DESDE 2023
https://revista.ccaitese.com

RICT Revista de Investigación Científica, Tecnológica e Innovación

Publicación Semestral RICT Vol. 3 No. 5 (2025) P. 32 – 39

ISSN: 2992-7315

# Algoritmo para la detección de URL's maliciosas y legítimas utilizando máquinas de vectores de soporte (SVM)
# Algorithm for detecting malicious and legitimate URLs using support vector machines (SVM)

Dylan Alejandro Fernández Molina[a,b], Arturo Hernández Martínez [a,b], Jimena Meléndez Ramírez [a,b], Juan Manuel Stein Carrillo[a,b]

[a] *Centro de Cooperación Academia-Industria, Tecnológico Nacional de México/ TES de Ecatepec, Ecatepec de Morelos, Estado de México, México.*

[b] *División de Ingeniería en Sistemas Computacionales, Tecnológico Nacional de México/ TES de Ecatepec, Ecatepec de Morelos, Estado de México, México.*

**Resumen**

Este estudio propone un enfoque basado en Máquinas de Vectores de Soporte (SVM) para la detección de URL's maliciosas, abordando un problema crítico en ciberseguridad. Se utilizaron repositorios públicos de URL's benignas y maliciosas para entrenar y evaluar el modelo. La metodología incluyó la selección y preprocesamiento de datos, el entrenamiento con diferentes configuraciones de SVM (kernels lineal, polinómico y RBF), y la evaluación del rendimiento mediante métricas como el F1 Score. Los resultados muestran que el modelo basado en SVM alcanzó una alta precisión en la clasificación de URL's, destacándose especialmente con el kernel RBF. Estos hallazgos confirman la efectividad de SVM en la detección de amenazas cibernéticas y sugieren su integración en sistemas de seguridad automatizados.

*Palabras clave:* Inteligencia Artificial, Ciberseguridad, Detección de Ataques, Máquinas de Vectores de Soporte, Phishing, Malware.

**Abstract**

This research examines the use of artificial intelligence (AI) in the field of cybersecurity, offering a comprehensive perspective on how advanced AI methods can enhance protection against cyber threats. The fundamentals of cybersecurity are reviewed, emphasizing their importance in safeguarding data and systems. The methodology of AI, specifically Support Vector Machines (SVM), is analyzed for its effectiveness in detecting attacks. Types of threats such as phishing and malware are also examined. To assess the practical effectiveness of these methods, experiments were conducted using public data repositories. The results indicate that SVM, among other techniques, can significantly improve the detection and response to cyber threats. This research highlights the critical role of AI in the evolution of cybersecurity and suggests future directions for research and application in this vital field.

*Keywords:* Artificial Intelligence, Cybersecurity, Attack Detection, Support Vector Machines, Phishing, Malware.

## 1. Introducción

La información que se presenta a continuación detalla las características contenidas en los repositorios almacenados en archivos CSV. Cada una de estas características se utiliza para analizar y clasificar los diferentes tipos de ataques cibernéticos. Esta clasificación es esencial para que los analistas de seguridad comprendan mejor el funcionamiento de estos ataques y puedan desarrollar estrategias basadas en inteligencia artificial (IA) para prevenir y mitigar su impacto. Los repositorios incluyen un amplio conjunto de datos que cubren múltiples tipos de amenazas cibernéticas, como URL's benignas, URL's de spam, URL's de phishing, URL's de malware y URL's de desfiguración. A través del análisis detallado de estas características, es posible identificar

*Autor para la correspondencia: 202021090@tese.edu.mx
**Correo electrónico**: 202021090@tese.edu.mx (Dylan Alejandro Fernández Molina), 202020600@tese.edu.mx (Arturo Hernández Martínez), 202020153@tese.edu.mx (Jimena Meléndez Ramírez), jmsteinc@tese.edu.mx (Juan Manuel Stein Carrillo)

**Historial del manuscrito:** recibido el 21/01/2025, última versión-revisada recibida el 25/2/2025, aceptado el 25/2/225, publicado el 8/4/2025. **DOI:** https://doi.org/10.5281/zenodo.15132851

patrones y comportamientos que distinguen a cada tipo de amenaza. Esta comprensión profunda permite a los profesionales de la ciberseguridad implementar medidas proactivas y reactivas más efectivas. Además, el uso de técnicas de IA y aprendizaje automático (machine learning) basadas en estos datos proporciona una herramienta poderosa para la detección anticipada y la reacción inmediata ante los ataques. Los modelos de IA pueden entrenarse con estos repositorios para reconocer señales sutiles de actividades maliciosas y adaptar continuamente sus estrategias a medida que evolucionan las amenazas.

En la era digital actual, la ciberseguridad ha emergido como un tema de vital importancia debido al aumento constante de amenazas en línea, entre ellas, las URL's maliciosas. Estas URL's pueden ser utilizadas para llevar a cabo diversas actividades maliciosas como el phishing, la distribución de malware, y otros ataques cibernéticos que pueden comprometer tanto la información personal como corporativa. Por lo tanto, es esencial desarrollar métodos eficaces para la detección y mitigación de estas amenazas. Este proyecto se centra en desarrollar un algoritmo para la identificación de URL's maliciosas utilizando técnicas de machine learning, con un enfoque particular en el algoritmo de Support Vector Machines (SVM). Este enfoque se basa en la capacidad de SVM para clasificar datos de alta dimensionalidad, permitiendo la distinción precisa entre URL's benignas y maliciosas.

Con este proyecto, pretendemos ofrecer una herramienta precisa y efectiva para identificar URL's maliciosas, contribuyendo así a fortalecer la ciberseguridad y mejorar la seguridad de los usuarios en el entorno digital.

## 2. Materiales y Método

### A. Python

*Fig. 1 – Logo Python* (Python (Oficial), s.f.)

Este programa, usando Python, está diseñado para detectar URL´s maliciosas utilizando métodos de machine learning, específicamente el algoritmo de Máquinas de Vectores de Soporte (SVM) (Python (Oficial), s.f.).

- **Descripción y Preparación del Conjunto de Datos:** Describe el conjunto de datos empleado, que abarca URL legítimas, de spam, phishing, malware y desfiguración.
  Los registros se limpian y preparan para su implementación en el modelo.

- **Entrenamiento del Modelo:** Entrena un modelo SVM utilizando diversas configuraciones de kernel (lineal, polinómico y gaussiano). Emplea técnicas de normalización de características para preparar los datos antes de ajustar el modelo.

- **Visualización y Evaluación:** Evalúa el rendimiento del modelo mediante métricas como el F1 Score.

- **Predicción:** Realiza predicciones con el modelo entrenado tanto en un conjunto reducido como en el conjunto completo. Imprime el F1 Score de las predicciones para medir la precisión del modelo.

Este programa construye, entrena y evalúa un modelo SVM para identificar URLs maliciosas, utilizando un enfoque de aprendizaje supervisado con diversas técnicas de procesamiento y visualización de datos.

### B. Pandas


*Fig. 2 – Logo Pandas* (Pandas, 2024)

Pandas es una librería en Python ampliamente reconocida para la gestión y análisis de datos estructurados. Algunas de las características más destacadas de Pandas incluyen (Pandas, 2024):

1. **Estructura de Datos:** Ofrece dos tipos principales de estructuras de datos: *Series* (para datos en una dimensión) y *DataFrames* (para datos en dos dimensiones, organizados en tablas).
2. **Lectura y Estructura de Datos:** Facilita la lectura y escritura de datos en múltiples formatos, incluyendo CSV, Excel, bases de datos SQL, entre otros.
3. **Manipulación de Datos:** Incluye una extensa variedad de funciones y métodos para manipular, filtrar, agrupar, ordenar y transformar datos.
4. **Análisis de Datos:** Permite realizar análisis estadísticos, crear gráficos y visualizar los datos.
5. **Manejo de Datos Faltantes:** Proporciona herramientas eficaces para detectar y gestionar valores faltantes en los datos.
6. **Rendimiento:** Está diseñada para ofrecer un rendimiento óptimo en el manejo de grandes volúmenes de datos.

### C. Scikit-learn


*Fig. 3 – Logo Scikit-learn* (Scikit-Learn, 2024)

Scikit-learn es una librería de machine learning de código abierto para Python. Algunas de las características clave de Scikit-learn son (Scikit-Learn, 2024):

1. **Técnicas de Machine Learning**

Scikit-learn ofrece una extensa selección de técnicas de machine learning, que incluyen clasificación, regresión, clustering, reducción de dimensiones, entre otras.

- Entre los algoritmos más utilizados se encuentran Máquinas de Soporte Vectorial (SVM), Árboles de Decisión, Bosques Aleatorios, Redes Neuronales y Análisis de Componentes Principales (PCA).

1. **Preparación de Datos**
   - Scikit-learn incluye herramientas para el preparación y modificación de datos, como escalado, codificación de variables categóricas, manejo de valores faltantes, etc.
   - Este proceso facilita la preparación de los datos para aplicarlos en modelos de aprendizaje automático.
2. **Evaluación de Modelos**
   - Scikit-learn ofrece funciones para dividir los datos en conjuntos de entrenamiento, validación y prueba, además de métricas de evaluación como precisión, exhaustividad, F1-score.
   - Esto ayuda a medir la efectividad de los modelos de aprendizaje automático.
3. **Usabilidad**
   - La biblioteca tiene una interfaz fácil de usar y consistente, lo que simplifica el proceso de implementación de técnicas de machine learning.
   - Además, Scikit-learn está bien documentada y con una amplia comunidad activa de usuarios y desarrolladores.

El propósito de las clases ***StandardScaler*** y ***RobustScaler*** en el código es realizar el escalado en los datos de entrada. Este escalado es fundamental en la etapa de preprocesamiento para modelos de machine learning, porque permite evitar que algunas características dominen el entrenamiento del modelo debido a diferentes escalas de magnitud. Específicamente:

- ***StandardScaler:*** Realiza un escalado de los datos al restar la media y dividir entre la desviación estándar correspondiente a cada característica. Esto resulta en una distribución con media 0 y varianza 1.

- ***RobustScaler:*** Realiza un escalado de los datos utilizando la mediana junto con el rango intercuartílico (IQR) en vez de utilizar la media y la desviación estándar. Esto lo hace más robusto a valores atípicos en comparación con el StandardScaler.

D. NumPy

*Fig. 4 – Logo NumPy*

NumPy (Numerical Python) es una librería de Python diseñada para manipular arrays de múltiples dimensiones y arreglos. Algunas de sus características clave son (NumPy, 2024):

- **Arrays Multidimensionales:** NumPy permite la creación y manipulación de arreglos de múltiples dimensiones, que son estructuras de datos que pueden almacenar datos de cualquier tipo.
- **Operaciones Matemáticas:** Ofrece diversas funciones matemáticas, como operaciones aritméticas, álgebra lineal y estadísticas, además de operaciones de transformación de datos.
- **Rendimiento:** Es optimizado para operaciones matemáticas, lo que lo hace más rápido y eficiente que las listas de Python.
- **Compatibilidad:** Integra fácilmente con bibliotecas de Python como SciPy, Pandas además de Matplotlib, lo que permite el análisis de datos y la visualización de resultados.
- **Funciones Especiales:** Incluye funciones útiles como números aleatorios y la búsqueda de elementos en arrays, y la manipulación de arrays.

NumPy es una biblioteca fundamental dentro del entorno Python para el análisis numérico y científico, proporcionando una manera rápida y eficiente de trabajar con datos numéricos.

Estos escaladores son comúnmente utilizados como parte de un pipeline de preprocesamiento de los datos previos a entrenar un modelo en machine learning, con el fin de asegurar que las características mantengan una escala uniforme y contribuyan de manera equitativa al entrenamiento del modelo.

E. Jupyter Notebook


*Fig. 5 – Logo Jupyter Notebook* (Jupyter Notebook, 2024)

Jupyter Notebook es una aplicación web de código abierto que permite crear y compartir documentos que contienen scripts ejecutables, gráficos y texto narrativo. Algunas de las principales funcionalidades de Jupyter Notebook son (Jupyter Notebook, 2024):

- Un entorno interactivo de programación que permite escribir, ejecutar y visualizar scripts en varios lenguajes, incluyendo Python, R, Julia, entre otros. El documento de Jupyter Notebook se divide en secciones, que pueden contener scripts, texto con formato Markdown, ecuaciones matemáticas, visualizaciones y otros elementos.

- **Ejecución de código y visualización de resultados:** Cada celda de código se puede ejecutar de forma independiente, lo que permite probar y depurar el código de manera iterativa.

- **Documentación integrada:** Jupyter Notebook permite combinar scripts, texto descriptivo, ecuaciones y visualizaciones en un único archivo, lo que simplifica la documentación y el reporte de análisis y proyectos.

- **Colaboración y compartición:** Los notebooks de Jupyter se pueden compartir fácilmente con otros usuarios, lo que promueve la cooperación y el trabajo en equipo.

- **Integración con múltiples lenguajes:** Jupyter Notebook es compatible con diversos lenguajes de programación, lo que lo convierte en una herramienta versátil para distintos tipos de proyectos y análisis. Jupyter Notebook es una plataforma poderosa que permite crear y compartir documentos interactivos que combinan scripts, visualizaciones y texto descriptivo, lo que la hace muy útil para el análisis de datos, la investigación científica y el desarrollo de aplicaciones.

F. Matplotlib


*Fig. 6 – Logo Matplotlib* (Matplotlib, 2024)

Matplotlib es una herramienta de Python muy utilizada para generar gráficos y visualizaciones de datos. Algunas de las características clave de Matplotlib son (Matplotlib, 2024):

- ***Creación de Gráficos:*** Permite producir una amplia gama de gráficos, como líneas, barras, dispersión, histogramas, y más.

- ***Personalización:*** Proporciona amplias opciones para ajustar los gráficos, como el estilo, los colores, las etiquetas, los títulos, etc.

- ***Integración:*** Se conecta sin problemas a otras bibliotecas de Python, como NumPy y Pandas, lo que simplifica la visualización de información estructurada.

- ***Interactividad:*** Permite crear gráficos interactivos que pueden ser manipulados y explorados por los usuarios.

- ***Múltiples Backends:*** Soporta diferentes backends de renderizado, lo que permite la visualización de gráficos en diferentes entornos, como Jupyter Notebook, aplicaciones de escritorio, entre otros.

Matplotlib es una herramienta eficaz y versátil para la creación de visualizaciones en Python, que ha llegado a ser una de las bibliotecas más reconocidas y empleadas en la comunidad de ciencia y análisis de datos.

**Repositorios de URLs Maliciosas y Legítimas**

En el contexto de nuestra investigación sobre inteligencia artificial aplicada a la ciberseguridad, hemos utilizado un conjunto de datos proporcionado por el ***Instituto Canadiense de Ciberseguridad de la Universidad de Nueva Brunswick*** (Brunswick, 2024). Estos repositorios contienen una extensa colección de URLs maliciosas y legítimas que permiten analizar diversas características léxicas y patrones asociados con distintos tipos de amenazas cibernéticas.

**Tipos de URLs en el Conjunto de Datos:**

- **URLs Benignas:** Se recopilaron más de 35,300 URLs benignas de sitios web clasificados por Alexa, utilizando herramientas de rastreo web para su extracción. Estas URLs fueron verificadas a través de Virus Total para garantizar su seguridad.

- **URLs de Spam:** Aproximadamente 12,000 URLs relacionadas con spam fueron obtenidas del repositorio público WEBSPAM-UK2007, un recurso ampliamente utilizado para estudios de spam en internet.

- **URLs de Phishing:** Se obtuvieron alrededor de 10,000 URLs de phishing desde OpenPhish, un repositorio que monitorea sitios de phishing activos. Estas URLs representan ataques donde los usuarios son engañados para revelar información confidencial.

- **URLs de Malware:** Más de 11,500 URLs de malware fueron recolectadas del proyecto DNS-BH, que mantiene una lista de sitios web asociados con malware.

- **URLs de Desfiguración (Defacement):** Se extrajeron más de 45,450 URLs de desfiguración, que corresponden a sitios web comprometidos, usualmente confiables según Alexa, pero que contienen páginas fraudulentas o maliciosas.

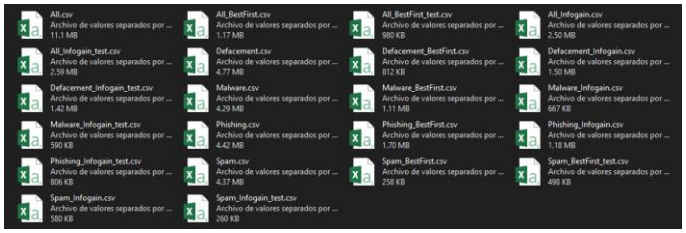


*Fig. 7 – Vista previa de todos los repositorios obtenidos por el Instituto Canadiense de Ciberseguridad de la Universidad de Nueva Brunswick* (Cybersecurity, 2024).

**Características de los Datos:**

Los datos almacenados en estos repositorios están estructurados en archivos CSV que contienen diversas características léxicas de las URLs, tales como:

- Longitud de la URL
- Cantidad de tokens en el dominio
- Longitud de la ruta y del nombre de archivo
- Frecuencia de caracteres específicos (vocales, números, símbolos)
- Relaciones entre la longitud del dominio, la ruta y los argumentos

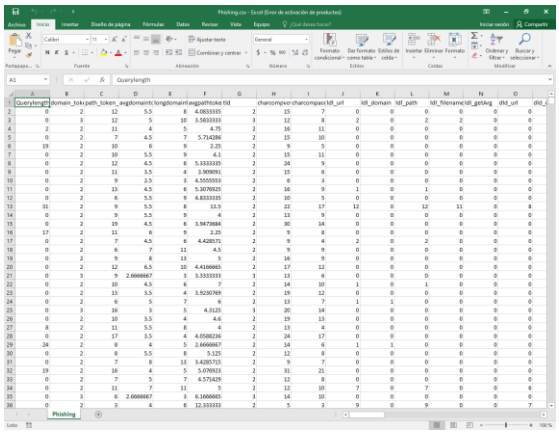

*Fig. 8 – Vista sobre los datos almacenados en los repositorios CSV, en este caso, en el repositorio Phishing.*

Estas características permiten a los investigadores identificar patrones comunes entre URLs maliciosas y legítimas, facilitando el entrenamiento de modelos de aprendizaje automático para la detección proactiva de amenazas. El análisis detallado de estas variables contribuye al desarrollo de metodologías más efectivas para la ciberseguridad.

**Algoritmo para la Detección de URLs Maliciosas Utilizando Support Vector Machines (SVM)**

En esta etapa de nuestro proyecto, hemos desarrollado un algoritmo utilizando ***Support Vector Machines (SVM)*** para detectar URLs maliciosas y benignas. A continuación, presentamos un resumen del análisis del código y su funcionamiento, detallando los puntos más relevantes para entender su estructura sin exponer el código completo.

**1. Preparación y Procesamiento de Datos**

El algoritmo comienza con la **lectura y preprocesamiento de los datos**. Los datos utilizados provienen de un conjunto de URLs benignas y maliciosas, incluyendo categorías como phishing, spam y malware. Las columnas principales del conjunto de datos incluyen características léxicas de las URLs, como la longitud del dominio, la ratio de URL en el dominio, y otras variables críticas para el análisis.

Los datos se dividen en conjuntos de **entrenamiento, validación y prueba**, utilizando un proceso estratificado para garantizar la correcta proporción entre clases. Antes de entrenar el modelo, se aplican técnicas de escalado de características utilizando *StandardScaler* y *RobustScaler* para evitar que las diferencias en las magnitudes de las variables influyan negativamente en el rendimiento del algoritmo (Scholar, 2024).

**2. Entrenamiento del Modelo SVM**

El modelo principal se basa en **Support Vector Machines (SVM)**, un algoritmo de aprendizaje automático ampliamente utilizado para tareas de clasificación en espacios de alta dimensionalidad.

El código implementa el SVM con diferentes configuraciones de **kernel**:

- **Kernel Lineal:** Se utiliza para problemas de clasificación linealmente separables. El kernel lineal define un hiperplano que divide las clases.

- **Kernel Polinómico y Gaussiano (RBF):** Para problemas no lineales, el modelo emplea transformaciones del espacio original de características mediante kernels polinómicos y de base radial. Esto permite identificar patrones más complejos en los datos de URLs.

El proceso de entrenamiento utiliza la función de pérdida **Hinge loss** y ajusta el parámetro de regularización **C**, que controla el equilibrio entre maximizar el margen y minimizar los errores de clasificación.

**3. Visualización y Evaluación del Modelo**

Durante el análisis, se representaron los **límites de decisión** del modelo entrenado mediante gráficos que muestran cómo el SVM separa las clases (URLs benignas y maliciosas). La fórmula general que define el límite de decisión en SVM lineal es:

$$\boldsymbol{\omega_0 x_0 + \omega_1 x_1 + b = 0}$$

Donde $\boldsymbol{\omega_0}$ y $\boldsymbol{\omega_1}$ son los coeficientes de los hiperplanos obtenidos del modelo, y $b$ es el término independiente.

Además, se implementaron métricas como el **F1 Score**, una métrica clave para evaluar el rendimiento de un modelo de clasificación en términos de **precisión y recall**. El F1 Score se calcula como:

$$F1 = 2 \cdot \frac{presicion * exhaustividad}{presicion + exhaustividad}$$

Donde:

- **Precisión:** Es la fracción de predicciones positivas correctamente clasificadas. Mide cuán preciso es un modelo.
- **Exhaustividad (Recall):** Es la proporción de casos positivos que un modelo identifica correctamente. Mide cuán completo es un modelo.

El valor del F1-score oscila entre 0 y 1, donde 1 representa el óptimo resultado alcanzable (cuando la precisión, así como la exhaustividad son ambas 1). En el código proporcionado, se utiliza la función ***f1_score()*** de Scikit-learn para obtener dicho valor del modelo de clasificación de URLs maliciosas. Esto permite evaluar cómo el modelo equilibra la precisión, así como la exhaustividad, lo cual resulta una métrica muy útil en escenarios de clasificación binaria como este (Scholar, 2024).

#### 4. Predicción

El modelo entrenado se utilizó para realizar predicciones sobre los conjuntos de datos de prueba. Las predicciones se compararon con las etiquetas reales, y se evaluaron mediante el F1 Score, que permite medir el equilibrio entre la precisión y la capacidad del modelo para identificar correctamente las URLs maliciosas.

Este algoritmo basado en **SVM** demostró ser eficaz para clasificar URLs maliciosas, proporcionando resultados precisos y una buena capacidad de generalización. La combinación de diferentes kernels y técnicas de preprocesamiento asegura que el modelo sea robusto y adaptable a diversos tipos de datos. Este enfoque es una herramienta valiosa para mejorar la seguridad cibernética, contribuyendo a la detección proactiva de amenazas en línea.

## 3. Resultados

#### 1. Límites de Decisión del SVM

Las gráficas muestran los límites de decisión generados por el modelo SVM para separar las URLs maliciosas de las benignas.

El SVM crea un **hiperplano** en el espacio de características, el cual se representa como una línea en las gráficas bidimensionales. Este hiperplano es el punto de división entre las clases de URLs, donde de un lado se encuentran las benignas y del otro las maliciosas.

- ✓ El hiperplano de decisión está definido por la ecuación:

$$\omega_0 x_0 + \omega_1 x_1 + b = 0$$

Donde $\omega_0$ y $\omega_1$ son los coeficientes de los hiperplanos obtenidos del modelo, y $b$ es el término independiente.

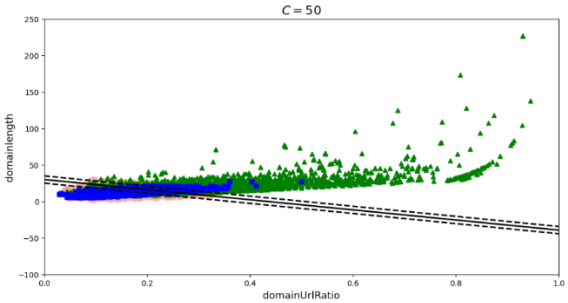


*Fig. 9 – Vista de los límites de decisión del SVM (Grafica 1).*

#### 2. Vectores de Soporte

En las gráficas se destacan los **vectores de soporte**, los cuales son los puntos de datos más cercanos al hiperplano y tienen una influencia directa en su ubicación. Estos puntos, representados como marcadores especiales, son esenciales para la clasificación, ya que determinan el margen máximo posible entre las dos clases de URLs. Los márgenes se representan como líneas paralelas al hiperplano, y la distancia entre estas líneas es el **margen** del clasificador.

- ✓ El margen se calcula como:

$$\frac{1}{\|\omega\|}$$

Donde $\|\omega\|$ es la norma del vector de pesos que define el hiperplano.

- **Puntos Verdes:** Los puntos verdes generalmente representan las URLs maliciosas (como phishing o malware). Estos puntos están situados en un lado del hiperplano, lo que indica que el modelo SVM ha clasificado estas URLs como amenazas. El objetivo del modelo es agrupar todas las URLs maliciosas en un lado del límite de decisión.

- **Puntos Azules:** Los puntos azules representan las URLs benignas. Estas URLs no representan una amenaza y deben ser clasificadas correctamente en el lado opuesto al de las URLs maliciosas. Los puntos azules separados del grupo verde indican que el modelo ha identificado correctamente que estas URLs no son peligrosas.

- **Puntos Morados:** Los puntos morados representan los vectores de soporte, que son los datos más cercanos al hiperplano de decisión. Estos puntos son críticos en la clasificación ya que definen el margen entre las clases. Los vectores de soporte son utilizados por el modelo SVM para establecer el hiperplano que maximiza la separación entre las URLs benignas y maliciosas.

- **Línea Negra:** La línea negra es el hiperplano de decisión. Este representa el límite que separa las dos clases de URLs (benignas y maliciosas). Todo punto que cae a un lado del hiperplano se clasifica como

malicioso o benigno, según la posición relativa a la línea. El hiperplano es el punto medio entre los vectores de soporte, y el objetivo del SVM es maximizar la distancia (margen) entre esta línea y los vectores de soporte de ambas clases.

### 3. Separación de Clases

La gráfica también incluye puntos que representan URLs maliciosas y benignas. Estos puntos se distribuyen a ambos lados del hiperplano de decisión.

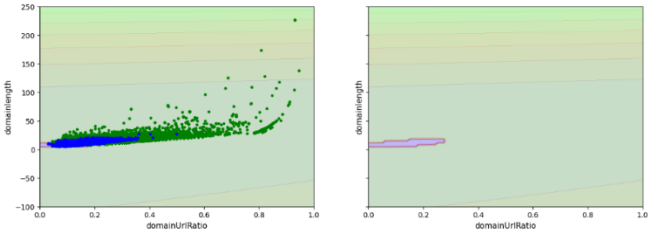


*Fig. 10 – Vista de la separación de clases (Grafica 2).*

Las URLs maliciosas se agrupan en un lado del hiperplano, mientras que las benignas se agrupan en el otro. Esta separación es crucial para evaluar la efectividad del modelo SVM en la detección de URLs maliciosas, ya que indica que el algoritmo ha aprendido a identificar patrones diferenciadores en las características extraídas.

- **Puntos Rojos o Rosados:** A menudo se utilizan puntos rojos o rosados para destacar los **vectores de soporte**. Estos puntos son los datos más cercanos al hiperplano de decisión y juegan un papel crucial en la definición del margen. Es posible que haya otro color como el rojo o rosado para diferenciarlos más claramente de las clases maliciosas y benignas.

- **Tonos de Gris:** En algunas representaciones gráficas, el área entre los márgenes (el espacio donde se encuentran los vectores de soporte) puede estar sombreada con un tono gris claro o intermedio. Esto ayuda a visualizar el margen que el SVM está maximizando entre las clases. El área más clara a ambos lados del hiperplano indica la separación de las clases con el margen máximo posible.

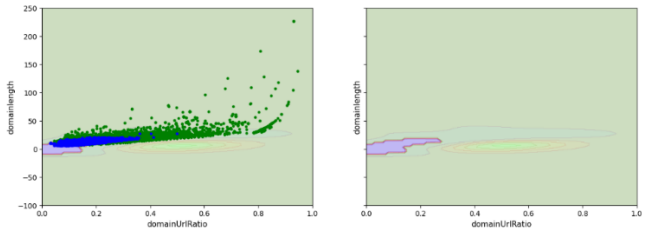


Fig. 11 – Vista de la separación de las clases (Grafica 3).

- **Sombras o Gradientes**: En algunos casos, se emplean sombras de colores o gradientes (amarillo, naranja, o tonos intermedios) para indicar regiones de mayor incertidumbre o baja confianza en la clasificación. Estas áreas se encuentran cerca del hiperplano de decisión, donde el modelo puede tener más dificultad para distinguir entre las dos clases.

- **Líneas Discontinuas:** Las líneas discontinuas suelen representar los márgenes superior e inferior del hiperplano. Estas líneas se ubican a una distancia equidistante del hiperplano y definen el margen entre las clases. Los vectores de soporte se encuentran en o cerca de estas líneas, y la distancia entre ellas refleja la capacidad del modelo para separar las clases.

### 4. Evaluación Visual del Modelo

Las gráficas proporcionan una manera visual de evaluar el rendimiento del modelo SVM. Si las URLs maliciosas y benignas están bien separadas por el hiperplano, el modelo tiene un buen rendimiento. En caso contrario, si se observan puntos mal clasificados en el lado incorrecto del hiperplano, esto indica áreas donde el modelo puede mejorar.

Asimismo, es capaz de generar las predicciones del modelo SVM con kernel RBF (***Radial Basis Function***) empleando el conjunto de validación limitado. Lo cual es un paso crucial para medir la efectividad del modelo antes de aplicarlo al conjunto final de test.

## 4. Discusión

El algoritmo propuesto para la detección de URLs maliciosas basado en Support Vector Machines (SVM) ha demostrado ser una herramienta eficiente en la clasificación de amenazas cibernéticas. Su capacidad para manejar datos de alta dimensionalidad le permite identificar con precisión patrones léxicos y características asociadas a URLs maliciosas como phishing, spam y malware. Los resultados obtenidos durante esta investigación son consistentes con hallazgos previos en la literatura, como lo señalado en el artículo de investigación de **ENISA**, "***Artificial Intelligence and Cybersecurity Research***" ((ENISA), 2024), donde se destaca la efectividad de las SVM en la detección de malware e intrusiones cibernéticas.

Sin embargo, es importante señalar que, si bien el uso de SVM ha mostrado una alta efectividad, existen oportunidades para mejorar la precisión del modelo. Investigaciones adicionales sugieren que la combinación de SVM con otros métodos de inteligencia artificial, como árboles de decisión, podría aumentar significativamente la precisión del modelo al analizar URLs. Este enfoque combinado permitiría una detección más robusta de amenazas cibernéticas al aprovechar las fortalezas de cada técnica.

En nuestro estudio, nos limitamos al uso exclusivo de SVM y no pusimos a prueba estas combinaciones. Adicionalmente, el alcance de nuestra investigación estuvo restringido a la evaluación de repositorios de URLs previamente categorizadas y obtenidas de fuentes académicas confiables como el Instituto Canadiense de Ciberseguridad. No

realizamos pruebas en entornos más dinámicos, como la evaluación de URLs en tiempo real provenientes directamente de la web, lo cual podría ofrecer una visión más realista y útil de la efectividad del algoritmo en un entorno de producción.

## 5. Conclusiones

El uso de Support Vector Machines (SVM) en la detección de URLs maliciosas ha demostrado ser un enfoque prometedor en el campo de la ciberseguridad, proporcionando resultados precisos en la clasificación de URLs benignas y maliciosas. A través de la implementación de este modelo, se ha logrado identificar de manera eficaz diversas amenazas cibernéticas, contribuyendo a la protección de los usuarios en el entorno digital.

No obstante, aunque los resultados obtenidos son alentadores, existen oportunidades para continuar mejorando la eficiencia del modelo, particularmente al combinarlo con otras técnicas de inteligencia artificial como los árboles de decisión. Además, futuros trabajos podrían centrarse en la implementación del algoritmo en entornos en tiempo real, lo que permitiría una evaluación más precisa de su rendimiento en situaciones dinámicas.

Esta investigación señala la importancia del uso de la inteligencia artificial en la evolución de las técnicas de ciberseguridad y abre la puerta a nuevos enfoques que optimicen la detección y mitigación de amenazas en el ciberespacio.

## 6. Agradecimientos

Deseo expresar mi más sincero agradecimiento al Centro de Cooperación Academia Industria (CCAI) del Tecnológico de Estudios Superiores de Ecatepec, por brindarnos la invaluable oportunidad de formar parte del equipo de ciberseguridad. Gracias a esta experiencia, hemos podido aplicar de manera práctica los conocimientos adquiridos durante nuestra formación en la carrera de Ingeniería en Sistemas Computacionales, contribuyendo no solo al desarrollo de este proyecto, sino también a nuestra evolución profesional.

Agradezco profundamente al Dr. Adolfo Meléndez Ramírez, nuestro coordinador y mentor, por su generosa guía, conocimientos y sabiduría. Su liderazgo ha sido clave para el éxito de este trabajo, y gracias a su apoyo constante, hemos podido alcanzar estos resultados. Además, nos llevamos valiosas lecciones tanto en el ámbito profesional como personal que, sin duda, marcarán nuestra trayectoria futura.

Extiendo un especial agradecimiento a mi familia, en particular a mis padres, cuyo apoyo incondicional ha sido fundamental en cada etapa de mi formación. Sin su respaldo, no habría sido posible llegar hasta este punto en mi desarrollo tanto personal como profesional, y estoy profundamente agradecido por todo lo que me han brindado.

Finalmente, quiero expresar mi gratitud a mis compañeros y amigos por compartir conmigo esta parte tan significativa de mi vida académica. De cada uno de ellos he aprendido algo valioso que me acompañará a lo largo de mi carrera, y les agradezco por su compañía, su amistad y las experiencias que hemos compartido.

## 7. Referencias

(ENISA), T. E. (17 de Abril de 2024). *Artificial Intelligence and Cybersecurity Research*. Obtenido de https://www.enisa.europa.eu/publications/artificial-intelligence-and-cybersecurity-research

Brunswick, U. d. (1 de Julio de 2024). *URL dataset (ISCX-URL2016)*. Obtenido de https://www.unb.ca/cic/datasets/url-2016.html

Cybersecurity, C. I. (24 de Junio de 2024). *URL dataset (ISCX-URL2016)*. Obtenido de http://205.174.165.80/CICDataset/ISCX-URL-2016/Dataset/

*Jupyter Notebook*. (12 de Junio de 2024). Obtenido de https://jupyter.org/

*Matplotlib*. (18 de Junio de 2024). Obtenido de https://matplotlib.org/

*NumPy*. (10 de Junio de 2024). Obtenido de https://numpy.org/

*Pandas*. (30 de Mayo de 2024). Obtenido de https://pandas.pydata.org/

*Python (Oficial)*. (s.f.). Recuperado el 21 de Mayo de 2024, de https://www.python.org/

Scholar, S. (1 de Julio de 2024). *Detecting Malicious URLs Using Lexical Analysis*. Obtenido de https://www.semanticscholar.org/paper/Detecting-Malicious-URLs-Using-Lexical-Analysis-Mamun-Rathore/01bb00b24fb2bcf1d11748d0c39ba60367b4c264

*Scikit-Learn*. (5 de Junio de 2024). Obtenido de https://scikit-learn.org/stable/