TRABAJO DE GRADO
Opción Seminario

Sistema Inteligencia de Detección de URLS Maliciosas

Jair Restrepo Quintero

Lorena Quiñones Rodríguez

Asesor
Danny López Segura

Opción de Trabajo de grado Seminario

Corporación Universitaria Remington
Facultad de Ciencias Básicas e
Ingenierías Programa Ingeniería de
Sistemas

Santiago de Cali – Valle del cauca

2025

TRABAJO DE GRADO
Opción Seminario

Nota de Aceptación

_____

_____

_____

_____

_____

_____
Asesor

_____
Jurado

_____
Jurado

_____
Jurado

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario

Dedicatoria

Dedico este trabajo de grado con profundo agradecimiento y admiración a todas las personas que han sido parte integral de mi trayectoria académica y que han dejado una huella imborrable en mi camino hacia la culminación de este proyecto.

A mis padres, quienes han sido mis pilares inquebrantables, mis guías sabios y mi constante fuente de inspiración. Su amor incondicional, su apoyo inquebrantable y su sacrificio desinteresado han sido los cimientos sobre los cuales he construido mi educación y mi crecimiento personal. Agradezco infinitamente su presencia constante, su aliento constante y su fe inquebrantable en mí. Cada paso que he dado en este camino ha sido impulsado por su amor y dedicación, y este logro es también suyo.

Jair restrepo

Quisiera expresar mi profundo agradecimiento a todas las personas que han contribuido de manera directa o indirecta en la realización de este trabajo. Su apoyo incondicional y esfuerzo ha sido esencial para lograr los resultados obtenidos.

En primer lugar, quiero agradecer sinceramente los profesores que hemos tenido a lo largo de nuestra carrera, por su dedicación, paciencia y valiosas orientaciones a lo largo de todo el proceso.

Asimismo, quiero expresar mi gratitud a mis compañeros de clase, quienes han contribuido con sus ideas, sugerencias y debates enriquecedores. Sus aportes han ampliado mi perspectiva sobre el tema y me han inspirado a esforzarme más en este proyecto.

Lorena Quiñones

TRABAJO DE GRADO
Opción Seminario

Agradecimientos

Queremos expresar nuestros más sinceros agradecimientos a:

Dios, por iluminarnos en cada paso durante este proceso formativo y permitirnos convertirnos en profesionales.

Al tutor DANNY LOPEZ SEGURA, asesor del proyecto por su compromiso, dedicación, por compartir sus conocimientos, experiencias y ser un guía durante toda la carrera.

A todos los docentes de la Corporación Universitaria Remington, quienes con su profesionalismo estuvieron con nosotros brindando su orientación y apoyo durante el proceso formativo.

Para nuestros padres quienes nos han brindado su apoyo incondicional para vernos crecer en nuestras metas.

A todas las personas allegadas que siempre han confiado en nuestras capacidades y han extendido su mano para permitir alcanzar nuestros sueños.

TRABAJO DE GRADO
Opción Seminario

Tabla de contenido

Dedicatoria	3
2.Agradecimientos	7
3.Resumen	10
4.Introducción	11
5.Planteamiento del Problema	12
6.Objetivos	14
7.Justificación	16
8.Metodología	18
8.121	
9.MODELO	23
10.Conclusiones	32
11.Bibliografía	33

Tabla de Ilustraciones

Ilustración 1Buenas Practicas *14*
Ilustración 2Diagrama de Proceso *23*
Ilustración 3Limpiar y preparándolos datos *24*
Ilustración 4 Importación de datos *25*
Ilustración 5Formato *25*
Ilustración 6Entrenar el modelo *26*
Ilustración 7Muestra *27*
Ilustración 8Matriz de Confusión *28*
Ilustración 9 Guardar el Modelo *28*
Ilustración 10 Pruebas URLs *29*
Ilustración 11Probabilidad de prueba *30*
Ilustración 12Prueba URLs *30*
Ilustración 13Probabilidad de prueba *31*
Ilustración 14Ultima prueba *31*

TRABAJO DE GRADO
Opción Seminario
Resumen

Este proyecto propone el desarrollo de una herramienta automatizada basada en inteligencia artificial para la detección de URLs maliciosas, con el objetivo de fortalecer la ciberseguridad en un entorno digital cada vez más expuesto a amenazas. Utilizando técnicas de aprendizaje automático supervisado y un conjunto de datos reales compuesto por más de 650,000 registros, el sistema es capaz de identificar enlaces asociados a malware, phishing, defacement y otros ataques comunes.

La metodología incluye limpieza, preprocesamiento, vectorización y codificación de datos, seguida del entrenamiento de un modelo con el algoritmo Random Forest. Se evalúa su desempeño mediante métricas clave como precisión, recall, f1-score y matriz de confusión, complementadas con visualizaciones que facilitan la interpretación de resultados.

Más allá de su utilidad inmediata, el proyecto se estructura de manera modular y reutilizable, permitiendo su integración en futuras soluciones de ciberseguridad. En conjunto, se presenta una propuesta técnica, escalable y proactiva frente a una amenaza persistente como lo son las URLs maliciosas, ayudando a proteger tanto a usuarios individuales como a organizaciones.

Palabras Claves: Ciberseguridad, URLs maliciosas, Machine Learning, Detección automática

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario

Introducción

La confianza en el entorno digital se ha convertido en un requisito fundamental para nuestra vida diaria. Desde enviar un simple mensaje hasta acceder a servicios bancarios o realizar trámites gubernamentales, dependemos cada vez más de plataformas en línea. Pero con esa dependencia también crecen los riesgos, y entre ellos, uno de los más silenciosos y efectivos es el uso de enlaces maliciosos.

Hoy en día, basta con un clic en una URL disfrazada para comprometer la seguridad de una persona, una empresa o incluso una institución pública. El problema no radica únicamente en el contenido malicioso al que conducen estos enlaces, sino en la sofisticación con la que logran pasar desapercibidos. Y aunque existen filtros y herramientas de seguridad, muchas veces no logran anticiparse a amenazas nuevas o poco conocidas.

Este proyecto busca ofrecer una solución proactiva a ese desafío: una herramienta basada en aprendizaje automático capaz de analizar URLs y clasificarlas como benignas o maliciosas. Para su desarrollo se empleó un conjunto de datos real y extenso, acompañado de un proceso riguroso de limpieza, transformación y entrenamiento con algoritmos robustos como Random Forest.Pero más allá del modelo técnico, este trabajo se enfoca en construir una base sólida y flexible, que pueda evolucionar con el tiempo y adaptarse a nuevos contextos. También se contempló la creación de visualizaciones que faciliten la interpretación del desempeño del modelo, haciendo que los resultados sean comprensibles tanto para expertos como para quienes toman decisiones en el ámbito de la seguridad. En definitiva, este proyecto no solo plantea una respuesta ante un tipo de amenaza muy frecuente, sino que también representa una apuesta por la automatización inteligente y la mejora continua en la protección del entorno digital.

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario

Planteamiento del Problema

Con el incremento del uso de internet en todas las esferas de nuestra vida cotidiana, las amenazas cibernéticas han evolucionado de manera alarmante. En particular, los ataques a través de **URLs maliciosas** se han convertido en una de las tácticas más comunes utilizadas por los ciberdelincuentes. Estos enlaces, que muchas veces parecen inofensivos, son diseñados para engañar a los usuarios, redirigiéndolos a sitios web falsos donde pueden ser víctimas de **phishing**, robarles datos sensibles o incluso infectar sus dispositivos con malware.

La dificultad para detectar estas amenazas radica en que las URLs maliciosas suelen imitar a las legítimas con gran precisión. Las diferencias pueden ser tan sutiles que incluso un usuario experimentado podría no notar que está siendo engañado. Esto resulta especialmente problemático porque muchos usuarios no cuentan con los conocimientos técnicos necesarios para identificar una URL peligrosa.

A pesar de los esfuerzos por mitigar estos ataques a través de listas negras o filtros estáticos, las amenazas continúan evolucionando, haciendo que los métodos tradicionales no sean suficientes. Los filtros basados en reglas fijas no pueden detectar ataques nuevos o variantes que no se encuentran en las bases de datos preexistentes. Esto ha llevado a un aumento en la necesidad de soluciones **más inteligentes y dinámicas**, que no solo respondan a amenazas conocidas, sino que sean capaces de identificar patrones y comportamientos anómalos que indiquen un riesgo incluso antes de que el ataque se materialice.

Es aquí donde la **inteligencia artificial** juega un papel crucial. Mediante el uso de algoritmos de **aprendizaje automático**, podemos entrenar modelos que aprendan a reconocer qué hace a una URL maliciosa, basándose en datos previos. Esto permitiría una detección **automática** y **proactiva**, sin necesidad de intervención manual o la dependencia de bases de datos limitadas.

Este proyecto se enfoca en construir una herramienta que utilice estas técnicas avanzadas para identificar URLs maliciosas, proporcionando una capa adicional de seguridad tanto para usuarios comunes como para organizaciones. La herramienta no solo pretende detectar amenazas de forma eficiente, sino que también será flexible y escalable, permitiendo su integración con otros sistemas de seguridad existentes.

*Ilustración SEQ Ilustración * ARABIC 1Buenas Practicas*

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario

Objetivos

Objetivo general

Desarrollar un modelo predictivo de clasificación de URLs maliciosas utilizando técnicas de Machine Learning

Objetivos Específicos

- **Aplicar limpieza y preprocesamiento de datos reales de URLs:** Realizar un tratamiento exhaustivo de los datos recolectados desde fuentes reales, eliminando duplicados, manejando valores nulos, extrayendo características relevantes.

- **Entrenar y evaluar un modelo de clasificación con técnicas supervisadas:** Seleccionar y aplicar algoritmos de clasificación supervisada (como Random Forest) para distinguir entre URLs benignas y maliciosas. Evaluar su desempeño mediante métricas como precisión, recall, F1-score y matriz de confusión, utilizando técnicas como validación cruzada para garantizar resultados robustos.

- **Implementar una visualización clara del desempeño del modelo:** Diseñar e implementar visualizaciones gráficas que representen los resultados del modelo, tales como la matriz de confusión, facilitando así la interpretación de los resultados por parte de analistas de seguridad.

- **Crear una base reutilizable para futuras herramientas de ciberseguridad:** Desarrollar una arquitectura modular que permita reutilizar el código y las técnicas desarrolladas en futuros proyectos de ciberseguridad, incluyendo la documentación del proceso, definición de estructuras de datos estandarizadas y buenas prácticas para el manejo de datos y modelos en contextos de detección de amenazas.

Justificación

En un mundo cada vez más conectado, donde la mayoría de nuestras actividades cotidianas como la banca, las compras, el trabajo y hasta la educación dependen de plataformas digitales, la ciberseguridad se ha convertido en un pilar fundamental para garantizar la confianza en el uso de la tecnología. Uno de los vectores de ataque más utilizados por los ciberdelincuentes en los últimos años ha sido el uso de URLs maliciosas. A través de estos enlaces, que muchas veces se camuflan como legítimos, los atacantes logran engañar a los usuarios para robar información confidencial, instalar software malicioso o suplantar identidades mediante técnicas como el phishing.

Este tipo de amenazas es especialmente peligroso porque no siempre es fácil de detectar a simple vista. A menudo, los usuarios no tienen el conocimiento técnico necesario para identificar que un enlace es sospechoso, y los métodos tradicionales de filtrado y bloqueo de URLs pueden no ser lo suficientemente efectivos frente a amenazas nuevas o sofisticadas. Por eso, surge la necesidad de desarrollar herramientas más inteligentes, capaces de identificar patrones y comportamientos anómalos que indiquen una posible amenaza, incluso cuando esta no ha sido reportada previamente.

Aquí es donde entra en juego la inteligencia artificial, y en particular el aprendizaje automático (machine Learning), que permite entrenar modelos capaces de aprender a partir de datos reales y tomar decisiones basadas en evidencias. Este proyecto busca precisamente

eso: construir una herramienta automatizada que, con ayuda de técnicas supervisadas de clasificación, pueda analizar URLs, identificar cuáles son maliciosas y alertar al usuario o al sistema de seguridad antes de que sea demasiado tarde.

Lo interesante de esta propuesta no es solo su utilidad inmediata, sino su potencial a futuro. Al trabajar con una base de datos real y actualizada, y al estructurar el proyecto de forma modular y reutilizable, se abre la puerta para seguir mejorando esta solución, adaptándola a nuevos entornos, integrándola en plataformas más grandes o combinándola con otras tecnologías de ciberseguridad. Además, el desarrollo de visualizaciones claras del desempeño del modelo facilita la interpretación de los resultados, lo que hace que esta herramienta sea útil no solo para expertos en tecnología, sino también para analistas de seguridad o personal de TI que necesita tomar decisiones rápidas y fundamentadas.

En resumen, este proyecto responde a una necesidad real y urgente en el ámbito de la ciberseguridad. No solo ayuda a mitigar riesgos asociados al uso cotidiano de internet, sino que también aporta una solución técnica con potencial de escalabilidad y mejora continua. Es una apuesta por un enfoque preventivo, inteligente y automatizado para proteger a usuarios y organizaciones frente a un tipo de amenaza que, lamentablemente, sigue en aumento.

Metodología

El desarrollo de esta herramienta de detección de URLs maliciosas se llevó a cabo a través de una metodología mixta que combina el enfoque experimental del análisis de datos y entrenamiento de modelos de aprendizaje automático, con una estructura ordenada y escalable basada en principios del desarrollo de software. Esta integración permite no solo evaluar la efectividad del modelo propuesto, sino también garantizar la reutilización, mantenibilidad y extensión del sistema a futuro.

- **Recolección y preparación de datos:** La primera fase del proyecto consistió en la adquisición y tratamiento de un conjunto de datos adecuado para el entrenamiento del modelo de clasificación. Se utilizó un dataset extraído de Kaggle, ampliamente referenciado en investigaciones de ciberseguridad, que contiene más de 650,000 registros de URLs categorizadas en cuatro clases: benign, malware, phishing y defacement. Esta diversidad de categorías permitió entrenar un modelo con un enfoque multiclase, más realista y cercano a las condiciones del entorno web actual.

- **la etapa de limpieza de datos**, se eliminaron registros duplicados y aquellos que contenían valores nulos en campos críticos, asegurando así la calidad del conjunto de entrenamiento.

- **proceso de preprocesamiento** sobre las URLs que incluyó la eliminación de prefijos como “http://”, “https://” y “www.”, así como caracteres separadores comunes como barras, guiones o puntos. Todo el texto se convirtió a minúsculas para homogenizar los datos y reducir la dimensionalidad del espacio de características.

- **la técnica de vectorización:** se utilizó con CountVectorizer, que transforma las cadenas de texto en vectores numéricos, permitiendo así el análisis cuantitativo por parte del modelo.

- **codificación de las etiquetas**, se empleó la herramienta LabelEncoder, que asigna un valor numérico entero a cada clase (por ejemplo, 0 para benign, 1 para malware, etc.), facilitando la categorización supervisada de los datos.

Entrenamiento y evaluación del modelo

- El dataset se dividió en dos subconjuntos: el 80% para entrenamiento del modelo y el 20% restante para evaluación, respetando una distribución aleatoria estratificada para preservar la proporción de clases.

- Para el entrenamiento se eligió el algoritmo RandomForestClassifier, conocido por su robustez, eficiencia y capacidad para manejar problemas de clasificación multiclase. Se entrenó el modelo utilizando los datos vectorizados y las etiquetas codificadas, ajustando los parámetros básicos del clasificador para

lograr un equilibrio entre complejidad y rendimiento.

- Una vez finalizado el entrenamiento, se procedió a la evaluación del desempeño del modelo mediante métricas estándar como precisión, recall, f1-score y la matriz de confusión. Estas métricas permitieron analizar qué tan bien el modelo distinguía entre las diferentes clases, identificando tanto los aciertos como los errores de clasificación en un entorno de prueba realista.

Metodología del Software

Paralelamente al análisis de datos, el desarrollo del sistema que implementa esta solución se llevó a cabo siguiendo un enfoque basado en el modelo incremental, una metodología ágil que permite construir el software de forma modular, incorporando funcionalidades de manera progresiva y ajustando según retroalimentación o pruebas.

Este modelo fue especialmente útil en este proyecto debido a que combina exploración técnica (entrenamiento del modelo) con la necesidad de presentar resultados funcionales desde etapas tempranas. El proyecto se organizó en los siguientes incrementos:

1. Prototipo inicial: Se desarrolló un primer módulo de consola capaz de cargar el dataset, procesar los datos, entrenar el modelo y mostrar las métricas básicas. Esta etapa sirvió como validación de la viabilidad técnica del enfoque y permitió ajustar los primeros parámetros del algoritmo.

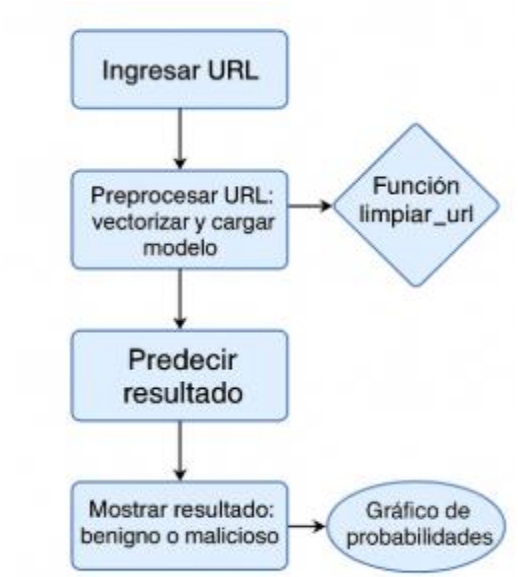
2. Refinamiento y modularización: En esta fase se separó la lógica del sistema en módulos independientes (carga de datos, preprocesamiento, entrenamiento, visualización), permitiendo una estructura más limpia, mantenible y fácil de extender.

3. Visualización y análisis de resultados: Se incorporaron herramientas de visualización utilizando bibliotecas como Matplotlib y Seaborn, lo que permitió representar gráficamente los resultados del modelo (matriz de confusión), facilitando su interpretación por parte de usuarios técnicos y no técnicos.

4. Documentación y reutilización: Finalmente, se creó una base de código documentada que puede ser reutilizada en proyectos futuros relacionados con ciberseguridad, tanto para análisis de URLs como para otros tipos de amenazas. Se dejó abierta la posibilidad de incorporar nuevas funcionalidades como clasificación en tiempo real o integración con sistemas de monitoreo web.

En conjunto, esta metodología asegura que tanto el modelo como la herramienta que lo implementa sean sólidos, escalables y útiles en contextos reales de detección de amenazas. Además, al seguir un enfoque iterativo, se garantiza que cada componente pueda ser mejorado de manera independiente conforme evolucionen los requerimientos o se descubran nuevas amenazas.

Ilustración 2Diagrama de Proceso



MODELO

El dataset utilizado en este proyecto se llama malicious_phish.csv, el cual fue descargado desde Kaggle, una plataforma reconocida a nivel mundial dedicada a la ciencia de datos y al aprendizaje automático.

Kaggle es una comunidad en línea perteneciente a Google, donde profesionales en ciencia de datos, inteligencia artificial, machine learning y ciberseguridad comparten datasets, compiten en retos de modelado predictivo, y publican notebooks con soluciones y análisis avanzados.

El dataset fue construido y validado por expertos en el área de ciberseguridad, y contiene una recopilación realista de más de 650,000 URLs, clasificadas en las siguientes categorías:

phishing: enlaces creados para suplantar identidades o robar datos.

malware: enlaces que distribuyen software malicioso.

defacement: páginas web modificadas o intervenidas por atacantes.

benign: URLs completamente seguras

.

Limpiar y preparando los datos

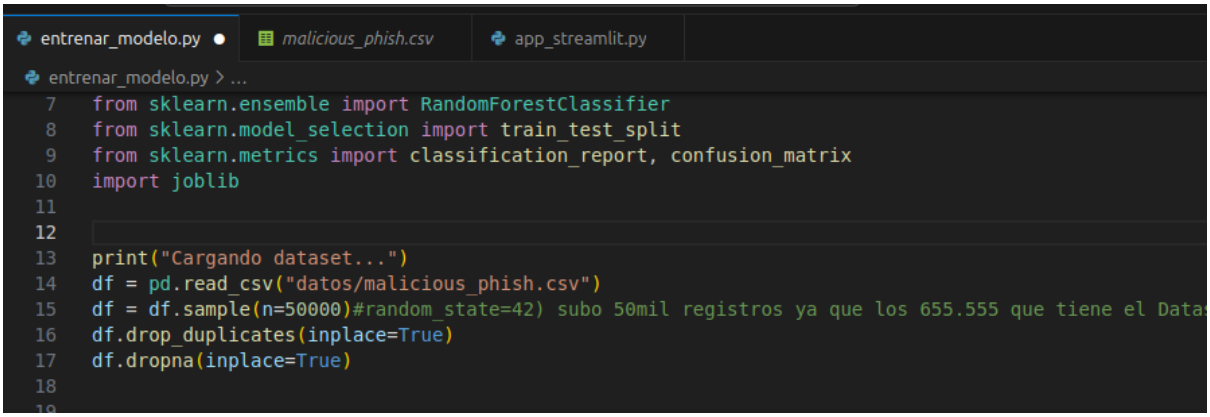


Ilustración 3Limpiar y preparándolos datos

Se importa un conjunto de datos (malicious_phish.csv) con miles de URLs clasificadas por tipo.

```
18
19
20  print("Preprocesando URLs...")
21  def limpiar_url(url):
22      url = url.lower()
23      url = url.replace('http://', '').replace('https://', '')
24      url = url.replace('/', ' ').replace('.', ' ')
25      return url
26
27  df['url_procesada'] = df['url'].apply(limpiar_url)
28
```

Ilustración 4 Importación de datos

La función **limpiar_url** lo que hace es eliminar partes innecesarias (http, /, .) y se convierten a minúsculas para que el modelo no distingue entre variantes visuales sin importancia un ejemplo sería:

URL original: https://login.bancodeoccidente.com/login.php

Después de limpiar: login bancodeoccidente com login php

El objetivo es limpiar y preparar las URLs para que puedan ser usadas como texto de entrada en el modelo de Machine Learning.

Después de limpiar las URLs, se necesita transformar ese texto en un formato que el modelo pueda entender.

```
vectorizador = CountVectorizer()
X = vectorizador.fit_transform(df['url_procesada'])


decodificador = LabelEncoder()
y = decodificador.fit_transform(df['type'])

X_entrena, X_test, y_entrena, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Este método transforma las URLs preprocesadas en una matriz de números. Cada fila representa una URL, y cada columna representa una “palabra” que apareció en alguna URL. El número en cada celda es la cantidad de veces que esa palabra aparece en esa URL

El modelo también necesita que las clases (tipos de amenaza) estén en formato numérico. Por ejemplo, transformar palabras como phishing, malware o benign en los números 1, 2, 3, etc. Esto se hace usando LabelEncoder():

Luego de preparar los datos, se dividen en dos subconjuntos: uno para entrenamiento (el 80% del total) y otro para pruebas reales (el 20%).

Esta división permite que el modelo aprenda con la mayoría de los datos y luego sea evaluado con datos que nunca ha visto, simulando un escenario real.

De esta forma, se comprueba si el modelo realmente puede identificar correctamente una URL maliciosa o benigna en situaciones reales.

Entrenar el modelo

```
print("Entrenando modelo...")                    (parameter) random_state: Int | RandomState | None
modelo = RandomForestClassifier(n_estimators=200, random_state=42)
modelo.fit(X_entrena, y_entrena)


y_pred = modelo.predict(X_test)
print("\n--- Reporte de Clasificación ---")
print(classification_report(y_test, y_pred, target_names=decodificador.classes_))


print("Mostrando matriz de confusión...")
matriz = confusion_matrix(y_test, y_pred)
sns.heatmap(matriz, annot=True, fmt='d', xticklabels=decodificador.classes_, yticklabels=decodificador
plt.xlabel('Predicción')
plt.ylabel('Real')
plt.title('Matriz de Confusión')
plt.show()
```

Ilustración 6Entrenar el modelo

Aquí es donde entrenamos el modelo utilizando un algoritmo de aprendizaje automático llamado Random Forest (Bosque Aleatorio). Random Forest es un modelo muy popular que utiliza múltiples árboles de decisión para hacer predicciones más precisas.

RandomForestClassifier(n_estimators=200): Este parámetro especifica cuántos árboles de decisión serán utilizados en el modelo. En este caso, se usan 200 árboles. Cuantos más árboles, mayor es la capacidad del modelo para realizar predicciones más precisas, pero también consume más recursos.

modelo.fit(X_entrena, y_entrena): Aquí se realiza el entrenamiento del modelo. X_entrena son las características (las URLs ya vectorizadas) y y_entrena son las etiquetas (los tipos de URLs, como "malware", "phishing", etc.). El modelo aprende a asociar las características con las etiquetas durante este proceso.

En este paso, el modelo que entrenamos se pone a prueba utilizando el conjunto de datos de prueba (X_test, y_test) que no vio durante el entrenamiento. Esto simula cómo el modelo se comportaría en un entorno real con datos nuevos.

y_pred = modelo. predict(X_test): Este código utiliza el modelo entrenado para hacer predicciones sobre el conjunto de datos de prueba (X_test). El modelo asignará una etiqueta (tipo de URL) a cada una de las URLs de prueba.

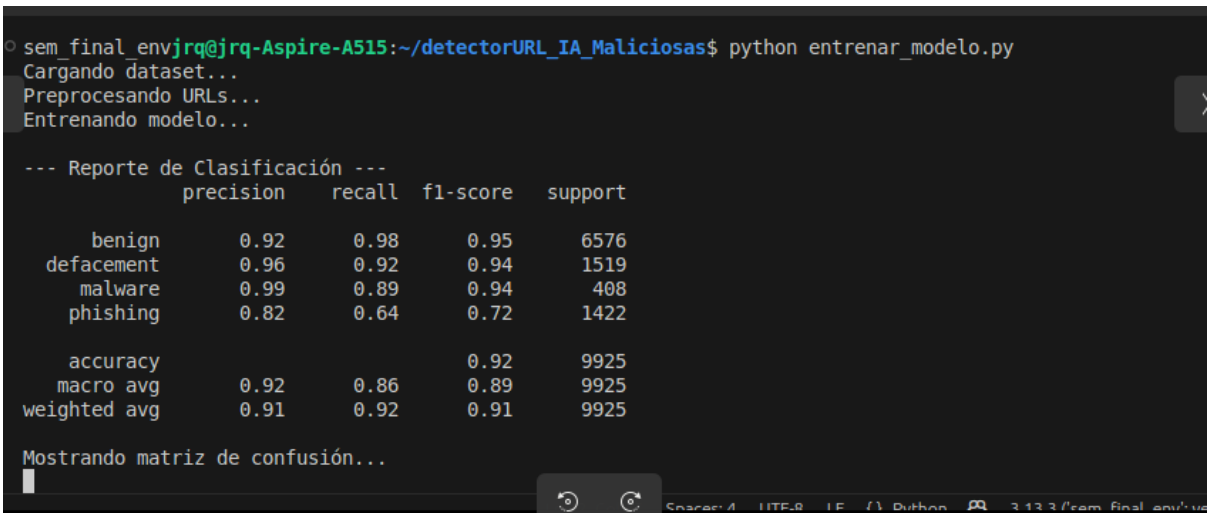
classification_report (y_test, y_pred, target_names=decodificador. classes_): Aquí se genera un informe de clasificación que evalúa el desempeño del modelo. Este reporte incluye varias métricas importantes, como:

Precisión (Accuracy): Qué tan bien el modelo clasifica las URLs en general.

Recall (Exhaustividad): Qué porcentaje de las URLs maliciosas se detectaron correctamente.

F1-Score: Una métrica que combina la precisión y el recall, proporcionando un balance entre ambos. Support: El número de muestras de cada clase

Ilustración 7Muestra



¿Por qué es importante esta evaluación?

Estas métricas nos permiten medir qué tan bien el modelo clasifica las URLs en cada categoría (benigna, malware, phishing, etc.). El F1-Score, por ejemplo, es útil cuando tenemos un conjunto de datos desbalanceado (por ejemplo, muchas más URLs benignas que URLs maliciosas). La combinación de precisión y recall ayuda a obtener una visión más completa del rendimiento del modelo.

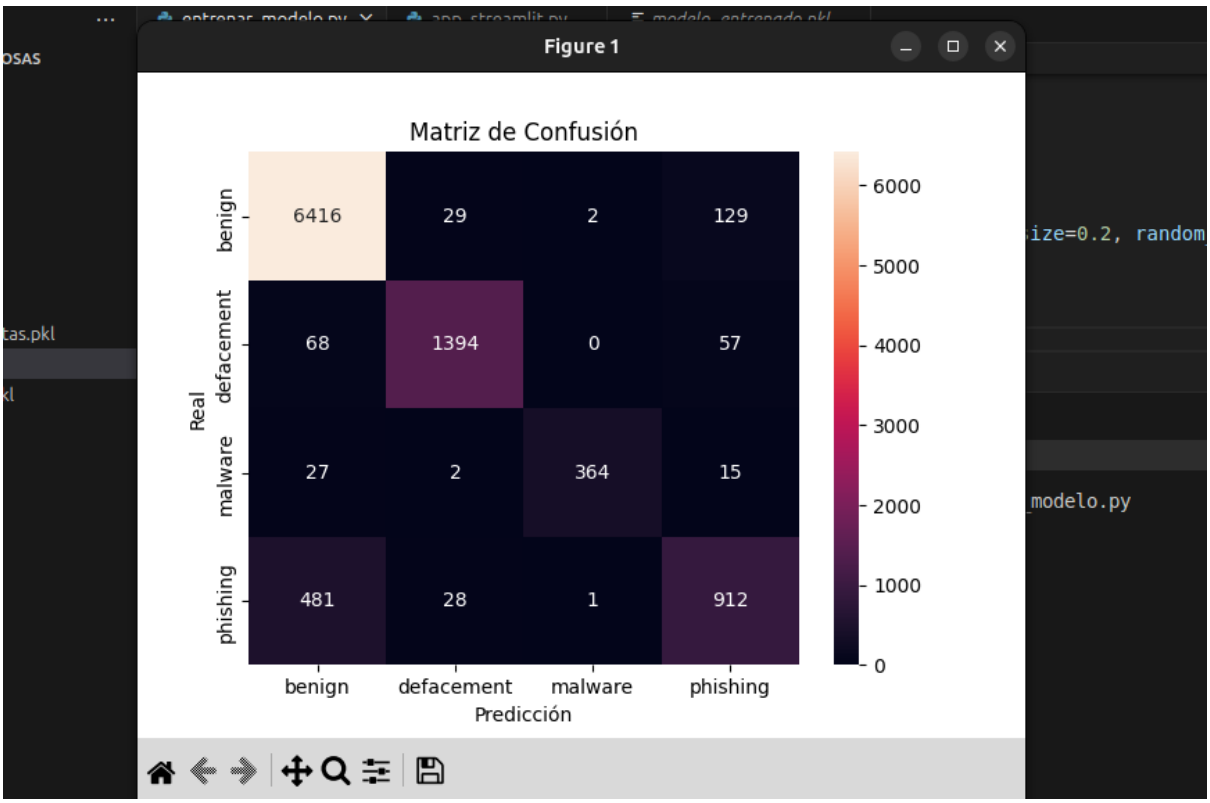


Ilustración 8Matriz de Confusión

La matriz de confusión es una tabla que permite visualizar el rendimiento del modelo al clasificar las URLs en las diferentes categorías posibles (por ejemplo, benign, phishing, malware, defacement.) en este caso la generamos para permitir ver no solo que tan preciso es el modelo, sino en qué tipo de errores incurre

Guardar el Modelo

Ilustración 9 Guardar el Modelo

```
58
59
60  joblib.dump(modelo, 'modelo_entrenado.pkl')
61  joblib.dump(vectorizador, 'vectorizador.pkl')
62  joblib.dump(decodificador, 'decodificador_etiquetas.pkl')
63  print("\nModelo y componentes guardados correctamente.")
64
```

Una vez que el modelo ha sido entrenado, evaluado y validado, esta sección se encarga de guardar los archivos esenciales para que el modelo pueda ser reutilizado.

La parte del Front está en el archivo app.py donde utilizamos la biblioteca Streamlit para crear una interfaz web sencilla e interactiva. El usuario ingresa una URL y, al hacer clic en “Analizar URL”, el sistema la procesa y la clasifica como benigna, phishing, malware o defacement usando el modelo de inteligencia artificial previamente entrenado.

Una vez obtenida la predicción, la aplicación también muestra una gráfica de barras que representa la probabilidad de pertenencia a cada clase. Esta gráfica se genera con Seaborn y Matplotlib, y ayuda a visualizar con qué nivel de confianza el modelo ha clasificado la URL. Por ejemplo, si la clase “phishing” tiene un 80%, es muy probable que la URL sea maliciosa, aunque también se muestran las otras clases con sus respectivos porcentajes.

Casos de Uso de la aplicación

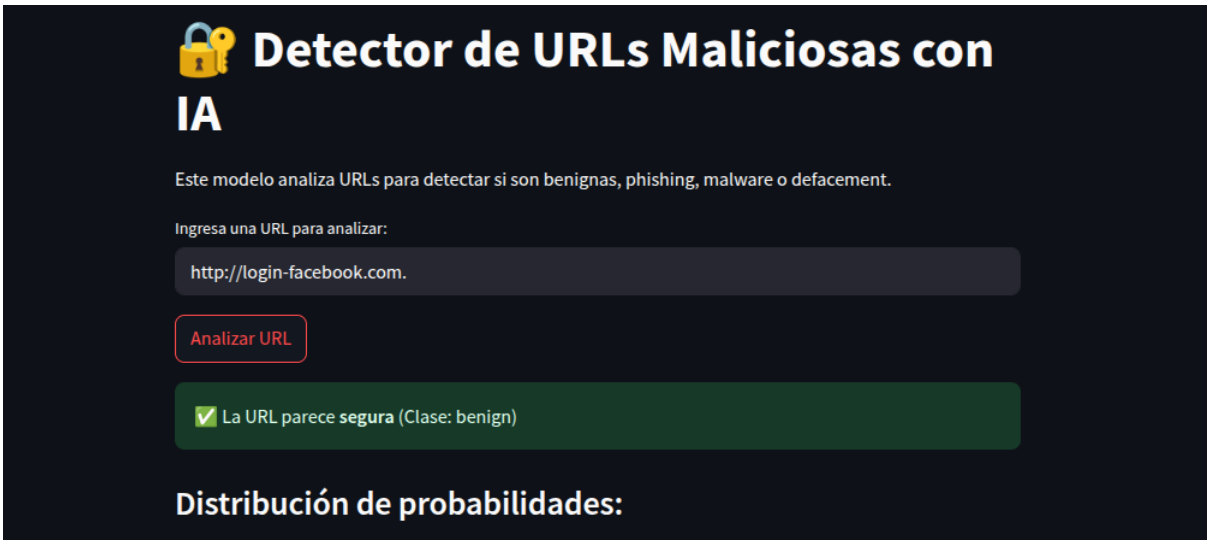


Ilustración 10 Pruebas URLs

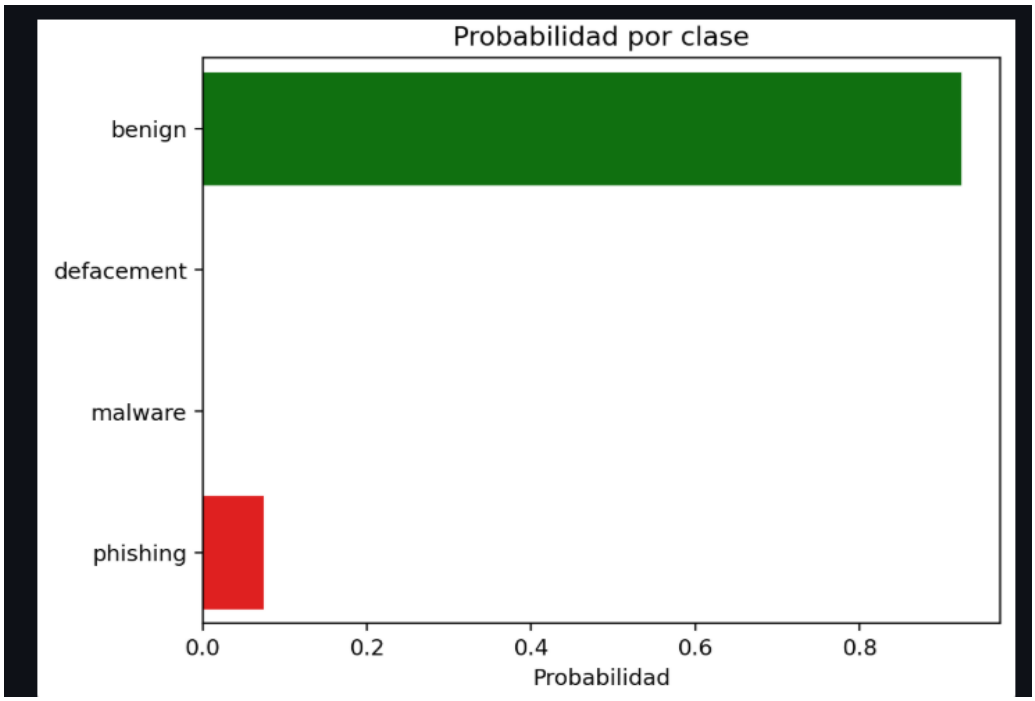


Ilustración 11Probabilidad de prueba

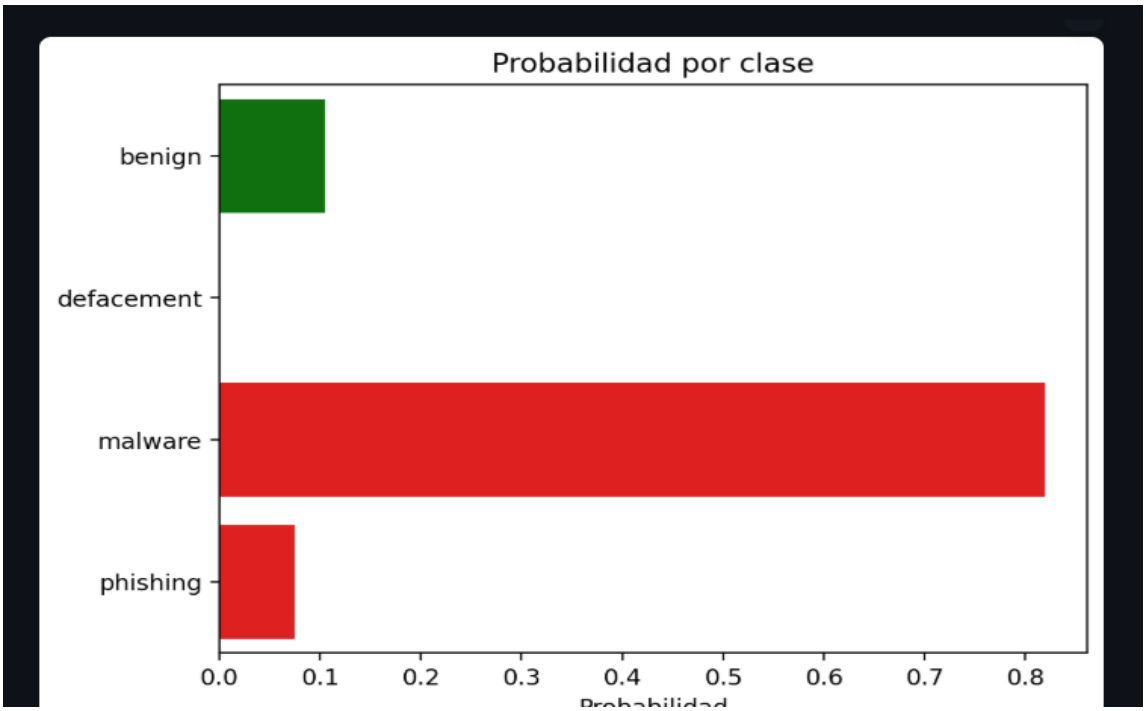


Ilustración 13Probabilidad de prueba

Este modelo analiza URLs para detectar si son benignas, phishing, malware o defacement.

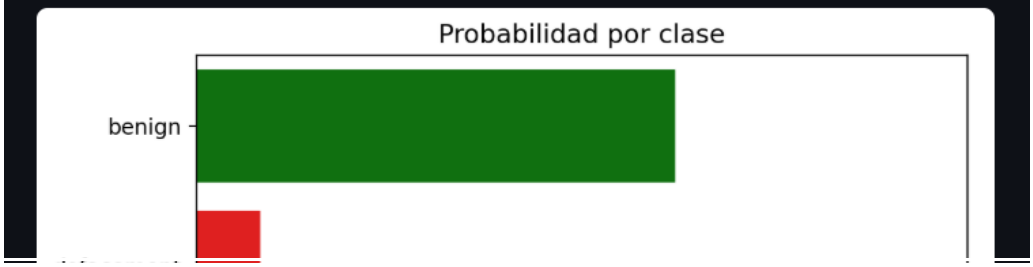
Ingresa una URL para analizar:

http://login-bankamericasbra-"77777///****##??¡bd--&%$=.es

Analizar URL

⚠️ Alerta: La URL es **potencialmente peligrosa** (Clase: phishing)

Distribución de probabilidades:



Conclusiones

El proyecto demostró que es posible construir un sistema de detección de amenazas web mediante Machine Learning con buenos resultados de precisión. Este enfoque puede integrarse fácilmente en firewalls, proxies o aplicaciones web para bloquear accesos peligrosos.

Además, este tipo de soluciones automatizadas representa una ventaja significativa frente a los métodos tradicionales de detección, ya que permiten identificar nuevas amenazas sin necesidad de reglas explícitas. Al estar entrenado con miles de ejemplos reales de URLs maliciosas, el modelo tiene la capacidad de aprender patrones ocultos que son difíciles de detectar manualmente, fortaleciendo así la defensa proactiva de los sistemas frente a ataques cibernéticos.

Bibliografías

Malicious URLs dataset

https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset?resource=download

Malicious URL Detection using Machine Learning: A Survey.

https://arxiv.org/abs/1701.07179

Artículo completo sobre detección de URLs maliciosas con técnicas de Machine Learning.

Verma, R., & Das, A. (2017).

https://doi.org/10.1109/IRI.2017.43

Explica la importancia de transformar y vectorizar URLs como parte del preprocesamiento.

Symantec Internet Security Threat Report (ISTR), Volume 24 (2019).

https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/istr-24

Estadísticas sobre prevalencia de phishing y malware por URLs maliciosas.

Chiew, K. L., Yong, K. S. C., & Tan, C. L. (2019).

https://doi.org/10.1016/j.eswa.2018.12.057

Tipología y vectores de ataques de phishing.

Google Safe Browsing Transparency Report.

https://transparencyreport.google.com/safe-browsing/overview

Informe sobre sitios web inseguros y detección en tiempo real.