ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Detección en tiempo real de phishing por email mediante técnicas de procesamiento de lenguaje natural y algoritmos de clasificación para una empresa corporativa.

PROYECTO DE TITULACIÓN

Previo la obtención del Título de:

Magister en Ciencias de Datos

Presentado por:
Carlos Patricio Samaniego Palacios
Eduardo Javier Yepez Montenegro

GUAYAQUIL - ECUADOR
Año: 2022

DEDICATORIA

A mis padres.

Eduardo Yepez

A Dios y a mi familia.

Patricio Samaniego

AGRADECIMIENTOS

A mis padres, profesores, colegas y organizaciones que me ayudaron en el desarrollo de este proyecto.

Eduardo Yepez

A Dios, mi familia, mi equipo de trabajo, la empresa donde laboro, profesores la MCD2, ESPOL.

Patricio Samaniego

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, me(nos) corresponde conforme al reglamento de propiedad intelectual de la institución; *(nombre de los participantes)* y doy(damos) mi(nuestro) consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Eduardo Yepez

Patricio Samaniego

COMITÉ EVALUADOR

MSc. Eduardo Cruz	**MSc. Allan Avendaño**
PROFESOR TUTOR	PROFESOR EVALUADOR

RESUMEN

La detección de phishing en tiempo real se ha convertido en una necesidad para empresas interesadas en minimizar los riesgos asociados a la pérdida o divulgación de información confidencial valiosa. En esta tesis se propone una solución para la detección de phishing en tiempo real en un entorno corporativo, usando un conjunto de datos de correos electrónicos en español como insumo para el entrenamiento de modelos de machine learning que son utilizados en tareas de procesamiento de lenguaje natural y clasificación. Además, se establecen métricas para evaluar la efectividad de los modelos y se desarrolla un pipeline para la inferencia en tiempo real de correos electrónicos que son phishing. El desarrollo de este trabajo se lo realizó con la infraestructura de Nvidia y el patrocinio de una empresa líder en telecomunicaciones del Ecuador. Se usaron modelos de NLP tipo BERT como BERTIN, roBERTa y Distil BERT para la clasificación de correos en español. Finalmente, se utilizó el framework Nvidia Morpheus para desplegar el modelo de machine learning en producción. El estudio concluye con la creación de un conjunto de datos de correos electrónicos de phishing en español con características léxicas comúnmente usadas por los atacantes. Se obtuvo un modelo con un F-beta score de 0.972 para clasificar correos tipo phishing. Se espera que, al implementar la solución, la compañía reduzca en un 95.83% su carga operacional al disminuir hasta 18 turnos operativos de triage de incidentes con una eficiencia de detección del 90.7%.

Palabras clave: Detección de phishing, NLP, Inferencia en tiempo real, Deep Learning, Machine Learning.

ABSTRACT

Real-time phishing detection has become a necessity for companies committed to minimizing the risks associated with the loss or disclosure of valuable confidential information. In this thesis, a solution is proposed for real-time phishing detection in a corporative environment, using a dataset of Spanish emails as an input for training machine learning models for natural language processing and classification tasks. In addition, metrics are established to evaluate the effectiveness of the models and a pipeline is developed for real-time inference of phishing emails. This research was developed with Nvidia infrastructure and the sponsorship of a leading telecommunications company in Ecuador. BERT-type NLP models such as BERTIN, roBERTa and Distil BERT were used for classification of emails in Spanish. Finally, the Nvidia Morpheus framework was adopted to bring the machine learning model to production. The study concludes with the creation of a dataset with phishing emails in Spanish with a lexical structure commonly used by attackers. A model with an F-beta score of 0.972 was obtained to classify phishing emails. The solution adopted by the company is expected to decrease operational costs by 95.83% with the reduction of up to 18 work shifts in incident triage with a detection efficiency of 90.7%.

Keywords: *Phishing detection, NLP, Real-time inference, Deep Learning, Machine Learning.*

ÍNDICE GENERAL

ABREVIATURAS

BEC	Business Email Compromise
BERT	Bidirectional Encoder Representations from Transformers
CRISP-DM	Cross-industry standard process for data mining
DKIM	Domain Keys Identified Mail
DMARK	Domain-based Message Authentication, Reporting & Conformance
ELK	Elasticsearch, Logstash, and Kibana
ESPOL	Escuela Superior Politécnica del Litoral
ESG	Email Security Gateway
EMAIL	Electronic Mail
ETL	Extract, Transform and Load
IMAP	Internet Messaging Access Protocol
ML	Machine Leaning
NLP	Natural language processing
SIEM	Security Information and Event Management
SMTP	Simple Mail Transfer Protocol
SPF	Sender Policy Framework
SVM	Support Vector Machine
TF-IDF	Term Frecuency-Inverse Document Frecuency
URL	Uniform Resource Locators

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

1.1 Descripción del problema

Los ataques cibernéticos han acompañado al internet por un largo tiempo, así como han evolucionado junto con este. En el año 2019, la empresa Paolo Alto Networks, un reconocido proveedor de ciberseguridad a nivel mundial publicó resultados de un análisis del crecimiento de ataques cibernéticos. En el estudio se muestra una evolución tanto en cantidad como en complejidad de las amenazas, haciendo que la atención de esta problemática sea de interés común para la ciencia y la industria [1].

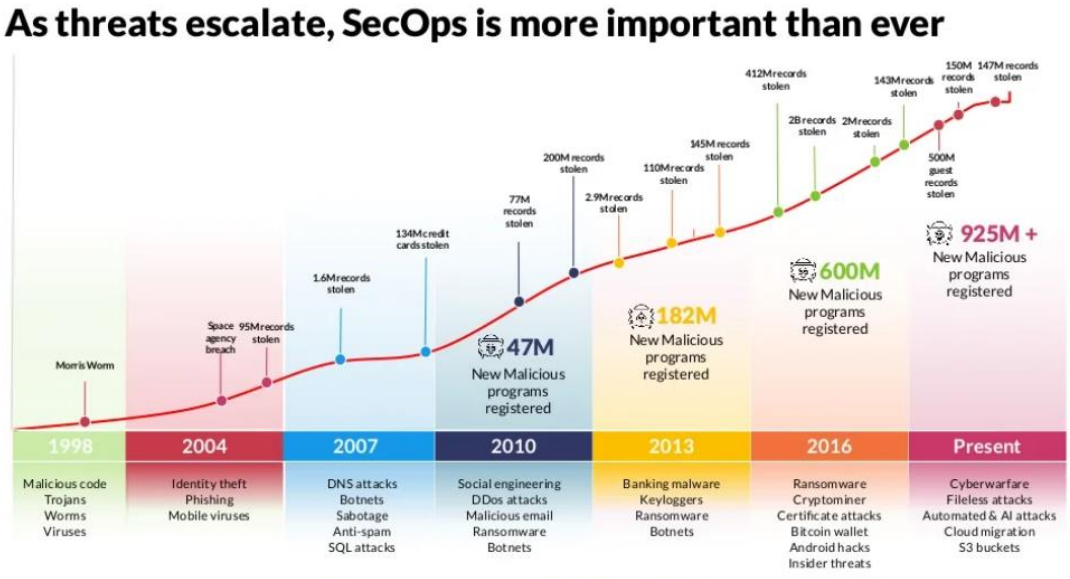


Figura 1.1 Ecosistema de ciberamenazas acorde a Palo Alto.

En la Figura 1.1 podemos apreciar una marcada tendencia al alza en la cantidad de programas maliciosos que se han registrados en el ecosistema de ciberamenazas, así como podemos notar que para el 2019 (año del reporte), Palo Alto Networks registra amenazas automatizadas y el uso de herramientas de inteligencia artificial para realizar ataques informáticos.

Acorde a la empresa Cloudflare, un proveedor a nivel mundial de infraestructura web y seguridad en la nube, el 95% de la causa raíz de los incidentes de ciberseguridad son ocasionados por ataques de phishing [2].

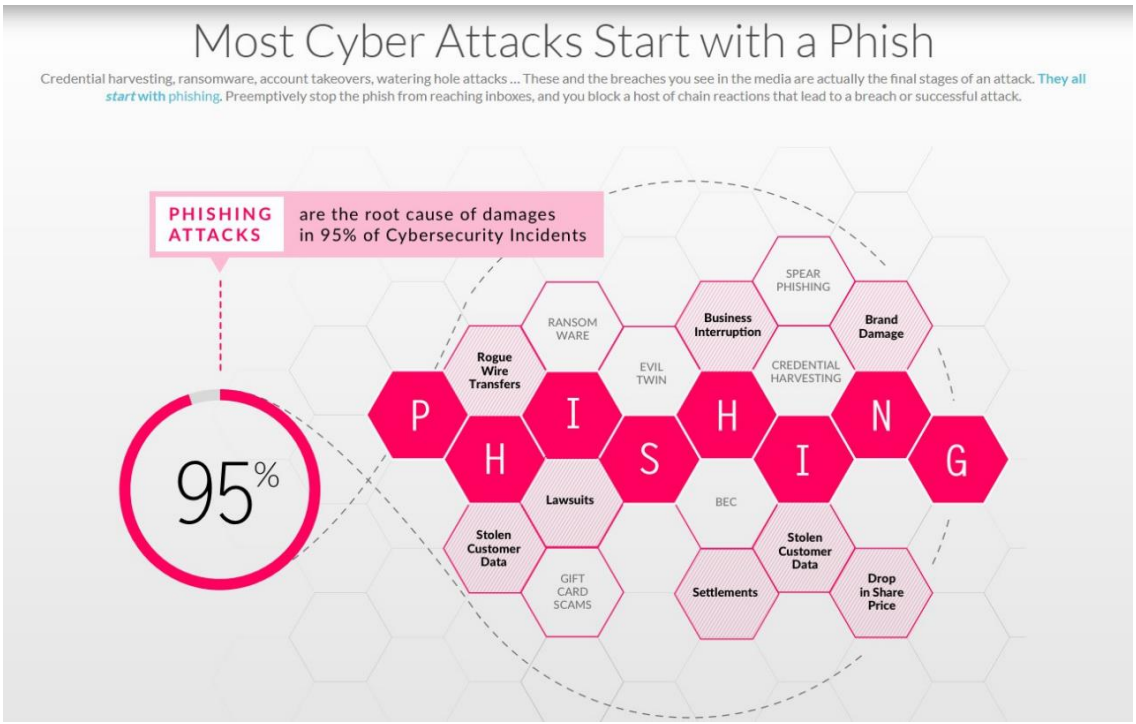


Figura 1.2 Causa raíz de incidentes según Cloudflare.

Acorde a la Figura 1.2, el phishing es uno de los principales medios para elaborar ataques informáticos exitosos, este ataque se aprovecha de la vulnerabilidad humana, donde las emociones como: el temor, la urgencia, la ambición, pueden ser usadas por los atacantes para manipular a su víctima y así ejecutar acciones necesarias para comprometer sus activos digitales, por ejemplo: dando clic a un enlace, abrir un archivo malicioso, lo cual posteriormente se convertirá en credenciales o computadores comprometidos.

Siendo el correo electrónico (Email) una de las herramientas corporativas más usadas para el intercambio de información de negocio, es predecible saber que los atacantes usarán este medio para intentar vulnerar a sus víctimas, convirtiendo esto en una amenaza muy importante de atender para las empresas por su implicancia

en operaciones y economía; de hecho, según el FBI para el año 2021 se reportó una pérdida por BEC (Busines Email Compromise/Compromiso de cuentas corporativas) en el mercado americano por alrededor de los $43 Billones de dólares convirtiendo esta amenaza en algo muy lucrativo para los ciberdelincuentes [3].

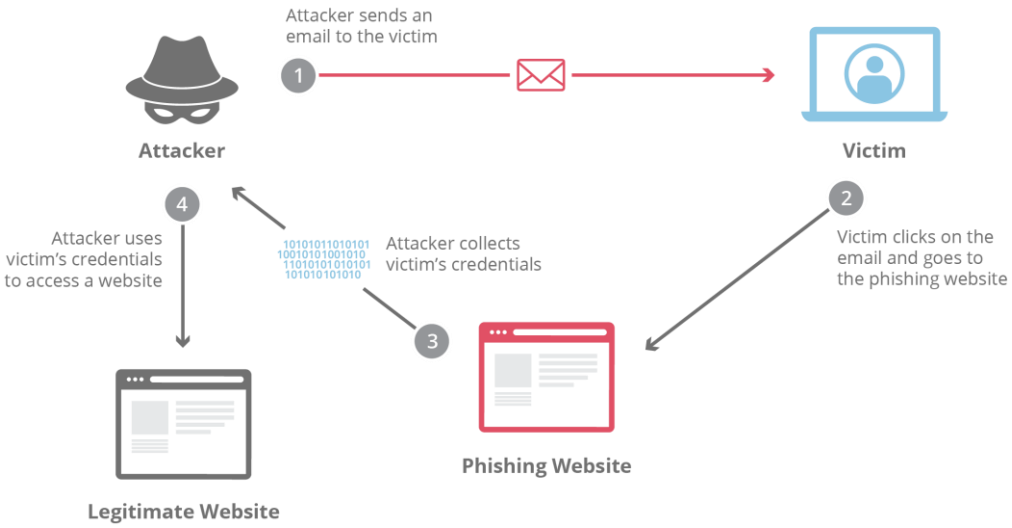


Figura 1.3 Phishing por redireccionamiento vía email.

En la Figura 1.3 podemos ver el uso de phishing por email para el robo de credenciales mediante el acceso a un sitio malicioso, 1) el atacante envía un email motivando a la víctima mediante técnicas de engaño a entrar a un enlace controlado por el atacante, 2) el usuario víctima accede al link y coloca las credenciales de acceso, 3) el atacante colecta las credenciales de la víctima, 4) el atacante usa las credenciales para acceder al sitio original con las credenciales de la víctima. Entre los vectores de mayor éxito para el compromiso de cuentas de usuario está el phishing por email, acorde a Mimecast, empresa especializa en la administración de correo electrónico basada en la nube. Phishing y Ransomware son los principales riesgos que enfrenta un negocio cuyo vector de ataque principal es el correo electrónico, en su informe nos dice que virtualmente toda compañía encuestada reportó haber sufrido un ataque de phishing, donde una amplia mayoría reportó que este vector está incrementando [4].

Fuera de lo cotidiano, los cibercriminales aprovechan situaciones de alto interés social para sus campañas de engaño, por ejemplo, en Ecuador, el Gobierno Electrónico reportó aumento de campañas de phishing con intensión de abusar de la información relacionada con el COVID, "Los Ciberdelincuentes utilizan el coronavirus como excusa para infectar a los usuarios. Se detectaron varias campañas de “phishing” a nivel mundial, en las que los cibercriminales usando este contexto buscan comprometer el equipo de sus víctimas o robar los datos personales." [5].

Cuando una empresa gestiona sus riesgos de ciberseguridad debe priorizar; según la fórmula universal del riesgo **R = I x P,** donde R = Riesgo, I = Impacto y P = Probabilidad de ocurrencia, el riesgo es directamente proporcional a aumento de estas 2 variables (Impacto y Probabilidad), y el phishing es considerado de alto impacto, así como muy probable que ocurra, por lo tanto, se vuelve un riesgo prioritario que motiva a la industria de ciberseguridad a mantener una continua investigación de cómo mitigar este tipo de ataque informático.

Debido a las continuas y crecientes amenazas por email, entre ellas el phishing, es casi un estándar contar con un sistema de defensa enfocado en detener emails maliciosos complementario a sus capas adicionales de seguridad como Firewalls, Antivirus, etc. Estos sistemas se los conoce como ESG (Email Security Gateway) o Antispam. Estas herramientas cubren una superficie de ataque por email necesaria, pero no suficiente; de hecho, su nombre Antispam viene debido a su función natural histórica que fue la de detener Emails de SPAM o correo no deseado, por ejemplo: correos de publicidad o informativos no esperado, así como correos maliciosos enviados por herramientas automatizadas de atacantes sin mayor cuidado de detalles en fallos de protocolos o buenas prácticas que fácilmente pueden ser detectados u contenidos por un ESG.

Dado que se ha vuelto muy lucrativo y efectivo para los ciberatacantes comprometer cuentas o dispositivos de usuarios por email, estas amenazas han ido evolucionando al punto que los ESGs no han podido detener las nuevas técnicas usadas, dejando

un riesgo latente que está siendo aprovechado tal y como lo vimos en las estadísticas previas. Este riesgo por muchos años es algo con lo que las organizaciones han tenido que sobrellevar, tomando medidas compensatorias para atenuar la posibilidad de ser víctimas de estos ataques y minimizar su impacto. Siendo un problema de muy alto riesgo para una organización, la ciencia y la industria continúan desarrollando investigación y mecanismos de defensa contra técnicas avanzadas de email de phishing, las cuales se caracterizan en cumplir con las buenas prácticas de uso de email que es lo primero en detener los ESG, por ejemplo: baja tasa de envío, envío desde remitentes confiables, uso de servidores no comprometidos, uso de direcciones IPs de buena reputación. En general, el uso de infraestructura sin mala reputación o comportamiento anormal, dejando la esencia del engaño en el cuerpo del correo electrónico (Body), el cual no es analizado por los ESG tradicionales para detectar si es definitivamente un correo electrónico es malicioso o no; es aquí donde nuestra solución busca abordar este problema y agregar valor.

Debido el contexto actual del conjunto de herramientas de defensa con las que cuenta una empresa, si esta quisiera cubrir el riesgo de detectar emails de phishing que han pasado su sistema de defensa y llegado a sus usuarios, tendría que revisar el 100% de los emails categorizados mediante el ESG como benignos, lo cual resultaría en una inversión altísima en recursos y posiblemente ineficiente, puesto que un operador bien entrenado debería revisarlos manualmente y su capacidad operativa sería limitada, sin contar la posible afectación a la confidencialidad o secreto empresarial, dado que este operador debería acceder al cuerpo de los correos entrantes donde pudiese haber información que no debe ser leída por una persona que no era el destinatario original. Aquí es donde la ciencia ha tenido avances mediante el uso de técnicas de Ciencia de Datos como lo es el NLP (procesamiento natural del lenguaje), donde se han desarrollado mecanismos que permiten que una máquina "comprenda" el lenguaje humano, pudiendo así ser entrenada y determinar si un texto contiene características deseadas, o en nuestro caso, características *no deseadas* asociadas a un phishing o engaño y que puedan ser detectadas por una herramienta con capacidad de inteligencia artificial.

1.2 Justificación del problema

Acorde a la empresa global Cisco Sytems, un referente en la industria de las redes, telecomunicaciones y seguridad, el problema de detección de phishing se lo aborda en función de capas, donde se utilizan técnicas de manera progresiva que van de menor a mayor uso de recursos así con el objetivo de ser eficientes en la detección de emails maliciosos.

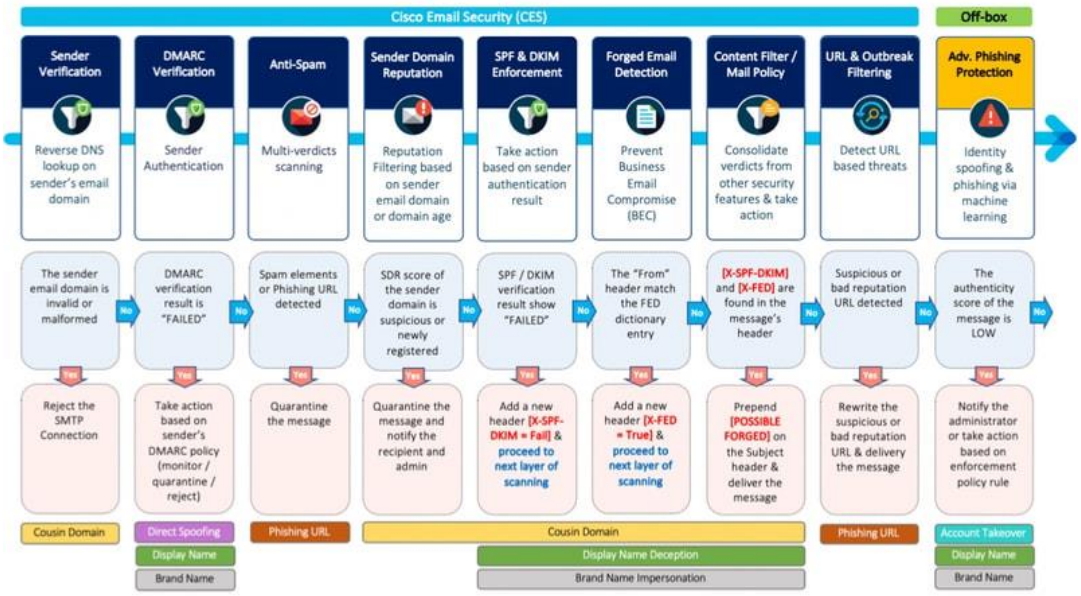


Figura 1.4 Detección de phishing por capas (CISCO).

En la Figura 1.4 se muestra el modelo de defensa por capas presentado por la empresa Cisco Systems [6], donde aprecia el uso de diferentes técnicas para la detección de emails maliciosos, comenzando con medidas asociadas a buenas prácticas de DNS, pasando por controles de reputación y detección de URLs maliciosas (inteligencia de amenazas), dejando para el final técnicas de Machine Learning, esto debido al esfuerzo en recursos que esta representa.

Según la firma consultora Osterman Research en su White Paper "How to Reduce the Risk of Phishing and Ransomware" publicado en el 2021, presenta las herramientas de seguridad que mejoran su rendimiento contra emails de phishing usando algoritmos de Machine Learning [7].

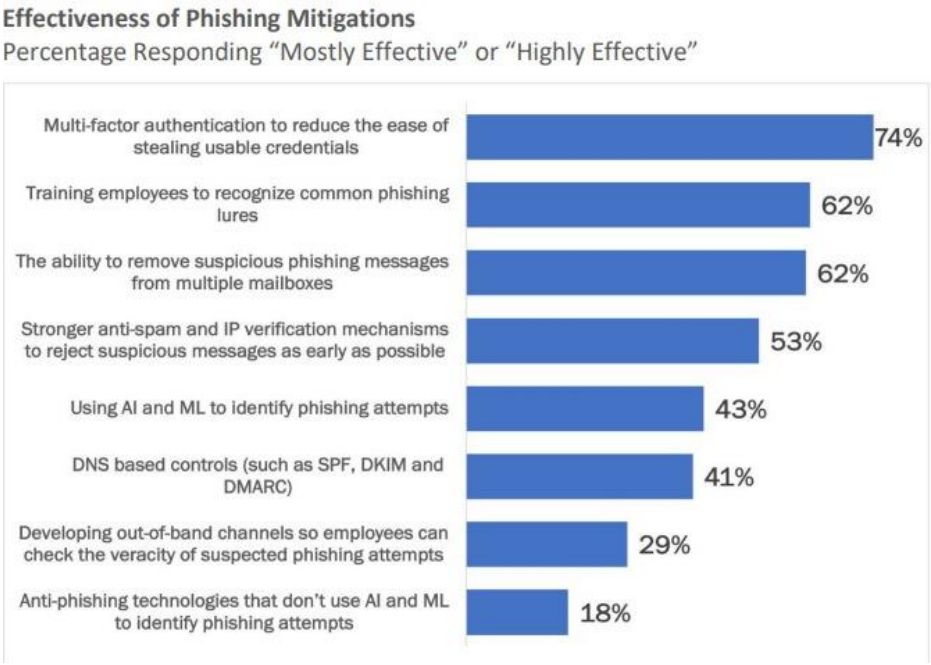


Figura 1.5 Efectividad de herramientas para mitigar phishing (Osterman).

En la Figura 1.5 podemos observar que una herramienta de detección de phishing que utiliza algoritmos de machine learning (43%) puede superar hasta en un 25% a una herramienta que no lo usa (18%).

La técnica de detección de phishing mediante machine learning también es vista por la empresa GreatHorn, este proveedor de ciberseguridad menciona en que una solución integral de seguridad por email necesita tener defensas adicionales a las ofrecidas con los Antispam tradicionales incluyendo técnicas de machine learning [8]. Incluso proveedores especializados en inteligencia artificial ven la oportunidad de proveer soluciones para la detección amenazas de phishing, como lo es el caso de NVIDIA, proveedor líder de infraestructura de procesamiento gráfico e inteligencia artificial, que dispone de una línea de investigación para detección de amenazas de ciberseguridad basadas en el uso de inteligencia artificial, donde específicamente un caso de uso está destinado a la detección de phishing basado en NLP, a este proyecto se lo conoce como NVIDIA Morpheus [9], donde si bien NVIDIA nos ofrece una arquitectura para inferencia en tiempo real y un modelo pre entrenado para la detección de phishing, este está trabajado en base al idioma inglés, el cual no es

eficiente en nuestro ecosistema habla hispana y para que sea efectivo deberá ser ajustado y en este proyecto abordaremos esta problemática.

En este proyecto, a fin de ofrecer una solución para la detección de phishing mediante el uso de técnicas de inteligencia artificial, contamos con el apoyo de una empresa de telecomunicaciones de vanguardia que brinda diferentes servicios, entre ellos internet, datacenter, ciberseguridad y soluciones basadas en inteligencia artificial, quienes conscientes de la importancia de atender riesgos asociados a phishing, han permitido desarrollar una investigación y prototipo de cómo detectar pishing usando modelos de machine learning. Dado que se cuenta con infraestructura de NVIDIA, se pude implementar una solución de detección de phishing en tiempo real basado en técnicas de inteligencia artificial y la solución de NVIDIA Morpheus. Actualmente, si esta empresa quisiera atender el riesgo de Emails de phishing que no han sido detectados por su EGS, deberían analizar manualmente hasta unos 750 Emails/hora, y dado que esto representa a sus analistas una revisión entre 5 minutos a 15 minutos por cada 30 Emails maliciosos (dato experimental facilitado por la empresa), tendrían que invertir de 6 a 19 personas de jornada completa dedicada (8 horas de trabajo) aproximadamente, resultando costoso e ineficiente resolverlo.

1.3 Solución propuesta

En el presente proyecto partiremos del conocimiento existente de detección de phishing usando NLP en la arquitectura de NVIDIA Morpheus [9], ajustando esta arquitectura/modelo para abordar la problemática de detección de phishing en español; para ello, primero estudiaremos los conjuntos de datos con los que cuenta la empresa corporativa y los que existen publicados en la web, construiremos un Dataset de emails de phishing en español que podamos usar para este fin; enfrentaremos un problema de datos desbalanceados debido a la naturaleza del problema (una gran cantidad de Emails benignos vs malignos) y los abordaremos usando técnicas de aumento de datos para resolverlo. Luego de esto, evaluaremos diferentes modelos de Deep learning de NLP para encontrar el de mejor rendimiento y finalmente, usaremos el modelo seleccionado en el pipeline de

inferencia en tiempo real, el cual será integrado a la herramienta de colección de eventos de ciberseguridad usada actualmente por la empresa.

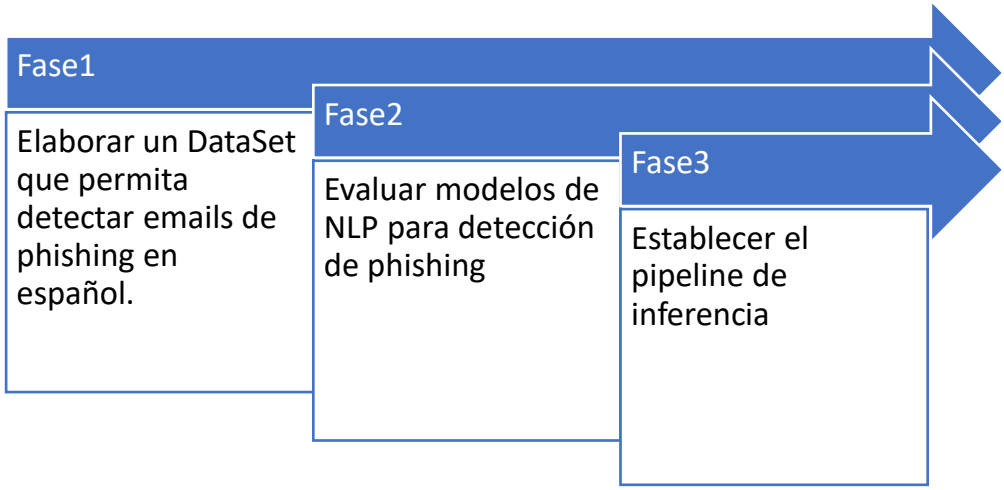


Figura 1.6 Estrategia de detección de phishing basado en NLP

En la Figura 1.6 podemos observar la estrategia a seguir para elaborar la solución de detección de phishing basado en NLP de la presente investigación, esta estrategia estará alineada a los objetivos del presente proyecto.

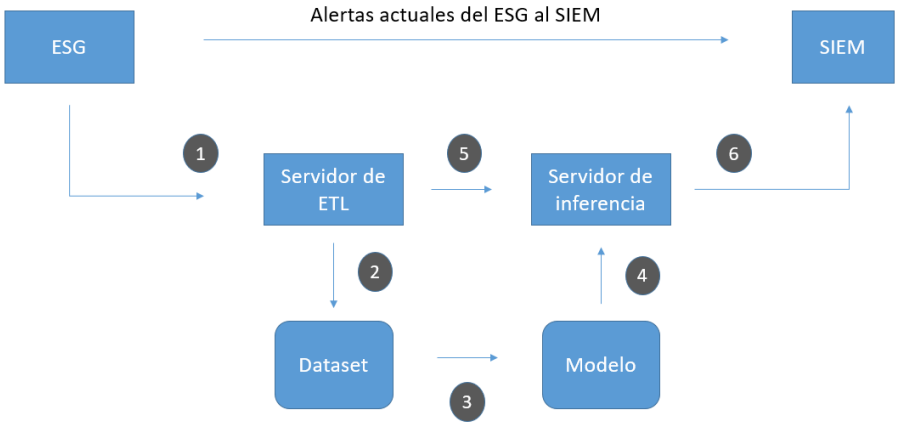


Figura 1.7 Diagrama de arquitectura de operación de la solución.

En la Figura 1.7 podemos ver como la solución planteada se integra al ecosistema de operaciones de ciberseguridad actual de la empresa (Alertas actuales del ESG

al SIEM), donde mediante el servidor de ETL (extract, transform y load) obtendremos la data del ESG que servirá para construir el DataSet que será usado para elaborar el modelo de detección en tiempo real (1 y 2), luego mediante un estudio se seleccionará el mejor modelo que pueda ser usado por el servidor de inferencia (3 y 4), en este punto, cabe recalcar que por requisitos de la empresa y condiciones técnicas usaremos el mejor modelo soportado por la solución de NVIDIA Morpheus. Con el modelo seleccionado, usaremos nuevamente el servidor de ETL para colectar en tiempo real los emails del ESG y enviarlos al servidor de inferencia en tiempo real cuyos resultados se depositarán en una herramienta de análisis de eventos de seguridad SIEM (5 ,6 y 7).

1.4 Objetivos

1.4.1 Objetivo General

Diseñar e implementar una solución que permita la detección de phishing por email en tiempo real mediante el uso de técnicas de machine learning permitiendo atenuar riesgos de ciberseguridad en un entorno corporativo.

1.4.2 Objetivos Específicos

1. Generar un dataset que permita el análisis y detección de correos maliciosos tipo phishing mediante algoritmos de machine learning.
2. Evaluar modelos de machine learning que permitan caracterizar y discriminar si un email es fraudulento.
3. Elaborar un pipeline para procesamiento e inferencia en tiempo real de correos electrónicos.

1.5 Metodología

En esta tesis se va a utilizar la metodología CRISP-DM (Cross-industry standard process for data mining) para la construcción del proyecto de ciencia de datos cuyo esquema se puede visualizar en la Figura 1.8. A continuación, describimos la

planificación de las etapas del proyecto junto con una breve descripción de las técnicas a usar relacionadas con el programa de ciencia de datos.

1. Entendimiento del negocio: Se define claramente la problemática de la empresa, se describe nuestro enfoque para mitigar el problema y se establecen los objetivos de negocio y técnicos junto con sus respectivas definiciones de cumplimiento. Finalmente se identifican los requisitos para el proyecto y la disponibilidad de datos.

   Este componente del proyecto se encuentra en el capítulo 1, se resalta la importancia de los conocimientos adquiridos en la materia de soluciones basados en datos para el desarrollo de esta etapa.
2. Entendimiento de los datos: Se recolecta los datos de emails de la compañía, así como los registros de estos correos de los equipos de seguridad, se incluye también el etiquetado de correos (phishing o no phishing) del departamento de ciberseguridad de la empresa. Se examina la naturaleza de estos datos no estructurados con el fin de generar correos tipo Phishing. Se identifican los campos, valores esperados y número de registros disponibles. Documentamos aspectos importantes respecto a la calidad de estos datos.
3. Preparación de los datos: Se realiza la limpieza de datos usando nuestras definiciones de calidad de datos. Se realiza la creación de los conjuntos de datos para entrenamiento y validación. Se emplean técnicas de aumento de datos para balanceo de clases. Finalmente, establecemos los formatos y normalizamos nuestros datos creando un conjunto de datos final listo para modelamiento. Al completar esta etapa pretendemos alcanzar el primer objetivo específico.
4. Modelamiento: Seleccionamos los modelos a utilizar. Buscamos explotar la parte del cuerpo del correo con técnicas de procesamiento de lenguaje. Para esto vamos a construir varios modelos y aumentar la complejidad de los modelos conforme sea necesario. Esperamos que un modelo de NLP en el estado del arte nos de los mejores resultados. Además, se incluye el componente de mejora mediante la búsqueda de hiperparámetros.

5. Evaluación: Definimos las métricas de evaluación de los modelos y comparamos el rendimiento de los modelos con nuestra definición de éxito en los resultados esperados.

   Las etapas 4 y 5 del proyecto se encuentran detalladas en el capítulo 3. Aquí vamos a aplicar lo aprendido en los cursos de machine learning y deep learning. Al finalizar esta etapa esperamos cumplir con el segundo objetivo específico.

6. Despliegue: Se va a diseñar un pipeline para el despliegue del modelo. La arquitectura se debe ajustar a los requerimientos del negocio y a los recursos disponibles. Se espera demostrar el funcionamiento de nuestra solución con procesamiento de correos en tiempo real.

   En el capítulo 3.5 se va a describir nuestro plan de despliegue. Aquí vamos a desarrollar un esquema basado en los talleres de la materia big data y computación en la nube.

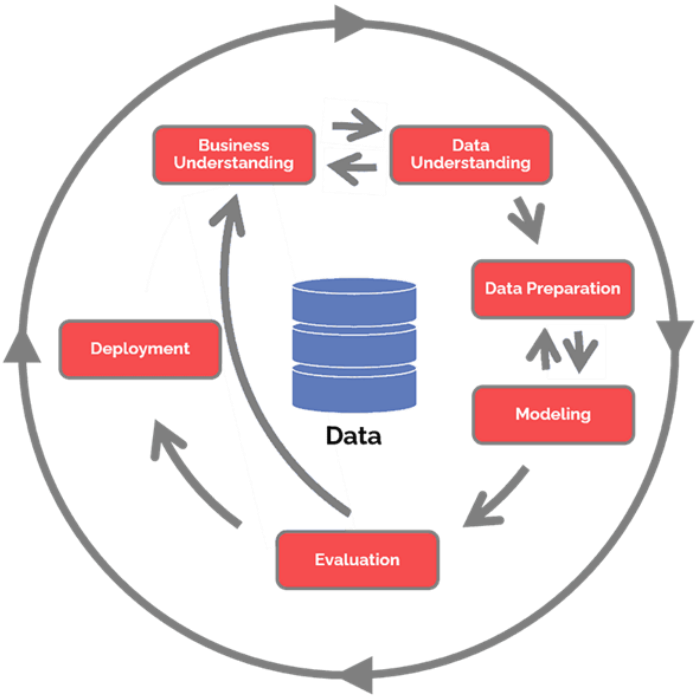


Figura 1.8 Esquema de metodología CRISP-DM

En la gráfica Figura 1.8 podemos ver la representación gráfica de la metodología CRISP-DM, la cual usamos de referencia para la construcción del proyecto de ciencia de datos para resolver la problemática planteada.

1.6 Resultados esperados

El presente trabajo genera un prototipo funcional de una herramienta de clasificación de emails que son phishing en tiempo real, basada en un modelo optimizado que pueda procesar un flujo de correos electrónicos e indique la existencia de un email cuyas características puedan ser consideradas como intento de phishing (engaño). Se espera que esta herramienta trabaje independiente del ESG (Email Security Gateway) o Antispam que actualmente usa la empresa corporativa, y sea una herramienta que complemente y mejore los niveles de detección de esta amenaza.

CAPÍTULO 2

2. ESTADO DEL ARTE

2.1 Soluciones actuales para la detección de phishing

2.1.1 Técnicas de defensa contra ataques de phishing

Enfocándonos específicamente en las técnicas que usan las soluciones de ESG como medidas de seguridad perimetral para defensa contra ataques por email, a alto nivel podemos clasificar 3 grandes grupos: el primero sería basado en reglas, donde aquí podemos controlar todo lo que es el estándar SMTP y demás protocolos asociados, por ejemplo, SPF, DKIM, DMARK, las segundas serían basadas en ciberinteligencia o inteligencia de amenazas que cubre la detección de amenazas conocidas, sea esto por reputación o análisis complementario como un sandboxing para determinar si algún componente del email es una amenaza conocida que dispone de una firma o patrón de detección, y la tercera, la que está tomando fuerza últimamente son las basadas en Inteligencia Artificial, donde estas estrategias serían complementarias y abordarían piramidalmente la dificultad de la detección.

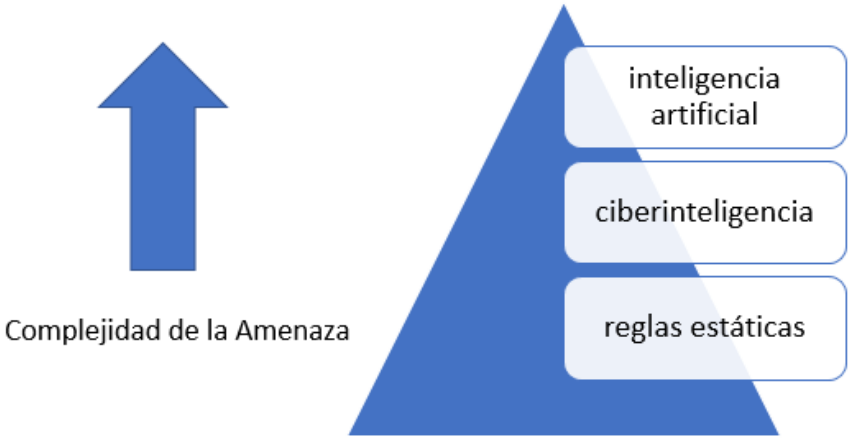


Figura 2.1 Técnicas de defensa contra phishing

En la Figura 2.1 se representa gráficamente como las técnicas de inteligencia artificial son usadas conforme la complejidad de la amenaza crece y por lo tanto requiere de una estrategia de optimización de recursos.

A continuación, describiremos algunas de las opciones que se contemplan en cada estas técnicas.

Tabla 2.1 Ejemplos de técnicas de detección de phishing

Reglas estáticas	**Ciberinteligencia**	**Inteligencia Artificial**
Cabeceras SMTP	Análisis de archivos	URL
Listas blancas/negras	Análisis de URL	BEC
Suplantación de dominio	Análisis de remitente	NLP

En la **¡Error! No se encuentra el origen de la referencia.** vemos algunos ejemplos de las técnicas más utilizadas para la detección de phishing con su respectivo tipo; a continuación, daremos una breve descripción del aporte y debilidad de cada una de estas:

Cabeceras SMTP: Establece las buenas prácticas desde la perspectiva de protocolo, se detectan comportamientos anormales como por ejemplo la

cantidad de identificadores saludos que realiza el servidor (EHLO), el cual en un servidor normal no debe ser mayor a 1, no obstante, las herramientas de ataque automatizadas podrían violar esta condición, asimismo a este nivel se pueden establecer reglas basadas en frecuencia de correos entrantes u otras características de la comunicación SMTP (protocolo usado para el envío de correos electrónicos) como la cantidad de sesiones establecidas por IP, la cantidad de receptores de correos únicos por servidor, estas son consideradas reglas estáticas.

Listas blancas/negras: Es un mecanismo que permite definir estáticamente reglas de negocio como políticas de control en el ESG, por ejemplo detener correos de fuentes que la empresa ya ha categorizado como maliciosa, asuntos de correos que la empresa no espera recibir, dominios los cuales por política la empresa no acepta correos electrónicos y demás, que permiten al ESG considerar un email como malo (Lista Negra) o en una lógica inversa definir lo que la empresa considera bueno (Lista Blanca). Si bien este control es muy práctico, se pueden cometer errores de confianza para atenuar problemas operativos. Por ejemplo, si ponemos como lista blanca a los correos enviados por Gmail o Hotmail, estos pasarán sin ser inspeccionados y un atacante puede crear emails de estos proveedores y aprovechar esta debilidad.

Suplantación de dominio: Existen diferentes mecanismos como SPF/DKIM/DMARK cuya intención es validar la autorización de un servidor para usar un dominio específico, permitiendo que un atacante no pueda usar un servidor de su autoridad o un servidor comprometido para enviar emails en nombre de otro. Estos controles tienen una gran fortaleza para evitar la suplantación de identidad, desafortunadamente aún para algunas organizaciones son considerados opcionales de implementación permitiendo que un atacante suplante una identidad conocida por la empresa y así generar confianza para su ataque.

Análisis de archivos: el objetivo es identificar si el archivo enviado es maligno, para esto se puede apoyar en una base de datos de amenazas conocidas, por ejemplo, mediante la comparación del hash del archivo, nombre; no obstante, estas características son basadas en conocimiento previo (base de amenazas). Una mejora a esta técnica es mediante análisis dinámico de archivos, también conocido como sandboxing donde en un ambiente virtual controlado se ejecuta el archivo y extrayendo sus características para así mediante estas determinar si es un archivo malicioso. Ambas técnicas ayudan para amenazas conocidas y es común que existan en los ESG, no obstante, por procesamiento, este mecanismo puede ser llevado para su procesamiento en la nube y con costos adicionales, esta es considerada una técnica necesaria, pero en un ambiente de defensa por capas no suficiente dado que un atacante puede modificar los archivos para saltar este control.

Análisis de URL: Similar al análisis de archivo, existen bases de datos de reputación de URLs donde se puede consultar si una URL (links dentro del cuerpo del correo) es maliciosa o no, existen las técnicas similares al análisis dinámico de archivos conocidas como URL sandboxing que identifica características maliciosas dentro de un sitio, pero los atacantes avanzados usan sitios con características muy similares a los benignos por lo que podrían evadir este control.

Análisis de remitente: en este control analizamos la base de inteligencia de amenazas basadas en recursos de red como IP o Dominio, por ejemplo, si la IP o dominio del remitente se encuentran identificados como generadoras de correos maliciosos, dado que han sido ya usados para atacar otras empresas, lo cual es común dado que un atacante optaría por reutilizar recursos por eficiencia mientras pueda. Algunas soluciones pueden también incorporar análisis estadístico básico, por ejemplo, tasas normales vs anormales de sesiones o correos de esta fuente, detectando así comportamiento anormal.

Inteligencia artificial - URL: Dado que el uso de mecanismos de detección de URLs maliciosas mediante análisis dinámico puede tomar mucho tiempo, existe investigación científica de cómo detectar URLs maliciosas basadas en técnicas de machine learninng, que si bien no han llegado a productos de seguridad podrían ayudar a esta problemática, no obstante, estas investigaciones son basadas en dataset de tipo SPAM y no phishing; si bien el estado del arte indica que existen buenos resultados para la detección de URLs maliciosas [10] no es común encontrar estas técnicas en los ESGs actuales.

Inteligencia artificial – BEC: El BEC o Business Email Compromise es un tipo de phishing enfocado en engaño a ejecutivos o personal clave mediante el engaño haciéndose pasar por parte de un círculo de confianza a fin de obtener principalmente dinero de la víctima mediante una trasferencia bancaria. Existen soluciones que mediante técnicas de machine learning ofrecen modelamiento de este tipo de amenazas para una organización soportadas actualmente para servicios de email en nube como Microsoft 365 o Gsuite [11].

Inteligencia artificial - NLP: Es el uso técnicas de NLP para detección de correos maliciosos mediante el modelamiento del contenido o cuerpo de correos maliciosos, si bien no es una técnica que podamos encontrar en los ESGs, existen herramientas complementarias que han abordado este problema con resultados alentadores como lo es la solución de NVIDIA Morpheus [9] que será usado en este proyecto.

2.1.2 Herramientas para defensa contra ataques de phishing

Las soluciones que podemos encontrar contra defensa de ataques de phishing pueden ser tanto comerciales y open source.

Soluciones Open Source: acorde a la empresa Comodo, proveedor de soluciones defensa contra ciberataques avanzados, existen 3 soluciones que abordan este problema [12]:

- Apache SpamAssassin
- MailScanner
- OrangeAsassing (Basado en SpamAsassin)

Estas soluciones permiten un nivel de defensa basado principalmente en reglas estáticas e integrables con herramientas open source muy populares para el manejo de correos como lo son Sendmail y Postfix, son herramientas que aportan mucho en la detección de emails maliciosos conocidos, pero no hay desarrollo especializado para detección y contención de amenazas avanzadas.

Soluciones Comerciales: tal como lo menciona GreatHorn [8] (Proveedor de soluciones de protección de amenazas por Email) acorde a Gartner, la evolución de las amenazas basadas en email como lo son el Phishing, BCE y Ransomware ha llevado a una nueva definición de los conocidos Email Security Gateway o llamados también Antispam, por las soluciones de Email Security que engloban protecciones no solo en el perímetro como la oferte de CISCO [11] sino incluso a nivel de end-point como por ejemplo la oferta de Sophos [13]. Ambas soluciones se apoyan en la nube dado que el costo de hardware en sitio es elevado.

2.1.3 Soluciones basadas en machine learning

De manera general podemos clasificar en 3 tipos principales de proveedores que abordan esta problemática: primero, quienes vinieron de ser un “Antispam” y han evolucionado su oferta de servicio incorporando algoritmos de machine learning en sus soluciones como por ejemplo CISCO, SOPHOS, segundo: los proveedores de ciberseguridad que nacen basados en

inteligencia Artificial e incorporan como caso de uso la seguridad de comunicaciones por Email, como Darktrace o GreatHorn, y finalmente quienes habilitan a los proveedores de servicio para el desarrollo de productos basados en soluciones de inteligencia artificial como lo es NVIDIA-Morpheus.

2.1.4 Desarrollo de servicios de ciberseguridad basados en AI

Desde la perspectiva de brindar servicios de ciberseguridad basados en AI, encontramos que es un mercado en auge, teniendo cada vez más proveedores que ofrecen esta solución y casi todas en el formato caja negra, que significa que el cliente no conoce el modelo o puede customizarlo, si bien estas cuentan con mecanismos de aprendizaje continuo hay que considerar que su diseño busca cubrir la mayor cantidad de superficie de clientes posibles y no son soluciones personalizadas para un cliente, lo que deja una oportunidad de investigación para ofrecer servicios en este campo.

2.2 Machine learning para la detección de phishing

La detección de phishing se lo puede plantear como un problema de clasificación binaria. Es decir, usando técnicas de aprendizaje supervisado donde tenemos las etiquetas de correos como phishing y no-phishing se pueden entrenar modelos que distingan entre estas categorías dado un conjunto de características de los correos. Por otra parte, el uso de aprendizaje no supervisado en tareas similares a la detección de phishing ha sido extenso; por ejemplo, en el desarrollo de filtros antispam usando clustering o en el desarrollo de sistemas de detección de intrusos con redes neuronales de detección de anomalías. Sin embargo, buscamos que los algoritmos a utilizar exploten las diferencias etiquetadas del conjunto de correos y limitar el uso de aprendizaje no supervisado.

A continuación, en la sección 2.2.1 describimos los algoritmos de machine learning que vamos a modelar en este estudio y algunos de sus resultados en la detección de phishing.

2.2.1 Algoritmos de machine learning

Regresión logística

Se establece el modelo de regresión lineal como una línea base. Este es un método popular para problemas de clasificación binaria. El modelo de regresión logística computa la probabilidad de que un vector de características se clasifique como una de las dos clases usando para esto la función logística. Una de las ventajas del uso de este algoritmo es el componente de explicabilidad del modelo que se obtiene al identificar las características más importantes para la clasificación y como los valores de estas contribuyen a determinar si un correo es o no phishing.

Support Vector Machine (SVM)

El algoritmo SVM intenta encontrar el hiperplano optimo que maximiza la distancia desde el punto más cercano, lo que claramente separa los datos de entrenamiento en dos categorías. Se explota el principio de minimización de riesgo estructural usando un enfoque de aprendizaje de margen máximo. En particular, se elige a la función de perdida de bisagra como función de costo y se maximiza el margen.

En este estudio se utilizó el kernel RBF y los hiperparámetros a optimizar son:

- C: El parámetro de regularización que penaliza el error. La penalización utilizada es l2 cuadrada.
- gamma: El coeficiente del kernel RFM, que cumple la función de ajustar la compensación entre varianza y sesgo.

Se elige el modelo de SVM para este problema dado que es uno de los métodos con mayor número de artículos publicados con respecto a clasificación de correos. El modelo SVM resulta útil en las aplicaciones de detección de phishing porque normalmente se requiere trabajar con características de tipo bolsa de palabras o muchas características indicadoras. En este caso el número de características generadas, así como la dispersión de la naturaleza de dichas características hace ideal el uso de SVM. La cantidad de muestras de instancias seleccionadas para el entrenamiento del modelo también es manejable por SVM.

Random Forest

Random Forest es un algoritmo de ensamblaje que aplicado a problemas de clasificación crea un conjunto de árboles de decisión a partir de características seleccionadas aleatoriamente. Luego, computa los votos de cada árbol de decisión para la etiqueta a predecir y la clase más votada se selecciona para la predicción final.

En este estudio se utilizó el criterio de Gini para medir la calidad de las divisiones y el número máximo de características para considerar la mejor división es la raíz del número de características totales. Los parámetros que se seleccionaron para optimizar este modelo son:

- n_estimators: El número de árboles en el bosque. El propósito es buscar el mínimo que permita obtener resultados estables y una latencia de ejecución mínima.
- max_depth: Máximo número de niveles permitido en cada árbol.
- min_samples_split: El número mínimo de muestras requeridas para dividir un nodo
- min_samples_leaf: El número mínimo de muestras requeridas para que un nodo sea una hoja.

Los últimos tres parámetros nos permiten ajustar la complejidad del modelo y sirven para ajustar la compensación entre varianza y sesgo.

Se elige el modelo RF para este problema por su capacidad para trabajar con grandes volúmenes de datos de forma eficiente (escalabilidad del modelo a futuro); porque, a pesar de tener alto costo computacional, es un modelo de ensamblaje que nos permite paralelizar procesos y aprovechar al máximo los recursos computacionales, a diferencia de GBM. Por último, RF nos otorga mayor interpretabilidad del modelo; es decir, se puede identificar las características más importantes en la clasificación lo que nos sirve para entender cómo funciona el proceso y poder realizar cambios educados en el modelo a futuro o definir un modelo de aprendizaje basado en reglas.

2.2.2 Manejo de desbalance de clases

El problema de desbalance de clases es frecuente en varias aplicaciones de clasificación usando machine learning. Entre las más representativas tenemos aquellas relacionadas con fenómenos poco comunes como son enfermedades, actividades fraudulentas y en eventos de seguridad. La limitación de los algoritmos de machine learning al momento de ser entrenados con datos desbalanceados consiste en dar mayor relevancia a la clase mayoritaria; es decir, el modelo tiende a favorecer la clase mayoritaria e ignorar aquella con pocas observaciones en su inferencia. Esto se debe a que los modelos tienden a maximizar métricas como accuracy y a minimizar errores en su entrenamiento.

Los métodos para el desbalance de clases se dividen en dos grupos:

- Métodos de muestreo: Estas técnicas se enfocan en modificar la distribución de las clases en el conjunto de datos. Los dos enfoques fundamentales en estos métodos son undersampling y oversampling. Undersampling consiste en realizar una muestra aleatoria de la clase mayoritaria; en cambio, oversampling consiste en replicar aleatoriamente las observaciones de la clase minoritaria. Ambos métodos tienen sus desventajas, undersampling puede llegar a eliminar información importante acerca de la clase mayoritaria y el oversampling puede resultar en un sobreajuste del modelo.
- Métodos del algoritmo: Estos métodos se enfocan en hacer modificaciones al algoritmo de machine learning en lugar de a los datos. Por ejemplo, el método de cost-sensitive learning busca que el algoritmo no este sesgado hacia la clase mayoritaria al modificar la penalización de los errores de clasificación en la función de perdida, dando mayor peso a la clase minoritaria. Otra forma de tratar el desbalance de clases modificando al algoritmo consiste en ajustar el umbral de clasificación para algoritmos que retornan un puntaje que representa la probabilidad de pertenecer a una clase [14].

Haixiang et al. recomiendan el método de oversampling cuando existe un extremo desbalance de clases limitadas observaciones de la clase minoritaria [15]. Este método de tratamiento para el balance de clases parece ser adecuado para la problemática del estudio.

2.2.3 Ingeniería de características

En las soluciones de ML para este problema se plantea el uso de las siguientes categorías de características:

Características léxicas: Se obtienen a partir del cuerpo del correo. Entre las más comunes están: longitud del texto, contadores de símbolos, letras y números, distribución de espacios, presencia de palabras sospechosas, numero de directorios y entropía del texto.

Características de remitente: Se obtienen a partir de la dirección de correo del remitente. Generalmente, se refiere a características que pueden obtenerse a través de consultas a fuentes externas del dominio de la cuenta de correo. Por ejemplo, geolocalización y popularidad.

El conjunto de datos es de naturaleza no estructurada, por lo que el componente de ingeniería de características es esencial para el uso de algoritmos de machine learning; del mismo modo que la extracción de características se realiza de forma automática con los modelos de deep learning. Para los modelos de machine learning se van a extraer las características tanto del cuerpo del correo como del remitente.
Se plantea extraer características léxicas en los siguientes tipos para resumir su función:

- Características de conteo: Variables numéricas discretas que representan contadores del número de caracteres. Estas son: número de @, ., %, *, &, /, ;. ,, #, ~, =, -, _, ?, total de símbolos, total de números, total de letras, numero de directorios.

- Características de medida: Variables numéricas discretas que representan la longitud de partes del correo. Estas son: longitud del párrafo, del espaciado, y cantidad de párrafos.
- Características de ratios: Variable numérica continua que representan la proporción de cierto tipo de caracteres en el correo. Estas son: porcentaje de símbolos, porcentaje de números y porcentaje de letras.
- Características binarias: Entre ellas están: si el correo contiene palabras sospechosas (banco, cuenta, ingresar, gratis, suerte), si tiene enlaces y si usa servicios de acortamiento de URL, si tiene archivos adjuntos.
- Características léxicas complejas: La mayoría de literatura sugiere el uso de la entropía del URL como una característica importante. Esto se debe a que la mayoría de las familias de ataques maliciosos usan algoritmos de generación de correos masivos para aumentar el alcance del ataque. En esta aplicación la entropía nos ayuda a medir la aleatoriedad o incertidumbre de la distribución de palabras en el cuerpo del correo.

2.2.4 NLP para la detección de phishing en correo

El procesamiento de lenguaje natural (NLP) es una tecnología que permite a las computadoras entender el lenguaje humano. El desarrollo creciente en las técnicas de NLP; en particular, de la incrustación de palabras ha contribuido al desarrollo de sistemas de detección de phishing robustos que explotan la morfología y semántica del cuerpo del correo [16]. La tarea principal del NLP en la problemática es la extracción de características a partir del texto. De acuerdo con este enfoque Bountakas et al. evaluó el rendimiento de tres modelos de NLP TF-IDF, Word2Vec y BERT para la extracción de características del cuerpo de correos y su posterior clasificación usando varios algoritmos de clasificación tradicionales de machine learning. El autor encontró que Word2Vec tuvo el mejor rendimiento independientemente del método de clasificación tanto con datos balanceados como no balanceados en los conjuntos de datos de Enron y Nazario [17].

Examinaremos brevemente ahora los modelos de NLP usados en el estudio anterior:

TF-IDF (Term Frequency – Inverse Document Frequency) Es un método que proporciona un peso a cada palabra representando su importancia en un conjunto de documentos. Este puntaje tiene dos componentes. TF mide el número de veces que la palabra aparece en el texto, este valor es normalizado con respecto a la extensión del documento. En cambio, IDF se concentra en la importancia general del término en un conjunto de documentos. El producto de estos componentes resulta en la construcción del conjunto de características de los documentos.

Word2Vec es una técnica de NLP con la capacidad de identificar el contexto de una palabra en un texto y su relación con otras palabras tanto en su significado como en la gramática. Este modelo genera una representación vectorial de las palabras en un espacio latente donde palabras similares son cercanas.

BERT (Bidirectional Encoder Representations from Transformers) es uno de los modelos más prominentes en el estado del arte y con resultados comprobados en un contexto similar, BERT obtuvo un accuracy de 98.67% en la detección de correos spam en inglés [18]. Este modelo de NLP obtiene representaciones del texto entendiendo el contexto de las palabras al incluir tanto el contenido previo y posterior a estas. La arquitectura base de BERT son los transformers que son sofisticadas redes neuronales de atención cuyo potencial aún se encuentra en exploración.

Los autores de BERT resaltan que el uso de un masked language model (MLM) es clave en el entrenamiento del modelo ya que esto implica predecir palabras ocultas en una oración mediante la cohesión de la representación de

la oración tanto en sentido izquierda a derecha, así como de derecha a izquierda [19].

En la revisión del estado del arte no se encontró estudios para la detección de phishing en correos en español utilizando estas técnicas de NLP. Sin embargo, modelos tipo BERT como M-BERT (Multilingual BERT) se han aplicado con éxito en la detección de correos tipo spam bilingües. Cao y Lai usaron M-BERT para detectar correos spam tanto en chino como en inglés y obtuvieron un accuracy de 96.48% [20]. Por otro lado, modelos BERT con soporte para lenguaje español tenemos a Spanish-BERT (BETO), Spanish-RoBERTa, ALBETO y Distil-BETO, todos estos modelos tienen un accuracy alrededor del 96% en tareas de clasificación de documentos en el conjunto de datos MLDoc [21].

2.3 Librerías y herramientas disponibles para la problemática

Las herramientas para este estudio las vamos a agrupar por su uso en las etapas definidas en la metodología. En el procesamiento de datos tenemos Rapids cuDF. Para el modelamiento: RAPIDS cuML, HuggingFace Transformers y pytorch. Por último, en producción esta NVIDIA MORPHEUS y una herramienta de SIEM como Splunk o ELK.

El estudio cuenta además con el soporte de infraestructura de la compañía que incluye una estación NVIDIA DGX A100 que es una infraestructura dedicada para el desarrollo de tecnologías de inteligencia artificial hecho por NVIDIA. Con el propósito de obtener los mejores resultados con este equipo se decide enfocarse en las herramientas y librerías optimizadas para GPU de NVIDIA como lo son RAPIDS y MORPHEUS.

RAPIDS es una librería de código abierto desarrollada por NVIDIA que se utiliza para el desarrollo de pipelines de proyectos de ciencia de datos y analítica con procesamiento de GPU. Esta librería esta optimizada para los procesos de ETL con

fácil integración a Python, escalabilidad para varios GPUs y compatibilidad con varios ambientes de despliegue de machine learning [22].

cuDF es un componente de RAPIDS para el manejo de DataFrames en Python con una sintaxis bastante similar a la de Pandas. Es decir; cuDF permite la carga, manipulación y transformación de datos usando GPU. Se ha llegado a observar que cuDF es hasta 16 veces más rápido que Pandas en la ejecución de estas tareas [23].

cuML es la parte de RAPIDS diseñada para el entrenamiento e inferencia de modelos de machine learning con GPU. Del mismo modo que cuDF, cuML posee una sintaxis parecida a la de su homólogo en CPU Scikit-Learn. Sin embargo, con una ventaja en infraestructura de GPU se puede llegar a tener un entrenamiento hasta 150 veces más rápido que usando Scikit-Learn [24].

HuggingFace Transformers es una API que permite el acceso a herramientas y pesos de modelos pre-entrenados de NLP en el estado del arte para aplicar fine-tunning y transfer learning [25]. Esta API tiene el soporte para la carga de pesos pre-entrenados de las versiones de BERT en español.

Pytorch es una librería para el desarrollo de modelos de Deep learning a bajo nivel. Pytorch es la librería de preferencia para investigadores en el desarrollo de estudios y nuevos modelos con una amplia popularidad sobre otras librerías similares como Tensorflow o MXNET. Esto también representa una considerable ventaja en cuanto a la documentación y soporte disponible en la web para los proyectos en Pytorch.

CAPÍTULO 3

3. METODOLOGÍA

3.1 Exploración y validación de datos

En la presente sección abordaremos el proceso de construcción del dataset mediante la extracción, etiquetado y el análisis exploratorio de datos.

El dataset inicial se obtuvo desde el ESG, que es la herramienta de seguridad por donde se procesan los correos entrantes de dominios externos a la compañía. El ESG permite exponer un servicio mediante el protocolo IMAP para obtener los emails entrantes en tiempo real; los datos más relevantes con los que cuenta el dataset se muestran en la siguiente tabla.

Tabla 3.1 Descripción del dataset inicial

Campo	Descripción	Tipo
Time	Fecha y hora de transacción	Date time
From	Remitente del correo	Texto
To	Destinatario del correo	Texto
Client IP	IP del servidor remitente del correo	Texto
Client name	FQDN del servidor remitente del correo	Texto
Subject	Asunto del correo	Texto
Classifier	Categoría del correo reportado por el Antispam	Texto
Disposition	Acciones tomadas por el Antispam, por ejemplo, aceptar o rechazar	Texto
URL	URL dentro del cuerpo del correo	Texto
Body	Cuerpo del correo electrónico	Texto
ID	Indicador único del mensaje	Texto

EL ESG dispone de campos adicionales a los listados en la **¡Error! No se encuentra el origen de la referencia.** los cuales son usados actualmente para la gestión de incidentes de la empresa. Para el establecimiento del modelo de detección de phishing mediante técnicas de NLP usaremos el cuerpo del correo (Body).

Extracción de datos

Pipeline de etiquetado

Análisis de datos

Preparación de datos para modelamiento

- División de datos
- Balanceo de clases

Extracción de características

-

3.1.1 **Pipeline de extracción de datos**

Para la extracción de datos se usó Apache NiFi como herramienta de ETL, la cual permite establecer conexión mediante protocolo IMAP al ESG y así obtener el contenido del correo electrónico entrante. Luego de obtener el correo, Apache NiFi dispone de funciones que le permiten procesar la estructura de un correo estándar y realizar la extracción de parámetros que serán usados para la construcción del Dataset.

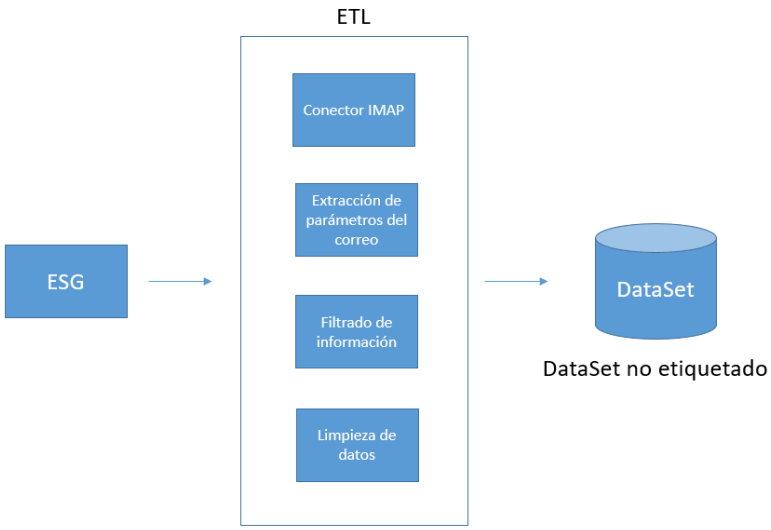


Figura 3.1 Extracción de datos y DataSet no etiquetado

En la Figura 3.1 se muestran los sub procesos realizados por el ETL, donde posterior a la conexión por IMAP y la extracción de parámetros del correo, se realiza un proceso de filtrado donde se excluyen correos asociados a lista blanca que no necesitan ser analizados por la empresa, por ejemplo, alertas

automatizadas de sistemas internos y cuentas de correo consideradas confidenciales que están fuera del alcance del prototipo y dado que este estudio se enfoca en características léxicas con técnicas de NLP se aplican filtros que incluyan correos cuyo cuerpo contenga únicamente texto, el cual es un dato identificado por el ESG.

La extracción de datos duró aproximadamente una semana donde se obtuvieron en total 2100 coreos únicos (diferentes entre sí) seleccionados para la fase de etiquetado.

3.1.2 **Pipeline de etiquetado**

Durante esta etapa se procedió con la clasificación 2100 correos de la fase anterior, el etiquetado fue realizado manualmente y con validación cruzada por parte de ingenieros con formación y experiencia en ciberseguridad obteniendo un total de 2030 correos clasificados como buenos y 70 como malos denotando un alto desbalance de clases e ineficiente para modelamiento, por lo que se requiere una fuente adicional para correos maliciosos. Para obtener estos correos se buscó fuentes externas, sin embargo, las bases de datos de emails de phishing en español son inexistentes públicamente pese a la búsqueda realizada, por consiguiente, se procedió a construir manualmente un set de datos basados en reportes de ataques previos a la empresa, así como informes de organismos de ciberseguridad en el Ecuador (ECUCERT); de este proceso se obtuvieron 549 correos maliciosos únicos diseñados por ingenieros especialistas ciberseguridad representativos acorde a las campañas de ataque recibidos por la empresa y el ecuador en los últimos años.

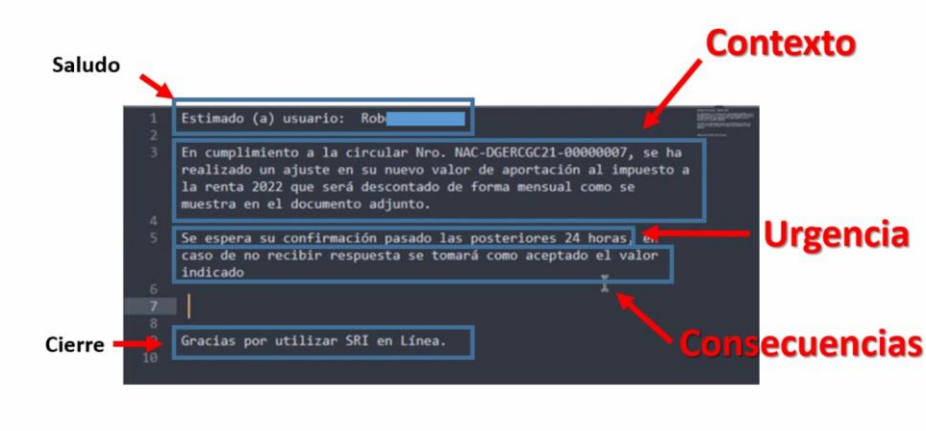


Figura 3.2 Criterio de construcción de Email de Phishing

En la Figura 3.2 podemos ver el criterio utilizado para la construcción del dataset de emails maliciosos con los siguientes componentes:

- Saludo: la cordialidad e identificación personalizada del usuario receptor.
- Contexto: una explicación de una situación que atrae la atención del usuario.
- Urgencia: cita una condición de importancia o urgencia basado en tiempo a fin de poder persuadir al usuario a tomar decisiones en tiempo corto.
- Consecuencias: afectación para el usuario si no realiza las acciones solicitadas, se espera que estas consecuencias inciten al usuario a tomar decisiones apresuradas para evitarlas.
- Cierre, la cordialidad de despedida personalizada según la campaña de phishing.

Finalmente, para mantener consistencia de la estrategia de construcción del dataset, se decidió usar los 2030 correos empresariales clasificados como buenos y los 549 maliciosos generados.

3.1.3 **Análisis de datos**

El análisis exploratorio de datos permitió identificar un desbalance de clases considerable para la problemática, así como características relevantes de los

datos que nos permiten caracterizar a los correos phishing y no phishing. Primero, se observó que, del total de los 2579 correos, solo 549 son tipo phishing y los 2030 restantes eran benignos como este ilustrado en la Figura 3.1. Esto implica que se debe aplicar técnicas de balanceo de clases en los datos de entrenamiento.

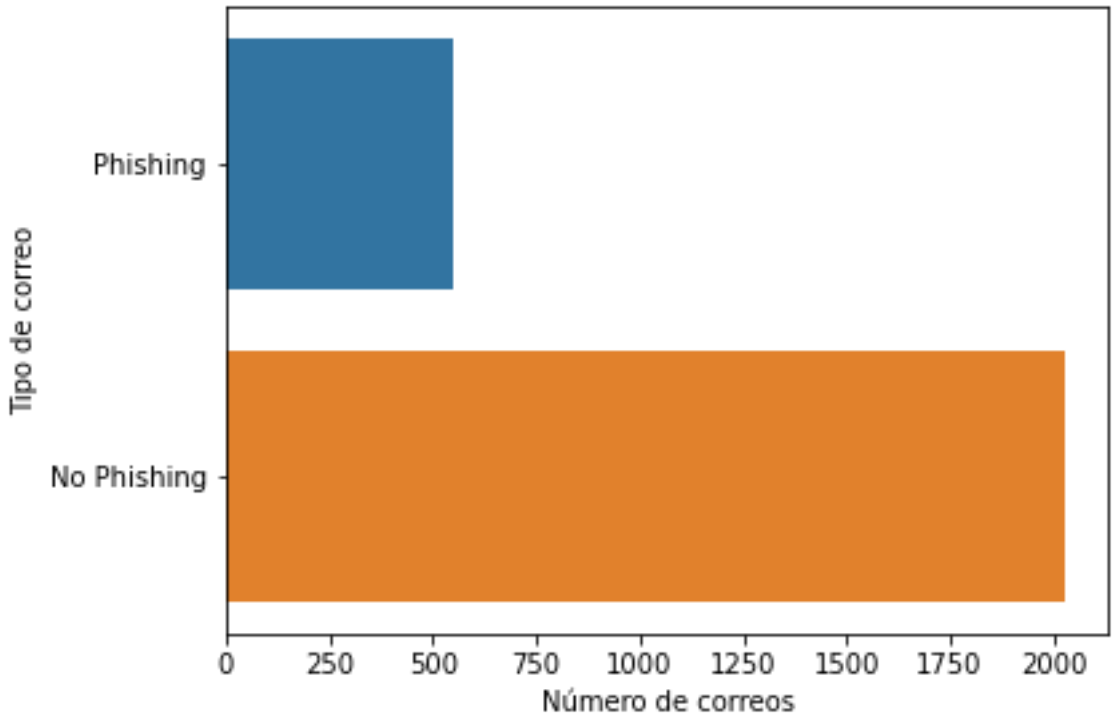


Figura 3.3 Número de correos en cada clase

Además, se examinó características como el número de caracteres, número de palabras y palabras más frecuentes en cada clase para tener un perfil de cada clase. En cuanto a la distribución de número de caracteres y de palabras se observó resultados similares. Los diagramas de caja en la Figura 3.4 y la Figura 3.5 muestran que los correos tipo phishing tienden a tener menos caracteres y menos palabras que los correos benignos. Del mismo modo, los correos tipo phishing pueden llegar a tener valores atípicos en ambos casos comparado con los correos benignos.

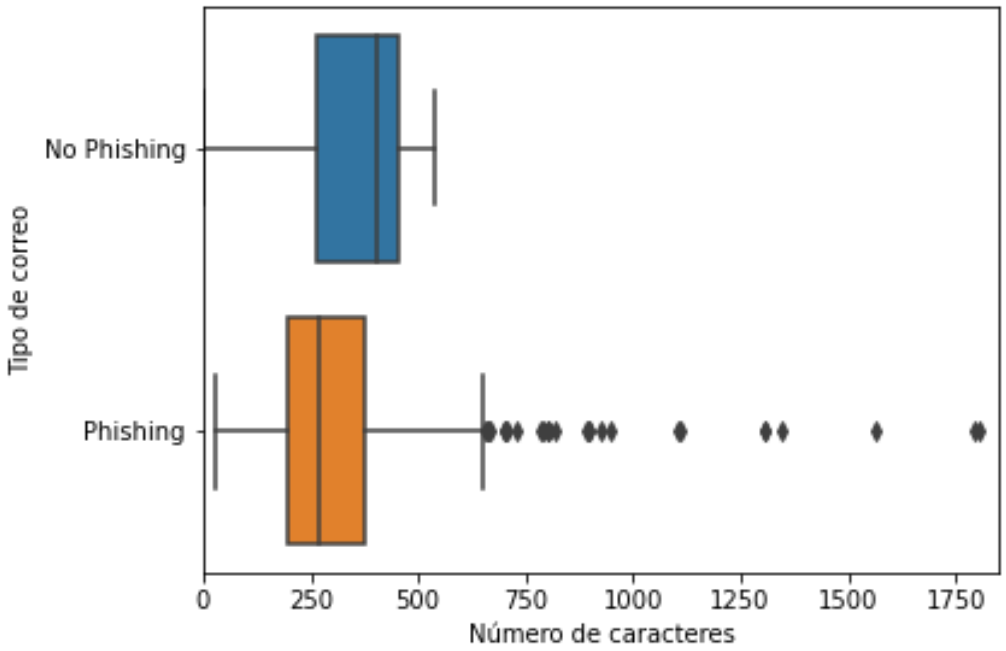


Figura 3.4 Diagrama de caja del número de caracteres en cada clase de correos

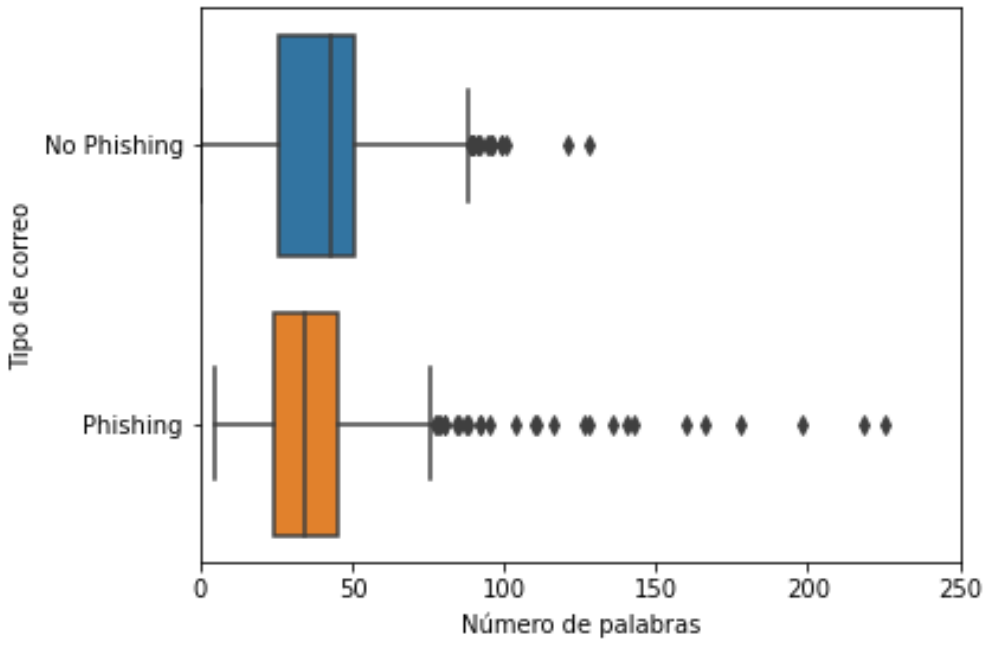


Figura 3.5 Diagrama de caja del número de palabras en cada clase de correos

Finalmente, se investigó si es que había alguna distinción en las clases en cuanto a la frecuencia de palabras. Como está ilustrado en las Figura 3.6 y Figura 3.7, la mayoría de los correos tipo phishing usan palabras relacionadas a extracción de información (datos, cuenta), amenazas (condicional sí) y redireccionamiento a un sitio web (clic, siguiente, enlace); por otro lado, los correos benignos tienden a tratar temas de soporte (ayuda, favor) y comunicación corporativa (cliente, agosto, enviado).

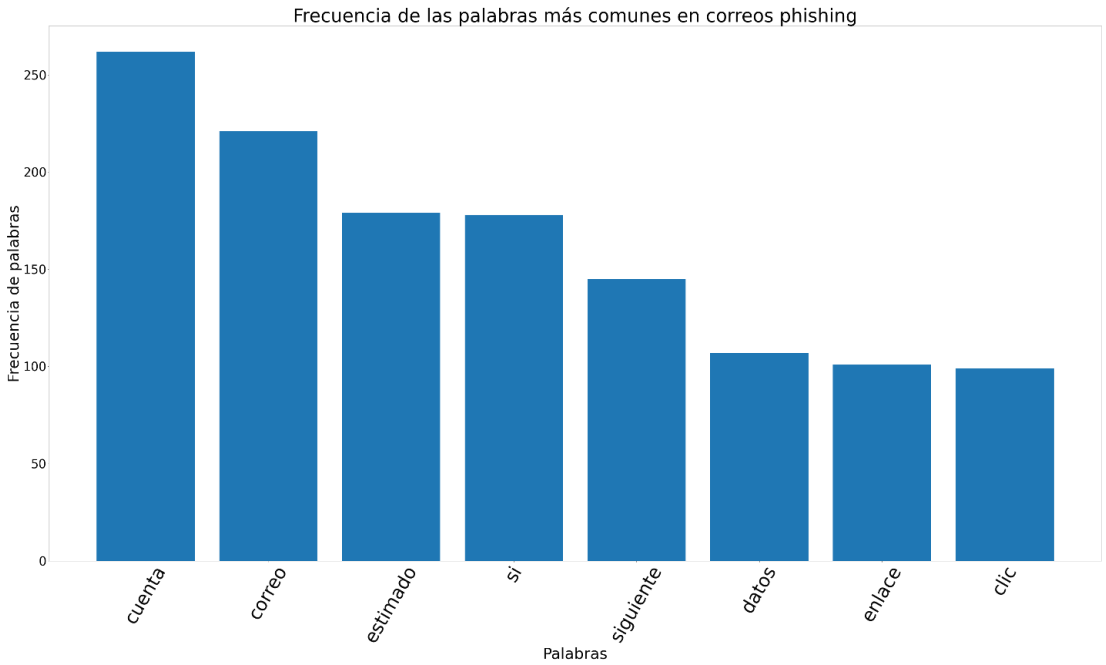


Figura 3.6 Frecuencia de las palabras más comunes en correos phishing

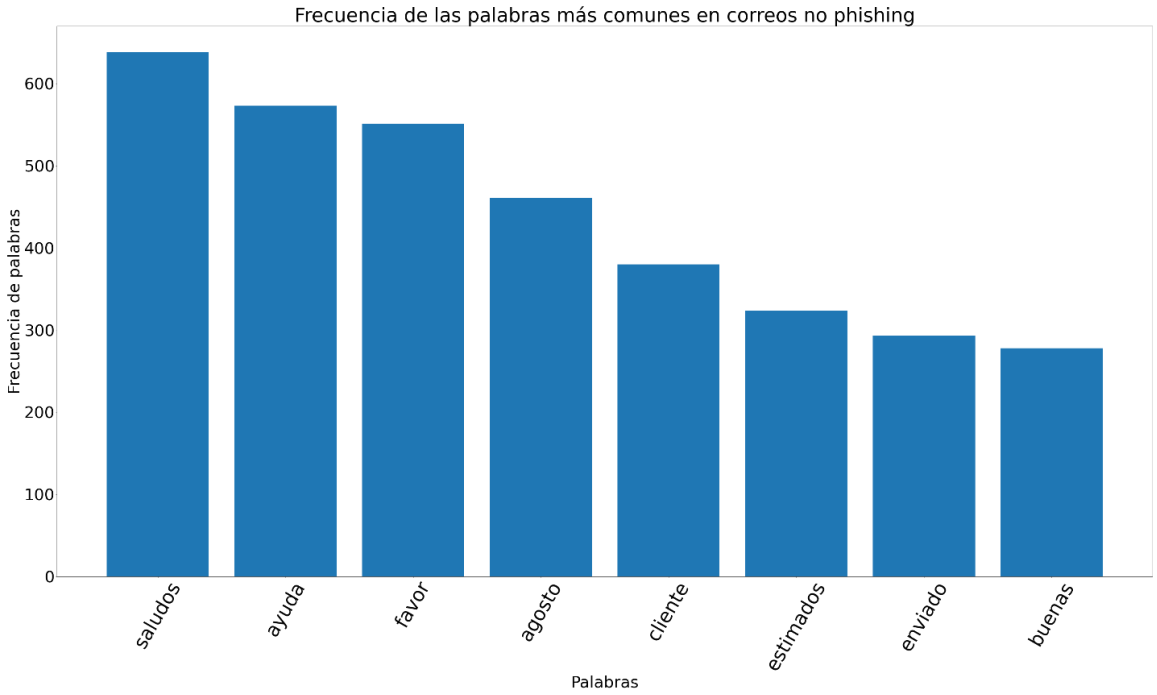


Figura 3.7 Frecuencia de palabras más comunes en correos no phishing

Por lo tanto, el análisis exploratorio de datos encontró que es viable detectar correos tipo phishing usando algoritmos de clasificación ya que ambas clases tienen un perfil definido. Sin embargo, se debe considerar técnicas para mitigar el desbalance de clases.

3.1.4 Preparación de datos para modelamiento

Previo a las etapas de extracción de características y modelamiento se realizan dos etapas de procesamiento:

1. **División de datos:** Se identificó 2030 correos benignos y 549 correos tipo phishing en los datos estructurados. Estos datos fueron divididos en los dataset de entrenamiento y dataset de evaluación con una proporción 0.8 para los datos de entrenamiento y con una división estratificada para mantener la proporción de clases en ambos conjuntos de datos. Los resultados de este proceso fueron un dataset de entrenamiento con 1528 correos benignos y 406 correos tipo phishing y un dataset de evaluación con 502 correos benignos y 143 correos tipo phishing. Se resalta la importancia de aplicar métodos para balanceo de clases en nuestro dataset de entrenamiento.

2. **Balanceo de clases:** Se aplicaron técnicas de generación de datos artificiales para NLP en los datos de entrenamiento con el fin de balancear las clases. Edward Ma indica en la documentación de la librería nlpaug que se utiliza traducción de ida y vuelta y generación por sinónimos para obtener documentos similares a los originales, pero con cierta variabilidad. En nuestro caso se seleccionó los 406 correos tipo phishing en el dataset de entrenamiento y se aplicó traducción de español a inglés y viceversa en estos correos usando el modelo OPUS MT de la librería transformers de HuggingFace como se ilustra en la Figura 3.8; también se usó BERT multilingual base uncased para encontrar sinónimos de palabras con entendimiento de su contexto y remplazar en un rango de 1 a 10 palabras del correo. Ambos métodos se ejecutaron envueltos en las funciones de la librería nlpaug. Finalmente, se obtuvo 810 correos phishing adicionales; con esto, se tiene un dataset de entrenamiento más balanceado con 1216 correos phishing y 1528 correos no phishing. Con el fin de obtener un dataset completamente balanceado se utilizó downsampling con los correos no phishing de modo que resultaron 1216 correos en cada clase.

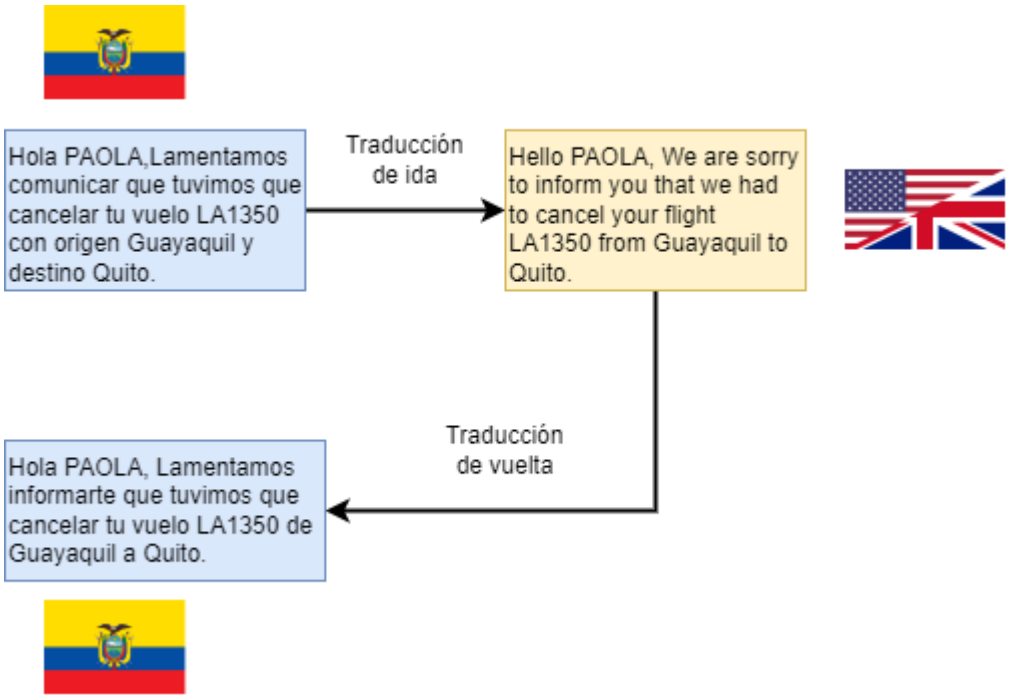


Figura 3.8 Traducción de ida y vuelta para balanceo de clases

3.2 Extracción de características

En esta tesis abordamos dos enfoques para resolver el problema de clasificación de correos tipo phishing:

- Extracción de características con TF-IDF y uso de modelos tradicionales de Machine Learning
- Extracción de características y clasificación usando modelos de Deep Learning tipo BERT pre-entrenados.

3.2.1 Extracción de características para modelos de Machine Learning

La extracción de características mediante TF-IDF se realiza en dos etapas como se ilustra en la Figura 3.9:

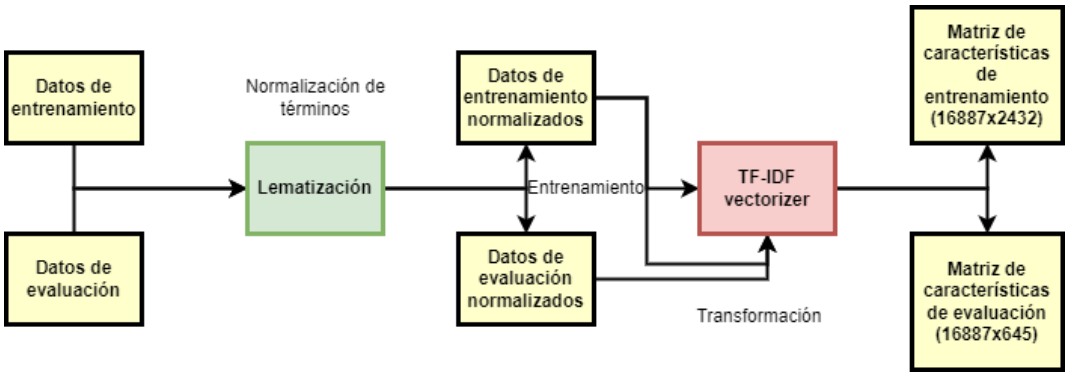


Figura 3.9 Extracción de características con TF-IDF

1. **Lematización:** Primero se normaliza el cuerpo de los correos mediante lematización. Aquí usamos la librería spacy de Python para agrupar palabras de significado similar significado en una sola representación textual con el propósito de remover ruido del conjunto de datos con palabras inusuales, pero con significado común. Se eligió spacy para esta tarea dado que era una de las pocas librerías con soporte para español. El resultado de este proceso son los datasets normalizados.
2. **Tokenización:** Finalmente, la extracción de características se realizó en dos etapas encapsuladas con la clase de modelo TfidfVectorizer de cuml rapids. En esta etapa se ajusta el modelo con los datos de entrenamiento y luego se transforma tanto los datos de entrenamiento como los de evaluación. Primero se reemplazan las palabras del correo con identificadores numéricos durante la tokenización creando una matriz de frecuencia para estas características. Luego, se aplica la técnica TF-IDF que pondera los valores de frecuencia de las características respecto a su distribución en el conjunto de correos de entrenamiento. Se seleccionó hiperparámetros para el ajuste de la extracción de características correspondiente al rango de n-gramas considerados, para esto se consideró los valores de unigramas junto con bigramas. El resultado de esta etapa es la matriz de características para ambos datasets con 16887 características. Cabe notar que este proceso se lo debe llevar a cabo después de la división de datos ya que el método TF-IDF depende de la

distribución de palabras en el conjunto de correos a analizar; es decir, si se realiza la división de datos después de la extracción de características puede ocasionar problemas de data leakage.

3.2.2 Extracción de características para modelos tipo BERT

La extracción de características para los modelos tipo BERT se la realiza en dos niveles como se ilustra en la Figura 3.10:

1. **Tokenización:** Los modelos tipo BERT usados en este proyecto corresponden a modelos con base en roBERTa para español; es decir, utilizan un tokenizador BPE (Byte-Pair Encoding) que asigna una representación numérica al cuerpo del correo basado en la identificación de pares comunes de caracteres (este proceso incluye la normalización que se realizaba con el método anterior). Es importante notar que este tokenizador es un estimador pre-entrenado y no requiere volver a entrenar. El resultado de esta etapa es dos componentes: input-ids que es la representación numérica del cuerpo del correo y la attention mask que sirve para identificar la posición de los tokens de relleno. La longitud de estos componentes es un hiperparámetro que se mantuvo fijo en 128 debido al framework de inferencia usado.
2. **Capas de extracción de características:** Las capas iniciales de los modelos tipo BERT se conocen como capas de extracción de características. Estas capas, dado que los modelos ya son entrenados previamente, permiten obtener una representación vectorial del resultado de la tokenización que captura el valor semántico del texto inicial. En este caso, la capa previa al clasificador resulta en un vector de características del cuerpo del correo. En nuestro estudio se mantiene fijo el número de unidades en 768 (el que usa por defecto los modelos tipo BERT) y no se altera la estructura de los clasificadores de cada modelo usado, siendo común el uso de una capa lineal.

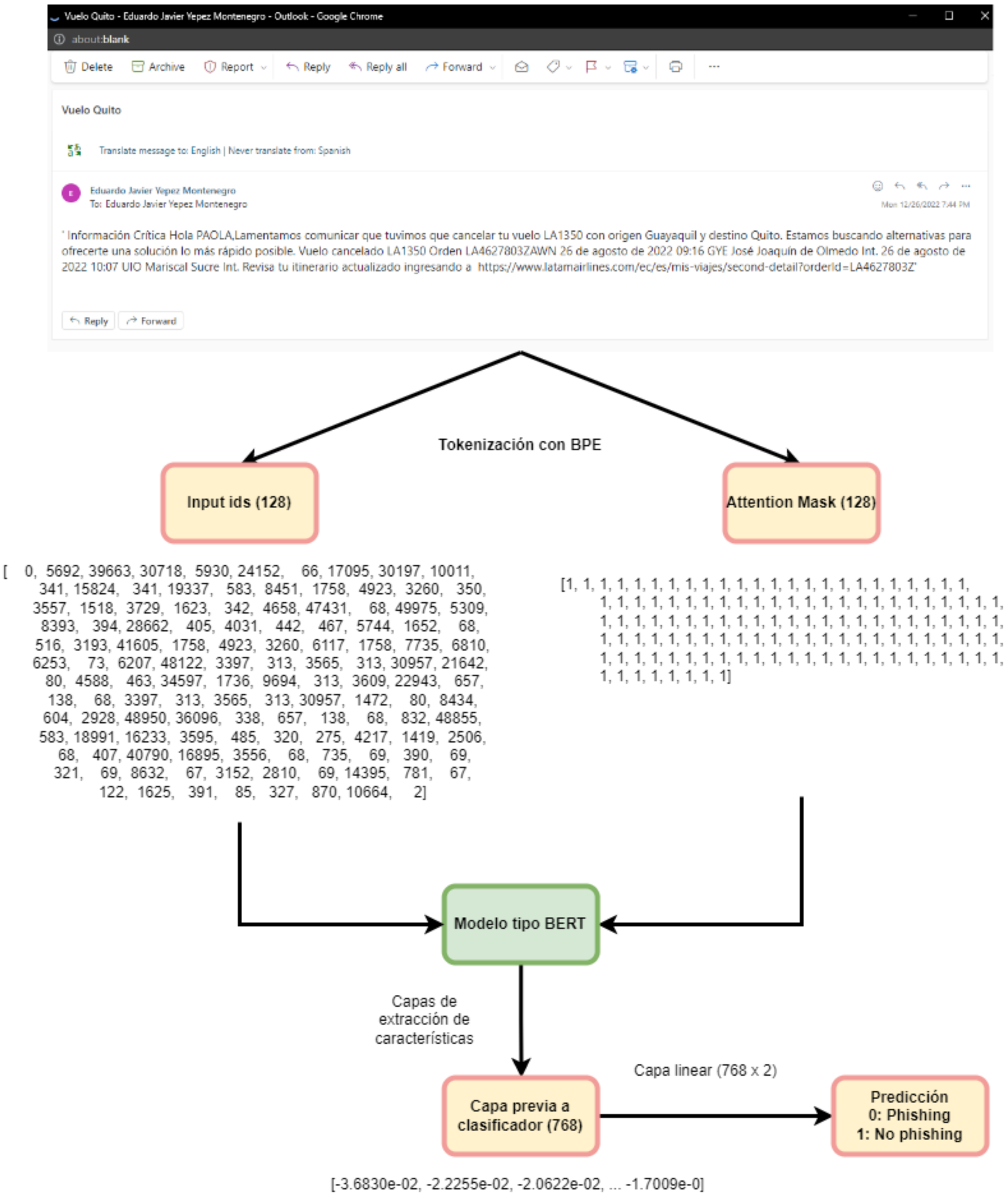


Figura 3.10 Extracción de características con modelos tipo BERT

3.3 Métricas de evaluación del modelo

En cuanto a la selección de métricas para la problemática se consideró como positivo como los correos con contenido tipo phishing y negativo a aquellos que no tienen contenido phishing ya que nuestro objetivo es medir la efectividad para la detección de phishing. Con esta definición tenemos que los Verdaderos Positivos (TP) son los correos identificados correctamente como phishing, los Falsos Positivos (FP) son los

correos erróneamente clasificados como phishing, los Verdaderos Negativos (TN) son los correos identificados correctamente como no phishing y los Falsos Negativos (FN) son el número de correos erróneamente clasificados como no phishing. Se resalta la importancia de minimizar falsos negativos en nuestra problemática dado el riesgo que representa el proveer al usuario un correo malicioso como benigno. Con esta consideración se decidió usar las siguientes métricas de clasificación en nuestro estudio:

- **Exactitud:** La proporción de predicciones correctas sobre totales. Se la usa en el estudio como una métrica inicial de referencia del rendimiento de los modelos en comparación con los resultados en el estado del arte.

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Exhaustividad:** La proporción de correos phishing correctamente identificados sobre el total de correos phishing predichos. Esta se considera como la métrica más importante en el estudio ya que nos permite medir el riesgo que representa el uso de esta herramienta de automatización. El objetivo del modelo es minimizar el riesgo que existe en que al usuario le llegue un correo phishing. Por lo tanto, a mayor exhaustividad menor riesgo de phishing.

$$Exhaustividad = \frac{TP}{TP + FN}$$

- **Precisión:** La proporción de correos phishing correctamente identificados sobre el total de correos phishing reales. Esta métrica se observó con el fin de estimar la efectividad del modelo en cuanto a la automatización en la detección de phishing dado que una baja exhaustividad implicaría revisión manual de los correos clasificados como phishing para reclasificar los benignos entre ellos. Por lo tanto, a mayor precisión menor costo operativo de la herramienta.

$$Precisión = \frac{TP}{TP + FP}$$

- $\boldsymbol{F_\beta}$ **Score:** Es una métrica que agrega exhaustividad y precisión donde β es un factor real positivo que representa la proporción de importancia de exhaustividad con respecto a precisión. En nuestro análisis se elige esta métrica para la optimización de hiperparámetros y la evaluación de modelos. Se estima que el valor de β es 2 mediante un análisis de costos del negocio.

$$\boldsymbol{F_\beta} = \frac{(1+\beta^2)\cdot TP}{(1+\beta^2)\cdot TP + \beta^2 \cdot FN + FP}$$

Se decide usar como métrica de evaluación al F_β score en lugar de otras métricas similares como AUC ya que el primero es más robusto a clases desbalanceadas como es en esta problemática. Con el fin de tener una representación tabular y grafica de la evaluación de modelos se va a registrar las matrices de confusión, así como la curva de precisión y exhaustividad. Un componente fundamental para la selección de modelos consiste en la velocidad de inferencia dado que la a solución requiere detección de phishing en tiempo real.

- **Precisión promedio:** Es un valor que resume la curva de precisión y exhaustividad en un solo valor. Se lo define como el promedio ponderado de los valores de precisión en cada umbral de la curva de precisión y exhaustividad donde los pesos son dados por los valores de exhaustividad del umbral anterior; es decir, es una aproximación del área bajo la curva de precisión y exhaustividad. Esta métrica se interpreta como mejor entre más alta.

3.4 Construcción del modelo de detección de phishing

3.4.1 Modelos de Machine Learning

Se consideró a los modelos de regresión logística, SVM, Random Forest y XGBoost para evaluar su rendimiento en la clasificación de correos tipo phishing. Para cada modelo se realizó un entrenamiento inicial con los hiperparámetros por defecto de su implementación en la librería rapids cuML. Luego, se realizó la optimización de hiperparámetros donde se establece un espacio de búsqueda acorde con lo mencionado en el capítulo 2 y se utilizó la librería Optuna que emplea optimización bayesiana de hiperparámetros con el algoritmo TPE (Tree-structured Parzen Estimators). Para esta etapa se usó validación cruzada con los datos de entrenamiento en 5 iteraciones y se buscó maximizar el F beta score. En total, se realizaron 120 experimentos para todos los modelos excepto SVM que dado su alto tiempo de ejecución se redujo a 50 experimentos.

3.4.2 Modelos tipo BERT

En cuanto a los modelos de deep learning tenemos los modelos basados en BERT como Spanish-BERT (BETO), RoBERTa, ALBERT Spanish, BERTIN y Distil-BETO que se encontraron en el estado del arte con un alto rendimiento en tareas similares. Aquí aplicamos un filtro de selección para aquellos modelos que tengan dos entradas y una salida; input_ids, attention_mask y output respectivamente ya que ese requisito es necesario al momento de desplegar el modelo en tiempo real usando la infraestructura de NVIDIA. Los modelos que cumplen este requisito son RoBERTa, BERTIN y Distil-BETO. Los modelos son cargados usando la librería HuggingFace transformers donde están pre-entrenados para la generación de lenguaje natural. Considerando que estos modelos se van a usar en la detección de phishing en correos se utiliza fine-tunning para adecuarlos a la tarea de clasificación binaria en este contexto usando la clase AutoModelForSequenceClassification de la librería transformers. Cada uno de estos modelos utiliza su propio tokenizador que se encuentra alojado en HuggingFace y tiene su propio vocabulario basado en el pre-entrenamiento realizado.

El entrenamiento de estos modelos fue similar a la de los modelos de machine learning con un entrenamiento inicial con parámetros por defecto y su posterior búsqueda de hiperparámetros. Los parámetros por defecto para todos los modelos se establecieron en 4 epochs, batch size 32, sequence length 128, learning rate 3e-05 y weight decay 0. En la búsqueda de hiperparámetros para todos los modelos se estableció un espacio de búsqueda acorde a la Tabla 3.2.

Tabla 3.2 Hiperparámetros a optimizar en modelos tipo BERT

Hiperparámetro	Desde	Hasta
Número de epochs	2	5
Learning Rate	4e-05	1e-02

Weight Decay	4e-05	1e-02

Se decide mantener batch size en 32 y sequence length en 128 dado que esto es necesario para la puesta en producción de los modelos usando el framework de Nvidia Morpheus. Dado el costo computacional del entrenamiento de estos modelos, se realizaron 20 experimentos (sin validación cruzada, pero con datos de validación extraídos de los datos de entrenamiento) para cada modelo usando la librería Optuna y maximizando el F beta score.

3.4.3 Resultados del entrenamiento de modelos y selección modelo

Finalmente, se obtuvo que el mejor modelo con sus parámetros optimizados fue BERTIN con un F-beta score de 0.99 como lo muestra la **¡Error! No se encuentra el origen de la referencia.** y se puede ver su rendimiento en los datos de evaluación en la Figura 3.12 y Figura 3.13. Se consideró incluir el asunto del correo junto con el cuerpo con el fin de mejorar el rendimiento de los modelos; sin embargo, la mejora fue marginal (0.007 en F-beta score para BERTIN) por lo que no se decidió incrementar la complejidad del modelo. Este modelo fue optimizado y aproximado usando la librería onnxruntime con el fin de reducir los tiempos de inferencia y obtener un modelo en un formato onnx para su despliegue en nuestro servidor de inferencia.

Tabla 3.3 Resultados de entrenamiento y optimización de modelos

Version	Modelo	Accuracy	Recall	Precision	F1	Fbeta	AP score	Runtime (ms)
No Optimizado	SVM	0.945736	0.895105	0.864865	0.922345	0.888889	0.962188	15369
	RF	0.903876	0.902098	0.728814	0.871166	0.861148	0.951747	177785
	LR	0.936434	0.902098	0.826923	0.910752	0.885989	0.960986	**1791**
	XGBCL	0.934884	0.895105	0.825806	0.908361	0.88033	0.95911	2743
	RoBERTa	0.886822	0.888112	0.690217	0.850477	0.839947	0.94406	341474
	BERTIN	0.982946	0.958042	0.964789	0.975229	0.959384	0.985982	340757
	DistillBERT	0.917829	0.902098	0.767857	0.887723	0.871622	0.955706	218409
Optimizado	SVM	0.944186	0.881119	0.868966	0.919536	0.878661	0.958896	15761
	RF	0.92093	0.909091	0.77381	0.891959	0.878378	0.958027	77527
	LR	0.941085	0.867133	0.867133	0.914642	0.867133	0.955193	1838
	XGBCL	0.924031	0.881119	0.797468	0.893832	0.863014	0.953185	5769
	RoBERTa	0.937984	0.902098	0.832258	0.912725	0.887208	0.961426	345297
	BERTIN	**0.992248**	**0.993007**	**0.972603**	**0.988852**	**0.988858**	**0.996245**	343318
	DistillBERT	0.962791	0.895105	0.934307	0.945262	0.90268	0.967027	217496

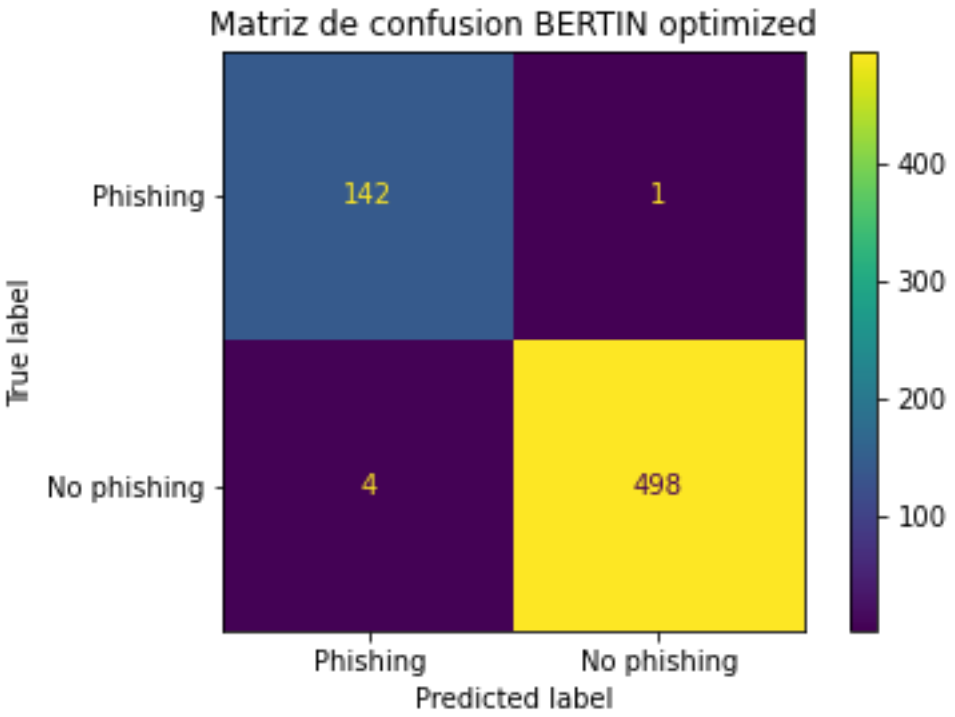


Figura 3.11 Matriz de confusión de BERTIN optimizado en test

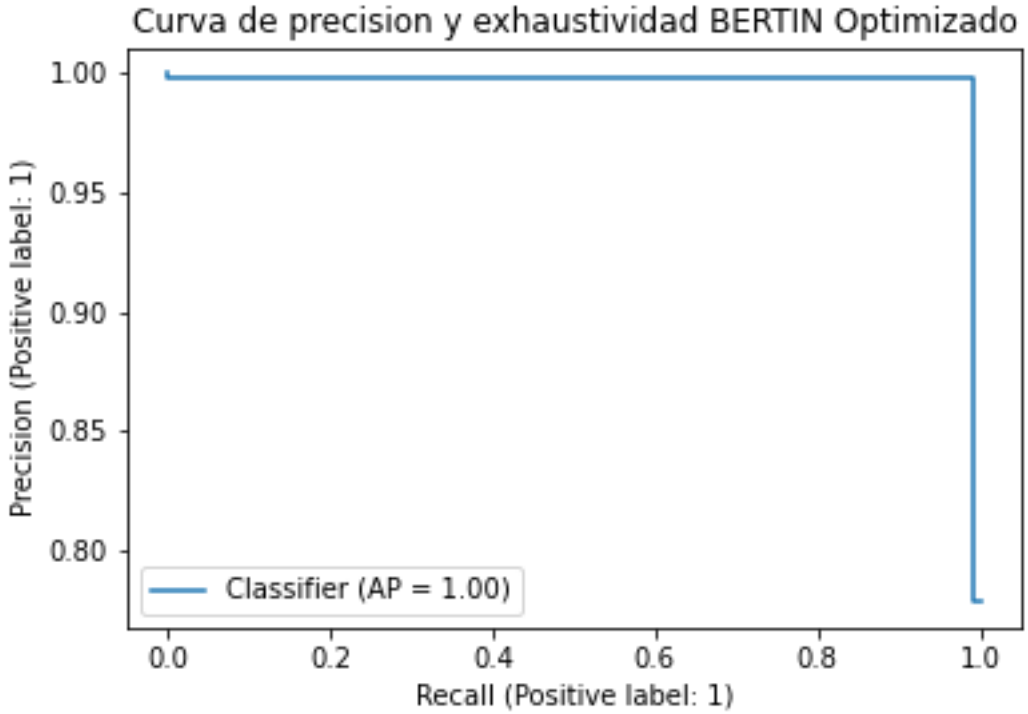


Figura 3.12 Curva de precisión y exhaustividad de BERTIN optimizado en test

3.5 Diseño e implementación de un pipeline de inferencia en tiempo real

La solución para inferencia en tiempo real se resume en tres etapas: procesamiento, inferencia y presentación las cuales se describen representan en la Figura 3.13:

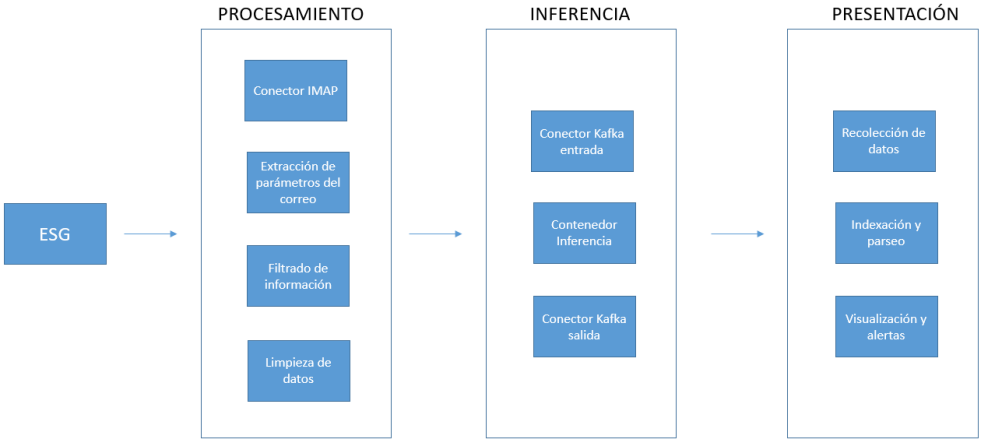


Figura 3.13 Arquitectura de inferencia en tiempo real

A continuación, se describe cada uno de los subprocesos ejecutados, es relevante recalcar que estos subprocesos serán usados de manera similar para el proceso de inferencia en tiempo real, descrito en una sección posterior.

1. **Procesamiento:** En esta etapa se realiza un tratamiento similar al usado para la creación del dataset como se detalla en la sección 3.1.1, con la diferencia de que los datos no serán almacenados sino reenviados como entrada al servidor de inferencia mediante Kafka, uno de los componentes basados en contenedores usado por la solución de NVIDIA Morpheus para la ingesta de datos.

2. **Inferencia:** NVIDIA Morpheus es un framework de inteligencia artificial optimizado para inferir en tiempo real, este fue instalado en un servidor DGX A-100 compatible con las liberarías que esta provee [9]. A nivel arquitectónico y como puntos relevantes de uso para nuestro proyecto, se dispone de una arquitectura de

principalmente de 2 componentes: Por un lado un mecanismo de comunicación de entrada y salida mediante 2 tópicos de Apache Kakfa referenciados en la Figura 3.13 como Contenedor Kafka entrada y Contenedor Kafka salida respectivamente y por otro lado existe un servidor Triton [26] diseñado para inferir en tiempo real y optimizado para el uso de GPU de la DGX A-100 el cual que recibe el modelo pre entrenado seleccionado en formato estándar onnx [26] que elegimos en el Capítulo 3. Finalmente, como resultado del proceso de inferencia, el flujo de datos es depositado en un conector Kafka de salida con el contenido entrante y agregando un campo adicional que representa si el correo entrante es phishing o no.

3. **Presentación:** El SIEM usado dispone herramientas para la lectura de los datos en el conector kafka de salida del servidor de inferencia. Esta información es procesada e indexada lo cual permite analizar los resultados así como representarlos gráficamente. Los resultados de la solución se presentan en un dashboard customizado que dispone de los siguientes indicadores de operación:

   Total: total de Emails procesados durante el periodo analizado
   Buenos: Emails detectados como NO Phishing
   Maliciosos: Emails detectados como Phishing
   Únicos buenos: Emails detectados como NO Phishing sin duplicados
   Únicos maliciosos: Emails detectados como Phishing sin duplicados
   Avg. Bueno por hora: Promedio de Emails NO phishing por hora
   Avg. Malicioso por hora: Promedio de Emails Phishing por hora

   También contamos filtros que permiten interactuar con la herramienta:

   Tiempo: Rango de tiempo a analizar
   Subject: Campo de búsqueda por texto en el campo subject del correo
   From: Campo de búsqueda por texto en el campo from del correo
   To: Campo de búsqueda por texto en el campo to del correo

Finalmente, una sección de resumen de información tabulada de los emails analizados y su resultado de inferencia:

Time: Fecha y hora

From: Contenido del campo from del correo

To: Contenido del campo to del correo

Subject: Contenido del campo subject del correo

Inferencia: Resultado de la inferencia “Bueno” y Malicioso

Figura 3.14 Dashboard de operaciones.

En la Figura 3.14 se muestra una vista del dashboard de operación de la herramienta con los campos previamente definidos. Esta representación gráfica puede ser ajustada según la necesidad del analista.

CAPÍTULO 4

4. ANÁLISIS DE RESULTADOS

4.1 Estrategia para la validación del proyecto

Se evaluó las estrategias de prueba backtesting, A/B testing y despliegue en la sombra (shadow testing) para validar el proyecto. Este proceso concluyó en la elección de despliegue en la sombra (shadow testing) por dificultades para implementar las otras técnicas y la compatibilidad con nuestra estrategia de simulación de ataques. Backtesting pierde relevancia en nuestro proyecto dado que más que un componente de estacionalidad existe un cambio de dominio frecuente; es decir, el desarrollo en los ataques tipo phishing quita valor al uso de datos históricos para validar el modelo. Del mismo modo, se descarta el uso de A/B testing dada la complejidad en el diseño de grupos de prueba y control, así como la asignación de muestras para su comparación en tiempo real manteniendo la misma distribución en ambos grupos con respecto a la variable de predicción si es o no phishing. Finalmente, el despliegue en la sombra nos permite tener el modelo en un ambiente de producción y evaluarlo con datos reales, pero sin afectar las operaciones de la compañía dado que el resultado de la herramienta no se va a integrar con las decisiones de la empresa sino hasta que se haya medido su impacto y costos asociados; además, el despliegue en la sombra facilita la integración con nuestra estrategia de alertas simuladas que se van a describir en el capítulo 4.3.

4.2 Métricas de evaluación de la solución.

Si bien en el campo científico las métricas usadas frecuentemente son las asociadas al rendimiento del modelo de machine learning, la empresa necesita indicadores asociados a costos para evaluar la puesta en producción y su eficiencia, por ello para la evaluación total de la solución consideramos 2 tipos de métricas, por un lado, las métricas científicas descritas en el capítulo 3 y por otro lado métricas organizacionales.

Tabla 4.1 Métricas de evaluación de la solución.

Tipo de métrica	Métrica
Científicas.	F β Score Exhaustividad
Organizacional.	Eficiencia de detección Costos operacionales

En la **¡Error! No se encuentra el origen de la referencia.** se muestra las métricas consideradas para evaluación del proyecto; los resultados serán presentados en la sección 4.4. A continuación se definen las métricas organizacionales seleccionadas:

Eficiencia de detección: Porcentaje de detecciones de email de phishing posterior e la evaluación de sus soluciones de seguridad perimetral (ESG).

Costos operacionales: Cantidad de investigadores de ciberseguridad necesarios para realizar el monitoreo (triaje) de las alertas de detección que se presentarían.

4.3 Simulación de alertas controladas

Dado que no controlamos los ataques en tiempo real que pueden llegar a la organización y sería un riesgo muy alto incitar a un atacante que lo haga; a fin de poder probar la solución, enviamos de manera controlada correos de tipo phishing que nos permita evaluar cómo se comporta nuestra solución en la arquitectura de defensa perimetral actual de la empresa. La empresa cuenta con 2 sistemas antispam en serie de diferentes marcas que ha decidido colocar para poder atenuar el riesgo de emails maliciosos entrantes, cabe recalcar que ninguno de estos, a pesar de que son tecnología de vanguardia, con buenas prácticas de configuración y revisadas por personal especializado en ciberseguridad, cuenta con mecanismos dedicados para la detección de emails tipo phishing mediante el análisis del cuerpo del correo. A continuación, los resultados de la prueba:

Tabla 4.2 Detección de Emails de phishing sin la solución propuesta

Detección	**Porcentaje**
Detectados por ESG 1	5,53 %
Detectados por ESG 2	2.27 %
No detectados por ESG que pasan a la Solución	91,70%
Total	100%

En la tabla anterior podemos ver que el 91.7 % de Emails de phishing que fueron puestos a prueba no fueron detenidos por los ESG y pasaron a ser evaluados por nuestra solución.

Tabla 4.3 Detección de Emails de phishing por la solución propuesta

Detección de la solución	**Porcentaje**
Maliciosos detectados	90.07 %
Maliciosos no detectados	9.93 %
Total	100%

En la tabla anterior podemos ver que el 90.07% de los emails no detectados por los ESG de la empresa fueron detectados por nuestra solución, si bien no llegamos los resultados de una exhaustividad del 97**%,** es de esperarse puesto algunos emails fueron detectados por los ESG producto de las técnicas de detección actuales implementadas por la empresa. Usaremos este valor para medir los costos operacionales métrica relevante para la empresa.

4.4 Monitoreo de la solución

Durante el transcurso de 2 semanas de monitoreo de nuestra solución con una arquitectura en paralelo a las operaciones actuales de la empresa (despliegue en la

sombra), pudimos estimar la cantidad de recursos que tendría que invertir la empresa para poder realizar el triage o revisión de las inferencias en tiempo real que alertaría nuestra solución. Se obtuvo una tasa de detección aproximada de 30 alertas de Emails Phishing por hora, donde tal como lo vimos en el capítulo 1, dependiendo del grado de madurez del ingeniero que va a analizar los resultados, podemos estimar que la empresa invertiría un mínimo de 2 horas al día y un máximo de 6 horas que representa 0.25 a 0.75 personas al día para la revisión de los resultados de la inferencia, confirmando que podemos ahorrar recursos para este propósito.

4.5 Evaluación final

4.5.1 Evaluación científica

La experimentación con modelos de Machine Learning y Deep Learning para NLP en la detección de phishing resultó en un modelo optimizado BERTIN con un F-beta score del 0.97 y PR-AUC del 0.93 con una mejora de 0.04 en F-beta score sobre el siguiente mejor modelo y 0.07 en PR-AUC. A pesar de tardar 6 segundos más que el modelo más rápido encontrado en evaluar los datos de entrenamiento, su versión optimizada para inferencia no registró problemas de latencia.

4.5.2 Evaluación organizacional

Luego del monitoreo de la solución considerando la arquitectura real de la empresa, se pudo comprobar que nuestra solución obtuvo una eficiencia de detección de emails tipo phishing en tiempo real, basado en el cuerpo del correo de un 90.7% y que aportaría en costos operacionales con una disminución de carga del 95.83% al poder reducir de 19 a 1 persona para realizar los potenciales riesgos asociados, estos indicadores permitieron a la empresa reevaluar su toma de decisión de un problema declarado ineficiente e impráctico a una inversión aceptable a considerar para desarrollo de su prototipo funcional.

4.5.3 **Aporte final**

La empresa que nos ayudó para la elaboración de este proyecto es líder en Telecomunicaciones y brinda servicios gestionados, entre ellos, de ciberseguridad y cuenta con infraestructura propia de inteligencia artificial. Para esta empresa, el proyecto ya ha sido categorizado como un proyecto de muy relevante tanto para uso interno como oferta de servicios dado que no existe solución similar en el mercado que aprovecha los recursos actualmente invertidos. Finalmente, este proyecto de investigación ha sido presentado por la empresa a clientes de diferentes industrias, quienes han dado una retroalimentación positiva tanto en el interés de su uso, así como en la imagen que proyecta la empresa al realizar este tipo de soluciones innovadores para la industria ecuatoriana lo que le permite la adopción de nuevos clientes y la fidelización de los actuales.

4.5.4 **Trabajo futuro**

Respecto al dataset, este puede ser mejorado estableciendo nuevos criterios para la construcción de emails de phishing en español que permitan nuevas estructuras de datos para el aprendizaje de comportamiento malicioso. Se puede buscar extender los escenarios de phishing reportados fuera del Ecuador, por ejemplo, la red de centros de respuesta a incidentes de FIRST consultando por emails de phishing reportados por diferentes países, especialmente los de Latinoamérica para así tener una base adicional de correos maliciosos que pueda afectar a la empresa, a pesar de que el contexto de otro país no sea igual al nuestro, podría servir de base para campañas similares.

Respecto al pipeline de inferencia, en este proyecto, dado que se dispone de recursos suficientes, no se tuvo la necesidad de evaluar la eficiencia y capacidad de uso; para un despliegue masivo y de servicios se podría evaluar los recursos mínimos necesarios para operar a un rendimiento (procesamiento en tiempo real) aceptable para la empresa, así como

contemplar variables de arquitectura como respaldos y contingencias para alta disponibilidad.

Así mismo, se puede explorar modelos más complejos como modelos de deep learning multimodales que permitan analizar no solo las características del texto del cuerpo del correo, sino también imágenes y metadatos de archivos adjuntos para obtener una solución que contemple un espectro más amplio de vulnerabilidades. Por último, investigación futura en operaciones de machine learning pueden evaluar mecanismos para detectar y responder ante cambios en los patrones de phishing, también se pueden explorar el diseño de modelos con soporte para online learning para responder a problemas de concept drift.

En el contexto operativo organizacional, se puede mejorar el pipeline de inferencia mejorando el filtrado de emails que van a ser enviados a inferencia, estableciendo una base con la empresa, evitar que correos de sistemas automatizados pasen por la solución así evitar gasto operacional en estas investigaciones. Adicionalmente se podría mejorar los tiempos invertidos en el triage (investigación de incidentes) si aplicamos filtros de duplicidad, esto es, evitar que correos con el mismo cuerpo, pero diferentes destinatarios sean tratados como independientes y no como una sola campaña de ataque.

Conclusiones

- Usando diferentes técnicas de aumentación, clasificación, y etiquetado de datos; se logró generar un dataset que permita el análisis y detección de correos tipo phishing redactados en el idioma español mediante modelos de machine learning y deep learning.
- Se evaluaron satisfactoriamente modelos de machine learning tradicionales y deep learning a fin de que permitan clasificar emails como phishing, seleccionando a modelos tipo BERT, en particular BERTIN como el mejor modelo que permite clasificar los correos que son phishing con un F-beta score del 0.97.
- Se logró elaborar y evaluar un pipeline de procesamiento e inferencia para detección de correos electrónicos tipo phishing en tiempo real con una arquitectura ajustable para el despliegue en una organización que se dedica a brindar servicios de tecnología corporativos. La simulación reveló que utilizando la solución propuesta en el presente trabajo permite disminuir la carga operacional de detectar los correos de phishing en un 95.83%.

BIBLIOGRAFÍA

[1] Palo Alto Networks, «PaloAlto Enterprise Security Solution,» 15 Febrero 2020. [En línea]. Available: https://www.slideshare.net/primeinfoserv/paloalto-enterprise-security-solution. [Último acceso: 15 11 2022].

[2] Cloudflare, «Cloudflare Area 1 Email Security,» Cloudflare, 2022. [En línea]. Available: https://www.cloudflare.com/products/zero-trust/email-security/. [Último acceso: 15 11 2022].

[3] Federal Bureau of Investigation FBI, «Business Email Compromise: The $43 Billion Scam,» Federal Bureau of Investigation FBI, 04 05 2022. [En línea]. Available: https://www.ic3.gov/Media/Y2022/PSA220504. [Último acceso: 16 11 2022].

[4] E. Kass, «State of Email Security 2022: Have Companies Begun Taking Cybersecurity More Seriously?,» Mimecast, 01 03 2022. [En línea]. Available: https://www.mimecast.com/blog/have-companies-begun-taking-cybersecurity-more-seriously/. [Último acceso: 16 11 2022].

[5] Gobierno de la República del Ecuador, «¡ALERTA! AUMENTO DE CAMPAÑAS DE PHISHING,» Gobierno de la República del Ecuador, 2022. [En línea]. Available: https://www.gobiernoelectronico.gob.ec/boletincampanasphishing/. [Último acceso: 16 11 2022].

[6] A. Turner, «The Future of the Email Security Market: The Importance of the Secure Email Gateway,» Cisco Blogs, 21 05 2020. [En línea]. Available: https://blogs.cisco.com/security/the-future-of-the-email-security-market-the-importance-of-the-secure-email-gateway. [Último acceso: 16 11 2022].

[7] Osterman Research, «How to Reduce the Risk of Phishing and Ransomware – White Paper,» Osterman Research, 03 2021. [En línea]. Available: https://ostermanresearch.com/2021/03/17/orwp_0336/. [Último acceso: 16 11 2022].

[8] GreatHorn, «The Death of the SEG (Magic Quadrant) and the Rise of Email Security,» GreatHorn, 19 06 2019. [En línea]. Available:

https://www.greathorn.com/blog/death-of-seg-magic-quadrant-the-rise-of-email-security/. [Último acceso: 17 11 2022].

[9] Nvidia, «NVIDIA Morpheus,» Nvidia, 201. [En línea]. Available: https://developer.nvidia.com/morpheus-cybersecurity. [Último acceso: 19 11 2022].

[10] A. K. S. a. N. Goyal, «Detection of Malicious Webpages Using Deep Learning,",» *IEEE International Conference on Big Data (Big Data),* pp. 3370-3379, 2021.

[11] Cisco, «Cisco Secure Email Phishing Defense,» 04 2021. [En línea]. Available: https://www.cisco.com/c/dam/en/us/products/collateral/security/cloud-email-security/at-a-glance-c45-740894.pdf. [Último acceso: 19 11 2022].

[12] Comodo, «The Best Open Source Anti Spam 2022,» Comodo, 2022. [En línea]. Available: https://www.comodo.com/home/email-security/best-open-source-anti-spam.php. [Último acceso: 20 11 2022].

[13] Sophos, «Sophos Email,» Sophos, 2022. [En línea]. Available: https://www.sophos.com/en-us/products/sophos-email. [Último acceso: 19 11 2022].

[14] S. &. K. D. &. P. P. Kotsiantis, «Handling imbalanced datasets: A review.,» *GESTS International Transactions on Computer Science and Engineering.,* nº 30, pp. 25-36, 2005.

[15] L. Y. J. S. G. M. H. Y. a. G. B. Guo Haixiang, «Learning from class-imbalanced data: review of methods and applications.,» *Expert Systems with Applications,* nº 73, p. 220–239, 2017.

[16] T. J. C. &. C. B. Gangavarapu, «Applicability of machine learning in spam and phishing email filtering: review and approaches.,» *Artif Intell,* nº 53, p. 5019–5081, 2020.

[17] K. K. a. C. X. Panagiotis Bountakas, «A Comparison of Natural Language Processing and Machine Learning Methods for Phishing Email Detection,» *Proceedings of the 16th International Conference on Availability, Reliability and Security (ARES 21),* nº 127, p. 1–12, 2021.

[18] I. A. a. Q. Yaseen, «Spam email detection using Deep Learning Techniques,» *Procedia Computer Science,* nº 184, p. 853–858, 2021.

[19] M.-W. C. K. L. a. K. T. Jacob Devlin, « BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,» *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* vol. 1, p. 4171–4186, 2019.

[20] J. &. L. C. Cao, «A Bilingual Multi-type Spam Detection Model Based on M-BERT,» *IEEE Global Communications Conference, GLOBECOM,* 2020.

[21] S. D. F. B.-M. A. C. V. A. José Cañete, «ALBETO and DistilBETO: Lightweight Spanish Language Models,» *Proceedings of the Thirteenth Language Resources and Evaluation Conference,* p. 4291–4298, 2022.

[22] Heavy.ia, «RAPIDS Definition,» Heavy.ia, 2022. [En línea]. Available: https://www.heavy.ai/technical-glossary/rapids. [Último acceso: 25 11 2022].

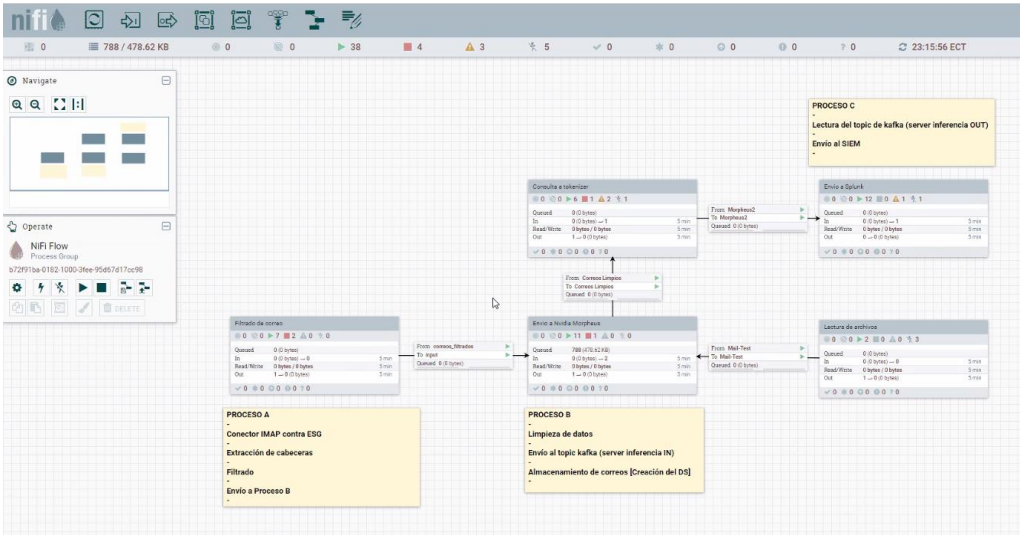
[23] G. Seif, «Here's how you can speedup pandas with cudf and gpus,» Medium, 11 02 2022. [En línea]. Available: https://towardsdatascience.com/heres-how-you-can-speedup-pandas-with-cudf-and-gpus-9ddc1716d5f2. [Último acceso: 25 11 2022].

[24] K. Tran, «rain your machine learning model 150X faster with cuml,» Medium, 04 11 2020. [En línea]. Available: https://towardsdatascience.com/train-your-machine-learning-model-150x-faster-with-cuml-69d0768a047a. [Último acceso: 25 11 2022].

[25] L. D. V. S. J. C. C. D. A. M. P. C. T. R. R. L. M. F. J. D. S. S. P. v. P. C. M. Y. J. J. P. C. X. Thomas Wolf, «Transformers: State-of-the-Art Natural Language Processing,» *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,* p. 38–45, 2020.

[26] Nvidia, «NVIDIA Triton Inference Server,» Nvidia, 2022. [En línea]. Available: https://developer.nvidia.com/nvidia-triton-inference-server. [Último acceso: 26 11 2022].

APÉNDICES

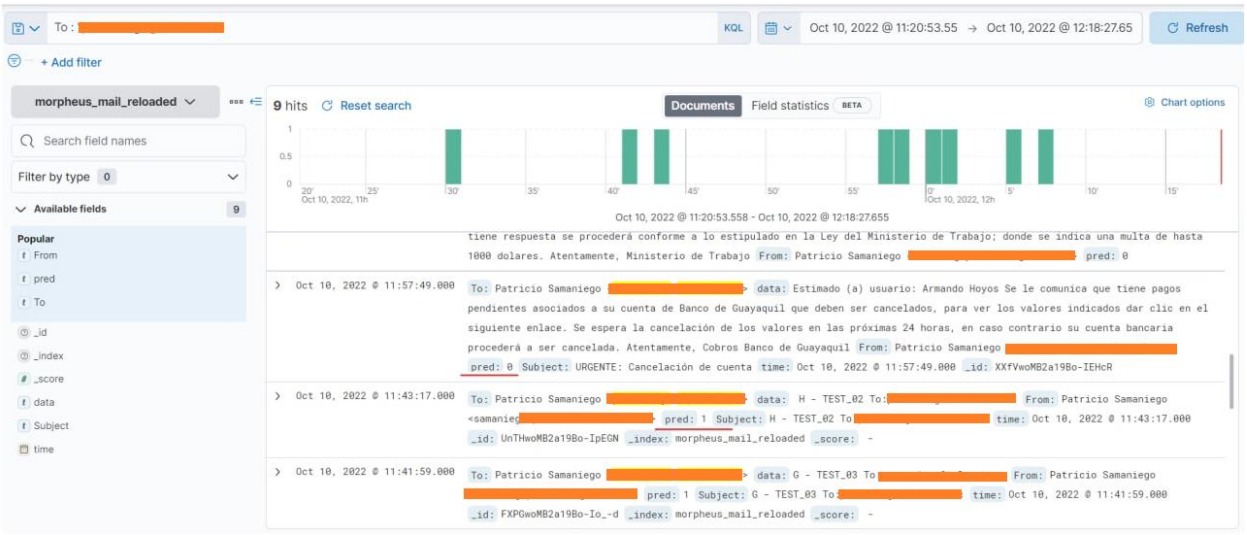


Captura de pantalla del ETL Nifi

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.103.01   Driver Version: 470.103.01   CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-SXM...  On   | 00000000:07:00.0 Off |                    0 |
| N/A   25C    P0    62W / 400W |  12557MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA A100-SXM...  On   | 00000000:0F:00.0 Off |                    0 |
| N/A   24C    P0    63W / 400W |   7026MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA A100-SXM...  On   | 00000000:47:00.0 Off |                    0 |
| N/A   24C    P0    62W / 400W |   7501MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   3  NVIDIA A100-SXM...  On   | 00000000:4E:00.0 Off |                    0 |
| N/A   25C    P0    63W / 400W |   7026MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   4  NVIDIA A100-SXM...  On   | 00000000:87:00.0 Off |                    0 |
| N/A   30C    P0    65W / 400W |   7026MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   5  NVIDIA A100-SXM...  On   | 00000000:90:00.0 Off |                    0 |
| N/A   29C    P0    67W / 400W |   7026MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   6  NVIDIA A100-SXM...  On   | 00000000:B7:00.0 Off |                    0 |
| N/A   29C    P0    65W / 400W |   7026MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   7  NVIDIA A100-SXM...  On   | 00000000:BD:00.0 Off |                    0 |
| N/A   29C    P0    63W / 400W |   6882MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A    560662      C   ...da/envs/rapids/bin/python      811MiB |
|    0   N/A  N/A    578547      C   ...da/envs/rapids/bin/python    11743MiB |
|    1   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    2   N/A  N/A     15401      C   tritonserver                      475MiB |
|    2   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    3   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    4   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    5   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    6   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     7023MiB |
|    7   N/A  N/A    578547      C   ...da/envs/rapids/bin/python     6879MiB |
+-----------------------------------------------------------------------------+
```

Captura de pantalla server inferencia NVIDIA A-100



Integración de solución de inferencia con SIEM ELK

```
====Building Pipeline Complete!====
FromFile Rate[Complete]: 25229messages [00:00, 42238.66messages/s]
                         kubectl logs sdk-cli-test1 -f
Configuring Pipeline via CLI ?inf/s]
Loaded labels file. Current labels: [['score', 'pred']]
Starting pipeline via CLI... Ctrl+C to Quit
Config:
{
  "ae": null,
  "class_labels": [
    "score",
    "pred"
  ],
  "debug": false,
  "edge_buffer_size": 4,
  "feature_length": 128,
  "fil": null,
  "log_config_file": null,
  "log_level": 10,
  "mode": "NLP",
  "model_max_batch_size": 32,
  "num_threads": 2,
  "pipeline_batch_size": 1024
}
CPP Enabled: True
====Registering Pipeline====
====Registering Pipeline Complete!====
====Starting Pipeline====
====Building Pipeline====
Added source: <from-file-0; FileSourceStage(filename=./data/email.jsonlines, iterative=False, file_type=FileTypes.Auto, repeat=1, filter_null=True, cudf_kwargs=None)>
  └─> morpheus.MessageMeta
Added stage: <monitor-1; MonitorStage(description=FromFile Rate, smoothing=0.001, unit=messages, delayed_start=False, determine_count_fn=None)>
  └─ morpheus.MessageMeta -> morpheus.MessageMeta
Added stage: <deserialize-2; DeserializeStage()>
  └─ morpheus.MessageMeta -> morpheus.MultiMessage
Added stage: <preprocess-nlp-3; PreprocessNLPStage(vocab_hash_file=./data/bert-base-uncased-hash.txt, truncation=True, do_lower_case=True, add_special_tokens=False, stride=-1)>
  └─ morpheus.MultiMessage -> morpheus.MultiInferenceNLPMessage
FromFile Rate: 0messages [00:00, ?messages/s]
Preprocess Rate: 0messages [00:00, ?messages/s]
Added stage: <monitor-4; MonitorStage(description=Preprocess Rate, smoothing=0.05, unit=messages, delayed_start=False, determine_count_fn=None)>
  └─ morpheus.MultiInferenceNLPMessage -> morpheus.MultiInferenceNLPMessage
Added stage: <inference-5; TritonInferenceStage(model_name=phishing-bert-onnx, server_url=ai-engine:8001, force_convert_inputs=True, use_shared_memory=False)>
  └─ morpheus.MultiInferenceNLPMessage -> morpheus.MultiResponseProbsMessage
Added stage: <monitor-6; MonitorStage(description=Inference Rate, smoothing=0.001, unit=inf, delayed_start=False, determine_count_fn=None)>
  └─ morpheus.MultiResponseProbsMessage -> morpheus.MultiResponseProbsMessage
Added stage: <add-class-7; AddClassificationsStage(threshold=0.7, labels=['pred'], prefix=)>
  └─ morpheus.MultiResponseProbsMessage -> morpheus.MultiResponseProbsMessage
Added stage: <serialize-8; SerializeStage(include=[], exclude=['^ID$', '^_ts_'], fixed_columns=True)>
  └─ morpheus.MultiResponseProbsMessage -> morpheus.MessageMeta
Added stage: <to-file-9; WriteToFileStage(filename=/common/data/output/phishing-bert-onnx-output.jsonlines, overwrite=True, file_type=FileTypes.Auto)>
  └─ morpheus.MessageMeta -> morpheus.MessageMeta
====Building Pipeline Complete!====
FromFile Rate[Complete]: 25229messages [00:00, 42238.66messages/s]
Preprocess Rate: 6144messages [00:19, 1584.18messages/s]
Inference Rate: 0inf [00:20, ?inf/s]
```

Pruebas de inferencia del modelo en NVIDIA