models achieved a maximum precision of 97.02% and 98.70%. Additionally, it was observed that the proposed models yielded results with minimal differences in detecting phishing in emails.

Keywords: phishing, supervised learning, NLP, Obfuscated URL, emailing

1. INTRODUCCIÓN

El *phishing* es un tipo de fraude lucrativo en el que el delincuente engaña a los destinatarios y obtiene información confidencial de ellos con falsas pretensiones (Rawal et al., 2017). Esta técnica puede ser utilizada a través de distintos medios de comunicación digital, como páginas web, aplicaciones, correos electrónicos, mensajes de texto, etc. Estos tipos de ataque con *phishing* son realizados gracias a la suplantación de identidad, la cual es un paso fundamental en el *phishing*, donde el atacante se hace pasar por una entidad confiable para ganarse la confianza de la víctima (Hu & Hang, 2018).

En los últimos años el incremento de ataques de *phishing* vía correo electrónico ha ido en aumento. Como indica el reporte del *Anti-phishing* Working Group-APWG (2022), el tercer cuarto de año del 2022, se encontró que los servicios de correo electrónico permanecen como uno de los mayores objetivos del *phishing* con 17% de todos los ataques realizados a servicios webmail y software como servicio (SaaS) online. En el caso de los correos electrónicos, el *phishing* consta de enviar comunicaciones que parezcan legítimas. Estos ataques son enviados a personas que utilizan información contextual relevante para engañarlos y que así revelen información confidencial a los atacantes o instalen malware en sus computadoras (Han & Shen, 2016).

Frente a este contexto, investigadores en la materia (Peng et al. (2018); Egozi y Verma (2018)) han buscado abordar la problemática haciendo uso de técnicas de procesamiento de lenguaje natural (PLN), el cual es un método utilizado para el análisis del cuerpo del correo electrónico por las ventajas que presenta. Una de estas ventajas es que el PLN nos permite detectar la intención de un correo electrónico, ya que se analiza de manera semántica el texto. Asimismo, otros autores como Jain y Gupta (2018) se enfocan más en la detección de URLs maliciosas que son enviadas a través de los correos electrónicos, ya que estos son utilizados de forma recurrente en ataques de phishing. Con esto los ciberdelincuentes logran engañar a la víctima, logrando que hagan click en uno de estos enlaces que les puede descargar un malware o redirigirlos a un sitio web falso. Sin embargo, frente a estos modelos de detección, cada vez son más los ciberdelincuentes que optan por utilizar URLs ofuscadas para evitar la detección de phishing.

Al ser el servicio de correo electrónico uno de los principales focos de ataque, el presente trabajo busca realizar una detección exitosa de *phishing* dentro de los correos electrónicos, mediante el análisis del contenido. Con esto se buscará que los usuarios que utilizan en su día a día correos electrónicos para sus actividades tengan mayor seguridad en sus comunicaciones. Si bien diversos investigadores (Peng et al. (2018); Egozi y Verma (2018); Jain y Gupta (2018)) han demostrado que los modelos de detección de phishing basados en análisis de contenido tienen buen rendimiento, es necesario investigar si estos modelos son suficientes para detectar el phishing cuando los ciberdelincuentes utilizan URLs ofuscadas o si es necesario un modelo de detección que se especialice en este tipo de ataques.

La presente investigación tiene por objetivo general desarrollar dos métodos de detección de phishing en correos electrónicos mediante procesamiento de lenguaje natural, el primero enfocado en el análisis del contenido y el segundo enfocado en el análisis de URLs ofuscadas. Se busca realizar este tipo de procesamiento en el contenido para determinar mediante el análisis de semántica y vectorización de palabras si los correos electrónicos presentan phishing o no. Asimismo, se busca determinar cuál método incrementa la tasa de detección de phishing.

La presente investigación se encuentra organizada de la siguiente manera. En la sección 2, se introduce la revisión del estado del arte, donde se presenta la literatura relevante relacionada con el tema investigado. La sección 3 presenta los antecedentes, donde se detallan los conceptos

fundamentales para el entendimiento de la investigación. La sección 4 abarca la metodología de modelo planteado, que describe desde la extracción de datos hasta la clasificación de los atributos. La sección 5 abarca los resultados obtenidos de todos los modelos que se implementaron. La sección 6 abarca la discusión de los resultados obtenidos. Finalmente, la sección 7 presenta las conclusiones de esta investigación.

2. ESTADO DEL ARTE

En esta sección se hará una revisión de métodos propuestos para la detección de phishing en correos electrónicos. Dentro de los métodos de detección de phishing en correos existen distintos enfoques que se pueden tomar en cuenta. Para fines de esta investigación se ha tomado en cuenta la literatura que abarca el análisis de URLs y análisis de contenido.

2.1. Métodos de análisis de URLs

Uno de los métodos de phishing utilizados por los ciberdelincuentes consiste en mandar localizadores de recursos uniformes (Uniform Resource Locator o URL en inglés) maliciosos en los correos electrónicos. Frente a esta problemática, Rathod y Nandy (2014) proponen un algoritmo llamado ObUrl, el cual consiste en realizar seis pruebas, las cuales son: prueba sobre el sistema de nombres de dominio (Domain Name System o DNS en inglés), prueba de dirección de protocolo de internet (Internet Protocol o IPs en inglés), URL codificada, URL recortada, prueba de lista negra o lista blanca y por último, prueba de coincidencias de patrones de URLs. Al realizar estas pruebas se puede determinar si es que una URL es maliciosa o que pueda redireccionar a un sitio web falso. Gracias a las distintas pruebas realizadas, su algoritmo tuvo una precisión de 95% de efectividad. Otra de las técnicas utilizadas por Sheng et al. (2009) es la utilización de listas negras y blancas para detectar si una URL es maliciosa. En este caso los autores proponen un método que aplica esta técnica de listas para probar la eficiencia de estos tipos de listas, usando una muestra de 200 URLs. Después de haber procesado sus modelos obtuvieron que las listas negras no protegían a los usuarios de manera eficiente, ya que obtuvieron un 30% de detección usando listas negras. Gracias a esta investigación se puede evidenciar que hacen falta modelos que examinan de manera más exhaustiva las URLs.

Buber et. al.(2017) propusieron un método para detectar ataques phishing analizando URLs a través de técnicas de procesamiento de lenguaje natural. Determinaron que con los atributos obtenidos de las técnicas de preprocesamiento acompañados de la vectorización de palabras se logra detectar mejor el phishing. Con la aplicación del algoritmo random forest obtuvieron resultados de 97.2%.

2.2 Métodos de detección de *phishing* basados en el análisis de contenido

Peng et al. (2018), proponen un método de detección de *phishing* en correos a través de PLN y *Machine Learning* aplicando el clasificador de Naive Bayes. El método propuesto realiza un análisis semántico del texto transmitido por el atacante para verificar la idoneidad de cada frase. Se considera maliciosa si la sentencia solicita información confidencial u ordena una acción que pueda exponer información personal. Los autores lograron probar la eficiencia de su método con 10000 correos electrónicos obteniendo 95% de precisión. Como resultado, el enfoque de los autores es eficaz para detectar correos electrónicos de *phishing* que se componen de texto puro y demuestra que la información semántica es un fuerte indicador de ingeniería social. Sin embargo, podríamos considerar que no todos los correos electrónicos se componen de texto puro, por lo que este método se vería limitado a esta característica.

Para combatir la amenaza de *phishing* mediante correo electrónico, Egozi y Verma (2018) proponen un modelo basado en procesamiento de lenguaje natural (PLN). El modelo propuesto consiste en dos fases: extracción de características y *machine learning*. Este modelo utiliza 26

atributos para determinar si los correos electrónicos son *phishing* o correos deseados. Si bien algunos de los atributos pueden no parecer efectivos, incluso atributos como el recuento de palabras pueden ser bastante efectivos por sí solos. Dado que los diferentes modelos de *machine learning* funcionan mejor en conjuntos de datos específicos, se probaron 17 modelos. De los 17 analizados, 14 dieron resultados aceptables, es decir que tuvieron una tasa de detección por encima del 80%. De los cinco modelos seleccionados, el Gaussian Naive Bayes no ponderado fue el que dio resultados más bajos, ya que solo detecta el 59% de los correos electrónicos de *phishing*; por lo tanto, tal modelo solo sería bueno para personas extremadamente cautelosas. Con su tasa de detección de *phishing* del 98%, el árbol de decisiones ponderado podría usarse en empresas con trabajadores que son menos conscientes de los correos electrónicos de *phishing*. Con solo un 77% de tasa negativa verdadera no debería ser usado en lugares de alta importancia.

Senturk et al. (2017) proponen un mecanismo de prevención contra ataques de *phishing* vía correo electrónico aplicando técnicas de *machine learning* y minería de datos. Primero se analizan los correos electrónicos teniendo en cuenta el cuerpo del correo electrónico. Después se calcula la probabilidad de *phishing* teniendo en cuenta características como existencia de símbolo de dólar, de URL, formularios, palabras clave (*login, password, link*). Por último, se aplica la minería de datos a cada correo electrónico con la ayuda de la herramienta WEKA que utilizando *supplied test* set clasifica los correos electrónicos. En los resultados se obtuvo un 89% de precisión para la clasificación. Se rescata de la investigación que los autores hacen énfasis en la necesidad de seguir desarrollando mecanismos de prevención de *phishing* en correos electrónicos, ya que descubrieron que muchos de los sistemas de prevención ya existentes han sido trabajados en datasets y ejemplos de lengua inglesa.

Rawal et al. (2017), proponen un modelo de clasificación de correos electrónicos de *phishing* haciendo uso de algoritmos de *machine learning*. El modelo propuesto extrae 9 características de todos los correos electrónicos que son: recuento de dominios, el número de links dentro del correo, presencia de javascript, presencia de html, presencia de etiqueta de formulario, número de palabras que incitan a una acción, presencia de las palabras: "paypal", "banco" y "cuenta". Para el modelo propuesto se utilizaron distintos clasificadores como: Support Vector Machines, Naive bayes, Random Forest, Logistic Regression y Voted Perceptron. En los resultados, se evidenció que SVM y Random Forest obtuvieron mejor rendimiento en términos de precisión con un 99.99% en clasificación de correos electrónicos *phishing* en comparación con otros clasificadores. Sin embargo, a pesar de obtener buenos resultados los autores recomiendan que es posible probar en datasets que contengan mayor cantidad de correos electrónicos para que los resultados se puedan asemejar más a la realidad.

Moradpoor et al. (2017) propusieron un modelo basado en redes neuronales para detectar y clasificar correos electrónicos de *phishing*. Cada correo electrónico del dataset fue clasificado por variables aleatorias como legítimo o de *phishing* y se guardaban en variables. Después, para cada correo clasificado se contaban los enlaces web en el cuerpo del correo electrónico ya sea un correo electrónico HTML o texto simple y el número de partes del correo electrónico. Se realizó purificación de cada correo electrónico y se procedió a la vectorización de palabras y cálculo del promedio de vectores para cada correo. Una vez determinadas todas las variables necesarias: el tipo de correo electrónico, el número de enlaces en el cuerpo del correo electrónico, si se trata de un correo HTML, el número de partes del correo electrónico y el promedio del vector se exportan como un conjunto de datos de tipo csv. Este conjunto de datos exportados se utilizó para entrenar, validar y probar el sistema basado en redes neuronales. Como resultados, se obtuvo que la red neuronal pudo clasificar con una precisión de 89.9% los correos electrónicos benignos y con un 94.4% los correos de *phishing*. De la investigación se rescata que muchos correos electrónicos benignos están cerca de los correos electrónicos de *phishing* por características de marcado HTML, texto en color y frases de *phishing* que se repiten, por lo que esto acarrea falsos positivos.

Smadi et al. (2018) proponen un modelo llamado PEDS (*Phishing* Email Detection System), el cual es un framework basado en técnicas de aprendizaje supervisado y no supervisado. La técnica de aprendizaje supervisado utiliza un dataset para construir el modelo de detección; mientras que, la técnica de aprendizaje no supervisado trata de adaptar el modelo utilizando un dataset con nuevos correos electrónicos recibidos, donde no se identifica cuál es de *phishing*. Este modelo (PEDS) utiliza una combinación de redes neuronales, minería de datos, aprendizaje reforzado y un conjunto de algoritmos, como listas negras y extracción de características del correo electrónico, para la detección de *phishing.* En la etapa de preprocesamiento se extrajeron distintas características del correo electrónico desde la cabecera hasta la firma. Los resultados de la experimentación con el modelo PEDS resultaron efectivos, con una precisión de hasta 98.6%, y demuestra cómo con redes neuronales y aprendizaje reforzado se puede crear un modelo efectivo como este.

Phomkeona y Okamura (2020), al identificar el problema del uso de las herramientas y servicios de correo electrónico, propusieron la creación de un modelo de detección de *phishing* de correos electrónicos basado en deep learning aplicando redes neuronales. El método propuesto consiste en el análisis de ciertas características en la cabecera y texto de los correos electrónicos, asimismo utiliza una bolsa de sujetos o datos. En la extracción de datos de la cabecera se recopilan tres características principales: la dirección de origen del correo, la etiqueta de tiempo y el asunto o título. Asimismo, del cuerpo del correo se extraen las siguientes características: URL, imagen y archivo adjunto. En este método se prueba ingresando las características de a poco, las experimentaciones fueron usando 4, 9, 12, 27 características. Los resultados fueron la detección de un correo electrónico malicioso con una efectividad de 66% y éste aumentaba dependiendo de las características ingresadas, por lo que se llegó a la conclusión de que mientras más características se procesen mejor es la detección. Sin embargo, esta última no aumenta de manera significativa, ya que se observó que la diferencia entre 12 características y 27 solo era 1%. Este método permite analizar todo el correo electrónico, lo cual hace que en la detección se pueda estar seguro de que se trata de un correo malicioso, pero el problema es que este método fue probado solo en correos electrónicos en ciertos lenguajes como inglés y japonés.

Alotaibi et al. (2020), proponen la detección de *phishing* mediante técnicas de deep learning. Plantean un modelo de clasificación de correo electrónico basado en redes neuronales convolucionales. Este modelo utiliza una fase de preprocesamiento para la creación de un dataset, en esta fase se extrae las etiquetas de sujeto y cuerpo del correo en formato html. Una vez completado el pre procesamiento de los emails los datos recopilados pasan a un análisis mediante redes neuronales donde se logra una clasificación de los distintos datos del correo electrónico mediante tareas de clasificación del modelo, las cuales incluyen: clasificación de documentación, categorización de texto corto y clasificación de intencionalidad. Después de haber realizado una experimentación se comprobó que el modelo tiene una efectividad de 99.42% en precisión al detectar un correo electrónico con intenciones de *phishing*. Este modelo es considerado prometedor, debido a que su efectividad alta compite con otros modelos similares como el propuesto por Rawal et.al.(2017) mencionado anteriormente, el cual se enfoca en la extracción de características de todos los correos electrónicos.

Las investigaciones disponibles resaltan que el análisis de correos electrónicos involucra diversas características y atributos, tanto en la cabecera como en el contenido del mensaje. Además, en el contexto del análisis mediante procesamiento de lenguaje natural, se lleva a cabo una clasificación de los correos electrónicos. Debido a esta modalidad de procesamiento, se opta por la utilización de los algoritmos de Random Forest (RF) y Support Vector Machines (SVM). Es importante

destacar que en las investigaciones en las cuales se emplearon dichos algoritmos, los resultados muestran una precisión notable, con un promedio de detección superior al 85%.

3. ANTECEDENTES

3.1 Phishing en correos electrónicos

El *phishing* puede ser definido como un tipo lucrativo de fraude en el que el delincuente engaña a los destinatarios y obtiene información confidencial de ellos con falsas pretensiones (Rawal et. al.,2017). Esta técnica puede ser utilizada a través de distintos medios de comunicación digital, como páginas web, aplicaciones, correos electrónicos, mensajes de texto, etc. En el caso de los correos electrónicos, el *phishing* consta de enviar correos electrónicos que parecen legítimos haciendo uso del spoofing. Estos ataques son enviados a personas que utilizan información contextual relevante para engañarlos para que revelen información confidencial a los atacantes o instalen malware en sus computadoras (Han & Shen, 2016). Dentro de las técnicas de clasificación que se utilizan en los métodos de phishing, resaltan las de *machine learning*. De estas podemos destacar algunas: Random Forest, Support Vector Machine, Naive Bayes, entre otras.

3.2 Procesamiento de lenguaje natural (PLN)

Razno (2019) define el procesamiento del lenguaje natural como un subcampo de la informática e inteligencia artificial que se ocupa de las interacciones entre computadoras y lenguajes humanos (naturales), en particular, cómo programar computadoras para procesar y analizar grandes cantidades de datos en lenguaje natural. Verma et. al. (2012) mencionan que determinar el contexto del mensaje del correo electrónico es un papel importante en la detección de phishing en correos electrónicos. Por esto mismo, la aplicación de PLN, se ha convertido en los últimos años en parte de las distintas soluciones que se han formulado para detectar el phishing. Estos autores también mencionan algunas de las principales técnicas de procesamiento de lenguaje natural que se utilizan en este ámbito:

1. Análisis léxico: Consiste en dividir el texto del correo electrónico en oraciones y dividir estas oraciones en palabras.
2. Etiquetado de parte del discurso de Stanford: etiquetar cada palabra con su parte del discurso usando el etiquetador POS de Stanford.
3. Reconocimiento de entidad nombrada: para identificar las entidades nombradas, que incluyen sustantivos propios, como nombres de organizaciones, personas o ubicaciones
4. Normalización de palabras a minúsculas.
5. Derivación: convertir cada palabra en su raíz haciendo uso de algoritmos como Stemmer de Porter (por ejemplo, reducir el verbo "seen" a "see").
6. Eliminación de palabras vacías (Stopwords): para eliminar palabras que aparecen con frecuencia, como "a", "an", etc. usando una lista de palabras vacías.

Normalmente, el procesamiento de lenguaje natural necesita algunas acciones de normalización del texto para que estos puedan ser analizados de una manera más sencilla. El proceso más conocido que se realiza es el de tokenización. Talame et. al. (2019) definen tokenización como el proceso de dividir un documento de texto en sus distintos componentes, descartando los espacios en blancos y saltos de línea. Por ejemplo, si tenemos la oración ¨Ingresa tu contraseña¨, esta tendría 3 tokens: ¨Ingresa¨, ¨tu¨, ¨contraseña¨.

Jalilifard et. al (2021) menciona que dentro de los proyectos de PLN, para poder ajustar el performance se calcula el TF-IDF (Term Frequency - Inverse Document Frequency). El uso de la técnica TF-IDF se utiliza para determinar la importancia de una palabra para un documento en una colección de documentos. Asimismo, la importancia de una palabra aumenta proporcionalmente

al número de veces que una palabra aparece en el documento (frecuencia de término) y es inversamente proporcional a la frecuencia de documento de la palabra en la colección. La IDF es una medida del poder de discriminación del término. Mide cómo un término común se encuentra en toda una colección de documentos. Por tanto, un término tiene un alto peso de TF-IDF al tener una frecuencia de término alta en un documento dado y una baja frecuencia de documentos en toda la colección de documentos.

3.3 Ofuscamiento de URL

Hoy en día, la mayoría de los ataques de phishing se realizan mediante un ataque de phishing de ofuscación de URL (IBM, 2020). Debido a los diferentes métodos utilizados en este ataque, el algoritmo de verificación en los servicios de correos electrónicos existente no puede detectar todas las URL ofuscadas (Volkamer et. al., 2017). La ofuscación de URLs consiste en modificar la URL maliciosa para que parezca legítima mediante métodos como: acortar URL, utilizar iframe en el correo electrónico, cambiar a dirección IP en lugar del nombre de dominio, utilizar URL codificada, utilizar formulario de entrada en el correo electrónico, entre otros métodos que no pueden ser detectados por los algoritmos de verificación de los servicios de correo electrónico.

Según Rathod, J. (2014), los ataques mediante ofuscamiento de URL siguen 4 pasos:

1. Primero el atacante debe crear un sitio web ilegítimo para que la víctima crea que está en un ambiente seguro.
2. Luego el URL de este sitio web es enviado mediante correo electrónico y el contenido del correo electrónico convencerá a la víctima de que haga clic en esa URL.
3. Si la víctima hace clic en esa URL ofuscada y visita este sitio web, el atacante convence a la víctima de que ingrese información confidencial.
4. Finalmente el atacante adquiere los datos ingresados y luego los utiliza para fines maliciosos, usualmente para lucrarse o extorsionar a los afectados.

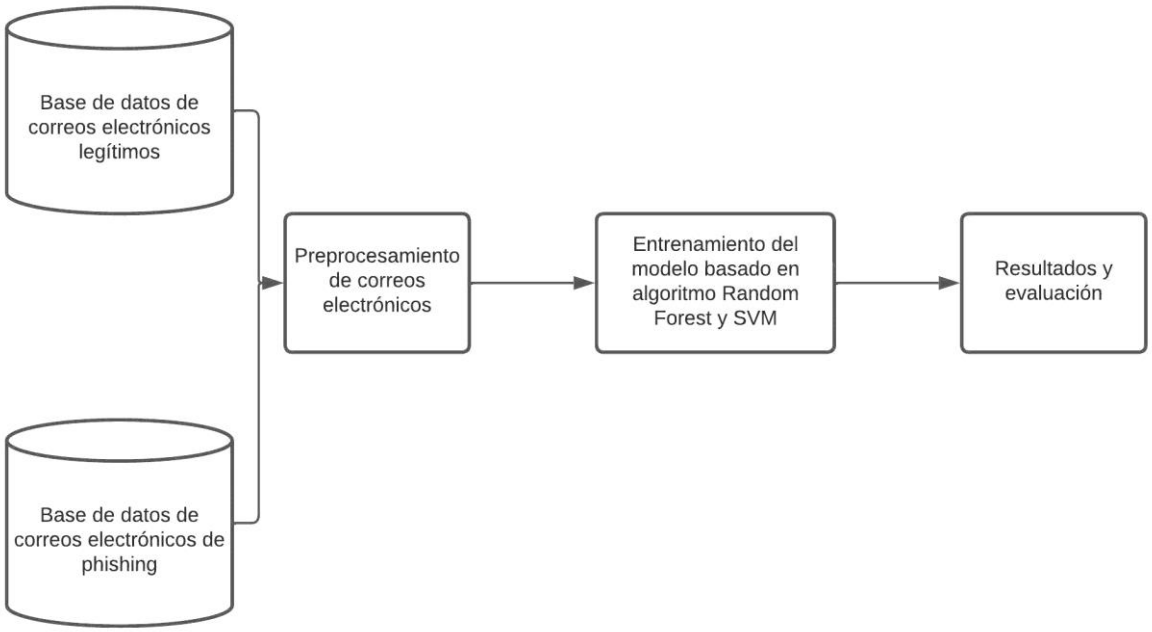
4. METODOLOGÍA Y EXPERIMENTACIÓN

A continuación, se describe la metodología seguida para los modelos basados en el análisis de contenido de correo electrónico y el análisis de URLs ofuscadas.

El método basado en análisis de contenido de correos electrónicos se trabajó en base a dos datasets, el primero llamado “Spam Archive”, cuya procedencia es http://untroubled.org/spam/, y el segundo llamado “Enron”, proveniente de https://www.cs.cmu.edu/~enron/. Estos datasets contienen correos electrónicos ilegítimos y legítimos, respectivamente. De estos datasets se extrae el contenido del correo (mensaje del correo electrónico). Este atributo pasa a la etapa de preprocesamiento donde además de eliminar caracteres innecesarios y eliminar nulos se utilizan técnicas PLN, las cuales se detallan en las siguientes secciones. Una vez completada la etapa de preprocesamiento, se procede a entrenar el modelo mediante los algoritmos de Random Forest (RF) y Support Vector Machine (SVM). Finalmente, se realiza la etapa de evaluación y resultados en base a la efectividad del modelo. En la Figura 4.1, se observa el diagrama de flujo de la metodología para el procesamiento de correos electrónicos.

Figura 4.1

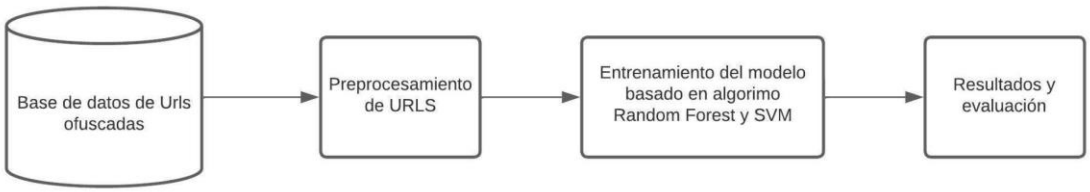
Diagrama de Flujo del Modelo Propuesto para el Procesamiento de contenido de Correos Electrónicos



El modelo basado en el análisis de URLs ofuscadas se desarrolló utilizando un dataset llamado "Ebbu 2017", cuya procedencia es https://github.com/ebubekirbbr/pdd/tree/master/input. Este, al igual que la metodología seguida en el modelo basado en análisis de contenido del correo electrónico y como en la Figura 4.2, pasa por las etapas de preprocesamiento, entrenamiento del modelo (basado en los algoritmos RF y SVM) y evaluación y resultados. Sin embargo, lo realizado internamente en la etapa de preprocesamiento varía en cada modelo y se explica a detalle en la sección 4.2.

Figura 4.2

Diagrama de Flujo del Modelo Propuesto para el Procesamiento de URLs ofuscadas



4.1 Datasets

Para el modelo basado en análisis de contenido, se utilizan como entradas dos datasets: Spam Archive y Enron. Spam Archive contiene 1619 correos electrónicos en formato EML, el cual es un tipo de formato que preserva el formato y las cabeceras HTML originales, que son ilegítimos, es decir que contienen *phishing.* De estos correos solo 299 contienen URLs. El segundo dataset llamado Enron fue creado por la fundación de software de Apache y contiene una cantidad de 2012 correos electrónicos que son legítimos y solo 352 contienen URLs. Asimismo, los correos electrónicos que componen los datasets constan de dos partes: cabecera y contenido; las cuales son diferenciadas por las palabras “From” y “Message”, respectivamente. Dentro de la cabecera podemos encontrar los atributos: From, To, Date, Subject, Return-Path, Domain Key and DKIM Signatures, Message-ID, MIME-Version y Received. En el caso del contenido este tiene los atributos body, alias message, y content type.

Para analizar el dataset Spam Archive, se procedió a analizar cada correo electrónico con el uso de la librería “parser” y su función "email.parser", la cual nos permite capturar el texto de un correo electrónico a la par que se extraía el atributo “contenido”. Para poder realizar esta extracción se crearon tres funciones, las cuales recorren el correo electrónico y encuentran los atributos, para extraer el contenido dependiendo del tipo de formato del contenido (html, plain o multipart). Los atributos extraídos fueron almacenados en un dataset con nombre "Fraudulentos", al cual se le agregó una columna que identifica estos correos como de *phishing* con el valor 1. En el caso del dataset Enron, se analizó cada correo electrónico con el uso de "email.parser", extrayendo de estos el atributo “contenido” haciendo uso de las mismas funciones utilizadas anteriormente con el dataset de correos ilegítimos. Estos atributos fueron almacenados en un dataset con nombre "Legítimos", al cual se le agregó una columna que identifica estos correos como libres de *phishing* con el valor 0.

Se procedió a concatenar los dataset generados (“Fraudulento” y “Legítimo”) en uno nuevo con el nombre de "data_set_crudo" resultando con un total de 3631 filas. Dentro del dataset generado, se logró evidenciar que existían valores vacíos en la columna "Contenido"; por lo que, se tomó en cuenta llenar estos valores en los siguientes pasos con el valor "NaN".

Para el modelo basado en análisis de URLs, se utiliza de entrada el dataset Ebbu. Este contiene 73575 URLs, de los cuales 36400 son URLs de sitios web legítimos y 37175 son URLs ofuscados utilizados con fines de phishing. Para el análisis de estas URLs, primero se transformó el dataset de formato JSON a formato CSV. Asimismo, se le agregó una columna con el nombre “Phishing” para identificar si era o no legítimo. A las URLs legítimas, se les asignó el valor de 0 y a las ilegítimas el valor de 1.

4.2 Preprocesamiento

Para las pruebas realizadas de la metodología se utilizó Python.

4.2.1 Contenido del correo

Para el procesamiento del modelo basado en contenido, se realizó un análisis de forma individual para cada uno de los correos contenidos en el dataset “data_set_crudo”, el cuál se describió en la sección 4.1. En el caso de los correos electrónicos para cada uno se aplicaron técnicas de PLN (Análisis léxico, eliminación de puntuación, normalización de palabras, eliminación de palabras vacías y tokenización). El procedimiento se realizó de la siguiente manera.

Primero, se realizó la extracción del atributo “Contenido”, luego se procedió a eliminar la puntuación. Para realizar esto se colocó los correos en una lista, la cual se recorrió para encontrar símbolos de puntuación. Una vez encontrados estos símbolos son eliminados de la lista. Asimismo, se procedió a normalizar palabras convirtiéndo cada letra a minúscula, y también se

procedió a eliminar palabras vacías (*stopwords*). Para la eliminación de las *stopwords* se comparaba la lista de palabras con una lista de stopwords, para este caso como eran correos en inglés se utilizaron *stopwords* en inglés, como "the", "and", "in" , etc. Esto último, se logró haciendo uso de la librería nltk para validar que las palabras no estuvieran en la lista de stopwords. Esta última librería permite llevar a cabo muchas tareas relacionadas con el PLN como: tokenización, lematización, parseo, etiquetado POS, etc. Después de este proceso, se realizó la tokenización del atributo contenido; es decir, se dividió en unidades más pequeñas (palabras). Por último, teniendo el atributo "contenido" tokenizado, se procedió a convertir el texto en una matriz que contiene el número de veces que se repite cada token (la palabra) haciendo uso de la técnica de Bolsa de Palabras la cual se aplica a través de la función "Count Vectorizer", que está disponible en la librería scikit-learn y proporciona algoritmos de aprendizaje supervisados y no supervisados. En la Tabla 4.1 se muestra un ejemplo de matriz de bolsa de palabras".

Tabla 4.1

Ejemplo de Bolsa de Palabras

	My	Goal	Is	Big
My Goal is Big	1	1	1	1
My Goal Big	1	1	0	1

4.2.2 URLs

Para el procesamiento de las URLs, se realizó un análisis de forma individual para todas las contenidas en el dataset Ebbu. Teniendo las URLs se aplicaron las pruebas descritas a continuación para poder tener la verdadera URL, en caso estas se encontraran ofuscadas. Para desofuscar las URLs; es decir, obtener las URLs verdaderas a las que alguna URL acortada o dirección IP hicieran referencia, se utilizó la función requests. Esta última función está disponible en python al importar la librería requests la cual permite enviar solicitud a las URL designadas y obtener como respuesta la verdadera URL asociada. Otro de los procedimientos que se realizó fue extraer la URL que estaba contenida dentro de una palabra. Esto se realizó mediante la comparación de palabras contra una variable que contenía el formato de URL, el cual si es que empezaba por "http" o se encontraban puntos seguidos de una palabra sin espacios dentro de esta, se podía saber que las palabras hacían referencia a una URL.

Por último, a las URLs se les aplica un método de PLN(Tokenización); ya que, sobre estas se usa un tokenizador personalizado. Este tokenizador divide las partes de la URL después de los caracteres: "/" , "-", ".". Asimismo, elimina los tokens redundantes y los "com". Por último, se utilizó la técnica TF-IDF, y para la vectorización de los datos se utilizó la función "TfidfVectorizer" con el uso de la librería scikit-learn. Esta matriz generada, contiene las ponderaciones de la importancia de cada palabra según su frecuencia de aparición en el documento.

4.3 Entrenamiento de los modelos usando algoritmos de Machine Learning

Para los modelos propuestos se decidió implementar dos algoritmos de clasificación de *machine learning*: RF y SVM. Los dos algoritmos se utilizaron para clasificar los datos resultantes de la

etapa de preprocesamiento. Para ambos modelos se hizo la división del dataset de tal manera que el 20% de los datos sea usado para las pruebas del modelo y el 80% restante para su entrenamiento.

4.3.1 Random Forest

Para poder alimentar el modelo, primero se definieron los parámetros mediante el método de grid search, el cual consiste en realizar la prueba del modelo con los distintos valores que puede tomar un parámetro hasta encontrar la combinación que tenga más porcentaje de precisión. En este caso las pruebas realizadas se pueden observar en la Tabla 4.1, donde se muestran los distintos parámetros que se utilizaron y los valores que se probó para cada parámetro. En el caso del número de estimadores se eligió un rango entre 10 a 800 estimadores, debido a que se realizaron pruebas con cantidades mayores a 800, en las cuales se pudo identificar que el mejor porcentaje de precisión se obtiene en un número de estimadores menor a 800. En el caso de los valores del criterio se utilizaron los únicos valores que puede tener este parámetro para verificar cuál brinda la mejor precisión. Se consideró valores entre 2 y 40 para el parámetro de mínimo de muestras necesaria para dividir un nodo, debido a que, según Mantovani et. al (2018), los valores ideales de para este parámetro tienden a estar entre 1 y 40 para el algoritmo CART, que es el algoritmo que se está utilizando para los árboles de decisión que forman parte de RF. Por último, para el parámetro de número mínimo de muestras necesaria para estar en un nodo hoja, el mismo autor afirma que los valores ideales de este parámetro tienden a estar entre 1 y 20 para el algoritmo CART. Asimismo, también indica que estos dos últimos parámetros descritos son los principales responsables del rendimiento de los árboles finales a partir de su análisis de importancia relativa.

Tabla 4.2

Parámetros del algoritmo RF

Parámetro	Descripción	Valores probados
Número de estimadores	Cantidad de árboles en el bosque.	De 10 a 800
Criterio de calidad	Mide la calidad de una división.	Gini y Entropía
Mínimo de muestras para dividir un nodo	El número de nodos en los que se dividirá un nodo predecesor. Permite controlar el sobreajuste.	2, 5, 10, 20, 30, 40
Número mínimo de muestras para un nodo hoja	El número mínimo de muestras requeridas para estar en un nodo hoja. Un punto de división sólo se considerará si deja al menos un número mínimo de muestras de entrenamiento en cada una	1, 2, 5, 10, 20

	de las ramas. Esto puede suavizar el modelo.

Después de realizar la evaluación por grid search utilizando 5 folds se obtuvieron los siguientes parámetros para el modelo de análisis de contenido: El criterio a utilizar será “Entropía”, el Número de estimadores será 4, la profundidad máxima será 4 y el mínimo de muestras necesarias para dividir un nodo serán 2.

Para la evaluación de la precisión del análisis de URLs se realizó el mismo procedimiento para definir los mejores parámetros que para el algoritmo de RF. Gracias a este procedimiento se obtuvieron los siguientes parámetros: El criterio a utilizar será “Entropía”, el número de estimadores será 448, la profundidad máxima será 4 y el mínimo de muestras necesarias para dividir un nodo serán 2.

4.3.2 Support Vector Machine

Para determinar los mejores parámetros al igual que con el modelo de SVM, se utilizó grid search utilizando 5 folds. En este caso las pruebas realizadas se pueden observar en la Tabla 4.2, donde se muestran los distintos parámetros que se utilizaron y los valores que se probaron para cada parámetro. Los valores utilizados para cada parámetro son los que sugiere la documentación de la librería scikit-learn.

Tabla 4.3

Parámetros del algoritmo SVM utilizados en el Grid Search.

Parámetro	Descripción	Valores utilizados
C	Compromiso entre el error de entrenamiento y la complejidad del modelo.	0.001, 1, 52 ,10, 100
Gamma	Coeficiente de kernel para "rbf", "poly" y "sigmoide".	0.001, 0.1 y 0.5
Kernel	Función para la separación de los grupos (transformación).	linear, poly, rfb y sigmoid
Grado de la función polinomial	Grado de la función del núcleo polinomial ("poly").	3 y 8
Coeficiente	Término independiente en función del kernel. Solo es significativo en "poly" y "sigmoide".	0.001, 0.10 y 0.5

Después de realizar la evaluación por grid search se obtuvo lo siguientes parámetros para el modelo de análisis de contenido:

- C: 52

- Grado: 3
- Kernel: Poly
- coef= 0.1

Para la evaluación de la precisión del análisis de URLs se realizó el mismo procedimiento para definir los mejores parámetros que para el algoritmo de SVM. Gracias a este procedimiento se obtuvieron los siguientes parámetros:

- C: 10
- Kernel: Linear

4.4 Evaluación de resultados

La evaluación del modelo se realizó en base a los datos que se destinaron a testeo. Se utilizó una matriz de confusión de los datos resultantes para realizar un análisis de las siguientes métricas: precisión, sensibilidad y exactitud del modelo. El cálculo de estas métricas se realizó en base a las siguientes reglas: Verdaderos Positivos (VP), resultados que el modelo declara positivos y son verdaderamente positivos;Verdaderos Negativos (VN), resultados que el modelo declara negativos y son verdaderamente negativos; Falsos positivos (FP), resultados que el modelo declara positivos y en realidad son negativos; y Falsos negativos (FN), resultados que el modelo declara negativos y en realidad son positivos.

Asimismo, para realizar una mejor evaluación se utilizó la curva ROC del modelo, la cual representa la razón o proporción de verdaderos positivos frente a la razón o proporción de falsos positivos. Se tomó como métrica el puntaje del área debajo de la curva (AUC), la cual proporciona una medida agregada del desempeño en todos los umbrales de clasificación posibles. Una forma de interpretar el AUC es como la probabilidad de que el modelo clasifique un ejemplo positivo aleatorio más alto que un ejemplo negativo aleatorio.

5. RESULTADOS

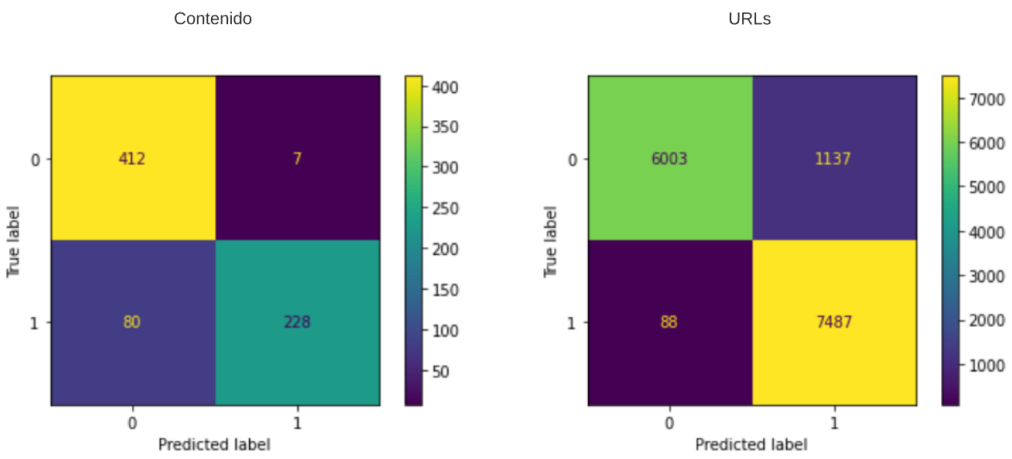
A continuación, se presentan los resultados obtenidos para todos los modelos de análisis de contenidos y análisis de URLs ofuscadas, para los cuales se utilizó los algoritmos de RF y SVM. Cabe recalcar, que las matrices de confusión y resultados se generaron a partir de la data destinada al testeo de cada modelo implementado. Asimismo, para todas las matrices de confusión la etiqueta de “0” hace referencia a los correos y URLs legítimas; mientras que, la etiqueta de “1” refiere a los correos y URLs de phishing.

5.1 Resultados obtenidos para RF

En la Figura 5.1, tenemos la matriz de confusión del modelo basado en el análisis de contenido. Se puede observar que, de los 727 datos analizados, 412 se clasificaron como Verdaderos Negativos (VN); es decir, como correos legítimos de forma correcta. Así también, se obtuvieron 228 Verdaderos Positivos (VP); es decir, se clasificaron como correos de *phishing* de forma correcta. Sin embargo, se obtuvieron 7 Falsos Positivos (FP) y 80 Falsos Negativos (FN), los cuáles hacen referencia a los correos legítimos que se clasificaron como *phishing* y los correos de *phishing* que se clasificaron como legítimos, respectivamente. Por otro lado, como se observa en la Fig. 5.1, para el modelo basado en análisis de URLs ofuscados se obtuvieron: 6003 VN, 7487 VP, 1137 FP y 88 FN.

Figura 5.1

Matrices de confusión de los modelos basados en análisis de contenido y análisis de URLs que utilizan RF

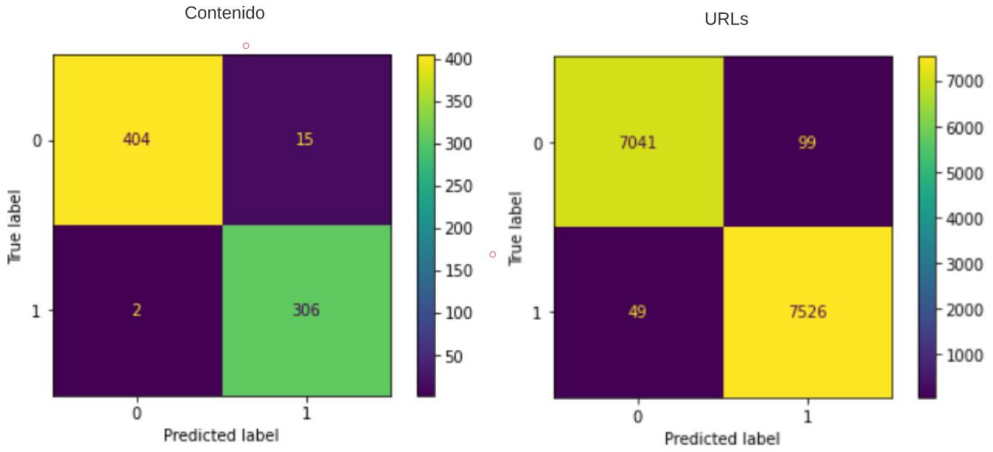


5.2 Resultados obtenidos para SVM

En la Figura 5.2, tenemos la matriz de confusión del modelo basado en el análisis de contenido. Se puede observar que de los 727 datos analizados, se obtuvieron 404 VN, 306 VP, 15 FP, 2 FN. Por otro lado, como se observa en la Fig.5.2, para el modelo basado en análisis de URLs ofuscados se obtuvieron 7041 VN, 7526 VP, 99 FP, 49 FN.

Figura 5.2

Matrices de confusión de los modelos basados en análisis de contenido y análisis de URLs que utilizan SVM.



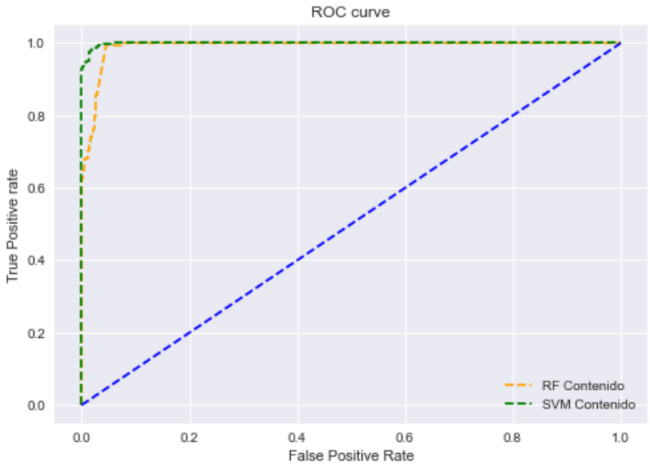
5.3 Resultados de Curva de ROC

En la Figura 5.3, se puede observar la curva ROC generada para comparar el modelo basado en análisis de contenido que aplica RF contra el modelo basado en análisis de contenido que aplica SVM. Como se puede observar, las líneas que representan los modelos están más orientadas a los 90 grados debido al alto número de detección de verdaderos positivos que se evidenció en las matrices de confusión, con lo cual se puede verificar que la efectividad de detección es alta. Asimismo, estas curvas se encuentran por encima de la recta azul, también conocida como línea de estimación aleatoria, que separa el área óptima de la evaluación de curva ROC y, en este caso, valida que los modelos que se están evaluando tienen mayor poder predictivo. Sin embargo, el modelo que aplica SVM tiene mayor distancia de la línea de estimación aleatoria; por lo que, tiene un mejor poder predictivo, pero en este caso al ser mínima la diferencia se puede ver que ambos modelos tienen un poder predictivo muy aceptable ya que es mayor al 90%. Además, para

respaldar esta comparación se tiene los puntajes AUC los cuales fueron 99.05% y 99.88% para RF y SVM respectivamente.

Figura 5.3

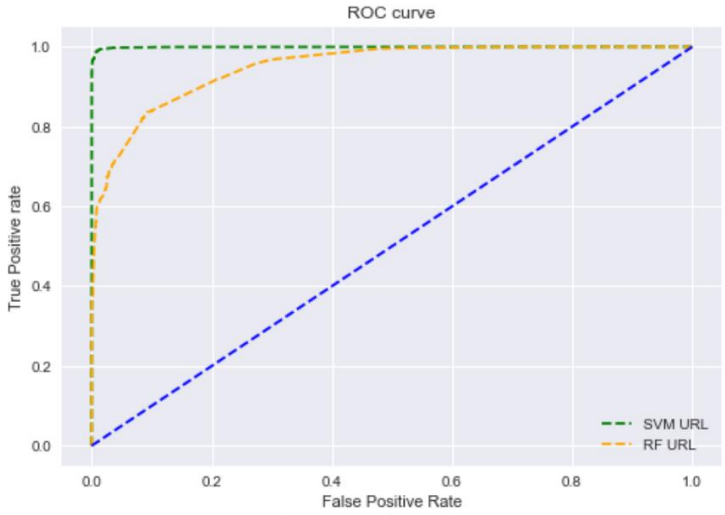
Curva ROC de comparación de los modelos RF y SVM basados en el análisis de contenido



En la Figura 5.4, se puede observar la curva ROC generada para comparar el modelo basado en análisis de URLs que aplica RF contra el modelo basado en análisis de URLs que aplica SVM. Se puede apreciar un escenario similar al que se presenta en el modelo de análisis de contenido donde la curva ROC del modelo de SVM tiene un mejor poder de detección de verdaderos positivos. Como se puede observar, la linea de color verde que representa al modelo de SVM, se acerca más a los 90 grados, por lo cual nos da a entender que tiene mayor precisión que el modelo de RF. Asimismo el puntaje AUC demuestra que la diferencia entre el modelo de RF contra el modelo SVM es de 4.51%, ya que se obtuvieron resultados de 94.98% y 99.49% para RF y SVM respectivamente. A pesar de que exista una diferencia entre los puntajes, se considera que ambos modelos son aceptables ya que tienen un puntaje mayor al 90%.

Figura 5.4

Curva ROC de comparación de los modelos RF y SVM basados en el análisis de urls ofuscadas



5.4 Resultados comparativos de los modelos

En la Tabla 5.1, se tienen los resultados para el modelo basado en análisis de contenido y el modelo basado en análisis de URLs que aplican RF. Dentro de estos resultados se tiene las métricas de precisión, sensibilidad, exactitud y puntaje AUC. Para estimar las métricas de análisis de precisión, sensibilidad y exactitud se utilizaron los ratios de FN (Falsos Negativos), VP (Verdaderos Positivos), FP (Falsos Positivos) y VN (Verdaderos Negativos).

Tabla 5.1

Resultados Comparativos para RF

Modelo/Métricas	Precisión	Sensibilidad	Exactitud	Puntaje AUC
Contenido	97.02%	74.02%	88.03%	99.05%
URLs	86.81%	98.83%	91,67%	94.98%

Como se puede observar en la Tabla 5.1, se evidencia que para ambos modelos se obtuvo un desempeño mayor al 70% en las métricas evaluadas. Por un lado, el modelo basado en análisis de contenido que aplica RF obtuvo 97.02% de precisión, 74.02% de sensibilidad, 88.03% de exactitud y 99.05% de puntaje AUC. Mientras que el modelo basado en análisis de URLs que aplica RF obtuvo 86.81% de precisión, 98.83% de sensibilidad, 91.67% de exactitud y 99.49% de puntaje AUC. Mientras que el modelo de contenido presenta mejor precisión, el modelo de URLs presenta mejores resultados en cuanto a sensibilidad y exactitud. Si tomamos la métrica de puntaje AUC, la cual es utilizada para comparar modelos, el modelo basado en análisis de contenido que aplica SVM presenta mejores resultados con un 99.88%.

En la Tabla 5.2, se tienen los resultados para el modelo basado en análisis de contenido y el modelo basado en análisis de URLs que aplican SVM. Dentro de estos resultados se tiene las métricas de precisión, sensibilidad, exactitud y puntaje AUC. Para estimar las métricas de análisis de precisión, sensibilidad y exactitud se utilizaron los ratios de FN, VP, FP y VN.

Tabla 5.2

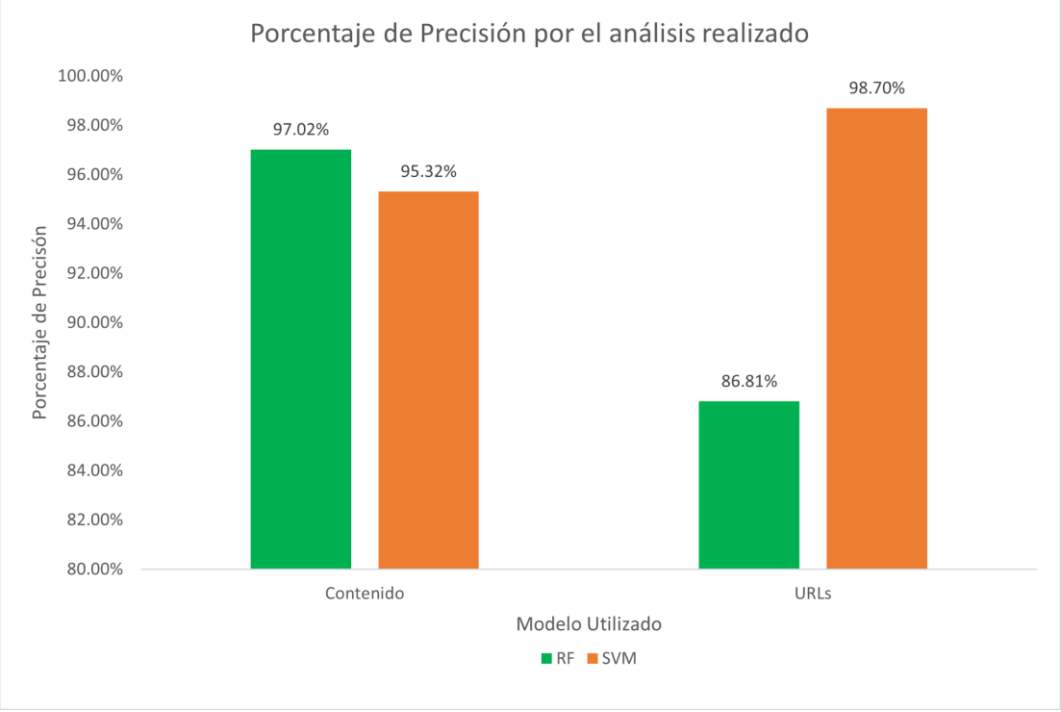
Resultados Comparativos para SVM

Modelo/Métricas	Precisión	Sensibilidad	Exactitud	Puntaje AUC
Contenido	0.9532	0.9935	0.9766	0.9988
URLs	0.9870	0.9980	0.9899	0.9949

Como se puede observar en la Tabla 5.2, se evidencia que para ambos modelos se obtuvo un buen desempeño en las métricas evaluadas con resultados mayores a 80%. Por un lado, el modelo basado en análisis de contenido que aplica SVM obtuvo 95.32% de precisión, 99.35% de sensibilidad, 97.66% de exactitud y 99.88% de puntaje AUC. Mientras que el modelo basado en análisis de URLs que aplica SVM, obtuvo 98.70% de precisión, 99.80% de sensibilidad, 98.99% de exactitud y 99.96% de puntaje AUC. Aunque la diferencia de los resultados entre los modelos no es grande, si tomamos la métrica del puntaje AUC, la cual es utilizada para comparar modelos, el modelo basado en contenido que aplica SVM presenta mejores resultados con un 99.88%.

Figura 5.5

Gráfico comparativo de los porcentajes de precisión obtenidos por cada modelo para los análisis de contenido y URLs



En cuanto a la precisión de los modelos de RF y SVM, como se observa en la Figura 5.5, cada algoritmo destaca en uno de los análisis realizados, en cuanto al algoritmo de SVM, este tiene una mayor precisión en el análisis de URLs con un 98.70% y el algoritmo de RF tiene un mejor resultado en el análisis de contenido con un 97.02% de precisión.

6. DISCUSIÓN

De los resultados obtenidos en esta investigación se puede deducir que un modelo de detección de phishing basado en contenido, que contenga tanto palabras importantes como urls, presenta ligeramente mejores resultados (99.05% - 99.88%) frente a un modelo que analice solo URLs ofuscados (94.98% - 99.49%), tomando en cuenta la métrica puntaje AUC, lo cual nos indica que el modelo suele ser más predictivo. Sin embargo, tomando en cuenta las métricas de precisión, sensibilidad y exactitud el modelo de detección de URLs ofuscadas tiene el mejor porcentaje en comparación al modelo de análisis de contenido de los correos electrónicos (98.70% - 86.81%, 99.80% - 98.83%, 98.99% - 91.67% respectivamente). Asimismo, con respecto a la métrica de precisión, la cual se enfoca en la proporción de verdaderos positivos a los positivos predichos, el modelo basado en el análisis de contenido tiene mejores resultados tanto en el algoritmo de RF como el de SVM con 97.02% y 95.32%, respectivamente. Esto nos indica que el análisis de contenido nos ofrece un mejor rendimiento en la detección de phishing, ya que se requiere una precisión extremadamente alta para que los correos electrónicos relevantes e importantes no sean clasificados como maliciosos y luego sean descartados. Asimismo, se observó que los casos principales en donde los algoritmos clasifican de manera errónea los correos electrónicos, es cuando hay presencia de imágenes con URLs indexadas, cuando el correo electrónico tiene formato html con referencia a distintas webs y cuando el texto contiene palabras que dan índice de phishing en un correo legítimo.

En cuanto a la métrica de sensibilidad se puede observar resultados altos utilizando el algoritmo de SVM para ambos modelos (99.35% y 99.80%). Sin embargo, el algoritmo de RF presentó unos resultados más bajos, especialmente en el modelo de análisis de contenido con un 74.02%. Este resultado nos indica que a pesar de que detecta bien los correos electrónicos con intención de phishing, se está teniendo un resultado elevado en la clasificación de correos electrónicos de phishing como legítimos.

Con respecto a la métrica de exactitud, se tiene un caso similar a la métrica de sensibilidad donde existen resultados altos usando el algoritmo de SVM (97.66% y 98.99%), pero en el algoritmo de RF para el modelo de análisis de contenido tiene 88.03% de resultado, lo cual es bajo con respecto a los demás. Esto nos quiere decir que, utilizando RF el modelo de análisis de contenido tienen menos detecciones correctas en comparación al otro modelo. Asimismo, podemos decir que de manera general y el modelo que obtiene mejores resultados en cuanto a predicciones correctas es el modelo basado en el análisis de URLs ofuscadas.

De la investigación realizada podemos notar que una de las principales diferencias en cuanto a otros trabajos que tienen como enfoque detectar URLs maliciosas, es que a los parámetros utilizados para el entrenamiento del modelo se les realizó un preprocesamiento, donde además de realizarle pruebas (DNS, desacortar URLs) para obtener las verdaderas URLs se aplicaron técnicas de PLN. En contraste, investigaciones como las de: Rathod y Nandy (2014) y Jain y Gupta (2018) solo hicieron pruebas de DNS, URLs acortadas, IP, número de caracteres en la URL o la presencia de protocolo HTTPS, si bien el análisis de estos protocolos pueden devolver la URL original, la falta del procesamiento de PLN evita que se puedan analizar los casos en los que los ciberatacantes crean dominios similares y con intenciones de phishing. Asimismo, un factor importante a tener en cuenta es que los datasets utilizados en dichos trabajos son distintos a los utilizados en la presente investigación. Respecto a la efectividad de los parámetros utilizados en las investigaciones anteriormente mencionadas, lograron un 85% y 92% de precisión de clasificación.

Por otro lado, en la presente investigación se alcanzó 98.70% de precisión en el modelo basado en análisis de URLs que aplica SVM, lo que evidencia que la aplicación de un análisis basado en procesamiento de lenguaje natural en las URLs podría beneficiar a la detección de phishing.

Al comparar los resultados obtenidos en esta investigación con trabajos que aplican un enfoque similar basado en análisis de contenido aplicando técnicas PLN y Machine Learning, se encontró que los modelos propuestos por Peng et al. (2018) y Egozi y Verma (2018) lograron una precisión de 95% y 98%, respectivamente. A diferencia de la presente investigación, en el modelo propuesto por Peng et al. (2018), el cual también utilizó el dataset "Enron", se realizó un análisis de texto puro de los correos electrónicos dejando de lado la posible existencia de URLs en ellos. Esto último, puede asociarse a la causa principal de la precisión obtenida (95%). Por otro lado, al igual que en el modelo Egozi y Verma (2018), en esta investigación, a pesar de utilizar bases de datos distintas, se obtuvieron resultados similares en contraste con el modelo basado en análisis de contenido que aplica RF con una precisión de 97.02%. Esto puede deberse a la cantidad de correos electrónicos examinados en el modelo. Por un lado, el modelo de Egozi y Verma utiliza 8899 correos electrónicos; mientras que la cantidad de correos electrónicos en el modelo con el que se hace contraste en esta investigación es de 3631 correos electrónicos.

Las investigaciones actuales relacionadas al tema en las cuales el promedio de los resultados de predicción es 93.12% y los resultados obtenidos nos lleva a afirmar que, si bien se obtuvieron buenos resultados, el modelo basado en análisis de contenido que aplica técnicas PLN y Support Vector Machine debe estar en constante actualización, ya que cada vez es más común la aparición de nuevas técnicas de phishing. Asimismo, hay que recalcar que este modelo aún requiere integrarse a otras tecnologías, como aplicativos web o móviles, para la utilización y aprovechamiento de los usuarios.

7. CONCLUSIONES

En esta investigación se desarrollaron dos modelos de detección de phishing basados en análisis de contenido de correos electrónicos utilizando procesamiento de lenguaje natural y análisis de urls ofuscados para así validar cual presenta mejores resultados. De los resultados analizados podemos concluir que ambos modelos presentaron resultados con una diferencia mínima. Por un lado, el modelo que analiza solo el contenido del mensaje obtuvo una precisión de 97.02% y 95.32% para los algoritmos de RF y SVM, respectivamente. Mientras que, el modelo que analizaba las URLs ofuscadas obtuvo precisiones de 86.81% y 98.70% en dichos algoritmos. Además, tomando en cuenta la métrica de exactitud podemos concluir que el modelo que analiza URLs ofuscadas resulta más efectivo ya que ha tenido mejor número de aciertos.

En cuanto a la utilización de los algoritmos, en base a nuestros resultados podemos concluir que el uso del algoritmo de RF para el modelo de análisis de contenido no resulta ser efectivo ya que se puede observar que en la métrica de sensibilidad existe un menor porcentaje, lo cual estaría causando que correos electrónicos que son relevantes para un usuario sean clasificados como phishing de manera errónea. Sin embargo, en cuanto a precisión de detección de phishing los dos algoritmos han logrado obtener puntajes altos lo que nos permite concluir que si se está teniendo una detección efectiva de correos de phishing.

Es importante, mantener actualizados los modelos de detección de phishing; ya que, la tecnología avanza para todos, incluyendo a los ciberdelincuentes. Para trabajos futuros, se pueden adaptar algunos mecanismos de detección de phishing para analizar partes insertadas en los correos que aún no se han considerado como: códigos QR, imágenes, entre otros. Asimismo, que los métodos sean integrados en herramientas que sean amigables para la utilización directa del usuario.

Si bien este análisis es hecho en un entorno simulado y controlado, el funcionamiento de los modelos en un ambiente real no debería tener complicaciones, ya que los correos electrónicos manejan el mismo formato y contienen el cuerpo, lo cual permite la extracción de los parámetros ya definidos. Sin embargo, se debe tener en cuenta que existen empresas que utilizan sus propios servidores de correo electrónico y se pueden tener atributos adicionales o distintos a los evaluados en esta investigación. Esto último, podría implicar realizar modificaciones en el código fuente para aplicar los modelos desarrollados.

REFERENCIAS

Anti-phishing Working Group.(2022). *Phishing Activity Trends Report*. Phishing Activity Trends Report, 3rd Quarter 2022

Buber, E., Diri, B., & Sahingoz, O. K. (2017). NLP based phishing attack detection from URLs. En *International Conference on Intelligent Systems Design and Applications* (pp. 608-618). Springer, Cham.

Egozi, G., & Verma, R. (2018). Phishing email detection using robust nlp techniques. En *2018 IEEE International Conference on Data Mining Workshops (ICDMW)* (pp. 7-12). IEEE.

Han, Y., & Shen, Y. (2016). Accurate spear phishing campaign attribution and early detection. *Proceedings of the ACM Symposium on Applied Computing*, *04-08-April-2016*, 2079–2086. https://doi.org/10.1145/2851613.2851801

Hu, H., & Wang, G. (2018). End-to-end measurements of email spoofing attacks. En 27th {USENIX} Security Symposium ({USENIX} Security 18) (pp. 1095-1112).

IBM (2020). Índice de información de amenazas X-Force. Recuperado de: https://www.ibm.com/downloads/cas/PR0ODMQM

Jain, A. K., & Gupta, B. B. (2018). *PHISH-SAFE: URL Features-Based Phishing Detection System Using Machine Learning. Cyber Security, 467–474*

Jalilifard, A., Caridá, V. F., Mansano, A. F., Cristo, R. S., & da Fonseca, F. P. C. (2021). Semantic sensitive TF-IDF to determine word relevance in documents. En *Advances in Computing and Network Communications* (pp. 327-337). Springer, Singapore.

Martín Guareño, J. J. (2016). Support vector regression: propiedades y aplicaciones.

Mantovani, R. G., Horváth, T., Cerri, R., Junior, S. B., Vanschoren, J., & de Carvalho, A. C. P. D. L. F. (2018). An empirical study on hyperparameter tuning of decision trees. *arXiv preprint arXiv:1812.02207*.

Moradpoor, N., Clavie, B., & Buchanan, B. (2017). Employing machine learning techniques for detection and classification of phishing emails. En 2017 Computing Conference (pp. 149-156). IEEE.

Odunibosi, O. (2019). Classification of email headers using random forest algorithm to detect email spoofing. *MSc Academic Internship National College of Ireland Supervisor* .

Peng, T., Harris, I., & Sawa, Y. (2018). Detecting phishing attacks using natural language processing and machine learning. En 2018 *ieee 12th international conference on semantic computing (icsc)* (pp. 300-301). IEEE.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D.,

Brucher, M., Perrot, M. & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine Learning research*, *12*, 2825-2830.

Rathod, J., & Nandy, D. (2014). Anti-phishing technique to detect URL obfuscation. *Int. J. Eng. Res. Appl.*, *4*(5), 172-179.

Rawal, S., Rawal, B., Shaheen, A., & Malik, S. (2017). Phishing Detection in E-mails using Machine Learning. International Journal of Applied Information Systems, 12(7), 21-24.

Razno, M. (2019). Machine learning text classification model with NLP approach. *Computational Linguistics and Intelligent Systems*, *2*, 71-73.

Sentürk, S., Yerli, E., & Sogukpınar, İ. (2017). Email phishing detection and prevention by using data mining techniques. In 2017 International Conference on Computer Science and Engineering (UBMK) (pp. 707-712). IEEE.

Smadi, S., Aslam, N., & Zhang, L. (2018). Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Decision Support Systems, 107*, 88–102. https://doi.org/10.1016/j.dss.2018.01.001

Steve Sheng, Brad Wardman, Gary Warner, Lorrie Faith Cranor, Jason Hong, and Chengshan Zhang. 2009. An empirical analysis of phishing blacklists. In Proceedings of Sixth Conference on Email and Anti-Spam (CEAS).

Talamé, L., Cardoso, A., & Amor, M. (2019). Comparación de herramientas de procesamiento de textos en español extraídos de una red social para Python. In *XX Simposio Argentino de Inteligencia Artificial (ASAI 2019)-JAIIO 48 (Salta)*.

Vapnik,V. & Cortes,C.(1995). Support-vector networks. Machine learning, 20(3).

Verma, R., Shashidhar, N., & Hossain, N. (2012). Detecting phishing emails the natural language way. In *European Symposium on Research in Computer Security* (pp. 824-841). Springer, Berlin, Heidelberg.

Verma, R., & Rai, N. (2015). Phish-idetector: Message-id based automatic phishing detection. En *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)* (Vol. 4, pp. 427-434). IEEE.

Volkamer, M., Renaud, K., Reinheimer, B., & Kunz, A. (2017). User experiences of torpedo: Tooltip-powered phishing email detection. *Computers & Security, 71*, 100-113.

DETECCIÓN DE PHISHING EN CORREOS ELECTRÓNICOS MEDIANTE PROCESAMIENTO DE LENGUAJE NATURAL DEL CONTENIDO y URLS OFUSCADAS

INFORME DE ORIGINALIDAD

9%	9%	1%	%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	hdl.handle.net Fuente de Internet	2%
2	repositorio.espe.edu.ec Fuente de Internet	1%
3	sedici.unlp.edu.ar Fuente de Internet	<1%
4	www.dropbox.com Fuente de Internet	<1%
5	ddd.uab.cat Fuente de Internet	<1%
6	Rafael MARFIL-CARMONA, Jesús MONTOYA-HERRERA, Pedro CHACÓN-GORDILLO. "LA CIUDAD COMO PERSONAJE. ESTUDIO DE CASO DEL CÓMIC “LAS CALLES DE ARENA”, DE PACO ROCA, DESDE EL PUNTO DE VISTA DE LA EDUCACIÓN ARTÍSTICA", 'Asociacion Cientifica ICONO14', 2018 Fuente de Internet	<1%