UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA INGENIERÍA DE SISTEMAS

TESIS DE GRADO

“MODELO DE DETECCION Y REDUCCION DE ATAQUES PHISHING MEDIANTE TECNICAS DE MACHINE LEARNING”

Para Optar al Título de Licenciatura en Ingeniería de Sistemas

MENCIÓN: INFORMATICA Y COMUNICACIONES

Postulante: Univ. Gabriel Omar Mamani Camayo

Tutor Metodológico: Ing. Dionicio Henry Pacheco Ríos

Tutor Revisor: M. Sc. Lic Zara Yujra Cama

Tutor Especialista: Ing. Lidia Maxima Rodríguez Choque

EL ALTO – BOLIVIA

2023

DECLARACION JURADA
DE AUTENTICIDAD Y RESPONSABILIDAD

Yo Gabriel Omar Mamani Camayo estudiante con C.I. 10920696 LP mediante la presente declaro de manera pública que la propuesta de **TESIS DE GRADO** titulado **"MODELO DE DETECCION Y REDUCCION DE ATAQUES PHISHING MEDIANTE TECNICAS DE MACHINE LEARNING"** es original, siendo resultado de mi trabajo personal y no constituye una copia o replica de trabajos similares elaborados,

Autorizo la publicación del resumen de mi propuesta en internet y me comprometo a responder a todos los cuestionarios que se desprenden de su lectura.

Asimismo, me hago responsable ante la universidad o terceros, de cualquiera irregularidad o daño que se pudiera ocasionar, por el incumplimiento de lo declarado.

De identificarse falsificación, plagio, fraude o que la **TESIS DE GRADO** haya sido publicado anteriormente; asumo las consecuencias y sanciones de mi acción se deriven responsabilizándome por todas las cargas legales que se deriven a ello sometiéndome a las normas establecidas y vigentes de la carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

El Alto, junio del 2023

Gabriel Omar Mamani Camayo

C.I. 10920696 LP

Email: gs59879@gmail.com

DEDICATORIA

Queridos padres,

Hoy, al culminar mi tesis de grado, me gustaría dedicar este logro a dos personas especiales que han sido mi fuente constante de apoyo, aliento y amor incondicional: ustedes, mis queridos padres.

A través de los años, han sido testigos de mi crecimiento, mis altibajos y mis sacrificios. Han compartido mis alegrías y me han animado en mis momentos de duda. Su confianza en mí nunca vaciló, incluso cuando yo mismo dudaba de mis propias capacidades.

Esta tesis de grado es un testimonio de su inmenso amor y apoyo. Cada página escrita, cada experimento realizado y cada descubrimiento alcanzado ha sido impulsado por su fe inquebrantable en mí. Han sido mi roca en los momentos difíciles y mi faro en los momentos de incertidumbre.

Hoy, en este hito académico, quiero expresar mi profundo agradecimiento por todo lo que han hecho por mí. Su sacrificio, dedicación y esfuerzo han allanado el camino para que pueda alcanzar mis metas. Sin su amor incondicional y su guía constante, esta tesis de grado no habría sido posible.

Gracias por ser mis padres, mis mentores y mis mayores admiradores. Su influencia en mi vida va más allá de las palabras y espero poder retribuirles con éxito, felicidad y gratitud en el futuro.

Con todo mi amor y agradecimiento,

Gabriel Omar Mamani Camayo

Agradecimiento

Agradezco a Dios por siempre cuidarme, guiar mi camino, por escuchar mis oraciones y darme la fortaleza e inteligencia para culminar con mi trabajo de grado.

A mi familia por todo el apoyo incondicional que me brindaron a lo largo del proceso de mi carrera.

A la Ing. Lidia Rodríguez Choque por su paciencia por sus consejos y su tiempo para la revisión del trabajo de grado.

Al Ing. Dionicio Henry Pacheco Ríos por sus consejos enseñanza, paciencia ytiempo para el desarrollo aplicativo del trabajo de grado.

A la M. Sc. Zara Yujra Cama por brindar su conocimiento en el ámbito médico para la realización del trabajo de grado

Gabriel Omar Mamani Camayo

INDICE

Paginas

INDICE DE TABLAS

Paginas

INDICE DE FIGURAS

Paginas

RESUMEN

El estudio se centra en la problemática relacionada con el trabajo, específicamente en la detección y reducción de ataques phishing mediante el uso de técnicas de aprendizaje automático. Otro factor significativo que ha influido en este ámbito es la evolución tecnológica en los últimos años, lo cual ha generado la necesidad de crear un modelo de detección capaz de identificar los ataques fraudulentos perpetrados por cibercriminales. Dado el enfoque en las URL, que en la actualidad son ampliamente utilizadas por personas de todo el mundo como medio de comunicación y ayuda para resolver problemas cotidianos, surge la necesidad de proponer una solución que abarque la implementación y el estudio de una disciplina tecnológica interesante: la detección de ataques fraudulentos.

Para desarrollar el modelo, se emplearon técnicas de aprendizaje automático y se utilizaron herramientas de desarrollo, entre las que destacan Python y Anaconda. Python es un lenguaje de programación de alto nivel que se eligió debido a su sintaxis sencilla y a las numerosas funciones que facilitan el desarrollo del modelo. Por otro lado, Anaconda es un programa que permite realizar entrenamientos para redes neuronales y proporciona paquetes y funciones que agilizan el proceso de creación del modelo. Además, se aplicaron metodologías tanto de predicción con redes neuronales como de ingeniería de software, e incluso se emplearon metodologías para evaluar la calidad del prototipo.

Palabras Claves: Machine Learning, Ataques Phishing, Modelo.

ABSTRACT

The study focuses on the issue related to work, specifically on the detection and reduction of phishing attacks using machine learning techniques. Another significant factor that has influenced this field is the technological evolution in recent years, which has created the need to develop a detection model capable of identifying fraudulent attacks carried out by cybercriminals. Given the emphasis on URLs, which are widely used by people around the world as a means of communication and assistance in solving everyday problems, there is a need to propose a solution that encompasses the implementation and study of an interesting technological discipline: fraudulent attack detection.

To develop the model, machine learning techniques were employed, and development tools, particularly Python and Anaconda, were used. Python is a high-level programming language that was chosen for its simple syntax and numerous functions that facilitate model development. On the other hand, Anaconda is a program that allows for training neural networks and provides packages and functions that streamline the model creation process. Additionally, methodologies for neural network prediction, software engineering, and even prototype quality evaluation were applied.

Keywords: *Machine Learning, Phishing Attacks, Model.*

GLOSARIO DE ABREVIACIONES

API. Interfaz de programación de Aplicaciones.

ISO. Organización internacional de Normalización que se aplica a los productos yservicios.

SGSI. Sistema de Gestión de Seguridad de la Información.

HTML. Lenguaje de Marcado.

RAM. Random Access Memory (Memoria de Acceso Aleatorio).

SO. Sistema Operativo.

SGBD. Sistema de Gestión de Base de Datos.

M.V.C. Modelo vista controlador.

LMD. Lenguaje de manipulación de datos.

COCOMO. COnstructive COst MOdel (Modelo constructivo de costos). IEC. Comisión Electrotécnica Internacional

CAPÍTULO I

MARCO PRELIMINAR

1.1. INTRODUCCION

En los últimos años, la evolución de la tecnología de información ha ocasionado que el software se encuentre involucrado cada vez más en los productos y servicios de uso cotidiano como la salud, educación, gobernanza, creación y expansión de empresas (Centeno, 2020).

Esto ha provocado que la demanda sea cada vez más alta (Olmedo, 2020), generando un crecimiento sin precedentes (97%) en la utilización del software y dispositivos electrónicos. De manera similar el Instituto Nacional de Estadísticas (INE) señala que el 78.7% de los hogares disponen acceso al Internet, lo que ocasiona que las personas sean dependientes del software para el diario vivir y la utilización en el desarrollo productivo de empresas, gobiernos, instituciones educativas, entre otros.

La suplantación de identidad es una forma de robo de información personal en línea. Los phishers, por ejemplo, usan Ingeniería Social para robar los datos de identidad propia de las víctimas y las credenciales de cuentas financieras. Otra forma de ataque de Ingeniería Social consiste en utilizar correos electrónicos ilegítimos para atraer a víctimas inocentes a sitios web falsos con la intensión de engañar a los usuarios a divulgar datos personales como números de tarjetas de crédito, nombres de usuarios de cuentas, contraseñas y números de seguridad social (Wang, 2019).

El aprendizaje automático o mejor conocido como machine Learning es un método de análisis de datos que automatiza la construcción de modelos analíticos. Es una rama de la inteligencia artificial basada en la idea de que los sistemas pueden aprender de datos, identificar patrones y tomar decisiones con mínima intervención humana (Converguers, 2017).

Donde podemos utilizar modelos para actuar contra sin fin de problemas que hoy en día conocemos y van apareciendo a cada momento. Uno de los más llamativos modelos que tiene machine Learning es la predicción, el cual puede predecir a futuro los problemas que vienen y actuar de manera automática, también puede aprender del problema y así poder usar patrones para poder resolverlos.

1.2. ANTECEDENTES

Se describen algunos proyectos realizados a nivel internacional y nacionales, que tienen puntos afines con la temática tratada en este trabajo.

1.2.1. Internacionales

- (Fuertes Díaz, 2020) “Detección y Mitigación de ataques de ingeniería social tipo Phishing utilizando minería de datos” consiste en una investigación sobre la detección de ingeniería social, tanto manual como automática de sitios web fraudulentos, una de las metodologías de minería de datos que se realizó para esta investigación es el CRISP-DM, La metodología relaciona los usuarios según las variables de estudio.
- (Ruiz, 2019)“Detección de malware, métodos estadísticos y machine Learning” consiste en proponer una solución que facilite en reconocimiento de software malicioso, así como la escalabilidad del sistema utilizando aprendizaje automático y una de las metodologías utilizadas fue unsupervised machine Learning, este tipo de algoritmos son utilizados cuando disponemos básicamente de un conjunto de datos de entrada. Donde las herramientas fueron tensorflow, Python, y java.

1.2.2. Nacionales

- (Rojas, 2018) “DETECCIÓN DE SQL INJECTION BASADA EN RANDOM FOREST” consiste en que Los ataques de SQL INJECTION ante este tipo de amenazas surgieron diferentes tipos de soluciones, estas soluciones basadas en firewall, antivirus y herramientas exclusivamente para detectar este tipo de amenaza. Utiliza la metodología de KDD y las herramientas utilizadas para la investigación fueron Sql server, Algoritmos, Python.etc.

- (GUARACHI, 2015) “MODELO DE SEGURIDAD EN LAS APLICACIONES WEB” La presente investigación es referente a las aplicaciones web que en la actualidad son de uso general en todos los ámbitos donde se utilizan la tecnología web. Tiene como énfasis de realizar un análisis de la seguridad en las aplicaciones web, en base a ello se plantea el Modelo de Seguridad en las Aplicaciones Web. está compuesto principalmente por cuatro capas; Capa de Presentación Visual, Capa lógica de negocio, Capa servicio de datos web y Capa de alojamiento web.

1.3. PLANTEAMIENTO DEL PROBLEMA

En la actualidad, la seguridad cibernética se ha vuelto una preocupación cada vez mayor debido a los constantes ataques informáticos. Uno de los ataques más comunes es el phishing, el cual consiste en engañar a los usuarios para obtener información confidencial o dinero. A medida que las tecnologías evolucionan, los atacantes están encontrando nuevas formas de engañar a los usuarios y obtener acceso no autorizado a sistemas y datos importantes (ITESEC, 2020).

Y que según El informe de Ciberseguridad Tendencias 2020 menciona la detección de alrededor de 760 millones de ataques de malware distribuidos en países como: chile, Brasil, china, Bolivia y Guatemala (APWG, 2016). Además, la documentación anual de la empresa Digiware revela que Bolivia es el país donde desconoce de estos ataques ya que la tecnología va avanzando y Bolivia hace más uso de la tecnología para hacer compras con banca internet, a los atacantes les facilita hacer sus ciberataques en la población que actualmente está vulnerable.

1.3.1. Problema Principal

En los últimos años, los ataques informáticos han aumentado en frecuencia y complejidad, con el phishing como una de las tácticas de ataque más comunes y peligrosas. Según el informe "2022 Data Breach Investigations Report" de Verizon, el phishing sigue siendo una técnica de ataque muy efectiva, que se ha vuelto cada vez más sofisticada y dirigida en los últimos años 2019 - 2022. Estos ataques pueden causar importantes daños económicos y reputacionales a empresas y usuarios de américa latina, y a menudo resultan en la filtración de información confidencial y robo de identidad (Data Breach Investigations Report, 2022).

¿De qué manera un modelo mediante técnicas de machine Learning ayudaría a determinar la detección y reducción de ataques Phishing a los usuarios de Bolivia de manera precisa y certera?

1.3.2. Problemas Secundarios

- El phishing sigue siendo una amenaza creciente: El phishing es una técnica efectiva utilizada por los ciberdelincuentes que ha experimentado un aumento significativo en los últimos años. Esto representa una amenaza cada vez mayor

para las empresas y los usuarios individuales.

- Los usuarios son vulnerables y la información es escasa: Los usuarios son el eslabón más débil en la cadena de seguridad y pueden ser engañados fácilmente por correos electrónicos de phishing bien elaborados, mientras que la información sobre los diferentes tipos de ataques que realizan los cibercriminales es escasa.
- Falta de modelos y apoyo de autoridades: La falta de un modelo para la reducción de ataques de phishing y la falta de apoyo informático por parte de las autoridades para prevenir estos ataques, resulta en pérdidas significativas de información para los usuarios y empresas en nuestra región.

1.4. FORMULACION DEL PROBLEMA

¿El modelo de Machine Learning computacional, ayudará a la detección y reducción de ataques Phishing en sitios web y correos electrónicos?

1.5. OBJETIVOS

1.5.1. Objetivo General

Desarrollar un modelo utilizando técnicas de Machine Learning para la detección y reducción de ataques Phishing. Es fundamental en la actualidad debido al aumento significativo de estos ataques y la falta de medidas efectivas para combatirlos. La implementación de un modelo de este tipo permitiría proteger a los usuarios de sitios web y correos electrónicos de manera efectiva, reducir los costos económicos y evitar la pérdida de información confidencial.

1.5.2. Objetivos Específicos

- Analizar el estado actual de las técnicas de detección de ataques Phishing: Se enfoca en analizar las técnicas actuales de detección de ataques Phishing, incluyendo el uso de machine learning, para identificar sus fortalezas y debilidades.
- Diseñar nuevas características (features) y algoritmos de machine learning para mejorar la detección de ataques phishing y comparar su rendimiento con las técnicas existentes
- Implementar un sistema de detección de ataques phishing basado en técnicas de machine learning que sea efectivo para reducir los ataques en nuestra

región, y evaluar su desempeño en diferentes escenarios y situaciones.

- Evaluación de técnicas de machine learning empleadas en los últimos 5 años para la reducción de ataques Phishing

1.6. HIPOTESIS

La aplicación del modelo de detección de ataques Phishing mediante técnicas de machine Learning detecta los ataques informáticos con una eficacia del 95%.

Hipótesis Nula

La aplicación del modelo de detección de ataques Phishing mediante técnicas de machine Learning no detecta los ataques informáticos por suplantación de identidad.

1.6.1. Operacionalización de Variables

Variable Independiente: Modelo de detección y reducción de ataques Phishing mediante técnicas de machine Learning

Variable Dependiente: Ataques Informáticos.

Tabla 1.1

Operacionalizacion de variables

Variables	Tipo de variable	Dimensiones	Indicadores
Modelo de detección y reduccion de ataques Phishing mediante técnicas de Machine Learning	**Variable Independiente**	Conceptos Datos Algoritmos	Tasa de detección de correos electrónicos maliciosos.
Ataques Informaticos	**Variable Dependiente**	Ataques de phishing Ataques de ingeniería social	Datos en Correos Diagnósticos Descripciones

Nota: Operacionalizacion de variables para el modelo de detección y reducción de ataques phishing

1.7. JUSTIFICACION

La presente propuesta de investigación el cual es la Predicción de reducción de ataques Phishing mediante técnicas de machine Learning hará un aporte significativo para la sociedad donde podrán tener toda su información más segura y confiable al momento de utilizar sus datos importantes.

1.7.1. Técnica

La detección y predicción de ataques phishing es un desafío importante en la seguridad cibernética. Los métodos de detección tradicionales han demostrado ser insuficientes debido a la naturaleza cada vez más sofisticada de los ataques. Sin embargo, las técnicas de machine learning han demostrado ser efectivas en la detección de ataques phishing mediante el análisis de patrones y características específicas. Por lo tanto, proponer un enfoque basado en técnicas de machine learning para la detección y predicción de ataques phishing puede ser una solución más efectiva y precisa.

1.7.2. Económica

Los ataques phishing pueden tener graves consecuencias económicas para individuos y organizaciones, incluyendo la pérdida de datos confidenciales y financieros, el daño a la reputación y la pérdida de productividad. La detección y predicción temprana de estos ataques pueden reducir los costos económicos asociados con los mismos. Además, la implementación de un sistema basado en técnicas de machine learning puede ser una inversión rentable a largo plazo ya que el sistema puede ser actualizado y mejorado con el tiempo.

1.7.3. Social

Los ataques phishing pueden tener graves consecuencias sociales, como la violación de la privacidad y la confianza del usuario en el uso de la tecnología. La detección temprana de estos ataques puede mejorar la confianza del usuario en el uso de tecnología y mejorar la privacidad y seguridad en línea. Además, la implementación de un sistema de detección basado en técnicas de machine learning puede ser una contribución valiosa a la sociedad en términos de seguridad cibernética.

1.7.4. Científica

La detección y prevención de ataques phishing es un desafío importante en la

seguridad de la información en la actualidad. Los ataques phishing son una técnica de ingeniería social en la que los atacantes utilizan mensajes de correo electrónico fraudulentos para obtener información confidencial de los usuarios, como contraseñas, información de tarjetas de crédito y datos personales. A medida que las tecnologías de la información avanzan, los ataques phishing también se vuelven más sofisticados y difíciles de detectar, lo que hace que la investigación en este ámbito sea crucial.

Por lo tanto, la propuesta de investigación en la detección y predicción de ataques phishing con técnicas de machine learning tiene una justificación científica clara. La investigación en este ámbito puede ayudar a desarrollar nuevas técnicas de machine learning que mejoren la precisión y eficacia en la detección de ataques phishing, lo que a su vez puede contribuir a la seguridad de la información y la protección de los usuarios. Además, la investigación en este ámbito puede llevar a la identificación de nuevas características y patrones que se pueden utilizar en la mejora de otras técnicas de seguridad de la información.

1.8. METODOLOGIA

1.8.1. Método Científico

Es un método de investigación usado para la producción de conocimiento en las ciencias basado en la observación, el cual da respuestas con conocimiento científico y no como conocimiento común, dando respuestas objetivas en base a teoría, practicas, comprobación las cuales tienen una respuesta del cómo y porque, además de ser precisas con un lenguaje técnico.

Según el Oxford English Dictionary, el método científico es: "un método o procedimiento que ha caracterizado a la ciencia natural desde el siglo XVII, que consiste en la observación sistemática, medición y experimentación, y la formulación, análisis y modificación de las hipótesis."

Fases

a. Definición del problema

Cuando identificamos el problema del cual se tiene dudas sobre su

existencia o sobre su solución el cual desea ser demostrada, en este paso se debe formular la idea de forma inteligente.

b. Formulación De La Hipótesis

Ya pensado el problema, con sus posibles soluciones y también la realización de la revisión con experiencias similares de otros autores, se enuncia la hipótesis la cual debe definir las relaciones entre variables de forma correcta indicando la necesidad de verificar tales relaciones, la hipótesis debe ser comprobada para ver si son falsas o verdaderas.

c. Recopilación y Análisis De Datos

En esta fase se analizará el efecto que causa cada variable y sus dependencias al aumentar o disminuir la variable.

d. Confirmación o rechazo de hipótesis

En esta etapa se hacen pruebas sobre las relaciones de variables de la hipótesis buscando la experiencia o la comprobación.

e. Conclusiones

En esta última etapa se exponen los resultados de la experimentación indicando la hipótesis inicial es verdadera o falsa.

1.8.2. Metodología de Desarrollo

La metodología SEMMA es un proceso utilizado en el análisis de datos y la minería de datos. La sigla SEMMA significa: Sample, Explore, Modify, Model, y Assess. Esta metodología se utiliza en proyectos de minería de datos para guiar al equipo a través de las diferentes etapas del proceso y ayudar a garantizar que los resultados sean precisos y útiles (Manuel, 2020).

Fases

1. Fase de Muestreo

   En esta fase se selecciona una muestra representativa del conjunto de datos. Es importante que la muestra sea lo suficientemente grande para que el modelo pueda ser representativo del conjunto de datos completo.

2. Fase de Exploración

   En esta fase se realiza un análisis exploratorio de los datos para comprender mejor las relaciones entre las diferentes variables y determinar cuáles son relevantes para el modelo. También se realizan técnicas de visualización de datos para identificar patrones y tendencias en los datos.

3. Fase de Modificación

   En esta fase se realizan preprocesamientos y limpieza de datos, que pueden incluir técnicas como la eliminación de valores atípicos, la eliminación de características irrelevantes, la normalización de los datos, la reducción de la dimensionalidad y la imputación de valores perdidos.

4. Fase de Modelado

   En esta fase se selecciona y se entrena el modelo de machine learning que se utilizará para predecir los resultados en función de las variables de entrada. Se evalúan diferentes algoritmos de machine learning y se selecciona el que mejor se ajuste a los datos.

5. Fase de Evaluacion

   En esta fase se evalúa el rendimiento del modelo utilizando diferentes métricas, como la precisión, el recall, la exactitud y el área bajo la curva ROC. También se realiza una validación cruzada para evaluar la capacidad del modelo para generalizar a nuevos datos.

En resumen, la metodología SEMMA es un proceso estructurado para el análisis de datos y la minería de datos que involucra la selección de datos, la exploración de los datos, la modificación de los datos, la construcción de modelos y la evaluación de los resultados. Es una metodología útil para la detección de ataques phishing utilizando técnicas de machine learning ya que proporciona una guía paso a paso para construir modelos precisos y útiles.

1.8.3. Métricas de Calidad de Software

La ISO 9126 es un estándar internacional para evaluar la calidad de los atributos relacionados al conjunto de las funciones y propiedades específicas, donde estas funciones cumplen con las necesidades específicas como aquellos aspectos relacionados con la capacidad del software para mantener usabilidad enfunción del

tiempo, son criterios sobre los cuales están estructurados las métricas de calidad ISO/IEC 9126, que permite especificar y evaluar la calidad del software desde diferentes criterios. (Hidalgo, 2020)

Entonces la ISO 9126 es un estándar que tiene una estructura conceptual para evaluar la calidad de un software, la cual tiene conjuntos de características y sub-características.

Lo que especifica esta ISO es que el software debe estar descrito en 7 características básicas como ser: Funcionalidad, Confiabilidad, Usabilidad, Eficiencia, Mantenibilidad, Portabilidad y Satisfacción.

1.8.4. Estimación de Costo

El Modelo Constructivo de Costes COCOMO II, permite realizar estimaciones en función del tamaño del software y de un conjunto de factores de costos y de escala. Con el cual se estima el esfuerzo de desarrollo, el tiempo de desarrollo y la cantidad de personas que se requieren para la realización del proyecto. (Hidalgo, 2020)

COCOMO II tiene 3 modelos: Modelo de composición de aplicación, Modelo de fase de diseño previo, Modelo de fase posterior a la arquitectura.

1.8.5. Seguridad

En cuanto a la seguridad se utilizará el conjunto de estándares internacionales ISO27000 el cual cuenta con buenas prácticas para poder establecer, implementar y mantener una mejora de sistemas de gestión de la seguridad de la información.

Teniendo 2 pilares fundamentales los cuales son 27001 el cual se basa en la gestión de la seguridad de la información, para identificar los riesgos, por otra parte, la 27002 es una guía de buenas prácticas describiendo objetivos de control y gestión que debe tener la organización.

Utilizaremos las siguientes normas de estándares de seguridad

- ISO27001 el cual nos permitirá asegurar la integridad y confidencialidad de la información de los datos.
- ISO27004 El cual nos describe recomendación para la medición de los sistemas de gestión de información donde nos especifica cómo es la configuración de

métricas que medir el cómo y la forma de conseguir objetivos.

- ISO27005 el cual nos brinda información sobre recomendaciones y directrices generales para la gestión de riesgo en sistemas de gestión de seguridad de la información, también es compatible con los conceptos del ISO27001 (Global Suite, 2023).

1.8.6. Pruebas al Software

Caja Blanca. Sabemos que prueba de caja blanca se basa en detalles de procedimientos del software que están ligados al código fuente. Para que haya menos porcentaje de errores en los sistemas dando mayor confianza y calidad alsoftware.

Caja Negra. Es una técnica el cual se basa en verificar el funcionamiento del software sin tomar en cuenta la estructura como lo hace la prueba de caja blanca, entonces solo se enfoca en las entradas y salidas basándose en requerimientos y la especificación de las funciones del software.

Pruebas de Estrés. Esta evaluación se basa exponiendo el software en condiciones de uso extremas, tales condiciones serían enviar peticiones excesivas y la ejecución de hardware limitadas, tendiendo como objetivo saturar el sistema hasta un punto de quiebre.

1.9. Herramientas

1.9.1. Hardware

Para las herramientas de hardware utilizaremos un equipo con alto rendimiento para instalar todas las herramientas necesarias para desarrollar el prototipo propuesto.

Tabla 1.2 *Herramientas de Hardware*

Marca	Memoria Grafica	Memoria Ram	Sistema Operativo	Procesador
Gateway	2GB	8GB	Windows 11 Home	AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx 2.30 GHz

Nota: Herramientas en cuanto al hardware para el desarrollo

1.9.2. Software

1.9.2.1 Python

Es un lenguaje de scripting orientado a objetos e independiente de plataforma, con él se puede realizar aplicaciones Windows a servidores de red o incluso páginas web, es un lenguaje que no necesita compilarse con ello ofrece rapidez en el desarrollo, en los últimos años este lenguaje se hizo realmente popular por la cantidad de librerías que contiene, tipos de datos y funciones incorporadas sin necesidad de programarlas desde cero, otro factor importante esla sencillez y velocidad con la que se crean los programas, Además es multiplataforma como ser Unix Windows OS/2 Mac y otros, y lo mejor de este lenguaje es que es gratuito incluso para empresas (Alvarez, 2003).

1.9.2.2 Jupyter

El Proyecto Jupyter es una organización sin ánimo de lucro creada para "desarrollar software de código abierto, estándares abiertos y servicios para computación interactiva en docenas de lenguajes de programación". Creado a partir de Python en 2014 por Fernando Pérez, el proyecto Jupyter soporta entornos de ejecución en varias docenas de lenguajes de programación. El nombre del proyecto Jupyter es una referencia a los tres lenguajes de programación principales soportados por Jupyter, que son Julia, Python y R, y también un homenaje a los cuadernos de Galileo que registran el descubrimiento de los satélites de Júpiter. El proyecto Jupyter ha desarrollado y respaldado los productos de computación interactiva Jupyter Notebook, JupyterHub y JupyterLab, la versión de próxima generación de Jupyter Notebook (Programacion.org, 2021).

1.9.2.3 Anaconda

Anaconda es una distribución de código abierto y plataforma de gestión de paquetes de software para los lenguajes de programación Python y R. Anaconda incluye cientos de paquetes y bibliotecas de software preinstalados que son comunes en la ciencia de datos, el aprendizaje automático y la informática científica, lo que facilita la instalación y configuración de entornos de desarrollo para proyectos en estas áreas. Anaconda también proporciona una interfaz gráfica de usuario (GUI) y una línea

de comandos (CLI) para administrar paquetes, crear y activar entornos virtuales, y administrar los recursos de hardware. Además, Anaconda permite crear y distribuir fácilmente paquetes personalizados para su uso en entornos de trabajo compartidos.

1.9.2.4 Flask

Flask es un framework de desarrollo web en Python que se utiliza para crear aplicaciones web de manera rápida y sencilla. Flask es una herramienta minimalista y flexible que se adapta a diferentes tipos de proyectos, desde aplicaciones pequeñas hasta grandes sistemas complejos. Flask es una de las opciones más populares para el desarrollo de aplicaciones web en Python debido a su facilidad de uso y a la gran cantidad de recursos y documentación disponibles en línea. Flask proporciona una amplia variedad de extensiones y herramientas para desarrollar aplicaciones web, incluyendo soporte para bases de datos, autenticación de usuarios, manejo de formularios y plantillas HTML, entre otros. Al ser un framework ligero y flexible, Flask permite a los desarrolladores crear aplicaciones web altamente personalizadas y adaptadas a las necesidades específicas del proyecto (Muñoz, 2017).

1.9.2.5 Numpy

Es el paquete fundamental para computación científica en Python, es una biblioteca de Python que proporciona un objeto matriz multidimensional, varios objetos derivados (como matrices y matrices enmascaradas) y una variedad de rutinas para operaciones rápidas en arreglos, incluidas matemáticas, lógicas, manipulación de formas, clasificación, selección, Entradas y Salidas, Fourier discreto transformadas, algebra lineal básica, operaciones estadísticas básicas, simulación aleatoria (Numpy, 2022).

1.9.2.6 Pandas

Pandas es una biblioteca de código abierto de Python diseñada para manipular y analizar datos en estructuras de datos llamadas DataFrames. Fue desarrollada inicialmente por Wes McKinney en 2008 y actualmente es uno de los paquetes más populares para la manipulación de datos en Python.

Pandas permite a los usuarios manipular datos tabulares de una manera fácil y rápida, lo que lo convierte en una herramienta valiosa para el análisis de datos y la

ciencia de datos. Ofrece funciones para leer y escribir datos desde y hacia una amplia variedad de formatos de archivo, como CSV, Excel, SQL y bases de datos NoSQL. También ofrece funciones para la limpieza de datos, transformación, agrupación y agregación de datos, así como para la creación de gráficos y visualizaciones.

En resumen, Pandas es una biblioteca de Python que permite manipular y analizar datos en estructuras de datos tabulares llamadas DataFrames, lo que lo hace muy útil para la ciencia de datos, análisis de datos y tareas relacionadas con la manipulación de datos.

1.9.2.7 Matplotlib

Es uno de los paquetes de Python más populares utilizados para lavisualización de datos. Es una biblioteca multiplataforma para hacer gráficos 2Da partir de datos en matrices. Matplotlib está escrito en Python y hace uso de numpy, la extensión numérica de Python. Eso proporciona una API orientada a objetos que ayuda a incrustar gráficos en aplicaciones que usan Python, juegos de herramientas GUI como PYQt., Matplotlib tiene una interfaz de procedimiento llamada Pylab, que está diseñada para parecerse a matlab. Matplotlib a lo largo con numpy puede considerarse como el equivalente decódigo abierto de MATLAB (Gonzales, 2022).

1.9.2.8 Scikit-learn

Scikit-learn (también conocido como sklearn) es una biblioteca de código abierto de Python para aprendizaje automático. Fue lanzada por primera vez en 2007 y es actualmente una de las bibliotecas de aprendizaje automático más utilizadas en Python. Scikit-learn ofrece una amplia variedad de algoritmos de aprendizaje supervisado y no supervisado, así como herramientas para la evaluación de modelos, selección de características y preprocesamiento de datos.

La biblioteca es fácil de usar y tiene una gran comunidad de usuarios y desarrolladores. Scikit-learn está diseñada para integrarse con otras bibliotecas de análisis de datos en Python, como NumPy, Pandas y Matplotlib. Esto la hace muy útil para tareas de análisis de datos y aprendizaje automático, como la clasificación, regresión, agrupamiento, reducción de dimensiones, selección de características y más.

En resumen, Scikit-learn es una biblioteca de Python para aprendizaje automático que ofrece una amplia variedad de algoritmos de aprendizaje supervisado y no supervisado, así como herramientas para la evaluación de modelos, selección de características y preprocesamiento de datos. Es fácil de usar y se integra bien con otras bibliotecas de análisis de datos en Python (Universidad de Alcala, 2019).

1.9.2.9 Microsoft Windows

Es un sistema operativo, es decir un conjunto de programas que posibilita la administración de los recursos de una computadora. Este tipo de sistemas empieza a trabajar cuando se enciende el equipo para gestionar el hardware a partir de los niveles más básicos (Pérez Porto & Merino, 2021)

1.10. Limites

- Se deberá tener acceso a internet para el resultado de las pruebas que se realizará, en sitios web.
- El modelo únicamente determinara los ataques Phishing en sitios web y correos electrónicos.
- El Modelo hará una comparación de resultados únicamente durante los últimos 5 años.

1.11. Alcances

El desarrollo del modelo de detección y reducción de ataques Phishing mediante técnicas de machine Learning, que se realizará para la universidad pública de el alto (UPEA), se utilizará un entorno controlado para la simulación del ataque, por tal motivo se deben cumplir con las normas y reglamentos vigentes que permitirán respaldar la seguridad del proyecto. Con ello se menciona a continuación los procesos que se realizarán:

- Estudio sobre los ataques Phishing orientados a la suplantación de identidad más común en los últimos cinco años.
- El modelo realizara la reducción de ataques phishing enfocados a la suplantación de identidad.
- El modelo determinara una rápida y oportuna detección de ataques al usuario.

- La validación del modelo se efectuará mediante una comparación con otros algoritmos de Machine Learning, esto permitirá validar el modelo y exponer los resultados sobre la solución para la detección y reducción de ataques Phishing orientados a la suplantación de identidad mediante técnicas de machine Learning.

1.12. Aportes

El principal aporte es el de crear un modelo para detectar y reducir ataques Phishing de los criminales informáticos, proponiendo mejorar la calidad de seguridad para las personas que día a día utilizan los servicios de correos electrónicos para introducir información importante.

Identificación de la progresividad como una característica esencial de los ataques de ingeniería social para futuras problemáticas.

Análisis de la aplicabilidad de las técnicas de machine Learning estratégicamente para el estudio de problemáticas similares a los ataques de ingeniería social.

CAPÍTULO II

MARCO TEORICO

2.1. INTRODUCCION

En el presente capítulo, se presenta una explicación detallada de los conceptos esenciales que se requieren comprender para contextualizar adecuadamente el trabajo en cuestión y la creación del prototipo correspondiente. El objetivo principal es proporcionar al lector una base sólida de conocimientos teóricos y prácticos que le permita adentrarse en los temas relacionados de manera efectiva.

2.2. MODELO

Para (Hanson, 1958), un modelo es una estructura conceptual que sugiereun marco de ideas para un conjunto de descripciones que de otra manera no podrían ser sistematizadas. Cumple esta función en virtud de que une de manerainferencial, las proposiciones que afirman algo sobre los fenómenos que en él seintegran. De esta manera, su estructura es diferente de la que se supone existe en el conjunto de fenómenos de la naturaleza. El modelo concebido en esta forma, impulsa la inteligibilidad y ayuda a la comprensión de los fenómenos, ya que proporciona los canales de interconexión entre hechos que, sin la existencia de los lazos inferenciales, podrían permanecer aislados e independientes unos de otros, citado por (Badillo, 2004).

En términos de (Kuhn, 1972), los modelos son una serie de realizacionesque sirven durante una época de ciencia normal para definir problemas y métodos legítimos en un campo específico de investigación. Estos son siempre incompletos, ya que no abarcan todos los problemas que se espera han de ser resueltos, citado por (Badillo, 2004).

Los modelos pueden ser estructuras conceptuales, una serie de pasos, pero en concreto un modelo es el que ayuda a describir ideas y que en estas puedan ser sistematizadas ya sea en un campo de investigación para resolver problemas en todo caso relacionado con fenómenos naturales.

2.3. MACHINE LEARNING

El aprendizaje automático (en inglés, machine learning) es una rama de la inteligencia artificial que se ocupa del diseño y desarrollo de algoritmos y modelos computacionales que permiten a los sistemas informáticos aprender a partir de los

datos y mejorar su desempeño en tareas específicas a medida que tienen acceso a más información. En lugar de ser programados explícitamente, los modelos de aprendizaje automático se entrenan con datos y aprenden a generalizar patrones, identificar regularidades y hacer predicciones o decisiones a partir de nuevos datos no vistos previamente.

El aprendizaje automático se puede dividir en dos categorías principales: aprendizaje supervisado y aprendizaje no supervisado. En el aprendizaje supervisado, el modelo se entrena con un conjunto de datos etiquetados que incluyen tanto las entradas como las salidas esperadas. El objetivo es aprender a predecir la salida para nuevas entradas no vistas previamente. En el aprendizaje no supervisado, el modelo se entrena con un conjunto de datos no etiquetados y el objetivo es descubrir patrones y estructuras ocultas en los datos.

El aprendizaje automático se aplica en una amplia variedad de campos, como la visión por computadora, el procesamiento de lenguaje natural, la robótica, la biotecnología, la predicción de mercados y el análisis de datos. Es una de las tecnologías más importantes y prometedoras en la actualidad y se espera que tenga un impacto significativo en muchos aspectos de la vida humana.

El aprendizaje automático (machine learning) tiene una amplia variedad de aplicaciones en la industria. Algunas de las principales aplicaciones son:

- Análisis de datos y predicción: El aprendizaje automático se utiliza para analizar grandes conjuntos de datos y predecir resultados en tiempo real. Por ejemplo, el análisis de datos de ventas puede ayudar a una empresa a predecir la demanda futura y ajustar su producción en consecuencia. (Aprende Machine Learning, 2020)
- Automatización de tareas: El aprendizaje automático puede automatizar tareas que normalmente requieren intervención humana, como la clasificación de imágenes, la transcripción de voz a texto y la detección de fraudes. (Cognodata, 2023)
- Optimización de procesos: El aprendizaje automático se puede utilizar para

optimizar procesos empresariales, como la gestión de inventarios, la programación de la producción y el mantenimiento preventivo.

- Análisis de sentimientos: El aprendizaje automático se utiliza para analizar grandes cantidades de datos de redes sociales y comentarios de los clientes para medir la satisfacción del cliente y obtener información sobre los sentimientos y opiniones de los clientes.
- Asistentes virtuales: Los asistentes virtuales, como Siri y Alexa, utilizan el aprendizaje automático para mejorar su comprensión del lenguaje natural y proporcionar respuestas más precisas y relevantes a las consultas de los usuarios.
- Automatización de conducción: El aprendizaje automático se utiliza para la automatización de la conducción y la toma de decisiones de los vehículos.

2.3.1. Machine Learning para el análisis y Predicción

El aprendizaje automático (machine learning) es una herramienta poderosa para el análisis y predicción de datos. El proceso de análisis y predicción con machine learning generalmente se divide en las siguientes etapas:

1. Recopilación de datos: El primer paso es recopilar los datos relevantes para el análisis y la predicción. Esto puede implicar la recolección de datos de diversas fuentes, como bases de datos, archivos CSV, APIs, etc.
2. Preprocesamiento de datos: Los datos recopilados pueden contener ruido o información redundante que debe ser eliminada o filtrada antes de ser utilizada para el análisis y la predicción. El preprocesamiento de datos también puede implicar la normalización de datos, la imputación de valores faltantes y la selección de características relevantes para el análisis.
3. Entrenamiento de modelos: Después de preprocesar los datos, se pueden utilizar varios algoritmos de aprendizaje automático para entrenar modelos. El objetivo del entrenamiento es hacer que los modelos aprendan a generalizar patrones en los datos y hacer predicciones precisas en datos no vistos previamente.
4. Validación de modelos: Una vez que se han entrenado los modelos, se

deben validar para asegurarse de que sean precisos y útiles. Esto puede implicar la división de los datos en conjuntos de entrenamiento y prueba, la evaluación de métricas de rendimiento como la precisión, la sensibilidad y la especificidad, y la optimización de los parámetros del modelo.

5. Implementación de modelos: Finalmente, después de validar y seleccionar el modelo más adecuado, se puede implementar para hacer predicciones en nuevos datos en tiempo real.

El análisis y la predicción con machine learning se aplican en una amplia variedad de campos, como el análisis de sentimientos, la detección de fraudes, la predicción de la demanda, el análisis de mercados, la detección de enfermedades, la identificación de patrones de comportamiento y mucho más.

Figura 2.1

Predicción de Objetos

Nota. Esta imagen muestra una de las aplicaciones de machine Learning. (Aprende Machine Learning, 2020)

2.3.2. Machine Learning para automatización de tareas

El aprendizaje automático (machine learning) es una herramienta valiosa para automatizar tareas que normalmente requerirían intervención humana. Algunas de las formas en que se puede utilizar el aprendizaje automático para automatizar tareas son:

- Clasificación de imágenes: El aprendizaje automático se puede utilizar para clasificar imágenes en categorías específicas, como reconocimiento de rostros, detección de objetos, diagnóstico médico a partir de imágenes de

rayos X, entre otros.

- Transcripción de voz a texto: El aprendizaje automático se utiliza para transcribir voz en texto. Esto es especialmente útil en situaciones en las que se necesita una transcripción precisa, como la transcripción de discursos, la transcripción de reuniones, la transcripción de entrevistas, entre otros.
- Detección de fraude: El aprendizaje automático se puede utilizar para detectar patrones en los datos que sugieran actividad fraudulenta. Por ejemplo, se puede utilizar para detectar transacciones inusuales en las cuentas bancarias, lo que permite a los bancos tomar medidas preventivas.
- Traducción automática: El aprendizaje automático se utiliza para mejorar la precisión y la velocidad de la traducción automática de textos.
- Servicios de atención al cliente: El aprendizaje automático se utiliza para automatizar servicios de atención al cliente, como chatbots, que pueden interactuar con los clientes y responder preguntas comunes sin necesidad de intervención humana.
- Clasificación de correos electrónicos: El aprendizaje automático se puede utilizar para clasificar automáticamente los correos electrónicos entrantes en categorías relevantes, como correo basura, correo promocional y correo importante. (Cognodata, 2023)

En resumen, el aprendizaje automático puede automatizar tareas que normalmente requerirían intervención humana, lo que ahorra tiempo y recursos, aumenta la eficiencia y mejora la precisión en diversas áreas.

2.3.3. Machine Learning para el análisis de Sentimientos

El análisis de sentimientos es una aplicación común del aprendizaje automático (machine learning) que se utiliza para determinar la polaridad de las opiniones expresadas en un texto, ya sea positivas, negativas o neutrales. El análisis de sentimientos se puede aplicar a diversos tipos de texto, como comentarios de redes sociales, reseñas de productos, noticias, transcripciones de discursos, entre otros.

Para realizar el análisis de sentimientos con machine learning, se puede utilizar un modelo de aprendizaje supervisado que se entrena con un conjunto de datos etiquetados, donde cada texto se clasifica según su polaridad. El modelo aprende a

identificar patrones en los datos etiquetados y luego se utiliza para predecir la polaridad de nuevos textos.

Existen varias técnicas de procesamiento de lenguaje natural (NLP) que se pueden utilizar en conjunto con el aprendizaje automático para realizar el análisis de sentimientos, entre las que destacan:

1. Pre procesamiento de texto: antes de utilizar los datos para entrenar el modelo de aprendizaje automático, se deben reprocesar los datos. Esto puede incluir la eliminación de palabras irrelevantes (stopwords), la normalización de palabras (por ejemplo, reducción de palabras a su forma raíz), y la tokenización (división del texto en unidades más pequeñas, como palabras).
2. Extracción de características: una vez que los datos están reprocesados, se pueden extraer características relevantes de los textos, como el número de palabras, la frecuencia de palabras específicas, y otros indicadores de sentimiento, como la presencia de emociones o palabras positivas o negativas.
3. Selección de modelo: existen varios modelos de aprendizaje automático que se pueden utilizar para el análisis de sentimientos, incluyendo árboles de decisión, redes neuronales, SVM, y otros. (GOV.CO, 2021)

Figura 2.2

Análisis de Sentimientos

Nota. Esta imagen muestra una de las aplicaciones de machine Learning. (GOV.CO, 2021)

2.3.4. Asistentes Virtuales

Los asistentes virtuales son programas de software que utilizan inteligencia artificial para proporcionar una experiencia de usuario más interactiva y personalizada. El aprendizaje automático (machine learning) es una herramienta clave en el desarrollo de asistentes virtuales, ya que permite que los asistentes virtuales aprendan de las interacciones con los usuarios y se adapten a sus necesidades y preferencias.

Algunas de las aplicaciones del aprendizaje automático en el desarrollo de asistentes virtuales incluyen:

1. Procesamiento de lenguaje natural: El aprendizaje automático se utiliza para mejorar la capacidad del asistente virtual para procesar y entender el lenguaje natural. Esto incluye la identificación de entidades (como nombres y lugares), la comprensión de las intenciones del usuario y la respuesta a preguntas complejas.
2. Personalización: El aprendizaje automático se utiliza para personalizar la experiencia del usuario con el asistente virtual. Esto incluye la adaptación del asistente a las preferencias del usuario y la identificación de patrones en el comportamiento del usuario para mejorar la interacción.
3. Recomendaciones: El aprendizaje automático se utiliza para hacer recomendaciones basadas en el historial de interacción del usuario con el asistente virtual. Por ejemplo, un asistente virtual de compras podría utilizar el aprendizaje automático para recomendar productos que se ajusten a los intereses y preferencias del usuario.
4. Identificación de emociones: El aprendizaje automático se utiliza para identificar las emociones del usuario a través de su tono de voz y otras señales no verbales. Esto permite al asistente virtual adaptarse a las necesidades emocionales del usuario y proporcionar una experiencia más personalizada.
5. Aprendizaje en tiempo real: El aprendizaje automático se utiliza para mejorar el rendimiento del asistente virtual en tiempo real, permitiendo que el asistente aprenda y se adapte a nuevas situaciones y necesidades del usuario.

En resumen, el aprendizaje automático es una herramienta esencial para el desarrollo de asistentes virtuales, ya que permite que el asistente se adapte y mejore continuamente a través de la interacción con los usuarios. (omicrono, 2020)

Figura 2.3

Machine learning para asistentes virtuales. (Siri)

Nota. Esta imagen muestra una de las aplicaciones de machine learning. (omicrono, 2020)

2.4. TECNICAS DE MACHINE LEARNING

Hay varias técnicas de machine Learning que se pueden utilizar para la detección y reducción de ataques phishing. Algunas de ellas son:

1. **Clasificación basada en árboles de decisión:** Esta técnica se basa en la construcción de árboles de decisión a partir de un conjunto de datos de entrenamiento. Cada nodo en el árbol representa una pregunta sobre las características del correo electrónico sospechoso y las respuestas a esas preguntas determinan en qué rama del árbol se encuentra el correo electrónico.
2. **Aprendizaje profundo:** Esta técnica utiliza redes neuronales profundas para analizar los patrones y características de los correos electrónicos sospechosos. Se pueden utilizar varias capas de redes neuronales para extraer características cada vez más complejas de los datos de entrada.
3. Análisis de reputación: Esta técnica se basa en el análisis de la reputación

del remitente del correo electrónico. Se pueden utilizar bases de datos de reputación para determinar si el remitente es sospechoso o ha sido previamente reportado como un remitente de correos electrónicos fraudulentos.

4. **Análisis de anomalías:** Esta técnica se basa en la identificación de patrones y comportamientos anormales en los correos electrónicos entrantes. Se puede utilizar un modelo de detección de anomalías para identificar correos electrónicos que difieran significativamente de los patrones normales de correo electrónico entrante.
5. **Análisis de contenido:** Esta técnica se basa en la extracción de características del contenido del correo electrónico, como el lenguaje utilizado, la estructura de las frases, las palabras clave y la gramática. Se pueden utilizar algoritmos de clasificación de texto para determinar si el contenido del correo electrónico es sospechoso. (Moncada Vargas, 2021)

2.5. ATAQUES

Los ataques son acciones maliciosas que tienen como objetivo dañar, robar, obtener acceso no autorizado o interferir con sistemas informáticos, redes y/o datos. En el contexto de la seguridad informática, los ataques pueden ser llevados a cabo por ciberdelincuentes, hackers, crackers o cualquier otro individuo o grupo que busque explotar vulnerabilidades en los sistemas informáticos y comprometer su seguridad.

Los ataques pueden ser de varios tipos y formas, desde ataques de fuerza bruta y ataques de inyección de código malicioso hasta phishing y ransomware. Cada tipo de ataque tiene sus propias características y metodologías, pero todos comparten el objetivo común de causar daño o obtener beneficios ilegales.

Es importante tener en cuenta que los ataques pueden tener consecuencias graves, tanto para los individuos como para las empresas y organizaciones. Los ataques pueden resultar en la pérdida de datos, la interrupción de servicios, la pérdida financiera y la reputación dañada. Por lo tanto, es esencial tomar medidas de seguridad adecuadas y estar al tanto de las últimas amenazas y tendencias en la seguridad informática para protegerse contra los ataques.

2.6. DETECCION

La detección es el proceso de identificar la presencia de algo específico, ya sea una actividad, un evento o una amenaza. En el contexto de la seguridad informática, la detección se refiere a la identificación de actividades maliciosas o sospechosas en sistemas informáticos y redes.

En el caso de la detección de ataques phishing, el objetivo es identificar los correos electrónicos que parecen ser legítimos pero que en realidad son intentos de robar información confidencial del usuario, como nombres de usuario, contraseñas y detalles de cuentas bancarias. La detección temprana de estos correos electrónicos sospechosos es esencial para proteger la seguridad y la privacidad de los usuarios, y evitar que los atacantes obtengan acceso no autorizado a información valiosa.

La detección de ataques phishing se puede lograr mediante varias técnicas, que incluyen el análisis de contenido, el análisis de dirección de correo electrónico, el análisis de reputación y el análisis de anomalías. Cada técnica tiene sus propias fortalezas y debilidades, por lo que se puede utilizar una combinación de estas técnicas para obtener los mejores resultados en la detección de ataques phishing. (Mieres, 2020)

2.7. REDUCCION

La reducción se refiere al proceso de disminuir o disminuir algo, ya sea en tamaño, cantidad o impacto. En el contexto de la seguridad informática, la reducción se refiere a la reducción de la superficie de ataque o la minimización de los riesgos de seguridad en los sistemas informáticos y redes.

La reducción es una práctica importante en la seguridad informática, ya que puede ayudar a minimizar el riesgo de exposición a vulnerabilidades y amenazas de seguridad. Al reducir la superficie de ataque, se pueden limitar los puntos de entrada potenciales para los atacantes y hacer que sea más difícil para ellos encontrar vulnerabilidades y explotarlas.

Algunas estrategias comunes de reducción de riesgos en la seguridad informática incluyen la implementación de políticas de seguridad sólidas, el uso de software de seguridad actualizado, la eliminación de servicios innecesarios o no utilizados y la segmentación de redes para reducir la propagación de ataques.

En resumen, la reducción es un proceso importante en la seguridad informática que implica la disminución de la exposición a vulnerabilidades y amenazas de seguridad a través de una variedad de prácticas y estrategias que es de gran utilidad y se toma mucho en cuenta en la actualidad (Muñoz Campuzano, 2021).

2.8. PHISHING

El phishing es una técnica de ingeniería social utilizada por los delincuentes cibernéticos para engañar a los usuarios de Internet y obtener información confidencial, como contraseñas, información bancaria y números de tarjetas de crédito.

Los atacantes utilizan correos electrónicos, mensajes de texto, sitios web y aplicaciones de mensajería instantánea para enviar mensajes que parecen ser legítimos, pero que en realidad son fraudulentos. Estos mensajes suelen ser diseñados para parecer que provienen de una fuente confiable, como una institución financiera o un proveedor de servicios en línea, y solicitan que el usuario proporcione información confidencial.

Por ejemplo, un correo electrónico de phishing que parece ser de una institución financiera puede solicitar que el usuario haga clic en un enlace para "verificar su cuenta" y luego ingresar su nombre de usuario y contraseña. Sin embargo, en realidad, el enlace conduce a un sitio web falso que parece legítimo, pero que en realidad es controlado por los atacantes.

El phishing puede tener graves consecuencias, como el robo de información personal y financiera, la suplantación de identidad, la pérdida de dinero y la violación de la privacidad. Por lo tanto, es importante que los usuarios estén al tanto de los riesgos del phishing y tomen medidas para protegerse, como no hacer clic en enlaces sospechosos y verificar la autenticidad de los mensajes antes de proporcionar información confidencial (Microsoft.com, 2020).

2.9. SITIOS WEB

Un sitio web es un conjunto de páginas web interconectadas y accesibles a través de Internet que comparten un dominio o dirección web común. Un sitio web puede contener diferentes tipos de contenido, como texto, imágenes, videos y otros

elementos multimedia, y puede ser utilizado para diferentes propósitos, como ofrecer información, vender productos o servicios, o permitir la interacción entre usuarios.

Los sitios web pueden ser diseñados y desarrollados de diversas maneras, utilizando diferentes lenguajes de programación y herramientas de diseño web. Algunos sitios web pueden ser simples y estáticos, mientras que otros pueden ser más complejos y dinámicos, con características interactivas y en tiempo real.

Los sitios web son una parte fundamental de la presencia en línea de una empresa o individuo, ya que pueden ser utilizados para llegar a una audiencia más amplia y ofrecer información, servicios y productos a través de Internet. Además, los sitios web pueden ser optimizados para los motores de búsqueda para mejorar la visibilidad en línea y aumentar el tráfico del sitio.

Es importante tener en cuenta que la seguridad de los sitios web también es crucial para proteger la información confidencial y la privacidad de los usuarios. Los sitios web pueden ser vulnerables a diferentes tipos de ataques, como inyección de código malicioso y phishing, por lo que es importante implementar medidas de seguridad adecuadas, como el uso de software de seguridad actualizado y la aplicación de prácticas de desarrollo seguro (B., 2019).

2.10. CORREOS ELECTRONICOS

Los correos electrónicos, también conocidos como e-mails, son mensajes de texto digitales que se envían a través de Internet y que pueden ser recibidos y leídos por cualquier persona con una cuenta de correo electrónico y acceso a Internet.

Los correos electrónicos se han convertido en una forma muy popular de comunicación en línea, ya que permiten a los usuarios enviar y recibir mensajes de manera rápida y eficiente, y pueden ser utilizados tanto para la comunicación personal como profesional.

Para enviar un correo electrónico, el remitente debe tener una cuenta de correo electrónico y debe conocer la dirección de correo electrónico del destinatario. El mensaje se compone de una serie de campos, como el campo "Para" (que indica la dirección de correo electrónico del destinatario), el campo "De" (que indica la dirección de correo electrónico del remitente), el campo "Asunto" (que indica el tema del mensaje) y el campo de texto del mensaje en sí.

Los correos electrónicos pueden contener diferentes tipos de contenido, como texto, imágenes, archivos adjuntos y enlaces a sitios web. Sin embargo, es importante tener en cuenta que los correos electrónicos también pueden ser utilizados para la propagación de malware, phishing y otros tipos de ataques cibernéticos, por lo que es importante ser cuidadoso al abrir correos electrónicos de remitentes desconocidos y verificar la autenticidad de los mensajes antes de proporcionar información confidencial.

2.11. METODOLOGIA

2.11.1. Método Científico

Es un método de investigación usado para la producción de conocimiento en las ciencias basado en la observación, el cual da respuestas con conocimiento científico y no como conocimiento común, dando respuestas objetivas en base a teoría, practicas, comprobación las cuales tienen una respuesta del cómo y porque, además de ser precisas con un lenguaje técnico.

Según el Oxford English Dictionary, el método científico es: "un método o procedimiento que ha caracterizado a la ciencia natural desde el siglo XVII, que consiste en la observación sistemática, medición y experimentación, y la formulación, análisis y modificación de las hipótesis."

El método que se aplicará será el **Método Hipotético Deductivo** ya que parte de una hipótesis para posteriormente ser comprobada con la experimentación, y de esa manera poder obtener resultados, por lo cual es la quemás se asemeja al trabajo de investigación planteado.

2.11.1.1 Técnicas

- **Observación**

  Es una técnica la cual consiste en observar de manera atenta al fenómeno, para registrar la información analizada. Se utilizará esta técnica ya que se tomarán en cuenta todo tipo de información desde fotografías grabaciones documentación.

- **Encuesta.**

  Es una técnica destinada a obtener datos de varias personas cuyasopiniones

impersonales interesan al investigador. Para ello se utiliza un listado de preguntas escritas que se entregan a las personas, tal listado se denomina cuestionario, el cual es impersonal ya que no lleva los datos personales de los encuestados, además de que no Interesan esos datos.

De esta manera es que se determinó utilizar esta técnica ya que nos ayudará a recabar más información para el trabajo planteado.

2.11.1.1 Etapas del Método Científico

Definición del Problema y planteamiento del problema. Pregunta para la cual no encontramos respuesta. Es necesario que sea resoluble y debe ser formulado en términos adecuados.

Formulación de la Hipótesis. La hipótesis exige una formulación elaborada con la aparición de las variables y la relación que esperamos encontrarentre ellas. Es la verdad provisional o como se explica el problema a la luz de lo que se sabe. Las hipótesis se pueden formular como objetivos o resultados que se requieren conseguir. Para aceptar o rechazar la hipótesis se elige un determinado diseño de estudio.

Recopilación de Análisis de datos. En esta fase se analizará el efecto que causa cada variable y sus dependencias al aumentar o disminuir la variable.

Confirmación o rechazo de Hipótesis. En esta etapa se hacen pruebas sobre las relaciones de variables de la hipótesis buscando la experiencia o la comprobación.

Conclusiones. Si los datos avalan la hipótesis será confirmada. En caso contrario se concluirá que en las circunstancias contempladas la hipótesis no ha sido confirmada y/o se volverá a la segunda etapa proponiendo una nueva ycoherente solución al problema (Cástan, 2018).

2.11.2. Método de Ingeniería

2.11.2.1 Metodología de Desarrollo IWEB

Es una metodología enfocada en la creación de aplicaciones como también sistemas web de alta calidad, basándose en principios científicos de ingeniería.

Fases de la Metodología IWEB

- **Formulación. -** se realiza la identificación de objetivos, metas es deesta manera que se logra establecer el alcance de la aplicación, siendo un

aspecto importante al momento de desarrollar es la motivación y la identificación por quien va a ser utilizado.

- **Planificación. –** En esta fase se debe tomar tres puntos importantesestimar el coste general, planes de contingencia, la descripción de la aplicación en cuanto a cambios.
- **Análisis. –** En esta fase se debe establecer los requerimientos de diseño y técnicos, también se analiza el contenido del mismo su iteración configuración y funcionalidad.
- **Ingeniería. –** Es en esta etapa donde se debe lograr la integración del diseño arquitectónico de navegación y de interfaz. Por esa situación se divide en diseño de contenido, producción, arquitectónico, navegación, interfaz.
- **Generación de Paginas.** – Es aquí donde se integran los diseños de la etapa anterior a través de herramientas como lenguajes de programación y etiquetado que sirvan como base para la construcción de la aplicación web
- **Pruebas o Test.** - En esta etapa se debe comprobar y verificar lasentradas y salidas de datos con la finalidad de descubrir errores de funcionalidad comportamiento o rendimiento de la aplicación web
- **Evaluación de Cliente. -** Permite corregir errores gracias a las iteraciones realizadas con el fin de pulir la aplicación con mayor buenfuncionamiento que se vio anteriormente (Rios y otros, 2018).

2.11.3. Otras Metodologías

2.11.3.1 Metodología General para la detección de ataques Phishing

1. Recopilar datos: Es necesario recolectar un conjunto de datos de correos electrónicos que contengan tanto correos legítimos como phishing. Estos datos pueden ser obtenidos a través de diversas fuentes, como bases de datos públicas, empresas de seguridad y redes sociales.
2. Pre procesamiento de datos: Los datos recolectados deben ser procesados para eliminar información innecesaria y convertirlos a un formato que pueda ser utilizado por los algoritmos de Machine Learning. El preprocesamiento

puede incluir tareas como eliminación de stopwords, eliminación de caracteres no alfabéticos y conversión a minúsculas.

3. **Selección de características:** Es necesario identificar las características que se utilizarán para entrenar los algoritmos de Machine Learning. Algunas características comunes pueden incluir la frecuencia de palabras específicas, la presencia de enlaces sospechosos y la longitud del correo electrónico.

4. Entrenamiento del modelo: Se utilizan técnicas de aprendizaje supervisado para entrenar el modelo de Machine Learning. Para ello, se debe dividir el conjunto de datos en dos partes: una para el entrenamiento y otra para la evaluación. Se deben probar varios algoritmos de clasificación, como Naive Bayes, SVM, Árboles de decisión, Random Forest, entre otros.

5. Validación del modelo: Una vez que se ha entrenado el modelo, se debe validar para comprobar su precisión. La validación cruzada es una técnica común para medir el rendimiento del modelo.

6. Implementación del modelo: Después de validar el modelo, se debe implementar en un sistema que pueda analizar y clasificar los correos electrónicos entrantes en tiempo real.

7. Actualización continua: Es importante mantener actualizado el modelo con nuevos datos para mejorar su precisión y eficacia.

2.12. CALIDAD DE SOFTWARE

2.12.1.1 ISO 9126

La norma ISO/IEC 9126, es una norma internacional para la evaluación dela calidad del software. Permite especificar y evaluar la calidad del software desde diferentes criterios asociados con adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de calidad y auditoría de software.

El estándar ISO/IEC 9126 se desarrolló con la intención de identificar los

atributos clave del software de cómputo. El estándar identifica seis atributos clavede calidad (Hidalgo, 2020).

- ✓ **Funcionalidad,** La funcionalidad es la capacidad que tiene el software de cumplir y proveer las funciones para satisfacer las necesidades explicitas de una organización. Los atributos que toma en cuenta son, adecuación, exactitud, interoperabilidad, seguridad yconformidad.

Tabla 2.1

Condiciones especificadas

Característica	**Ponderación**
Adecuación	90%
Exactitud	90%
Conformidad	90%
Cumplimiento funional	90%

Nota: Recuperado de ingeniería de Software. (Hidalgo, 2020)

- ✓ **Confiabilidad,** Es la capacidad que tiene el software que nospermite asegurar un nivel de funcionamiento adecuado a las condiciones específicas, esta se refiere a cuatro criterios, madurez,tolerancia, y facilidad de recuperación.

La confiabilidad se calcula mediante la siguiente fórmula:

$$Confiabilidad = 1 - (n\ error/n\ LDC) \times 100$$

Donde:

N error = número de erroresLDC = líneas de código

Usabilidad, Es la capacidad del software de ser usado con facilidad de forma atractiva. La usabilidad está determinada por los usuarios finales y los usuarios indirectos del software, dirigidos a todos los ambientes, a la preparacióndel uso y el resultado obtenido. Los parámetros que toma en cuenta son la facilidad de comprensión, facilidad de aprendizaje y operatividad.

Eficiencia, La eficiencia del software se refiere a la forma del desempeño

adecuado según al número de recursos utilizados y las condiciones planteadas, asimismo se debe tomar en cuenta los aspectos como la configuración dehardware, el sistema operativo, entre otros. Toma en cuenta el comportamiento de tiempos, utilización de recurso, conformidad de recursos y conformidad de eficiencia.

Su fórmula es:

Donde:

$$Eficiencia\ del\ software = Eficiencia/LCC$$

Eficiencia del software= disponibilidad

Confiabilidad *Mantenibilidad * Capacidad

LCC= Costo de ciclo de vida

Capacidad de mantenimiento**,** Es la cualidad que tiene el software para ser modificado en ella se incluyen correcciones, mejoras del software, referidos a los cambios en el entorno, y especificaciones de requerimientos funcionales. Comprende los atributos de facilidad de análisis, facilidad de cambio, estabilidad.

Para el cálculo de mantenibilidad se usa la fórmula:

$$Mantenibilidad = (Mt - (Fc + Fa + Fd))/Mt$$

Donde:

Mt= número de módulos en la versión actual.

Fc= número de módulos en la versión actual que han cambiado.Fa= número de módulos en la versión actual añadido.

Fd= número de módulos en la versión anterior que se ha borrado.

Portabilidad, Es la capacidad que tiene el software para ser trasladado de un entorno a otro. Se refiere a la facilidad de instalación, facilidad de ajuste y facilidad de adaptación al cambio.

Su fórmula es: $Portabilidad\ 1 - (ndpm/ndim)$

Donde:

ndpm= número de días para portar el modelo.

ndim= número de días para implementar el modelo. (Hidalgo, 2020)

El autor del libro de Ingeniería de Software (Pressman, Ingenieria de software enfoque práctico, 2010), menciona que al igual que otros factores ISO 9126 no necesariamente conducen a una medición directa. Sin embargo, son unabase útil para hacer mediciones indirectas y una lista de comprobaciones excelente para evaluar la calidad del sistema

2.13. COSTOS

2.13.1.1 Cocomo II

Debido a deficiencias encontradas es donde surge COCOMO II, siendo un modelo que permite calcular el costo, el esfuerzo y el tiempo cuando se planifica una nueva actividad de desarrollo de software. Estos cambios influyen en el gasto de tanto esfuerzo en diseñar y gestionar el proceso de desarrollo software como en la creación del producto de software.

Por tanto, COCOMO II es un modelo que permite estimar el coste, esfuerzoy tiempo cuando se planifica una nueva actividad de desarrollo de software. El principal calculo en el modelo COCOMO II es el uso de la ecuación del esfuerzo para estimar el número de personas o de meses necesarios para desarrollar el proyecto. El resto del modelo se derivan de esta medida. Por un lado, COCOMO II define tres modos de desarrollo o tipos de proyectos:

Orgánico**,** proyectos de software pequeños y sencillos, menores de 50 KDLC líneas de código, en los cuales se tiene experiencia de proyectos similares y se encuentran en entornos estables.

Semi **–** acoplado**,** proyectos intermedios de complejidad y tamaño (menores de KDLC), donde la experiencia en este tipo de proyectos es variable ylas restricciones intermedias.

Empotrado**,** proyectos bastante complejos, en los que apenas se tiene experiencia y se engloban en un entorno de gran innovación técnica. Además, se trabaja con unos requisitos muy restrictivos y de gran volatilidad.

Y por otro lado existen diferentes modelos que define COCOMO:

Modelo básico**,** Calcula el esfuerzo (y el costo) del desarrollo en funcióndel tamaño del software, expresado en líneas estimadas de código (LDC).

Modelo intermedio**,** Calcula el esfuerzo del desarrollo en función deltamaño del programa y de un conjunto de “conductores de coste”.

Modelo avanzado**,** Incluye todo del modelo intermedio, además delimpacto de cada conductor de coste en las distintas fases de desarrollo.

Las Ecuaciones cocomo son:

- **Estimación del esfuerzo de desarrollo**

  La ecuación básica que utilizan los 3 modelos es:

$$E = a * (KLDC)^b * m(X)(personas\ x\ mes)$$

  Donde:

  E = es el esfuerzo estimado de desarrollo, representa las personas-mes necesarios para ejecutar el proyecto.

  a y b son constantes con valores definidos en cada submodelo.

  KLDC= es la cantidad de líneas de código, en miles. m(X)= es un multiplicador que depende de 15 atributos.

- **Estimación del tiempo de desarrollo**

Para el cálculo del tiempo de desarrollo se utiliza la siguiente fórmula:

$$T = c * (E)^d(meses)$$

Donde:

T es el tiempo de duración del desarrollo.E es el esfuerzo estimado.

c y d son constantes con valores definidos en cada submodelo.

- **Estimación del número de personal promedio**

Para estimar la cantidad de personas se utiliza la fórmula:

$$P = E/T(personas)$$

Donde:

P es el número de personas.E es el esfuerzo estimado.

T es el tiempo de duración del desarrollo.

- **Estimación de productividad**

Para estimar la productividad se utiliza la fórmula:

$$PR = LDC/E(^{LDC}/_{persona-mes})$$

Donde:

PR es la relación cantidad de líneas de código por persona al mes.KLDC es la cantidad de líneas de código, en miles.

E es el esfuerzo estimado

Estos modos de desarrollo permiten utilizar cuatro valores constantes. Estos valores constantes, codificados aquí como “a”, “b”, “c”, “d”, son propuestospor el modelo COCOMO para complementar las ecuaciones de cálculo usadas enel modelo.

Tabla 2.2

Tabla de estimación de esfuerzo

Modo	**A**	**B**	**C**	**D**
Orgánico	2.40	1.05	2.50	0.38
Semi acoplado	3.00	1.12	2.50	0.35
Empotrado	3.60	1.20	2.50	0.32

Nota. Grupo de investigación de Costos (Hidalgo, 2020)

- Estos valores son para fórmulas (E).
- Esfuerzo nominal en personas (T).
- Tiempo de desarrollo del proyecto (NP).
- Personas necesarias para realizar el proyecto (CT).

Se puede observar que a medida que aumenta la complejidad del proyecto (modo), las constantes aumentan de 2.4 a 3.6, que corresponde a un incremento del

esfuerzo personal. Hay que utilizar con mucho cuidado el modelo básico puesto que se obvian muchas características de entorno.

Cada uno de estos multiplicadores de esfuerzo, tiene una valoración que se clasifica en una escala de 6 valores desde “muy bajo”, “bajo”, “nominal”, “alto”,”muy alto”, “extraordinariamente alto”. Estos multiplicadores de esfuerzo ajustan elvalor real del esfuerzo. Los factores seleccionados se agrupan en 4 categorías:

- **RELY:** garantía de funcionamiento requerida al software. Indica lasposibles consecuencias para el usuario en el caso que existan defectos en el producto. Va desde la sola inconveniencia de corregirun fallo (muy bajo) hasta la posible pérdida de vidas humanas (extremadamente alto, software de alta criticidad)
- **DATA:** tamaño de la base de datos en relación con el tamaño del programa. El valor del modificador se define por la relación: D / K, donde D corresponde al tamaño de la base de datos en bytes y K esel tamaño del programa en cantidad de líneas de código.
- **CPLX:** representa la complejidad del producto.

Atributos del Hardware

- **TIME:** limitaciones en el porcentaje del uso de la CPU.
- **STOR:** limitaciones en el porcentaje del uso de la memoria.
- **VIRT:** volatilidad de la máquina virtual.
- **TURN:** tiempo de respuesta requerido.

Atributos del personal Involucrado en el proyecto

- **ACAP:** calificación de los analistas.
- **AEXP:** experiencia del personal en aplicaciones similares.
- **PCAP:** calificación de los programadores.
- **VEXP:** experiencia del personal en la máquina virtual.
- **LEXP:** experiencia en el lenguaje de programación a usar.

Atributos propios del proyecto

- **MODP:** uso de prácticas modernas.
- **TOOL:** uso de herramientas de desarrollo de software.

- **SCED:** limitaciones en el cumplimiento de la planificación. (Hidalgo,2020)

2.14. SEGURIDAD

Normas ISO/IEC 27000. Contiene las mejores prácticas recomendadas en seguridad de la información para desarrollar, implementar y mantener especificaciones para los SGSI

- ISO27001. Es la certificación para las organizaciones. Especifica los requisitos para la implantación SGSI. La más importante de la familia. Adopta un enfoque de gestión de riesgos y promueve la mejora continua de los procesos.
- ISO27004 El cual describe recomendación para la medición de los sistemas de gestión de información donde nos especifica cómo es la configuración de métricas que medir el cómo y la forma de conseguir objetivos.
- ISO27005 el cual brinda información sobre recomendaciones y directrices generales para la gestión de riesgo en sistemas de gestión de seguridad de la información, también es compatible con los conceptos del ISO 27001 (IBNORCA, 2020).

2.15. PRUEBAS DE SOFTWARE

La prueba de software es el proceso de verificar, evaluar que un determinado sistema prototipo o producto de software cumple con losrequerimientos establecidos con anterioridad, siendo los beneficios la prevención de errores, reducción de costos de desarrollo y para mejorar el rendimiento del mismo (Valuarte, 16).

2.15.1. Prueba de Caja Negra

Es una técnica el cual se basa en verificar el funcionamiento del software sin tomar en cuenta la estructura como lo hace la prueba de caja blanca, entoncessolo se enfoca en las entradas y salidas basándose en requerimientos y la especificación de las funciones del software.

Se realiza pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa.

No es necesario conocer la lógica del programa, únicamente lafuncionalidad que debe realizar

- Caso de Prueba. Conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para evaluar una funcionalidad del software.
- Caso de Prueba Funcional. Casos de prueba en sistemas no determinísticos (No resultados esperados fijos)

Técnicas de Particiones de Equivalencia.

Campo de entrada de un programa = ∑ de clases de datos. Los datos en una clase son equivalentes.

Objetivo. Definición del menor número de casos de prueba que descubran clases de errores. El diseño de casos de prueba según esta técnica consta de dos pasos.

- Identificar las clases de equivalencia.
- Identificar los casos de prueba.

Una prueba realizada con un valor representativo de cada clase es equivalente a una prueba realzada con cualquier otro valor de dichaclase.

Si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error y viceversa.

Conjunto de Clases Equivalentes. Conjunto de datos que definen entradas válidas y No válidas al sistema.

- Entradas válidas: generan un valor esperado.
- Entradas No válidas: Generan un valor inesperado (excepciones).

Pauta 1: Si un parámetro de entrada especifica un rango de valores. Sedefine una clase de equivalencia válida y dos inválidas.

Ejemplo: Un contador puede ir de 1 a 99

- ✓ Clases válidas -> 1< nro < 99.

- ✓ Clases inválidas-> nro< 1; nro>=99.

Pauta 2: Si un parámetro de entrada especifica un valor numérico o númerode valores. Identificar una clase válida y dos clases no válidas.Ejemplo: No. De unidades que se compran.

- Clases válidas -> Cualquier número de unidades disponibles.
- Clase inválidas -> Cualquier número no válido de unidades a comprar; No entra número.

Cómo Diseñar Pruebas con PE

- Identifique los valores de entrada del software.
- Se identifican las clases de equivalencia (Válidas - Inválidas).
- Se especifican los casos de prueba (Escenario – Resultado Esperado).

Identificar los casos de prueba

El objetivo es minimizar el número de casos de prueba, así cada caso deprueba debe considerar tantas condiciones de entrada como sea posible.

No obstante, es necesario realizar con cierto cuidado los casos de pruebade manera que no se enmascaren faltas.

Así, para crear los casos de prueba a partir de las clases de equivalencia deberán seguir los siguientes pasos:

- ➢ Asignar a cada clase de equivalencia un número único.

- ➢ Hasta que todas las clases de equivalencia hayan sido cubiertas porlos casos de prueba, se tratará de escribir un caso que cubra tantas clases válidas no incorporadas como sea posible.
- ➢ Hasta que todas las clases de equivalencia no válidas hayan sido cubiertas por casos de prueba, escribir un caso para cubrir una única clase no válida no cubierta.

Análisis de Valor Límite. La técnica se enfoca en la identificación de loscasos de prueba asociados con los valores límites del dominio de la función tantode entrada

como de salida.

Ventajas de la Técnica.

- La técnica reduce el número de casos de pruebas que deben ser creadosy ejecutados.
- Esta técnica permite elegir un subconjunto de las pruebas que son eficientey eficaces en encontrar no conformidades.
- La experiencia muestra que los casos de prueba que exploran lascondiciones límite producen mejor resultado que aquellos que no lo hacen.

Desventajas de la técnica.

- No prueba todas las entradas posibles.
- No prueba las dependencias entre las combinaciones de entrada.
- No se puede identificar qué porcentaje del sistema ha sido probado(Florian, 2019)

2.16. HERRAMIENTAS

2.16.1. Python

Es un lenguaje de programación presente en una gran multitud de aplicaciones y sistemas operativos, es versátil multiplataforma y multiparadigma que se destaca por su código legible y limpio, una de las razones de su éxito es que es de código abierto permitiendo así su utilización en cualquier escenario.

Python es ideal para trabajar con grandes volúmenes de datos ya que siendo multiplataforma favorece su extracción y procesamiento, por eso lo elidenlas empresas de Big Data.

A nivel científico tiene una gran biblioteca de recursos con especial énfasisen las matemáticas.

Las ventajas de programar en Python.

Simplificado y rápido elegante y flexible programación sana y productiva ordenado y limpio portable (Robledano, 2019).

2.16.2. Numpy

Es una librería de Python para computación científica, numpy significa Python

numérico.

Es un poderoso objeto de arreglo N-dimensional, funciones (radiodifusión) sofisticadas, herramientas para integrar código en C/C++ y Fortran, útiles capacidades de algebra lineal, transformación de Fourier y números aleatorios.

Además de sus obvios usos científicos, Numpy puede ser utilizado como un eficiente contenedor multidimensional de datos genéricos de esta manera numpy puede integrarse sin problemas y rapidez a una amplia variedad de base de datos.

El principal beneficio de numpy es que permite una generación y manejo de datos extremadamente rápido numpy tiene su propia estructura de datos incorporada llamado arreglo que es similar a la lista normal de Python (Cardellino,2021).

2.16.3. Mathplotlib

Es una librería de Python especializada en la creación de gráficos en dos dimensiones, permitiendo crear y personalizar los tipos de gráficos más comunes,entre ellos:

- Diagramas de barras
- Histograma
- Diagramas de sectores
- Diagramas de caja y bigotes
- Diagrama de violín
- Diagramas de dispersión de puntos
- Diagramas de líneas
- Diagrama de áreas
- Diagramas de contorno
- Mapas de color (aprendeconalf.es, s.f.)

2.16.4. Flask

Flask es un framework de desarrollo web en Python que se utiliza para crear aplicaciones web de manera rápida y sencilla. Flask es una herramienta minimalista y flexible que se adapta a diferentes tipos de proyectos, desde aplicaciones pequeñas hasta grandes sistemas complejos. Flask es una de las opciones más populares para el desarrollo de aplicaciones web en Python debido a su facilidad de uso y a la gran cantidad de recursos y documentación disponibles en línea. Flask proporciona una

amplia variedad de extensiones y herramientas para desarrollar aplicaciones web, incluyendo soporte para bases de datos, autenticación de usuarios, manejo de formularios y plantillas HTML, entre otros. Al ser un framework ligero y flexible, Flask permite a los desarrolladores crear aplicaciones web altamente personalizadas y adaptadas a las necesidades específicas del proyecto (Muñoz, 2017).

2.16.5. Bootstrap

Es un framework CSS desarrollado incialmente en el año 2011, permite darforma a un sitio web mediante librerías CSS que incluyen tipografías, bolones, cuadros, menús y otros elementos que pueden ser utilizados en cualquier sitio web.

2.16.6. Jupyter

El Proyecto Jupyter es una organización sin ánimo de lucro creada para "desarrollar software de código abierto, estándares abiertos y servicios para computación interactiva en docenas de lenguajes de programación". Creado a partir de IPython en 2014 por Fernando Pérez, el proyecto Jupyter soporta entornos de ejecución en varias docenas de lenguajes de programación. El nombre del proyecto Jupyter es una referencia a los tres lenguajes de programación principales soportados por Jupyter, que son Julia, Python y R, y también un homenaje a los cuadernos de Galileo que registran el descubrimiento de los satélites de Júpiter. El proyecto Jupyter ha desarrollado y respaldado los productos de computación interactiva Jupyter Notebook, JupyterHub y JupyterLab, la versión de próxima generación de Jupyter Notebook (Programacion.org, 2021).

2.16.7. Anaconda

Anaconda es una distribución de código abierto y plataforma de gestión de paquetes de software para los lenguajes de programación Python y R. Anaconda incluye cientos de paquetes y bibliotecas de software preinstalados que son comunes en la ciencia de datos, el aprendizaje automático y la informática científica, lo que facilita la instalación y configuración de entornos de desarrollo para proyectos en estas áreas. Anaconda también proporciona una interfaz gráfica de usuario (GUI) y una línea de comandos (CLI) para administrar paquetes, crear y activar entornos virtuales, y administrar los recursos de hardware. Además, Anaconda permite crear y distribuir

fácilmente paquetes personalizados para su uso en entornos de trabajo compartidos.

2.16.8. Pandas

Pandas es una biblioteca de código abierto de Python diseñada para manipular y analizar datos en estructuras de datos llamadas DataFrames. Fue desarrollada inicialmente por Wes McKinney en 2008 y actualmente es uno de los paquetes más populares para la manipulación de datos en Python.

Pandas permite a los usuarios manipular datos tabulares de una manera fácil y rápida, lo que lo convierte en una herramienta valiosa para el análisis de datos y la ciencia de datos. Ofrece funciones para leer y escribir datos desde y hacia una amplia variedad de formatos de archivo, como CSV, Excel, SQL y bases de datos NoSQL. También ofrece funciones para la limpieza de datos, transformación, agrupación y agregación de datos, así como para la creación de gráficos y visualizaciones.

En resumen, Pandas es una biblioteca de Python que permite manipular y analizar datos en estructuras de datos tabulares llamadas DataFrames, lo que lo hace muy útil para la ciencia de datos, análisis de datos y tareas relacionadas con la manipulación de datos.

2.16.9. Scikit-Learn

Scikit-learn (también conocido como sklearn) es una biblioteca de código abierto de Python para aprendizaje automático. Fue lanzada por primera vez en 2007 y es actualmente una de las bibliotecas de aprendizaje automático más utilizadas en Python. Scikit-learn ofrece una amplia variedad de algoritmos de aprendizaje supervisado y no supervisado, así como herramientas para la evaluación de modelos, selección de características y preprocesamiento de datos.

La biblioteca es fácil de usar y tiene una gran comunidad de usuarios y desarrolladores. Scikit-learn está diseñada para integrarse con otras bibliotecas de análisis de datos en Python, como NumPy, Pandas y Matplotlib. Esto la hace muy útil para tareas de análisis de datos y aprendizaje automático, como la clasificación, regresión, agrupamiento, reducción de dimensiones, selección de características y más.

En resumen, Scikit-learn es una biblioteca de Python para aprendizaje automático que ofrece una amplia variedad de algoritmos de aprendizaje supervisado

y no supervisado, así como herramientas para la evaluación de modelos, selección de características y preprocesamiento de datos. Es fácil de usar y se integra bien con otras bibliotecas de análisis de datos en Python.

Es una excelente herramienta para crear interfaces de usuario limpias y totalmente adaptables a todo tipo de dispositivos y pantallas, sea cual sea su tamaño.

Brinda soporte, además que es compatible con la mayoría de los navegadores web del mercado, y más desde la versión 3. En la actualidad, es totalmente compatible con los siguientes navegadores: Google Chrome, Safari, Mozilla Firefox, Internet Explorer, Opera.

Su principal objetivo es construir sitios web responsivos. (Gónzales, 2016)

2.16.10. Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft, Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS, otra de sus ventajas es que tiene buena integración con GIT, cuenta con soporte para depuración de código, y dispone de un sin número de extensiones, que básicamente te da la posibilidad de escribir y ejecutar códigoen cualquier lenguaje de programación

Características

- Multiplataforma. Disponible para diferentes plataformas de sistemas operativos.
- IntelliSense. Relacionada con la edición de código, te proporciona el autocompletado y resaltado de sintaxis, lo que permite ser más ágil a la hora de escribir código, y terminaciones inteligentes en base a los tipos de variables, funciones, etc.
- Depuración. Incluye una función de depuración que ayuda a detectar errores en el código. De esta manera, nos evitamos de revisar línea por línea a puro ojo humano para encontrar los errores.
- Uso de controles de versiones. Tiene compatibilidad con Git por lo que puedes revisar diferencias a lo que conocemos con git diff, organizar archivos, realizar commits desde el editor y hacer push y pull, desde cualquier

servicio de gestión de código fuente.

- Extensiones. Las extensiones que te provee Visual nos permiten personalizar y agregar funcionalidad adicional de forma modular y aislada.Por ejemplo, para programar en diferentes lenguajes y conectar con otros servicios (Flores, 2022).

CAPÍTULO III

DISEÑO METODOLOGICO

3.1. INTRODUCCION

El objetivo principal del presente estudio es desarrollar un modelo de detección y reducción de ataques phishing mediante el uso de técnicas de Machine Learning, el cual será de gran utilidad para los usuarios de diferentes plataformas web. En este capítulo se describen en detalle los conceptos, herramientas y metodologías que se plantearon de manera teórica, con el fin de sentar las bases necesarias para el desarrollo del prototipo y, por lo tanto, llevar a cabo las pruebas correspondientes. Este capítulo tiene un enfoque de investigación cuantitativa, ya que implica la recolección de gran cantidad de datos en sitios web para el proceso de predicción. Además, se aborda un nivel de investigación explicativo o causal, que se centra en comprender las relaciones de causa y efecto entre variables. Esto se conoce como operacionalización de variables, donde se identifica una variable dependiente que actúa como causa y una variable independiente que funciona como efecto o consecuencia.

Además, se presentan los objetivos específicos del estudio y se describen las metodologías empleadas para su consecución, lo que permitirá a los lectores tener una visión clara y completa del marco teórico y metodológico utilizado en el estudio.

3.2. ESTRUCTURA DEL MODELO

Figura 3.1

Estructura del Modelo General

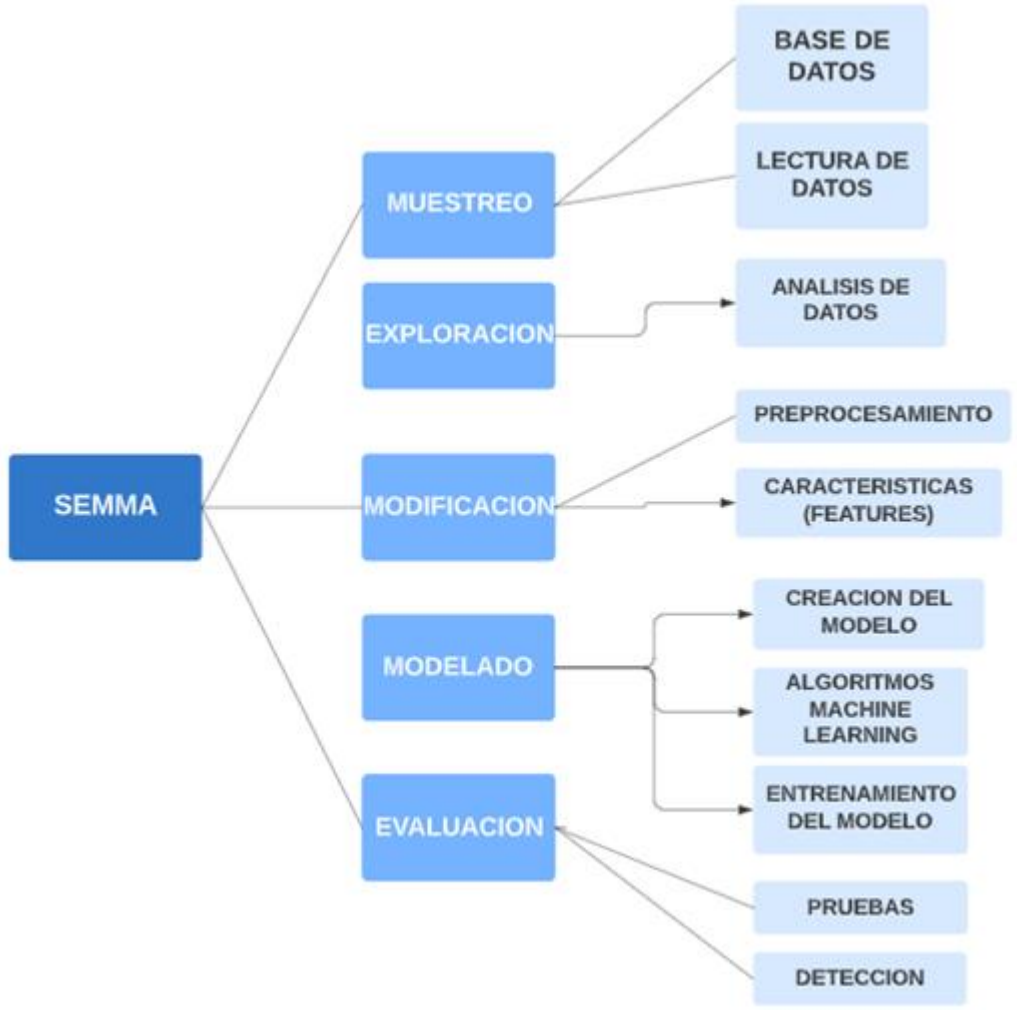
Nota. En la siguiente figura se puede observar la estructura del proceso que realiza el modelo para obtener el resultado esperado.

Metodología SEMMA para la Detección de Sitios

En este capítulo se presenta la metodología del modelo desarrollado la detección de sitios web

Figura 3.2

Modelo Desarrollado



Nota: El Grafico que se puede observar tenemos la obtención y el proceso de clasificación de datos

Donde:

- La base de datos es la parte fundamental para empezar a marcar inicio en la investigación ya que es un modelo de detección y reducción de ataques phishing

La base de datos que fue obtenida de los sitios que son usados como muestra para hacer la predicción están obtenidas de la fuente ScienceDirect ya que es una de las mayores fuentes de información para la investigación, ofrece dataset completos lo que los hace un líder de datos para todo tipo de investigación científica, técnica y medica

- Análisis de Datos permite obtener información valiosa y útil a partir de grandes conjuntos de datos. Algunos de los beneficios clave del análisis de datos incluyen:

1. Identificar patrones comunes: El análisis de datos ayudaa a identificar patrones comunes en los ataques phishing, como palabras clave, URLs sospechosas o formatos de correo electrónico inusuales.

2. Identificar comportamientos sospechosos: El análisis de datos ayuda a identificar comportamientos de usuario sospechosos, como patrones de clics inusuales o tiempos de permanencia anormalmente largos en páginas web sospechosas.

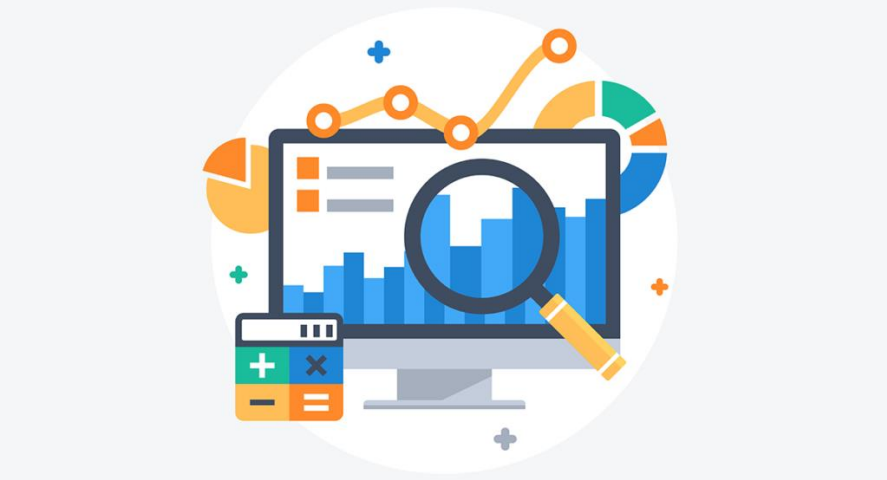
3. Desarrollar modelos predictivos: El análisis de datos ayuda a desarrollar modelos predictivos que puedan identificar automáticamente los correos electrónicos phishing y otras formas de ataques.

4. Evaluar la efectividad de las estrategias de detección: El análisis de datos ayuda a evaluar la efectividad de las estrategias de detección existentes y mejorarlas si es necesario.

5. Identificar nuevas estrategias de detección: El análisis de datos ayuda a identificar nuevas estrategias de detección de ataques phishing que no se habían considerado antes.

Figura 3.3

Análisis de datos



Nota: En el grafico se puede observar el análisis de datos

- Reprocesamiento de datos es una etapa importante en el análisis de datos y en la construcción de modelos de aprendizaje automático. Esta etapa implica la transformación de los datos crudos en una forma que pueda ser utilizada para el análisis, lo que también hace es

1. Limpieza de datos: Esto implica la eliminación de datos irrelevantes o duplicados, la corrección de errores y la eliminación de valores atípicos o datos faltantes.

2. Normalización de datos: Esto implica la escala de los datos para que estén en una escala similar, lo que permite una comparación más fácil de los datos.

3. Transformación de datos: Esto puede implicar la transformación de datos categóricos en datos numéricos, la codificación de datos para el análisis estadístico o la transformación de los datos para una mejor distribución.

4. Selección de características: Esto implica la selección de las características más importantes para el análisis, lo que puede mejorar la precisión del modelo y reducir la complejidad.

5. División de datos: Esto implica dividir los datos en conjuntos de entrenamiento y prueba para evaluar la precisión del modelo.

- El remuestreo de datos es una técnica utilizada en el análisis de datos para reutilizara un conjunto de datos nuevo y diferente de los datos originales. Esto se hace para mejorar la precisión del modelo y prevenir el sobreajuste o el subajuste del modelo a los datos originales.

Existen dos técnicas principales de remuestreo de datos:

1. Sobremuestreo: Esta técnica se utiliza cuando hay una cantidad desproporcionada de observaciones en una clase en comparación con otra. Se puede sobremuestrear la clase minoritaria duplicando algunas de sus observaciones o utilizando técnicas de síntesis para generar nuevas observaciones.

2. Submuestreo: Esta técnica se utiliza cuando hay una cantidad desproporcionada de observaciones en una clase en comparación con otra. Se puede submuestrear la clase mayoritaria eliminando algunas de sus observaciones hasta que haya una distribución más equilibrada de las observaciones.

- Algoritmos de machine learning se refiere a la cantidad de parámetros o características que se utilizan en el modelo. Los modelos de Machine Learning se construyen para aprender patrones y relaciones en los datos de entrada y la dimensión de los algoritmos de Machine Learning puede tener un gran impacto en la precisión y la eficiencia de los modelos.

- Entrenamiento se refiere a que el modelo de detección y reducción de ataques phishing será previamente entrenado para realizar las pruebas a eso adjuntando todos los procesos que se mencionaron y tener una predicción optima

- Detección se refiere a que el modelo ya podrá hacer la detección de ataques fraudulentos haciendo la clasificación de sitios falsos o

frauduentos con sitios legitimos asi poder sacar una conclusión de de que sitio o url es maligno y tomara las acciones correspondientes.

3.2.1. Recolección de datos

La recolección de datos es un proceso fundamental en cualquier investigación. En este caso, para recolectar datos de la fuente ScienceDirect, primero se definió claramente la pregunta de investigación y luego se identificó las palabras clave relevantes para la búsqueda. A continuación, se realizó una búsqueda en ScienceDirect y se refino los resultados para incluir solo los artículos más relevantes. Después de analizar cuidadosamente los artículos seleccionados, se registro la información relevante en una hoja de cálculo para su posterior análisis. La fuente ScienceDirect resultó ser una herramienta muy útil, ya que permitió acceder a una gran cantidad de información científica de alta calidad, lo que ayudo a respaldar la investigación con datos y referencias sólidas.

Características de los sitios Web por barra de direcciones:

Los sitios web por barra de direcciones son aquellos que se acceden mediante la inserción de una dirección URL en la barra de direcciones del navegador web. Esta barra es una herramienta fundamental en la navegación en línea, ya que permite al usuario acceder a cualquier sitio web simplemente ingresando su dirección URL.

Las características de los sitios web por barra de direcciones son varias, y algunas de ellas incluyen la estructura jerárquica, la facilidad de acceso, la inclusión de un motor de búsqueda interno, el menú de navegación y las herramientas interactivas.

La estructura jerárquica se refiere a la organización del sitio en diferentes niveles, lo que permite al usuario navegar fácilmente entre las diferentes secciones y páginas del sitio. La facilidad de acceso se refiere a la capacidad de los usuarios para acceder a los sitios web rápidamente y de manera eficiente.

La inclusión de un motor de búsqueda interno permite a los usuarios buscar información específica dentro del sitio, lo que ahorra tiempo y facilita la navegación. El menú de navegación proporciona una visión general de las diferentes secciones del sitio, lo que ayuda al usuario a encontrar lo que está buscando. Por último, las herramientas interactivas, como formularios de contacto o encuestas en línea,

permiten a los usuarios interactuar con el sitio y proporcionar comentarios valiosos.

En conclusión, la barra de direcciones es una herramienta fundamental en la navegación web y los sitios web por barra de direcciones tienen varias características importantes que los hacen únicos, incluyendo la estructura jerárquica, la facilidad de acceso, la inclusión de un motor de búsqueda interno, el menú de navegación y las herramientas interactivas.

3.2.2. Muestreo

3.2.2.1 Base de Datos

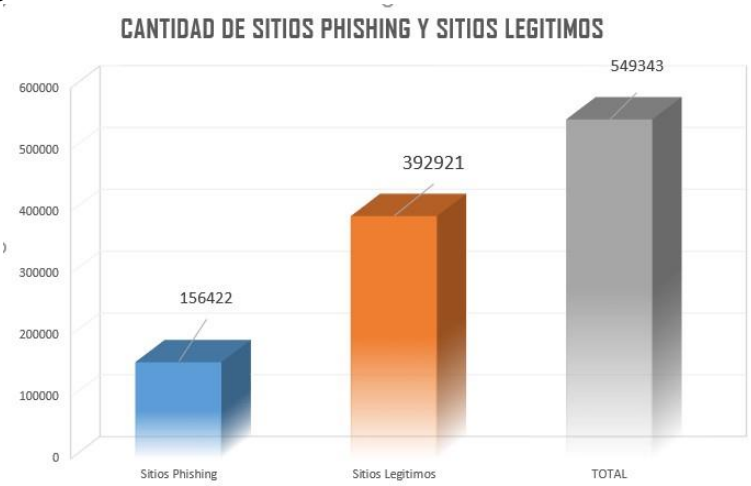
ScienceDirect es una plataforma en línea ampliamente reconocida que ofrece acceso a una extensa variedad de artículos académicos y científicos revisados por pares en múltiples disciplinas. Utilizar esta plataforma como fuente de datos para la investigación es una práctica común y válida. En el contexto de este estudio, se recurrió a ScienceDirect para recopilar información relevante sobre el phishing, un problema de seguridad cibernética importante.

Los investigadores accedieron a un artículo científico específico relacionado con el phishing en ScienceDirect para obtener un entendimiento más profundo del tema. Este artículo proporcionó una base sólida para la investigación, ya que estaba respaldado por expertos en el campo y había pasado por un riguroso proceso de revisión por pares para garantizar su calidad y precisión.

Además del artículo científico, se utilizó un conjunto de datos disponible en ScienceDirect que contenía información sobre sitios web legítimos y sitios de phishing. La base de datos incluía un total de 392,921 sitios legítimos y 156,422 sitios de phishing, lo que resultó en una muestra de datos completa de 549,346 sitios. Este conjunto de datos fue fundamental para desarrollar el modelo de machine learning requerido para la investigación.

Figura 3.4

Muestra de Datos



Nota: la imagen muestra la base de datos para el Modelo

3.2.2.2 Lectura de Datos

Una vez concluida la recolección de datos en la prestigiosa fuente de investigación ScienceDirect, el siguiente paso consiste en llevar a cabo una lectura de los datos obtenidos. Para tal fin, se procedió a cargar los datos obtenidos en el formato adecuado, según se muestra en la imagen, lo que permitirá una adecuada manipulación y exploración de los datos para su posterior análisis. Este proceso de carga de datos es fundamental para asegurar la fiabilidad y precisión de los resultados de la investigación, ya que garantiza que se estén trabajando con los datos correctos y que los mismos estén organizados de manera adecuada para su análisis posterior. De esta manera, se podrá llevar a cabo un análisis exhaustivo y riguroso de los datos obtenidos, permitiendo una comprensión profunda de los mismos y la identificación de patrones y tendencias relevantes para la investigación.

Figura 3.5

Cargado de datos del dataset

```
# cargando el conjunto de datos
phish_data = pd.read_csv("C:/Users/GABRIEL/Pictures/Tesis Gabriel/PROTOTIPO DE TESIS/tesis phising/tesis phising/phishing_site_urls.csv")
```

Nota: la presente imagen se observa la carga de datos

Visualización de los datos del dataset:

Una vez que hemos cargado exitosamente los datos, es pertinente proceder a visualizar y comprender la naturaleza de los datos contenidos en el dataset. Para ello, haremos uso de la función phish_data.head(), la cual nos permitirá obtener una vista preliminar de los datos almacenados. Dicha función devolverá los datos iniciales del dataset, lo que nos permitirá conocer la estructura y la naturaleza de los datos que poseemos. A partir de esta primera inspección, podremos determinar las características principales del dataset, incluyendo su tamaño, su forma y los tipos de datos almacenados, lo que nos permitirá diseñar estrategias adecuadas para el análisis y la interpretación de los datos.

Figura 3.6

Visualización de los datos del dataset

```
phish_data.head() #mostrando los datos de la base de datos
```

	URL	Label
0	nobell.it/70ffb52d079109dca5664cce6f317373782/...	bad
1	www.dghjdgf.com/paypal.co.uk/cycgi-bin/webscrc...	bad
2	serviciosbys.com/paypal.cgi.bin.get-into.herf....	bad
3	mail.printakid.com/www.online.americanexpress....	bad
4	thewhiskeydregs.com/wp-content/themes/widescre...	bad

Nota: la presente imagen se observa la visualización del dataset

Visualización de las últimas filas del dataset

Después de haber obtenido una vista preliminar de los datos almacenados en el dataset, que incluye información relevante como la URL y la etiqueta que indica si dicha URL es considerada bad "mala" o good "buena", resulta necesario llevar a cabo una revisión adicional para asegurarnos de que el conjunto de datos esté completo y sea adecuado para su análisis posterior. Con este fin, se procedió a utilizar la función phish_data.tail(), la cual permite obtener una vista de las últimas filas del dataset. Esta acción permite verificar la integridad del conjunto de datos, asegurándonos de que no

existan registros incompletos o faltantes, lo que podría afectar la calidad y confiabilidad de los resultados obtenidos. Por tanto, este paso resulta crucial para garantizar la validez y la precisión de la investigación, y así poder llevar a cabo un análisis completo y exhaustivo de los datos obtenidos.

Figura 3.7

Visualización de las últimas filas del dataset

```
phish_data.tail()
```

	URL	Label
549341	23.227.196.215/	bad
549342	apple-checker.org/	bad
549343	apple-iclods.org/	bad
549344	apple-uptoday.org/	bad
549345	apple-search.info	bad

Nota: la presente imagen se observa la visualización de las últimas filas del dataset
Descripción general del dataset

Luego de haber inspeccionado exhaustivamente las últimas filas del dataset, resulta fundamental llevar a cabo una descripción general del mismo, con el objetivo de obtener una comprensión más profunda de las características de los datos contenidos en el conjunto de datos. Para llevar a cabo esta tarea, se hará uso del dataframe de la librería pandas, la cual nos permitirá analizar los datos contenidos en el dataset de manera detallada y rigurosa. A través de esta descripción general, podremos obtener información relevante sobre la estructura del dataset, como la cantidad de columnas y filas, así como también estadísticas descriptivas, tales como media, desviación estándar y percentiles, que nos permitirán tener una idea más clara de la distribución y variabilidad de los datos contenidos en el dataset. De esta manera, podremos diseñar estrategias de análisis adecuadas y precisas para la investigación, y así poder obtener resultados confiables y precisos que puedan ser utilizados para el desarrollo de conclusiones y recomendaciones pertinentes.

Figura 3.8

Visualización de las últimas filas del dataset

```
phish_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 549346 entries, 0 to 549345
Data columns (total 2 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   URL     549346 non-null  object
 1   Label   549346 non-null  object
dtypes: object(2)
memory usage: 8.4+ MB
```

Nota: la presente imagen se observa la Descripción general del dataset

3.2.3. Exploración

3.2.3.1 Análisis de Datos

Visualización de la cantidad de valores nulos en cada columna del conjunto de datos:

Una vez que hemos llevado a cabo una descripción general del conjunto de datos, resulta importante verificar la integridad de los mismos, con el objetivo de asegurarnos de que no existan valores nulos o faltantes que puedan afectar la calidad y la confiabilidad de los resultados obtenidos. Para llevar a cabo esta tarea, se hará uso de la función phish_data.isnull().sum(), la cual nos permitirá obtener información detallada sobre la cantidad de valores nulos en cada columna del conjunto de datos. De esta manera, podremos detectar y corregir de manera oportuna los valores faltantes, lo que nos permitirá asegurar la precisión y la fiabilidad de los resultados obtenidos. Este es un paso fundamental en el proceso de análisis de datos, ya que nos permite obtener una comprensión más profunda de la calidad de los datos contenidos en el conjunto de datos, y así poder llevar a cabo un análisis riguroso y preciso de los mismos.

Figura 3.9

Descripción General del Datased

```
phish_data.isnull().sum()
```

```
URL      0
Label    0
dtype: int64
```

Nota: la presente imagen se observa la Descripción general del dataset

Conteos de cada clase o etiqueta en el conjunto de datos:

Una vez realizada la verificación de la presencia de valores nulos en cada columna del conjunto de datos, se procede a obtener los conteos de cada clase o etiqueta mediante la utilización de la función "label_counts= pd.DataFrame(phish_data.Label.value_counts())". Este paso es fundamental para verificar si el conjunto de datos está equilibrado o desequilibrado, ya que un conjunto de datos desequilibrado puede generar un sesgo en el modelo de Machine Learning, afectando su capacidad para realizar predicciones precisas y confiables. Por lo tanto, es importante evaluar si la cantidad de muestras en cada clase es lo suficientemente grande como para permitir un entrenamiento adecuado del modelo y evitar problemas de sobreajuste o subajuste. De esta manera, se garantiza que el modelo pueda generalizar de manera efectiva a nuevas muestras y ser utilizado en diferentes contextos y escenarios.

Figura 3.10

Conteo de cada clase o etiqueta en el conjunto de datos

```
label_counts = pd.DataFrame(phish_data.Label.value_counts())
```

Nota: la presente imagen se observa el conteo de cada clase o etiqueta en el conjunto de datos.

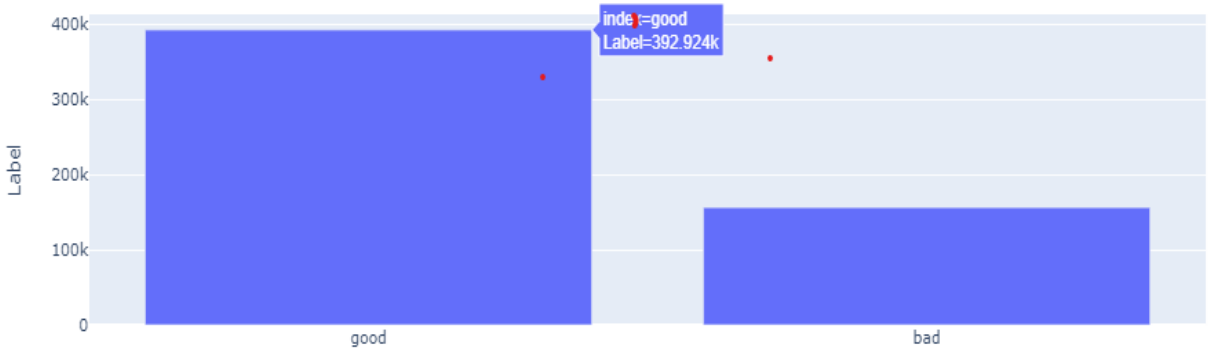
Creación de un gráfico de barras:

Con el fin de lograr una observación detallada y un análisis exhaustivo de cada muestra, se procede a la creación de un gráfico de barras que permita visualizar con

claridad la cantidad de datos correspondiente a los sitios de phishing y a los sitios legítimos. De esta forma, se busca proporcionar una representación gráfica que permita identificar de manera precisa y concisa las diferencias en la cantidad de datos entre ambas categorías de sitios web. Tal representación resulta de gran utilidad al momento de llevar a cabo un análisis minucioso de cada muestra, lo que posibilita una toma de decisiones informada y certera en torno a la evaluación de la confiabilidad y seguridad de los sitios web estudiados.

Figura 3.11

Creación de un gráfico de barras de datos entrenamiento y prueba (features)



Nota: la presente imagen se observa la creación de un gráfico de barras.

3.2.4. Modificación

3.2.1.1 Pre procesamiento de Datos

Creación de un tokenizador que se utiliza en el preprocesamiento de texto:

Después de haber realizado la verificación de la presencia de valores nulos en el conjunto de datos, se procede a la creación de un tokenizador que será utilizado en la etapa de preprocesamiento de texto. El objetivo principal del tokenizador es eliminar los caracteres no alfabéticos y generar tokens o palabras individuales a partir de las cadenas de texto. De esta manera, se busca convertir la información textual en una representación numérica que pueda ser utilizada en los algoritmos de Machine Learning. Para ello, se emplean técnicas de procesamiento del lenguaje natural que permiten la normalización de las palabras y la eliminación de caracteres especiales, acentos y signos de puntuación. De esta manera, se logra una mayor consistencia en

los datos y se mejora la calidad de la información que se utilizará para el entrenamiento del modelo.

Figura 3.12

Creación de un tokenizador que se utiliza en el preprocesamiento de texto

```
tokenizer = RegexpTokenizer(r'[A-Za-z]+')
```

Nota: la presente imagen se observa la *creacion de un tokenizador que se utiliza en el preprocesamiento de texto.*

Devolución de la Url de la primera muestra en el conjunto de datos:

Después de haber creado el tokenizador para el preprocesamiento de texto, se procede a utilizar la función phish_data.URL[0], que permite obtener la URL correspondiente a la primera muestra del conjunto de datos. Esta función es de gran utilidad, ya que permite conocer de manera más detallada la estructura de la información que se está manejando, lo cual es fundamental para la realización del proceso de preprocesamiento de los datos. Además, este análisis preliminar permite identificar posibles patrones y características relevantes que serán de utilidad para la posterior elaboración del modelo de detección y reducción de ataques phishing. Cabe destacar que la obtención de esta información de manera detallada es esencial para garantizar la efectividad y calidad del modelo, así como para su correcta implementación y aplicación en diferentes contextos y situaciones.

Figura 3.13

Devolución de la Url de la primera muestra en el conjunto de datos.

```
#devuelve la URL de la primera muestra en el conjunto de datos,
phish_data.URL[0]
```

```
'nobell.it/70ffb52d079109dca5664cce6f317373782/login.SkyPe.com/en/cgi-bin/verification/login/70ffb52d079109dca5664cce6f317373/index.php?cmd=_profile-ach&outdated_page_tmpl=p/gen/failed-to-load&nav=0.5.1&login_access=1322408526'
```

Nota: la presente imagen se observa la devolución de la Url de la primera muestra en el conjunto de datos.

Devolución de una lista de tokens (palabras individuales) que representan la URL de la primera muestra en el conjunto de datos

Luego de haber obtenido la URL correspondiente a la primera muestra del conjunto de datos, procedemos a generar una lista de tokens que representa las palabras individuales que componen dicha URL, utilizando la función phish_data.URL[0]. Dicha lista será útil para el posterior análisis y procesamiento de texto, en el cual se buscará identificar patrones y características que permitan detectar posibles ataques de phishing. Es importante destacar que el proceso de tokenización es una técnica fundamental en el preprocesamiento de texto, ya que permite descomponer el texto en unidades más pequeñas para su posterior análisis y procesamiento.

Figura 3.14

Devolución de una lista de tokens

```
tokenizer.tokenize(phish_data.URL[0])
```

Nota: la presente imagen se observa la devolución de una lista de tokens

Figura 3.2.29

Devolución de una lista de tokens

```
' ,
'login',
'SkyPe',
'com',
'en',
'cgi',
'bin',
'verification',
'login',
'ffb',
'd',
'dca',
'cce',
'f',
'index',
'php',
'cmd',
'profile',
'ach',
'outdated',
'page',
'tmpl',
'p',
'gen',
'failed',
'to',
'load',
'nav',
'login',
'access']
```

Nota: la presente imagen se observa la devolución de una lista de tokens

Medición del tiempo de inicio de una tarea o proceso:

En esta etapa procedemos a medir el tiempo de una tarea o proceso al guardar el tiempo actual en la variable “t0” con la función t0= time.perf_counter() también se ha creado una nueva columna llamada "text_tokenized" en el conjunto de datos, la cual contiene una lista de tokens generados a partir de las URLs almacenadas en la columna "URL" del conjunto de datos. Esta operación se realizó mediante el uso de la función de mapeo de pandas "map", la cual aplica una función especificada (en este caso, la función lambda que hace referencia al tokenizador previamente creado) a cada elemento de la columna "URL". El resultado de esta operación es una lista de tokens para cada URL, que se almacena en la nueva columna "text_tokenized".

Figura 3.15

Medicion del tiempo de inicio de una tarea o proceso

```
#En este caso, el código se utiliza para medir el tiempo de inicio de una tarea o proceso.
#Al guardar el tiempo actual en la variable "t0"
print('obteniendo palabras tokenizadas
t0= time.perf_counter()
phish_data['text_tokenized'] = phish_data.URL.map(lambda t: tokenizer.tokenize(t)) #
t1 = time.perf_counter() - t0 #tiempo
print('Tiempo Empleado',t1 ,'sec')
```

```
Getting words tokenized ...
Time taken 4.950578500051051 sec
```

Nota: la presente imagen se observa la medicion del tiempo de inicio de una tarea o proceso.

Devolución de una muestra aleatoria de 5 filas (muestras) del conjunto de datos:

Una vez medido el tiempo de inicio del proceso procedemos a hacer una devolución de una muestra aleatoria de 5 filas. La función phish_data.sample(5) devuelve una muestra aleatoria de 5 filas del conjunto de datos phish_data. Esto es útil para obtener una idea de cómo se distribuyen los datos y qué tipos de datos hay en el conjunto de datos. Cada vez que se llama a esta función, se obtendrán diferentes filas aleatorias del conjunto de datos, lo que permite una exploración más completa del conjunto de datos.

Figura 3.16

Devolución de una muestra aleatoria de 5 filas (muestras) del conjunto de datos

```
In [13]: #devuelve una muestra aleatoria de 5 filas (muestras) del conjunto de datos
         phish_data.sample(5)
```

Out[13]:

	URL	Label	text_tokenized
61529	www.dev-archive.net/articles/pointers/bugs.html	good	[www, dev, archive, net, articles, pointers, b...
266848	aboutmexico.net/mexico/films.asp	good	[aboutmexico, net, mexico, films, asp]
129858	thechrisomatic.com/blog/log/googledoc.html	bad	[thechrisomatic, com, blog, log, googledoc, html]
288612	baseball-reference.com/players/s/stewada01.shtml	good	[baseball, reference, com, players, s, stewada...
300988	cdbaby.com/cd/waxeom	good	[cdbaby, com, cd, waxeom]

Nota: la presente imagen se observa la devolución de una muestra aleatoria de 5 filas (muestras) del conjunto de datos.

Creación de un objeto de stemming en inglés que se puede utilizar para reducir las palabras a sus formas de raíz en el procesamiento de texto**:**

Una vez hecho la devolución de una muestra aleatoria de las 5 filas (muestras) procedemos a usar la función stemmer = SnowballStemmer("english") crea un objeto de la clase SnowballStemmer de la biblioteca nltk en Python, que se utiliza para realizar la técnica de stemming (o reducción de palabras a su raíz) en el idioma inglés. La técnica de stemming es comúnmente utilizada en el procesamiento de lenguaje natural (NLP) y tiene como objetivo reducir las palabras a su raíz o a una forma más simple para poder realizar operaciones de análisis de texto de manera más efectiva y eficiente. En el caso de SnowballStemmer("english"), se realiza la reducción de palabras en inglés utilizando el algoritmo Snowball que fue desarrollado por Martin Porter en 1980.

Figura 3.17

Creación de un objeto de stemming en inglés

```
stemmer = SnowballStemmer("english")
```

Nota: la presente imagen se observa la creación de un objeto de stemming en inglés.

Medición del tiempo de inicio de una tarea o proceso:

En esta etapa procedemos a medir el tiempo de una tarea o proceso al guardar el tiempo actual en la variable “t0” con la función t0= time.perf_counter() también se ha creado una nueva columna llamada "text_tokenized" en el conjunto de datos, la cual contiene una lista de tokens generados a partir de las URLs almacenadas en la columna "URL" del conjunto de datos. Esta operación se realizó mediante el uso de la función de mapeo de pandas "map", la cual aplica una función especificada (en este caso, la función lambda que hace referencia al tokenizador previamente creado) a cada elemento de la columna "URL". El resultado de esta operación es una lista de tokens para cada URL, que se almacena en la nueva columna "text_tokenized".

Figura 3.18

Medición del tiempo de inicio de una tarea o proceso

```
#En este caso, el código se utiliza para medir el tiempo de inicio de una tarea o proceso.
#Al guardar el tiempo actual en la variable "t0" utli para el rendimiento de aprendizaje automatico
print('Obteniendo palabras deribadas...')
t0= time.perf_counter()
phish_data['text_stemmed'] = phish_data['text_tokenized'].map(lambda l: [stemmer.stem(word) for word in l])
t1= time.perf_counter() - t0
print('tiempo empleado',t1 ,'sec')
```

```
Getting words stemmed ...
Time taken 80.847471499932 sec
```

Nota: la presente imagen se observa la medicion del tiempo de inicio de una tarea o proceso.

Creación de una nueva columna que contiene el texto preprocesado con el stemming realizado para cada URL en el conjunto de datos:

Después de haber hecho la medición del tiempo de inicio de una tarea o proceso procederemos a crear una nueva columna llamada “text_sent” que contiene el texto preprocesado con el stemming realizado para cada URL en el conjunto de datos.

Figura 3.19

Creación de una nueva columna que contiene el texto preprocesado

```
print('obteniendo palabras de union ...')
t0= time.perf_counter()
phish_data['text_sent'] = phish_data['text_stemmed'].map(lambda l: ' '.join(l))
t1= time.perf_counter() - t0
print('Tiempo empleado',t1 ,'sec')
```

```
Getting joiningwords ...
Time taken 0.5657930000452325 sec
```

Nota: la presente imagen se observa la creación de una nueva columna que contiene

el texto preprocesado.

Creación de un nuevo conjunto de datos:

Una vez creado una nueva columna que contiene el texto preprocesado con el stemming procedemos a la creación de un nuevo conjunto de datos en esta ocacion crearemos 2 conjunto de datos.

La primera función bad_sites = phish_data[phish_data.Label == 'bad'] está creando un nuevo conjunto de datos llamado bad_sites que contiene solo las muestras etiquetadas como "mala" (en inglés "bad").

La segunda funcion good_sites = phish_data[phish_data.Label == 'good'] está creando otro conjunto de datos llamado good_sites que contiene solo las muestras etiquetadas como "buena" (en inglés "good").

Esto es útil porque permitirá entrenar un modelo de machine learning para distinguir entre sitios web buenos y malos, utilizando cada uno de estos subconjuntos para el entrenamiento y validación del modelo.

Figura 3.20

Creación de una nueva columna que contiene el texto preprocesado

```
bad_sites = phish_data[phish_data.Label == 'bad']
good_sites = phish_data[phish_data.Label == 'good']
```

Nota: la presente imagen se observa la creación de una nueva columna que contiene el texto preprocesado.

Clasificando los sitios Falsos y Legítimos:

Después de crear la columna que clasifica los sitios web entre legítimos y falsos, procedemos a exhibir ambas categorías. Esta separación en dos tablas distintas es muy beneficiosa, ya que permite una clasificación más precisa y detallada de cada sitio web. Concretamente, utilizando la función "bad_sites.head()" podemos visualizar los sitios web que han sido clasificados como falsos, mientras que con la función "good_sites.head()" podemos ver los sitios web que han sido clasificados como legítimos.

Figura 3.21

Clasificación de sitios falsos y sitios legítimos

```
bad_sites.head()
good_sites.head()
```

Nota: la presente imagen se observa la clasificación de sitios falsos y sitios legítimos

Figura 3.22

Clasificación de sitios falsos y sitios legítimos

	URL	Label	text_tokenized	text_stemmed	texto_enviado
0	nobell.it/70ffb52d079109dca5664cce6f317373782/...	bad	[nobell, it, ffb, d, dca, cce, f, login, SkyPe...	[nobel, it, ffb, d, dca, cce, f, login, skype,...	nobel it ffb d dca cce f login skype com en cg...
1	www.dghjdgf.com/paypal.co.uk/cycgi-bin/webscrc...	bad	[www, dghjdgf, com, paypal, co, uk, cycgi, bin...	[www, dghjdgf, com, paypal, co, uk, cycgi, bin...	www dghjdgf com paypal co uk cycgi bin webscrc...
2	serviciosbys.com/paypal.cgi.bin.get-into.herf....	bad	[serviciosbys, com, paypal, cgi, bin, get, int...	[serviciosbi, com, paypal, cgi, bin, get, into...	serviciosbi com paypal cgi bin get into herf s...
3	mail.printakid.com/www.online.americanexpress....	bad	[mail, printakid, com, www, online, americanex...	[mail, printakid, com, www, onlin, americanexp...	mail printakid com www onlin americanexpress c...
4	thewhiskeydregs.com/wp-content/themes/widescre...	bad	[thewhiskeydregs, com, wp, content, themes, wi...	[thewhiskeydreg, com, wp, content, theme, wide...	thewhiskeydreg com wp content theme widescreen...

Nota: la presente imagen se observa a los sitios falsos

Figura 3.23

Creación de una nueva columna que contiene el texto preprocesado.

	URL	Label	text_tokenized	text_stemmed	texto_enviado
18231	esxcc.com/js/index.htm?us.battle.net/noghn/en/...	good	[esxcc, com, js, index, htm, us, battle, net, ...	[esxcc, com, js, index, htm, us, battl, net, n...	esxcc com js index htm us battl net noghn en r...
18232	www□eira¯&nvinip¿ncH¯wVö%ÆåyDaHðû/ÏyÉùu□Ë\nÓ□6...	good	[www, eira, nvinip, ncH, wV, yDaH, yÉ, u, rT, ...	[www, eira, nvinip, nch, wv, ydah, ye, u, rt, ...	www eira nvinip nch wv ydah ye u rt u g m i xz...
18233	'www.institutocgr.coo/web/media/syqvem/dk-□óij...	good	[www, institutocgr, coo, web, media, syqvem, d...	[www, institutocgr, coo, web, media, syqvem, d...	www institutocgr coo web media syqvem dk ij r ...
18234	□□Yié□♠koäÕ»Î§DéÍ□I½ñ¡áâqtò,/à; Í	good	[Y, ko, D, I, qt]	[y, ko, d, I, qt]	y ko d I qt
18236	ruta89fm.com/images/AS@Vies/1i75cf7b16vc<F□d16...	good	[ruta, fm, com, images, AS, Vies, i, cf, b, vc...	[ruta, fm, com, imag, as, vie, i, cf, b, vc, f...	ruta fm com imag as vie i cf b vc f d b g sd v...

Nota: la presente imagen se observa a los sitios legítimos.

Generando Nube de palabras y nube de etiquetas:

La creación de la nube de palabras y etiquetas se realiza con el uso de un generador de nubes de palabras y etiquetas llamado WORLDCLOUDS. Este

generador permite generar visualizaciones que son ampliamente utilizadas para explorar el contenido de los datos de texto y descubrir patrones o tendencias interesantes en los mismos. Las nubes de palabras y etiquetas son una herramienta de visualización que presenta las palabras más frecuentes en un conjunto de datos en forma de una imagen gráfica en la que el tamaño de cada palabra se determina por su frecuencia de aparición en el texto. Este tipo de visualización puede ser muy útil para resaltar las palabras clave en un conjunto de datos de texto y puede ayudar a identificar temas o patrones importantes.

Figura 3.24

Generando Nube de palabras y nube de etiquetas.

```
def plot_wordcloud(text, mask=None, max_words=400, max_font_size=120, figure_size=(24.0,16.0),
                   title = None, title_size=40, image_color=False):
    stopwords = set(STOPWORDS)
    more_stopwords = {'com','http'}
    stopwords = stopwords.union(more_stopwords)

    wordcloud = WordCloud(background_color='white',
                    stopwords = stopwords,
                    max_words = max_words,
                    max_font_size = max_font_size,
                    random_state = 42,
                    mask = mask)
    #generando la nube de palabras de tipo texto
    wordcloud.generate(text)

    plt.figure(figsize=figure_size)
    if image_color:
        image_colors = ImageColorGenerator(mask);
        plt.imshow(wordcloud.recolor(color_func=image_colors), interpolation="bilinear");
        plt.title(title, fontdict={'size': title_size,
                                   'verticalalignment': 'bottom'})
    else:
        plt.imshow(wordcloud);
        plt.title(title, fontdict={'size': title_size, 'color': 'green',
                                   'verticalalignment': 'bottom'})
    plt.axis('off');
    plt.tight_layout()
d = 'input/mask/' #Creando el directorio para subir imagenes para la nube de palabras local
```

Nota: la presente imagen se la generación de nube de palabras y nube de etiquetas.

Seleccionando una columna específica de un DataFrame y reindexación en el objeto resultante.

Después de haber generado las nubes de palabras y etiquetas, se procedió a establecer un directorio local para cargar las imágenes generadas por el programa. Posteriormente, se seleccionó una columna específica del dataframe utilizando la

función good_sites.text_sent, y se realizó una operación de reindexación en el objeto resultante mediante la función data.reset_index(drop=True, inplace=True). Este proceso es importante para garantizar la precisión y la calidad del análisis de los datos, ya que permite una organización y estructuración adecuada de los mismos.

Figura 3.25

Seleccionando una columna específica de un DataFrame y reindexación

```
#selecciona una columna específica de un DataFrame,
data = good_sites.texto_enviado
#realiza una operación de reindexación en el objeto resultante.
data.reset_index(drop=True, inplace=True)
```

Nota: la presente imagen se observa el Seleccionando una columna específica de un DataFrame y re indexación

Creación de características (Features):

Conversión de las imágenes de texto en una matriz de recuento de términos o palabras

Después de haber generado la imagen para la nube de palabras, se procede a realizar un proceso de transformación para convertir las palabras en una matriz de recuento de términos utilizando la biblioteca de aprendizaje automático "scikit-learn". Para llevar a cabo este proceso, se utiliza la función "CountVectorizer()", la cual permite transformar un conjunto de textos en una matriz de recuento de palabras que se utiliza en el análisis de texto. De esta forma, se puede analizar la frecuencia de las palabras y determinar cuáles son las más comunes o relevantes en el conjunto de datos.

Figura 3.26

Conversión de las imágenes de texto en una matriz de recuento de términos o

```
cv = CountVectorizer()
```

Nota: la presente imagen se observa la conversión de las imágenes de texto en una matriz de recuento de términos o palabras.

3.2.4.1 Características (Features)

Una vez hecho la conversión de imágenes de texto en una matriz de recuento de términos o palabras procedemos con la función cv.fit_transform(phish_data.text_sent) utiliza el objeto cv que fue creado previamente con la función CountVectorizer() para convertir los datos de texto de las columnas seleccionadas en una matriz de términos de frecuencia de documentos (TF-IDF). La matriz resultante es una representación numérica de los datos de texto que se puede utilizar para entrenar modelos de aprendizaje automático. Cada fila de la matriz representa un ejemplo (sitio web) y cada columna representa una palabra o término en el conjunto de datos. El valor en cada celda de la matriz es la frecuencia de ese término en ese ejemplo. La función fit_transform() realiza tanto el ajuste del vectorizador al conjunto de datos como la transformación de los datos a la matriz de términos de frecuencia de documentos.

Figura 3.27

Conversión de las imágenes de texto en una matriz de recuento de términos o

```
feature = cv.fit_transform(phish_data.text_sent)
```

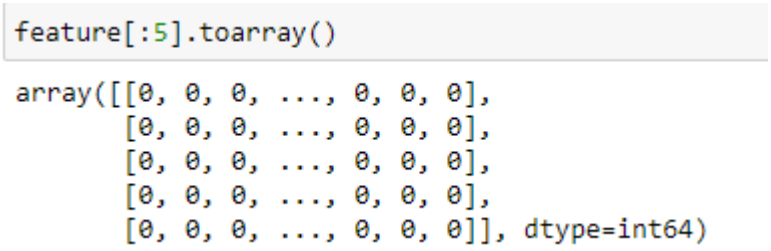
Nota: la presente imagen se observa la conversión de las imágenes de texto en una matriz de recuento de términos o palabras.

Visualización de las primeras cinco filas de la matriz de características generada a partir de los datos de texto:

Una vez completada la etapa de preprocesamiento y transformación de los datos de texto en una matriz de recuento de términos, procedemos a explorar las características generadas a partir de los datos. En este sentido, para visualizar las primeras cinco filas de la matriz de características, utilizamos la función feature[:5].toarray(). Esta operación nos permitirá observar cómo las diferentes palabras del vocabulario se distribuyen a lo largo de las diferentes muestras de texto en el conjunto de datos. Cada fila de la matriz representa una muestra y cada columna corresponde a una palabra del vocabulario. La entrada en cada celda indica cuántas veces aparece una palabra en una muestra determinada.

Figura 3.28

Visualización de las primeras cinco filas de la matriz



Nota: la presente imagen se observa la Visualización de las primeras cinco filas de la matriz

3.2.5. Modelado

3.2.5.1 Creación de un Modelo

División de Datos de Feature y Label (Entrenamiento y Prueba):

Después de haber recolectado y procesado los datos, es esencial llevar a cabo el entrenamiento de un modelo de detección de ataques phishing utilizando técnicas de machine learning. Para lograrlo, es necesario separar el conjunto de datos, la función "train_test_split()" de la biblioteca "sklearn", que nos permite dividir los datos de manera aleatoria. Posteriormente, se asignan los conjuntos de datos a las variables "trainX", "testX", "trainY" y "testY", de manera que los datos de entrenamiento se utilizan para ajustar el modelo y los datos de prueba se utilizan para evaluar su capacidad de generalización. Esta fase de entrenamiento es crucial para obtener un modelo preciso y confiable que pueda ser utilizado para la detección de ataques phishing con alto grado de eficiencia y eficacia.

La función train_test_split() es una función de la biblioteca Scikit-Learn que se utiliza para dividir un conjunto de datos en dos subconjuntos: uno para entrenamiento y otro para pruebas. En este caso, los argumentos que se pasan a la función son feature y phish_data.Label. La variable feature contiene la matriz de características generada a partir de los datos de texto de la columna text_sent del dataframe phish_data, mientras que phish_data.Label contiene las etiquetas asociadas a cada URL, es decir, si la URL es legítima o phishing (**1 si es phishing y 0 si es legítima).**

La función devuelve cuatro objetos: trainX, testX, trainY y testY**. trainX y trainY son los datos de entrenamiento** de las características y las etiquetas, respectivamente, mientras que **testX y testY son los datos de prueba** de las características y las etiquetas, respectivamente. La división se realiza aleatoriamente y se puede especificar la proporción de los datos que se desea utilizar para entrenamiento y prueba mediante el parámetro test_size de la función, como en este caso estamos poniendo un valor para el entrenamiento = %70 y para la prueba = 20% de datos.

Figura 3.29

División de datos en dos conjuntos (entrenamiento y prueba)

Estamos Dividiendo EL FEATURE Y EL LABEL = si es bueno o malo) no estamos dividiendo el LABEL y la URL aun

```
# divide los datos en dos conjuntos (entrenamiento y prueba)
trainX, testX, trainY, testY = train_test_split(feature, phish_data.Label,train_size=0.7,test_size=0.3)
```

Nota: la presente imagen se observa que divide los datos en dos conjuntos (entrenamiento y prueba)

Creación de un gráfico de barras que muestra la proporción de datos que se asignaron a cada conjunto.

Una vez se ha llevado a cabo la pertinente división de los datos y se ha asignado un valor específico a cada conjunto de entrenamiento y prueba, se procede a la creación de una representación gráfica en forma de gráfico de barras que permita una mejor visualización y análisis conjunto de los datos en cuestión. Esta representación gráfica posibilita la observación detallada y la identificación de patrones y tendencias relevantes en los datos de entrenamiento y prueba, lo que se traduce en un análisis más profundo y preciso de los mismos. De esta forma, se facilita la toma de decisiones informada y certera en torno a la evaluación y optimización de los modelos y algoritmos que se utilicen para el procesamiento de los datos.

Figura 3.29.1

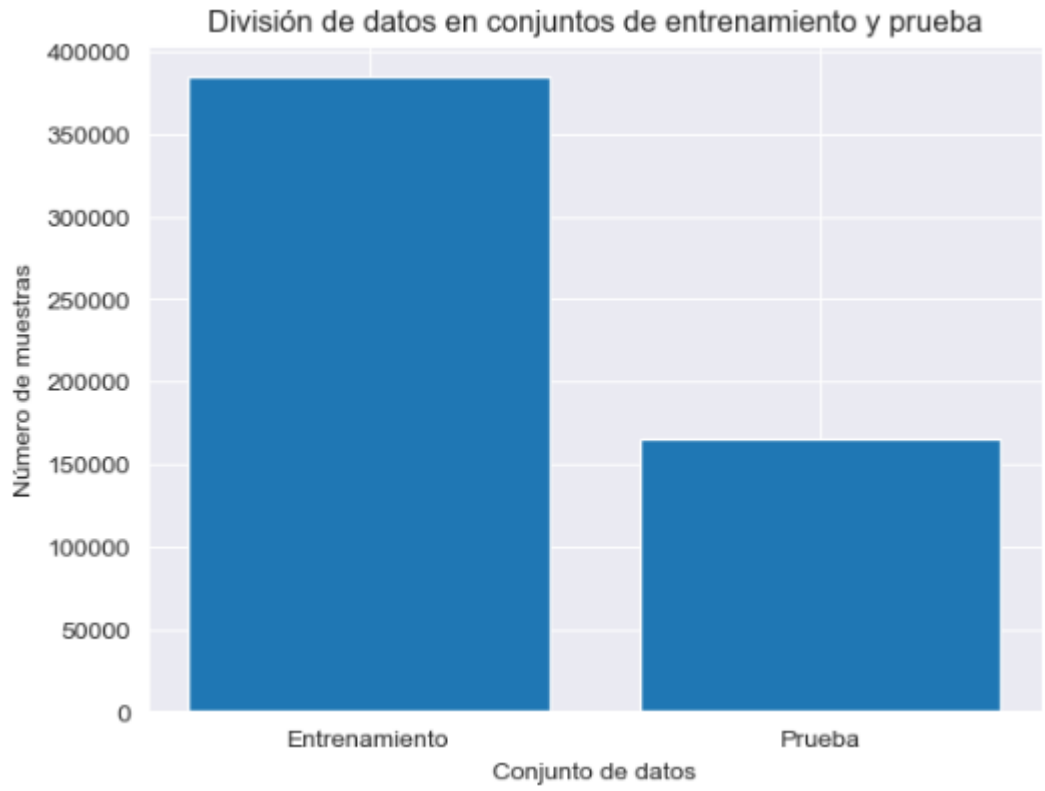
Creación de un gráfico de barras

Entrada

```
#Se Calcula el tamaño de cada conjunto
train_count = len(trainY)
test_count = len(testY)
```

```
plt.bar(['Entrenamiento', 'Prueba'], [train_count, test_count])
plt.xlabel('Conjunto de datos')
plt.ylabel('Número de muestras')
plt.title('División de datos en conjuntos de entrenamiento y prueba')
plt.show()
```

Salida.



Nota: la presente imagen se observa la creación de un gráfico de barras.

3.2.5.2 Algoritmos de Machine Learning

Regresión Logística:

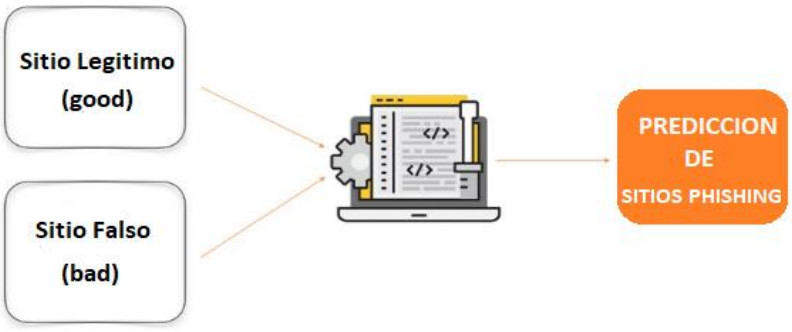
La regresión logística es un modelo de aprendizaje supervisado que se utiliza para predecir una variable binaria, es decir, una variable que toma uno de dos valores posibles (por ejemplo, **"sí" o "no", "1 o 0", "abierto o cerrado", "good o bad**"). En el contexto de la detección de ataques phishing, la regresión logística será utilizada para predecir si un sitio web es **legítimo o falso** como en los datos que dimos para los sitios en esta ocasión si un sitio es **legítimo esta como "good" y si es falso o fraudulento esta como "bad".**

Para utilizar la regresión logística en la detección de ataques phishing, es necesario entrenar el modelo utilizando un conjunto de datos etiquetados que incluya ejemplos de sitios web legítimos (good) y falsos (bad). Durante el entrenamiento, el modelo aprenderá a reconocer los patrones con los datos proporcionados que le permitirán hacer predicciones precisas.

Una vez que se ha entrenado el modelo de regresión logística, se puede utilizar para hacer predicciones sobre nuevos datos. "Por lo tanto, el modelo de regresión logística no es un modelo ya entrenado para la detección de ataques Phishing", "sino que necesita ser entrenado" con datos específicos proporcionados para poder hacer predicciones precisas.

Figura 3.30

Análisis Predictivo Con la Regresión Logística



Nota: la presente imagen se observa el análisis predictivo con el algoritmo de regresión logística.

Creación de un modelo que se debe entrenar utilizando datos del entrenamiento:

En esta línea de código se crea un objeto de modelo de regresión logística **vacío** utilizando la clase "LogisticRegression" de la biblioteca "sklearn". La regresión logística es un modelo de clasificación binaria que se utiliza ampliamente en machine learning para la predicción de variables de salida binarias. Este modelo vacío es la base para entrenar y ajustar el modelo con los datos que se proporcionan posteriormente.

Al entrenar el modelo, el algoritmo de regresión logística se ajusta a los datos de entrenamiento mediante la búsqueda de un plano o hiperplano que divide el espacio de características en dos regiones, una para cada clase de salida. El objetivo es maximizar la probabilidad de que los datos de entrenamiento pertenezcan a su respectiva clase, lo que se logra mediante el ajuste de los parámetros del modelo a través de un proceso iterativo de descenso de gradiente.

En resumen, **la creación de un objeto "Lr"** utilizando la clase "LogisticRegression" de la biblioteca "sklearn" Es el primer paso para entrenar un modelo de regresión logística para la detección de ataques Phishing**.**

Figura 3.31

Creación de un objeto "lr" que es un modelo de regresión logística

Nota: la presente imagen se observa la creación de un objeto "lr" que es un modelo de regresión logística

3.2.5.3 Entrenamiento del Modelo con el Algoritmo Regresión Logística

Entrenamiento del modelo de regresión logística:

Una vez creado el objeto “lr” que es un modelo de regresión logística vacía, procedemos a entrenar el modelo de regresión logística utilizando los datos de entrenamiento que proporcionamos. En particular, la función "fit" ajusta los parámetros del modelo utilizando los datos de entrenamiento proporcionados **(trainX y trainY)** para minimizar la función de pérdida definida para el modelo de regresión logística. Una vez entrenado, el modelo puede ser utilizado para predecir la variable objetivo (en

este caso, la etiqueta de phishing) para nuevos datos no vistos.

Figura 3.32

Entrenamiento del modelo de regresión logística con los datos del entrenamiento y

```
lr.fit(trainX,trainY)
```

Nota: la presente imagen se observa el entrenamiento de regresión logística.

Calculando el puntaje de precisión:

Bien ya utilizado los datos proporcionados para el entrenamiento procedemos a la línea "lr.score(testX,testY)" calcula el puntaje de precisión (accuracy score) del modelo de regresión logística en los datos de prueba. Para hacerlo, el modelo se aplica a los datos de prueba "testX" y se compara con las etiquetas de prueba "testY" para ver cuántas predicciones son correctas y cuántas son incorrectas. El valor de precisión será un número entre 0 y 1, donde 1 significa que el modelo tiene una precisión del 100%, en este caso el modelo tiene una precisión del 96%.

Figura 3.32

Calculando el puntaje de precisión.

```
lr.(scoretestX,testY)

0.9637606762926233
```

Nota: la presente imagen se observa el cálculo de puntaje de precisión.

Devolviendo la precisión media del modelo en los datos de prueba:

Después de haber obtenido el porcentaje de precisión del modelo de regresión logística, el cual arrojó un valor del 0.9637606762926233, procedemos a crear un diccionario llamado "Scores_ml" para almacenar el valor de precisión obtenido. Para ello, utilizamos la función "score" de la biblioteca "sklearn", la cual nos permite calcular el valor de precisión en los datos de prueba "testX" y "testY". Luego, para redondear el valor de precisión a dos decimales que vendría a ser **0.96** utilizamos la función "np.round()". Finalmente, agregamos el valor de precisión al diccionario "Scores_ml" con la clave "Logistic Regression". De esta manera, podemos tener un registro del desempeño de nuestro modelo de regresión logística en la detección de ataques

phishing.

Tener una estructura de datos como un diccionario ayuda a mantener la organización y facilita la comparación de diferentes modelos. Por ejemplo, si desea comparar la precisión de varios modelos, simplemente puede acceder a los valores de precisión almacenados en el diccionario y compararlos.

Figura 3.33

Devolución de la precisión media del modelo en los datos de prueba

```
Scores_ml = {}
Scores_ml['Logistic Regression'] = np.round(lr.score(testX,testY),2)
```

Nota: la presente imagen se observa la devolución de la precisión media del modelo.

Proporcionamos un informe de la predicción y el rendimiento del modelo utilizando la regresión logística:

Para comenzar con el informe de predicción empezamos a imprimir la precisión de los datos de entrenamiento y los datos de prueba. Donde:

Lr.score(trainX,trainY) = Precisión de los datos del Entrenamiento

Lr.score(testX,textY) = Precisión de los datos de prueba

Figura 3.34

Informe de la precisión de los datos de entrenamiento y datos de prueba

Entrada.

```
print('Training Accuracy :',lr.score(trainX,trainY))
print('Testing Accuracy :',lr.score(testX,testY))
```

Salida

```
Training Accuracy : 0.9773669992645792
Testing Accuracy : 0.9637606762926233
```

Nota: la presente imagen se observa la precisión de los datos de entrada y salida

Luego de imprimir las precisiones de los datos de entrenamiento y prueba procedemos a crear una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

La matriz de confusión es una herramienta importante para evaluar la precisión

de un modelo de clasificación, ya que permite visualizar el número de predicciones correctas e incorrectas del modelo en función de las clases reales.

El código utiliza la biblioteca de Pandas para crear un DataFrame que muestra la matriz de confusión. La matriz de confusión se crea utilizando los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación (lr.predict(testX)).

La matriz de confusión tiene dos dimensiones: filas y columnas. Las filas representan las clases reales (Actual:Bad y Actual:Good) y las columnas representan las predicciones del modelo (Predicted:Bad y Predicted:Good).

Filas = (Actual:Bad y Actual:Good)

Columnas = (Predicted:Bad y Predicted:Good).

En la diagonal de la matriz se encuentran los valores correctos, es decir, las predicciones que coinciden con las clases reales. Fuera de la diagonal, se encuentran los valores incorrectos, es decir, las predicciones que no coinciden con las clases reales.

En resumen, este código crea una matriz de confusión para evaluar el rendimiento de un modelo de clasificación y permite visualizar de manera clara y sencilla el número de predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

Figura 3.35

Creación de una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

```
con_mat = pd.DataFrame(confusion_matrix(lr.predict(testX), testY),
          columns = ['Predicted:Bad', 'Predicted:Good'],
          index = ['Actual:Bad', 'Actual:Good'])
```

Nota: la presente imagen se observa la creación de una matriz de confusión.

Creación de un informe de clasificación:

Una vez ya creada la matriz procedemos a crear un informe de clasificación utilizando la función **“classification_report”** de la biblioteca Scikit-learn.

Este informe proporciona una evaluación más detallada del rendimiento del

modelo de clasificación en términos de varias métricas de evaluación, como la **precisión, el recall, el puntaje F1 y el soporte** para cada clase.

En concreto, el código utiliza los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación "**(lr.predict(testX))"** para generar el informe de clasificación.

El argumento **"target_names"** se utiliza para especificar los nombres de las clases en el informe. En este caso, las clases se denominan 'Bad' y 'Good'.

La función classification_report genera un informe de clasificación que incluye las siguientes métricas:

- Precisión: La precisión mide la proporción de predicciones positivas que son correctas. En otras palabras, la precisión mide la capacidad del modelo de clasificación para identificar correctamente los casos positivos.
- **Recall:** El recall mide la proporción de casos positivos que se identificaron correctamente. En otras palabras, el recall mide la capacidad del modelo de clasificación para detectar todos los casos positivos.
- **F1 Score:** El puntaje F1 es una medida de la precisión general del modelo de clasificación. Es la media armónica de la precisión y el recall.
- **Soporte (Support):** El soporte es el número de observaciones de cada clase en los datos de prueba.

En resumen, este código crea un informe de clasificación que proporciona una evaluación detallada del rendimiento del modelo de clasificación en términos de varias métricas de evaluación, lo que permite una comprensión más completa del rendimiento del modelo.

Figura 3.36

Creación de un informe de clasificación.

Entrada.

```
#imprime la precisión del modelo en los datos de entrenamiento y prueba con la función
print('Precicion de entrenamiento :',lr.score(trainX,trainY))
print('Exactitud de Prueba :',lr.score(testX,testY))
```

Nota: la presente imagen se observa la creación de un informe de clasificación.

Figura 3.37

Creación de un informe de clasificación.

Salida.

```
Precicion de entrenamiento : 0.9779763978967186
Exactitud de Prueba : 0.9620154850610422
```

Creación de un gráfico de matriz de confusión.

Entrada.

```
#crea una matriz de confusión utilizando la función confusion_matrix() de scikit-learn, que muestra cuántas predicciones
#de cada clase fueron correctas e incorrectas. La matriz de confusión se muestra en un mapa de calor utilizando
#la biblioteca seaborn.
con_mat = pd.DataFrame(confusion_matrix(lr.predict(testX), testY),
            columns = ['Predicted:Bad', 'Predicted:Good'],
            index = ['Actual:Bad', 'Actual:Good'])

#imprime un informe de clasificación que muestra la precisión, recuperación, puntuación muestra los resultados en tiempo actual
print('\nInforme de Clasificacion\n')
print(classification_report(lr.predict(testX), testY,
                           target_names =['Bad','Good']))
```

Salida.

```
CLASSIFICATION REPORT

              precision    recall  f1-score   support

         Bad       0.90      0.97      0.93     36614
        Good       0.99      0.96      0.97    100723

    accuracy                           0.96    137337
   macro avg       0.95      0.96      0.95    137337
weighted avg       0.97      0.96      0.96    137337
```

Nota: la presente imagen se observa la creación de un informe de clasificación.

Creación de un gráfico de matriz de confusión:

Una vez ya creado el informe de clasificación procedemos a crear un gráfico de matriz de confusión utilizando la biblioteca Seaborn.

El gráfico de matriz de confusión es una herramienta visual que permite analizar el rendimiento de un modelo de clasificación. En el eje x se representan las predicciones del modelo y en el eje y se representan las clases reales.

El código utiliza la matriz de confusión creada previamente (en el código que presentaste anteriormente) para generar el gráfico. El argumento annot=True hace que los valores de la matriz de confusión se muestren dentro de cada celda. El argumento fmt='d' hace que los valores se muestren como números enteros.

El argumento cmap="YlGnBu" establece la paleta de colores utilizada en el gráfico. En este caso, se utiliza la paleta de colores "YlGnBu", que va desde el amarillo

claro (representando valores bajos) hasta el azul oscuro (representando valores altos).

La función plt.figure(figsize= (6,4)) establece el tamaño de la figura del gráfico.

En resumen, este código crea un gráfico de matriz de confusión que permite visualizar de manera clara y sencilla el rendimiento del modelo de clasificación en términos de las predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

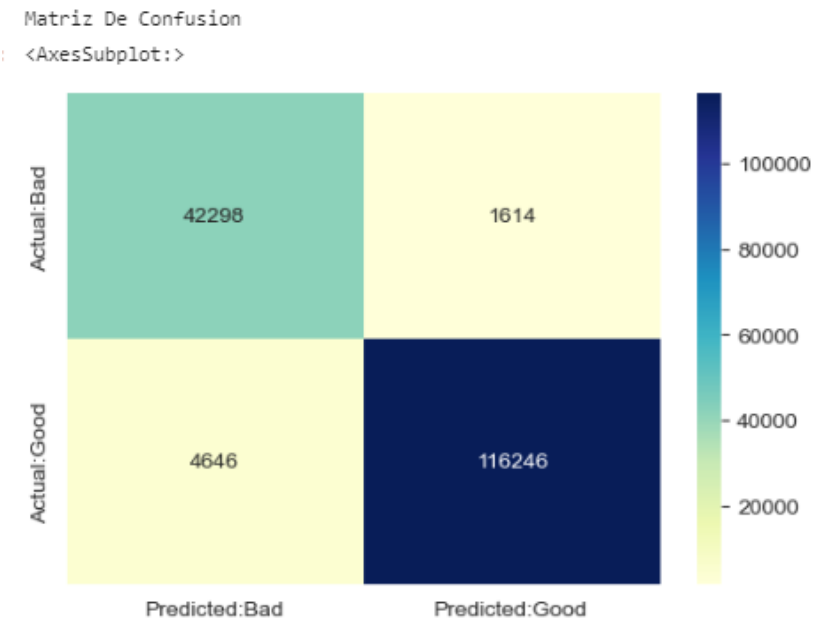
Figura 3.38

Creación de una matriz de confusión.

Entrada.

```
print('\nMatriz De Confusion')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

Salida.



Nota: la presente imagen se observa la creación de una matriz de confusión.

3.2.5.4 Creación de un Modelo con el algoritmo Naive Bayes Multinomial:

Bien ya hicimos un informe acerca de la creacion de la matriz de confusión procedemos a crear un objeto de la clase MultinomialNB que es parte del módulo

naive_bayes de la biblioteca Scikit-learn.

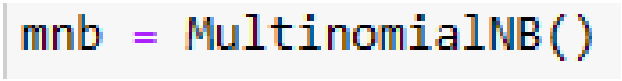
MultinomialNB es un modelo de clasificación basado en el teorema de Bayes, que se utiliza específicamente para clasificar datos discretos, como conteos de palabras en un documento de texto.

El modelo MultinomialNB se utiliza comúnmente en problemas de clasificación de texto, como la clasificación de correos electrónicos como spam o no spam, la clasificación de noticias en diferentes categorías, entre otros.

Una vez que se crea el objeto MultinomialNB, se puede entrenar con datos de entrenamiento utilizando el método fit y se puede utilizar para realizar predicciones en nuevos datos utilizando el método predict.

Figura 3.39

Creación de una instancia del clasificador Naive Bayes Multinomial



Nota: la presente imagen se observa la Creación de una instancia del clasificador Naive Bayes Multinomial

3.2.5.5 Entrenamiento del Modelo con el Algoritmo MultinomialNB

Entrenamiento del modelo de clasificación MultinomialNB

Una vez creado objeto de clase MultinomialNB Procedemos a realizar el entrenamiento del modelo de clasificación MultinomialNB utilizando los datos de entrenamiento.

trainX y trainY son las características (variables independientes) y las etiquetas (variable dependiente) de los datos de entrenamiento, respectivamente.

El método fit de la clase MultinomialNB se utiliza para entrenar el modelo con los datos de entrenamiento. Durante el entrenamiento, el modelo ajustará los parámetros internos utilizando el algoritmo de máxima verosimilitud. Esto se hace para que el modelo pueda aprender a clasificar los datos correctamente y así poder hacer predicciones precisas en datos nuevos.

Una vez que el modelo ha sido entrenado utilizando los datos de entrenamiento, se puede usar para hacer predicciones en nuevos datos.

En resumen, el código que presentas entrena el modelo MultinomialNB con los datos de entrenamiento trainX y trainY para que pueda aprender a clasificar los datos de entrada y hacer predicciones precisas en datos nuevos.

Figura 3.40

Entrenamiento del modelo MultinomialNB

```
mnb.fit(trainX,trainY)
```

Nota: la presente imagen se observa el entrenamiento del modelo MultinomialNB con los datos del entrenamiento y datos de prueba.

Calculando el puntaje de precisión.

Bien ya utilizado los datos proporcionados para el entrenamiento procedemos a la línea "mnb.score(testX,testY)" calcula el puntaje de precisión (accuracy score) del modelo de MultinomialMB en los datos de prueba. Para hacerlo, el modelo se aplica a los datos de prueba "testX" y se compara con las etiquetas de prueba "testY" para ver cuántas predicciones son correctas y cuántas son incorrectas. El valor de precisión será un número entre 0 y 1, donde 1 significa que el modelo tiene una precisión del 100%, en este caso el modelo tiene una precisión del 95%.

Figura 3.41

Calculando el puntaje de precisión.

```
mnb.score(testX,testY)
```

```
0.9572220159170508
```

Nota: la presente imagen se observa el cálculo de puntaje de precisión.

Almacenando el resultado de la precisión del Modelo Multinomial Naive Bayes en el diccionario "Scores_ml" con la clave "MultinomialNB".

Después de haber obtenido el porcentaje de precisión del modelo de regresión logística, el cual arrojó un valor del 0.953760676292623 3, procedemos a almacenar el resultado de la precisión en el diccionario llamado "Scores_ml" Para ello, utilizamos la función "score" de la biblioteca "sklearn", la cual nos permite calcular el valor de precisión en los datos de prueba "testX" y "testY". Luego, para redondear el valor de

precisión a dos decimales que vendría a ser **0.95** utilizamos la función "np.round()". Finalmente, agregamos el valor de precisión al diccionario "Scores_ml" con la clave "Logistic Regression". De esta manera, podemos tener un registro del desempeño de nuestro modelo de regresión logística en la detección de ataques phishing.

Tener una estructura de datos como un diccionario ayuda a mantener la organización y facilita la comparación de diferentes modelos. En esta ocasión ya tenemos 2 modelos con diferentes precisiones dentro del diccionario el cual vamos hacer la comparación más adelante.

Figura 3.42

Devolución de la precisión media del modelo en los datos de prueba.

```
Scores_ml['MultinomialNB'] = np.round(mnb.score(testX,testY),2)
```

Nota: la presente imagen se observa la devolución de la precisión media del modelo y agregación del valor de la precisión al diccionario.

Proporcionamos un informe de la predicción y el rendimiento del modelo MultinomialNB

Para comenzar con el informe de predicción empezamos a imprimir la precisión de los datos de entrenamiento y los datos de prueba. Donde:

mnb.score(trainX,trainY) = Precisión de los datos del Entrenamiento

mnb.score(testX,textY) = Precisión de los datos de prueba

Figura 3.43

Informe de la precisión de los datos de entrenamiento y datos de prueba

Entrada.

```
print('Precicion del entrenamiento:',mnb.score(trainX,trainY))
print('Exactitud de prueba :',mnb.score(testX,testY))
```

Salida

```
Precicion del entrenamiento: 0.9737765965746265
Exactitud de prueba : 0.95606902745079
```

Nota: la presente imagen se observa la precisión de los datos de entrada y salida

Luego de imprimir las precisiones de los datos de entrenamiento y prueba

procedemos a crear una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

La matriz de confusión es una herramienta importante para evaluar la precisión de un modelo de clasificación, ya que permite visualizar el número de predicciones correctas e incorrectas del modelo en función de las clases reales.

El código utiliza la biblioteca de Pandas para crear un DataFrame que muestra la matriz de confusión. La matriz de confusión se crea utilizando los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación (lr.predict(testX)).

La matriz de confusión tiene dos dimensiones: filas y columnas. Las filas representan las clases reales (Actual:Bad y Actual:Good) y las columnas representan las predicciones del modelo (Predicted:Bad y Predicted:Good).

Filas = (Actual:Bad y Actual:Good)

Columnas = (Predicted:Bad y Predicted:Good).

En la diagonal de la matriz se encuentran los valores correctos, es decir, las predicciones que coinciden con las clases reales. Fuera de la diagonal, se encuentran los valores incorrectos, es decir, las predicciones que no coinciden con las clases reales.

En resumen, este código crea una matriz de confusión para evaluar el rendimiento de un modelo de clasificación y permite visualizar de manera clara y sencilla el número de predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

Figura 3.44

Creación de una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

```
con_mat = pd.DataFrame(confusion_matrix(mnb.predict(testX), testY),
            columns = ['Predicted:Bad', 'Predicted:Good'],
            index = ['Actual:Bad', 'Actual:Good'])
```

Nota: la presente imagen se observa la creación de una matriz de confusión.

Creación de un informe de clasificación:

Una vez ya creada la matriz procedemos a crear un informe de clasificación utilizando la función **“classification_report”** de la biblioteca Scikit-learn.

Este informe proporciona una evaluación más detallada del rendimiento del modelo de clasificación en términos de varias métricas de evaluación, como la **precisión, el recall, el puntaje F1 y el soporte** para cada clase.

En concreto, el código utiliza los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación “**(lr.predict(testX))”** para generar el informe de clasificación.

El argumento **“target_names”** se utiliza para especificar los nombres de las clases en el informe. En este caso, las clases se denominan 'Bad' y 'Good'.

La función classification_report genera un informe de clasificación que incluye las siguientes métricas:

- Precisión: La precisión mide la proporción de predicciones positivas que son correctas. En otras palabras, la precisión mide la capacidad del modelo de clasificación para identificar correctamente los casos positivos.
- Recall**:** El recall mide la proporción de casos positivos que se identificaron correctamente. En otras palabras, el recall mide la capacidad del modelo de clasificación para detectar todos los casos positivos.
- F1 Score: El puntaje F1 es una medida de la precisión general del modelo de clasificación. Es la media armónica de la precisión y el recall.
- Soporte (Support): El soporte es el número de observaciones de cada clase en los datos de prueba.

En resumen, este código crea un informe de clasificación que proporciona una evaluación detallada del rendimiento del modelo de clasificación en términos de varias métricas de evaluación, lo que permite una comprensión más completa del rendimiento del modelo.

Figura 3.45

Creación de una matriz de confusión

Entrada.

```
print('\nINFORME DE CLASIFICACION\n')
print(classification_report(mnb.predict(testX), testY,
                           target_names =['Bad','Good']))
```

Salida.

```
CLASSIFICATION REPORT

              precision    recall  f1-score   support

         Bad       0.91      0.93      0.92     38296
        Good       0.97      0.97      0.97     99041

    accuracy                           0.96    137337
   macro avg       0.94      0.95      0.95    137337
weighted avg       0.96      0.96      0.96    137337
```

Nota: la presente imagen se observa la creación de un informe de clasificación.

Creación de un gráfico de matriz de confusión.

Una vez ya creado el informe de clasificación procedemos a crear un gráfico de matriz de confusión utilizando la biblioteca Seaborn.

El gráfico de matriz de confusión es una herramienta visual que permite analizar el rendimiento de un modelo de clasificación. En el eje x se representan las predicciones del modelo y en el eje y se representan las clases reales.

El código utiliza la matriz de confusión creada previamente (en el código que presentaste anteriormente) para generar el gráfico. El argumento annot=True hace que los valores de la matriz de confusión se muestren dentro de cada celda. El argumento fmt='d' hace que los valores se muestren como números enteros.

El argumento cmap="YlGnBu" establece la paleta de colores utilizada en el gráfico. En este caso, se utiliza la paleta de colores "YlGnBu", que va desde el amarillo claro (representando valores bajos) hasta el azul oscuro (representando valores altos).

La función plt.figure(figsize= (6,4)) establece el tamaño de la figura del gráfico.

En resumen, este código crea un gráfico de matriz de confusión que permite visualizar de manera clara y sencilla el rendimiento del modelo de clasificación en términos de las predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

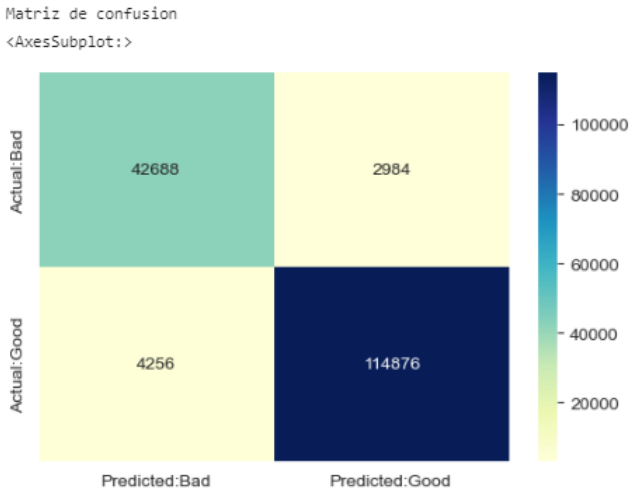
Figura 3.46

Creación de una matriz de confusión

Entrada.

```
print('\nMatriz de confusion')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

Salida.



Nota: la presente imagen se observa la creación de una matriz de confusión.

Creación de un gráfico de barras para visualizar la precisión de diferentes modelos de aprendizaje almacenados en el diccionario Scores_ml:

Una vez que ya acabamos de hacer el informe del modelo MultinomialNB procedemos a crear un gráfico de barras para visualizar la precisión de los 2 modelos de aprendizaje automático, Regresión Logística y MultinomialNB almacenados en el diccionario Scores_ml.

La función pd.DataFrame.from_dict se utiliza para convertir el diccionario en un objeto DataFrame de Pandas, donde las claves del diccionario se convierten en el

índice del DataFrame y los valores del diccionario se convierten en una sola columna 'Accuracy'.

El argumento orient se establece en 'index' para especificar que las claves del diccionario se utilizan como índice del DataFrame.

Luego, se utiliza la librería seaborn para crear un gráfico de barras utilizando la función sns.barplot. El argumento acc.index se utiliza para especificar los valores del eje x del gráfico, que corresponden a las claves del diccionario. El argumento acc.Accuracy se utiliza para especificar los valores del eje y del gráfico, que corresponden a la precisión de cada modelo de aprendizaje automático.

El argumento sns.set_style se utiliza para establecer el estilo del gráfico en 'darkgrid'.

En resumen, este código crea un gráfico de barras que muestra la precisión de diferentes modelos de aprendizaje automático almacenados en el diccionario, lo que permite comparar y visualizar fácilmente el rendimiento relativo de los modelos.

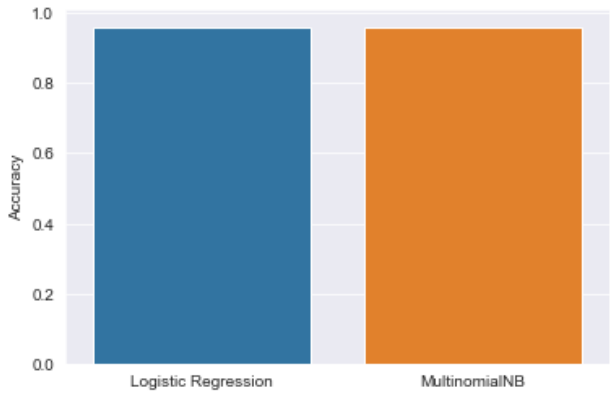
Figura 3.47

Creación de un gráfico de barras para la comparación de modelos

Entrada.

```
print('\nCONFUSION MATRIX')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

Salida.



Nota: la presente imagen se observa la creación un gráfico de barras de los 2 modelos. Entrenamiento con el modelo de regresión logística y canalización:

3.2.5.5 Organización y Automatización del Modelo entrenado

Como vemos el modelo de regresión logística su precisión es mejor que la MultinomialNB por ende utilizaremos la regresión logística para canalización con sklearn

La Canalizacion (pipeline) se crea utilizando la función make_pipeline de la librería sklearn.pipeline, que permite concatenar varios pasos de procesamiento de datos en un solo objeto.

En este caso, la tubería tiene dos pasos:

El primer paso utiliza la clase CountVectorizer de la librería sklearn.feature_extraction.text para realizar una transformación de los datos de texto de entrada. El argumento tokenizer se establece en RegexpTokenizer(r'[A-Za-z]+').tokenize para especificar que se deben utilizar solo palabras alfabéticas en minúsculas como tokens. El argumento stop_words se establece en 'english' para indicar que se deben eliminar las palabras comunes en inglés durante el proceso de transformación. La función CountVectorizer crea una matriz de términos de frecuencia de documentos (TF-IDF) para representar los datos de texto.

El segundo paso utiliza la clase LogisticRegression de la librería sklearn.linear_model para entrenar un modelo de clasificación de regresión logística en la matriz de TF-IDF creada en el primer paso.

La tubería completa se puede entrenar con los datos de entrada utilizando el método fit de la clase Pipeline.

En resumen, este código define un modelo de clasificación de regresión logística que utiliza una canalizacion para realizar una transformación de los datos de texto y entrenar el modelo de forma automática. La canalizacion utiliza la clase CountVectorizer para crear una matriz de términos de frecuencia de documentos y la clase LogisticRegression para entrenar un modelo de clasificación.

Figura 3.48

Definición de un modelo de clasificación de regresión logística

```
pipeline_ls = make_pipeline(CountVectorizer(tokenizer = RegexpTokenizer(r'[A-Za-z]+').
tokenize,stop_words='english'), LogisticRegression())
```

Nota: la presente imagen se observa la definición de un modelo de regresión logística

División del conjunto de datos phish_data en un conjunto de entrenamiento (trainX, trainY) y un conjunto de prueba (testX, testY):

Una vez ya hecho la definición del modelo de regresión logistica procedemos a el código trainX, testX, trainY, testY = train_test_split(phish_data.URL, phish_data.Label) divide el conjunto de datos phish_data en dos conjuntos: uno para entrenamiento y otro para pruebas. En particular, train_test_split es una función de la librería sklearn.model_selection que divide aleatoriamente los datos en un conjunto de entrenamiento y un conjunto de pruebas.

Los dos primeros argumentos de la función son phish_data.URL y phish_data.Label, que representan las características (URL) y la etiqueta (Label) del conjunto de datos. **train_test_split:** divide estos datos en cuatro conjuntos de salida:

trainX: el conjunto de entrenamiento de características (URL) para el modelo.

testX: el conjunto de prueba de características (URL) para el modelo.

trainY: el conjunto de entrenamiento de etiquetas (Label) correspondientes a las características en trainX.

testY: el conjunto de prueba de etiquetas (Label) correspondientes a las características en testX.

El propósito de dividir los datos en conjuntos de entrenamiento y pruebas es evaluar la capacidad del modelo para generalizar, es decir, su capacidad para realizar predicciones precisas en datos no vistos previamente. El modelo se entrena en el conjunto de entrenamiento y se evalúa en el conjunto de pruebas para estimar su precisión en datos no vistos previamente.

Se está dividiendo los datos utilizando solo la variable "phish_data.URL", que es la única característica que se está utilizando para entrenar el modelo. En este caso, no se están utilizando otras características adicionales.

Figura 3.49

Definición del conjunto de datos en un conjunto de entrenamientos.

```
trainX, testX, trainY, testY = train_test_split(phish_data.URL, phish_data.Label)
```

Nota: la presente imagen se observa la definicion del conjunto de datos phish_data en un conjunto de entrenamiento (trainX, trainY) y un conjunto de prueba (testX, testY) Creación de un gráfico de barras para los datos de entrenamiento y prueba:

Una vez se ha llevado a cabo la pertinente división de los datos y se ha asignado un valor específico a cada conjunto de entrenamiento y prueba, se procede a la creación de una representación gráfica en forma de gráfico de barras que permita una mejor visualización y análisis conjunto de los datos en cuestión. Esta representación gráfica posibilita la observación detallada y la identificación de patrones y tendencias relevantes en los datos de entrenamiento y prueba, lo que se traduce en un análisis más profundo y preciso de los mismos. De esta forma, se facilita la toma de decisiones informada y certera en torno a la evaluación y optimización de los modelos y algoritmos que se utilicen para el procesamiento de los datos.

Figura 3.50

Creación de un gráfico de barras

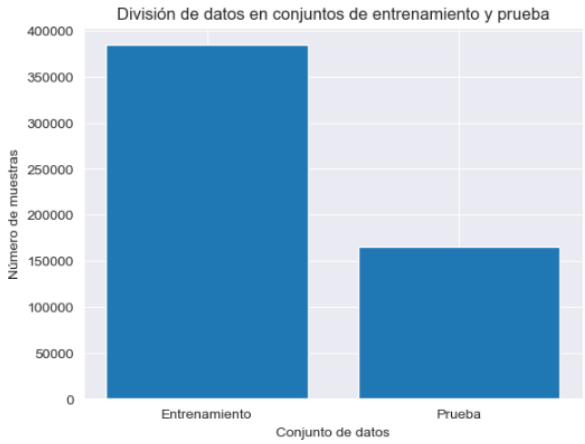
Entrada.

```
#Se Calcula el tamaño de cada conjunto
train_count = len(trainY)
test_count = len(testY)
```

```
plt.bar(['Entrenamiento', 'Prueba'], [train_count, test_count])
plt.xlabel('Conjunto de datos')
plt.ylabel('Número de muestras')
plt.title('División de datos en conjuntos de entrenamiento y prueba')
plt.show()
```

Nota: la presente imagen se observa la creación de un gráfico de barras.

Salida.



Nota: la presente imagen se observa la creación de un gráfico de barras.

Entrenamiento de un modelo de clasificación de regresión logística utilizando una canalizacion (pipeline) que procesa los datos de texto de entrada.

Una vez hecho la definición del conjunto de datos procedemos a entrenar los datos con la línea de codigo pipeline_**ls.fit(trainX,trainY) que entrena un modelo de clasificación de regresión logística** utilizando una tubería (pipeline) que procesa los datos de texto de entrada.

El primer argumento trainX es un conjunto de características de entrenamiento que se utilizarán para entrenar el modelo. El segundo argumento trainY es un conjunto correspondiente de etiquetas de clase que se utilizan para entrenar el modelo supervisado.

La tubería pipeline_ls procesa los datos de texto de entrada, tokeniza el texto, elimina las palabras comunes y crea una matriz de términos de frecuencia de documentos (TF-IDF) para representar los datos de texto. Luego, el modelo de regresión logística se entrena en la matriz de TF-IDF creada.

En resumen, pipeline_ls.fit(trainX,trainY) entrena un modelo de clasificación de regresión logística utilizando una tubería que procesa los datos de texto de entrada y crea una matriz de TF-IDF para representar los datos.

Figura 3.51

Entrenamiento de un modelo de clasificación de regresión logistica

Nota: la presente imagen se observa el entrenamiento de un modelo de clasificación de regresión logistica.

Calculando el puntaje de precisión del modelo de clasificación de regresión logística entrenado en la tubería pipeline_ls utilizando un conjunto de datos de prueba.

Una vez ya entrenado el modelo de clasificación de regresión logistica procedemos a usar la función pipeline_ls.score(testX, testY) que evalúa la precisión (accuracy) del modelo de clasificación de regresión logística entrenado en la tubería pipeline_ls utilizando un conjunto de datos de prueba.

El primer argumento testX es un conjunto de características de prueba que se utilizarán para evaluar el modelo. El segundo argumento testY es un conjunto correspondiente de etiquetas de clase que se utilizarán para evaluar el modelo supervisado.

La función score del modelo de regresión logística devuelve la precisión del modelo en el conjunto de datos de prueba testX y testY. La precisión se define como el número de predicciones correctas del modelo dividido por el número total de predicciones.

En resumen, pipeline_ls.score(testX, testY) evalúa la precisión del modelo de clasificación de regresión logística entrenado en la tubería pipeline_ls utilizando un conjunto de datos de prueba por lo cual nos da un porcentaje del 0.9661489620422755 de precisión.

Figura 3.52

Puntaje de precisión del modelo de clasificación de regresión logistica

```
pipeline_ls.score(testX,testY)
```

```
0.9661489620422755
```

Nota: la presente imagen se observa la precisión del modelo.

Proporcionamos un informe de la predicción y el rendimiento del modelo de regresión logística entrenado en la tubería pipeline_ls

Para comenzar con el informe de predicción empezamos a imprimir la precisión de los datos de entrenamiento y los datos de prueba. Donde:

pipeline_ls.score(trainX,trainY) = Precisión de los datos del Entrenamiento

pipeline_ls.score(testX,textY) = Precisión de los datos de prueba

Figura 3.53

Informe de la precisión de los datos de entrenamiento y datos de prueba

Entrada.

```
print('Precicion Del Entrenamiento :',pipeline_ls.score(trainX,trainY))
print('Exactitud de la prueba :',pipeline_ls.score(testX,testY))
```

Salida

```
Precicion Del Entrenamiento : 0.9806411995854459
Exactitud de la prueba : 0.9657412059386764
```

Nota: la presente imagen se observa la precisión de los datos de entrada y salida

Luego de imprimir las precisiones de los datos de entrenamiento y prueba procedemos a crear una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

La matriz de confusión es una herramienta importante para evaluar la precisión de un modelo de clasificación, ya que permite visualizar el número de predicciones correctas e incorrectas del modelo en función de las clases reales.

El código utiliza la biblioteca de Pandas para crear un DataFrame que muestra la matriz de confusión. La matriz de confusión se crea utilizando los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación (pipeline_ls.predict(testX)).

La matriz de confusión tiene dos dimensiones: filas y columnas. Las filas representan las clases reales (Actual:Bad y Actual:Good) y las columnas representan las predicciones del modelo (Predicted:Bad y Predicted:Good).

Filas = (Actual:Bad y Actual:Good)

Columnas = (Predicted:Bad y Predicted:Good).

En la diagonal de la matriz se encuentran los valores correctos, es decir, las predicciones que coinciden con las clases reales. Fuera de la diagonal, se encuentran los valores incorrectos, es decir, las predicciones que no coinciden con las clases reales.

En resumen, este código crea una matriz de confusión para evaluar el rendimiento de un modelo de clasificación y permite visualizar de manera clara y sencilla el número de predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

Figura 3.54

Creación de una matriz de confusión para evaluar el rendimiento de un modelo de clasificación de aprendizaje automático.

```
con_mat = pd.DataFrame(confusion_matrix(pipeline_ls.predict(testX), testY),
            columns = ['Predicted:Bad', 'Predicted:Good'],
            index = ['Actual:Bad', 'Actual:Good'])
```

Nota: la presente imagen se observa la creación de una matriz de confusión.

Creación de un informe de clasificación:

Una vez ya creada la matriz procedemos a crear un informe de clasificación utilizando la función **“classification_report”** de la biblioteca Scikit-learn.

Este informe proporciona una evaluación más detallada del rendimiento del modelo de clasificación en términos de varias métricas de evaluación, como la **precisión, el recall, el puntaje F1 y el soporte** para cada clase.

En concreto, el código utiliza los datos de prueba (testX y testY) y las predicciones realizadas por el modelo de clasificación “**(pipeline_ls.predict(testX))”** para generar el informe de clasificación.

El argumento **“target_names”** se utiliza para especificar los nombres de las clases en el informe. En este caso, las clases se denominan 'Bad' y 'Good'.

La función classification_report genera un informe de clasificación que incluye las siguientes métricas:

- Precisión: La precisión mide la proporción de predicciones positivas que son correctas. En otras palabras, la precisión mide la capacidad del modelo de clasificación para identificar correctamente los casos positivos.
- Recall: El recall mide la proporción de casos positivos que se identificaron correctamente. En otras palabras, el recall mide la capacidad del modelo de clasificación para detectar todos los casos positivos.
- F1 Score: El puntaje F1 es una medida de la precisión general del modelo de clasificación. Es la media armónica de la precisión y el recall.
- Soporte (Support): El soporte es el número de observaciones de cada clase en los datos de prueba.

En resumen, este código crea un informe de clasificación que proporciona una evaluación detallada del rendimiento del modelo de clasificación en términos de varias métricas de evaluación, lo que permite una comprensión más completa del rendimiento del modelo.

Figura 3.55

Creación de un informe de clasificación.Creación de un informe de clasificación.

Entrada.

```
print('\nINFORME DE CLASIFICACION\n')
print(classification_report(pipeline_ls.predict(testX), testY,
                            target_names =['Bad','Good']))
```

Salida.

```
INFORME DE CLASIFICACION

              precision    recall  f1-score   support

         Bad       0.91      0.97      0.94     36855
        Good       0.99      0.97      0.98    100482

    accuracy                           0.97    137337
   macro avg       0.95      0.97      0.96    137337
weighted avg       0.97      0.97      0.97    137337
```

Nota: la presente imagen se observa la creación de un informe de clasificación.

Creación de un gráfico de matriz de confusión:

Una vez ya creado el informe de clasificación procedemos a crear un gráfico de matriz de confusión utilizando la biblioteca Seaborn.

El gráfico de matriz de confusión es una herramienta visual que permite analizar el rendimiento de un modelo de clasificación. En el eje x se representan las predicciones del modelo y en el eje y se representan las clases reales.

El código utiliza la matriz de confusión creada previamente (en el código que presentaste anteriormente) para generar el gráfico. El argumento annot=True hace que los valores de la matriz de confusión se muestren dentro de cada celda. El argumento fmt='d' hace que los valores se muestren como números enteros.

El argumento cmap="YlGnBu" establece la paleta de colores utilizada en el gráfico. En este caso, se utiliza la paleta de colores "YlGnBu", que va desde el amarillo claro (representando valores bajos) hasta el azul oscuro (representando valores altos).

La función plt.figure(figsize= (6,4)) establece el tamaño de la figura del gráfico.

En resumen, este código crea un gráfico de matriz de confusión que permite visualizar de manera clara y sencilla el rendimiento del modelo de clasificación en términos de las predicciones correctas e incorrectas realizadas por el modelo en función de las clases reales.

Figura 3.56

Creación de una matriz de confusión.

Entrada.

```
print('\nMATRIZ DE CONFUSION')
plt.figure(figsize= (6,4))
sns.heatmap(con_mat, annot = True,fmt='d',cmap="YlGnBu")
```

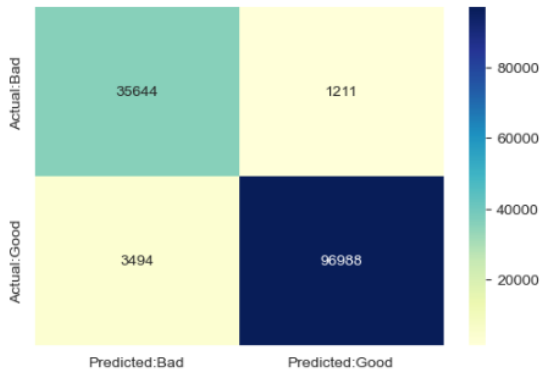
Nota: la presente imagen se observa la creación de una matriz de confusión.

Figura 3.56

Creación de una matriz de confusión.

Salida.

```
MATRIZ DE CONFUSION
<AxesSubplot:>
```



Nota: la presente imagen se observa la creación de una matriz de confusión.

Guardando el modelo entrenado en un archivo pkl:

Una vez entrenado el modelo procedemos a guardar el modelo entrenado con la función pickle.dump(pipeline_ls,open('ataques.pkl','wb')) que serializa y guarda el objeto pipeline_ls en un archivo binario llamado ataques.pkl.

La función pickle.dump() convierte el objeto pipeline_ls a un formato binario que se puede guardar en disco. El primer argumento es el objeto que se va a serializar, mientras que el segundo argumento especifica el nombre y la ubicación del archivo de

salida. El modo 'wb' indica que el archivo se abrirá en modo escritura binaria.

La serialización y guardado del objeto pipeline_ls permite reutilizar el modelo entrenado en el futuro, sin tener que volver a entrenar el modelo desde cero cada vez que se necesite utilizar.

En resumen, pickle.dump(pipeline_ls,open('ataques.pkl','wb')) guarda el objeto pipeline_ls entrenado en un archivo binario llamado ataques.pkl para su posterior reutilización.

Figura 3.57

Guardando el modelo entrenado

```
pickle.dump(pipeline_ls,open('ataques.pkl','wb'))
```

Nota: la presente imagen se observa el guardado del modelo entrenado en un archivo ataques.pkl.

3.2.6. Pruebas

Cargado el objeto pipeline_ls previamente guardado en el archivo binario ataques.pkl utilizando la función pickle.load()

Una vez ya guardado el modelo entrenado procedemos hacer el cargado del mismo modelo entranado con **loaded_model = pickle.load(open('ataques.pkl', 'rb'))** es una línea de código que carga el objeto pipeline_ls previamente guardado en el archivo binario ataques.pkl utilizando la función pickle.load().

El primer argumento de pickle.load() es el nombre del archivo binario que contiene el objeto serializado y el segundo argumento 'rb' indica que el archivo se abrirá en modo de lectura binaria.

result = loaded_model.score(testX, testY) utiliza el modelo cargado loaded_model para hacer predicciones en el conjunto de datos de prueba testX y testY, y calcula la precisión del modelo en el conjunto de datos de prueba utilizando la función score().

Finalmente, print(result) **imprime la precisión del modelo** en el conjunto de datos de prueba en la consola.

En resumen, este código carga el modelo previamente entrenado a partir del

archivo binario ataques.pkl, evalúa su precisión en el conjunto de datos de prueba y finalmente imprime la precisión en la consola.

Figura 3.58

Cargado el objeto pipeline_ls previamente guardado en el archivo binario ataques.pkl

```
loaded_model = pickle.load(open('ataques.pkl', 'rb'))
result = loaded_model.score(testX,testY)
print(result)
```

```
0.9661489620422755
```

Nota: la presente imagen se observa el cargado del objeto pipeline_ls

3.2.7. Detección

Predicción de Ataques Phishing

Una vez hecho el cargado procedemos a utilizar el modelo cargado **"loaded_model"** para hacer predicciones en dos listas de URLs predict_bad y predict_good.

"result = loaded_model.predict(predict_bad)" utiliza el modelo para hacer predicciones en la lista **predict_bad** y guarda los resultados en la variable **result.**

"result2 = loaded_model.predict(predict_good)" utiliza el modelo para hacer predicciones en la lista predict_good y guarda los resultados en la variable result2.

"print(result)" imprime las predicciones del modelo para la lista predict_bad en la consola.

"print("*"*30)" imprime una línea separadora en la consola.

"print(result2)" imprime las predicciones del modelo para la lista predict_good en la consola.

En resumen, este código utiliza el modelo previamente entrenado para hacer predicciones en dos listas de URLs, una que contiene ejemplos de URLs maliciosas **(predict_bad)** y otra que contiene ejemplos de URLs legítimas **(predict_good).**

Luego, imprime las predicciones del modelo para cada lista en la consola.

Figura 3.59

Predicción de sitios Phishing

Entrada

```
#en una fila pondremos 2 sitios uno bueno y el otro malo
predict_bad = ['yeniik.com.tr/wp-admin/js/login.ali','facebook.com']
#en esta fila pondremos 2 sitios que son buenos o sea legitimos y lo guardamos en una variable para visualizar
predict_good = ['youtube.com/','youtube.com/watch?v=qI0TQJI3vdU']
#cargamos el modelo entrenado para que detecte
loaded_model = pickle.load(open('ataques.pkl', 'rb'))
#mostramos el resultado en variables de prediccion
result = loaded_model.predict(predict_bad)
result2 = loaded_model.predict(predict_good)
print("La Prediccion de los sitios Son  ")
print(result)
print("_____________________________________________________")
print("La Prediccion de los sitios Son  ")
print(result2)
#carga el modelo guardado luego con ese modelo empieza a predecir 2 condjuntos con sitios legitimos y falsos
```

Salida.

```
La Prediccion de los sitios Son
['bad' 'good']
_____________________________________________________
La Prediccion de los sitios Son
['good' 'good']
```

Efectivamente esta haciendo la deteccion de sitios phishing y sitios Legitimos

Nota: la presente imagen se observa la predicción de sitios Phishing y Sitios Legitimo.

3.3. METODOLOGIA IWEB

Para el desarrollo de la interfaz de usuario se recurre a la siguiente metodología que está enfocada a sistemas web.

3.3.1. Formulación

En la etapa de la formulación se identifican las metas y objetivos el sistema, estableciendo de este modo la motivación del desarrollo del sistema, su importancia y los usuarios potenciales

1. ¿Cuál es la motivación principal para la aplicación web?

2. ¿Por qué es necesaria esta aplicación?
3. ¿Quién va a utilizar esta aplicación?

Siendo las respuestas:

1. R. Desarrollo de una aplicación web con una buena experiencia de usuario
2. R. Es necesaria ya que brindara un diagnóstico sobre si un sitio web es Falso o Legitimo.
3. R. La aplicación será utilizada un Administrador del sitio y personas usuarias.

Las metas identificadas se dividirán en 2:

Figura 3.60

Metas Aplicadas

Nota: Herramientas utilizadas para el desarrollo del prototipo.

✓ **Metas aplicables.** En esta etapa es que seleccionamos los programas, quenos ayudarán a desarrollar, en este caso el prototipo para que vierta información al usuario de manera que se entendible y de fácil manipulación.

1. **Identificar requisitos de contenido**

Se necesitará el acceso de administrador y acceso del usuario para: Registrar al usuario donde pueda llenar datos y poner la url del sitio en el formulario de detección para ser analizada, posterior a eso habrá un cuestionario donde el usuario podrá responder esto es opcional. Para

Identificar requisitos funcionales.

Serán:

- Formulario de detección
- Registro de Usuario.
- Login de Usuario.
- Cuestionario de preguntas referente al tema de estudio.
- Extensión Web

2. Definir escenarios de interacción para diferentes tipos de usuarios

El prototipo tendrá Un formulario de detección de sitios web maliciosos, Admin y usuario

- ✓ Metas Informativas. En esta etapa se realiza la recopilación, el análisis dey procesamiento de información para que sea visualizado en el prototipo, en este caso se buscó información de cómo realizar el modelo e implementarlo conjuntamente en la interfaz.

3.3.2. Planificación

En cuanto a la planificación para el desarrollo del mismo deberá ser efectuado en la realización de aproximadamente 6 meses, en cuanto a la escalabilidad y seguridad del proyecto si estará efectuado ya que será escalable e incluso a será brindado a modificaciones futuras.

3.3.3. Análisis

En esta etapa ya que tenemos establecidos los requerimientos del usuario se procederá a obtener lo siguiente.

- Formulario de detección
- Registro de Usuario.
- Login de Usuario.
- Cuestionario de preguntas referente al tema de estudio.

- Extensión Web
- Administrador De Usuarios

3.3.4. Ingeniería

3.3.5. Diseño de Contenido

En esta etapa se realiza lo que es el análisis de contenido, más que todo ya que el prototipo va relacionado con la seguridad será en ese ámbito.

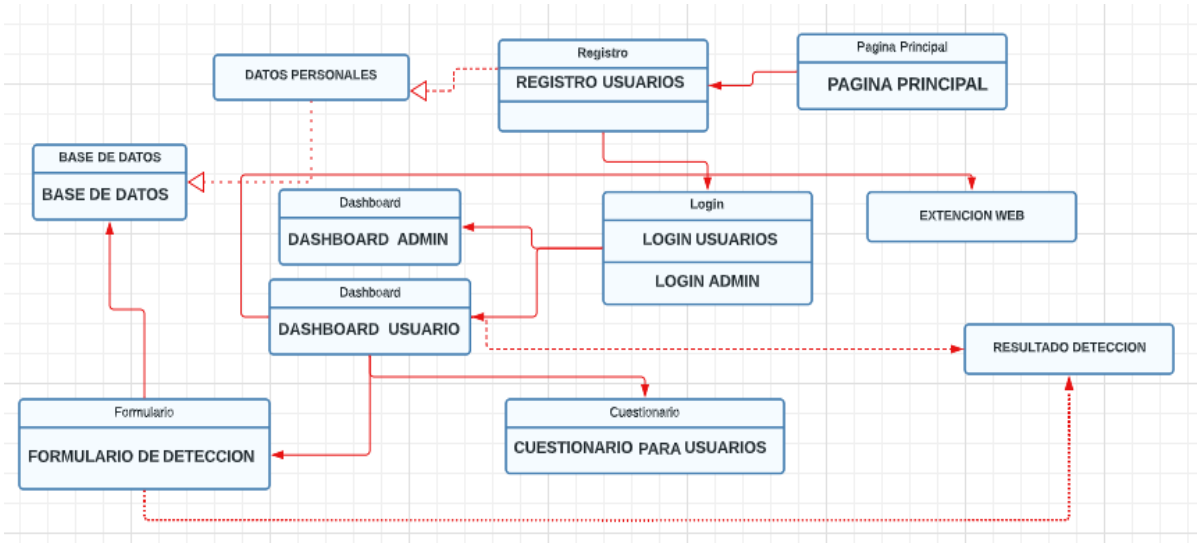
Tendrá colores relacionados con la seguridad e información que brinden información con respecto al diagnóstico de sitios web.

3.3.6. Diseño Arquitectónico

En esta etapa se realiza el diagrama mediante figuras y líneas pararepresentar los diversos componentes del sistema

Figura 3.61

Diseño arquitectónico

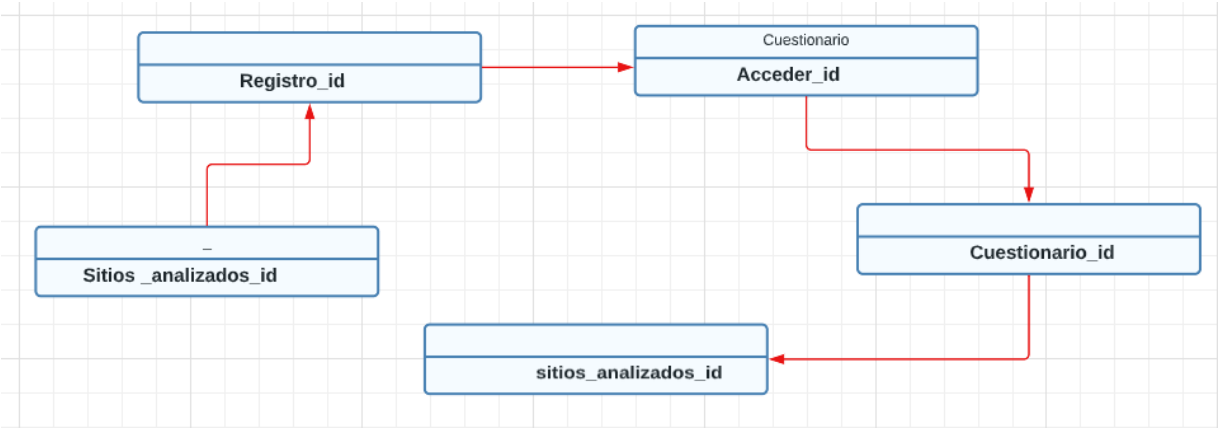


Nota. Se observa como sera el desenlaze de la aplicación

3.3.7. Diseño de Navegación

Figura 3.62

Diseño de Navegacion



Nota. Accesos a paginas por usuarios

3.3.8. Etapa de Pruebas

En esta etapa se realizó y se verifico las pruebas de seguridad ydisponibilidad e integridad de datos

3.3.9. Evaluación del Usuario

En la evaluación del usuario la interfaz es amigable y de fácil manejo e intuitivo.

3.4. DESARROLLO E IMPLEMENTACION DEL MODELO

3.4.1. Desarrollo de Modelo de Detección y Reducción de Ataques Phishing

Una vez creado y entrenado el modelo de detección y reducción de ataques phishing se guardo el modelo en un archivo “ ataques.pkl “ para que pueda ser usado en el formulario de detección. Esto se hace para que no volvamos a entrenar una vez que este en el servidor, esto ayuda a optimizar el tiempo en ser entregado el análisis de detección.

Figura 3.63

Maquetación 1

Nota: *Guardado del modelo para poder usarlo en el formulario de detección*

Figura 3.64

Maquetación 1

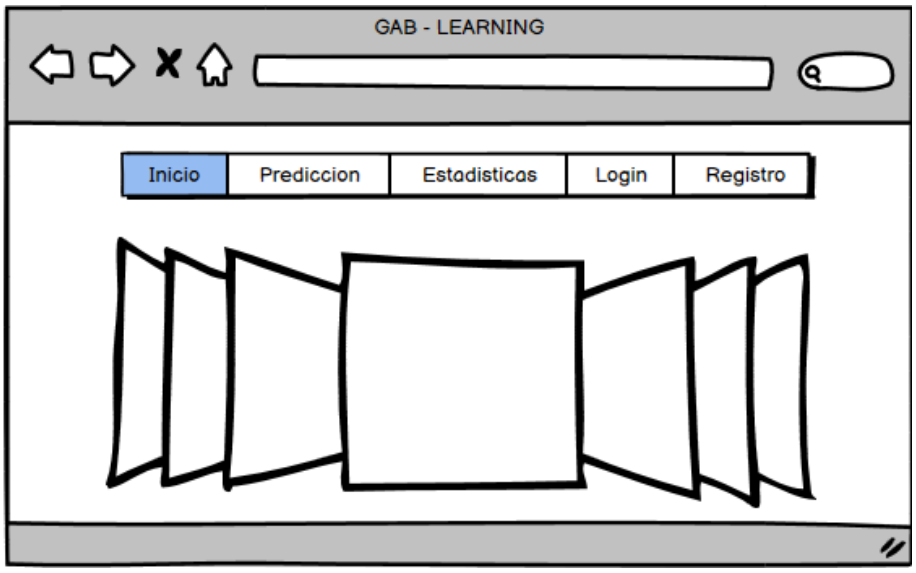
```
# subiendo el archivo picle pkl generado en jupyter
phish_model_ls = pickle.load(open(r'c:\Users\GABRIEL\Pictures\Tesis Gabriel\PROTOTIPO DE TESIS\tesis phising\tesis phising\ataques.pkl', 'rb'))

urlError = {
    "Por favor ingrese el campo de URL"
}

@app.route('/predict',  methods=['POST'])
def predict():
    cur = mysql.connection.cursor()
    cur.execute("SELECT * FROM sitios_analizados")
    roles = cur.fetchall()
    cur.close()

    X_predict = []

    url = request.form.get("EnterYourSite")
    print(url, "000000000000000000000")
    if url:
        X_predict.append(str(url))
        y_Predict = ''.join(phish_model_ls.predict(X_predict))
        print(y_Predict)
        if y_Predict == 'bad':
            result = "Este Es Un Sitio Phishing"
            name = request.form['EnterYourSite']
            cur = mysql.connection.cursor()
            cur.execute("INSERT INTO `sitios_analizados` (`id`, `url`, `descripcion`) VALUES (NULL, %s, 'Phishing')", [name])
            mysql.connection.commit()
            print("sitio agregado")
        else:
            result = "Este No Es Un Sitio Phishing"
            name = request.form['EnterYourSite']
            cur = mysql.connection.cursor()
            cur.execute("INSERT INTO `sitios_analizados` (`id`, `url`, `descripcion`) VALUES (NULL, %s, 'Legitimo')", [name])
            mysql.connection.commit()
            print("sitio agregado")
        return render_template('prediccion.html', prediction_text = result)
```

Nota. Código para el formulario de detección

3.4.2. Maquetación

Figura 3.65

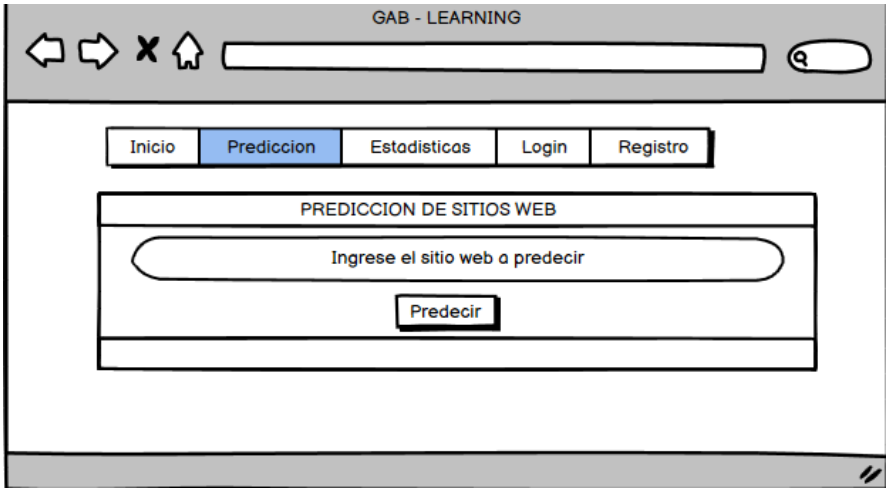
Maquetación 1



Nota. Página principal

Figura 3.66

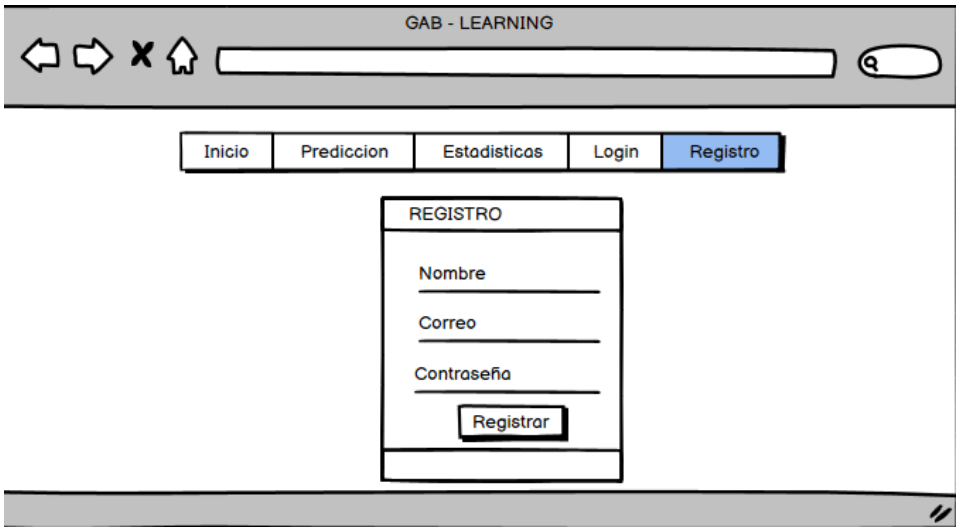
Maquetación 2



Nota: Predicción De Sitios Web

Figura 3.67

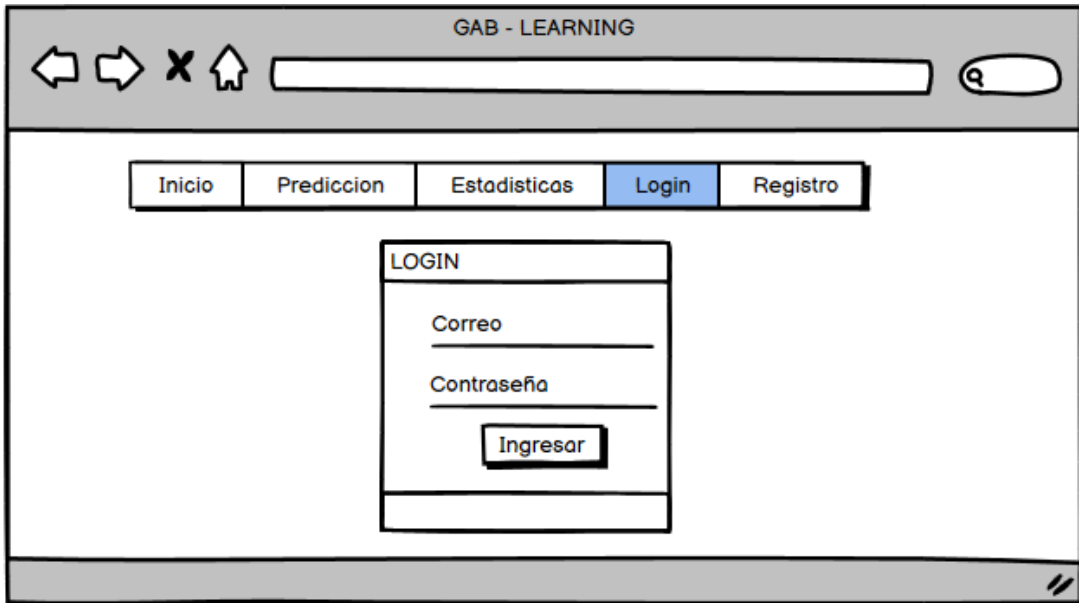
Maquetación 3



Nota: Registro de Usuario

Figura 3.68

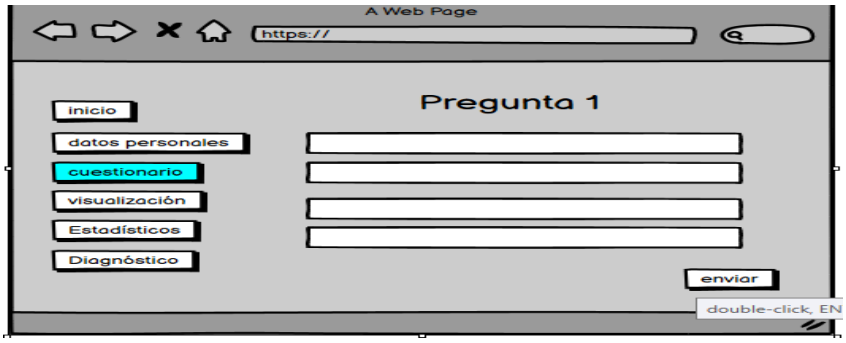
Maquetación 4



Nota: Acceso de usuario

Figura 3.69

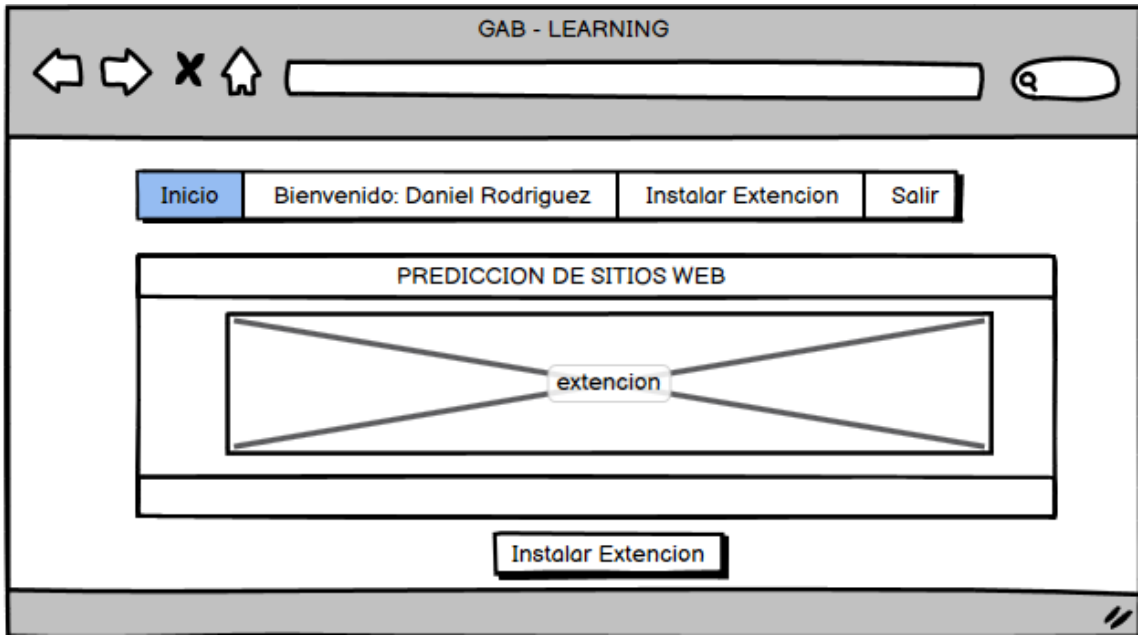
Maquetación 5



Nota: Cuestionario

Figura 3.70

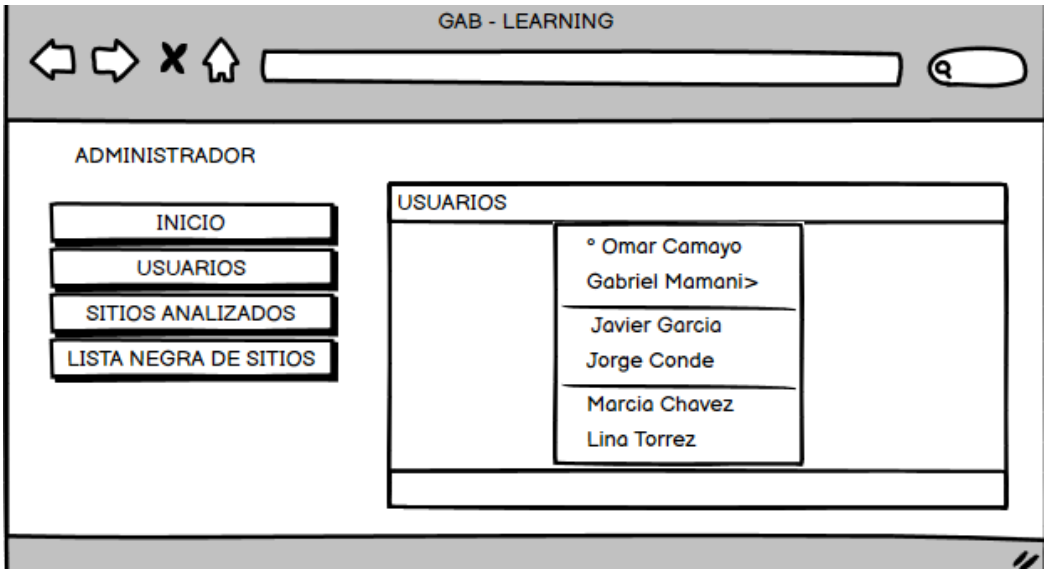
Maquetación 6



Nota: Instalar Extencion

Figura 3.71

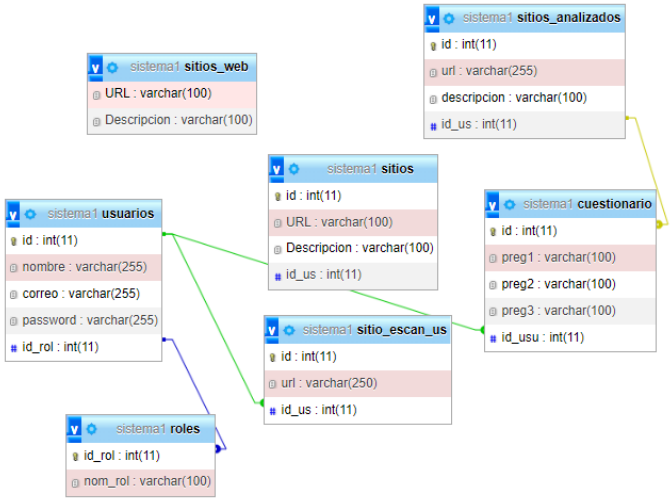
Maquetación 7



Nota: Administrador De Usuarios

Figura 3.72

Base de Datos



Nota: Tablas que conforman la Base de datos

Figura 3.73

Código para creación de Base de datos y Tablas

```sql
CREATE TABLE `cuestionario` (
  `id` int(11) NOT NULL,
  `preg1` varchar(100) NOT NULL,
  `preg2` varchar(100) NOT NULL,
  `preg3` varchar(100) NOT NULL,
  `id_usu` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

CREATE TABLE `roles` (
  `id_rol` int(11) NOT NULL,
  `nom_rol` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

INSERT INTO `roles` (`id_rol`, `nom_rol`) VALUES
(1, 'Admin'),
(2, 'Usuario');

CREATE TABLE `sitios` (
  `id` int(11) NOT NULL,
  `URL` varchar(100) NOT NULL,
  `Descripcion` varchar(100) NOT NULL,
  `id_us` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;


INSERT INTO `sitios` (`id`, `URL`, `Descripcion`, `id_us`) VALUES
```

Nota: Código para la creación de Base De datos y tablas relacionadas.

3.4.3. Desarrollo *del Módulo de Formulario de Detección*

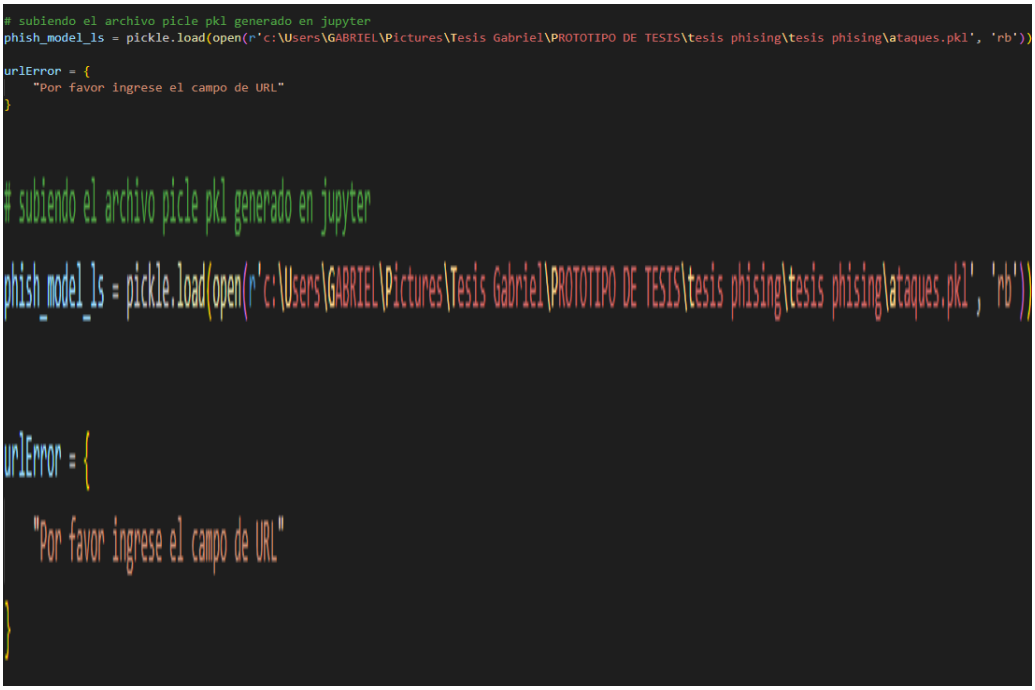
Figura 3.74

Modulo de Formulario de Deteccion

Nota: Formulario de Detección

Figura 3.75

Código de Modulo de Formulario de Deteccion



Nota. Código para el formulario de deteccion

3.4.4. Desarrollo *del Módulo de Registro De Usuarios*

Figura 3.76

Módulo de Registro de Usuarios

Nota: Registro de Usuarios

Figura 3.77

Código de Modulo de Registro de Usuarios

```python
@app.route('/registro')
def reg():
    return render_template('registro.html')

@app.route('/registro-crear', methods = ["GET", "POST"])
def registro():

    cur = mysql.connection.cursor()
    cur.execute("SELECT * FROM roles")
    roles = cur.fetchall()
    cur.close()


    if request.method == 'GET':
        return render_template("registro.html", tipo = roles )

    else:
        name = request.form['txtNombre']
        email = request.form['txtCorreo']
        password = request.form['txtPassword']

        cur = mysql.connection.cursor()
        cur.execute("INSERT INTO usuarios (nombre, correo, password, id_rol) VALUES (%s,%s,%s,'2')", (name, email, password))
        mysql.connection.commit()

        return render_template('login.html',mensaje2="Registro Exitoso")
```

Nota. Código para el registro de usuarios

3.4.5. Desarrollo *del Módulo de Validación de Usuarios*

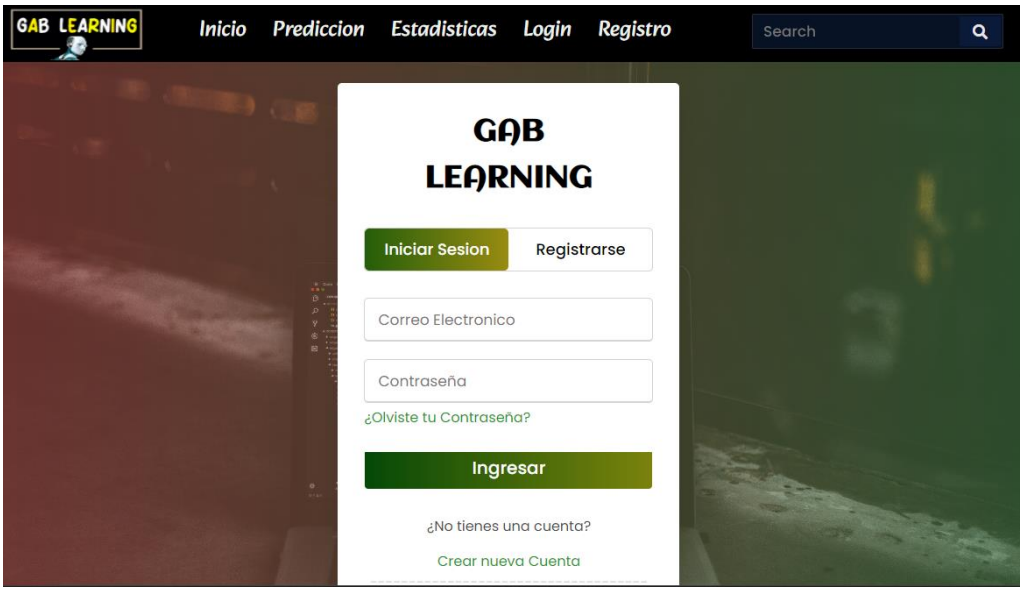
Figura 3.78

Módulo de validación de Usuario

Nota: Página principal

Figura 3.79

Validacion de Usuario



Nota. Acceso del usuario

Figura 3.80

Código de Validación de Usuario

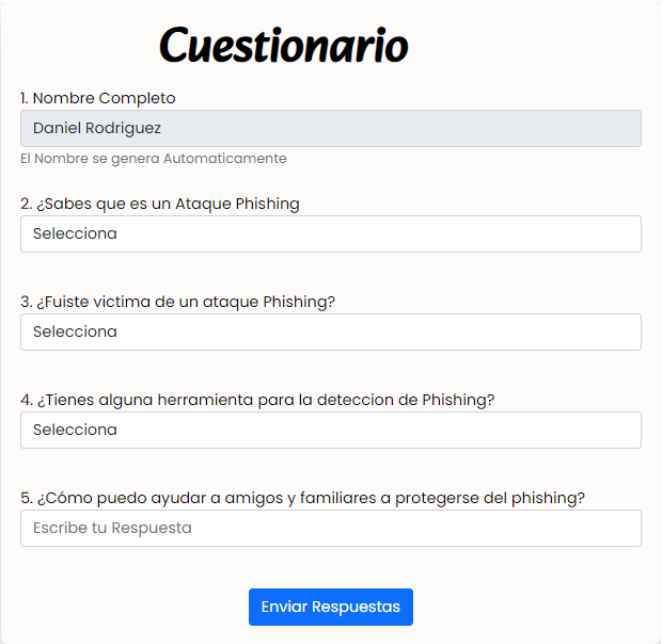
```
#---------- LOGIN ------------------------
@app.route('/acceso-login', methods= ["GET", "POST"])
def login():
    actividad = """SELECT usuarios.nombre, sitios_analizados.descripcion, sitios_analizados.url FROM sitios_analizados JOIN usuarios ON sitios
    if request.method == 'POST' and 'txtCorreo' in request.form and 'txtPassword' in request.form:

        _correo = request.form['txtCorreo']
        _password = request.form['txtPassword']

        cur = mysql.connection.cursor()
        cur.execute('SELECT * FROM usuarios WHERE correo = %s AND password = %s', (_correo, _password,))
        account = cur.fetchone()

        if account:
            session['logueado'] = True
            session['nombre'] = account['nombre']
            session['correo'] = account['correo']
            session['password'] = account['password']
            session['id_rol'] = account['id_rol']


            if session['id_rol'] == 1:
                    return render_template("admin.html")
            elif session['id_rol'] == 2:
                    return render_template("usuario.html")
            print(account[0])
        else:

            return render_template('login.html',mensaje="Usuario O Contraseña Incorrectas")
```

Nota. Código del módulo login

3.4.6. Desarrollo *del Módulo de Cuestionario*

Figura 3.81

Módulo de Cuestionario



Nota: Módulo cuesto análisis

Figura 3.82

Código de Módulo de Cuestionario

```python
@app.route('/crear-cuestionario', methods = ["GET", "POST"])
def preguntas():
    cur = mysql.connection.cursor()
    cur.execute("SELECT * FROM cuestionario")
    roles = cur.fetchall()
    cur.close()
    user_id=session.get('id')

    if request.method == 'GET':
        return render_template("cuestionario.html", tipo = roles )

    else:
        name = request.form['txtNombre']
        preg1 = request.form['txtPreg1']
        preg2 = request.form['txtPreg2']
        preg3 = request.form['txtPreg3']
        preg4 = request.form['txtPreg4']


        cur = mysql.connection.cursor()
        cur.execute("INSERT INTO cuestionario (nombre, preg1, preg2, preg3, preg4,id_usu) VALUES (%s,%s,%s,%s,%s,%s)", (name, preg1, preg2, preg
        mysql.connection.commit()

        return render_template('cuestionario.html',mensaje2="Registro Exitoso")
```

Nota:. Código del cuestionario

3.4.7. Desarrollo *del Módulo de Extensión Web Para Detección De Sitios Para Usuarios*

Figura 3.83

Módulo de Extensión De Sitios

Nota: Página Principal

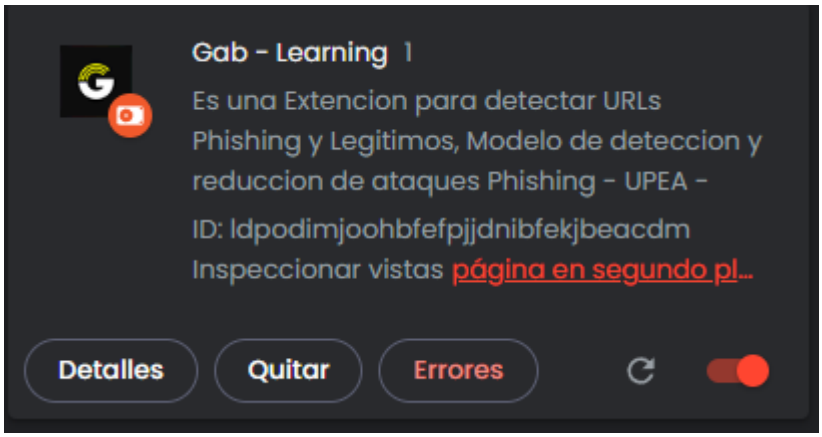
Figura 3.84

Módulo de Extensión De Sitios legitimo 2

Nota: Extensión De Sitios

Figura 3.85

Módulo de Extensión De Sitios 2



Nota: Extensión De Sitios

Figura 3.86

Módulo de Extensión De Sitios falso 3

Nota: Extensión De Sitios

Figura 3.87

Módulo de Extensión De Sitios falso 3

Nota. Extensión De Sitios

Figura 3.88

Código de Módulo de Detección de Sitios

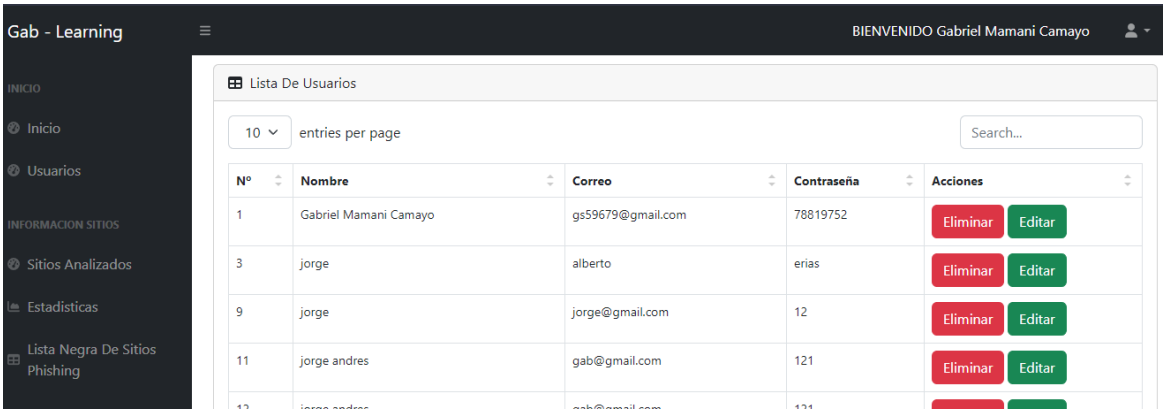
```
function verificarModelo(url, resultado) {
  fetch('https://api.myfilechecker.com/extract_features?url=' + url)
    .then(response => response.json())
    .then(data => {
      fetch('https://api.mymodelchecker.com/check', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({
          features: data,
          model_file: 'ataques.pkl'
        })
      })
      .then(response => response.json())
      .then(data => {
        if (data.isLegit) {
          resultado.innerText = 'bad', 'Peligo ESTE ES UN SITIO PHIHSING';
        } else {
          resultado.innerText = 'good', 'SITIO LEGITIMO';
        }
      })
      .catch(error => {
        resultado.innerText = 'Ocurrió un error al verificar el archivo';
      });
    })
    .catch(error => {
      resultado.innerText = 'Ocurrió un error al verificar el archivo';
    });
}
```

Nota: Código de Extensión De Sitios

3.4.8. Desarrollo *del Módulo de Administrador de Usuarios y sitios Usuarios*

Figura 3.89

Módulo de Administrador De Usuarios



Nota: Administrador de Usuarios

Figura 3.90

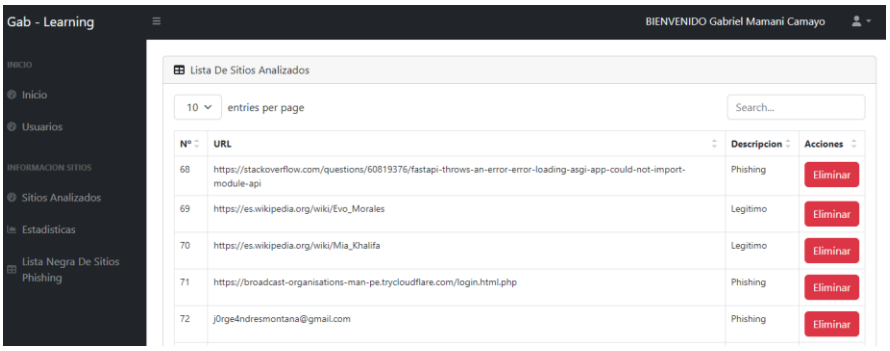
Código De Modulo De Administrador de usuarios

```
@app.route('/us_admin', methods = ["GET", "POST"])
def us_admin():
    cur = mysql.connection.cursor()

    querySQL = ("SELECT * FROM usuarios")
    cur.execute(querySQL)
    dato = cur.fetchall()
    cur.close()
    return render_template('usuarios_admin.html', datoEmpleados = dato )
```

Nota. Administrador de Usuarios

Figura 3.91

Administrador de Sitios



Nota. Administrador de Usuarios

Figura 3.92

Código de Administrador de sitios

```python
@app.route('/sit_an', methods = ["GET", "POST"])
def sit_an():
    cur = mysql.connection.cursor()

    querySQL = ("SELECT * FROM sitios_analizados")
    cur.execute(querySQL)
    dat_ur = cur.fetchall()
    cur.close()
    return render_template('admin_sit_an.html', datos_url = dat_ur )
```

Nota. Administrador de Sitios

Figura 3.93

Código de Administrador de sitios

```python
@app.route('/predict',  methods=['POST'])
def predict():
    cur = mysql.connection.cursor()
    cur.execute("SELECT * FROM sitios_analizados")
    roles = cur.fetchall()
    cur.close()

    X_predict = []

    url = request.form.get("EnterYourSite")
    print(url, "000000000000000000000")
    if url:
        X_predict.append(str(url))
        y_Predict = ''.join(phish_model_ls.predict(X_predict))
        print(y_Predict)
        if y_Predict == 'bad':
            result = "Este Es Un Sitio Phishing"
            name = request.form['EnterYourSite']
            cur = mysql.connection.cursor()
            cur.execute("INSERT INTO `sitios_analizados` (`id`, `url`, `descripcion`) VALUES (NULL, %s, 'Phishing')",
            mysql.connection.commit()
            print("sitio agregado")
        else:
            result = "Este No Es Un Sitio Phishing"
            name = request.form['EnterYourSite']
            cur = mysql.connection.cursor()
            cur.execute("INSERT INTO `sitios_analizados` (`id`, `url`, `descripcion`) VALUES (NULL, %s, 'Legitimo')",
            mysql.connection.commit()
            print("sitio agregado")
        return render_template('prediccion.html', prediction_text = result)
```

Nota. Administrador de Sitios

3.5.1 Implementación y Despliegue

3.5.2.1 Requerimientos de Hardware

Descripción de requerimiento de equipo:

- ✓ Procesador: AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx 2.30 GHz
- ✓ RAM 8 GB
- ✓ Tipo de sistema: Sistema operativo de 64 bits, procesador x64
- ✓ Disco duro 200 GB

3.5.2.2 Requerimientos del Software

Descripción de requerimientos de programas software:

- ✓ absl-py==1.2.0
- ✓ asgiref==3.5.2
- ✓ attrs==22.1.0
- ✓ cycler==0.11.0
- ✓ Flask==2.2.3
- ✓ Flask_MySQL==1.5.2
- ✓ Flask_MySQLdb==1.0.1
- ✓ fonttools==4.37.1
- ✓ imutils==0.5.4
- ✓ kiwisolver==1.4.4
- ✓ matplotlib==3.5.3

- ✓ mediapipe==0.8.10.1
- ✓ numpy==1.23.2
- ✓ opencv-contrib-python==4.6.0.66
- ✓ Java Script
- ✓ packaging==21.3
- ✓ Pillow==9.2.0
- ✓ protobuf==3.20.1
- ✓ Jupyter==1.0.0
- ✓ Jupyter_server==2.5.0
- ✓ scikit-learn==0.24.1
- ✓ mysql==0.0.3
- ✓ html5
- ✓ PyMySQL==1.0.2
- ✓ pyparsing==3.0.9
- ✓ python-dateutil==2.8.2
- ✓ pytz==2022.2.1

- ✓ six==1.16.0
- ✓ sqlparse==0.4.2
- ✓ boostrap 5

3.1.1. Despliegue

Tener instalado un servidor puede ser apache, también se debe tener unabase de datos MariaDB para la conexión del prototipo con la base de datos y de esta manera pueda levantarse el servidor.

Instalar por la terminal estas librerías y paquetes:Se debe acceder a la carpeta: Dirección del archivo\MODELO_PHISHING\

Se debe ejecutar el comando **pip install -r .\requirements.txt**, con este comandose instalarán automáticamente todas las librerías y paquetes que necesita el software.

Después se debe crear la base de datos con el nombre de sistema1 e importar el archivo sql

```
    DATABASES = {
'default': {
    'NAME': 'nombre de la base de datos',
    'USER'.'Usuariodelabasededatos'
    'PASSWORD':'contraseñadelabaseded
    atos'
    ','HOST':'localhost',
    'PORT
    ':'3306
    ',
    "OPTI
    ONS":
```

```
        {
            'init_command':                                    "SET
    sql_mode='STRICT_TRANS_TABLES',
    innodb_strict_mode=1",
            'charset':
            'utf8mb4',
            "autocommi
            t": True,
        },
    }
}
```

Posterior a esto se levanta el servidor desde la terminal con el comando. python manage.py runserver

Figura 3.94

Despliegue

Nota: Despliegue del prototipo en otro equipo.

CAPÍTULO IV

PRUEBAS Y RESULTADOS

4.1. INTRODUCCION

El objetivo de la realización de esta etapa es poner a prueba el prototipo, para que se pueda verificar si efectivamente el prototipo cumple con los objetivos planteados en primera instancia y genera los resultados esperados.

4.2. METRICA DE CALIDAD ISO/IEC 9126

Para poder tener un producto de calidad se deberá realizar el proceso de medición en cuanto a la calidad del software, para esto se utilizar la métrica de calidad ISO/IEC 9126 la cual tiene una serie de etapas:

4.2.1. *Funcionalidad*

Si hablamos de funcionalidad seria aquellas que cumplen con las necesidades implícitas o explicitas a continuación se muestra la ponderación de las funcionalidades.

Tabla 4.1

Ponderación de funcionalidad

Característica	Ponderación
Adecuación	90%
Exactitud	90%
Conformidad	90%
Cumplimiento funcional	90%
Promedio	90%

Nota. Se describe la ponderación funcional

Por lo tanto, se deduce que el prototipo tiene una funcionalidad del 90%

4.2.2. Confiabilidad

Las características o atributos relacionados con la capacidad del software dado para su uso bajo condiciones que están establecidas en un determinado tiempo.

Dónde: LCD=cantidad de líneas de código.

Fórmula: Confiabilidad= 1-(5/LCD) *100

Confiabilidad = 1-(5 /4560) x100 Confiabilidad = 99%.

El sistema tiene una confiabilidad del 99%.

4.2.3. Usabilidad

El esfuerzo necesario para su uso, y en la valoración individual de este determinado uso, está relacionado con los atributos y su conjunto

Tabla 4.2

Ponderación de métricas internas usabilidad

Característica	Métrica interna	Puntaje
Interfaz de usuario amigable	I1: Interfaz de datos amigable I2: Interfaz de gráficos amigable	90 88
Comprensión	C1: Comprensión de datos C2: Comprensión de gráficos	95 95
Operatividad	O1 Correcta operacionalidad de visualización de datos O2: Correcta operacionalidad de los gráficos	90 95
Atractividad	A1 Atractividad de la interfaz A2: Atractividad de los gráficos A3: Atractividad de la visualización de los datos.	90 90 86

Nota. *Se describe la ponderación de métricas internas de usabilidad*

Tabla 4.3 *Totales de métricas internas de usabilidad*

	Métrica	Puntaje promedio(xi)
1	Interfaz de usuario amigable (I)	89
2	Comprensión (C)	95
3	Operatividad (O)	92.5
4	Atractividad (A)	88.6
4	**total**	365.1

Nota. Promedios totales

Con los datos obtenidos en la tabla 6 se aplica en la fórmula:

$$Usabilidad = \sum(xi/n)$$

$$Usabilidad = 365.1/4$$

$$Usabilidad = 91\%$$

4.2.4. Eficiencia

Si deseamos obtener el cálculo de la eficiencia se tiene que ponderar características esenciales que el sistema desempeña.

Tabla 4.4

Evaluación de desempeño

n	Características de desempeño	Ponderación (xi)
1	Rapidez de inicio	5
2	Rapidez de procesamiento de imagen	4
3	Proceso rápido de visualización de imagen	4
4	Fluidez	4
5	Disponibilidad	5
= 5	**total**	**∑xi=** 22

Nota. *Evaluación del desempeño del sistema*

En base a los datos de la anterior tabla se podría llegar a tener una idea dela

eficiencia, utilizando la siguiente fórmula.

$$Eficiencia = \sum(xi/n) * 100/n\ (\%)$$

$$Eficiencia = 22/5 * 100/5$$

$$Eficiencia = 88\%$$

4.2.5. Mantenibilidad

Se refiere al conjunto de atributos que nos da la opción de poder corregir aumentar o modificar errores del software, dicho resultado se obtiene mediante la siguiente fórmula.

$$Mantenibilidad = (Mt - (Fc + Fa + Fd))/Mt$$

Donde:

Mt= número de módulos en la versión actual

Fc= número de módulos en la versión actual que han cambiado.Fa= número de módulos en la versión actual añadido

Fd= número de módulos en la versión anterior que se ha borrado

Entonces:

Mt=1; Fc=1; Fa=0; Fd=0

$$Mantenibilidad = (3 - (0 + 0 + 0))/3$$

$$Mantenibilidad = 1$$

$$Mantenibilidad = 100\%$$

4.2.6. Portabilidad

Se refiere cuando el software tiene la capacidad de ser trasladado de un entorno a otro. Esta etapa se calcula de esta manera:

$$Portabilidad = 1 - (ndpm/ndim)$$

Donde:

ndpm = número de días para portar el modelo

días. ndim = número de días para implementar el modelo

$$Portabilidad = 1 - (1/6)$$
$$Portabilidad = 0.83 * 100$$
$$Portabilidad = 83\%$$

4.2.7. Resultados

Una vez que ya realizamos el cálculo independiente de los anteriores factores ahora podemos realizar el cálculo global.

Tabla 4.5

Análisis global de calidad

N°	Características	Resultado
1	Funcionalidad	90%
2	Confiabilidad	99%
3	Usabilidad	91%
4	Eficiencia	88%
5	Mantenibilidad	100%
6	Portabilidad	83%
	Evaluación de Calidad Global	92%

Nota. Se describe el análisis global de calidad

Según Pressman dice que el resultado de evaluación de una métrica o modelo si supera el promedio en un 65% es aceptado. Por lo que el 92% encontrado en la medición es aceptable para el modelo propuesto.

4.3. ESTIMACION DE COSTO COCOMO

Para estimar el costo del software será realizado con ayuda de COCOMO II,

4.3.1. Punto de función

La estimación por puntos de función está en la medida de la funcionalidad del sistema de información y un conjunto de factores individuales del sistema, los puntos de función son estimadores que puede ser de utilidad en las etapas iniciales del modelo. La medida de puntos de función está cuantificada en base a diferentes funcionalidades. La tabla siguiente describe los componentes relacionados con su complejidad asignada a cada uno de los factores que se deben considerar para la estimación del modelo.

Tabla 4.6

Puntos de función no ajustado

Tipos de parámetros	**Cantidad**	**Factor de ponderación**	**Total**
Entrada	3	5	15
Salida	3	6	18
Archivos	3	5	15
Consultas	3	6	18
Interfaces	3	5	15
	Total, de función no ajustado		81

Nota. Ponderación de puntos de función siendo Pf=81.

Según el estimado de interfaces de la tabla anterior, se procede a clasificarlos según su complejidad y luego multiplicar por los pesos establecidos de acuerdo a COCOMO II, para estimar los puntos de función ajustados.

En la siguiente tabla se 14 factores de ajuste donde se pondera con un puntaje que se encuentra entre 0 y 5.

Tabla 4.7

Ponderación de ajuste de complejidad

Nro. de Factor	Factor	Valor 0-5
1	Mecanismo de recuperación	3
2	Comunicación de datos	5
3	Rendimiento	5
4	Configuración usada rigurosamente	5
5	Entrada de datos en línea	1
6	Factibilidad operativa	4
7	Actualización en línea	1
8	Interfaces complejas	3
9	Proceso interno complejo	5
10	Reusabilidad de código	5
11	Fácil instalación	5
12	Instalaciones múltiples	1
13	Facilidad de cambios	5
14	Funciones de proceso distribuido	1
	∑Fi	49

Nota. En esta tabla se puede observar los puntajes de acuerdo a los factores del software.

Con el promedio encontrado, se reemplaza los datos en la fórmula de puntode función asignado.

Donde:

PFA=punto de función asignado.

∑Fi= ponderación de ajuste de complejidad.Pf= Puntos de función

no ajustado

$$PFA = Pf * (0.65 + 0.01 * \sum \text{Fi})$$

$$PFA = 81 * (0.65 + 0.01 * 49)$$

$$PFA = 92.34$$

4.3.2. *A*plicación de COCOMO II

Para el desarrollo del sistema se deben considerar diversas plataformas que soporten el trabajo, así mismo el lenguaje.

Para poder calcular las líneas de código, utilizamos el valor del punto de función ajustado, de igual forma utilizaremos el valor de Factor en línea de código del lenguaje de programación utilizada para el desarrollo.

Tabla 4.8

Factor LCD/PF de lenguaje de programación

Lenguaje	Factor LCD/PF
Html	500
Css	300
Javascript	280
Python	1100

Nota. Cantidad de líneas de código

Reemplazamos los datos en la fórmula para calcular las líneas de código:

$$LDC = PFA * Factor\ LDC/PF$$

$$LDC = 92.34 * 2180/81$$

$$LDC = 2485$$

$$KLDC = 2485/1000$$

$$KLDC = 2.48$$

Tomando en cuenta lo visto en el capítulo dos, el proyecto se enmarca en el modelo básico. De acuerdo a la cantidad de líneas de código del modelo pertenece a

un modo orgánico por lo que los valores para a y b serán 2.40 y 1.05

Respectivamente de COCOMO II brinda la siguiente forma:

- **Estimación de esfuerzo de desarrollo**

$$E = a^{*}(KLCD)^{b}$$

El esfuerzo se estima

$$E = 2.40 * 2.48^{1.05}$$

$$E = 6{,}22(personas/mes)$$

Es decir que se requiere el esfuerzo de personas trabajando en eldesarrollo del sistema.

- **Estimación del tiempo de desarrollo**

Para el cálculo del tiempo se utiliza nuevamente la expresión de COCOMOII para la determinación del tiempo estimado del proyecto.

$$T = C * (E)^{d}$$

Donde c y d son constantes que de acuerdo al modo orgánico establecidopor COCOMO estos valores son 2.50 y 0.38 respectivamente.

Reemplazando estos valores en la formula se tiene el cálculo del tiempo expresado en meses.

$$T = 2.5 * (6{,}22)^{0.38}$$

$$T = 5(meses)$$

El tiempo estimado de trabajo es de aproximadamente es 6 meses segúnlos datos obtenidos de la ecuación.

- **Estimación de la productividad**

La productividad que se debe esperar de cada programador está dada porla siguiente expresión:

Donde:

PR= Estimación de productividad.

LCD=Cantidad e líneas de

código.E= Esfuerzo.

$$PR = LDC/E$$

$$PR = 2485/6$$

$$PR = 414.16(LDC/persona - mes)$$

Se espera que un programador genere 414.16 líneas de código al mes

- **Cálculo de personal promedio**

Para el cálculo del personal promedio se aplica la fórmula.

Donde:

P=cantidad de personas

E=cantidad de esfuerzo

T=cantidad de tiempo

$$P = E/T$$

$$P = 6/5$$

$$P = 1.2(personas)$$

Estos resultados indican que se requiere una persona trabajando por unos cinco meses, desarrollando 414 líneas

Para el cálculo total del software se considera el sueldo aproximado de un ingeniero de sistemas junior de 3200 bs mensuales.

Donde:

CT: Cálculo total del software.

$$CT = 3200 * (P * T)$$

$$CT = 3200 * (1 * 5)$$

$$CT = 16000\ Bs.$$

Por lo que se concluye que el costo estimado del prototipo es de 14790 Bs,un tiempo de 5 meses y 1 personas trabajando en el mismo.

4.3.3. Costo de desarrollo del sistema

Para esta etapa se considera diversos factores, principalmente los relacionados con el desarrollo del prototipo.

Tabla 4.9

Costo de elaboración del prototipo

Detalle	Importe
Análisis y diseño del prototipo	200 Bs.
Material de Escritorio	100 Bs
Conexión a internet	300 Bs.
Total	600 Bs.

Nota. Costo de elaboración

4.3.4. Costo total

Para el cálculo del costo total se tomó en cuenta el costo del software calculado anteriormente y el costo de elaboración.

Tabla 4.10

Costo total del prototipo

Detalle	Importe
Costo del software	10000Bs.
Costo de elaboración	400 Bs.
Total	10400Bs.

Nota: Totalidad del costo.

Considerando la tabla anterior se concluye que el costo total del software es de 16600 Bs.

4.4. SEGURIDAD

En cuanto a la seguridad se utilizará el conjunto de estándares internacionales ISO27000 el cual cuenta con buenas prácticas para poder establecer, implementar y mantener una mejora de sistemas de gestión de la seguridad de la información.

Teniendo 2 pilares fundamentales los cuales son 27001 el cual se basa enla gestión de la seguridad de la información, para identificar los riesgos, por otra parte, la 27002 es una guía de buenas prácticas describiendo objetivos de controly gestión que debe tener la organización.

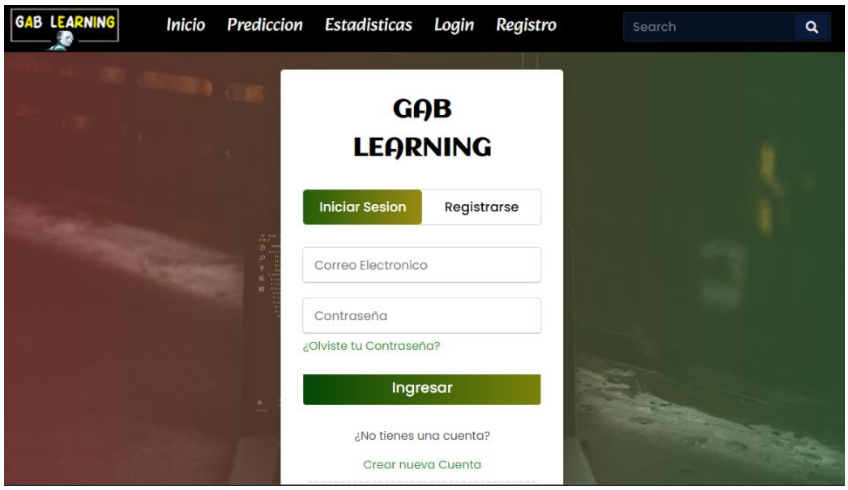
Utilizaremos las siguientes normas de estándares de seguridad

- ISO27001 el cual nos permitirá asegurar la integridad y confidencialidad dela información de los datos.

  En esta parte del software se puede observar la seguridad que brinda en cuanto al acceso de los datos del usuario.

Figura 4.1

Acceso de datos del usuario



Nota. Acceso de datos del usuario

- ISO27004 El cual nos describe recomendación para la medición de los sistemas de gestión de información donde nos especifica cómo es la configuración de métricas que medir el cómo y la forma de conseguir objetivos. El prototipo cuenta con esta norma ya que para la gestión organización del mismo está realizado de manera ordenada, para que de esta forma se vayan sacando los resultados esperados
- ISO27005 el cual nos brinda información sobre recomendaciones y directrices generales para la gestión de riesgo en sistemas de gestión de seguridad de la información, también es compatible con los conceptos del ISO27001.

4.5. PRUEBAS DE SOFTWARE

Para las pruebas del software se utilizarán las siguientes:

- **Caja Negra**

Como bien se decidió utilizar la prueba de caja negra, la cual se enfoca en la evaluación de la funcionalidad del software, entonces para ello utilizaremos el método que se basa en las historias de usuario en el prototipo.

Tabla 4.11

Pruebas funcional Ingreso al prototipo

Suite de Pruebas del Prototipo								
No	**Caso de prueba**	**Priori dad**	**Precondiciones**	**Datos de entrada**	**Pasos**	**Resultado esperado**	**Resultado obtenido**	**Estado prueba**
Área Funcional: Ingreso al Prototipo								
1	Ingreso al prototipo	Alta	El servidor debe estar levantado	Tener la dirección url	1 abrir la página 2 Vista de los menús de la página	El usuario tuvo acceso satisfactoria mente al inicio de la página	abajo	Ok
Resultado Obtenido								

Tabla 4.12

Prueba Funcional Registro de Usuario

Suite de Pruebas del Prototipo								
No	**Caso de prueba**	**Prioridad**	**Precondiciones**	**Datos de entrada**	**Pasos**	**Resultado esperado**	**Resultado obtenido**	**Estado prueba**
Área Funcional: Registro de Usuario								
22	Registro de Usuario	Alta	Llenar datos correctamente	Tener la dirección url Nombre Correo contraseña	1 abrir la página 2 Vista de los menus de la página 3 clic en botom registrar se	El usuario se registró satisfactoriamente en la base de datos	Abajo	ok
Resultado Obtenido								

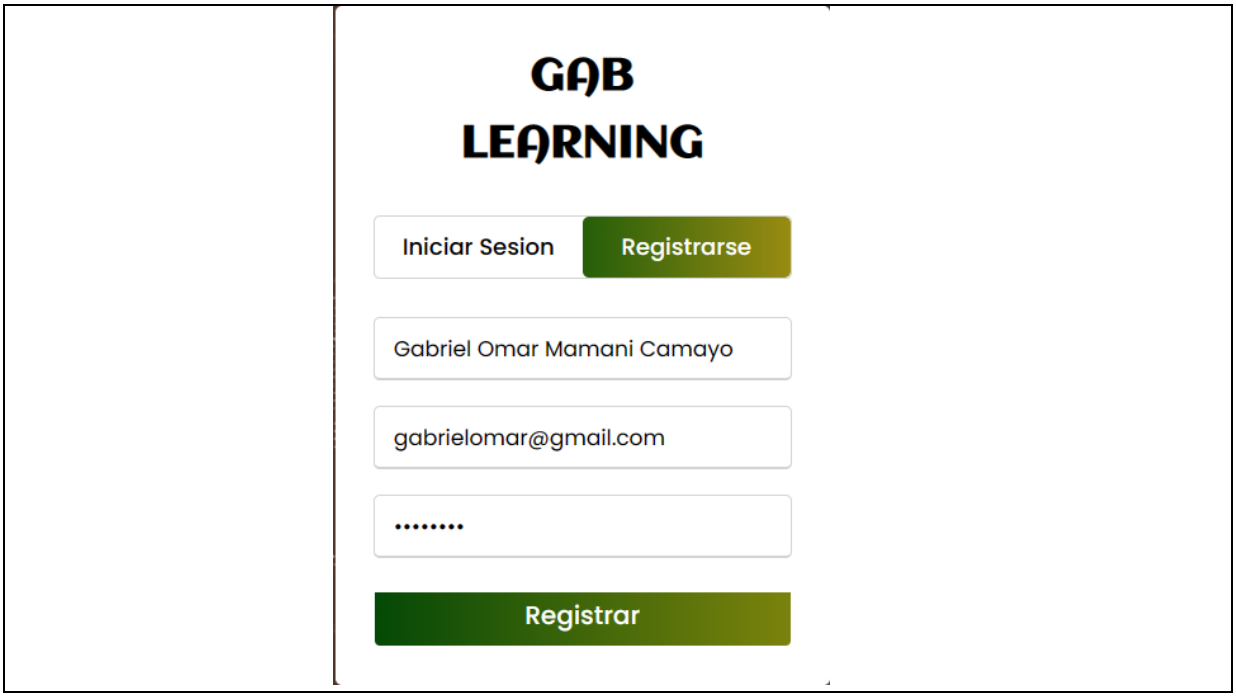


Tabla 4.13

Prueba Funcional Acceso al Prototipo

Suite de Pruebas del Prototipo								
No	**Caso de prueba**	**Prioridad**	**Precondiciones**	**Datos de entrada**	**Pasos**	**Resultado esperado**	**Resultado obtenido**	**Estado prueba**
Área Funcional: Acceder al Prototipo								
3	Acceder al prototipo	Alta	Llenar datos correctamente	Tener la dirección url Registrar usuario Nombre usuario, contraseña	1 Abrir la página 2 Vista de losmenus de la página 3 clic en botom acceder	El usuario inicio sesión correctamente Direccionamiento a la interfaz de usuario	Abajo	Ok
Resultado Obtenido								

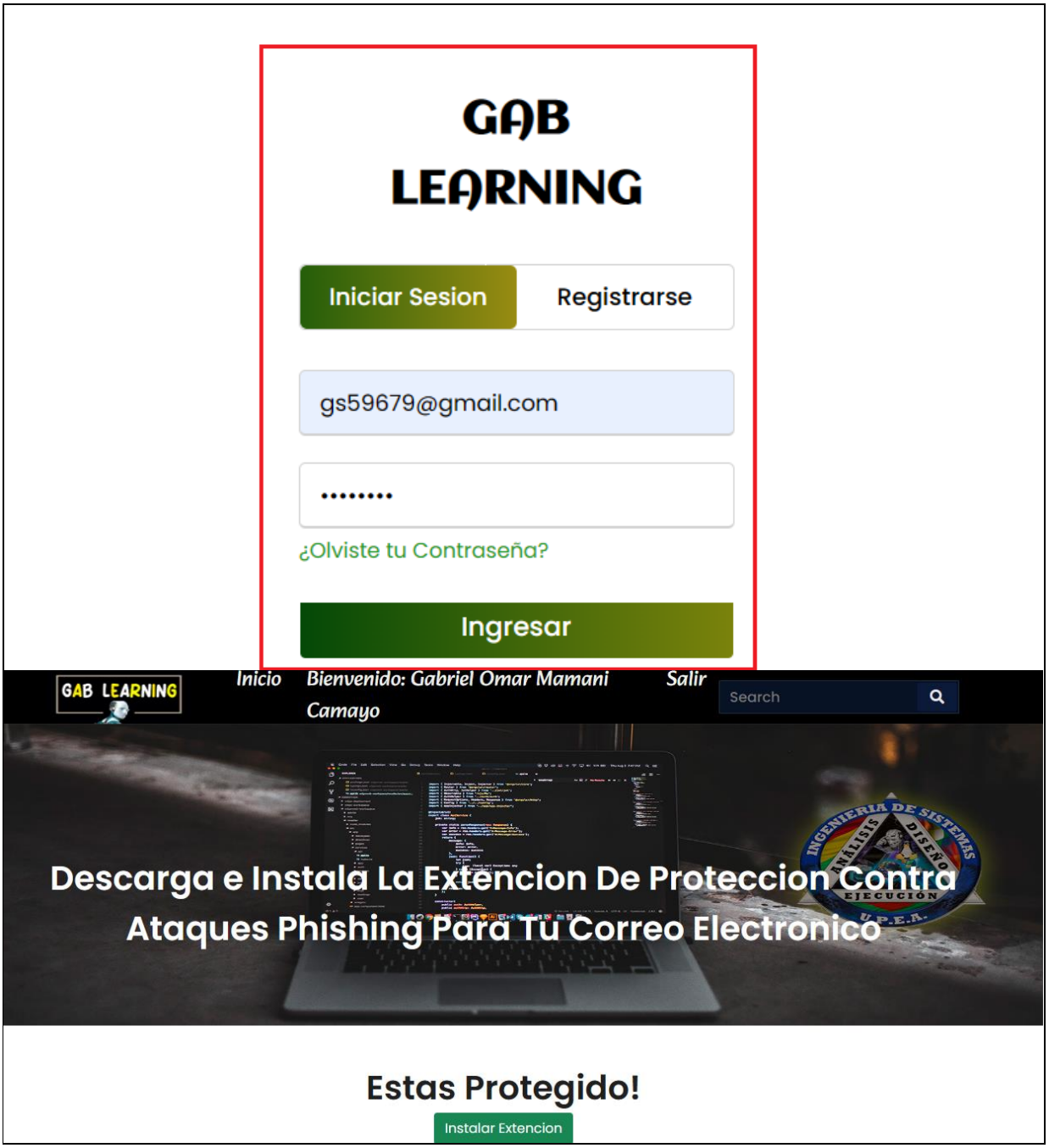

GAB
LEARNING
Iniciar Sesion
Registrarse
gs59679@gmail.com
••••••••
¿Olviste tu Contraseña?
Ingresar
GAB LEARNING
Inicio
Bienvenido: Gabriel Omar Mamani Camayo
Salir
Search
Descarga e Instala La Extencion De Proteccion Contra Ataques Phishing Para Tu Correo Electronico
Estas Protegido!
Instalar Extencion

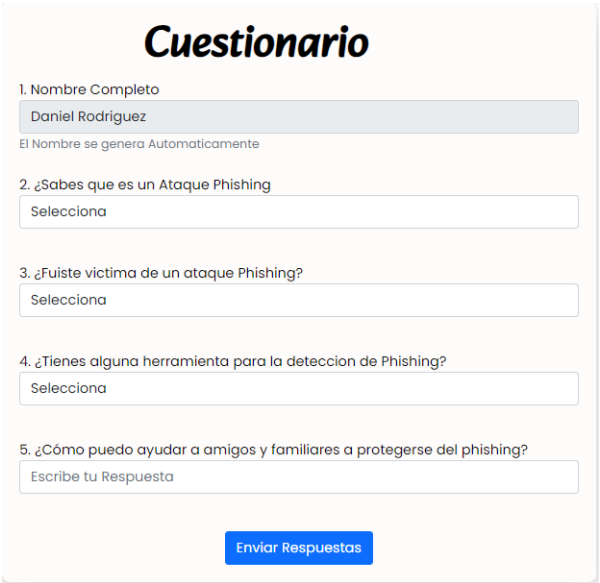
Tabla 4.14

Prueba Funcional Modulo Formulario de Detección

Suite de Pruebas del Prototipo								
No	**Caso de prueba**	**Prioridad**	**Precondiciones**	**Datos de entrada**	**Pasos**	**Resultado esperado**	**Resultado obtenido**	**Estado prueba**
Área Funcional: Módulo Formulario de Deteccion								
4	Módulo Formulario de Deteccion	Alta	Llenar datos correctamente	Tener la dirección url Ingresar la Url del sitio Web	1 abrir la página 2 Vista del formulario de detección. 3 Clic en el botón de Predecir	El usuario inicio sesión correctamente Direccionamiento a la interfaz de usuario	Abajo	Ok
Resultado Obtenido								
								

Tabla 4.15

Prueba Funcional Modulo Cuestionario

Suite de Pruebas del Prototipo								
No	**Caso de Prueba**	**Prioridad**	**Precondiciones**	**Datos de entrada**	**Pasos**	**Resultado esperado**	**Resultado obtenido**	**Estado prueba**
Área Funcional: Módulo Cuestionario								
5	Módulo Cuestionario	Media	Responder correctamente	Tener la dirección url Acceder al sistema con usuario y contraseña Nombre usuario, contraseña Respuestas seleccionadas (si o no)	1 abrir lapágina 2 Vista de losmenús de la página 3Registrarse 4 Acceder con usuario y contraseña 6 responder preguntas	Las respuestas del usuario se guardaron satisfactoriamente a la base de datos Direccionamiento a módulo de Formulario de deteccion	abajo	OK
Resultado Obtenido								
								

4.6. PRUEBA DE HIPOTESIS

Para la prueba de hipótesis primero debemos obtener resultados individuales de los módulos que ayudan a obtener la detección de sitios web falsos o legítimos y luego obtener un totalgeneral de resultados del prototipo.

Para esta prueba se tomó como muestra 10 personas del área de contadores de la empresa “LA ESPAÑOLA” en el cual se dio la probabilidad diagnóstica del prototipo.

4.6.1. Formulación de la Prueba de Hipótesis

La aplicación del modelo de detección de ataques Phishing mediante técnicas de machine Learning detecta los ataques informáticos con una eficacia del 95%.

4.6.2. Estado de la Hipótesis.

Para la demostración de la hipótesis se plantea lo siguiente.

Hipótesis

H0: La aplicación del modelo de detección de ataques Phishing mediante técnicas de machine Learning Detecta los ataques informáticos con una eficacia del 95%.

Hipótesis Nula

H1: La aplicación del modelo de detección de ataques Phishing mediante técnicas de machine Learning no detecta los ataques informáticos por suplantación de identidad.

Para la prueba de hipótesis se utilizará el método de T student:

Prueba de T Student

La prueba de T Student es un método muy utilizado para comprobar la hipótesis, se basa en comparar resultados de medias en muestras con la distribución normal calculando las diferencias que pueden llegar a tener en un antes y después.

Procedimiento

En la siguiente tabla se muestra resultados que se obtuvieron mediante el diagnóstico de detección de ataques Phishing del prototipo de detección, con parámetros de (0,1).

Donde:

x1=Probabilidad Diagnóstica del prototipo de deteccion de ataques phishing

Tabla 4.16

Evaluación del Diagnóstico

	Muestra (x1)
1	1
2	1
3	0
4	1
5	0
6	1
7	1
8	1
9	1
10	1
Total	

Nota. Resultados obtenidos a usuarios de la empresa de diferentes tipos de ataques.

- Fórmulas a utilizar la prueba de T – Student presenta las siguientes fórmulas para el cálculo estadístico.

Donde:

n = 10 Tamaños de muestra

x1= Probabilidad Diagnóstica del prototipo de detección de ataques phishing

Tomando en cuenta 0 = Datos Sitios no detectados

Tomando en cuenta 1 = Datos Sitios detectados.

Media común estimado:

Formula

$$\bar{x} = \frac{x_1 + x_2 + ... + x_n}{n}$$

Datos de la muestra

La media es el promedio de todos los datos dividido por la muestra.

La media seria:

$$X = \frac{1+1+0+1+0+1+1+1+1+1}{10}$$

X = 0.8

Varianza común estimada:

$$Varianza = \sum_{I=1}^{n} \frac{(x_i - \overline{x})^2}{n}$$

Tabla 4.17

Varianza Estimada

X	X-$\overline{X}$	(X - $\overline{X}$)²
1	1 - 0,8 = 0,2	(0,2)² = 0.04
1	1 - 0,8 = 0,2	(0,2)² = 0.04
0	0 - 0,8 = -0,8	(-0,8)² = 0.64
1	1 - 0,8 = 0,2	(0,2)² = 0.04
0	0 - 0,8 = -0,8	(-0,8)² = 0.64
1	1 - 0,8 = 0,2	(0,2)² = 0.04
1	1 - 0,8 = 0,2	(0,2)² = 0.04
1	1 - 0,8 = 0,2	(0,2)² = 0.04
1	1 - 0,8 = 0,2	(0,2)² = 0.04
1	1 - 0,8 = 0,2	(0,2)² = 0.04

Nota: Varianza estimada

La varianza seria:

$$R = \frac{2,2}{9} = 0.2444$$

Desviación Estándar común estimada:

$$S = \sqrt{0.2444} \qquad S = 0,4944$$

Prueba estadística T-Student

Formula:

$$t = \frac{X - \mu}{s/\sqrt{n}}$$

La prueba t student con los resultados comparando con el 95% de eficacia:

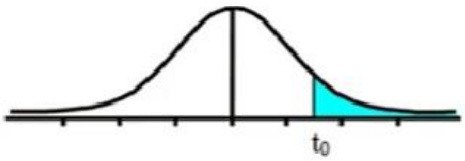
$$t = \frac{0{,}8 - 0.95}{0{,}4944 / \sqrt{10}} = -0{,}9594$$

Dando grados de libertad que es n – 1 serian 9 grados de libertad y con una nivel de significancia del 5% hacemos la comparación en la tabla estadística para hallar el punto crítico.

Figura 4.2

Tabla t-Student

Tabla t-Student



Grados de libertad	0.25	0.1	0.05	0.025	0.01	0.005
1	1.0000	3.0777	6.3137	12.7062	31.8210	63.6559
2	0.8165	1.8856	2.9200	4.3027	6.9645	9.9250
3	0.7649	1.6377	2.3534	3.1824	4.5407	5.8408
4	0.7407	1.5332	2.1318	2.7765	3.7469	4.6041
5	0.7267	1.4759	2.0150	2.5706	3.3649	4.0321
6	0.7176	1.4398	1.9432	2.4469	3.1427	3.7074
7	0.7111	1.4149	1.8946	2.3646	2.9979	3.4995
8	0.7064	1.3968	1.8595	2.3060	2.8965	3.3554
9	0.7027	1.3830	1.8331	2.2622	2.8214	3.2498
10	0.6998	1.3722	1.8125	2.2281	2.7638	3.1693
11	0.6974	1.3634	1.7959	2.2010	2.7181	3.1058
12	0.6955	1.3562	1.7823	2.1788	2.6810	3.0545
13	0.6938	1.3502	1.7709	2.1604	2.6503	3.0123
14	0.6924	1.3450	1.7613	2.1448	2.6245	2.9768
15	0.6912	1.3406	1.7531	2.1315	2.6025	2.9467

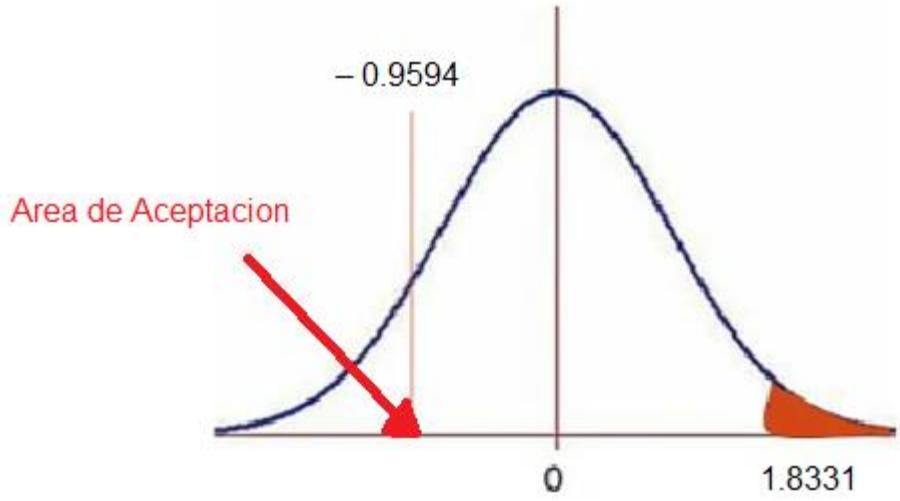
Nota: Tabla t-Student

Una vez hallado el punto critico podemos observar que tenemos un punto crítico de 1.8331 donde si cae nuestro resultado de la pruba t student estaría aceptando la

hipótesis nula. Pero en este caso si nuestro resultado fue – 0.9594 estaría rechazando la hipótesis nula y aceptando la hipótesis alterna que en este caso es que el Modelo de detección y reducción de ataques phishing si detecta ataques informáticos con un 95% de eficacia.

Figura 4.3

Grafico de aceptacion de hipotesis



Nota: Cuadro de aceptación de hipótesis

Segunda Prueba de Hipótesis

Para la segunda prueba de hipótesis se tomó como muestra 30 personas del área de sistemas de la empresa “MABET” en el cual se dio la probabilidad diagnóstica del prototipo.

Donde:

x1=Probabilidad Diagnóstica del prototipo de deteccion de ataques phishing

Tabla 4.18

Evaluación del Diagnóstico

	Muestra (x1)
1	0
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	1
14	1
15	1
16	1
17	1
18	1
19	1
20	1
21	0
22	0
Total	

Nota. Resultados obtenidos a usuarios de la empresa de diferentes tipos de ataques.

- Fórmulas a utilizar la prueba de T – Student presenta las siguientes fórmulas para el cálculo estadístico.

Donde:

n = 22 Tamaños de muestra

x1= Probabilidad Diagnóstica del prototipo de detección de ataques phishing

Tomando en cuenta 0 = Datos Sitios no detectados

Tomando en cuenta 1 = Datos Sitios detectados.

Media común estimado:

Formula

$$X = \frac{X1 + X2 + .. + Xn}{n}$$

Datos de la muestra

La media es el promedio de todos los datos dividido por la muestra.

La media seria:

$$X = \frac{0+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+0+0+0}{22}$$

Resultado de La Media

$$X = 0.818$$

Varianza común estimada:

Fórmula para hallar la varianza:

$$Varianza = \sum_{I=1}^{n} \frac{(x_i - \overline{x})^2}{n}$$

Tabla 4.19

Varianza Estimada Hipótesis 2

X	X-X̅	(X - X̅)²
0	0 - 0,818 = 0,818	(0,818)² = 0.669
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
1	1 - 0,818 = 0.19	(0,19)² = 0.0361
0	0 - 0,818 = 0,818	(0,818)² = 0.669
0	0 - 0,818 = 0,818	(0,818)² = 0.669
0	0 - 0,818 = 0,818	(0,818)² = 0.669

Nota: Varianza estimada De Hipótesis 2

La varianza seria:

$$R = \frac{2,2}{9} = 0.158$$

Desviación Estándar común estimada:

$S = \sqrt{0.158}$ S = 0,3974

Prueba estadística T-Student

Formula:

$$t = \frac{X - \mu}{S/\sqrt{n}}$$

La prueba t student con los resultados comparando con el 95% de eficacia:

$$\frac{0.818-0.95}{0.3974/\sqrt{22}} = -1.5574$$

Dando grados de libertad que es n – 1 serian 21 grados de libertad y con un nivel de significancia del 5% hacemos la comparación en la tabla estadística para hallar el punto crítico.

Figura 4.4

Tabla t-Student Hipotesis 2

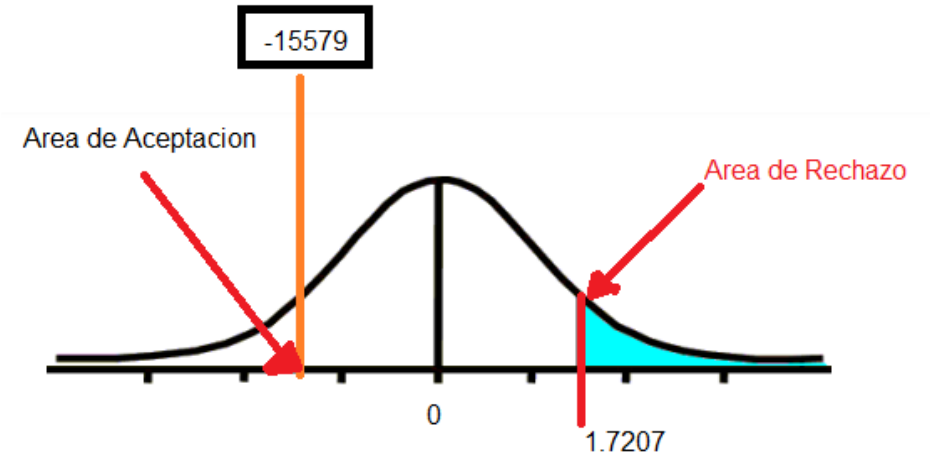
Grados de libertad	0.25	0.1	0.05	0.025	0.01	0.005
1	1.0000	3.0777	6.3137	12.7062	31.8210	63.6559
2	0.8165	1.8856	2.9200	4.3027	6.9645	9.9250
3	0.7649	1.6377	2.3534	3.1824	4.5407	5.8408
4	0.7407	1.5332	2.1318	2.7765	3.7469	4.6041
5	0.7267	1.4759	2.0150	2.5706	3.3649	4.0321
6	0.7176	1.4398	1.9432	2.4469	3.1427	3.7074
7	0.7111	1.4149	1.8946	2.3646	2.9979	3.4995
8	0.7064	1.3968	1.8595	2.3060	2.8965	3.3554
9	0.7027	1.3830	1.8331	2.2622	2.8214	3.2498
10	0.6998	1.3722	1.8125	2.2281	2.7638	3.1693
11	0.6974	1.3634	1.7959	2.2010	2.7181	3.1058
12	0.6955	1.3562	1.7823	2.1788	2.6810	3.0545
13	0.6938	1.3502	1.7709	2.1604	2.6503	3.0123
14	0.6924	1.3450	1.7613	2.1448	2.6245	2.9768
15	0.6912	1.3406	1.7531	2.1315	2.6025	2.9467
16	0.6901	1.3368	1.7459	2.1199	2.5835	2.9208
17	0.6892	1.3334	1.7396	2.1098	2.5669	2.8982
18	0.6884	1.3304	1.7341	2.1009	2.5524	2.8784
19	0.6876	1.3277	1.7291	2.0930	2.5395	2.8609
20	0.6870	1.3253	1.7247	2.0860	2.5280	2.8453
21	0.6864	1.3232	1.7207	2.0796	2.5176	2.8314
22	0.6858	1.3212	1.7171	2.0739	2.5083	2.8188

Nota: Punto Crítico para el rechazo de hipótesis

Una vez hallado el punto crítico podemos observar que tenemos un punto crítico de 1.7207 donde si cae nuestro resultado de la prueba t student estaría aceptando la hipótesis nula. Pero en este caso si nuestro resultado fue – 1.5574 estaría rechazando la hipótesis nula y aceptando la hipótesis alterna que en este caso es que el Modelo de detección y reducción de ataques phishing si detecta ataques informáticos con un 95% de eficacia.

Figura 4.5

Gráfico de aceptación de hipótesis



Nota: Cuadro de aceptación de hipótesis 2

4.6.3. Análisis de Resultados

Tomando en cuenta que el resultado final del porcentaje de diagnóstico en la muestra

Se observó de las 2 pruebas de hipótesis con un nivel de confianza del 95%, por lo tanto, en la primera prueba una aceptación del -0,9594 y un punto crítico de 1.8831 donde para la segunda prueba de hipótesis con un nivel de confianza del 95% con una aceptación del -15579 con un punto crítico de 1.7207 según la regla de la prueba de T- Student planteada, se llega a la conclusión que el diagnóstico probabilístico del prototipo tiene un nivel de confianza del 95%.

CAPÍTULO V

CONCLUSIONES Y RECOMENDA-CIONES

5.1. CONCLUSIONES

Realizamos la verificación del cumplimiento de los objetivos planteados:

- Analizar el estado actual de las técnicas de detección de ataques Phishing: Se enfoca en analizar las técnicas actuales de detección de ataques Phishing, incluyendo el uso de machine learning, para identificar sus fortalezas y debilidades se cumplió en un 90% el otro 10% fue a causa de que son demasiadas técnicas de aprendizaje automático por ende se analizó las más frecuentes y robustas para la prediccion.
- Diseñar nuevas características (features) y algoritmos de machine learning para mejorar la detección de ataques phishing y comparar su rendimiento con las técnicas existentes se cumplió en un 100%.
- Implementar un sistema de detección de ataques phishing basado en técnicas de machine learning que sea efectivo para reducir los ataques en nuestra región, y evaluar su desempeño en diferentes escenarios y situaciones se cumplió en un 100%.
- Evaluación de técnicas de machine learning empleadas en los últimos 5 años para la reducción de ataques Phishing se cumplió en un 100%.

Conclusión general. El prototipo cumple con un 97.5% en los objetivos planteados.

5.2. RECOMENDACIONES

El manejo y el aprendizaje de Machine Learning resultó ser muy eficiente, además que tiene gran aplicación en lo que es la detección de ataques Phishing.

Se recomienda usar el framework Flask ya que fue de gran ayuda al manejar los datos ya que también su lenguaje es Python.

Las url de los sitios deben ser claros y precisos para poder cumplir los requerimientos planteados anteriormente y para su estudio y análisis.

Mejorando el software hasta se puede obtener un rastreo de ip para poder saber quién nos está enviando el sitio Phishing, de esta manera podría ayudar usuarios y también a las autoridades para frenar el uso delictivo.

Se recomienda poder ampliar el estudio para la detección y reducción de ataques Phishing

BIBLIOGRAFÍA

Antonio Latorre. (2003, 20 de Febrero). Antonio Latorre. Obtenido de https://arteydocencia.files.wordpress.com/2013/08/investigacion-accion-antonio-latorre-2003-capc3adtulo-2.pdf

Aprende Machine Learning. (2020, 21 de agosto). Aprende Machine Learning. Obtenido de https://www.aprendemachinelearning.com/

Aprende Machine Learning. (2020). Modelos de deteccion de objetos. Machine Learning, 1.

B., M. C. (2019, 18 de Octubre). Universidad De Panama. Obtenido de https://upanama.educativa.org/archivos/repositorio/6000/6126/html/3_qu_es_.htm

Centeno, C. A. (2020). Ciberseguridad Informatica. Tecnologia Informatica, 2.

Cognodata. (2023, 2 de Enero). Cognodata. Obtenido de https://www.cognodata.com/automatizar-tareas-de-analisis-de-datos-con-machine-learning/

Converguers, J. (2017). Machine learning. Notiblog, 3.

Data Breach Investigations Report. (2022, 23 de Noviembre). Data Breach Investigations Report. Obtenido de https://enterprise.verizon.com/resources/reports/dbir/

El portal de la tesis. (2019, 20 de Octubre). Universidad de Colima. Obtenido de Universidad de Colima: https://n9.cl/ycx1

Fuertes Díaz, W. M. (2020, 18 de Diciembre). Universidad De Las Fuerzas Armadas.

Obtenido de http://repositorio.espe.edu.ec/xmlui/bitstream/handle/21000/23409/T-ESPE-044176.pdf?sequence=1&isAllowed=y

Global Suite. (2023, 30 de Marzo). Global Suite. Obtenido de https://www.globalsuitesolutions.com/es/la-familia-de-normas-iso-27000/

GOV.CO. (2021, 16 de Octubre). GOV.CO. Obtenido de https://herramientas.datos.gov.co/usos/modelo-de-analisis-de-sentimientos-en-redes-sociales-con-un-modelo-de-mineria-de-datos-y

ITESEC. (2020, 20 de Febrero). ITESEC. Obtenido de ITESEC: https://n9.cl/sxgtc

Manuel, J. (2020, 10 de Marzo). METODOLOGÍAS PARA LA REALIZACIÓN DE PROYECTOS DE. Obtenido de https://www.aeipro.com/files/congresos/2003pamplona/ciip03_0257_0265.2134.pdf

Microsoft.com. (2020, 22 de Marzo). Microsoft. Obtenido de https://www.microsoft.com/es-es/security/business/security-101/what-is-phishing

Mieres, J. (2020, 12 de Enero). Ataques Informaticos. Obtenido de https://www.evilfingers.org/publications/white_AR/01_Ataques_informaticos.pdf

Moncada Vargas, A. E. (2021, 20 de Junio). Repositorio institucional de la Universidad de Lima. Obtenido de https://hdl.handle.net/20.500.12724/13842

Muñoz Campuzano, P. S. (2021, 20 de Agosto). Universidad Politecnica Salesiana. Obtenido de http://dspace.ups.edu.ec/handle/123456789/20932

Muñoz, J. D. (2017, 17 de Noviembre). Open Webinars. Obtenido de

https://openwebinars.net/blog/que-es-flask/

Olmedo, J. (2020). Seguridad Informatica. Informaticos, 1.

omicrono. (2020, 28 de Mayo). omicrono. Obtenido de https://www.elespanol.com/omicrono/software/20200528/apple-compra-empresa-machine-learning-mejorar-siri/493450908_0.html

Programacion.org. (2021, 20 de Abril). Obtenido de Programacion.org: https://es.wikipedia.org/wiki/Proyecto_Jupyter

Ruiz, J. R. (2019). Deteccion de Malware, Metodos estadisticos y machine learning. Barcelona.

Sánchez Peño, J. M. (16 de junio de 209). Archivo Digital UPM. Obtenido de https://oa.upm.es/40012/1/PFC_JOSE_MANUEL_SANCHEZ_PENO_3.pdf

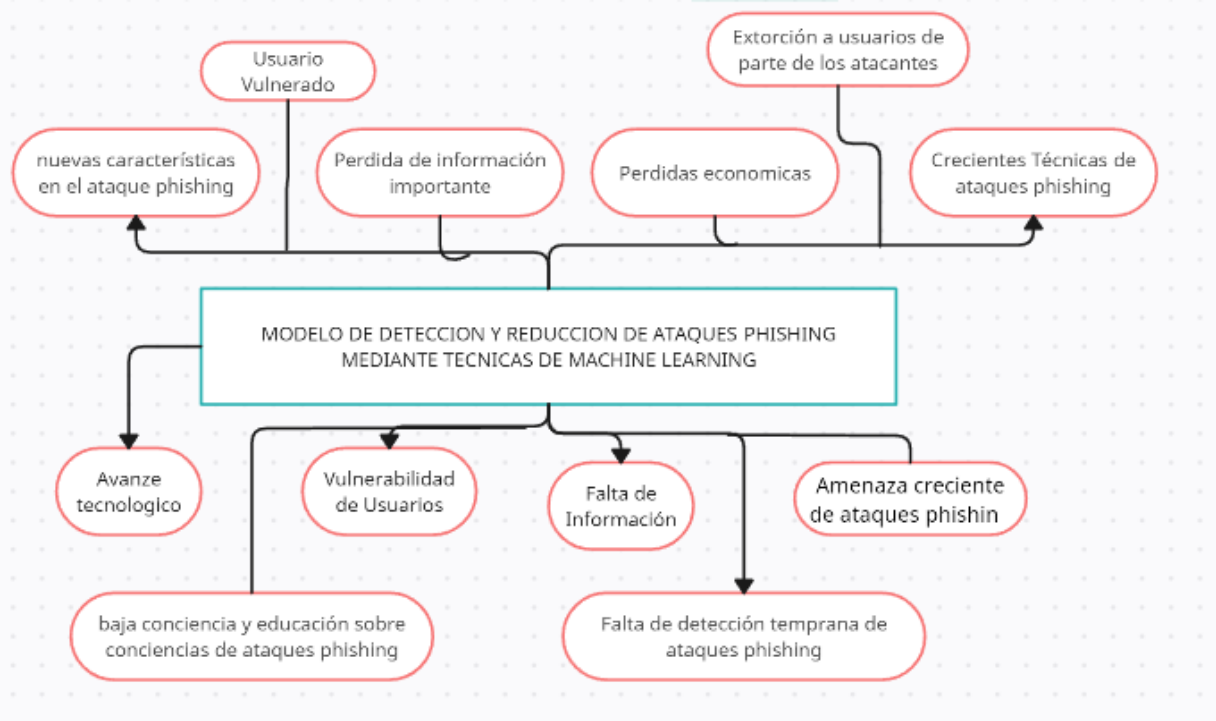
Universidad de Alcala. (2019, 13 de Febrero). Universidad de Alcala. Obtenido de https://www.master-data-scientist.com/scikit-learn-data-science/

Wang, S. (2019). Suplantacion de identidades. Quito: Magnacia.

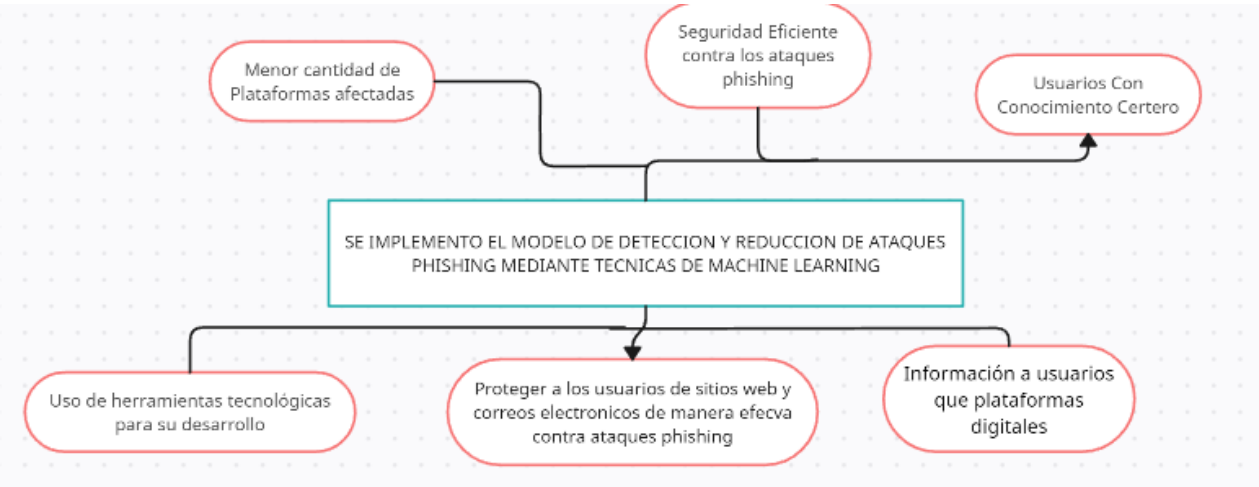
Wesner, F. B. (2020). Métodos de Detección Automática de Fraudes. Buenos Aires: Facultades de Ciencias Económicas.

ANEXOS

ARBOL DE PROBLEMAS



ARBOL DE OBJETIVOS



AVAL DE CONFORMIDAD

(TUTOR MEDOLOGICO)

El Alto, 9 de junio de 2023

Señor:
M. Sc Ing. David Carlos Mamani Quispe
DIRECTOR CARRERA
INGENIERIA DE SISTEMAS
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido director de la carrera:

Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: MODELO DE DETECCION Y REDUCCION DE ATAQUES PHISHING MEDIANTE TECNICAS DE MACHINE LEARNING.
MODALIDAD: TESIS DE GRADO
Univ. GABRIEL OMAR MAMANI CAMAYO
Registro Universitario: 200007105
Cedula de Identidad: 10920696 LP.

Para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de el Alto.

Atentamente,

.....

Ing. Dionicio Henry Pacheco Ríos
TUTOR METODOLOGICO
TALLER DE GRADO II

AVAL DE CONFORMIDAD

(TUTOR ESPECIALISTA)

El Alto, 2 de junio de 2023

Señor:
Ing. Dionicio Henry Pacheco Ríos
TUTOR METODOLOGICO
TALLER DE GRADO II
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido tutor metodológico:

Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: MODELO DE DETECCION Y REDUCCION DE ATAQUES PHISHING MEDIANTE TECNICAS DE MACHINE LEARNING.
MODALIDAD: TESIS DE GRADO
Univ. GABRIEL OMAR MAMANI CAMAYO
Registro Universitario: 200007105
Cedula de Identidad: 10920696 LP.

Para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de el Alto .

Atentamente,

.....

Ing. Lidia Máxima Rodríguez Choque
TUTOR ESPECIALISTA

AVAL DE CONFORMIDAD

(TUTOR REVISOR)

El Alto, 2 de junio de 2023

Señor:
Ing. Dionicio Henry Pacheco Ríos
TUTOR METODOLOGICO
TALLER DE GRADO II
Presente. -

REF. AVAL DE CONFORMIDAD

Distinguido tutor metodológico:

Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: MODELO DE DETECCION Y REDUCCION DE ATAQUES PHISHING MEDIANTE TECNICAS DE MACHINE LEARNING.
MODALIDAD: TESIS DE GRADO
Univ. GABRIEL OMAR MAMANI CAMAYO
Registro Universitario: 200007105
Cedula de Identidad: 10920696 LP.

Para su defensa publica y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de el Alto.

Atentamente,

.....

M. Sc. Lic, Zara Yujra Cama
TUTOR REVISOR

MANUAL DE USUARIO ADMINSTRADOR

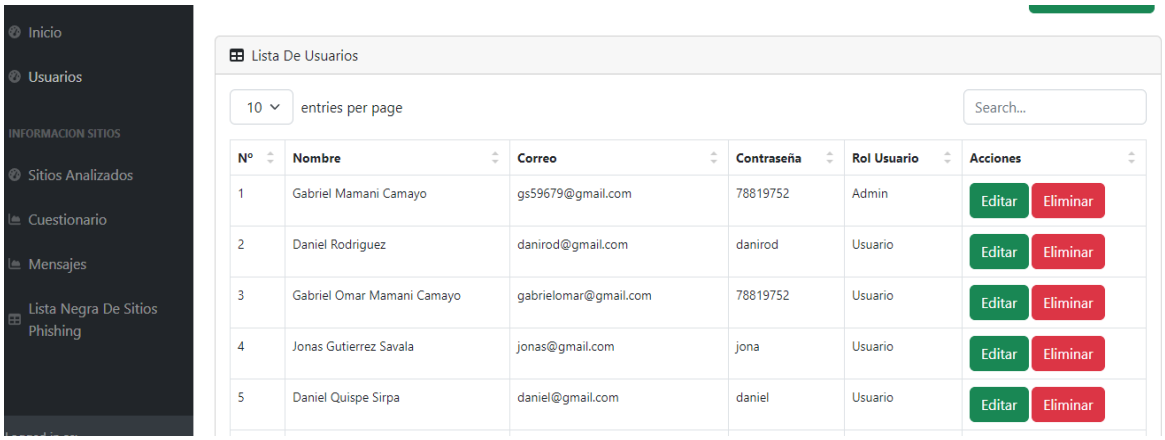
1. VALIDACION DE ACCESO.

PASO 1. En la página principal dar click en el campo Login

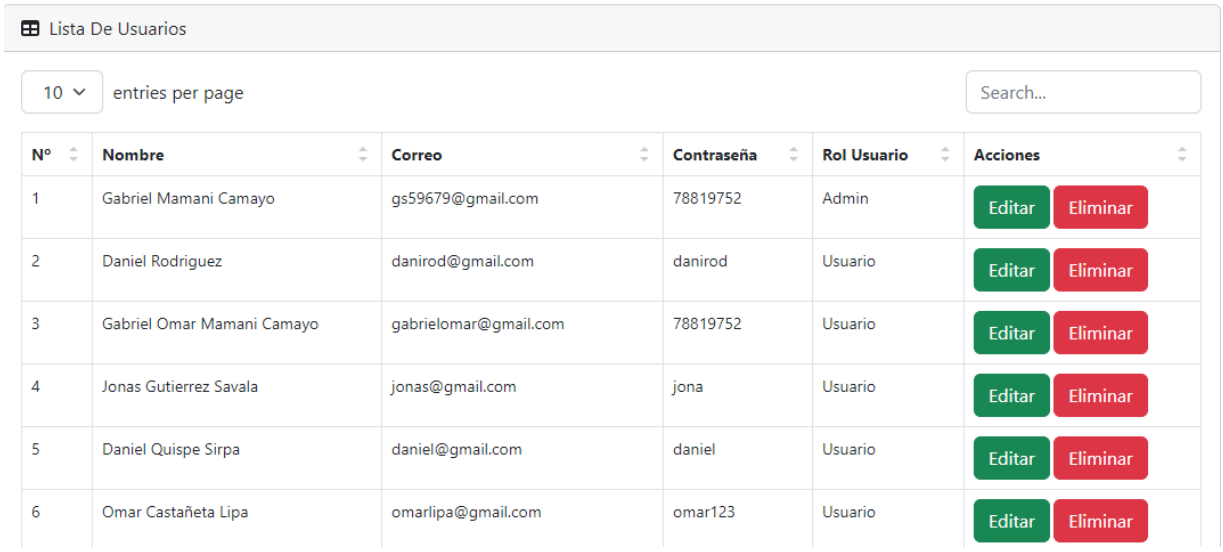
PASO 2. Introducir Usuario y Contraseña dar click en Ingresar.

2. VISTA DE USUARIOS

PASO 1. Una vez dentro debemos dar Click en el modulo Usuarios y se nos abrirá la tabla completa de usuarios Registrados.



PASO 2. Podemos Empezar a Editar los usuarios dando click en **Editar** como también Podemos Empezar a Eliminar los Usuarios Dando Click en el botón de **Eliminar.**

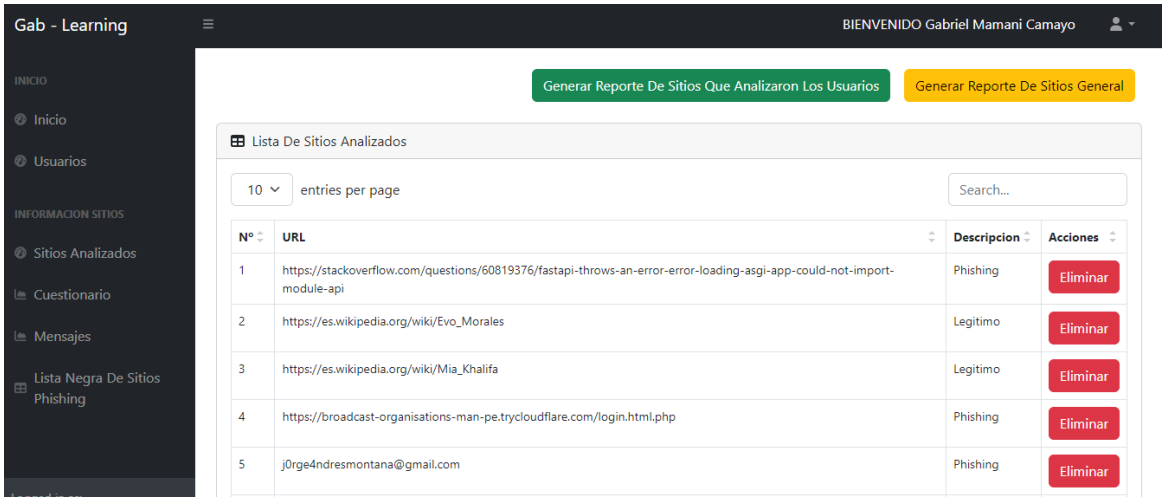


PASO 4. Podemos Generar el Reporte de Usuarios dando Click en el botón **Generar Reporte** y se nos generara un reporte en Excel de todos los usuarios Registrados.

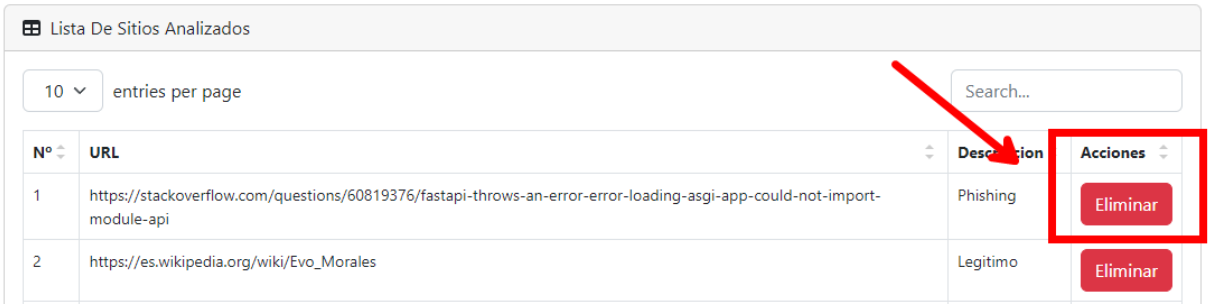
	A	B	C	D	E	F
1	id	nombre	correo	password	id_rol	rol
2	1	Gabriel Mamani Camayo	gs59679@gmail.com	78819752	1	Admin
3	2	Daniel Rodriguez	danirod@gmail.com	danirod	2	Usuario
4	3	Gabriel Omar Mamani Camayo	gabrielomar@gmail.com	78819752	2	Usuario
5	4	Jonas Gutierrez Savala	jonas@gmail.com	jona	2	Usuario
6	5	Daniel Quispe Sirpa	daniel@gmail.com	daniel	2	Usuario
7	6	Omar Castañeta Lipa	omarlipa@gmail.com	omar123	2	Usuario
8	7	Luna Garcias Lima	luna@gmail.com	luna	2	Usuario
9	8	German Garcia Linera	german2@gmail.com	German22	2	Usuario
10	9	German Lucio Canaviri	luciogerman@gmail.com	Lucio222	2	Usuario
11	10	Gabrielito	gabriel@gmail.com	Gabriel123	2	Usuario
12	11	Julio Cesar Machaca	julio34@gmail.com	Juliocesar1	2	Usuario
13	12	Mercedes Luque	mercedesluque_2@gmail.com	Merce455	2	Usuario
14	13	Ginelda Erika Choque Cruz	gine@gmail.com	Gine1212	2	Usuario

3. VISTA DE SITIOS ANALIZADOS.

PASO 1. Debemos hacer click en **Sitios Analizados** para tener una vista de la tabla de los sitios analizados de manera general.



PASO 2. Para Eliminar un Sitio analizado debemos hacer click en el botón **Eliminar**



PASO 3. Para generar un reporte de sitios analizados por los usuarios registrados debemos hacer click en el botón Generar **Reporte de los sitios que analizaron los usuarios** y se genera un reporte en Excel.

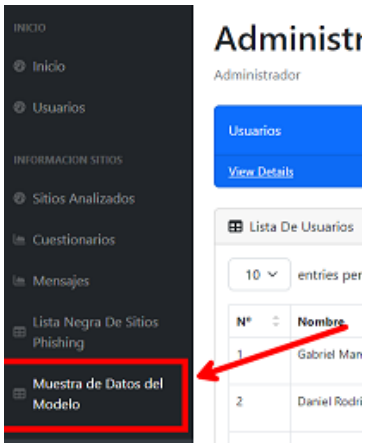
	A	B	C	D	E
1	id	url	descripcion	id_us	nombre
2	150	https://es.wikipedia.org/w	Legitimo	9	German Lucio Canaviri
3	208	https://www.youtube.com	Legitimo	2	Daniel Rodriguez
4	209	https://chat.openai.com/	Phishing	11	Julio Cesar Machaca
5	210	https://getbootstrap.esdoc	Phishing	11	Julio Cesar Machaca
6	211	http://localhost/phpmyadr	Phishing	11	Julio Cesar Machaca
7	212	https://es.wikipedia.org/w	Legitimo	11	Julio Cesar Machaca
8	213	https://facebook.com/	Phishing	12	Mercedes Luque
9	214	https://www.facebook.con	Legitimo	12	Mercedes Luque
10	215	https://es.wikipedia.org/w	Legitimo	2	Daniel Rodriguez

PASO 4. Para generar un reporte de sitios analizados de manera general debemos hacer click en el botón **Generar reporte de Sitios General.**

	A	B	C
1	id	url	descripcion
2	1	https://stackoverflow.com/questions/60819376/fastapi-t	Phishing
3	2	https://es.wikipedia.org/wiki/Evo_Morales	Legitimo
4	3	https://es.wikipedia.org/wiki/Mia_Khalifa	Legitimo
5	4	https://broadcast-organisations-man-pe.trycloudflare.co	Phishing
6	5	j0rge4ndresmontana@gmail.com	Phishing
7	6	holaamiguitos@gmail.com	Phishing
8	7	http://127.0.0.1:5000/sit_an	Phishing
9	8	https://www.youtube.com/results?search_query=que+e	Legitimo

4. VISTA DE LA BASE DE DATOS DE SITIOS PARA EL ENTRENAMIENTO DEL MODELO.

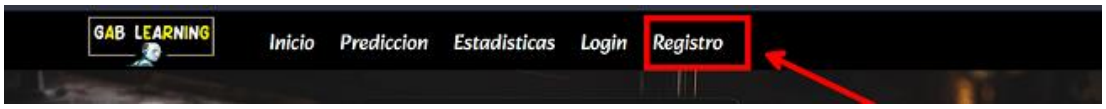
PASO 1. Hacer click en el módulo de **Muestra de Datos del Modelo** y se descargara un archivo de tipo csv. Con todos los datos legítimos y phishing del modelo entrenado.



MANUAL PARA EL USUARIO

1. REGISTRO DE USUARIO

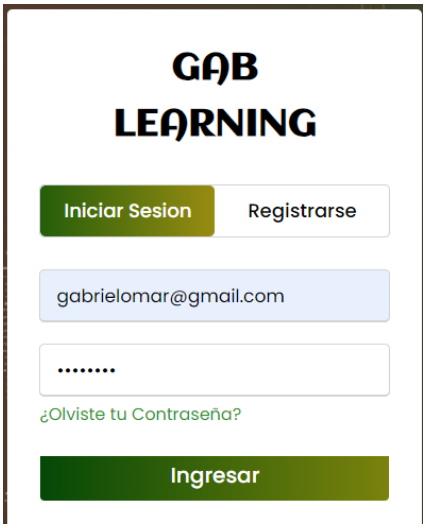
PASO 1. Del menú principal dar click en el botón registro.



PASO 2. Ingresar los datos correspondientes en el formulario de registro y dar click en **Registrar**

2. VALIDACION DE USUARIO.

PASO 1. Del menú principal dar click en login e ingresar los datos en el formulario de inicio de sesión y dar click en el botón **Ingresar**



3. FORMULARIO DE DETECCION PARA USUARIOS

PASO 1. Del menú hacer click en el botón de **Analizar Sitio.**

PASO 2. Ingresar un sitio cualquiera y dar click en el botón de analizar sitio y comenzara a analizar el sitio y dar el resultado si es un sitio legitimo o sitios Phishing.

4. VISTA DE SITIOS ANALIZADOS POR EL USUARIO.

PASO 1. Hacer click en el botón de sitios analizados y mostrara los sitios que analizo el usuario Registrado

PASO 2. Hacer click en el botón de generar reporte de sitios analizados.

5. DESCARGA DE EXTENSION WEB PARA PROTECCION EN TIEMPO REAL

PASO 1. De la página principal Hacer click en el botón de descargar Instalar Extensión y se instalara la extensión web de protección contra ataques phishing.

PASO 2. Hacer click en el icono de la barra de tareas del navegador y ver en tiempo real si un sitio es legítimo o Phishing.

MANUAL DE USUARIO TECNICO

REQUERIMIENTOS:

1. EQUIPO.

- ✓ Procesador: AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx 2.30 GHz
- ✓ RAM 8 GB
- ✓ Tipo de sistema: Sistema operativo de 64 bits, procesador x64
- ✓ Disco duro 200 GB

2. SOFTWARE

Descripción de requerimientos de programas software:

- ✓ absl-py==1.2.0
- ✓ asgiref==3.5.2
- ✓ attrs==22.1.0
- ✓ cycler==0.11.0
- ✓ Flask==2.2.3
- ✓ Flask_MySQL==1.5.2
- ✓ Flask_MySQLdb==1.0.1
- ✓ fonttools==4.37.1
- ✓ imutils==0.5.4
- ✓ kiwisolver==1.4.4
- ✓ matplotlib==3.5.3
- ✓ mediapipe==0.8.10.1
- ✓ numpy==1.23.2
- ✓ opencv-contrib-python==4.6.0.66
- ✓ Java Script
- ✓ packaging==21.3
- ✓ Pillow==9.2.0
- ✓ protobuf==3.20.1
- ✓ Jupyter==1.0.0
- ✓ Jupyter_server==2.5.0

- ✓ scikit-learn==0.24.1
- ✓ mysql==0.0.3
- ✓ PyMySQL==1.0.2
- ✓ pyparsing==3.0.9
- ✓ python-dateutil==2.8.2
- ✓ pytz==2022.2.1
- ✓ six==1.16.0
- ✓ sqlparse==0.4.2
- ✓ boostrap 5