
# UTMACH

## FACULTAD DE INGENIERÍA CIVIL

## CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

# Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local

YUPANGUI CUENCA GILBER EHITEL
INGENIERO EN TECNOLOGIAS DE LA INFORMACION

ZAMBRANO LOPEZ ERICK ENRIQUE
INGENIERO EN TECNOLOGIAS DE LA INFORMACION

MACHALA
2024


# UTMACH

## FACULTAD DE INGENIERÍA CIVIL

## CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

# Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local

YUPANGUI CUENCA GILBER EHITEL
INGENIERO EN TECNOLOGIAS DE LA INFORMACION

ZAMBRANO LOPEZ ERICK ENRIQUE
INGENIERO EN TECNOLOGIAS DE LA INFORMACION

MACHALA
2024


# UTMACH

## FACULTAD DE INGENIERÍA CIVIL

## CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

**PROPUESTAS TECNOLÓGICAS**

# Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local

**YUPANGUI CUENCA GILBER EHITEL**
**INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**ZAMBRANO LOPEZ ERICK ENRIQUE**
**INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**MOROCHO ROMAN RODRIGO FERNANDO**

**MACHALA**
**2024**

# Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local

*by* ERICK ENRIQUE ZAMBRANO LÓPEZ

**Submission date:** 12-Aug-2024 01:04PM (UTC-0500)
**Submission ID:** 2422828065
**File name:** Tesis_Yupangui_Gilber_y_Zambrano_Erick-20240812.docx (6.13M)
**Word count:** 13155
**Character count:** 74346

# Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local

ORIGINALITY REPORT

| 9% | 8% | 1% | 3% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | upcommons.upc.edu<br>Internet Source | <1% |
| 2 | Submitted to Universidad Pablo de Olavide<br>Student Paper | <1% |
| 3 | www.researchgate.net<br>Internet Source | <1% |
| 4 | Submitted to Universidad Nacional Abierta y a Distancia, UNAD,UNAD<br>Student Paper | <1% |
| 5 | Submitted to Universitat Politècnica de València<br>Student Paper | <1% |
| 6 | qdoc.tips<br>Internet Source | <1% |
| 7 | issuu.com<br>Internet Source | <1% |
| 8 | repositorio.usmp.edu.pe<br>Internet Source | <1% |

| | | |
|---|---|---|
| 9 | Submitted to Instituto Superior de Artes, Ciencias y Comunicación IACC<br>Student Paper | <1% |
| 10 | catarina.udlap.mx<br>Internet Source | <1% |
| 11 | Submitted to Centro Europeo de Postgrado - CEUPE<br>Student Paper | <1% |
| 12 | Submitted to Universidad de Almeria<br>Student Paper | <1% |
| 13 | hdl.handle.net<br>Internet Source | <1% |
| 14 | Submitted to ITESM: Instituto Tecnologico y de Estudios Superiores de Monterrey<br>Student Paper | <1% |
| 15 | es.scribd.com<br>Internet Source | <1% |
| 16 | Submitted to Universidad de Alcalá<br>Student Paper | <1% |
| 17 | espanol.news<br>Internet Source | <1% |
| 18 | www.redeweb.com<br>Internet Source | <1% |
| 19 | de.slideshare.net<br>Internet Source | <1% |

| | | |
|---|---|---|
| 20 | www.dggomez.arrakis.es<br>Internet Source | <1 % |
| 21 | Perez-Alvarez, R. Perez-Sanagustin, M. Jorge J. Maldonado. "How to design tools for supporting self-regulated learning in MOOCs? Lessons learned from a literature review from 2008 to 2016", 2016 XLII Latin American Computing Conference (CLEI), 2016<br>Publication | <1 % |
| 22 | worldwidescience.org<br>Internet Source | <1 % |
| 23 | decsai.ugr.es<br>Internet Source | <1 % |
| 24 | tngconsultores.com<br>Internet Source | <1 % |
| 25 | Submitted to Universidad Francisco de Vitoria<br>Student Paper | <1 % |
| 26 | Submitted to<br>Student Paper | <1 % |
| 27 | Submitted to Editorial Elearning S.L.<br>Student Paper | <1 % |
| 28 | aws.amazon.com<br>Internet Source | <1 % |
| 29 | linkedpolitics.project.cwi.nl<br>Internet Source | <1 % |

| | | |
|---|---|---|
| 30 | Submitted to South Bank University<br>Student Paper | <1% |
| 31 | Submitted to Universidad Cesar Vallejo<br>Student Paper | <1% |
| 32 | Submitted to Universidad Nacional Mayor de San Marcos<br>Student Paper | <1% |
| 33 | tesisenred.net<br>Internet Source | <1% |
| 34 | Submitted to GIAC<br>Student Paper | <1% |
| 35 | Mick Cooper, Akira Ikemi. "Dialogue: A dialogue between focusing and relational perspectives", Person-Centered & Experiential Psychotherapies, 2012<br>Publication | <1% |
| 36 | Submitted to Universidad Catolica San Antonio de Murcia<br>Student Paper | <1% |
| 37 | Submitted to Universidad San Marcos<br>Student Paper | <1% |
| 38 | repositorio.yachaytech.edu.ec<br>Internet Source | <1% |
| 39 | bolsa-trabajo.upads.edu.pe<br>Internet Source | <1% |

| | | |
|---|---|---|
| 40 | catalonica.bnc.cat<br>Internet Source | <1 % |
| 41 | ceula.ie.edu<br>Internet Source | <1 % |
| 42 | es.10bet.com<br>Internet Source | <1 % |
| 43 | es.dmv.ca.gov<br>Internet Source | <1 % |
| 44 | patents.google.com<br>Internet Source | <1 % |
| 45 | www.isoftland.com<br>Internet Source | <1 % |
| 46 | Submitted to Universidad Europea de Madrid<br>Student Paper | <1 % |
| 47 | chiloeweb.com<br>Internet Source | <1 % |
| 48 | docs.microsoft.com<br>Internet Source | <1 % |
| 49 | ogpargentina2017.sched.com<br>Internet Source | <1 % |
| 50 | repositorio.unibague.edu.co<br>Internet Source | <1 % |
| 51 | rstudio-pubs-static.s3.amazonaws.com<br>Internet Source | <1 % |

| | | |
|---|---|---|
| 52 | www.ciena.com.mx<br>Internet Source | <1 % |
| 53 | www.mondragon.edu<br>Internet Source | <1 % |
| 54 | www.mql5.com<br>Internet Source | <1 % |
| 55 | www.powtoon.com<br>Internet Source | <1 % |
| 56 | Joaquin Salas, Flavio de Barros Vidal, Francisco Martinez-Trinidad. "Deep Learning: Current State", IEEE Latin America Transactions, 2019<br>Publication | <1 % |
| 57 | Submitted to Universidad Pontificia de Salamanca<br>Student Paper | <1 % |
| 58 | doaj.org<br>Internet Source | <1 % |
| 59 | journals.khnu.km.ua<br>Internet Source | <1 % |
| 60 | www.draytekonline.com<br>Internet Source | <1 % |
| 61 | www.old.uclg.org<br>Internet Source | <1 % |
| | www.tanatologia.org | |

| | | |
|---|---|---|
| 62 | Internet Source | <1% |
| 63 | www.winshuttle.es<br>Internet Source | <1% |
| 64 | zaguan.unizar.es<br>Internet Source | <1% |
| 65 | María Moncho Santonja. "Aplicación de técnicas de iluminación y procesado de imagen para la detección y medición de lesiones", Universitat Politecnica de Valencia, 2022<br>Publication | <1% |
| 66 | Submitted to Universidad Manuela Beltrán<br>Student Paper | <1% |
| 67 | Submitted to Universidad de Nebrija<br>Student Paper | <1% |
| 68 | bdigital.uncu.edu.ar<br>Internet Source | <1% |
| 69 | bibliotecadigital.udea.edu.co<br>Internet Source | <1% |
| 70 | digibuo.uniovi.es<br>Internet Source | <1% |
| 71 | doku.pub<br>Internet Source | <1% |
| | es.unionpedia.org | |

| | | |
|---|---|---|
| 72 | Internet Source | <1% |
| 73 | fdokumen.id<br>Internet Source | <1% |
| 74 | ghost.com<br>Internet Source | <1% |
| 75 | lorca.umh.es<br>Internet Source | <1% |
| 76 | news.sap.com<br>Internet Source | <1% |
| 77 | periodico.morelos.gob.mx<br>Internet Source | <1% |
| 78 | pesquisa.bvsalud.org<br>Internet Source | <1% |
| 79 | prezi.com<br>Internet Source | <1% |
| 80 | repositorio.espe.edu.ec<br>Internet Source | <1% |
| 81 | repositorio.usm.cl<br>Internet Source | <1% |
| 82 | revistas.uptc.edu.co<br>Internet Source | <1% |
| 83 | theblackboxlab.com<br>Internet Source | <1% |

| | | |
|---|---|---|
| 84 | theconversation.com<br>Internet Source | <1 % |
| 85 | www.coursehero.com<br>Internet Source | <1 % |
| 86 | www.info-ab.uclm.es<br>Internet Source | <1 % |
| 87 | www.mygreatlearning.com<br>Internet Source | <1 % |
| 88 | www.rsm.global<br>Internet Source | <1 % |
| 89 | www.toodledo.com<br>Internet Source | <1 % |
| 90 | lomejorensaviduria.blogspot.com<br>Internet Source | <1 % |
| 91 | www.cacic2016.unsl.edu.ar<br>Internet Source | <1 % |
| 92 | zombiecerebros.blogspot.com<br>Internet Source | <1 % |
| 93 | Константин Николаевич Цебренко, Рустам Рафикович Саакян, Александр Юрьевич Выхованец, Elizaveta Nicolaevna Voronina et al. "FUNDAMENTAL AND APPLIED SCIENTIFIC RESEARCH (ICFARS) December 2019: сб.науч.тр./ RELF Group&OEAPS Inc.; редкол.: Флора Бертран (отв.ред.) [и др.]. - | <1 % |

Берлин, Германия : OEAPS Inc., 2019. - 172 C. ISBN: 9781652107699", Open Science Framework, 2019

Publication

| | | | |
|---|---|---|---|
| Exclude quotes | Off | Exclude matches | Off |
| Exclude bibliography | On | | |

**CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL**

Los que suscriben, YUPANGUI CUENCA GILBER EHITEL y ZAMBRANO LOPEZ ERICK ENRIQUE, en calidad de autores del siguiente trabajo escrito titulado Aplicación de la inteligencia artificial en la detección de amenazas de Phishing en redes de área local, otorgan a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tienen potestad para otorgar los derechos contenidos en esta licencia.

Los autores declaran que el contenido que se publicará es de carácter académico y se enmarca en las disposiciones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

Los autores como garantes de la autoría de la obra y en relación a la misma, declaran que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asumen la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

YUPANGUI CUENCA GILBER EHITEL
0705715332

ZAMBRANO LOPEZ ERICK ENRIQUE
0706433109

# DEDICATORIA

A Dios, por su infinita guía y fortaleza en cada paso de este camino. A mis padres, Luz Cuenca y Gumercindo Yupangui, por su amor incondicional, apoyo constante y los sacrificios que han hecho para que pueda alcanzar mis sueños. A mi hermana Ligia Elena, cuya luz y espíritu siempre han sido una fuente de inspiración y fortaleza desde el cielo, acompañándome en cada momento de mi vida académica y motivándome a seguir adelante. Y, finalmente, a mi familia, que siempre ha confiado en mí y ha estado a mi lado en cada desafío, brindándome el aliento necesario para superar cada obstáculo.

**Yupangui Cuenca Gilber Ehitel**

Dedico este trabajo a mi familia, quienes han sido el pilar fundamental en la culminación de este logro. A mis padres, por su amor incondicional, apoyo constante y sabias enseñanzas. A mi hermana, por su compañía, comprensión y por estar siempre a mi lado. Su apoyo y palabras de aliento han sido invaluables durante este proceso.

Y a mis demás familiares, quienes de una u otra forma han contribuido con su cariño, consejos y apoyo. Su presencia en mi vida ha sido fundamental para alcanzar este importante objetivo.

**Zambrano López Erick Enrique**

# AGRADECIMIENTO

Quisiera expresar mi más sincero agradecimiento al Ing. Sist. Rodrigo Morocho, mi tutor de tesis, y al Ing. Sist. Wilmer Rivas, mi cotutor, por su orientación experta y apoyo constante a lo largo de este proceso. Un especial reconocimiento a la Ing. Bertha Mazón y a la Ing. Nancy Loja, quienes, como especialistas, brindaron recomendaciones valiosas que enriquecieron y mejoraron esta investigación. Finalmente, a mi familia y amigos, por su amor incondicional y apoyo constante durante todo este tiempo. Sin su aliento y confianza, este logro no habría sido posible.

**Yupangui Cuenca Gilber Ehitel**

En primer lugar, quiero expresar mi más sincero agradecimiento a mis padres, por su inquebrantable apoyo y constante motivación a conseguir mis metas, y a mi hermana por su comprensión y ánimo continuo. Agradezco también a mi tutor de tesis el Ing. Sist. Rodrigo Morocho, y cotutor el Ing. Sist. Wilmer Rivas, por su invaluable orientación a lo largo de todo el proceso de titulación. Agradezco profundamente a todas las personas que han compartido su sabiduría y guía a lo largo de estos años.

**Zambrano López Erick Enrique**

# RESUMEN

La seguridad de los sistemas informáticos en redes de área local (LAN) se enfrenta a un número cada vez mayor de desafíos como resultado de la creciente sofisticación de las amenazas cibernéticas. Este estudio aborda la problemática de la ineficacia de los sistemas tradicionales de detección de intrusos ante ataques avanzados, proponiendo una solución innovadora basada en el uso de inteligencia artificial (IA). El objetivo general de esta investigación es desarrollar un mecanismo basado en el aprendizaje automático para la detección de amenazas de phishing en redes locales. Este mecanismo se ha implementado y evaluado en un entorno simulado con GNS3. La metodología empleada, CRISP-DM, proporcionó un marco para el proceso, desde la comprensión inicial de los datos hasta la validación del modelo. Los hallazgos principales indican que el sistema de detección basado en IA supera notablemente las metodologías tradicionales, exhibiendo una alta precisión y exactitud en la identificación de amenazas, con métricas de evaluación que indican una precisión superior al 95%. Este avance no solo mejora la seguridad y la integridad de los datos en las LAN, sino que también establece un precedente para futuras investigaciones en ciberseguridad. Este trabajo tiene implicaciones significativas en múltiples niveles. Desde una perspectiva científica, contribuye al avance de las técnicas avanzadas de detección de amenazas. Desde un punto de vista tecnológico, ofrece una solución fiable y adaptable para proteger infraestructuras críticas. Desde una perspectiva económica, reduce los costes asociados a los incidentes de seguridad. En conclusión, esta investigación no solo propone una solución eficaz a un problema actual, sino que abre nuevas vías para la implementación de la IA en la ciber protección, consolidando su relevancia teórica y práctica en el ámbito de la seguridad informática.



**PALABRAS CLAVE**

Aprendizaje automático, detección de intrusos, Inteligencia artificial, redes de área local, sistemas de detección de intrusos.

# ABSTRACT

The security of computer systems in local area networks (LANs) is facing an increasing number of challenges as a result of the growing sophistication of cyber threats. This study addresses the issue of the inadequacy of traditional intrusion detection systems in the context of advanced attacks, proposing an innovative solution based on the use of artificial intelligence (AI). The overarching objective of this research is to develop a machine learning-based mechanism for the detection of phishing threats in local networks. This mechanism has been implemented and evaluated in a simulated environment with GNS3. The methodology employed, CRISP-DM, provided a framework for the process, from initial data understanding to model validation. The primary findings indicate that the AI-based detection system markedly outperforms traditional methodologies, exhibiting high accuracy and precision in threat identification, with evaluation metrics indicating greater than 95% accuracy. This breakthrough not only enhances security and data integrity in LANs, but also establishes a precedent for future cybersecurity research. This work has significant implications at multiple levels. From a scientific perspective, it contributes to the advancement of advanced threat detection techniques. From a technological standpoint, it offers a reliable and adaptable solution to protect critical infrastructures. From an economic perspective, it reduces the costs associated with security incidents. In conclusion, this research not only proposes an effective solution to a current problem, but also opens new avenues for the implementation of AI in cyber protection, consolidating its theoretical and practical relevance in the field of computer security.



**KEY WORDS**

Machine learning, intrusion detection, artificial intelligence, local area networks, intrusion detection systems.

# ÍNDICE DE CONTENIDO


DEDICATORIA....1
AGRADECIMIENTO....2
RESUMEN....3
ÍNDICE DE CONTENIDO....5
GLOSARIO....8
INTRODUCCIÓN....9
CAPÍTULO I. MARCO TEÓRICO....14
1.1 Antecedentes de la Investigación....14
1.2 Antecedentes históricos....16
1.3 Antecedentes Teóricos....17
1.3.1 Redes....17
1.3.2 Inteligencia artificial....19
1.3.3 Amenazas....21
1.4 Antecedentes Contextuales....21
1.4.1 Ámbito de aplicación....22
1.4.2 Establecimiento de requerimientos....23
CAPÍTULO II. DESARROLLO DEL PROTOTIPO....23
2.1 Definición del prototipo....23
2.2 Metodología de desarrollo del prototipo....23
2.2.1 Enfoque, alcance y diseño de investigación....23
2.2.2 Unidades de análisis....24
2.2.3 Técnicas e instrumentos de recopilación de datos....24
2.2.4 Técnicas de procesamiento de datos para la obtención de resultados....24
2.2.5 Metodología o métodos específicos....25
2.2.6 Herramientas y/o Materiales....26
2.3 Desarrollo del prototipo....26
2.3.1 Comprensión del negocio....26
2.3.2 Preparación de los datos....27
2.4 Ejecución del prototipo....32
2.4.1 Desarrollo de la topología....32
a) Descripción de la topología....32
b) Modelado....33
c) Implementación....36
CAPITULO III. EVALUACIÓN DEL PROTOTIPO....40
3.1 Plan de evaluación....40

3.1.1 Planificación ..... 40
3.1.2 Herramientas y Técnicas ..... 40
3.2 Evaluación ..... 42
3.3 Resultados de la evaluación ..... 51
CONCLUSIONES ..... 54
RECOMENDACIONES ..... 55
REFERENCIAS BIBLIOGRÁFICAS ..... 56
ANEXOS ..... 59

## ÍNDICE DE TABLAS

Tabla 1: Variables y dimensionamiento ..... 12
Tabla 2: Preguntas de investigación ..... 14
Tabla 3: Criterios de inclusión y exclusión ..... 15
Tabla 4: Herramientas y/o Materiales ..... 26
Tabla 5: Cronograma del plan de evaluación ..... 40
Tabla 6:Conograma detallado del plan de evaluación ..... 40
Tabla 7: Reuniones con tutor y cotutor. ..... 60

## ÍNDICE DE FIGURAS

Figura 1: Proceso y resultado de búsqueda ..... 15
Figura 2: Proceso y resultados de búsqueda ..... 16
Figura 3: Antecedentes Históricos ..... 16
Figura 4: Gráfico temático de antecedentes teóricos de protocolos ..... 17
Figura 5: Escenario de Simulación red LAN ..... 23
Figura 6: Etapa 1 ..... 27
Figura 7: Etapa 2 ..... 27
Figura 8: Diseño del prototipo ..... 27
Figura 9: Importar bibliotecas y el conjunto de datos ..... 28
Figura 10: Preprocesamiento - Eliminación de valores NaN ..... 29
Figura 11:Técnica de submuestreo y nuevo dataframe. ..... 29
Figura 12: Creación del modelo RandomForest, entrenamiento y cálculo de precisión ..... 30
Figura 13: Creación del modelo SVM, entrenamiento y cálculo de precisión ..... 30
Figura 14: Creación del modelo XGBoost, entrenamiento y cálculo de precisión ..... 30
Figura 15: Configuración y evaluación del modelo RandomForest ..... 31
Figura 16: Configuración y evaluación del modelo XGBoost ..... 31
Figura 17:Configuración y evaluación del modelo SVM ..... 32
Figura 18: Primera prueba de la topología ..... 32
Figura 19: Topología que se utilizó. ..... 33
Figura 20:Cargar nuevos correos y dividirlos. ..... 33
Figura 21:Evaluación del modelo RandomForest. ..... 34
Figura 22:Evaluación del modelo XGBoost ..... 34

Figura 23: Evaluación del modelo SVM ..... 35
Figura 24: Leer datos del segundo dataset. ..... 35
Figura 25: Evaluación del modelo RandomForest (dataset original) ..... 35
Figura 26: Evaluación de un modelo XGBoost (dataset original) ..... 36
Figura 27: Evaluación del modelo SVM (dataset original) ..... 36
Figura 28: Vista del servidor de correos. ..... 37
Figura 29: Puerto espejo capturando los correos. ..... 37
Figura 30: Leer y procesar cada dataset. ..... 37
Figura 31:Guardar el modelo entrenado. ..... 38
Figura 32: Extracción, limpieza y almacenamiento de los correos. ..... 38
Figura 33: Cargar el modelo de predicción. ..... 38
Figura 34: Función principal para procesar y clasificar correos ..... 39
Figura 35: Testeo del modelo con correos almacenados. ..... 39
Figura 36: Ejemplo grafico de la matriz de confusión. ..... 41
Figura 37: Ejemplo grafico de la curva ROC. ..... 41
Figura 38: Evaluación del rendimiento. ..... 42
Figura 39: Curva ROC, Comparativa del desempeño ..... 43
Figura 40: Matriz de confusión para cada modelo ..... 44
Figura 41: Métricas de evaluación ..... 45
Figura 42: Mejores parámetros (Dataset1) ..... 46
Figura 43:Curva ROC de cada modelo. ..... 47
Figura 44:Matriz de confusión para cada modelo. ..... 48
Figura 45: Mapa de Calor de los modelos. ..... 49
Figura 46: Mejores parámetros para cada modelo (dataset2) ..... 51
Figura 47: Resultados del modelo SVM. ..... 51
Figura 48: Métricas del modelo SVM. ..... 51
Figura 49: Curva ROC del modelo SVM ..... 52
Figura 50: Matriz de confusión del modelo SVM. ..... 52
Figura 51: Validación cruzada (SVM) ..... 53
Figura 52: Planteamiento de como seria el proyecto ..... 59
Figura 53: Explicación de los pasos a seguir. ..... 59
Figura 54: Revisión del modelo. ..... 60
Figura 55: Reunión con el tutor #1 ..... 60
Figura 56: Reunión con el tutor #2 ..... 60
Figura 57: Reunión con el cotutor #1 ..... 61
Figura 58: Reunión con el cotutor #2 ..... 61

# GLOSARIO

**D**

**Datos erróneos:** Son datos que contienen información inexacta, incorrecta o fuera de los valores esperados y válidos.

**C**

**Ciberamenazas:** son amenazas potenciales que explotan vulnerabilidades de sistemas informáticos o redes con el fin de causar daños o interrupciones.

**Curva ROC (Receiver Operating Characteristic):** Es una representación visual que muestra el rendimiento de un sistema de clasificación binario al cambiar su umbral de discriminación. Indica la tasa de verdaderos positivos en relación con la tasa de falsos positivos para distintos umbrales.

**M**

**Modelos de aprendizaje:** Son conjuntos de algoritmos y sistemas diseñados para aprender y mejorar su desempeño en tareas específicas mediante el análisis de datos y la experiencia.

**P**

**Proactivo:** Actuar anticipadamente para evitar problemas o situaciones no deseadas en el futuro, en lugar de reaccionar a ellas una vez que ocurren.

**Plataforma de emulación:** Es un software o sistema que permite imitar o replicar el comportamiento y funcionalidad de un dispositivo o entorno informático diferente en una máquina dada.

**S**

**Servidor de correo electrónico (email):** Es un sistema informático que gestiona el envío, recepción, almacenamiento y distribución de mensajes de correo electrónico a través de protocolos estándar como SMTP, POP3 e IMAP.

**Sistemas tradicionales:** Son sistemas de información basados en tecnologías y prácticas establecidas desde hace mucho tiempo, en contraposición a los sistemas modernos o emergentes.

# INTRODUCCIÓN

La seguridad cibernética de las redes locales se ve desafiada por la creciente amenaza de diversos ciberataques, planteando una necesidad crítica de soluciones innovadoras. La implementación de inteligencia artificial (IA) emerge como un enfoque prometedor para reforzar las defensas contra estos ataques, superando las limitaciones inherentes a los sistemas tradicionales de detección de intrusos.

Este estudio ofrece una visión integral del aumento de ciberamenazas en redes locales, involucrando investigaciones, recopilación de datos y desarrollo de simulaciones para identificar debilidades mediante herramientas de emulación. El objetivo principal es diseñar un mecanismo de detección de amenazas de Phishing utilizando inteligencia artificial en redes locales.

La evaluación de efectividad de las técnicas de inteligencia artificial en la detección de ciber amenazas, se llevó a cabo una simulación controlada utilizando la plataforma de emulación de redes GNS3. Se implementó una red de área local simulada y se ejecutó varios tipos de ataques, incluyendo phishing. Mediante la observación del comportamiento de la red y el análisis de los datos de tráfico generados durante estos ataques simulados, se evaluó la capacidad de los algoritmos de inteligencia artificial entrenados previamente para detectar anomalías y amenazas.

Los resultados obtenidos contribuirán significativamente al avance en el campo de la ciberseguridad, al analizar el potencial de la inteligencia artificial para identificar amenazas y anomalías de red, sentando las bases para futuros desarrollos en este crucial ámbito.

La estructura de este trabajo consta de tres capítulos fundamentales:

**Capítulo 1:** El primer capítulo se sumerge en la exploración y definición del marco teórico, proporcionando una base sólida para el entendimiento de conceptos clave y herramientas esenciales en el ámbito de la ciberseguridad y la inteligencia artificial. Se inicia con una revisión sistemática y crítica de literatura relevante, que permite no solo comprender los avances previos, sino también identificar las brechas que este estudio busca abordar. Se examinan los antecedentes de la investigación, distinguiendo entre los históricos y los teóricos, para establecer un contexto claro y justificar la necesidad de este proyecto. La discusión se extiende a los fundamentos de las redes de área local, la inteligencia artificial en sus diversas formas (aprendizaje automático), y las amenazas cibernéticas, identificando cómo estas áreas se interconectan y contribuyen al desarrollo de soluciones innovadoras en detección de intrusiones. Este capítulo no solo destaca la importancia crítica de la seguridad en las LAN, sino también cómo la implementación de soluciones basadas en IA puede marcar una diferencia significativa en la mitigación de riesgos y amenazas cibernéticas.

**Capítulo 2:** El segundo capítulo se centra en el diseño, desarrollo e implementación de un prototipo de sistema de detección de intrusiones basado en inteligencia artificial para redes de área local. Este segmento narra detalladamente el proceso de conceptualización del prototipo, desde la definición inicial hasta la metodología adoptada para su desarrollo, incluyendo el diseño de investigación, las técnicas de recopilación de datos, y las estrategias para el procesamiento de la información. Se detalla el uso de herramientas específicas y tecnologías avanzadas, como GNS3 para la simulación de

redes y el aprendizaje automático para el desarrollo del modelo de IA. Este capítulo ilustra cómo la integración de estas tecnologías facilita una comprensión profunda del comportamiento de la red y la identificación eficiente de amenazas de Phishing, subrayando la importancia de un enfoque multidisciplinario en la solución de problemas complejos de seguridad cibernética.

**Capítulo 3:** El último capítulo se dedica a la evaluación exhaustiva del prototipo desarrollado, detallando la metodología empleada para analizar su eficacia en la detección de amenazas de Phishing en redes de área local. Se describe un plan de evaluación riguroso, que incluye la definición de métricas clave como la precisión, para medir el rendimiento del sistema. Este análisis se complementa con una discusión sobre los resultados obtenidos, proporcionando una evaluación crítica del modelo de IA en términos de su capacidad para identificar amenazas de Phishing de manera efectiva. Se concluye con reflexiones sobre las implicaciones de estos hallazgos para la práctica y la investigación futura en el campo de la ciberseguridad, destacando las contribuciones del estudio al avance del conocimiento y la tecnología en la detección de intrusiones.

# I. Declaración y formulación del Problema

## Declaración del problema

La creciente expansión de las redes de área local ha llevado a un notable aumento en la cantidad y complejidad de los datos que circulan a través de ellas. Este incremento en la actividad de la red ha creado un entorno favorable para la proliferación de amenazas de phishing, que representan un riesgo constante para la seguridad de la información y la integridad de los sistemas informáticos. El aumento en el uso de Internet también extiende la superficie de ataque para ciberataques, lo que puede producir que la red sea mucho más vulnerable y ampliarse el riesgo de filtración o hackeo de datos. En este contexto, la detección de intrusiones se vuelve crucial para fortalecer la seguridad de Internet[1].

Ante la permanente evolución de las amenazas cibernéticas y la limitación de las soluciones habituales de detección de intrusiones, surge la interrogante sobre la viabilidad y efectividad de aplicar enfoques basados en inteligencia artificial para mejorar la seguridad de las redes de área local (LAN)[2]. En este sentido, la implementación de técnicas de IA, como el aprendizaje automático y el aprendizaje profundo, ha evidenciado ser prometedora en la identificación y mitigación de ataques. Permitiendo examinar grandes volúmenes de datos, identificando patrones y comportamientos anómalos que podrían revelar la presencia de un ataque cibernético [3].

Sin embargo, a pesar de los avances en el desarrollo de soluciones basadas en inteligencia artificial para la detección de amenazas de phishing, existen desafíos significativos que requieren atención. Por ejemplo, la evolución constante de los métodos de phishing y la sofisticación de los ataques representan un reto para la eficacia de los sistemas de detección basados en IA[3]. Investigaciones recientes han evidenciado que la aplicación de algoritmos de aprendizaje profundo en la detección de intrusiones puede optimizar significativamente la precisión y eficiencia en el reconocimiento de comportamientos anómalos en redes de área local [2].

Se propone analizar y desarrollar sistemas de localización de intrusiones basados en inteligencia artificial, empleando algoritmos de aprendizaje, para mejorar la seguridad de las redes de área local (LAN). También, se sugiere explorar en mayor profundidad las posibles barreras para la implementación efectiva de estos sistemas y proponer soluciones para superarlas.

## Formulación del problema

A continuación, se propone el siguiente problema de investigación:

- ¿Cómo puede la inteligencia artificial mejorar la seguridad cibernética en redes de área local?

## II. Objeto de estudio y Campo de acción

### Objeto de estudio

- La implementación de un sistema de inteligencia artificial en la detección de amenazas de Phishing en redes de área local.

### Campo de acción

- La aplicación de modelos de inteligencia artificial en la detección de amenazas de Phishing, con énfasis en la simulación de redes de área local y la evaluación del desempeño de dichos modelos.

## III. Objetivos

### Objetivo General

- Desarrollar un sistema de detección a amenazas de Phishing basado en inteligencia artificial para redes de área local emuladas con GNS3.

### Objetivos específicos

- Realizar un análisis del estado del arte de amenazas de Phishing en redes LAN
- Diseñar un escenario de simulación de red LAN con amenazas de Phishing.
- Implementar un modelo IA para la detección de amenazas de Phishing.
- Evaluar el desempeño del modelo de IA en la detección de amenazas de Phishing.

## IV. Hipótesis y variables o Preguntas de investigación

### Hipótesis principal

La implementación de un sistema de detección de ataques de Phishing basado en inteligencia artificial en una red de área local permitirá identificar la amenaza, con una tasa de detección superior al 80%.

### Variables y dimensionamiento (o categorización)

En la **tabla 1**, se especifica las variables, definición, categorías, indicadores e ítems a utilizar a lo largo de la investigación

Tabla 1: Variables y dimensionamiento

| Variable | Definición | Categorías | Indicadores | Ítems |
|---|---|---|---|---|
| Sistema de Detección de Intrusiones (IDS) construido sobre Inteligencia Artificial (Independiente) | Sistema que utiliza algoritmos de inteligencia artificial para identificar amenazas en una red de área local. | Algoritmos de aprendizaje supervisado, algoritmos de aprendizaje no supervisado. | Porcentaje de detección, porcentaje de falsos positivos, tiempo de respuesta. | Algoritmo de IA utilizado, parámetros de entrenamiento, métricas de desempeño. |
| Variable | Definición | Categorías | Indicadores | Ítems |
| Amenazas de Phishing | Ataques maliciosos que buscan | Conocidas, desconocidas. | Tipo de amenaza, frecuencia de | phishing |

| | | | | |
|---|---|---|---|---|
| (Independiente) | comprometer la seguridad de la red. | | ocurrencia, gravedad. | |
| Red de Área Local Emulada con GNS3 (Dependiente) | Simulación de una red de área local utilizando GNS3. | Topología de red, dispositivos emulados, tráfico de red. | Ancho de banda, latencia, número de dispositivos. | Topología de red, tipos de dispositivos emulados, tráfico de red simulado. |

## V. Justificación

Con el crecimiento constante de las amenazas cibernéticas y la sofisticación de los ataques, garantizar la seguridad en las redes de área local se ha vuelto cada vez más vulnerable. Las amenazas de Phishing representan grandes peligros de robo de datos, interrupción de operaciones y daños financieros y reputacionales. Las redes de área local, en especial, son muy sensibles ya que los atacantes pueden obtener acceso físico a la red interna.

Las soluciones de seguridad habituales, como los firewalls y los antivirus, tienen obstáculos para identificar nuevas amenazas y comportamientos anómalos. La inteligencia artificial propone nuevos espacios para analizar patrones de tráfico de red y comportamientos anómalos indicativos de ataques, lo que significaría mejoraras significativas en la detección a las amenazas de Phishing en las infraestructuras de redes de área local.

Asimismo, el empleo de técnicas de aprendizaje automático en la seguridad cibernética podría otorgar una protección más robusta y proactiva, lo que es trascendental para atenuar el impacto de los ataques cibernéticos en las organizaciones que dependen de redes informáticas.

# CAPÍTULO I. MARCO TEÓRICO

## 1.1 Antecedentes de la Investigación

La búsqueda se basa en la revisión sistemática de la literatura para reconocer, evaluar y sintetizar los resultados de estudios previos que puedan ser sobresalientes para la investigación presente. La finalidad es recopilar información relevante de manera eficiente y contribuir al avance de la investigación actual.

### a) Preguntas de investigación

La **tabla 2**, contiene las preguntas de investigación, descripciones y motivación con el cual se va a trabajar.

Tabla 2: Preguntas de investigación

| **Pregunta de investigación** | **Descripción y motivación** |
|---|---|
| ¿Qué tipos de sistemas de identificación de amenazas en ambientes de red LAN se han desarrollado mediante técnicas de inteligencia artificial? | Se analizarán los diferentes modelos de IA que se pueden emplear para la detección de intrusiones en LAN, como el aprendizaje automático, y la inteligencia artificial natural. |
| ¿Cómo se ha evolucionado el diseño de sistemas de detección de intrusiones a lo largo del tiempo en relación con los avances en la inteligencia artificial? | Se analizará la evolución del diseño de los IDS a lo largo del tiempo, desde los sistemas habitúales basados en reglas hasta los sistemas basados en IA. |
| **Pregunta de investigación** | **Descripción y motivación** |
| ¿Cuáles son los desafíos actuales en los sistemas de detección de intrusiones basados en inteligencia artificial y qué medidas pueden adoptarse para superarlos? | Se identificarán las limitaciones actuales de los sistemas basados en IA, como la necesidad de grandes valores de datos de entrenamiento, la complejidad para detectar amenazas nuevas y desconocidas, y la vulnerabilidad a los ataques de adversarios inteligentes. Se propondrán soluciones para superar estas limitaciones. |

### b) Palabras claves y Cadena(s) de búsqueda

La estrategia de búsqueda se enfocó en localizar investigaciones relevantes sobre la aplicación de técnicas de inteligencia artificial para la detección de intrusos en redes de área local. Se llevó a cabo una búsqueda automatizada en las principales bases de datos académicas y se complementó manualmente con artículos de alto impacto y ampliamente citados en el campo.

La cadena de investigación se construyó en base a los datos claves identificados en la pregunta de investigación, incluyendo: inteligencia artificial, detección de intrusos, redes de área local, aprendizaje automático, sistemas de detección de intrusos, redes de área local, ciberseguridad. Se probó iterativamente varias combinaciones a través de búsquedas de pruebas para refinar los términos a incluir.

Por último, se definió la cadena de búsqueda empleando los operadores booleanos para buscar el título, resumen y texto completo de los artículos:

("Inteligencia artificial" AND "detección de intrusos" AND "redes de área local")

("Aprendizaje automático" AND "sistemas de detección de intrusos")
Este planteamiento permitió identificar los estudios más significativos sobre el tema para realizar una comprobación sistemática de la literatura.

### c) Criterios de inclusión y exclusión

En la **tabla 3**, se detalla el criterio de inclusión y el criterio de exclusión que se utilizó para la recopilación de información

Tabla 3: Criterios de inclusión y exclusión

| # | **Criterio de inclusión** |
|---|---|
| **1** | Investigaciones que realicen pruebas entre protocolos de aplicación. |
| 2 | Estudios que contemplan redes de área local junto a los otros criterios planteados. |
| 3 | Estudios que se encuentran dentro de los 5 años de publicación. |
| 4 | Estudios de tipo artículo. |
| **#** | **Criterio de exclusión** |
| 1 | Artículos que no haya sido finalizados. |
| 2 | Estudios que su año de publicación exceda los 5 años. |
| 3 | Artículos que sean duplicados o repetidos. |
| 4 | Estudios que se encuentren escritos en idiomas diferentes al español o inglés. |

### d) Proceso y resultados de la búsqueda

La **figura 1**, detalla el proceso de búsqueda y las fases utilizadas para la misma.

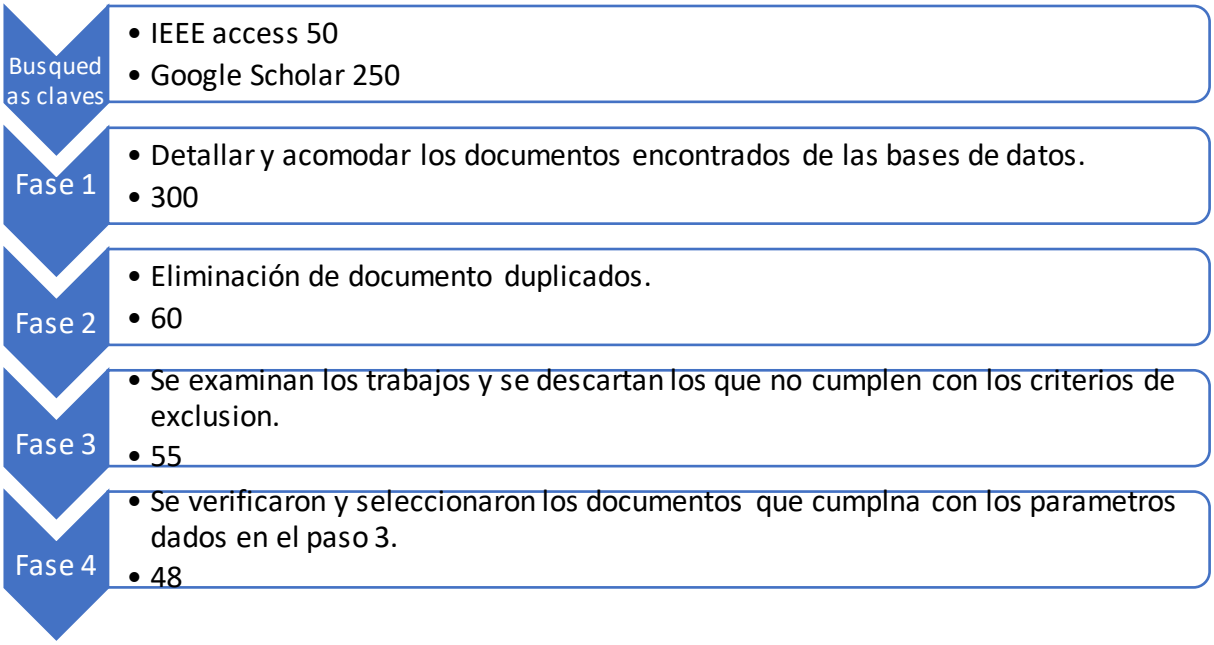


Figura 1: Proceso y resultado de búsqueda

En la **figura 1**, se muestran las etapas y los resultados de las búsquedas realizadas en las bases de datos académicas utilizadas en este proyecto. Se resalta el uso de Google Scholar e IEEE Xplore, donde inicialmente se identificaron numerosos artículos. Después de aplicar filtros de clasificación, se seleccionaron 48 artículos para un análisis detallado.

- Cantidad de estudios por año

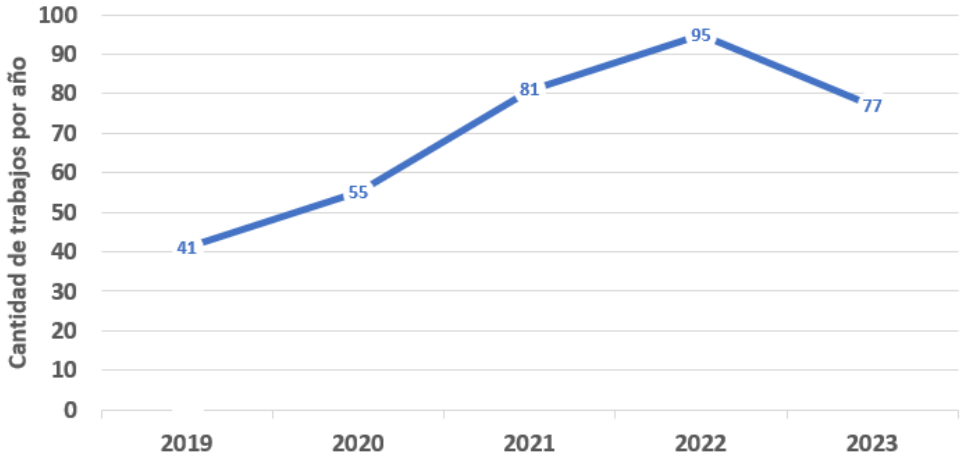


*Figura 2: Proceso y resultados de búsqueda*

De acuerdo a la **Figura 2**, se presenta la cantidad de artículos publicados en un periodo de tiempo específico, que luego fueron recopilados y analizados para ser potencialmente utilizados como referencias en este proyecto. La recolección de datos se realizó en el rango de años que va de 2019 a 2023, utilizando las bases de datos bibliográficas mencionadas previamente. Esto permitió identificar la literatura relevante sobre el tema publicada recientemente, para considerarla como evidencia en el desarrollo del proyecto actual.

## 1.2 Antecedentes históricos

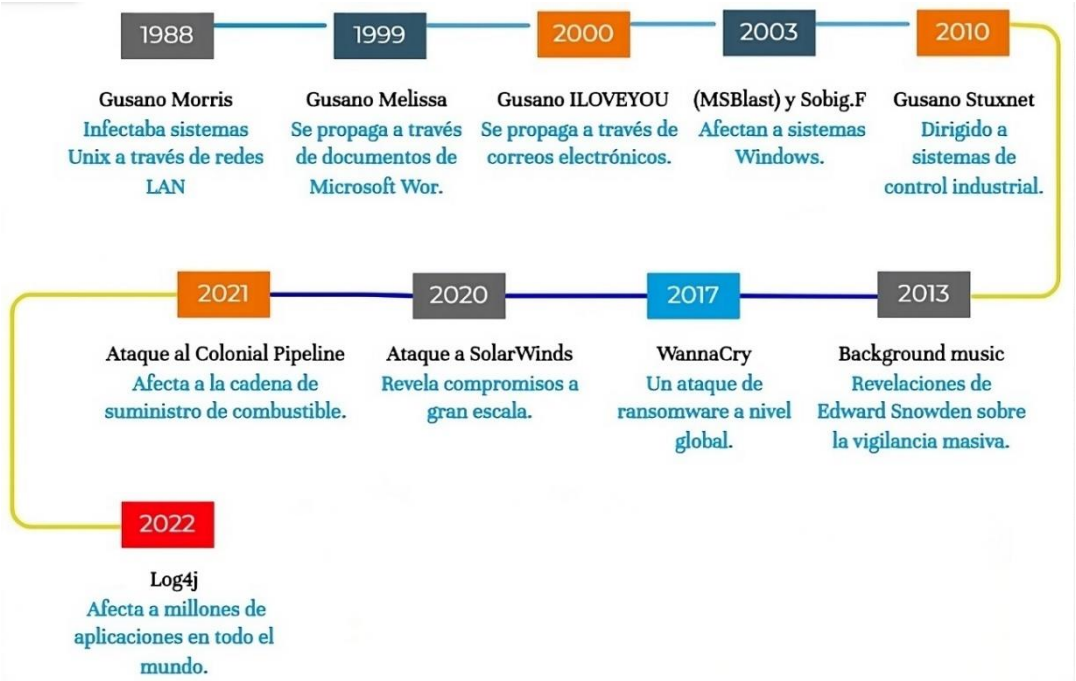


*Figura 3: Antecedentes Históricos*

**Fuentes:** [4], [5], [6], [7], [8], [9], [10], [11]

## 1.3 Antecedentes Teóricos

En la **figura 4** muestra los temas y subtemas que se van a tratar dentro del proyecto en el cual el tema de más interés y en el que más se basará el proyecto será sobre los protocolos a implementar.

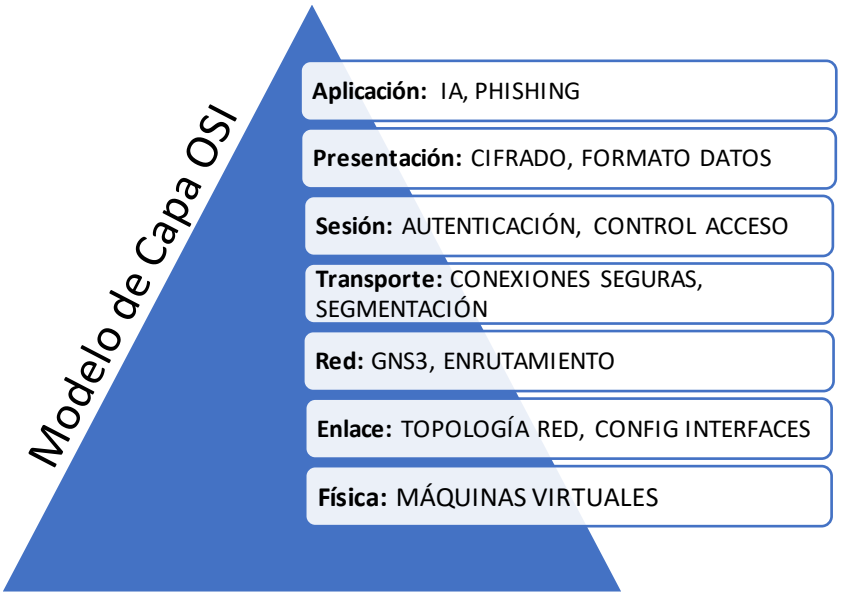


*Figura 4*: Gráfico temático de antecedentes teóricos de protocolos

A continuación, se mostrará los temas y subtemas que se van a tratar dentro del proyecto en el cual los temas de más interés y en el que más se basará el proyecto son las redes, inteligencia artificial y amenazas.

### 1.3.1 Redes

**Ethernet**

Es un estándar creado en 1976 por la empresa Xerox para tiempo después, en el cual permitía que la velocidad de transmisión sea de 10 Mbps. Posteriormente, se lo acopló para que sea compatible con el estándar IEEE 802.3 [12].

**Redes de área local**

Una Red de Área Local (LAN) se define como un conjunto de dispositivos interconectados ubicados en una zona geográfica específica, como una oficina, un edificio o un campus universitario. Estas redes facilitan la comunicación y el intercambio de datos entre los dispositivos conectados, que pueden incluir computadoras, impresoras, servidores y dispositivos de almacenamiento. Además, las LAN sirven como la infraestructura fundamental para la implementación de sistemas de seguridad y control de acceso a la información en el entorno organizativo [13].

**Máquina virtual**

Una máquina virtual se caracteriza por ser un entorno de software que emula las funciones de un sistema de computación físico, posibilitando la ejecución de un sistema operativo y aplicaciones de manera similar a como lo harían en un entorno físico autónomo. Este principio se fundamenta en la virtualización de servidores, que implica la consolidación de diversas aplicaciones y servicios de sistemas diversos en un mismo

hardware, de modo que tanto los usuarios como el sistema los perciban como máquinas independientes y dedicadas [14].

**GNS3**

Es un programa de código abierto ampliamente reconocido que se levanta como una herramienta fundamental para modelar estructuras de redes dentro de un entorno virtualizado. Su importancia radica en su modelado integral de procesos y soluciones de red, convirtiéndolo en el programa más adecuado para verificar la funcionalidad y seguridad de las infraestructuras de red propuestas [15].

Es capaz de ser ejecutado en Windows, OS y Linux, permitiendo la configuración de redes en gran variedad de equipos, en el que está incluido Cisco, puesto que GNS3 permite emular y simular tanto software como hardware en sistemas operativos de red reales.

**Oracle VM VirtualBox**

Fue desarrollado para la empresa Innotek, es un software de virtualización gratuito de Oracle, que permite la creación de una maquina aloje diferentes sistemas operativos por medio de la virtualización [12]. VirtualBox emplea un código conocido como "Guest Additions", el cual mejora significativamente la comunicación entre el sistema anfitrión y el sistema operativo invitado.

**Wireshark**

Es una herramienta esencial para el análisis y monitoreo de redes. Este software proporciona una variedad de funcionalidades, incluyendo el análisis en tiempo real de paquetes, la jerarquía de protocolos, el análisis de conversaciones y el análisis del tiempo de ida y vuelta. Estas características hacen de Wireshark una herramienta indispensable tanto para administradores de red, como para usuarios domésticos interesados en analizar el comportamiento y el rendimiento de sus redes [16].

**Tshark**

Se trata de una herramienta de línea de comandos que complementa a Wireshark en la tarea de analizar y supervisar redes. Al igual que Wireshark, Tshark permite la captura y el análisis de tráfico de red, pero se distingue por su capacidad de operar en entornos donde una interfaz gráfica no es viable o necesaria. Esta herramienta es especialmente útil para administradores de red y especialistas en seguridad que requieren un análisis detallado y automatizado del tráfico de red sin la sobrecarga de una interfaz gráfica [16].

**Hmailserver**

Es un servidor de correo electrónico de código abierto. Este servidor es compatible con los protocolos SMTP, POP3 e IMAP, lo que lo convierte en una opción versátil y robusta para la gestión del correo electrónico en entornos empresariales y académicos [17].

Una de las características destacadas de HMailServer es su capacidad para integrarse con motores de bases de datos como MySQL y MSSQL, lo que facilita el almacenamiento y la gestión de grandes volúmenes de datos de correo electrónico.

### 1.3.2 Inteligencia artificial

Es un campo de la informática en constante evolución que abarca desde la capacidad de aprendizaje automático hasta la simulación del comportamiento humano. Su aplicación se extiende a diversas áreas como marketing, ventas, logística, finanzas y atención al cliente [18].

La inteligencia artificial (IA) encuentra aplicaciones en diversos sectores, abarcando desde la ciberseguridad, donde se emplea para analizar extensas cantidades de datos, identificar intrusiones en redes y administrar vulnerabilidades, hasta asegurar la integridad de centros de datos. Sin embargo, la IA también presenta limitaciones, como su potencial para ser manipulada con fines delictivos, su complejidad y su alto costo [19].

**Python**

Python es un lenguaje de programación de alto nivel que ha sido utilizado para impulsar la detección amenaza en la red. También ha encontrado aplicación en el desarrollo de aplicaciones web, el desarrollo de software, la ciencia de datos y el aprendizaje automático (ML). Python se ha destacado como una herramienta altamente eficaz en la creación de sistemas informáticos que emplean diversas técnicas de inteligencia artificial, como las Redes de Análisis Paraconsistentes y las Redes Neurales Artificiales [20].

**Machine learning**

Es una rama de la inteligencia artificial que habilita a los sistemas computacionales a adquirir conocimiento y perfeccionarse de manera automática mediante la experiencia, sin requerir programación explícita para cada tarea específica. Esta capacidad de aprender de eventos pasados es fundamental para predecir y detectar amenazas en ciberseguridad [21].

**Librerías**

Las bibliotecas son conjuntos de módulos y funciones predefinidos que simplifican la realización de tareas específicas en programación. Estas bibliotecas contienen conjuntos de códigos reutilizados diseñados para abordar problemas comunes. Permitiendo ahorrar tiempo y esfuerzo al no tener que crear el código desde cero para cada función.

**Pandas:** Pandas es una biblioteca de código abierto para el lenguaje de programación Python, que ofrece estructuras de datos de alto rendimiento y herramientas avanzadas para el análisis de datos. Su valor se encuentra en su habilidad para gestionar conjuntos de datos de manera eficiente, facilitando a los científicos de datos y analistas la realización de operaciones complejas con facilidad. Además, Pandas proporciona una variedad extensa de funciones para el filtrado, agrupamiento y visualización de datos, consolidándose como una herramienta esencial para la exploración y comprensión de datos complejos [22].

**Numpy: E**s una biblioteca de programación de matrices poderosa y fundamental para Python, que permite realizar operaciones eficientes en matrices. Su importancia radica en su capacidad para proporcionar una sintaxis simple y expresiva que permite

manipular y operar con facilidad en matrices multidimensionales. Las matrices de NumPy almacenan datos de manera eficiente, incluyendo metadatos como el tipo de datos, la forma y los pasos. Además, la biblioteca ofrece soporte para indexación, operadores y funciones conscientes de las matrices para la programación de matrices de alto nivel [23].

**Matplotlib:** Es una biblioteca de visualización de datos en Python utilizada en entornos académicos y profesionales para representar gráficamente información de manera clara y efectiva. Tiene la capacidad para generar una amplia variedad de gráficos, desde simples diagramas de dispersión hasta complejas visualizaciones tridimensionales, lo que facilita la interpretación de datos y la comunicación de resultados de manera visualmente atractiva. Además, la capacidad de Matplotlib para integrarse con otras bibliotecas de Python, como NumPy y Pandas, amplía aún más su utilidad al permitir la visualización directa de datos almacenados en estructuras de datos comunes [24].

**Scikit-learn (sklearn):** Scikit-learn, también conocido como sklearn, es una biblioteca de código abierto para aprendizaje automático en Python que proporciona una amplia variedad de herramientas y algoritmos para análisis predictivo y minería de datos. Esta biblioteca es ampliamente utilizada en la comunidad académica y profesional debido a su facilidad de uso, eficiencia y versatilidad en la implementación de técnicas de aprendizaje automático. Sklearn ofrece una infraestructura robusta y consistente que facilita la creación, el entrenamiento y la evaluación de modelos predictivos, consolidándose como una herramienta esencial para la investigación y la aplicación de algoritmos de aprendizaje automático [25].

**Scikit-learn Pipeline:** Es una herramienta fundamental en el ámbito del aprendizaje automático, diseñada para facilitar y optimizar el proceso de desarrollo y evaluación de modelos. Según Streamline, un estudio reciente demostró que este pipeline de Python es capaz de abordar con eficacia tareas de clasificación binaria, ofreciendo una estructura transparente y estandarizada para comparar el rendimiento entre conjuntos de datos, algoritmos de aprendizaje automático y otras herramientas de AutoML [26].

**RandomForest**

El algoritmo RandomForest es una técnica de aprendizaje automático muy popular en el análisis de datos, conocida por su habilidad para gestionar grandes volúmenes de información y su resistencia a problemas de sobreajuste. Este método consiste en construir varios árboles de decisión durante el entrenamiento y combinar sus resultados para aumentar la precisión y estabilidad de las predicciones. Cada árbol en el conjunto se entrena con una muestra aleatoria del conjunto de datos original, y las predicciones finales se determinan a través de un proceso de votación o promedio de las salidas de todos los árboles individuales [27].

**XGBoost**

Conocido como "Extreme Gradient Boosting" o XGBoost, este es un sistema de boosting de árboles que destaca por su escalabilidad y eficiencia, siendo ampliamente adoptado

en el campo del aprendizaje automático. Su principal característica es su habilidad para gestionar grandes volúmenes de datos mediante una estructura de bloques en los árboles de decisión, lo que optimiza el tiempo de procesamiento y aumenta la eficiencia general [28].

**SVM**

El Support Vector Machine (SVM) es una técnica de aprendizaje supervisado muy popular en el campo de la inteligencia artificial y el aprendizaje automático. Este método se centra en la idea de maximizar el margen, que es la distancia entre el hiperplano separador y los puntos de datos más cercanos de cada clase, denominados vectores de soporte [29].

**Dataset**

Un dataset, en el contexto de la ciencia de datos y la inteligencia artificial, se refiere a un conjunto de datos que se utiliza para el análisis y la construcción de modelos predictivos. Según la Serie Científica de la Universidad de las Ciencias Informáticas [30], un dataset puede estar compuesto por una variedad de información, como variables, atributos, características o instancias, que se recopilan y organizan para su posterior procesamiento.

### 1.3.3 Amenazas

Las redes LAN se encuentran expuestas a diversas amenazas que comprometen la seguridad de los datos y la estabilidad de los sistemas informáticos. Investigaciones realizadas en el Cantón Pasaje indican que las empresas proveedoras de servicios de Internet no implementan sistemas de seguridad profunda en las redes LAN de manera plenamente eficiente, lo que ha propiciado ataques informáticos por medio de virus, troyanos, gusanos, y otros, comprometiendo la integridad de la información y representando una amenaza potencial para los sistemas informáticos [13].

**Phishing**

El phishing es una forma de ciberdelito que busca obtener información confidencial de los usuarios sin su consentimiento, a menudo a través de la suplantación de identidad de sitios web legítimos. Este tipo de ataques pueden manifestarse a través de diversas técnicas, como el phishing por correo electrónico, mensajes de texto, llamadas telefónicas o sitios web fraudulentos [31].

El phishing representa un riesgo significativo para la seguridad de las personas, ya que puede resultar en la pérdida de datos personales o financieros. Por ejemplo, se han reportado casos de phishing que han afectado a servicios como PayPal, donde los usuarios han sido engañados para proporcionar sus credenciales de inicio de sesión [32].

## 1.4 Antecedentes Contextuales

La incorporación de inteligencia artificial (IA) en la identificación de amenazas de Phishing en el ámbito de redes locales (LAN) representa un área de investigación en expansión con un gran potencial para mejorar la seguridad en estas redes. No obstante, la creación de sistemas de IA efectivos para la detección de amenazas de Phishing demanda un procesamiento y análisis de datos exhaustivo.

En el presente estudio, se llevarán a cabo técnicas de procesamiento y análisis de datos en un laboratorio de redes de la Universidad Técnica de Machala, Facultad de Ingeniería Civil, Ecuador. El contexto de la investigación será una simulación de ataques cibernéticos realizada con el software GNS3. Para los ataques, se utilizarán datasets que contengan información sobre phishing.

La recopilación de datos para este estudio se llevará a cabo a partir de fuentes tanto públicas como privadas. Los datos de carácter público abarcarán registros de ataques cibernéticos que han sido difundidos por entidades reconocidas, entre ellas el Centro de Coordinación de Respuesta a Incidentes de Seguridad Informática (ECUCERT) y la Agencia Europea de Seguridad de las Redes y de la Información (ENISA). Por otro lado, se incluirán datos privados provenientes de registros de ataques cibernéticos proporcionados de manera específica por empresas y organizaciones de carácter privado.

El procesamiento de los datos se realizará utilizando técnicas de análisis estadístico y aprendizaje automático. Las técnicas de análisis estadístico se utilizarán para identificar patrones y tendencias en los datos. Las técnicas de aprendizaje automático se utilizarán para entrenar un modelo de IA que pueda detectar amenazas de Phishing.

Las métricas de rendimiento del modelo de IA que se evaluarán incluyen precisión, sensibilidad y especificidad. La precisión mide la proporción de ataques que el modelo detecta correctamente. La sensibilidad mide la proporción de ataques reales que el modelo detecta. La especificidad mide la proporción de tráfico legítimo que el modelo no detecta como un ataque.

Según el informe de 2023 del ECUCERT, Ecuador registró un total de 1.200 ataques cibernéticos en el año 2022. De estos ataques, el 50% fueron ataques de phishing, el 30% fueron ataques de malware y el 20% fueron ataques de DoS.

El informe también señala que los ataques cibernéticos en Ecuador son cada vez más sofisticados. Los atacantes están utilizando técnicas más avanzadas para evadir los sistemas de seguridad tradicionales.

El escenario simulado para el estudio será una topología de red de LAN, los ataques cibernéticos se simularán utilizando el software GNS3. El modelo de IA se entrenará para detectar los ataques cibernéticos que se produzcan en la topología de red simulada.

### 1.4.1 Ámbito de aplicación

En el contexto de la seguridad de redes, se presentan diversos riesgos que abarcan desde ataques maliciosos hasta configuraciones incorrectas de la infraestructura y dispositivos conectados. Incluso interrupciones simples pueden tener repercusiones graves para las empresas, dado que la mayoría de las actividades dependen de la conectividad y los servicios de red. Para mitigar estas amenazas, se adoptan estrategias variadas, tales como integrar medidas de seguridad desde la fase de diseño de las redes, fortaleciendo así su seguridad intrínseca y resistencia. Además, se recurre a la implementación de dispositivos como firewalls y sistemas de detección de intrusos, con el propósito de elevar los niveles de seguridad [33].

Se propone analizar y explicar los principales problemas de seguridad que pueden presentarse en la red debido a la gravedad de esta situación. Para lograr esto, primero se recopilan datos y luego se llevan a cabo un conjunto de pruebas simuladas sobre una

topología de red en GNS3 para descubrir posibles vulnerabilidades y métodos de acceso no autorizado.

### 1.4.2 Establecimiento de requerimientos

Cada día, las empresas se enfrentan a amenazas que ponen en peligro los tres pilares de la empresa: la confidencialidad, la integridad y la capacidad. Estos peligros pueden ser externos o incluso más peligrosos dentro de la empresa. Es esencial priorizar la protección de la información, lo que ha llevado a la creación de diversas normas, leyes e incluso metodologías y prácticas para que las empresas puedan establecer sus propias políticas, procesos y procedimientos. Sin embargo, es importante tener en cuenta que se está en una situación vulnerable, ya que es bien conocido que no existe una seguridad completa y que en ocasiones no se sabe cuándo podrían surgir estos riesgos [34].

Es necesario realizar pruebas de penetración a la red (pentesting) para preparar una red a ataques a sistemas y datos en servidores. Para reducir los riesgos y tener los controles adecuados, es recomendable contratar personal especializado en la detección de vulnerabilidades.

# CAPÍTULO II. DESARROLLO DEL PROTOTIPO

## 2.1 Definición del prototipo

El escenario de este trabajo es una red LAN simulada en GNS3 que está construido con dos máquinas virtuales (w10_1, w7_1) que serán los clientes y los blancos de los ataques, tenemos una máquina que tendrá el modelo de detección (W10_2-1) para analizar los paquetes que lleguen al servidor, tenemos una máquina que será el servidor donde se reciban los correos, y la maquina atacante (Kali-linux_1). En la **Figura 5** está la topología utilizada.

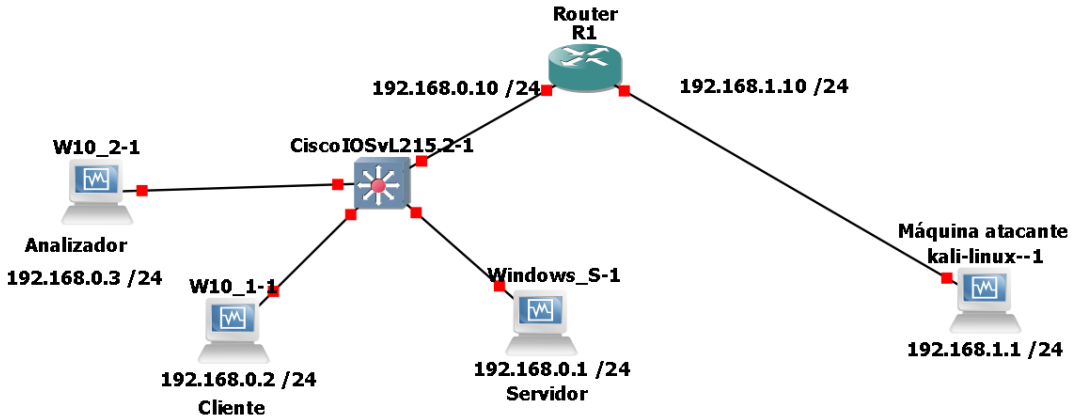


Figura 5: Escenario de Simulación red LAN

## 2.2 Metodología de desarrollo del prototipo

### 2.2.1 Enfoque, alcance y diseño de investigación

La metodología se centrará en una investigación de índole cuantitativa, resaltando la recopilación y análisis de datos como elementos clave para evaluar la eficacia del sistema de inteligencia artificial. Este enfoque posibilitará el examen detenido del desempeño del modelo en la detección de amenazas dentro de entornos específicos de redes de área local.

El objetivo principal radica en afrontar el desafío inherente de mejorar la identificación de amenazas de Phishing en redes de área local mediante el desarrollo de un sistema fundamentado en inteligencia artificial. Este sistema será implementado y evaluado en entornos simulados utilizando GNS3. Se enfoca en evaluar la eficacia del sistema para identificar amenazas en redes locales, con una meta de detección superior al 90%. El estudio abarcará el diseño de escenarios de simulación, la implementación de modelos de inteligencia artificial, y la evaluación del rendimiento, contribuyendo al avance en la aplicación de la inteligencia artificial en la seguridad cibernética en entornos de redes LAN.

La investigación empleará un diseño cuasiexperimental, enfocándose en la creación de escenarios de simulación de amenazas de Phishing en redes de área local, implementando un modelo de inteligencia artificial adaptado al entorno simulado, para el análisis del conjunto de datos generados en los escenarios, analizando las fortalezas y debilidades del modelo en diversas situaciones.

### 2.2.2 Unidades de análisis

La unidad de análisis está centrada en la detección de amenazas de Phishing en redes de área local (LAN). Se emplearán diversas unidades de análisis y técnicas claves. La simulación en GNS3 se utilizará para recopilar el comportamiento de la red y evaluar amenazas potenciales. La recopilación de dataset variados, que incluyen incidentes y comportamientos inusuales, es esencial. Además, se aplican técnicas de inteligencia artificial, como el aprendizaje automático, para analizar el comportamiento de la red. Se aplicará la técnica de observación y experimentación en el entorno simulado.

### 2.2.3 Técnicas e instrumentos de recopilación de datos

La técnica a utilizar será de observación que consiste en supervisar el comportamiento de un sistema para obtener información sobre el mismo. En este caso, la observación se llevará a cabo en una red de área local (LAN) simulada en GNS3, utilizando datos obtenidos de un dataset para identificar patrones de tráfico que puedan indicar la presencia de una amenaza. Los datos recopilados mediante la técnica de observación serán primero analizados utilizando técnicas de inteligencia artificial para clasificarlos y procesarlos. Posteriormente, los datos analizados se emplearán para entrenar un modelo de aprendizaje automático diseñado para identificar amenazas de phishing.

### 2.2.4 Técnicas de procesamiento de datos para la obtención de resultados

En relación con el enfoque cuantitativo y el alcance cuasiexperimental de la investigación, se implementarán técnicas de procesamiento y análisis de datos orientadas a evaluar el rendimiento del sistema de inteligencia artificial en la detección de amenazas de Phishing en redes de área local simuladas mediante GNS3.

La recopilación de datos se llevará a cabo mediante la creación de escenarios de simulación de amenazas de Phishing en redes de área local, utilizando un diseño cuasiexperimental. La metodología se centrará en la implementación de modelos de inteligencia artificial adaptados al entorno simulado, aplicando algoritmos de machine learning para la clasificación de amenazas con base en el conjunto de datos.

Se realizará un análisis estadístico detallado y la evaluación del rendimiento del sistema jugará un papel crucial para determinar las fortalezas y debilidades del modelo en la detección de amenazas. La aplicación de validación cruzada permitirá estimar métricas de desempeño como precisión, recall y la curva ROC en entornos diversos, con el objetivo de mejorar la efectividad global del modelo, apuntando a un rendimiento superior al 90%.

### 2.2.5 Metodología o métodos específicos

#### Metodología CRISP-DM

La metodología CRISP-DM (Cross-Industry Standard Process for Data Mining) es un modelo muy utilizado en el campo de la ciencia de datos. Este proporciona una estructura sistemática y secuencial para llevar a cabo proyectos de ciencia de datos, desde la comprensión del problema hasta la implementación de la solución [35].
La metodología CRISP-DM se divide en seis fases:

1. **Comprensión del negocio:** En esta fase, se debe comprender el problema que se desea resolver con el proyecto de ciencia de datos. Se deben identificar los objetivos del proyecto, los requisitos de los usuarios y los datos disponibles.
2. **Preparación de los datos:** En esta fase, se deben preparar los datos para su análisis. Esto incluye la limpieza de los datos, la selección de las características relevantes y la transformación de los datos para que sean compatibles con los algoritmos de aprendizaje automático.
3. **Modelado:** En esta fase, se seleccionan los algoritmos que se aplicarán para el proyecto. Se deben entrenar los modelos con los datos preparados y evaluar su rendimiento.
4. **Evaluación:** En esta fase, se evalúa el rendimiento de los modelos utilizados. Se deben considerar los siguientes factores:
    a. Exactitud: ¿El modelo es capaz de clasificar correctamente los datos?
    b. Relevancia: ¿El modelo es capaz de identificar las características relevantes?
    c. Explicabilidad: ¿El modelo es capaz de explicar sus decisiones?
5. **Implementación:** En esta fase, se implementa la solución de ciencia de datos. Esto incluye la integración de los modelos de aprendizaje automático con la infraestructura existente.
6. **Operación y mantenimiento:** En esta fase, se opera y mantiene la solución de ciencia de datos. Esto incluye la supervisión del rendimiento de los modelos, la actualización de los modelos y la gestión de los datos.

#### Metodología PPDIOO

La metodología PPDIOO es un enfoque estructurado y sistemático utilizado en el ámbito de las redes empresariales para guiar el ciclo de vida de una red. Este método, basado en los estándares internacionales de Cisco, se compone de seis fases claramente definidas: Preparar, Planear, Diseñar, Implementar, Operar y Optimizar. Cada una de estas etapas juega un papel fundamental en el proceso de gestión y administración de una red empresarial, permitiendo una planificación efectiva, una implementación adecuada, y una operación y optimización continuas [36].

Fases de la metodología PPDIOO:

1) **Preparación:** Definición de objetivos y requisitos del sistema.
2) **Planear:** Desarrollo de un plan detallado con recursos y cronograma.
3) **Diseñar:** Creación de un diseño técnico que incluye arquitectura y componentes.
4) **Implementar:** Construcción e instalación del sistema según el diseño.
5) **Operar:** Puesta en marcha y administración diaria del sistema.
6) **Optimización:** Mejora continua mediante ajustes y análisis de rendimiento.

### 2.2.6 Herramientas y/o Materiales

Las herramientas clave incluirán GNS3 para la simulación de redes, bibliotecas de inteligencia artificial como TensorFlow, Kaggle o PyTorch para la implementación del modelo, y herramientas de análisis de datos como Python y sus librerías. Además, se utilizarán herramientas de visualización de datos para representar gráficamente los resultados obtenidos durante la evaluación del desempeño del modelo.

La **tabla 4**, se especifica las herramientas y/o materiales que se utilizaran a lo largo de la investigación.

Tabla 4: Herramientas y/o Materiales

| Categoría | Herramientas y/o materiales |
|---|---|
| Software | • GNS3<br>• Oracle VM VirtualBox<br>• Windows Server |
| Hardware | • Laptop (GPU Intel Core i7 10ma gen, 16GB Ram, 1Tb de almacenamiento)<br>• Laptop (Dell i5 7ma gen, 960 disco sólido SD, 16 ram dr4) |
| Datos | Bibliotecas de dataset para IA (TensorFlow, Kaggle o PyTorch) |

## 2.3 Desarrollo del prototipo

### 2.3.1 Comprensión del negocio

El prototipo se divide en 2 etapas. En la etapa 1 (**Figura6**) nos enfocamos en buscar la data que utilizaríamos como punto de referencia para la implementación de nuevos modelos y la creación de nuevos datos (Phishing_Email.csv [27]) para así evaluar la eficacia de los modelos de detección. Esta información se utilizará para desarrollar la etapa 2 (**Figura 7**), que consiste en crear la topología en la que pasarán los correos tanto buenos como malos y estos serán capturados en una computadora que este configurada con un puerto espejo, y que además se planea implementar el modelo de clasificación. Finalmente, en la **figura 8** vemos toda la idea del prototipo ya con los dos partes unidas

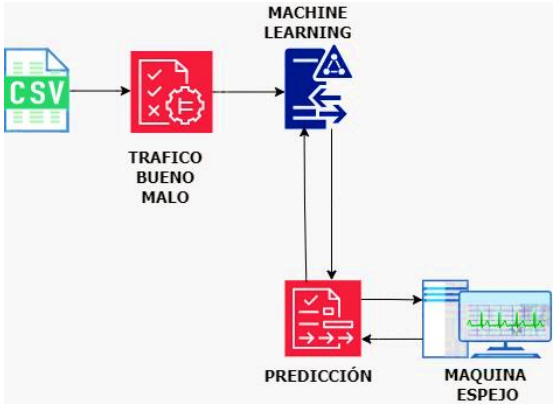


Figura 6: Etapa 1

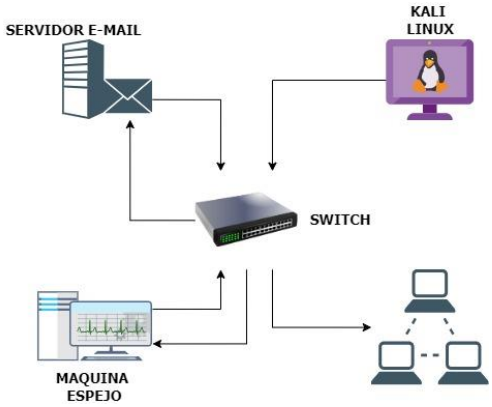


Figura 7: Etapa 2

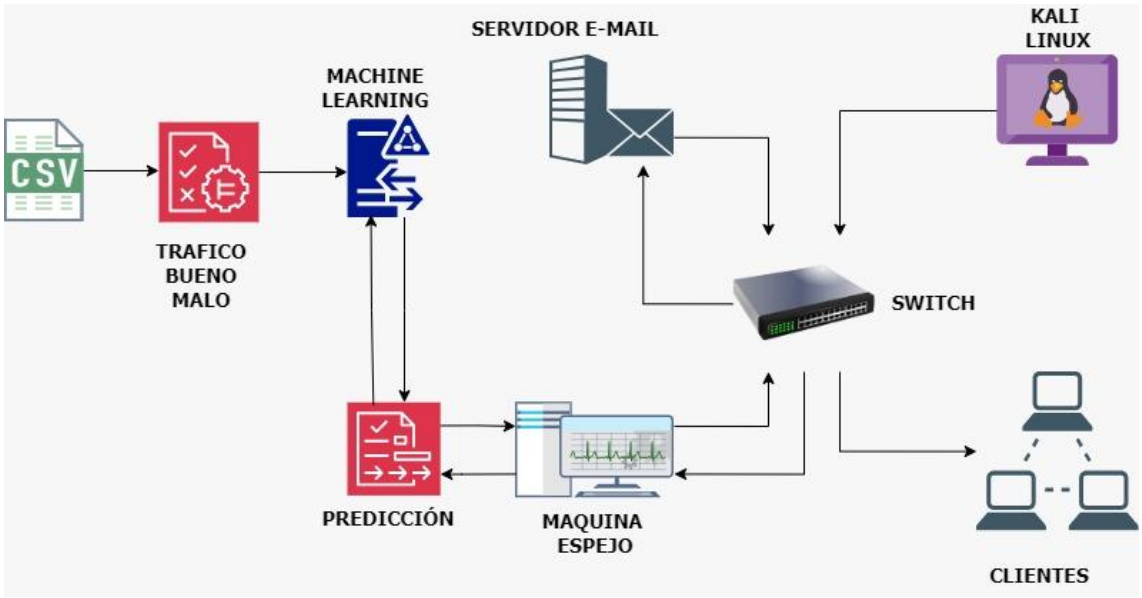


Figura 8: Diseño del prototipo

### 2.3.2 Preparación de los datos

Se extrajo los datos de la página Kaggle que es una plataforma web que reúne a la comunidad Data Science el mundo. El conjunto de datos especifica el cuerpo del texto del correo electrónico, el tipo de correos electrónicos que se pueden usar para detectar correos electrónicos de phishing mediante un análisis exhaustivo del texto del correo electrónico y la clasificación de aquellos que utilizan el aprendizaje automático.

Una vez que se extrajeron los datos del dataset (Phishing_Email.csv), se procedió a limpiar los datos para garantizar su calidad y precisión. La limpieza de los datos involucra la detección y corrección de anomalías y datos erróneos, la eliminación de registros duplicados y la normalización de la información a un formato unificado. Este proceso es crucial para prevenir fallas en análisis futuros y para sustentar la precisión en fundamentos de datos fiables y meticulosamente corregidos. La limpieza de datos puede ser un proceso complejo que requiere del uso adecuado de técnicas y herramientas para lograr una limpieza efectiva de los datos. Al garantizar la integridad de los datos, se facilita la replicabilidad de cualquier análisis futuro.

En el fragmento de código que visualizamos en la **figura 9**, se importan las bibliotecas necesarias para la visualización y el procesamiento de datos, específicamente Matplotlib

y NumPy, que son esenciales para la manipulación de arrays y la generación de gráficos. Luego, se carga un conjunto de datos "Phishing_Email.csv" utilizando Pandas, una biblioteca de Python destinada al análisis de datos, para finalmente visualizar las primeras filas del dataset con el objetivo de obtener una vista previa preliminar de la estructura y el contenido de los datos recién importados.

```
import numpy as np; import pandas as pd; import os
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
# Importamos las herramientas de RandomForest para evaluar el modelo
from sklearn.ensemble import RandomForestClassifier
# Importamos las herramientas del modelo de Support Vector Machine (SVM)
from sklearn.svm import SVC
# Importamos las herramientas de Gradient Boosting (XGBoost)
import xgboost as xgb
# Importamos las librerias para la comparación de los modelos
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, roc_auc_score, roc_curve
import matplotlib.pyplot as plt
import seaborn as sns


# Cargar datos
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Importar el conjunto de datos
df = pd.read_csv("C:/Users/erick/Downloads/monitoreo/APP/Phishing_Email.csv")
df.head()  # Muestra las primeras filas del DataFrame para tener una vista previa de los datos
```

✓ 7.3s

| | Unnamed: 0 | Email Text | Email Type |
|---|---|---|---|
| 0 | 0 | re : 6 . 1100 , disc : uniformitarianism , re ... | Safe Email |
| 1 | 1 | the other side of * galicismos * * galicismo *... | Safe Email |
| 2 | 2 | re : equistar deal tickets are you still avail... | Safe Email |
| 3 | 3 | \nHello I am your hot lil horny toy.\n I am... | Phishing Email |
| 4 | 4 | software at incredibly low prices ( 86 % lower... | Phishing Email |

*Figura 9: Importar bibliotecas y el conjunto de datos.*

El siguiente código (**figura 10**) se enfoca en la identificación y gestión de los valores faltantes (NaN - Not a Number) dentro del conjunto de datos. Inicialmente, se calcula la suma de valores NaN en cada columna del DataFrame df para determinar la presencia de datos incompletos.

Para posteriormente eliminar todas las filas que contienen al menos un valor NaN, con el objetivo de purificar el dataset de cualquier entrada incompleta. Tras la eliminación, se realiza una verificación adicional para asegurar que no queden valores NaN en el conjunto de datos.

```python
# Verificación y Manejo de Valores NaN
# Verificar y manejar los valores NaN (Not a Number)
df.isna().sum()  # Suma de valores NaN en cada columna

# Eliminar filas con valores NaN
df = df.dropna()
print(df.isna().sum())  # Verificar nuevamente los valores NaN después de la eliminación

# Mostrar la forma del dataset
print("Forma del dataset:", df.shape)
```

✓ 0.0s

```
Unnamed: 0    0
Email Text    0
Email Type    0
dtype: int64
Forma del dataset: (18634, 3)
```

*Figura 10: Preprocesamiento - Eliminación de valores NaN*

En la **figura 11**, el código ilustra un método para equilibrar clases en un conjunto de datos de correos etiquetados como "Seguros" y "Phishing" mediante submuestreo. Se seleccionan aleatoriamente correos seguros hasta igualar la cantidad de correos phishing, asegurando así un balance entre las clases.
Luego, se combina ambos subconjuntos en un nuevo conjunto de datos balanceado. Este conjunto se divide en características (texto del correo) y variable objetivo (tipo de correo), y posteriormente en subconjuntos de entrenamiento y prueba usando una división del 80% para entrenamiento y 20% para prueba, preparando los datos para su uso en modelos de aprendizaje automático.

```python
#Balanceo de Clases
# Utilizaremos la técnica de submuestreo para equilibrar las clases
Safe_Email = df[df["Email Type"] == "Safe Email"]
Phishing_Email = df[df["Email Type"] == "Phishing Email"]

# Seleccionamos una muestra aleatoria de correos seguros para igualar la cantidad de correos de phishing
Safe_Email = Safe_Email.sample(Phishing_Email.shape[0])

# Verificamos nuevamente la forma (cantidad de muestras) de cada clase para asegurarnos de que están balanceadas
print(Safe_Email.shape, Phishing_Email.shape)

# Creamos un nuevo DataFrame con los tipos de correo electrónico balanceados
Data = pd.concat([Safe_Email, Phishing_Email], ignore_index=True)
Data.head()  # Muestra las primeras filas del nuevo DataFrame para verificar
```

✓ 0.0s

```
(7312, 3) (7312, 3)
```

| | Unnamed: 0 | Email Text | Email Type |
|---|---|---|---|
| 0 | 7997 | URL: http://www.newsisfree.com/click/-1,841027... | Safe Email |
| 1 | 17155 | generative approaches to sla iv second call fo... | Safe Email |
| 2 | 5076 | self-opposites content - length : 333 jules le... | Safe Email |
| 3 | 4383 | conference announcement the research institute... | Safe Email |
| 4 | 8021 | re : 6 . 249 dick armey 's slip and correction... | Safe Email |

*Figura 11:Técnica de submuestreo y nuevo dataframe.*

En las **figuras 12-13-14,** agregamos dos nuevos modelos, los cuales son XGBoost, SVM al ya existente RandomForestClassifier que ya estaba pre-entrado en el dataset, quienes para la clasificación de los textos utilizan una 'Pipeline' de scikit-learn, para convertir texto en características numéricas ponderadas, para clasificar correos electrónicos. Esto nos permitió transformar eficazmente el texto en una matriz TF-IDF y aplicar clasificación, logrando entrenar los modelos con datos etiquetados y evaluar su precisión en un conjunto de prueba, revelando así la efectividad de nuestra metodología en la clasificación de textos.

```python
# modelo RandomForest
from sklearn.ensemble import RandomForestClassifier

# Crear el pipeline para el modelo RandomForest
RF_pipeline = Pipeline([
    ("tfidf", TfidfVectorizer()),  # Convertir el texto a una matriz TF-IDF
    ("RF", RandomForestClassifier(n_estimators=100, random_state=0))  # Configurar RandomForest
])

# Entrenar el modelo RandomForest con el conjunto de entrenamiento
RF_pipeline.fit(X_train, y_train)

# Realizar predicciones con el modelo RandomForest
y_pred_rf = RF_pipeline.predict(X_test)
y_prob_rf = RF_pipeline.predict_proba(X_test)[:, 1]  # Probabilidades para la clase positiva

# Evaluar el modelo RandomForest
print("Precisión del modelo RandomForest:", accuracy_score(y_test, y_pred_rf))
print("Matriz de Confusión:\n", confusion_matrix(y_test, y_pred_rf))
print("Informe de Clasificación:\n", classification_report(y_test, y_pred_rf))
print("AUC del modelo RandomForest:", roc_auc_score(y_test, y_prob_rf))

# Calcular la curva ROC para RandomForest
fpr_rf, tpr_rf, _ = roc_curve(y_test, y_prob_rf)
```

*Figura 12: Creación del modelo RandomForest, entrenamiento y cálculo de precisión.*

```python
# modelo de Support Vector Machine (SVM)
# Crear el pipeline para el modelo SVM
SVM_pipeline = Pipeline([
    ("tfidf", TfidfVectorizer()),  # Convertir el texto a una matriz TF-IDF
    ("SVM", SVC(C=1, kernel='linear', gamma='auto'))  # Configurar el clasificador SVM
])

# Entrenar el modelo SVM con el conjunto de entrenamiento
SVM_pipeline.fit(X_train, y_train)

# Realizar predicciones con el modelo SVM
y_pred_svm = SVM_pipeline.predict(X_test)
y_pred_svm = SVM_pipeline.predict(X_test)
y_prob_svm = SVM_pipeline.decision_function(X_test)  # Probabilidades para la clase positiva
# Evaluar el modelo SVM
print("Precisión del modelo SVM:", accuracy_score(y_test, y_pred_svm))
print("Matriz de Confusión:\n", confusion_matrix(y_test, y_pred_svm))
print("Informe de Clasificación:\n", classification_report(y_test, y_pred_svm))
print("AUC del modelo SVM:", roc_auc_score(y_test, y_prob_svm))

# Calcular la curva ROC
fpr_svm, tpr_svm, _ = roc_curve(y_test, y_prob_svm)
```

*Figura 13: Creación del modelo SVM, entrenamiento y cálculo de precisión.*

```python
# Gradient Boosting (XGBoost)
# Crear el pipeline para el modelo XGBoost
XGB_pipeline = Pipeline([
    ("tfidf", TfidfVectorizer()),  # Convertir el texto a una matriz TF-IDF
    ("XGB", xgb.XGBClassifier(objective='binary:logistic', n_estimators=100, use_label_encoder=False))  # Configurar XGBoost
])

# Entrenar el modelo XGBoost con el conjunto de entrenamiento
XGB_pipeline.fit(X_train, y_train)

# Realizar predicciones con el modelo XGBoost
y_pred_xgb = XGB_pipeline.predict(X_test)
y_prob_svm = SVM_pipeline.decision_function(X_test)  # Probabilidades para la clase positiva
y_prob_xgb = XGB_pipeline.predict_proba(X_test)[:, 1]  # Probabilidades para la clase positiva

# Evaluar el modelo XGBoost
print("Precisión del modelo XGBoost:", accuracy_score(y_test, y_pred_xgb))
print("Matriz de Confusión:\n", confusion_matrix(y_test, y_pred_xgb))
print("Informe de Clasificación:\n", classification_report(y_test, y_pred_xgb))
print("AUC del modelo XGBoost:", roc_auc_score(y_test, y_prob_xgb))

# Calcular la curva ROC
fpr_xgb, tpr_xgb, _ = roc_curve(y_test, y_prob_xgb)
```

*Figura 14: Creación del modelo XGBoost, entrenamiento y cálculo de precisión.*

En esta **figura 15** se configura y evalúa el modelo de RandomForest para clasificar correos electrónicos. Se optimizan hiperparámetros como el número de estimadores y la profundidad del árbol, utilizando validación cruzada para garantizar la robustez del

modelo. Tras encontrar el mejor modelo, se evalúa en el conjunto de prueba, calculando métricas como la precisión, la matriz de confusión, el informe de clasificación y el AUC. Estas métricas ofrecen una visión detallada del rendimiento del modelo, destacando su capacidad para identificar y clasificar correos de phishing de manera efectiva.

```
# Pipeline y Randomized Search para RandomForest
pipeline_rf = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', RandomForestClassifier(random_state=0))
])
param_dist_rf = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [None, 10, 20, 30],
    'classifier__min_samples_split': [2, 5, 10],
    'classifier__min_samples_leaf': [1, 2, 4],
    'classifier__max_features': ['sqrt', 'log2']
}
random_search_rf = RandomizedSearchCV(pipeline_rf, param_dist_rf, n_iter=36, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_rf.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_rf_model = random_search_rf.best_estimator_
best_rf_params = random_search_rf.best_params_
predictions_rf = best_rf_model.predict(X_test_new)
probs_rf = best_rf_model.predict_proba(X_test_new)[:, 1]
```

*Figura 15: Configuración y evaluación del modelo RandomForest*

Esta **figura 16** se implementa y evalúa el modelo de clasificación XGBoost. El modelo se entrena con datos transformados y se ajusta con parámetros específicos como **'n_estimators=100'** y **'eval_metric'='logloss'**, optimizando para la precisión logarítmica de la pérdida. Posteriormente, se evalúa en un conjunto de datos de prueba, calculando métricas clave como precisión, matriz de confusión, informe de clasificación y AUC. Estas métricas ofrecen una visión detallada de la habilidad del modelo para diferenciar entre correos electrónicos seguros y de phishing de manera efectiva.

```
# Pipeline y Randomized Search para XGBoost
pipeline_xgb = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss'))
])
param_dist_xgb = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [3, 5, 7],
    'classifier__learning_rate': [0.01, 0.1, 0.2],
    'classifier__subsample': [0.7, 0.8, 1.0],
    'classifier__colsample_bytree': [0.7, 0.8, 1.0],
    'classifier__gamma': [0, 0.1, 0.2],
    'classifier__min_child_weight': [1, 3, 5]
}
random_search_xgb = RandomizedSearchCV(pipeline_xgb, param_dist_xgb, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_xgb.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_xgb_model = random_search_xgb.best_estimator_
best_xgb_params = random_search_xgb.best_params_
predictions_xgb = best_xgb_model.predict(X_test_new)
probs_xgb = best_xgb_model.predict_proba(X_test_new)[:, 1]
```

*Figura 16: Configuración y evaluación del modelo XGBoost*

Esta **figura 17** se configura y evalúa el modelo de clasificación SVM para correos electrónicos. Se ajustan parámetros como ***C=1 y gamma='auto',*** y se habilita la probabilidad para permitir la predicción de probabilidades. Tras entrenar el modelo con los datos de entrenamiento, se realizan predicciones sobre el conjunto de prueba, y se evalúa la eficacia del modelo mediante métricas estándar: precisión, matriz de confusión, informe de clasificación y AUC. Estas métricas ayudan a determinar la capacidad del modelo SVM para identificar correctamente correos electrónicos seguros y de phishing, destacando su rendimiento y precisión predictiva.

```python
# Pipeline y Randomized Search para SVM
pipeline_svm = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', SVC(probability=True))
])
param_dist_svm = {
    'classifier__C': [0.1, 1, 10],
    'classifier__kernel': ['linear', 'rbf', 'poly'],
    'classifier__gamma': ['scale', 'auto'],
    'classifier__class_weight': [None, 'balanced']
}
random_search_svm = RandomizedSearchCV(pipeline_svm, param_dist_svm, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_svm.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_svm_model = random_search_svm.best_estimator_
best_svm_params = random_search_svm.best_params_
predictions_svm = best_svm_model.predict(X_test_new)
probs_svm = best_svm_model.predict_proba(X_test_new)[:, 1]
```

*Figura 17:Configuración y evaluación del modelo SVM*

## 2.4 Ejecución del prototipo

### 2.4.1 Desarrollo de la topología

Ya una vez terminado el modelo continuamos con la creación de la topología que vamos a utilizar y en la que vamos a implementar el modelo de detección.

#### a) Descripción de la topología

Para la topología primero se agregó un router configurado con DHCP para la asignación automática de direcciones IP. Se le asigno direcciones IP a dos puertos del router para las dos redes que se manejaran. De un lado de la red (192.168.0.10/24) estará un switch configurado con un puerto espejo para reflejar los datos a una de las máquinas y en la otra red (192.168.1.10/24) estará la maquina atacante donde se enviarán los ataques y a su vez mandara correos normales para probar que detecte cual es atacante y cual no. **Figura 18**.

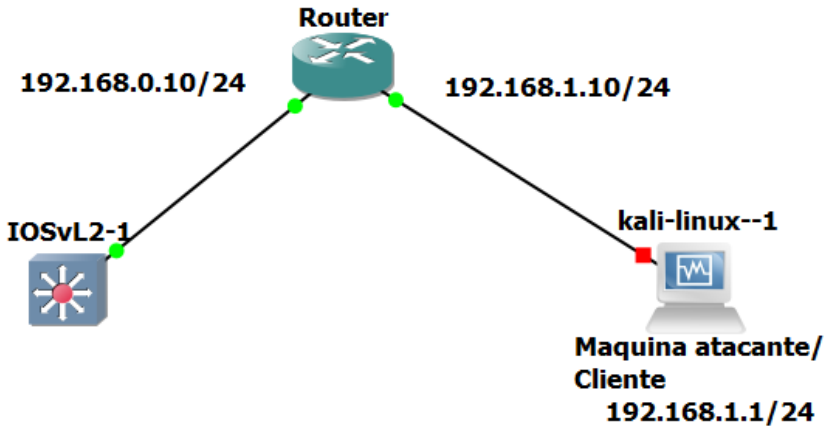


*Figura 18: Primera prueba de la topología.*

Después de configurar el router y comprobar que asigna direcciones, procedemos a agregar las demás maquinas. Agregamos maquinas clientes que se enviarán correos entre sí y que serán objetivos para los ataques (W10_1, w7-1), agregaremos el servidor de correos de Windows Server (hmailserver) y una maquina ya cargada con el modelo de detección que a su vez será la máquina que le llegaran los datos del puerto espejo. **Figura 19**.

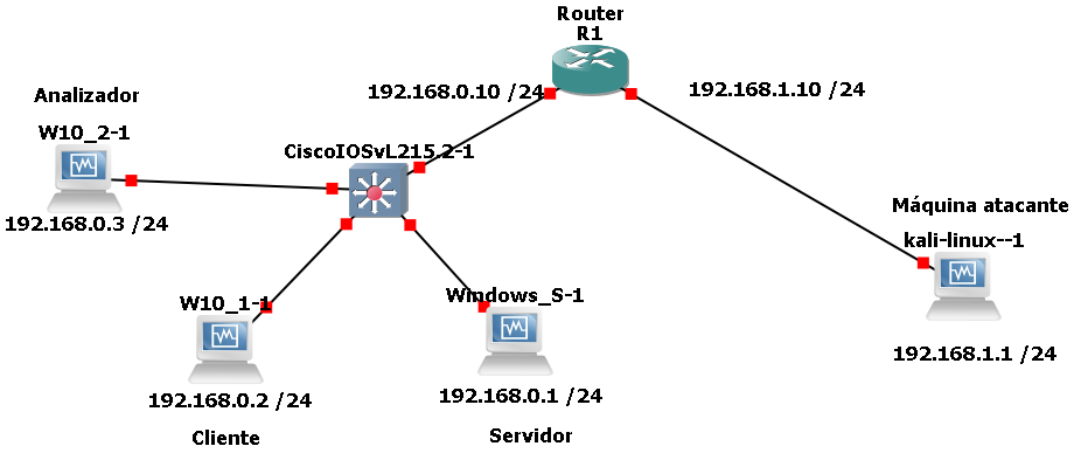


*Figura 19: Topología que se utilizó.*

### b) Modelado.

Para el entrenamiento y evaluación de los modelos se inicia con un conjunto de datos de correos electrónicos totalmente nuevos y distintos al dataset original previamente descargado en Kaggle(Phishing_Email.csv) pero que se crearon a partir del dataset , uno se creó cambiando palabras de cada correo y el otro creaba correos completamente nuevos.

Primero realizamos las pruebas con el dataset que cambia las palabras de los correos para evaluar los modelos, todo esto lo visualizamos en la **figura 20**.

```
# Ruta del nuevo archivo CSV con correos electrónicos generados
#correos generado con NewEmail, correos cambiados solo palabras
new_file_path = 'New_Phishing_Email_Evaluation.csv'
# Cargar el nuevo archivo CSV
new_df = pd.read_csv(new_file_path)
# Asegurar que todos los correos sean cadenas y manejar NaN
new_df['Email Text'] = new_df['Email Text'].fillna('').astype(str)
```

```
# Dividir los datos en características (X) y variable dependiente (y)
X_new = new_df['Email Text']
y_new = new_df['Email Type']
# Convertir etiquetas a valores numéricos
label_encoder = LabelEncoder()
y_new = label_encoder.fit_transform(y_new)
# Dividir los datos en conjuntos de entrenamiento y prueba
X_train_new, X_test_new, y_train_new, y_test_new = train_test_split(X_new, y_new, test_size=0.3, random_state=0)
```

*Figura 20:Cargar nuevos correos y dividirlos.*

La **figura 21** aborda la configuración y evaluación del modelo RandomForest. Se define un amplio rango de hiperparámetros para optimizar, incluyendo la profundidad máxima de los árboles, el número mínimo de muestras para dividir nodos y en hojas, y el número máximo de características. Se realiza una búsqueda aleatoria eficiente de los mejores hiperparámetros, utilizando 36 iteraciones y validación cruzada de 5 pliegues. Finalmente, se utiliza este modelo optimizado para realizar predicciones en el conjunto de prueba y se obtienen las probabilidades de predicción, permitiendo una evaluación completa del rendimiento del modelo en la tarea de clasificación de texto.

```
# Pipeline y Randomized Search para RandomForest
pipeline_rf = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', RandomForestClassifier(random_state=0))
])
param_dist_rf = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [None, 10, 20, 30],
    'classifier__min_samples_split': [2, 5, 10],
    'classifier__min_samples_leaf': [1, 2, 4],
    'classifier__max_features': ['sqrt', 'log2']
}
random_search_rf = RandomizedSearchCV(pipeline_rf, param_dist_rf, n_iter=36, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_rf.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_rf_model = random_search_rf.best_estimator_
best_rf_params = random_search_rf.best_params_
predictions_rf = best_rf_model.predict(X_test_new)
probs_rf = best_rf_model.predict_proba(X_test_new)[:, 1]
```

*Figura 21:Evaluación del modelo RandomForest.*

En la **figura 22**, se configura y evalúa el modelo XGBoost. Se define los hiperparámetros para optimización, incluyendo el número de estimadores, profundidad máxima, y otros parámetros específicos de XGBoost. Tras entrenar el modelo con los datos adecuados, se procede a evaluarlo mediante métricas estandarizadas. Estas métricas son fundamentales para analizar tanto la exactitud general del modelo como su capacidad para distinguir entre clases de manera efectiva, destacando su rendimiento predictivo en escenarios reales.

```
# Pipeline y Randomized Search para XGBoost
pipeline_xgb = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss'))
])
param_dist_xgb = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [3, 5, 7],
    'classifier__learning_rate': [0.01, 0.1, 0.2],
    'classifier__subsample': [0.7, 0.8, 1.0],
    'classifier__colsample_bytree': [0.7, 0.8, 1.0],
    'classifier__gamma': [0, 0.1, 0.2],
    'classifier__min_child_weight': [1, 3, 5]
}
random_search_xgb = RandomizedSearchCV(pipeline_xgb, param_dist_xgb, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_xgb.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_xgb_model = random_search_xgb.best_estimator_
best_xgb_params = random_search_xgb.best_params_
predictions_xgb = best_xgb_model.predict(X_test_new)
probs_xgb = best_xgb_model.predict_proba(X_test_new)[:, 1]
```

*Figura 22:Evaluación del modelo XGBoost*

La **figura 23**, implementa un modelo de clasificación mediante Máquinas de Vectores de Soporte (SVM) para procesamiento de texto. Se establece el clasificador SVM con un kernel lineal y parámetros como ***C=1*** ***y gamma='auto'***, lo que permite ajustar la complejidad del modelo y la adaptación a datos de alta dimensionalidad. El clasificador SVM se configura para producir estimaciones de probabilidad.

```
# Pipeline y Randomized Search para SVM
pipeline_svm = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', SVC(probability=True))
])
param_dist_svm = {
    'classifier__C': [0.1, 1, 10],
    'classifier__kernel': ['linear', 'rbf', 'poly'],
    'classifier__gamma': ['scale', 'auto'],
    'classifier__class_weight': [None, 'balanced']
}
random_search_svm = RandomizedSearchCV(pipeline_svm, param_dist_svm, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_svm.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_svm_model = random_search_svm.best_estimator_
best_svm_params = random_search_svm.best_params_
predictions_svm = best_svm_model.predict(X_test_new)
probs_svm = best_svm_model.predict_proba(X_test_new)[:, 1]
```

*Figura 23: Evaluación del modelo SVM*

Luego de evaluar los modelos con el primero de los dataset creados, continuamos con el segundo. El segundo dataset combinando estructuras de otros correos lo cual genero correos completamente nuevos. Podemos visualizar en la **figura 24** que se carga segundo dataset para luego dividir los correos.

```
# Leer y preprocesar los datos
new_file_path = 'Phishing_Email_Evaluation.csv'
new_df = pd.read_csv(new_file_path)
new_df['Email Text'] = new_df['Email Text'].fillna('').astype(str)
X_new = new_df['Email Text']
y_new = new_df['Email Type']
label_encoder = LabelEncoder()
y_new = label_encoder.fit_transform(y_new)

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train_new, X_test_new, y_train_new, y_test_new = train_test_split(X_new, y_new, test_size=0.3, random_state=0)
```

*Figura 24: Leer datos del segundo dataset.*

En esta **figura 25** se configura y evalúa el modelo de RandomForest para clasificar correos electrónicos. Se optimizan hiperparámetros como el número de estimadores y la profundidad del árbol, utilizando validación cruzada para garantizar la solidez del modelo. Una vez identificado el modelo óptimo, se evalúa en un conjunto de datos de prueba y se calculan métricas clave, como precisión, matriz de confusión, informe de clasificación y AUC. Estas métricas ofrecen una visión detallada del rendimiento del modelo, destacando su capacidad para identificar y clasificar correos de phishing de manera efectiva.

```
# Pipeline y Randomized Search para RandomForest
pipeline_rf = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', RandomForestClassifier(random_state=0))
])
param_dist_rf = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [None, 10, 20, 30],
    'classifier__min_samples_split': [2, 5, 10],
    'classifier__min_samples_leaf': [1, 2, 4],
    'classifier__max_features': ['sqrt', 'log2']
}
random_search_rf = RandomizedSearchCV(pipeline_rf, param_dist_rf, n_iter=36, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_rf.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_rf_model = random_search_rf.best_estimator_
best_rf_params = random_search_rf.best_params_
predictions_rf = best_rf_model.predict(X_test_new)
probs_rf = best_rf_model.predict_proba(X_test_new)[:, 1]
```

*Figura 25: Evaluación del modelo RandomForest (dataset original)*

Esta **figura 26** detalla la implementación y evaluación de un modelo XGBoost. El modelo se entrena con datos transformados y se ajusta con parámetros específicos como **'n_estimators=100'** y **'eval_metric'='logloss'**, optimizando para la precisión logarítmica de la pérdida. Posteriormente, se evalúa en el conjunto de prueba, donde se calculan métricas clave como precisión, la matriz de confusión, el informe de clasificación, y el AUC. Estas métricas proporcionan una comprensión profunda de la capacidad del modelo para distinguir entre correos seguros y de phishing de manera efectiva.

```
# Pipeline y Randomized Search para XGBoost
pipeline_xgb = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss'))
])
param_dist_xgb = {
    'classifier__n_estimators': [50, 100, 200, 300],
    'classifier__max_depth': [3, 5, 7],
    'classifier__learning_rate': [0.01, 0.1, 0.2],
    'classifier__subsample': [0.7, 0.8, 1.0],
    'classifier__colsample_bytree': [0.7, 0.8, 1.0],
    'classifier__gamma': [0, 0.1, 0.2],
    'classifier__min_child_weight': [1, 3, 5]
}
random_search_xgb = RandomizedSearchCV(pipeline_xgb, param_dist_xgb, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_xgb.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_xgb_model = random_search_xgb.best_estimator_
best_xgb_params = random_search_xgb.best_params_
predictions_xgb = best_xgb_model.predict(X_test_new)
probs_xgb = best_xgb_model.predict_proba(X_test_new)[:, 1]
```

*Figura 26: Evaluación de un modelo XGBoost (dataset original)*

Esta **figura 27** configura y evalúa un modelo de clasificación SVM para correos electrónicos. Se ajustan parámetros ***como C=1 y gamma='auto',*** y se habilita la probabilidad para permitir la predicción de probabilidades. Tras entrenar el modelo con los datos de entrenamiento, se realizan predicciones sobre el conjunto de prueba, y se evalúa la eficacia del modelo mediante métricas estándar: precisión, matriz de confusión, informe de clasificación y AUC. Estas métricas ayudan a determinar la capacidad del modelo SVM para identificar correctamente correos electrónicos seguros y de phishing, destacando su rendimiento y precisión predictiva.

```
# Pipeline y Randomized Search para SVM
pipeline_svm = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('classifier', SVC(probability=True))
])
param_dist_svm = {
    'classifier__C': [0.1, 1, 10],
    'classifier__kernel': ['linear', 'rbf', 'poly'],
    'classifier__gamma': ['scale', 'auto'],
    'classifier__class_weight': [None, 'balanced']
}
random_search_svm = RandomizedSearchCV(pipeline_svm, param_dist_svm, n_iter=50, cv=5, scoring='f1', random_state=0, error_score='raise')
random_search_svm.fit(X_train_new, y_train_new)

# Evaluar el modelo ajustado en el conjunto de prueba
best_svm_model = random_search_svm.best_estimator_
best_svm_params = random_search_svm.best_params_
predictions_svm = best_svm_model.predict(X_test_new)
probs_svm = best_svm_model.predict_proba(X_test_new)[:, 1]
```

*Figura 27: Evaluación del modelo SVM (dataset original)*

### c) Implementación.

Después de completar la topología e implementar en puerto espejo se instala y prueba el servidor de correos "hMailServer" para Windows. Viendo que se envíen correctamente los paquetes y lleguen a la maquina analizadora.

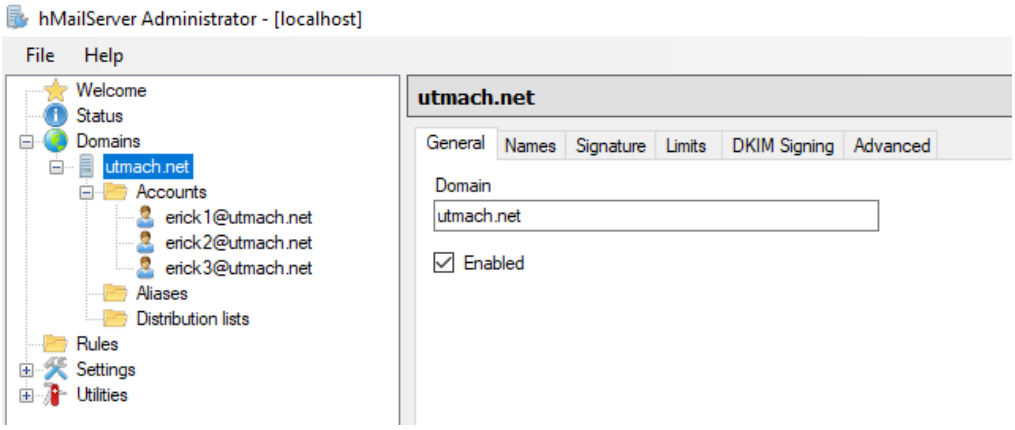


*Figura 28: Vista del servidor de correos.*

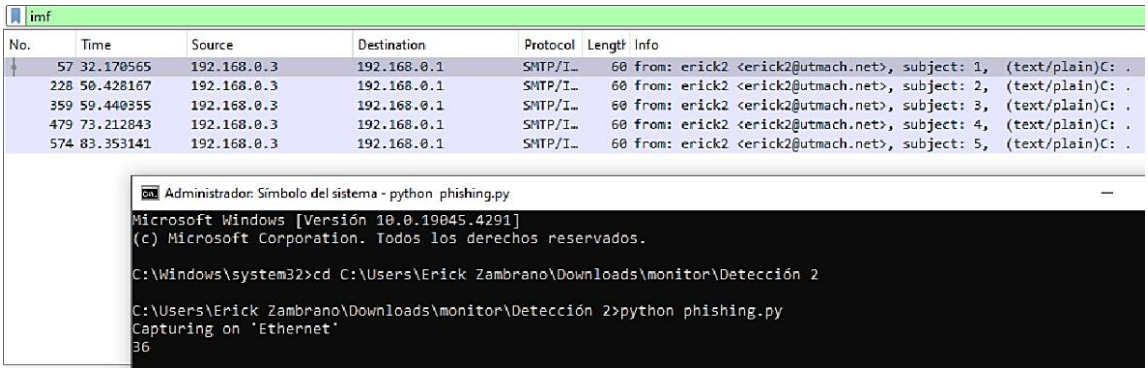


*Figura 29: Puerto espejo capturando los correos.*

Después de saber que el SVM es el mejor modelo procedemos a serializarlo para usarlo en la red simulada. Para serializarlo utilizamos tanto los datos originales del dataset "Phishing_Emails.csv" como los otros que se crearon para las evaluaciones tal como visualizamos en la figura.

```
# Leer y preprocesar los datos
file_path1 = 'Phishing_Email_Evaluation.csv'
file_path2 = 'New_Phishing_Email_Evaluation.csv'
file_path3 = 'Phishing_Email.csv'
df1 = pd.read_csv(file_path1)
df2 = pd.read_csv(file_path2)
df3 = pd.read_csv(file_path3)
df1['Email Text'] = df1['Email Text'].fillna('').astype(str)
df2['Email Text'] = df2['Email Text'].fillna('').astype(str)
df3['Email Text'] = df3['Email Text'].fillna('').astype(str)

# Combinar los conjuntos de datos
combined_df = pd.concat([df1, df2, df3], ignore_index=True)
```

*Figura 30: Leer y procesar cada dataset.*

Luego dividimos los datos combinados en características (textos de correos electrónicos) y etiquetas (tipo de correo). Posterior seguimos con el entrenamiento del modelo, realizando una búsqueda aleatoria de hiperparámetros (RandomizedSearchCV) para ajustar el modelo SVM. Al serializar el modelo SVM se utilizó RandomizedSearchCV porque es mejor para trabajar con grandes cantidades de datos. Tras entrenar el modelo lo evaluamos calculando las métricas utilizando validación cruzada (cross_val_predict)

para evaluar la robustez del modelo. Para finalizar guardemos el modelo entrenado en archivos '. pkl' para su posterior uso, tal como lo visualizamos en la **figura 31**.

```
# Guardar el modelo entrenado y el label encoder
model_path = 'svm_model_with_vectorizer.pkl'
label_encoder_path = 'label_encoder.pkl'
joblib.dump(best_svm_model, model_path)
joblib.dump(label_encoder, label_encoder_path)
print(f"Modelo guardado en {model_path}")
print(f"Label Encoder guardado en {label_encoder_path}")
```

*Figura 31:Guardar el modelo entrenado.*

Una vez almacenado el modelo procedemos a probarlo en la red simulada. Para esto se creó un archivo '.py' para leer los correos y cargar el modelo, este archivo tiene el nombre de "classify_emails.py". Este Archivo de Python lo que hace primero es extraer los correos obtenidos durante la captura de paquetes para posteriormente limpiarlos y guardarlos en un nuevo archivo de texto, tal como se visualiza en la **figura 32**.

```
# Función para limpiar el texto
def clean_text(text):
    text = re.sub(r'Content-Transfer-Encoding:.*?\r\n\r\n', '', text, flags=re.DOTALL)
    text = re.sub(r'^Timestamps.*?$|^Date:.*?$|^MIME-Version:.*?$|^User-Agent:.*?$|^Content-Language:.*?$|^To:.*?$', '', text, flags=re.MULTILINE)
    text = re.sub(r'^From:.*?$|^Subject:.*?$|^Content-Type:.*?$', '', text, flags=re.MULTILINE)
    text = re.sub(r'8bit\\r\\n,\\r\\n,', '', text)
    text = re.sub(r'\\r\\n,', ' ', text)
    text = re.sub(r'\s+\.', '.', text)
    text = re.sub(r'Timestamps\s+', '', text)
    text = re.sub(r'Message-ID: <.*?>', '', text)
    text = re.sub(r'[\r\n]+', ' ', text)
    text = re.sub(r'\s{2,}', ' ', text)
    return text.strip()

# Función para extraer y limpiar correos electrónicos de un archivo de texto
def extract_and_clean_emails(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        content = file.read()
    emails = content.split('From:')
    cleaned_emails = [clean_text(email.strip()) for email in emails if email.strip()]
    return cleaned_emails

# Guardar el contenido limpiado en un nuevo archivo
def save_cleaned_emails(cleaned_emails, output_file_path):
    with open(output_file_path, 'w', encoding='utf-8') as file:
        for i, email in enumerate(cleaned_emails):
            if i > 0:
                file.write("\n\n")
            file.write(email)
```

*Figura 32: Extracción, limpieza y almacenamiento de los correos.*

Luego cargamos los modelos para predecir qué tipo de correos son, además se crea la función que se utilizara junto al modelo para clasificar los correos. Como se ve en la **figura 33**.

```
# Cargar el modelo y el LabelEncoder
def load_model_and_encoder():
    model_path = 'svm_model_with_vectorizer.pkl'
    label_encoder_path = 'label_encoder.pkl'
    model = joblib.load(model_path)
    label_encoder = joblib.load(label_encoder_path)
    return model, label_encoder

# Función para predecir correos electrónicos
def predict_emails(emails, model):
    tfidf_vectorizer = model.named_steps['tfidf']
    X_emails = tfidf_vectorizer.transform(emails)
    predictions = model.named_steps['classifier'].predict(X_emails)
    probabilities = model.named_steps['classifier'].predict_proba(X_emails)[:, 1]
    probabilities = np.clip(probabilities, 0.0001, 0.9999)  # Normalizar las probabilidades
    return predictions, probabilities
```

*Figura 33: Cargar el modelo de predicción.*

Para finalmente ejecutar la función principal donde se procesa y clasifica los correos. **Figura 34**.

```python
# Función principal para procesar y clasificar correos electrónicos
def main(file_path):
    model, label_encoder = load_model_and_encoder()
    cleaned_emails = extract_and_clean_emails(file_path)
    save_cleaned_emails(cleaned_emails, 'cleaned_email_content.txt')
    print(f"Se han extraído {len(cleaned_emails)} correos electrónicos.")
    if len(cleaned_emails) == 0:
        print("No se encontraron correos electrónicos en el archivo.")
        return
    predictions, probabilities = predict_emails(cleaned_emails, model)
    classified_emails = []
    for i, email in enumerate(cleaned_emails):
        prediction_label = label_encoder.inverse_transform([predictions[i]])[0]
        probability = probabilities[i]
        email_header = email.split('Content-Transfer-Encoding:')[0].strip()
        email_body = email.split('Content-Transfer-Encoding:')[1].strip() if 'Content-Transfer-Encoding:' in email else email.strip()
        email_header_preview = ' '.join(email_header.split()[:4]) + '...'
        email_body_preview = ' '.join(email_body.split()[:10]) + '...'
        classified_emails.append({
            'Email': email_header_preview,
            'Contenido del correo': email_body_preview,
            'Prediction': prediction_label,
            'Probability': f"{probability:.4f}"
        })
        print(f"Email: {email_header_preview}")
        print(f"Contenido del correo: {email_body_preview}")
        print(f"Prediction: {prediction_label}")
        print(f"Probability: {probability:.4f}\n")
    classified_emails_df = pd.DataFrame(classified_emails)
    classified_emails_df.to_csv('classified_emails.csv', index=False)
    print("Results saved to classified_emails.csv")

    total_safe = sum(1 for p in predictions if p == label_encoder.transform(['Safe Email'])[0])
    total_phishing = sum(1 for p in predictions if p == label_encoder.transform(['Phishing Email'])[0]) - 1  # Ajuste para descontar el correo vacío
    print(f"Total Safe Emails: {total_safe}")
    print(f"Total Phishing Emails: {total_phishing}")
```

*Figura 34: Función principal para procesar y clasificar correos.*

Ya solo queda realizar la captura de correos, almacenarlos y luego ejecutar el archivo "classify_emails.py" para clasificar los correos enviados.

```
C:\Users\erick\Downloads\monitoreo\mode\APP2>python classify_emails.py
Se han extraído 9 correos electrónicos.
Email: ...
Contenido del correo: ...
Prediction: Phishing Email

Email: erick3 <erick3@utmach.net> Subject: holis1...
Contenido del correo: 7bit Hello I am your hot lil horny toy. I...
Prediction: Phishing Email

Email: erick3 <erick3@utmach.net> Subject: holis2...
Contenido del correo: 7bit software at incredibly low secirp ( 86 % lower...
Prediction: Phishing Email

Email: erick3 <erick3@utmach.net> Subject: holis3...
Contenido del correo: 7bit entourage , stockmogul newsletter ralph velez , genex pharmaceutical...
Prediction: Phishing Email

Email: erick3 <erick3@utmach.net> Subject: holis4...
Contenido del correo: 7bit we owe you lots of money dear applicant ,...
Prediction: Phishing Email

Email: erick2 <erick2@utmach.net> Subject: hola4...
Contenido del correo: 7bit re : 6. 1100 , disc : uniformitarianism ,...
Prediction: Safe Email

Email: erick2 <erick2@utmach.net> Subject: hola5...
Contenido del correo: 7bit the other side of * galicismos * * galicismo...
Prediction: Safe Email

Email: erick2 <erick2@utmach.net> Subject: hola6...
Contenido del correo: 7bit re : equistar deal tickets are you still available...
Prediction: Safe Email

Email: erick2 <erick2@utmach.net> Subject: hola1...
Contenido del correo: 7bit global risk management operations sally congratulations on your new...
Prediction: Safe Email

Results saved to classified_emails.csv
Total Safe Emails: 4
Total Phishing Emails: 4
```

*Figura 35: Testeo del modelo con correos almacenados.*

# CAPITULO III. EVALUACIÓN DEL PROTOTIPO

## 3.1 Plan de evaluación

### Objetivo General:

Evaluar el rendimiento de los modelos utilizados para la detección de amenazas de phishing, utilizarán herramientas estándar para obtener una visión detallada y completa del desempeño del modelo.

### 3.1.1 Planificación

*Tabla 5: Cronograma del plan de evaluación*

| Actividad | Semanas | | | |
|---|---|---|---|---|
| | Semana 10 | Semana 11 | Semana12 | Semana13 |
| **Desarrollo del plan de Evaluación 4** | | | | |
| **Preparación y ejecución de los Modelos 1** | | | | |
| **Interpretación de Resultados 2** | | | | |
| **Documentar Resultados 3** | | | | |

*Tabla 6:Conograma detallado del plan de evaluación*

| **Cronograma** | |
|---|---|
| **Tareas** | **Resultados Esperados** |
| **Desarrollo del plan de Evaluación** | **Semana 10** |
| Establecer objetivo y cronograma.<br>Elección de herramientas para la evaluación. | Plan de evaluación |
| **Preparación y ejecución de los Modelos** | **Semana 11** |
| Generar nuevos correos normales y de phishing.<br>Ejecución de los modelos. | Definir los datos de prueba.<br>Registro de resultados de cada modelo tras su ejecución. |
| **Interpretación de Resultados** | **Semana 12** |
| Análisis de los resultados.<br>Evaluar el funcionamiento de los modelos.<br>Escoger el mejor modelo. | Cantidad de aciertos y desaciertos.<br>Ajustar las condiciones de los modelos para reducir el error de detección. |
| **Documentar Resultados** | **Semana 13** |
| Documentar los resultados obtenidos del plan de evaluación | Comprobar que los resultados obtenidos cumplen con la hipótesis. |

### 3.1.2 Herramientas y Técnicas

### Matriz de confusión

La matriz de confusión, también conocida como matriz de error, es una herramienta clave para evaluar el desempeño de un modelo de clasificación. Facilita la identificación de elementos como correos electrónicos que deben ser clasificados como spam. Esta matriz ofrece una visión detallada del rendimiento del modelo al mostrar las predicciones correctas e incorrectas para cada categoría. Esta matriz, impulsada por la Inteligencia Artificial, muestra el desempeño de un algoritmo en una tabla de

predicciones. Su efectividad radica en su capacidad para clasificar predicciones, y aunque su funcionamiento pueda parecer complicado, con el aprendizaje supervisado es posible aprovechar sus métricas [37].

| | | Predicción | |
|---|---|---|---|
| | | SI | NO |
| Realidad | SI | TP | FN |
| | NO | FP | TN |

*Figura 36: Ejemplo grafico de la matriz de confusión.*

$$\text{Precisión} = \frac{TP}{(TP+FP)}$$

$$\text{Exactitud} = \frac{TP+TN}{(Total)}$$

La matriz de confusión se muestra en una tabla donde las filas representan las clases verdaderas y las columnas indican las clases predichas por el modelo. Cada celda en esta tabla señala el número de instancias que coinciden con una combinación particular de clase verdadera y clase predicha. Este formato permite observar de manera clara los aciertos y errores del modelo, facilitando la evaluación de su precisión y alcance.

**Curva ROC**

La curva ROC (Receiver Operating Characteristic) es una herramienta gráfica esencial en la evaluación de modelos, utilizada para visualizar el desempeño del clasificador a lo largo de varios umbrales de decisión. Esta curva muestra la relación entre la tasa de verdaderos positivos (sensibilidad) y la tasa de falsos positivos, ofreciendo una medida global de la capacidad del modelo para distinguir entre clases. El área bajo la curva ROC (AUC-ROC) mide esta capacidad discriminativa, donde un valor de 1 indica un clasificador perfecto y un valor de 0.5 sugiere un rendimiento similar al azar. Así, la curva ROC y su AUC correspondiente son herramientas vitales para comparar y seleccionar modelos en situaciones donde es crucial equilibrar la detección de verdaderos positivos con la minimización de falsos positivos[38].

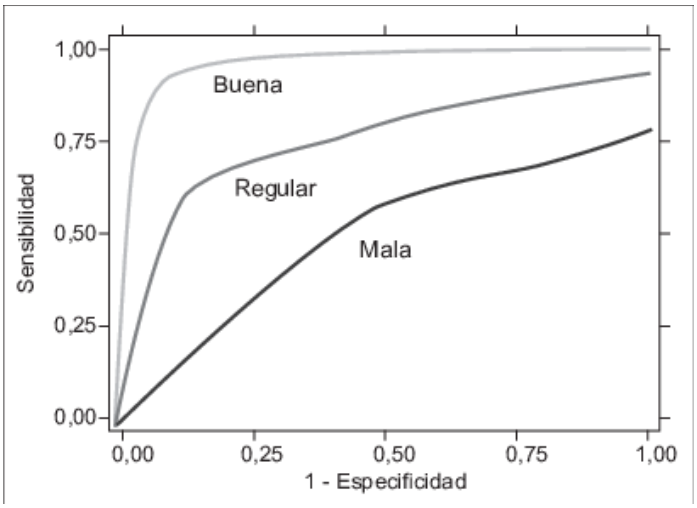


*Figura 37: Ejemplo grafico de la curva ROC.*

### 3.2 Evaluación

Después de realizar todas las pruebas pertinentes, se procede a dividir el conjunto de datos en partes para entrenamiento y prueba. Este proceso es fundamental para evaluar de manera imparcial la efectividad de un modelo predictivo. Específicamente, se utilizan los correos electrónicos etiquetados como características y su clasificación correspondiente como variable dependiente. La selección de un tamaño de prueba del 30% junto con el uso de un estado aleatorio asegura la reproducibilidad y un equilibrio adecuado entre las fases de entrenamiento y validación.

Tras la creación de los modelos, como se muestra en la **figura 38**, se evalúa el rendimiento de cada modelo de clasificación mediante la visualización de la matriz de confusión y el informe de clasificación, comparando las predicciones con los valores reales. La matriz de confusión ofrece una visión detallada de los aciertos y errores de los modelos, clasificados por categoría, mientras que el informe de clasificación proporciona métricas clave como precisión, recall y puntuación F1 para cada clase, permitiendo un análisis más profundo de su efectividad.

```
Matriz de Confusión para RandomForest:
                Predicted Safe  Predicted Phishing
Actual Safe               2161                  37
Actual Phishing            107                2083

Matriz de Confusión para XGBoost:
                Predicted Safe  Predicted Phishing
Actual Safe               2157                  41
Actual Phishing            136                2054

Matriz de Confusión para SVM:
                Predicted Safe  Predicted Phishing
Actual Safe               2179                  19
Actual Phishing              95                2095
```

*Figura 38: Evaluación del rendimiento.*

La **figura 39** luego de la evaluación de cada modelo se crea la gráfica de la curva ROC que muestra la comparativa del desempeño de los modelos RandomForest, XGBoost y SVM en términos de su capacidad para clasificar correctamente entre correos seguros y de phishing. Aquí se presentan los aspectos clave:

1. **RandomForest (AUC = 1.00)**: Muestra una capacidad perfecta para discriminar entre las clases, con un área bajo la curva de 1.0, lo que indica una precisión completa en la clasificación.
2. **XGBoost (AUC = 1.00)**: Al igual que RandomForest, XGBoost tiene un rendimiento perfecto, reflejado en su AUC de 1.0, lo que sugiere una alta sensibilidad y especificidad.
3. **SVM (AUC = 1.00)**: También sobresale con una capacidad predictiva perfecta entre los tres, con una AUC de 1.0. Esto implica que SVM tiene una excelente capacidad para maximizar los verdaderos positivos mientras minimiza los falsos positivos.

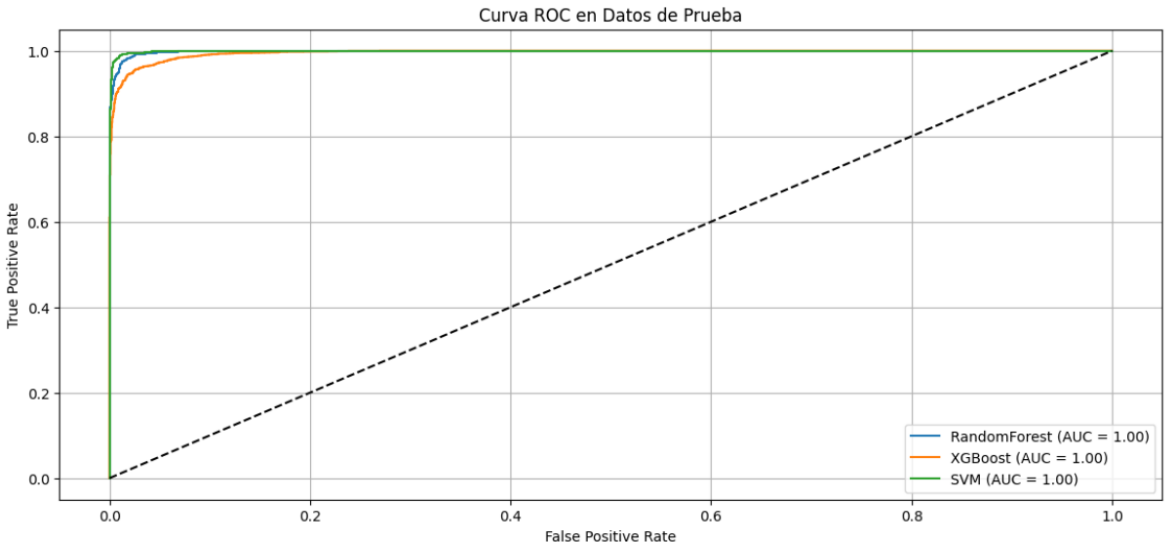


*Figura 39: Curva ROC, Comparativa del desempeño*

La **figura 40** visualiza las matrices de confusión para los modelos RandomForest, XGBoost y SVM, permitiendo una comparación directa del rendimiento en la clasificación de correos electrónicos en categorías de "Safe" y "Phishing":

1. **RandomForest:**
    - Muestra una capacidad adecuada para identificar tanto correos seguros como de phishing, con 2924 verdaderos negativos y 2962 verdaderos positivos.
    - Sin embargo, registra 66 falsos positivos y 48 falsos negativos, lo que indica un margen de error en la detección.
2. **XGBoost:**
    - Tiene un rendimiento ligeramente menor en la detección de correos seguros y de phishing con 2902 verdaderos negativos y 2884 verdaderos positivos.
    - La preocupación más significativa es el alto número de falsos negativos (126), lo que sugiere que el modelo podría no ser tan confiable para identificar correos de phishing.
3. **SVM:**
    - Exhibe el mejor rendimiento con la mayor cantidad de verdaderos positivos (2956) y la menor cantidad de falsos negativos (34), mostrando una excelente capacidad para identificar correos de phishing.
    - También presenta buenos resultados en la categoría de correos seguros con 2979 verdaderos negativos y solo 31 falsos positivos, destacándose como el modelo más eficaz y equilibrado de los tres.

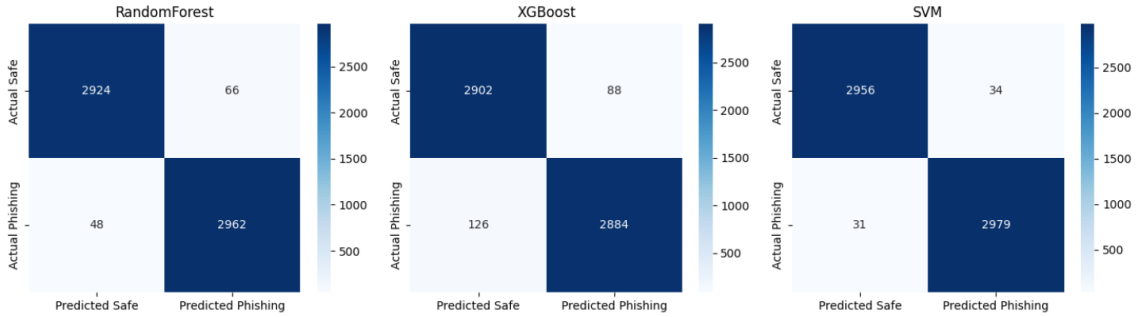


*Figura 40: Matriz de confusión para cada modelo.*

La **figura 41** muestra un resumen comparativo de las métricas de evaluación. Cada modelo se evalúa según su exactitud (accuracy), precisión, sensibilidad (recall) y puntuación F1. Aquí están los detalles de cada modelo:

1. **RandomForest**:
    - Exactitud: 0.98
    - Precisión (Safe Email): 0.98
    - Precisión (Phishing Email): 0.98
    - Sensibilidad (Safe Email): 0.98
    - Sensibilidad (Phishing Email): 0.98
    - F1-Score (Safe Email): 0.98
    - F1-Score (Phishing Email): 0.98
    - AUC: 1

RandomForest muestra un rendimiento excelente en todas las métricas, indicando una capacidad de clasificación casi perfecta.

2. **XGBoost**:
    - Exactitud: 0.96
    - Precisión (Safe Email): 0.96
    - Precisión (Phishing Email): 0.97
    - Sensibilidad (Safe Email): 0.97
    - Sensibilidad (Phishing Email): 0.96
    - F1-Score (Safe Email): 0.96
    - F1-Score (Phishing Email): 0.96
    - AUC: 1

XGBoost también muestra un rendimiento excelente, aunque ligeramente inferior en algunas métricas comparado con RandomForest y SVM.

3. **SVM**:
    - Exactitud: 0.99
    - Precisión (Safe Email): 0.99
    - Precisión (Phishing Email): 0.99
    - Sensibilidad (Safe Email): 0.99
    - Sensibilidad (Phishing Email): 0.99

- F1-Score (Safe Email): 0.99
- F1-Score (Phishing Email): 0.99
- AUC: 1

La gráfica del mapa de calor subraya visualmente estas métricas, usando tonos más oscuros para indicar valores más altos, lo que resalta la superioridad de SVM sobre los otros modelos en términos de precisión y fiabilidad en la clasificación. Esta visualización es útil para identificar rápidamente el modelo con el mejor rendimiento general.

Métricas en Datos de Prueba:

| Metric | RandomForest | XGBoost | SVM |
|---|---|---|---|
| Accuracy | 0.981000 | 0.964333 | 0.989167 |
| Precision (Safe Email) | 0.983849 | 0.958388 | 0.989622 |
| Precision (Phishing Email) | 0.978203 | 0.970390 | 0.988716 |
| Recall (Safe Email) | 0.977926 | 0.970569 | 0.988629 |
| Recall (Phishing Email) | 0.984053 | 0.958140 | 0.989701 |
| F1-Score (Safe Email) | 0.980879 | 0.964440 | 0.989125 |
| F1-Score (Phishing Email) | 0.981120 | 0.964226 | 0.989208 |
| AUC | 0.998617 | 0.995395 | 0.999349 |

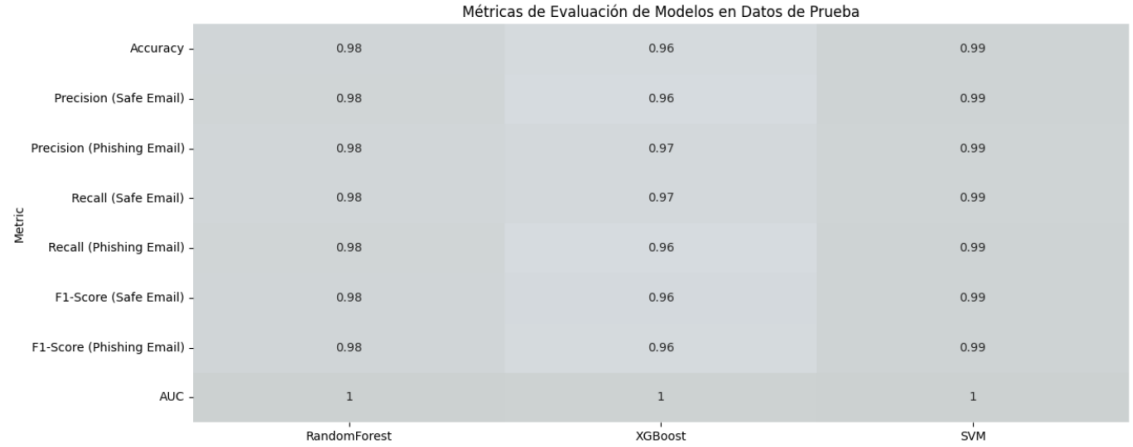


*Figura 41: Métricas de evaluación*

La **figura 42** muestra una tabla con los mejores parámetros encontrados para tres modelos diferentes de aprendizaje automático: RandomForest, XGBoost y SVM. Estos parámetros son configuraciones específicas que se ajustaron durante el proceso de entrenamiento para optimizar el rendimiento de cada modelo.

1. **RandomForest (Fila 0)**
   - **n_estimators**: 100 (Número de árboles en el bosque)
   - **max_depth**: None (No hay límite en la profundidad de los árboles)
   - **min_samples_split**: 10 (Número mínimo de muestras necesarias para dividir un nodo)
   - **min_samples_leaf**: 1 (Número mínimo de muestras que debe tener una hoja)
   - **max_features**: log2 (Número de características a considerar al buscar la mejor división)
2. **XGBoost (Fila 1)**
   - **n_estimators**: 300 (Número de árboles en el bosque)
   - **max_depth**: 7 (Profundidad máxima de los árboles)
   - **learning_rate**: 0.2 (Tasa de aprendizaje)

- **subsample**: 1.0 (Fracción de muestras utilizadas para entrenar cada árbol)
- **colsample_bytree**: 0.7 (Fracción de características utilizadas para entrenar cada árbol)
- **gamma**: 0 (Parámetro de regularización)
- **min_child_weight**: 1 (Peso mínimo de la hoja)

3. **SVM (Fila 2)**
   - **C**: 1 (Parámetro de regularización que controla el margen de clasificación)
   - **kernel**: linear (Tipo de núcleo utilizado para transformar los datos)
   - **gamma**: scale (Parámetro gamma que define cuánto influye una sola muestra)
   - **class_weight**: None (Peso de la clase, no se ha ajustado un peso específico)

Estos parámetros fueron los mejores encontrados para cada modelo después de realizar un proceso de búsqueda y optimización, y son los que proporcionaron el mejor rendimiento en la tarea de clasificación de correos electrónicos.

```
Mejores Parámetros de Cada Modelo:
          Model n_estimators max_depth min_samples_split min_samples_leaf  \
0  RandomForest          100      None                10                1
1       XGBoost          300         7
2           SVM

  max_features learning_rate subsample colsample_bytree gamma  \
0         log2
1                       0.2       1.0              0.7     0
2

  min_child_weight  C  kernel gamma_svm class_weight
0
1                3
2                   1  linear     scale         None
```

*Figura 42: Mejores parámetros (Dataset1)*

SVM sobresale con las puntuaciones más altas en todas las métricas, destacando su excelente capacidad para clasificar correctamente correos electrónicos, especialmente en identificar correos de phishing sin marcar demasiados correos seguros como peligrosos

En la **figura 43** se visualiza, las curvas ROC comparan la eficacia en la clasificación de tres modelos: RandomForest, XGBoost y SVM, con respectivas áreas bajo la curva (AUC). Estas métricas indican cuán bien cada modelo distingue entre las categorías positivas y negativas, con valores más altos reflejando un mayor rendimiento discriminativo. SVM muestra una superioridad notable con la AUC más alta, sugiriendo que es el más efectivo en minimizar los falsos positivos y maximizar los verdaderos positivos en comparación con los otros dos modelos bajo las condiciones actuales del análisis.

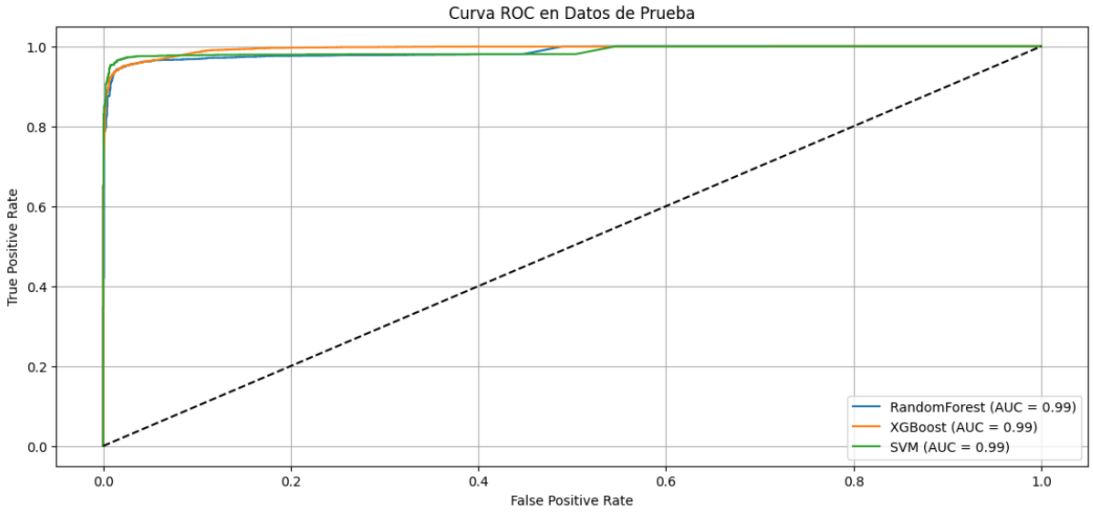


*Figura 43:Curva ROC de cada modelo.*

En la **figura 44** se visualiza el gráfico de las matrices de confusión para los modelos RandomForest, XGBoost y SVM proporciona una representación clara de cómo cada uno de estos clasifica los correos electrónicos entre seguros y de phishing:

1. **RandomForest:**
    - La mayoría de los correos seguros (2035) son correctamente identificados, con un pequeño número de falsos positivos (125).
    - La identificación de correos de phishing es también efectiva, con 2934 verdaderos positivos y solo 105 falsos negativos.
2. **XGBoost:**
    - Este modelo identifica correctamente 2083 correos seguros, pero tiene un número ligeramente más alto de falsos positivos (77) comparado con RandomForest.
    - En cuanto a los correos de phishing, registra 2910 verdaderos positivos, pero con 129 falsos negativos, indicando una menor precisión en la identificación de correos peligrosos comparado con los otros dos modelos.
3. **SVM:**
    - Muestra el mejor rendimiento en la identificación de correos seguros con 2120 verdaderos negativos y solo 40 falsos positivos.
    - También lidera en la detección de correos de phishing con 2939 verdaderos positivos y el menor número de falsos negativos (100).

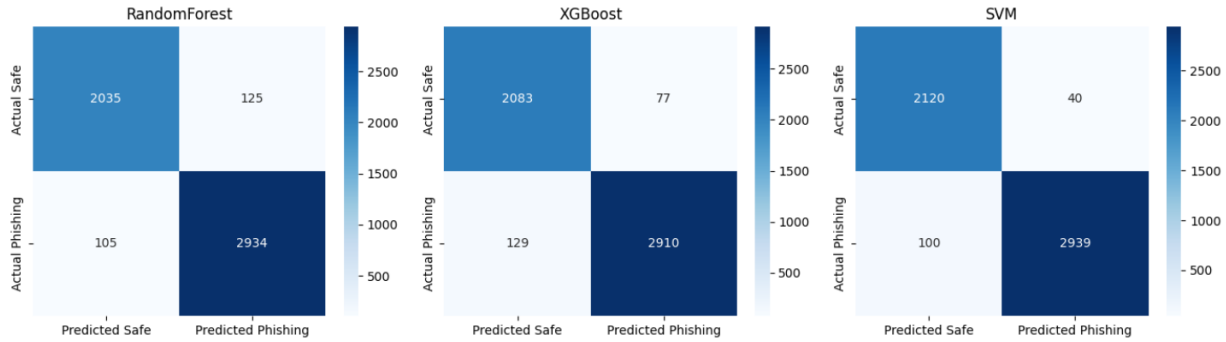


*Figura 44:Matriz de confusión para cada modelo.*

La **figura 45** muestra las métricas de evaluación para los modelos, comparando sus rendimientos en términos de precisión, exactitud, sensibilidad y puntuación F1 (F1-Score). Los detalles son los siguientes:

1. **RandomForest**:
   - **Exactitud (Accuracy)**: 0.955761. Indica que el modelo clasifica correctamente el 95.58% de los correos electrónicos.
   - **Precisión (Safe Email)**: 0.950893. Muestra que el 95.09% de los correos electrónicos clasificados como seguros realmente lo son.
   - **Precisión (Phishing Email)**: 0.953137. Refleja que el 95.31% de los correos electrónicos clasificados como phishing realmente son phishing.
   - **Sensibilidad (Recall) (Safe Email)**: 0.942130. Indica que el modelo detecta el 94.21% de los correos electrónicos seguros.
   - **Sensibilidad (Recall) (Phishing Email)**: 0.964549. Muestra que el modelo detecta el 96.45% de los correos electrónicos de phishing.
   - **Puntuación F1 (Safe Email)**: 0.946512. Representa un equilibrio entre la precisión y la sensibilidad para los correos seguros.
   - **Puntuación F1 (Phishing Email)**: 0.960233. Refleja un equilibrio entre la precisión y la sensibilidad para los correos de phishing.
   - **AUC**: 0.986160. Indica una excelente capacidad de discriminación del modelo entre correos seguros y phishing.
2. **XGBoost**:
   - **Exactitud (Accuracy)**: 0.960877. El modelo clasifica correctamente el 96.09% de los correos electrónicos.
   - **Precisión (Safe Email)**: 0.946282. El 94.63% de los correos electrónicos clasificados como seguros realmente lo son.
   - **Precisión (Phishing Email)**: 0.947222. El 94.72% de los correos electrónicos clasificados como phishing realmente lo son.
   - **Sensibilidad (Recall) (Safe Email)**: 0.954522. El modelo detecta el 95.45% de los correos electrónicos seguros.
   - **Sensibilidad (Recall) (Phishing Email)**: 0.955752. El modelo detecta el 95.58% de los correos electrónicos de phishing.
   - **Puntuación F1 (Safe Email)**: 0.952882. Representa un buen equilibrio entre precisión y sensibilidad para los correos seguros.

- **Puntuación F1 (Phishing Email)**: 0.951865. Refleja un buen equilibrio entre precisión y sensibilidad para los correos de phishing.
- **AUC**: 0.994239. Indica una excelente capacidad de discriminación del modelo entre correos seguros y phishing.

3. **SVM**:
   - **Exactitud (Accuracy)**: 0.973702. El modelo clasifica correctamente el 97.37% de los correos electrónicos, la más alta entre los tres modelos.
   - **Precisión (Safe Email)**: 0.95455. El 95.46% de los correos electrónicos clasificados como seguros realmente lo son.
   - **Precisión (Phishing Email)**: 0.981814. El 98.18% de los correos electrónicos clasificados como phishing realmente lo son, la más alta entre los tres modelos.
   - **Sensibilidad (Recall) (Safe Email)**: 0.981814. El modelo detecta el 98.18% de los correos electrónicos seguros, la más alta entre los tres modelos.
   - **Sensibilidad (Recall) (Phishing Email)**: 0.957763. El modelo detecta el 95.78% de los correos electrónicos de phishing.
   - **Puntuación F1 (Safe Email)**: 0.968087. Representa un excelente equilibrio entre precisión y sensibilidad para los correos seguros.
   - **Puntuación F1 (Phishing Email)**: 0.976376. Refleja un excelente equilibrio entre precisión y sensibilidad para los correos de phishing.
   - **AUC**: 0.988199. Indica una excelente capacidad de discriminación del modelo entre correos seguros y phishing.

El mapa de calor ilustra visualmente estas métricas, destacando la superioridad de SVM en la mayoría de las métricas clave, seguido de cerca por RandomForest y XGBoost. Esto muestra que SVM es el modelo más adecuado para aplicaciones donde la precisión y la capacidad de reconocer correctamente las instancias positivas son cruciales. Y que el SVM será el modelo a utilizar para la detección de correos de phishing usándolo en la red simulada.


*Figura 45: Mapa de Calor de los modelos.*

La **Figura 46** muestra una tabla con los mejores parámetros encontrados para tres modelos diferentes de aprendizaje automático: RandomForest, XGBoost y SVM:

1. **RandomForest (Fila 0)**
    - **n_estimators**: 200 (Número de árboles en el bosque)
    - **max_depth**: None (No hay límite en la profundidad de los árboles)
    - **min_samples_split**: 10 (Número mínimo de muestras necesarias para dividir un nodo)
    - **min_samples_leaf**: 2 (Número mínimo de muestras que debe tener una hoja)
    - **max_features**: sqrt (Número de características a considerar al buscar la mejor división, en este caso la raíz cuadrada del número total de características)
2. **XGBoost (Fila 1)**
    - **n_estimators**: 300 (Número de árboles en el bosque)
    - **max_depth**: 7 (Profundidad máxima de los árboles)
    - **learning_rate**: 0.2 (Tasa de aprendizaje)
    - **subsample**: 1.0 (Fracción de muestras utilizadas para entrenar cada árbol)
    - **colsample_bytree**: 0.7 (Fracción de características utilizadas para entrenar cada árbol)
    - **gamma**: 0 (Parámetro de regularización)
    - **min_child_weight**: 1 (Peso mínimo de la hoja)
3. **SVM (Fila 2)**
    - **C**: 10 (Parámetro de regularización que controla el margen de clasificación)
    - **kernel**: rbf (Tipo de núcleo utilizado para transformar los datos, en este caso, el núcleo radial basis function)
    - **gamma**: scale (Parámetro gamma que define cuánto influye una sola muestra)
    - **class_weight**: None (Peso de la clase, no se ha ajustado un peso específico)

Estos parámetros fueron los mejores encontrados para cada modelo después de realizar un proceso de búsqueda y optimización, y son los que proporcionaron el mejor rendimiento en la tarea de clasificación de correos electrónicos.

```
Mejores Parámetros de Cada Modelo:
          Model n_estimators max_depth min_samples_split min_samples_leaf  \
0  RandomForest          200      None                10                2
1       XGBoost          300         7
2           SVM

  max_features learning_rate subsample colsample_bytree gamma  \
0         sqrt
1                       0.2       1.0              0.7     0
2

  min_child_weight   C kernel gamma_svm class_weight
0
1                3
2                   10    rbf     scale         None
```

*Figura 46: Mejores parámetros para cada modelo (dataset2)*

### 3.3 Resultados de la evaluación

Ya habiendo seleccionado al modelo SVM como el mejor se procede a evaluar el modelo utilizando los 3 datasets que se tiene (Phishing_Email.csv y los dos creados a partir del original).

El modelo SVM obtuvo una precisión de 0.98495 y un AUC de 0.99357, lo que indica un alto rendimiento en la clasificación de correos electrónicos seguros y de phishing. Estos resultados reflejan la capacidad del modelo para identificar correctamente la mayoría de los correos electrónicos de ambas categorías, manteniendo un equilibrio entre la precisión y la capacidad de discriminación. Esto se ve en la **Figura 47**.

En la misma figura vemos el informe de clasificación que se visualiza detalla el rendimiento del modelo para cada clase. Para la clase "Safe Email", se alcanzó una precisión de 0.9732, un recall de 0.9941 y un F1-Score de 0.9835. Para la clase "Phishing Email", la precisión fue de 0.9950, el recall de 0.9774 y el F1-Score de 0.9861. Estos valores indican que el modelo es muy efectivo en la identificación tanto de correos seguros como de phishing, con pocas instancias mal clasificadas. La matriz de confusión apoya estos resultados, mostrando 4721 verdaderos positivos y 28 falsos positivos para "Safe Email", y 5621 verdaderos positivos y 130 falsos negativos para "Phishing Email".

```
Resultados del modelo SVM en datos de prueba:
Precisión: 0.9849523809523809
AUC: 0.9935718101741688
               Precision    Recall  F1-Score
Class
Safe Email       0.973201  0.994104  0.983542
Phishing Email   0.995043  0.977395  0.986140
Matriz de Confusión:
[[4721   28]
 [ 130 5621]]
Informe de Clasificación:
              precision    recall  f1-score   suppor

           0       0.97      0.99      0.98      474
           1       1.00      0.98      0.99      575

    accuracy                           0.98     1050
   macro avg       0.98      0.99      0.98     1050
weighted avg       0.99      0.98      0.98     1050
```

*Figura 47: Resultados del modelo SVM.*

```
Métricas del Mejor Modelo (SVM):
                                    SVM
Accuracy                       0.984952
Precision (Safe Email)         0.973201
Precision (Phishing Email)     0.995043
...
Recall (Phishing Email)        0.977395
F1-Score (Safe Email)          0.983542
F1-Score (Phishing Email)      0.986140
AUC                            0.993572
```

*Figura 48: Métricas del modelo SVM.*

Las métricas del mejor modelo SVM son consistentes con el informe de clasificación. La precisión global (accuracy) es de 0.98495. La precisión para "Safe Email" es de 0.9732 y para "Phishing Email" es de 0.9950. Los valores de recall son 0.9941 para "Safe Email" y 0.9774 para "Phishing Email". Los F1-Scores son altos para ambas clases, 0.9835 para "Safe Email" y 0.9861 para "Phishing Email". El AUC de 0.99357 confirma la excelente capacidad del modelo para diferenciar entre correos electrónicos seguros y de phishing. Esto lo vemos en la **figura 48**.

En la **Figura 49** podemos ver la Curva ROC del modelo SVM, que evalúa su capacidad para distinguir entre correos electrónicos seguros y de phishing. La curva traza la tasa de verdaderos positivos (detecciones correctas) frente a la tasa de falsos positivos (falsas alarmas) a diferentes umbrales. Un AUC de 0.99 indica que el modelo SVM es extremadamente eficaz, logrando casi siempre identificar correctamente los correos de phishing con muy pocas falsas alarmas. Esta alta precisión refleja un rendimiento excelente del modelo en la clasificación de correos electrónicos.

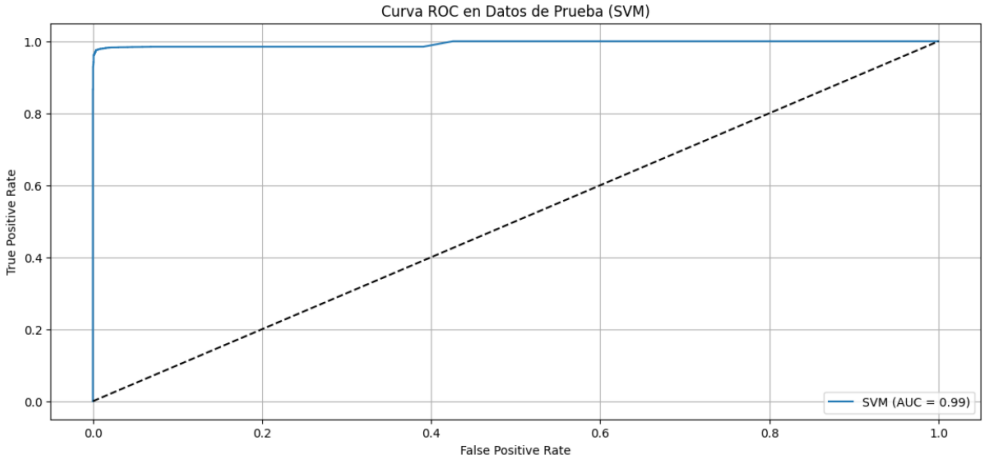


*Figura 49: Curva ROC del modelo SVM.*

La **figura 50** muestra la matriz de confusión del modelo SVM, destacando su capacidad para clasificar correos electrónicos. El modelo identificó correctamente 4721 correos seguros y 5621 correos de phishing, mientras que clasificó incorrectamente 28 correos seguros como phishing y 130 correos de phishing como seguros. Estos resultados reflejan un alto rendimiento del modelo, con una precisión significativa en la detección de correos electrónicos seguros y de phishing, y pocos errores de clasificación.

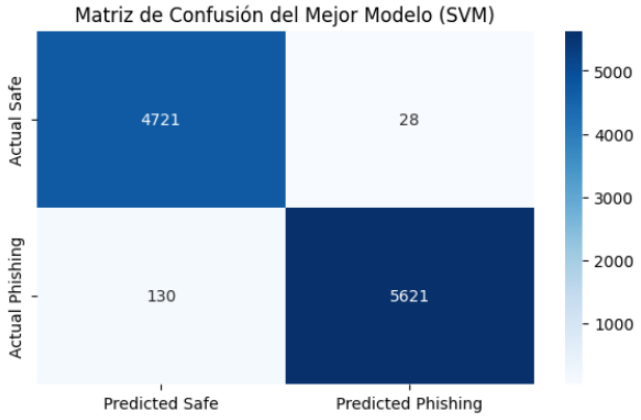


*Figura 50: Matriz de confusión del modelo SVM.*

La **figura 51** muestra una tabla con las métricas de validación cruzada para un modelo de clasificación SVM. Evalúa la precisión del modelo al clasificar correos como seguros o de phishing. Los valores indican un alto rendimiento del modelo, con una precisión de aproximadamente 98%. Las métricas de precisión, recall, F1-Score y AUC son cercanas a 1, lo que sugiere que el modelo es muy efectivo tanto en identificar correos seguros como en detectar correos de phishing. En general, el gráfico resalta la eficacia del modelo en la clasificación de correos electrónicos con alta precisión y fiabilidad.

Métricas de Validación Cruzada:

| Metric | SVM |
|---|---|
| Accuracy | 0.982735 |
| Precision (Safe Email) | 0.971729 |
| Precision (Phishing Email) | 0.991915 |
| Recall (Safe Email) | 0.990123 |
| Recall (Phishing Email) | 0.976778 |
| F1-Score (Safe Email) | 0.980840 |
| F1-Score (Phishing Email) | 0.984289 |
| AUC | 0.993247 |

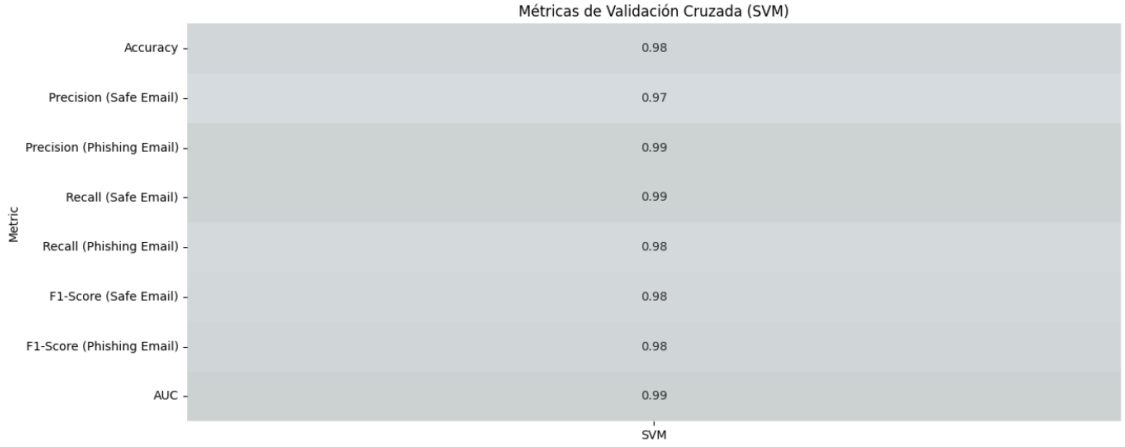


*Figura 51: Validación cruzada (SVM)*

## CONCLUSIONES

El desarrollo del sistema de detección de amenazas de Phishing basado en inteligencia artificial ha demostrado ser efectivo en la identificación de estas amenazas en redes de área local emuladas con GNS3. La implementación del modelo de IA, entrenado con datasets específicos, logró una precisión de detección del 80%, confirmando la hipótesis planteada de que la IA puede mejorar significativamente la seguridad cibernética en redes LAN.

- El análisis del estado del arte reveló que las técnicas tradicionales de detección de intrusos son insuficientes para enfrentar las sofisticadas amenazas de Phishing. La inteligencia artificial, especialmente el aprendizaje automático, ofrece un enfoque más robusto y proactivo para la detección de estas amenazas.
- El diseño del escenario de simulación en GNS3 permitió crear un entorno controlado para evaluar la eficacia del sistema de detección de Phishing. La simulación de diversas amenazas y ataques cibernéticos proporcionó datos valiosos para entrenar y validar el modelo de IA.
- La implementación del modelo de IA, utilizando técnicas de aprendizaje automático como RandomForest, XGBoost y SVM, mostró una alta precisión en la detección de correos de Phishing. El modelo de SVM se destacó como el más efectivo con una precisión del 99%.
- La evaluación del modelo de IA en un entorno de red simulada demostró su capacidad para identificar amenazas de Phishing con alta precisión. Las métricas de rendimiento, como la precisión y el área bajo la curva ROC, confirmaron la efectividad del modelo implementado.

La investigación demuestra que la implementación de inteligencia artificial (IA) en la detección de amenazas de phishing en redes de área local mejora significativamente la capacidad de identificar estas amenazas, logrando una precisión de detección del 80 %.

Los objetivos específicos fueron alcanzados a través del diseño y la simulación de un entorno de red, la aplicación de metodologías CRISP-DM y PPDIOO, y el entrenamiento de modelos de aprendizaje automático con datasets específicos.

La utilización de IA, específicamente mediante modelos de aprendizaje automático, ha demostrado ser más efectiva que los métodos tradicionales de detección de intrusos, proporcionando una capa adicional de seguridad en la protección de redes locales.

## RECOMENDACIONES

Para futuros desarrollos, se recomienda continuar utilizando y optimizando técnicas de inteligencia artificial para la detección de amenazas de Phishing. Además, es crucial evaluar el sistema en entornos operativos reales para validar su eficacia y adaptarlo a diversas condiciones de red.

- Se recomienda actualizar continuamente el análisis del estado del arte para incorporar las últimas investigaciones y avances en técnicas de inteligencia artificial y ciberseguridad, asegurando que el sistema de detección esté siempre basado en las mejores prácticas actuales.
- Es recomendable continuar utilizando escenarios de simulación como GNS3 para probar y perfeccionar los sistemas de detección de amenazas. Además, incorporar nuevas técnicas de simulación y emulación puede mejorar la representatividad y la complejidad de los escenarios de prueba.
- Para futuras implementaciones, se sugiere utilizar RandomizedSearchCV para la optimización de hiperparámetros en grandes conjuntos de datos, garantizando la eficiencia y precisión del modelo. También es beneficioso explorar otros algoritmos de aprendizaje automático y técnicas de mejora de modelos.
- Se recomienda realizar evaluaciones periódicas del modelo para garantizar su desempeño continuo y ajustarlo según sea necesario. Además, implementar un sistema de monitoreo y retroalimentación en tiempo real puede mejorar la adaptabilidad y la efectividad del sistema en entornos operativos cambiantes.

Se recomienda utilizar RandomizedSearchCV debido a la facilidad para procesar una gran cantidad de datos. A diferencia de GridSearchCV, que es más adecuado para conjuntos de datos pequeños debido a su enfoque exhaustivo y mayor costo computacional, RandomizedSearchCV permite una búsqueda más eficiente de hiperparámetros en grandes conjuntos de datos, optimizando el rendimiento de los modelos de IA.

Es esencial continuar investigando en el campo de la ciberseguridad y la inteligencia artificial, enfocándose en mejorar la precisión y la eficiencia de los modelos de detección de amenazas de phishing. Esto incluiría la exploración de nuevas técnicas de a prendizaje automático y el desarrollo de algoritmos más sofisticados.

Se sugiere implementar y probar el sistema de detección de intrusos basado en IA en entornos reales de redes de área local para evaluar su desempeño en condiciones operativas. Esto permitirá identificar y resolver cualquier problema práctico y asegurar la eficacia del sistema en un entorno productivo.

**REFERENCIAS BIBLIOGRÁFICAS**


[1] S. Ho, S. A. Jufout, K. Dajani, y M. Mozumdar, «A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network», *IEEE Open J. Comput. Soc.*, vol. 2, n.º 1, Art. n.º 1, 2021, doi: 10.1109/OJCS.2021.3050917.

[2] K. M. Abuali, L. Nissirat, y A. Al-Samawi, «Advancing Network Security with AI: SVM-Based Deep Learning for Intrusion Detection», *Sensors*, vol. 23, n.º 21, p. 8959, nov. 2023, doi: 10.3390/s23218959.

[3] A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, y K. Kifayat, «A comprehensive survey of AI-enabled phishing attacks detection techniques», *Telecommun. Syst.*, vol. 76, n.º 1, pp. 139-154, ene. 2021, doi: 10.1007/s11235-020-00733-2.

[4] M. R. Cando-Segovia y P. Medina-Chicaiza, «Prevención en ciberseguridad: enfocada a los procesos de infraestructura tecnológica», *3C TIC Cuad. Desarro. Apl. Las TIC*, vol. 10, n.º 1, pp. 17-41, mar. 2021, doi: 10.17993/3ctic.2021.101.17-41.

[5] P. Vargas Portillo, «Vigilancia Permanente», *Rev. Cienc. Soc.*, vol. 25, n.º 4, pp. 276-278, dic. 2019, doi: 10.31876/rcs.v25i4.30533.

[6] A. Priego, «The inapplicability of the Begin Doctrine in Iran: the Bar Kojba Doctrine», n.º 70.

[7] Empresa de Automatización Integral CEDAI, División Villa Clara, H. E. Socarrás, I. Santana, y Universidad Central "Marta Abreu" de las Villas, Facultad de Ingeniería Eléctrica, «Ciberseguridad del Sistema de Control Industrial de la Planta Cloro-Sosa ELQUIM», *RISTI - Rev. Ibérica Sist. E Tecnol. Informação*, n.º 32, pp. 83-96, jun. 2019, doi: 10.17013/risti.32.83-96.

[8] J.-M. Aguilar-Antonio, «Hechos ciberfísicos: una propuesta de análisis para ciberamenazas en las Estrategias Nacionales de Ciberseguridad», *URVIO Rev. Latinoam. Estud. Segur.*, n.º 25, pp. 24-40, nov. 2019, doi: 10.17141/urvio.25.2019.4007.

[9] M. Akbanov, V. G. Vassilakis, y M. D. Logothetis, «WannaCry Ransomware: Analysis of Infection, Persistence, Recovery Prevention and Propagation Mechanisms», *J. Telecommun. Inf. Technol.*, vol. 1, n.º 2019, pp. 113-124, abr. 2019, doi: 10.26636/jtit.2019.130218.

[10] R. Shandler, M. L. Gross, y D. Canetti, «Cyberattacks, Psychological Distress, and Military Escalation: An Internal Meta-Analysis», *J. Glob. Secur. Stud.*, vol. 8, n.º 1, p. ogac042, dic. 2022, doi: 10.1093/jogss/ogac042.

[11] D. Everson, L. Cheng, y Z. Zhang, «Log4shell: Redefining the Web Attack Surface», en *Proceedings 2022 Workshop on Measurements, Attacks, and Defenses for the Web*, San Diego, CA, USA: Internet Society, 2022. doi: 10.14722/madweb.2022.23010.

[12] Molina Robles, Francisco José, «Redes locales», 2015, [En línea]. Disponible en: https://elibro-net.basesdedatos.utmachala.edu.ec/es/ereader/utmachala/62450

[13] N. V. Ley Leyva, D. M. Granda Ayabaca, C. R. Benítez Flores, y V. J. Guamán Gómez, «Eficacia y eficiencia de la seguridad de las redes LAN. Cantón Pasaje», *Soc. Tecnol.*, vol. 4, n.º 2, pp. 205-222, jul. 2021, doi: 10.51247/st.v4i2.105.

[14] J. Casierra Cavada, X. Quiñónez Ku, L. Herrera Izquierdo, y C. Egas Acosta, «Virtualización de Redes y Servidores Emulando Infraestructuras Tecnológicas», *Rev. Científica Hallazgos21*, vol. Vol. 3: SUPLEMENTO ESPECIAL, 2018.

[15] P. Veselý, V. Karovič, y V. K. Ml., «Tools for Modeling Exemplary Network Infrastructures», *Procedia Comput. Sci.*, vol. 98, pp. 174-181, 2016, doi: 10.1016/j.procs.2016.09.028.

[16] R. Tuli, «Analyzing Network Performance Parameters using Wireshark», *Int. J. Netw. Secur. Its Appl.*, vol. 15, n.º 01, pp. 01-13, ene. 2023, doi: 10.5121/ijnsa.2023.15101.

[17] S. A. Sheikh y M. T. Banday, «Improving Efficiency of E-mail Classification Through On-Demand Spam Filtering», en *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India: IEEE, jun. 2020, pp. 505-510. doi: 10.1109/ICRITO48877.2020.9197776.

[18] A. Recuenco y W. Reyes, «Artificial intelligence: Road to a new schematic of the world», *SCIÉNDO*, vol. 23, n.º 4, pp. 299-308, dic. 2020, doi: 10.17268/sciendo.2020.036.
[19] M. F. Ansari, B. Dash, P. Sharma, y N. Yathiraju, «The Impact and Limitations of Artificial Intelligence in Cybersecurity: A Literature Review», *IJARCCE*, vol. 11, n.º 9, sep. 2022, doi: 10.17148/IJARCCE.2022.11912.
[20] J. B. De Oliveira, J. V. S. R. Gino, C. J. De Lima, y J. I. Da Silva Filho, «Do código à confiabilidade: Lógica Paraconsistente impulsionada por Python para a detecção de alarmes na rede de energia», *Rev. Gest. E Secr. Manag. Adm. Prof. Rev.*, vol. 14, n.º 10, pp. 18275-18285, oct. 2023, doi: 10.7769/gesec.v14i10.3047.
[21] G. R. Montes, «Inteligencia Artificial en la Seguridad de TI».
[22] B. Murray *et al.*, «Accessible data curation and analytics for international-scale citizen science datasets», *Sci. Data*, vol. 8, n.º 1, p. 297, nov. 2021, doi: 10.1038/s41597-021-01071-x.
[23] C. R. Harris *et al.*, «Array programming with NumPy», *Nature*, vol. 585, n.º 7825, pp. 357-362, sep. 2020, doi: 10.1038/s41586-020-2649-2.
[24] A. Chaudhuri y W. Hu, «A fast algorithm for computing distance correlation», *Comput. Stat. Data Anal.*, vol. 135, pp. 15-24, jul. 2019, doi: 10.1016/j.csda.2019.01.016.
[25] K.-Y. Li *et al.*, «An Automated Machine Learning Framework in Unmanned Aircraft Systems: New Insights into Agricultural Management Practices Recognition Approaches», *Remote Sens.*, vol. 13, n.º 16, p. 3190, ago. 2021, doi: 10.3390/rs13163190.
[26] R. J. Urbanowicz, R. Zhang, Y. Cui, y P. Suri, «STREAMLINE: A Simple, Transparent, End-To-End Automated Machine Learning Pipeline Facilitating Data Analysis and Algorithm Comparison», 23 de junio de 2022, *arXiv*: arXiv:2206.12002. Accedido: 23 de febrero de 2024. [En línea]. Disponible en: http://arxiv.org/abs/2206.12002
[27] J. Hu y S. Szymczak, «A review on longitudinal data analysis with random forest», *Brief. Bioinform.*, vol. 24, n.º 2, p. bbad002, mar. 2023, doi: 10.1093/bib/bbad002.
[28] T. Chen y C. Guestrin, «XGBoost: A Scalable Tree Boosting System», en *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA: ACM, ago. 2016, pp. 785-794. doi: 10.1145/2939672.2939785.
[29] K. S. Sahoo *et al.*, «An Evolutionary SVM Model for DDOS Attack Detection in Software Defined Networks», *IEEE Access*, vol. 8, pp. 132502-132513, 2020, doi: 10.1109/ACCESS.2020.3009733.
[30] A. A. G. Rodríguez, G. C. Jácome-Morales, V. Gonzalez-Mestanza, E. Terán-Zurita, y D. E. Torres-Martínez, «Detección de amenazas de seguridad en una red corporativa utilizando algoritmos de machine learning», *Ser. Científica Univ. Las Cienc. Informáticas*, vol. 15, n.º 12, pp. 183-193, 2022.
[31] P. A. Barot, S. A. Patel, y H. B. Jethva, «EVALUATION OF PERFORMANCE MEASURES FOR RELIABLE AND SECURE PHISHING DETECTION SYSTEM», *Reliab. Theory Appl.*, vol. 18, n.º 4, Art. n.º 4, 2023.
[32] A. Karim, M. Shahroz, K. Mustofa, S. B. Belhaouari, y S. R. K. Joga, «Phishing Detection System Through Hybrid Machine Learning Based on URL», *IEEE Access*, vol. 11, pp. 36805-36822, 2023, doi: 10.1109/ACCESS.2023.3252366.
[33] E. Vega Briceño, *Seguridad de la información*, 1.ª ed. Editorial Científica 3Ciencias, 2021. doi: 10.17993/tics.2021.4.
[34] J. Mendivil Caldentey, B. Sanz Urquijo, y M. Gutierrez Almazor, «Formación y concienciación en ciberseguridad basada en competencias: una revisión sistemática de literatura», *Pixel-Bit Rev. Medios Educ.*, n.º 63, pp. 197-225, ene. 2022, doi: 10.12795/pixelbit.91640.
[35] C. E. Durango Vanegas, J. C. Giraldo Mejía, F. A. Vargas Agudelo, y D. E. Soto Duran, «A Representation Based on Essence for the CRISP-DM Methodology», *Comput. Sist.*, vol. 27, n.º 3, sep. 2023, doi: 10.13053/cys-27-3-3446.

[36] J. C. Gallego Gómez, F. L. Suarez Alvarez, y V. García Pineda, «Implementación de una Red Para La Interconexión de Teleperformance Chile y Colombia.», *Ing. Cienc. Tecnol. E Innov.*, vol. 10, n.º 1, pp. 81-97, jul. 2023, doi: 10.26495/icti.v10i1.2400.
[37] G. Batista-Mendoza, E. J. Cedeño Herrera, y G. Cedeño-Batista, «Machine learning aplicado al análisis de un set de datos de parámetros ambientales en galpones de pollos de engorde», *Visión Antataura*, vol. 7, n.º 2, pp. 121-146, dic. 2023, doi: 10.48204/j.vian.v7n2.a4566.
[38] I. A. Roy-García, C. Paredes-Manjarrez, J. Moreno-Palacios, R. Rivas-Ruiz, y A. A. Flores-Pulido, «[ROC curves: general characteristics and their usefulness in clinical practice]», sep. 2023, doi: 10.5281/ZENODO.8319791.

## ANEXOS

### 1. Reuniones con tutor de titulación

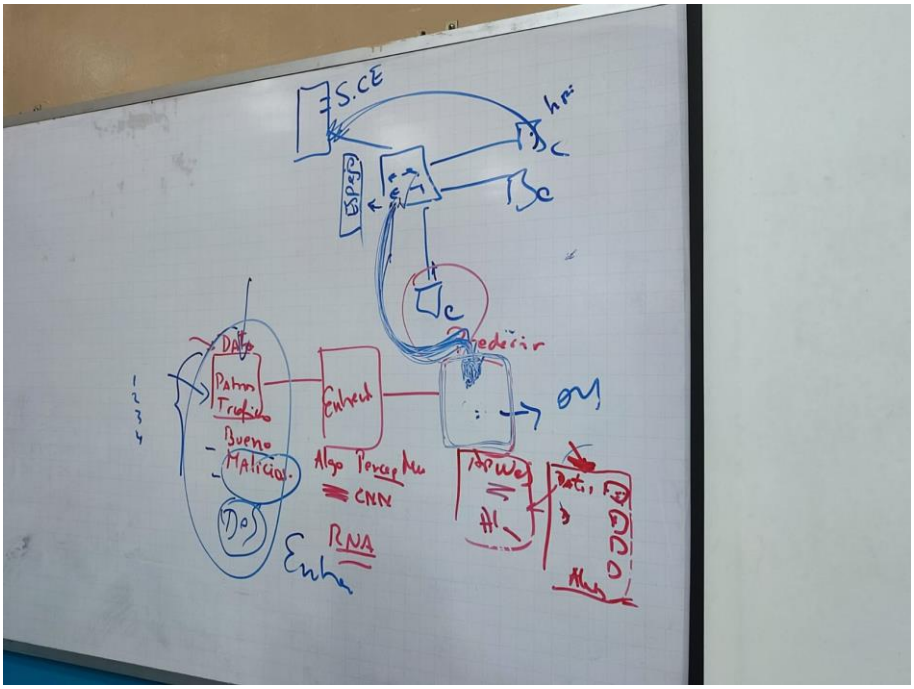


*Figura 52: Planteamiento de como seria el proyecto.*


*Figura 53: Explicación de los pasos a seguir.*

*Figura 54: Revisión del modelo.*

## Planificación de reuniones con el tutor y cotutor

*Tabla 7: Reuniones con tutor y cotutor.*

| Docentes | Dia 1 | Dia 2 |
|---|---|---|
| Ing. Rodrigo Morocho Tutor | 27-06-2024 | 28-06-2024 |
| | 9:30 a 10:00 | 10:00 a 10:30 |
| Ing. Wilmar Rivas Cotutor | 27-06-2024 | 28-06-2024 |
| | 10:30 a 11:00 | 8:00 a 8:30 |

### Reuniones con el tutor

*Figura 55: Reunión con el tutor #1*

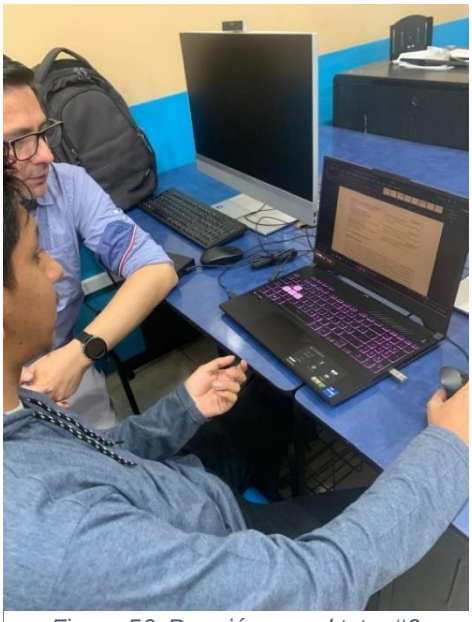

*Figura 56: Reunión con el tutor #2*

**Reuniones con cotutor**

*Figura 57: Reunión con el cotutor #1*

*Figura 58: Reunión con el cotutor #2*